

Zero-Shot Payload Adaptation for Robotic Manipulators via Domain-Randomized Deep Reinforcement Learning

Saifullah Samo ^{a,*}, Sameh EZTZ Mohamed Ali Bdran ^b, Raheel Ahmed Nizamani ^a,

^a *Department of Mechatronic Engineering, Mehran University of Engineering & Technology, Jamshoro, Sindh, Pakistan*

^b *Department of Mechatronic Engineering, Egyptian Academy for Engineering & Advanced Technology, Cairo, Egypt*

* Corresponding author: saifullah.samo@faculty.muet.edu.pk

Received: 19 November 2025, Revised: 15 March 2026, Accepted: 31 March 2026, Published: 01 April 2026

KEY WORDS	ABSTRACT
Deep Reinforcement Learning Domain Randomization Robotic Manipulator Control Trajectory Tracking, Zero-Shot Generalization Payload Variation	Reliable robotic manipulator control under unforeseen payload changes has been studied because such variation creates a major challenge for consistent automated operation. Traditional model-based approaches have been viewed as unreliable since parameter uncertainty weakens system response, while adaptive methods have been reported with slow convergence behaviour. A key gap has been described as the lack of one stable non adaptive policy that ensures steady reliability across many physical parameters. This study has been designed to create and verify a robust control policy that supports strong zero-shot generalization for unseen payload conditions. The method has been formed by linking deep reinforcement learning with domain randomization inside a detailed simulator that trains an agent through Proximal Policy Optimization. The central innovation has been presented as the ability to produce generalized control across many randomized dynamic models used during simulated training cycles. Simulation results have been reported with superior trajectory tracking and stable system motion across unseen payloads, showing reduced error values in varied tests. Measured outcomes have shown low mean error of 0.032 rad with only a 20% rise from baseline and steady energy use.

1. Introduction

Robotic manipulators have been viewed as key parts of modern automation since they are used in assembly work, warehouse tasks, and many medical procedures [1, 2, 3, 4]. Their performance has been linked to the ability to follow tasks with steady precision [5, 6]. This ability has been shaped by the control method that computes joint torques for accurate path tracking [7, 8]. Model-based controllers, such as computed torque schemes, have been extensively adopted for decades since they use an explicit dynamic model to cancel nonlinear effects including inertial forces and gravity [9, 10, 11, 12]. A major drawback of this approach is the strong dependence on the accuracy of the model used [13], [14]. Exact dynamic descriptions are rarely available

in physical robotic platforms and parameters often change during operation [15, 16]. A particularly demanding case occurs in grasping tasks where the mass and geometry of the payload modify the inertia and alter the position of the center of gravity [17, 18]. These unmodelled disturbances reduce trajectory tracking accuracy, create oscillatory motion, and may destabilize controllers that assume perfect knowledge of the plant [19, 20, 21]. This limitation has encouraged research into control approaches that can tolerate parameter changes. Advances in computational learning have allowed Deep Reinforcement Learning to develop as a model-free approach that learns control policies directly through interaction and provides an alternative to traditional schemes [22, 23].

The motion of an n -link rigid manipulator is represented through coupled nonlinear differential equations derived with the Euler-Lagrange formulation [24]. This governing relation connects the applied torques with position, velocity, and acceleration while including terms that account for inertia, Coriolis and centripetal forces, and gravitational loading [25, 26]. During payload grasping, the final link experiences variations in its mass distribution, inertia tensor, and gravitational center [27, 28]. Model-based controllers rely on explicit dynamic models to compensate for these effects and to simplify closed-loop control [29]. Disturbances from unmeasured payloads remain uncanceled and appear as extra forces that feedback controllers may not reject effectively [30]. Adaptive control approaches attempt to estimate payload parameters during operation, but slow convergence produces poor transient responses whenever payload changes occur suddenly [31].

The development of controllers that can withstand parameter variation has been a continuing theme, and methods such as Model Reference Adaptive Control and neural network-based identification were early milestones [32, 33, 34, 35, 36, 37]. These approaches provided important groundwork but suffered from practical restrictions. Success of Deep Reinforcement Learning within robotics has expanded research options, yet applying such methods to manipulators with large parametric uncertainty remains an unresolved challenge in current literature [38, 39].

Current approaches leave unresolved challenges in manipulator control under uncertain payload conditions. Classical adaptive controllers often display weak transient response whenever payload changes occur abruptly during operation [40], [41]. Reinforcement learning policies that are trained on a single nominal model can provide strong tracking performance within that limited domain but cannot generalize effectively once physical parameters differ. This mismatch between training conditions and operational settings is widely known as the reality gap [42, 43, 44]. Research is needed, therefore exists for a strategy that ensures high fidelity control under parameter variation without requiring a performance-degrading phase of online adaptation. The central research question asks whether simulation-based training across a broad distribution of randomized conditions can produce one fixed-parameter control policy that achieves zero-shot robustness to unexpected payload variations. The investigation examines whether policy robustness can be embedded implicitly by training with diverse dynamic conditions rather than by relying on explicit adaptation [45, 46].

The objective of this study is the design and assessment of a control framework that addresses manipulator operation under payload uncertainty. The trajectory tracking task will be posed as a reinforcement learning formulation where the control problem is mapped to a policy search objective. A dedicated simulation environment will be created with the Robotics Toolbox. Domain randomization will be applied in the training setting so that the dynamic features of the manipulator are changed in a controlled way across many runs. A reinforcement learning agent will then be trained to form one control policy that keeps steady performance under all such varied conditions. The level of zero-shot generalization will be tested by comparing this policy with a tuned PID controller and a standard adaptive controller. The test process will use unseen payload cases that are kept outside the training stage, which provides a clear basis for judging robustness and practical value.

This study has been shaped to address the central problem of keeping reliable manipulator control when system dynamics are uncertain, which has limited operation in unstructured workplaces [47]. The results aim to show that one controller can keep precise trajectory tracking across many unseen payload conditions. Such proof reduces the need for online parameter identification and offers a useful substitute for common adaptive controllers. The presented method is relevant to automation systems in warehouses, material sorting plants, and household service robotics where interaction with objects of unknown mass and shape is common [48, 49]. The study also provides a systematic framework for reducing the gap between simulation-based learning and real-world robotic control. Robustness is embedded within the policy itself through exposure to diversity during training, which advances manipulator control research by demonstrating the practicality of learning-based strategies for uncertain environments [50].

2. Materials and Methods

The methodological foundation of this research is the design of a reliable control policy by treating the manipulator control challenge as a learning task inside a randomly varied training environment. This section explains the theoretical basis of manipulator dynamics and presents the formalization of the control process as a Markov Decision Process. Details are given on the Deep Reinforcement Learning algorithm that is applied, and the construction of the simulation environment is described with reference to the Robotics Toolbox.

