TECHNICAL ENGINEERING REPORT

UR5e Heavy-Payload Transportation

Model-Based Control and Residual Reinforcement Learning

Analytical Modeling, Cartesian Task Design, ROS 2–PhysX Validation, and Payload-Mismatch Experiments

Sooyong Kim
October 2026

ROS 2 Humble • NVIDIA Isaac Sim 6.1 • UR5e Manipulator

Abstract

This study investigates bounded residual reinforcement learning as an additional joint-torque correction to an established computed-torque controller for Cartesian payload transportation with a six-degree-of-freedom UR5e manipulator. A physically motivated three-second transportation reference is constructed from the robot's workspace, kinematic, and dynamic characteristics. A nominal controller is selected through 25 computed-torque PD and 30 model predictive control configurations, followed by independent ROS 2–Isaac Sim PhysX validation. The selected G025 computed-torque PD controller achieves 1.195 mm Cartesian tracking RMSE under matched 3 kg payload conditions. When the controller assumes 3 kg but the simulated plant carries 5 kg, its RMSE increases to 10.560 mm. Residual PPO, SAC, and TD3 policies are trained to generate bounded six-joint torque corrections; in PhysX, PPO reduces the 5 kg mismatch-case RMSE to 6.061 mm, approximately a 42.6% reduction. However, the same trained policies degrade performance under matched 3 kg and lighter 1 kg actual payload conditions. A separately trained PPO policy for the reverse 5 kg model/3 kg actual condition reduces PhysX RMSE from 8.039 to 4.585 mm. At an intermediate actual mass of 4 kg, the two policies yield 1.931 and 1.763 mm RMSE with different ascent/descent error characteristics. These results show that learned compensation can extend the usefulness of model-based control under selected uncertainties, but its effectiveness depends on the direction and magnitude of payload-model mismatch. Physics-based startup and effort-interface validation further demonstrate the importance of distinguishing calculated torque from the motion produced by the executed control system. The study remains simulation-based and does not claim formal stability, hardware safety certification, or universal payload generalization.

Keywords: UR5e; Cartesian trajectory tracking; computed-torque control; residual reinforcement learning; PPO; SAC; TD3; payload-model mismatch; ROS 2; Isaac Sim PhysX.

Contents

Abstract **i**

1 Introduction **1**
1.1 Industrial Motivation and Task-Oriented Robotic Manipulation 1
1.2 Payload Variability and the Reusability of Model-Based Controllers 2
1.3 Model-Based Control, Adaptive Methods, and Residual Reinforcement Learning 3
1.4 Research Questions and Experimental Scope 4
1.5 Engineering Methodology and Contributions 5

2 System Modeling **7**
2.1 Physical Model and Kinematic Representation 7
2.1.1 Robot configuration and coordinate frames 7
2.1.2 Forward kinematics and Cartesian task representation 8
2.1.3 Differential kinematics and Jacobian properties 9
2.1.4 Inverse kinematics and configuration consistency 10
2.2 Rigid-Body Dynamics and Payload Representation 10
2.2.1 Manipulator dynamics 10
2.2.2 Payload mass, center of mass, and inertia 11
2.2.3 Complete robot–payload dynamics 13
2.2.4 Model uncertainty and the inverse-dynamics difference 14
2.3 Model Reconstruction and Cross-Validation in Isaac Sim 14
2.3.1 Joint definitions and articulation consistency 15
2.3.2 Forward-kinematics cross-validation 15
2.3.3 Payload attachment and physical-model consistency 15
2.3.4 From model agreement to physical execution 16

3 Task Design **17**
3.1 Physical Constraints and Workspace Design 17
3.1.1 Manufacturer specifications and physical operating constraints 17
3.1.2 Configuration-dependent feasibility 18
3.1.3 Workspace configuration and endpoint selection 18
3.2 Cartesian Trajectory and Time Parameterization 20
3.2.1 Geometric construction of the transportation path 20
3.2.2 Curvature transition and its dynamic significance 20