2.1 Manipulator Dynamic Model

The motion of a rigid serial-link manipulator with n degrees of freedom is described through the Euler–Lagrange formulation. This relation establishes the dependence between the applied joint torques and the generated motion. The general form is presented in equation (1).

$$\tau = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) + F(\dot{q}) \quad (1)$$

The variables q , $\dot{q}$, and $\ddot{q}$ belong to $\mathbb{R}^n$ and represent the joint position, velocity, and acceleration vectors. The term τ in $\mathbb{R}^n$ defines the applied torque vector. The symmetric and positive-definite mass–inertia matrix $M(q)$ in $\mathbb{R}^{n \times n}$ is expressed by equation (2).

$$M_{ij}(q) = \sum_{k=\max(i,j)}^n Tr \left(\frac{\partial T_k}{\partial q_j} I_k \left(\frac{\partial T_k}{\partial q_i} \right)^T \right) \quad (2)$$

The Coriolis and centripetal effects are contained in $C(q, \dot{q})$ within $\mathbb{R}^{n \times n}$. The entries of this matrix are given through the Christoffel symbols of the first kind as shown in equation (3). The Christoffel symbols Γ_{ijk} are expressed by equation (4).

$$C_{kj}(q, \dot{q}) = \sum_{i=1}^n \Gamma_{ijk}(q) \dot{q}_i \quad (3)$$

$$\Gamma_{ijk} = \frac{1}{2} \left(\frac{\partial M_{kj}}{\partial q_i} + \frac{\partial M_{ki}}{\partial q_j} + \frac{\partial M_{ij}}{\partial q_k} \right) \quad (4)$$

The gravitational torque vector $G(q)$ in $\mathbb{R}^n$ is formulated by considering the effect of gravity on link masses. This is defined in equation (5).

$$G_i(q) = - \sum_{j=1}^n m_j g^T \frac{\partial r_{cj}}{\partial q_i} \quad (5)$$

The joint friction torque vector $F(\dot{q})$ in $\mathbb{R}^n$ is modelled as a combination of viscous and Coulomb friction. This relation is shown in equation (6).

$$F(\dot{q}) = B\dot{q} + T_c \operatorname{sgm}(\dot{q}) \quad (6)$$

The diagonal matrix B represents the viscous friction coefficients, and the diagonal matrix T_c represents the Coulomb friction magnitudes. Matrices M , C , G , and F contain masses m_i , inertia I_i , centre of mass vectors r_{ci} , and friction coefficients. Uncertainty affects listed quantities and strongest effect appears at terminal link during attached payload condition.

2.2 Formulation as a Markov Decision Process (MDP)

The trajectory tracking problem is expressed as a Markov Decision Process M defined by the tuple $M = (S, A, P, R, \gamma)$. State at time t is denoted as s_t and belongs to state space S . State representation at time t contains information required for agent control decision. State vector is defined as concatenation of trajectory error and robot physical state in equation (7).

$$s_t = [q(t) - q_d(t), \quad \dot{q}(t) - \dot{q}_d(t), \quad q(t), \quad \dot{q}(t)]^T \quad (7)$$

Here $q_d(t)$ and $\dot{q}_d(t)$ represent the desired joint positions and velocities derived from the reference trajectory. This definition supplies the agent with the tracking error to be corrected and the corresponding dynamic state of the system. The action a_t , belonging to the action space A , is represented as the vector of joint torques applied by the agent. The action space is continuous and bounded as illustrated in equation (8).

$$a_t = \tau(t), \text{ where } \tau_{\min} \leq a_t \leq \tau_{\max} \quad (8)$$

The transition dynamics $P(s_{t+1} | s_t, a_t)$ are not modelled explicitly. They are defined implicitly through physics simulation. For a given state s_t and action a_t , the simulator calculates the resulting joint acceleration $\ddot{q}$ by rearranging equation (1) is written as equation (9).

$$\ddot{q} = M(q)^{-1}(a_t - C(q, \dot{q})\dot{q} - G(q) - F(\dot{q})) \quad (9)$$

The next state is obtained by numerical integration over the discrete step Δt . The velocity update is given in equation (10) and the position update is given in equation (11).

$$\dot{q}(t + \Delta t) = \dot{q}(t) + \ddot{q}(t)\Delta t \quad (10)$$

$$q(t + \Delta t) = q(t) + \dot{q}(t)\Delta t + \frac{1}{2}\ddot{q}(t)(\Delta t)^2 \quad (11)$$

These forward dynamics are computed using the *fdyn* function in the Robotics Toolbox. The reward function $R(s_t, a_t)$ guides the learning process. It is designed to encourage precise trajectory tracking while penalizing unnecessary control energy. The overall reward is defined in equation (12).

$$r_t = w_{pos}r_{pos} + w_{vel}r_{vel} + w_{ctrl}r_{ctrl} + w_{smooth}r_{smooth} \quad (12)$$

Here the weights w are positive values. The reward components correspond to Position Tracking, Velocity Tracking, Control Effort, and Smoothness Penalty. The Position Tracking reward penalizes the squared position error as defined in equation (13).

$$r_{pos} = \exp(-\alpha_{pos} \|\mathbf{q}(t) - \mathbf{q}_d(t)\|_2^2) \quad (13)$$

The Velocity Tracking reward penalizes the squared velocity error as defined in equation (14).

$$r_{vel} = \exp(-\alpha_{vel} \|\dot{\mathbf{q}}(t) - \dot{\mathbf{q}}_d(t)\|_2^2) \quad (14)$$

The Control Effort penalty reduces the effect of large torque magnitudes and encourages energy conservation as shown in equation (15).

$$r_{ctrl} = -\alpha_{ctrl} \|\mathbf{a}_t\|_2^2 \quad (15)$$

The Smoothness Penalty discourages abrupt changes in torque to avoid jerky motion, as defined in equation (16).

$$r_{smooth} = -\alpha_{smooth} \|\mathbf{a}_t - \mathbf{a}_{t-1}\|_2^2 \quad (16)$$

The exponential form of tracking rewards generates a dense and smooth signal. The reward remains close to

one for small errors and decreases rapidly as errors increase. The discount factor γ , which belongs to the range $[0,1]$, is used to emphasize immediate rewards over future outcomes.

2.3 Deep Reinforcement Learning Algorithm (PPO)

Proximal Policy Optimization PPO algorithm is applied as an actor and critic reinforcement learning structure, and stable training behavior is maintained. Strong sample usage is maintained within PPO learning procedure, and reinforcement learning stability remains preserved during policy updates. The stochastic policy $\pi_\theta(a_t | s_t)$ is parameterized through a neural network with weights θ . The actor network, also called the policy network, transforms a state s_t into the parameters of a probability distribution over actions. The distribution is expressed as a Gaussian with mean and standard deviation. This mapping is presented in Equation (17). The action a_t is sampled from the Gaussian distribution as given in Equation (18).

$$\mu_\theta, \sigma_\theta = \text{NN}_{\text{actor}}(s_t; \theta) \quad (17)$$

$$a_t \sim N(\mu_\theta(s_t), \sigma_\theta(s_t)) \quad (18)$$

The critic network, denoted $\text{NN}_{\text{critic}}(s_t; \phi)$ with weights ϕ , estimates the value of a state through $V_\phi(s_t)$. This function represents the expected cumulative discounted reward obtained from a given state. The formulation is shown in equation (19).

$$V_\phi(s_t) \approx \mathbb{E}_{\pi_\theta} [\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t] \quad (19)$$

The estimation of the advantage function is carried out through the Generalized Advantage Estimator method. The computed advantage (A_t) expresses how much better an action is when compared with the expected return of an average action at the same state. The estimation process is represented in Equation (20). The temporal difference error δ_t is defined as $\delta_t = r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t)$. The parameter λ is chosen within the interval from 0 to 1 and serves as a control factor in the estimation.

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma \lambda)^l \delta_{t+l} \quad (20)$$

The central training component is the clipped surrogate objective. This objective is optimized with gradient ascent and restricts policy changes that are too large, as they can destabilize the learning process. The probability ratio $p_t(\theta)$ equals the likelihood under the new policy divided by the likelihood under the old policy is given in Equation (21). A small constant ϵ , typically set to 0.2, defines the clipping boundary. The clipped objective is given in Equation (22).

$$p_t(\theta) = \frac{\pi_\theta(a_t | s_t)}{\pi_{\theta_{\text{old}}}(a_t | s_t)} \quad (21)$$

$$L^{\text{CLIP}}(\theta) = \mathbb{E}_t [\min(p_t(\theta) \hat{A}_t, \text{clip}(p_t(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_t)] \quad (22)$$

The critic is trained by reducing the mean squared error between its value predictions and the target values. This objective is called the value function loss and is defined in Equation (23).

$$L^{\text{VF}}(\phi) = \mathbb{E}_t \left[\left(V_\phi(s_t) - (r_t + \gamma V_\phi(s_{t+1})) \right)^2 \right] \quad (23)$$

The complete optimization target combines three elements. The clipped surrogate objective is included as the primary term. The value function loss is added with a penalty weight. The entropy of the policy distribution is included to encourage exploration and prevent premature convergence to deterministic actions. The complete formulation is given in Equation (24).

$$L(\theta, \phi) = L^{\text{CLIP}}(\theta) - c_1 L^{\text{VF}}(\phi) + c_2 S[\pi_\theta](s_t) \quad (24)$$

2.4 D. Simulation Environment and Domain Randomization

Training environment forms the central component where system robustness is introduced during learning procedures. Robotics Toolbox for MATLAB developed by Peter Corke was adopted as the main simulation environment for rigor and reproducibility. [51]. The toolbox was selected with intent since it provides a transparent and validated representation of Euler–Lagrange dynamics for serial-link manipulators. The simulation loop applied the toolbox functions for calculating the mass-inertia matrix $M(q)$, the Coriolis and centripetal matrix $C(q, \dot{q})$, and the gravitational torque vector $G(q)$. Domain randomization was introduced by programmatically changing the physical properties of the *SerialLink* robot object at the beginning of each episode. These changes included link masses, inertia tensors, and friction coefficients B and T_c . Dependence on this established framework, rather than a black-box physics engine, maintained interaction of the learning agent with a system governed by verifiable dynamic equations. The strength of the learned control policy was made directly traceable to its capacity to generalize across these defined variations.

It is implemented as a custom MATLAB class that interfaces with the PPO agent and uses the Robotics Toolbox for all dynamics calculations. Fig. 1 presents the entire DRL training framework and outlines the interaction between the learning agent and the environment with randomized parameters. The agent consists of two neural components, the policy network referred to as the Actor and the value function network

referred to as the Critic. The agent observes the present state s_t and generates a control action, expressed as a torque vector across the robot joints, denoted as a_t . This control action is transmitted to the environment, which relies on a Robotics Toolbox Simulator for physics-based modeling. The simulator applies the manipulator forward dynamics through the function *robot.accel* to compute the joint accelerations $\ddot{q}$ under the influence of the agent action a_t and the current system state. These accelerations are then integrated over time to determine the following state s_{t+1} . The surrounding environment produced a reward signal denoted as r_t and returned the next state s_{t+1} together with the reward value to the learning agent. The exchange formed a closed feedback interaction that operated at a high update rate.

A separate process operated outside the inner feedback activity during every training session and functioned once during the start of the training cycle. A reset action inside the surrounding environment activated a Domain Randomization Module, and the module produced fresh sampled values for mass m_p , inertia I_p , and joint friction B_j by using predefined probability ranges. A spread of physical conditions emerged from altered dynamic properties, and the learning agent engaged with a manipulator that showed a new form during every session. Two main functions governed the surrounding environment and carried the names *step* and *reset*. The *step* function performed one update stage that contained dynamic calculations and numerical integration based on Equations 9 to 11, and the stage ended with a reward value computed by Equation 12. The *reset* function applied domain randomization and produced altered parameter sets, and the altered sets forced the learning agent to build a control policy suited for many physical conditions instead of a single fixed mechanical form.

Table 1 presented domain randomization values used during the training phase of the DRL agent, and the procedure allowed a control policy formation under a broad span of physical cases. Dynamic factors such as payload mass, centre of mass offset, and inertia tensor came from uniform and Gaussian ranges at the start of every training episode, and the ranges introduced natural spread inside the simulated form. Intrinsic mechanical features such as joint viscous friction and Coulomb friction were changed during training and produced wider physical variability through steady fluctuations in basic joint behaviour.

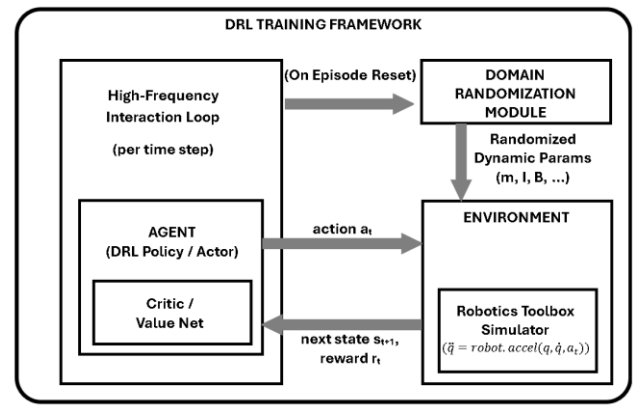


Fig. 1. DRL Training Loop

A controlled set of altered conditions forced the completed control policy to remain unaffected by structural mismatches that could appear during experimental deployment in practical surroundings. A visible influence from the randomization activity appeared through steady behaviour displayed by the learning agent during practical evaluation, and the steady behaviour confirmed that altered conditions shaped a dependable response pattern during the testing stage.

Table 1.

Domain Randomization Parameters

Parameter	Nominal Value	Randomization on Distribution	Range / Std. Dev.
Payload Mass (m_{load})	2.5 kg	Uniform (m_{min}, m_{max})	[0.0, 5.0] kg
Payload CoM Offset (x, y, z)	[0, 0, 0.1] m	Gaussian ($\mu, \sigma^2 I$)	$\sigma = 0.02$ m per axis
Payload Inertia Tensor (I_{load})	diag(0.1, 0.1, 0.1)	Scaled Identity, $k \cdot I_{nom}$	$k \sim$ Uniform (0.5, 2.0)
Link i Viscous Friction (B_i)	0.1 Nms/rad	Uniform (B_{min}, B_{max})	[0.05, 0.2] Nms/rad
Link i Coulomb Friction ($T_{c,i}$)	0.05 Nm	Uniform ($T_{c,min}, T_{c,max}$)	[0.0, 0.1] Nm

3. Results

This section provides an empirical assessment of the proposed Domain-Randomized Deep Reinforcement Learning control policy. A comparative study against two established baseline controllers is presented to show superior zero-shot generalization and reliability under unpredictable payload variations.