3.2.3 Time parameterization and transport duration . . . 22
3.2.4 Reference resolution versus execution frequency . . . 23
3.3 Payload Modeling and Dynamic Feasibility . . . 23
3.3.1 Rigid payload representation . . . 23
3.3.2 Inverse-dynamics assessment . . . 24
3.3.3 Physical interpretation of payload-dependent effort . . . 25
3.4 Experimental Conditions and Payload-Mismatch Design . . . 26
3.4.1 Payload selection and model discrepancy . . . 26
3.4.2 Experimental comparison cases . . . 27
3.4.3 Maintaining a consistent transportation task . . . 27
3.4.4 Performance measures and task relevance . . . 28
3.4.5 Engineering significance of the experimental design . . . 29

4 Nominal Controller Design 30
4.1 Control Strategy Selection for Task-Oriented Payload Transport . . . 30
4.1.1 Control objectives of the transportation task . . . 30
4.1.2 Why computed-torque PD and MPC? . . . 31
4.1.3 Relationship to the preceding UR5e control study . . . 31
4.2 Computed-Torque PD: Formulation and Gain Selection . . . 32
4.2.1 Nominal error dynamics and modeling assumptions . . . 32
4.2.2 Gain parameterization . . . 33
4.2.3 Evaluation criteria . . . 33
4.2.4 Results of the PD screening . . . 33
4.3 Model Predictive Control: Formulation and Comparative Evaluation . . . 34
4.3.1 Prediction model . . . 34
4.3.2 Cost function and input constraints . . . 35
4.3.3 Initial MPC parameter search . . . 35
4.3.4 Second-stage MPC refinement . . . 35
4.3.5 Comparing MPC candidates by engineering objective . . . 36
4.3.6 Computation time and real-time feasibility . . . 36
4.4 Comparative Assessment and Selection of G025 . . . 37
4.4.1 Selection criteria . . . 37
4.4.2 Why G025 rather than the lowest-RMSE G022? . . . 37
4.4.3 Final nominal controller . . . 38
4.4.4 Implications for the subsequent learning study . . . 39

5 ROS 2–Isaac Sim PhysX Validation 40
5.1 From Analytical Control to an Executed Physics System . . . 40
5.1.1 Simulation and ROS 2 architecture . . . 40
5.1.2 Controller and physics time scales . . . 41
5.1.3 Why the controller frequency matters . . . 41
5.1.4 Command generation and actuation evidence . . . 42
5.2 Startup Gravity Sag and Deterministic Controller Initialization . . . 42

5.2.1 Why passive settling was insufficient . . . 43
5.2.2 Gravity preload and active return . . . 43
5.2.3 Stability gate . . . 43
5.2.4 Experimental state machine . . . 44
5.3 Closed-Loop Validation and Quantitative Results . . . 44
5.3.1 Tracking measurements . . . 45
5.3.2 Validation results . . . 45
5.3.3 Interpreting the tracking error . . . 46
5.3.4 Terminal-state trade-offs . . . 46
5.4 Validation Boundaries and Implications for Learning-Augmented Control . . . 48
5.4.1 Distinguishing modeling errors from execution errors . . . 48
5.4.2 Evidence required for a meaningful learning comparison . . . 48
5.4.3 Reproducibility and data interpretation . . . 49
5.4.4 Engineering interpretation . . . 49