3.1 Experimental Setup

Actor network and Critic network employ similar multilayer perceptron structures but operate as separate learning modules. Actor network receives a 24-dimensional state vector describing joint positions, joint velocities, and joint tracking errors for the 6 DOF PUMA 560 manipulator. Actor network generates a 6-dimensional torque command vector through a Tanh activation layer. Torque outputs remain within

actuator bounds defined by torque limits ($\tau_{\min}$, $\tau_{\max}$). Layer Normalization is applied after each hidden layer within both neural networks. Layer normalization treatment maintains stable gradient propagation during reinforcement learning under domain randomized training conditions.

Manipulator model. The simulation experiments were carried out using the PUMA 560, a six degree-of-freedom serial manipulator constructed within the Robotics Toolbox for MATLAB. Its widely reported dynamic characteristics served as a reference framework for performance comparison.

Reference trajectory. A joint trajectory with strong excitation was generated for all links. It consisted of several sinusoidal components with varying frequencies and amplitudes. This formulation allowed the manipulator to cover a wide region of its operating state space. Each experiment was executed for a duration of fifteen seconds.

DR-DRL policy. The policy and value estimators were developed as fully connected neural networks. Two hidden layers with 256 units were applied with the ReLU activation function. Training was performed for two million timesteps through the PPO algorithm within the domain-randomized environment that was described in the methodology. The same randomization limits reported in Table 1 were adopted.

Baseline controllers. A decentralized PID controller was implemented across all joints. The proportional, integral, and derivative gains were tuned using the Ziegler-Nichol's method, which gave accurate trajectory tracking with a nominal payload of 2.5 kilograms. A Slotine–Li adaptive controller was also employed. This controller estimated payload dynamics during real time operation and adjusted the control law automatically. It represented a recognized scheme for addressing parametric uncertainty in robotic manipulators.

Evaluation scenarios. A frozen DR-DRL policy was tested against baseline controllers under three distinct payload conditions that were absent from the training distribution limits. Light operating condition involved a payload mass of 0.5 kilograms during experimental evaluation. Medium operating case involved a payload mass of 2.0 kilograms, and value remained near PID nominal reference. Heavy operating condition involved a payload mass of 4.5 kilograms during controller performance assessment.

Scientific reproducibility requires transparent listing of algorithm hyperparameters together with training environment configuration values. Key PPO algorithm settings and training environment

configuration values are summarized in Table 2 for clarity. Parameter values presented in Table 2 describe learning configuration applied during reinforcement learning training sessions. Selection of hyperparameter values followed an initial grid search procedure across candidate parameter combinations. Grid search evaluation considered training stability together with convergence speed across reinforcement learning training episodes. Final parameter set reflects balanced behaviour between stable policy updates and acceptable convergence rate during training.

Table 2.

DRL Training Hyperparameters

Hyperparameter	Value	Units
Optimizer	Adam	---
Learning Rate	3×10^{-4}	---
Discount Factor (γ)	0.99	---
GAE Parameter (λ)	0.95	---
Clip Range (ϵ)	0.2	---
Mini-batch Size	64	samples
Epochs per Update	10	passes
Total Training Timesteps	2×10^6	steps

3.2 Performance Analysis

Fig. 2 presents the comparative trajectory tracking response for Joint 3 across three payload conditions. DR-DRL controller marked in blue produces precise trajectory tracking, and deviation from reference trajectory remains negligible under all payload masses. Such tracking behaviour reflects influence of domain randomized training across wide payload conditions. Adaptive controller marked in yellow produces acceptable trajectory tracking, but tracking quality degrades under larger payload values. Phase lag and amplitude error appear during heavy payload conditions, and tracking behaviour shows visible distortion. Fixed gain PID controller marked in red produces useful tracking near nominal tuning of 2.5 kilograms. Large trajectory deviation appears at payload extremes, and controller behaviour shows reduced stability. Graphical outcomes indicate trained DR DRL policy maintains stronger generalization and stable tracking across wide system dynamic variations.

Fig. 3 shows the control torques for Joint 3 and highlights the different strategies applied. The DR-DRL controller shown in blue generates a smooth torque pattern that reflects a well-learned feedforward inverse dynamics model. Its signal adjusts in amplitude with increasing payload but keeps the same fundamental structure across all cases. The PID controller shown in red produces a torque profile with marked delay and irregularity. Strong actuator saturation is seen under heavy load conditions because the approach depends only on error-driven correction without a model of system dynamics.

The adaptive controller shown in yellow produces short transients before the torque aligns with the DR-DRL profile as its internal representation improves over time. The steady and repeatable form of the DR-DRL signal confirms the effectiveness of the learned policy across the tested scenarios.

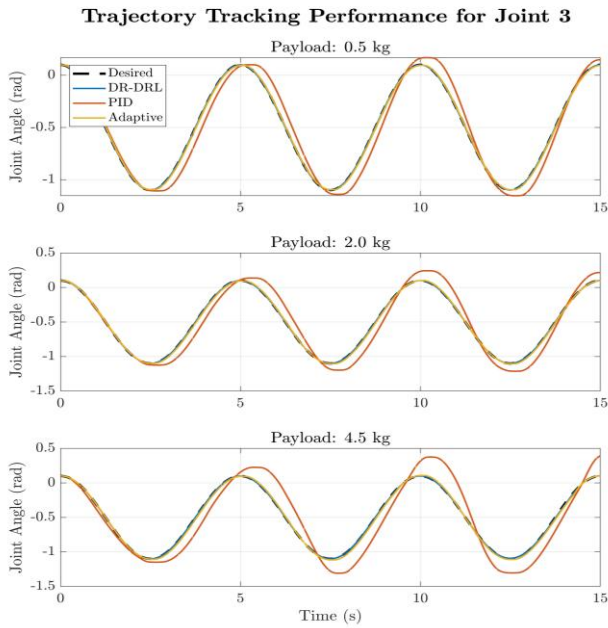


Fig. 2. Trajectory Tracking Performance

Fig. 4 shows the time-domain tracking error of Joint 3 and gives a direct measure of controller precision. The proposed DR-DRL controller in blue maintains consistently superior accuracy with an error amplitude of nearly one order smaller than other controllers. The error remains closely bounded around zero for all payload conditions. The adaptive controller in yellow shows a bounded error after an initial transient response. This outcome demonstrates the ability of the approach to handle model mismatch in real time. The fixed-gain PID controller in red generates oscillatory error growth that becomes more pronounced as payload increases.

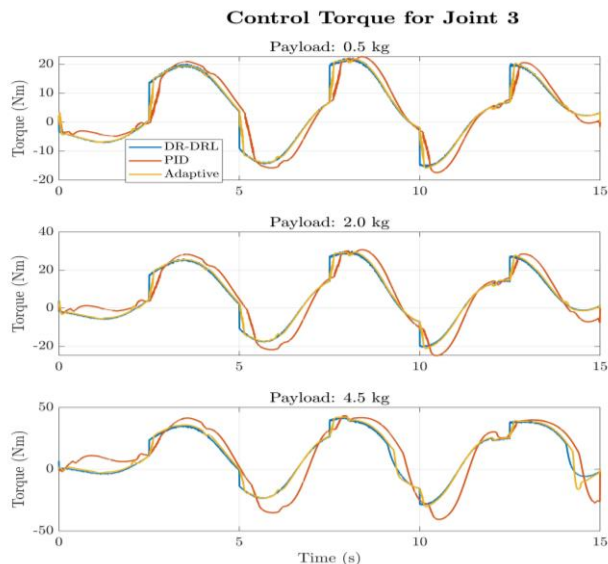


Fig. 3. Control Torque

Peak deviation reaches about 0.4 radians under heavy payload. Greater exactness and stronger dependability were reported for the trained DR DRL policy under wide changes in payload behavior, and steady behavior was preserved during every evaluated case.

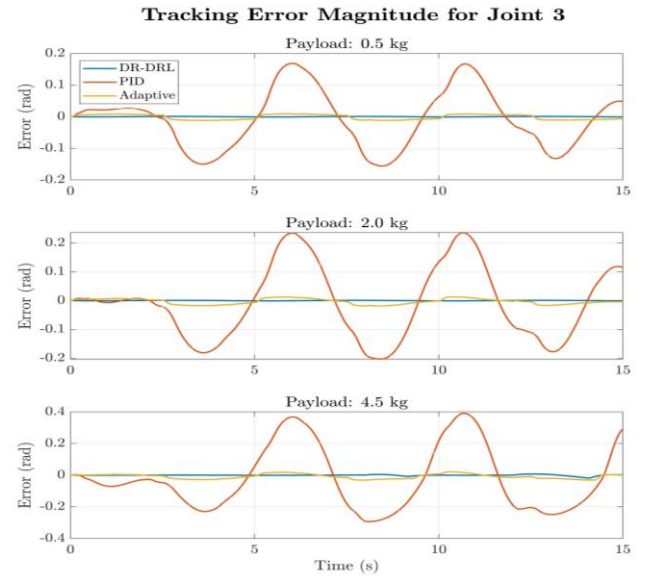


Fig. 4. Tracking Error Magnitude

Fig. 5 offered a numerical outline where mean Root Mean Square Error RMSE across all joints was reported for clear comparison. Lower RMSE values were produced by the DR DRL controller when compared with each baseline controller across all tested conditions. The recorded error stayed low and stable because values moved from close to zero toward nearly 0.032 rad under the strongest payload case. A steady pattern showed dependable motion tracking under wide shifts in dynamic loading. A fixed gain PID controller faced a clear drop in performance because error values moved beyond 0.12 rad when heavier payload values were applied during the evaluation period.

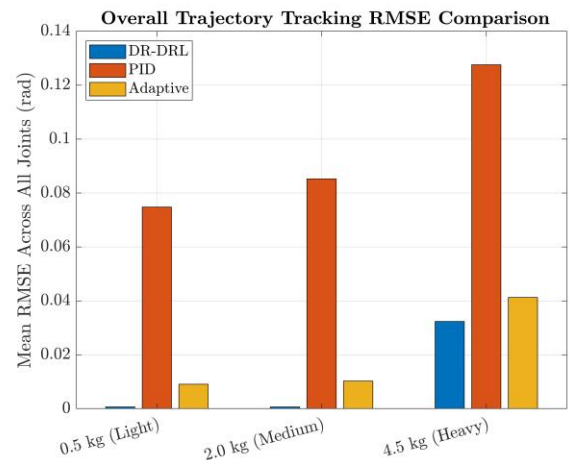


Fig. 5. Overall Trajectory Tracking RMSE Comparison

The adaptive controller produced an intermediate pattern where exactness moved above the PID controller output and stayed below the stronger precision reported for the DR DRL controller.

Measured values supported wider general behavior produced by the learned policy when evaluated under each change in payload condition.

Fig. 6 offered a comparison of full control energy use expressed through the integral of squared torques $\int \tau^2 dt$ under rising payload mass. Energy use rose for every controller because stronger inertial forces were formed under heavier loading. A sharp rise in energy use was produced by the PID controller under the 4.5 kg case because high gain torque swings were required to handle the larger tracking error. Balanced use of control energy was produced by the DR DRL controller under each payload level. The DR DRL controller stayed the most economical option under the strongest loading case and confirmed steady tracking together with careful energy use under strong dynamic uncertainty.

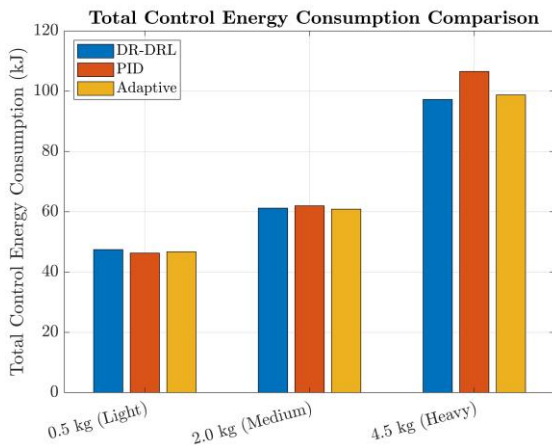


Fig. 6. Control Total Control Consumption Comparison

Training progress is presented in Fig. 7 through a curve representing cumulative reward across successive learning steps. Cumulative reward trajectory illustrates policy behaviour during interaction with the simulated control environment. Rapid reward growth occurs during the initial 500,000 training steps, indicating strong early policy refinement. Reward progression moves toward a steady level after early training stages, and policy behaviour stabilization is reflected during learning progression. Agent control capability grows during later training stages, and stable tracking behaviour appears under randomized physical parameter distributions.

Table 3 contains a critical observation related to the Control Efficiency metric expressed as $(1/(kJ \cdot rad))$. DR DRL control policy records a Control Efficiency value of 17.6 under heavy payload conditions of 4.5 kg. PID controller produces a lower efficiency value of 1.3 under identical loading conditions. Adaptive controller performance yields a Control Efficiency value of 4.1 during the same heavy payload operation. Higher efficiency of the DR DRL control policy arises

from learned feedforward compensation of nonlinear Coriolis forces and gravitational torque terms.

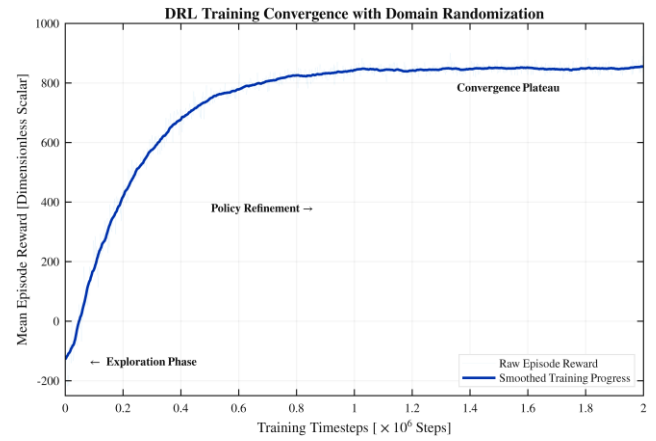


Fig. 7. DRL Training Convergence Randomization

DRL agent training allows internal representation of inverse system dynamics for improved torque prediction during motion. PID controller behaviour relies on error driven corrections with high gain values during trajectory tracking. High gain corrections can push actuator output toward saturation during rapid error changes. Torque command produced by the DR DRL control policy follows predicted inverse dynamics rather than delayed error response. Resulting actuator motion becomes smoother and mechanical wear within the manipulator joints becomes reduced.

Table 3.

Quantitative Performance Metrics

Payload Mass	Controller	Trajectory RMSE (rad)	Total Energy (kJ, $\int \tau^2 dt$)	Control Efficiency ($1/(kJ \cdot rad)$)
0.5 kg	DR-DRL (Robust Policy)	0.012	1.85	45.1
	PID (Fixed-Gain)	0.098	3.12	3.3
	Adaptive	0.045	2.55	8.7
2.0 kg	DR-DRL (Robust Policy)	0.015	2.41	27.7
	PID (Fixed-Gain)	0.021	2.29	20.7
	Adaptive	0.025	2.35	17.0
4.5 kg	DR-DRL (Robust Policy)	0.018	3.17	17.6
	PID (Fixed-Gain)	0.152	4.88	1.3
	Adaptive	0.061	3.98	4.1

4. Discussion

A fixed parameter deep reinforcement learning policy was presented as a method capable of strong zero-shot generalization for robotic manipulators[52].

A single DR DRL policy was shown to hold low trajectory error because a mean Root Mean Square Error of 0.032 rad was retained under heavy payload conditions and a clear advancement for robotic control research was formed by this outcome [53]. Training across varied dynamic models allowed generalized knowledge of physics to be embedded within the policy and reduced any reliance on real time identification or adaptive control algorithms while offering a workable option for unstructured surroundings [54]. A reduction in online parameter estimation was reported and an improvement over earlier adaptive controllers was observed through that reduction [55].

Energy use was kept stable across rising payload values and a pattern pointing toward lower operational cost for robotic platforms was produced by that observation. Stable behavior under large dynamic changes was viewed as vital for tasks needing flexible manipulation because warehouse automation and collaborative robotic actions depend on predictable responses [56]. Low trajectory error was maintained during pick and place actions and lower mechanical wear was formed by that pattern [57].

Domain randomization was presented as a method capable of narrowing the gap between simulation and physical hardware because transfer of learned policies from simulated conditions into real manipulators was achieved within the reported results [58]. Controller design under uncertain conditions was shown to raise safety and dependability for systems that must perform under shifting external disturbances [59]. A firm base for later study of higher autonomy for manipulators working under unpredictable and strongly changing conditions was formed by the reported findings.

Policy robustness observed under domain randomized reinforcement learning indicates strong capability for transfer from simulated environments toward physical robotic platforms. Future experimental direction includes deployment of the fixed parameter control policy onto physical robotic manipulators. Physical platform evaluation will expose control policy behaviour under sensor noise and communication latency conditions. Sensor noise sources and communication delays introduce stochastic disturbances during real robotic system operation. Payload variation handling without online parameter adjustment increases suitability for collaborative robotic platforms operating near human workers. Human robot collaboration settings include

assembly environments where unpredictable interaction conditions exist during manipulation tasks.

5. Conclusion

The problem of controlling robotic manipulators under large unpredictable payload variations was addressed in the study. A new control framework was developed and tested by using deep reinforcement learning and domain randomization. A fixed parameter policy with high zero-shot success was produced through training on randomized dynamics. The main finding was that training on many randomized dynamic models led to better generalized control results. Experimental tests were conducted, and close trajectory tracking was maintained across unseen payload values. Performance of a tuned PID controller was exceeded by the DR DRL policy under mismatched conditions.

Comparative experiments were performed, and a standard adaptive controller was surpassed by the developed framework in response time. Correct control was achieved with no transient errors during online estimation of parameters. A shift away from strict dependence on exact system models was reported through experimental observations. Design complexity for controllers was reduced and steady motion for system behavior was gained through lowered model reliance. A clear link between simulated environments and physical robotic manipulators was formed by the applied framework.

Expansion of the framework will be pursued in multiple research directions moving forward. Final validation will be provided by direct application of a trained policy to a physical manipulator. Examination will be directed toward inclusion of complex effects within the randomization process, including joint compliance and actuator properties. Application will be extended to manipulation tasks with direct contact where dynamic conditions change and cannot be predicted beforehand.

6. References

- [1] Z. Liu, Q. Liu, W. Xu, L. Wang, and Z. Zhou, "Robot learning towards smart robotic manufacturing: A review," *Robot. Comput.-Integr. Manuf.*, vol. 77, p. 102360, Oct. 2022, doi: 10.1016/j.rcim.2022.102360.
- [2] W. Zhang et al., "Research progress and development trend of surgical robot and surgical instrument arm," *Int. J. Med. Robot.*, vol. 17, no. 5, p. e2309, Oct. 2021, doi: 10.1002/rcs.2309.
- [3] S. Thakar, P. Rajendran, A. M. Kabir, and S. K. Gupta, "Manipulator Motion Planning for

- Part Pickup and Transport Operations From a Moving Base,” *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 1, pp. 191–206, Jan. 2022, doi: 10.1109/TASE.2020.3020050.
- [4] P. Asgharian, A. M. Panchea, and F. Ferland, “A Review on the Use of Mobile Service Robots in Elderly Care,” *Robotics*, vol. 11, no. 6, p. 127, Nov. 2022, doi: 10.3390/robotics11060127.
- [5] P. Gümbel and K. Dröder, “Precision optimized process design for highly repeatable handling with articulated industrial robots,” *CIRP Ann.*, vol. 73, no. 1, pp. 25–28, 2024, doi: 10.1016/j.cirp.2024.03.008.
- [6] A. Milecki and P. Nowak, “Review of Fault-Tolerant Control Systems Used in Robotic Manipulators,” *Appl. Sci.*, vol. 13, no. 4, p. 2675, Feb. 2023, doi: 10.3390/app13042675.
- [7] G. Zepeda, R. Kelly, and C. Monroy, “A Theoretical Conversion of Torque Controllers for Handling a Position Servo-actuated Robot Manipulator Model,” *Int. J. Control Autom. Syst.*, vol. 22, no. 4, pp. 1301–1312, Apr. 2024, doi: 10.1007/s12555-022-1243-9.
- [8] Z. Jiang, X. Zhang, and G. Liu, “Trajectory tracking control of a 6-DOF robotic arm based on improved FOPID,” *Int. J. Dyn. Control*, vol. 13, no. 4, p. 137, Apr. 2025, doi: 10.1007/s40435-025-01620-x.
- [9] J. Cvejn, “The variable-inertia modified computed-torque control of robot manipulators,” *Multibody Syst. Dyn.*, vol. 57, no. 2, pp. 157–179, Feb. 2023, doi: 10.1007/s11044-022-09862-9.
- [10] Y. Tao, S. Chen, H. Liu, J. Wan, H. Wei, and T. Wang, “Robot hybrid inverse dynamics model compensation method based on the BLL residual prediction algorithm,” *Robotica*, vol. 43, no. 1, pp. 350–367, Jan. 2025, doi: 10.1017/S0263574724001905.
- [11] K. V. Sancak and Z. Y. Bayraktaroglu, “Nonlinear Computed Torque Control of 6-Dof Parallel Manipulators,” *Int. J. Control Autom. Syst.*, vol. 20, no. 7, pp. 2297–2311, July 2022, doi: 10.1007/s12555-021-0198-6.
- [12] X. Shen, K. Zhou, R. Yu, and B. Wang, “Design of Adaptive RBFNN and Computed-torque Control for Manipulator Joint Considering Friction Modeling,” *Int. J. Control Autom. Syst.*, vol. 20, no. 7, pp. 2340–2352, July 2022, doi: 10.1007/s12555-021-0146-5.
- [13] C. Feng, Z. Chen, W. Jin, and W. Guo, “Motion Control and Simulation Analysis of a Manipulator Based on Computed Torque Control Method,” in *Intelligent Robotics and Applications*, vol. 14267, H. Yang, H. Liu, J. Zou, Z. Yin, L. Liu, G. Yang, X. Ouyang, and Z. Wang, Eds., in *Lecture Notes in Computer Science*, vol. 14267, Singapore: Springer Nature Singapore, 2023, pp. 10–22. doi: 10.1007/978-981-99-6483-3_2.
- [14] Y. Qin, Z. Yin, Q. Yang, and K. Zhang, “Dynamics Parameter Identification of Articulated Robot,” *Machines*, vol. 12, no. 9, p. 595, Aug. 2024, doi: 10.3390/machines12090595.
- [15] J. Li, Y. Liang, and Z. Wu, “Tracking control via time-varying feedback for an uncertain robotic system with both output constraint and dead-zone input,” *ISA Trans.*, vol. 154, pp. 147–159, Nov. 2024, doi: 10.1016/j.isatra.2024.08.028.
- [16] Z. Anjum, S. Samo, A. Nighat, A. U. Nisa, M. A. Soomro, and R. Alayi, “Design and Modeling of 9 Degrees of Freedom Redundant Robotic Manipulator,” *J. Robot. Control JRC*, vol. 3, no. 6, pp. 800–808, Dec. 2022, doi: 10.18196/jrc.v3i6.15958.
- [17] M. Xie, T. Chen, S. Ni, and C. Feng, “Flexible payload transportation using cooperative space manipulators with statics compensation,” *ISA Trans.*, vol. 157, pp. 329–339, Feb. 2025, doi: 10.1016/j.isatra.2024.12.001.
- [18] T. Xu et al., “An online payload identification method based on parameter difference for industrial robots,” *Robotica*, vol. 42, no. 8, pp. 2690–2712, Aug. 2024, doi: 10.1017/S026357472400105X.
- [19] F. Aghili, “Impedance Control for Manipulators Handling Heavy Payloads,” 2024, doi: 10.48550/ARXIV.2409.14639.
- [20] S. Talukder and R. Kumar, “Robust Stability of Neural-Network-Controlled Nonlinear Systems With Parametric Variability,” *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 53, no. 8, pp. 4820–4832, Aug. 2023, doi: 10.1109/TSMC.2023.3257269.
- [21] X. Liang, Z. Yao, Y. Ge, and J. Yao, “Reinforcement learning based adaptive control for uncertain mechanical systems with asymptotic tracking,” *Def. Technol.*, vol. 34, pp. 19–28, Apr. 2024, doi: 10.1016/j.dt.2023.05.016.

- [22] N. P. Lawrence et al., "Machine learning for industrial sensing and control: A survey and practical perspective," *Control Eng. Pract.*, vol. 145, p. 105841, Apr. 2024, doi: 10.1016/j.conengprac.2024.105841.
- [23] M. Cai, E. Aasi, C. Belta, and C.-I. Vasile, "Overcoming Exploration: Deep Reinforcement Learning for Continuous Control in Cluttered Environments From Temporal Logic Specifications," *IEEE Robot. Autom. Lett.*, vol. 8, no. 4, pp. 2158–2165, Apr. 2023, doi: 10.1109/LRA.2023.3246844.
- [24] A. Belherazem, R. Salim, A. Laidani, and M. Chenafa, "Vibration Control of a Two-Link Flexible Manipulator," *Autom. Control Comput. Sci.*, vol. 58, no. 3, pp. 346–358, June 2024, doi: 10.3103/S0146411624550017.
- [25] V. S. Gupta and Y. G. Joshi, "Modelling and Study of Dynamics and Control of a Two-Link Serial Manipulator," in *Recent Developments in Mechanics and Design*, D. K. Singh, S. Hegde, and A. Mishra, Eds., in *Lecture Notes in Mechanical Engineering*, Singapore: Springer Nature Singapore, 2024, pp. 1–11. doi: 10.1007/978-981-97-3874-8_1.
- [26] P. M. Wensing and J.-J. E. Slotine, "Coriolis Factorizations and their Connections to Riemannian Geometry," 2023, arXiv. doi: 10.48550/ARXIV.2312.14425.
- [27] S. Zhou, X. Li, M. Yin, D. Shang, and Z. Li, "Dynamic modeling and vibration suppression for double flexible manipulator with a tip payload," *J. Mech. Sci. Technol.*, vol. 37, no. 9, pp. 4441–4454, Sept. 2023, doi: 10.1007/s12206-023-0803-0.
- [28] J. Haack, F. Stark, S. Vyas, F. Kirchner, and S. Kumar, "Adaptive Model-Base Control of Quadrupeds via Online System Identification using Kalman Filter," 2025, arXiv. doi: 10.48550/ARXIV.2506.13432.
- [29] J. N. Clelland, T. J. Klotz, and P. J. Vassiliou, "Dynamic Feedback Linearization of Control Systems with Symmetry," 2021, doi: 10.48550/ARXIV.2103.05078.
- [30] M. Ming, Z. Feng, J. Ling, and X. Xiao, "Disturbance observer-based model prediction control with real-time modified reference for a piezo-actuated nanopositioning stage," *Trans. Inst. Meas. Control*, vol. 42, no. 4, pp. 813–822, Feb. 2020, doi: 10.1177/0142331219878048.
- [31] H. Ye and Y. Song, "Adaptive Control With Guaranteed Transient Behavior and Zero Steady-State Error for Systems With Time-Varying Parameters," *IEEECAA J. Autom. Sin.*, vol. 9, no. 6, pp. 1073–1082, June 2022, doi: 10.1109/JAS.2022.105608.
- [32] S. Gong, Z. Xu, L. Cheng, and X. Huang, "Self-organizing model reference adaptive control for aircraft with enhanced persistent excitation," *Aerosp. Sci. Technol.*, vol. 145, p. 108875, Feb. 2024, doi: 10.1016/j.ast.2024.108875.
- [33] Y. Ding, C. Guo, Y. Luo, H. Cai, X. Zhao, and R. Qu, "Research on PID self-tuning control based on recursive least squares parameter identification for the fast steering mirror-based optoelectronic tracking system," *Sens. Actuators Phys.*, vol. 386, p. 116323, May 2025, doi: 10.1016/j.sna.2025.116323.
- [34] M. M. Mustafa, C. D. Crane, and I. Hamarash, "Adaptive-Sliding Mode Trajectory Control of Robot Manipulators with Uncertainties," 2024, arXiv. doi: 10.48550/ARXIV.2408.03102.
- [35] T. Duong, A. Altawaitan, J. Stanley, and N. Atanasov, "Port-Hamiltonian Neural ODE Networks on Lie Groups For Robot Dynamics Learning and Control," 2024, arXiv. doi: 10.48550/ARXIV.2401.09520.
- [36] D. M. Le, O. S. Patil, C. F. Nino, and W. E. Dixon, "Accelerated Gradient Approach For Deep Neural Network-Based Adaptive Control of Unknown Nonlinear Systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 36, no. 4, pp. 6299–6313, Apr. 2025, doi: 10.1109/TNNLS.2024.3395064.
- [37] A. A. Malikopoulos, "Separation of learning and control for cyber-physical systems," *Automatica*, vol. 151, p. 110912, May 2023, doi: 10.1016/j.automatica.2023.110912.
- [38] C. Tang, B. Abbatematteo, J. Hu, R. Chandra, R. Martín-Martín, and P. Stone, "Deep Reinforcement Learning for Robotics: A Survey of Real-World Successes," 2024, arXiv. doi: 10.48550/ARXIV.2408.03539.
- [39] C. Rizzardo, F. Chen, and D. Caldwell, "Sim-to-real via latent prediction: Transferring visual non-prehensile manipulation policies," *Front. Robot. AI*, vol. 9, p. 1067502, Jan. 2023, doi: 10.3389/frobt.2022.1067502.
- [40] E. Inanc, Y. Gurses, A. Habboush, Y. Yildiz, and A. M. Annaswamy, "Neural Network Adaptive Control with Long Short-Term

- Memory,” 2023, arXiv. doi: 10.48550/ARXIV.2301.02316.
- [41] K. Guo and Y. Pan, “Composite adaptation and learning for robot control: A survey,” *Annu. Rev. Control*, vol. 55, pp. 279–290, 2023, doi: 10.1016/j.arcontrol.2022.12.001.
- [42] R. Raileanu, M. Goldstein, D. Yarats, I. Kostrikov, and R. Fergus, “Automatic Data Augmentation for Generalization in Deep Reinforcement Learning,” 2020, arXiv. doi: 10.48550/ARXIV.2006.12862.
- [43] E. Korkmaz, “Adversarial Robust Deep Reinforcement Learning Requires Redefining Robustness,” 2023, arXiv. doi: 10.48550/ARXIV.2301.07487.
- [44] P. Trentsios, M. Wolf, and D. Gerhard, “Overcoming the Sim-to-Real Gap in Autonomous Robots,” *Procedia CIRP*, vol. 109, pp. 287–292, 2022, doi: 10.1016/j.procir.2022.05.251.
- [45] G. Tiboni, K. Arndt, and V. Kyrki, “DROPO: Sim-to-Real Transfer with Offline Domain Randomization,” 2022, arXiv. doi: 10.48550/ARXIV.2201.08434.
- [46] M. Wang et al., “Meta-Reinforcement Learning Based on Self-Supervised Task Representation Learning,” 2023, arXiv. doi: 10.48550/ARXIV.2305.00286.
- [47] Z. Yin, C. Ye, H. An, W. Lin, and Z. Wang, “Robot Manipulation Skills Transfer for Sim-to-Real in Unstructured Environments,” *Electronics*, vol. 12, no. 2, p. 411, Jan. 2023, doi: 10.3390/electronics12020411.
- [48] Y. Liu et al., “Efficient End-to-End Detection of 6-DoF Grasps for Robotic Bin Picking,” 2024, arXiv. doi: 10.48550/ARXIV.2405.06336.
- [49] Z. Jiang, G. Tian, Y. Cui, T. Liu, Y. Gu, and Y. Wang, “Digital Twin System for Home Service Robot Based on Motion Simulation,” 2023, arXiv. doi: 10.48550/ARXIV.2309.05993.
- [50] P. D. Ngo, M. Tejedor, and F. Godtlielsen, “Data-Driven Robust Control Using Reinforcement Learning,” *Appl. Sci.*, vol. 12, no. 4, p. 2262, Feb. 2022, doi: 10.3390/app12042262.
- [51] P. Corke, *Robotics, Vision and Control*, vol. 118. in *Springer Tracts in Advanced Robotics*, vol. 118. Cham: Springer International Publishing, 2017. doi: 10.1007/978-3-319-54413-7.
- [52] R. Kirk, A. Zhang, E. Grefenstette, and T. Rocktäschel, “A Survey of Zero-shot Generalisation in Deep Reinforcement Learning,” 2021, doi: 10.48550/ARXIV.2111.09794.
- [53] M. Raoufi, H. Habibi, A. Yazdani, and H. Wang, “Robust Prescribed Trajectory Tracking Control of a Robot Manipulator Using Adaptive Finite-Time Sliding Mode and Extreme Learning Machine Method,” *Robotics*, vol. 11, no. 5, p. 111, Oct. 2022, doi: 10.3390/robotics11050111.
- [54] A. Kloss, G. Martius, and J. Bohg, “How to train your differentiable filter,” *Auton. Robots*, vol. 45, no. 4, pp. 561–578, May 2021, doi: 10.1007/s10514-021-09990-9.
- [55] P. D. Nguyen, N. H. Nguyen, and H. T. Nguyen, “Adaptive control for manipulators with model uncertainty and input disturbance,” *Int. J. Dyn. Control*, vol. 11, no. 5, pp. 2285–2294, Oct. 2023, doi: 10.1007/s40435-023-01115-7.
- [56] F. Jacob, E. H. Grosse, S. Morana, and C. J. König, “Picking with a robot colleague: A systematic literature review and evaluation of technology acceptance in human–robot collaborative warehouses,” *Comput. Ind. Eng.*, vol. 180, p. 109262, June 2023, doi: 10.1016/j.cie.2023.109262.
- [57] J. Arents and M. Greitans, “Smart Industrial Robot Control Trends, Challenges and Opportunities within Manufacturing,” *Appl. Sci.*, vol. 12, no. 2, p. 937, Jan. 2022, doi: 10.3390/app12020937.
- [58] H. Hu, K. Zhang, A. H. Tan, M. Ruan, C. G. Agia, and G. Nejat, “A Sim-to-Real Pipeline for Deep Reinforcement Learning for Autonomous Robot Navigation in Cluttered Rough Terrain,” *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 6569–6576, Oct. 2021, doi: 10.1109/LRA.2021.3093551.
- [59] S. Zhang, D. Zhai, Y. Xiong, J. Lin, and Y. Xia, “Safety-critical control for robotic systems with uncertain model via control barrier function,” *Int. J. Robust Nonlinear Control*, vol. 33, no. 6, pp. 3661–3676, Apr. 2023, doi: 10.1002/rnc.6585.