6 Residual Reinforcement Learning Formulation 50
6.1 Learning-Augmented Control Under Payload-Model Uncertainty . . . 50
6.1.1 Nominal and actual dynamics . . . 50
6.1.2 Physical interpretation of model mismatch . . . 51
6.1.3 Residual control architecture . . . 51
6.2 Observation, Action, and Reward Design . . . 52
6.2.1 Observation vector . . . 52
6.2.2 Residual torque action . . . 53
6.2.3 Residual saturation versus total effort saturation . . . 54
6.2.4 Reward design . . . 54
6.2.5 Why Cartesian accuracy receives the dominant weight . . . 54
6.3 Numerical Learning Environment and Algorithm-Screening Protocol . . . 55
6.3.1 Numerical integration and control updates . . . 55
6.3.2 Episode duration and initialization . . . 55
6.3.3 PPO, SAC, and TD3 as candidate methods . . . 55
6.3.4 Training-budget design . . . 56
6.3.5 Evaluation and policy selection . . . 56
6.4 Scope, Identifiability, and Physical Interpretation of Learned Compensation . . . 57
6.4.1 Residual torque is not direct parameter identification . . . 57
6.4.2 Action limits and compensation authority . . . 58
6.4.3 Fixed-mass learning and conditional applicability . . . 58
6.4.4 Numerical training versus PhysX deployment . . . 58
6.4.5 Evaluation beyond aggregate RMSE . . . 59
6.4.6 Engineering significance of the formulation . . . 59

7 Comparative Evaluation of PPO, SAC, and TD3 60
7.1 Experimental Conditions and Learning Convergence . . . 60
7.1.1 Why compare learning progression rather than only final checkpoints? . . . 60

7.1.2 Best-checkpoint evaluation . . . 61
7.2 Joint-Resolved Residual Torque and Physical Interpretation . . . 62
7.2.1 Residual torque distributions . . . 62
7.2.2 Residual-bound occupancy . . . 63
7.2.3 Analytical comparison with payload-model torque mismatch . . . 64
7.3 Algorithm Assessment and Selection for PhysX Validation . . . 65
7.3.1 Limits of the offline comparison . . . 67

8 Payload-Mismatch Experiments in ROS 2–PhysX 68
8.1 Physics-Based Evaluation and Controlled Payload Conditions . . . 68
8.1.1 Maintaining the nominal control architecture . . . 68
8.1.2 Measurement consistency . . . 69
8.2 Tracking Results Across 1, 3, and 5 kg Payloads . . . 69
8.2.1 Heavier-than-modeled payload: 3 kg model, 5 kg actual . . . 69
8.2.2 Matched payload: 3 kg model, 3 kg actual . . . 70
8.2.3 Lighter-than-modeled payload: 3 kg model, 1 kg actual . . . 71
8.3 Physical Interpretation and Controller Applicability . . . 71
8.3.1 Compensation as a load-dependent action . . . 71
8.3.2 Why the matched-mass result matters . . . 72
8.3.3 Why PhysX validation changed the policy preference . . . 72
8.3.4 Limits of the payload-transfer experiment . . . 73

9 Bidirectional Payload-Mismatch Learning 74
9.1 Motivation for Reverse-Mismatch Learning . . . 74
9.1.1 Signed payload-model discrepancy . . . 74
9.1.2 Preserving the learning architecture . . . 74
9.2 Reverse-Mismatch PhysX Validation . . . 75
9.2.1 Interpretation of the tracking improvement . . . 75
9.3 Direction of Learned Torque Compensation . . . 76
9.3.1 Shoulder, elbow, and wrist compensation . . . 76
9.3.2 Why learned actions need not be exact negatives . . . 77
9.3.3 The significance of residual action bounds . . . 77
9.4 Engineering Implications and Transition to Cross-Evaluation . . . 78
9.4.1 Controller reuse does not imply policy universality . . . 78
9.4.2 Why evaluate an intermediate payload? . . . 78
9.4.3 Experimental boundaries . . . 79

10 Cross-Evaluation Under an Intermediate Payload 80
10.1 Experimental Rationale and Cross-Evaluation Design . . . 80
10.1.1 Two independently trained policies . . . 80
10.1.2 Common task and measurement conditions . . . 81
10.2 Overall Cartesian Tracking Results . . . 81
10.2.1 What the aggregate RMSE establishes . . . 81
10.2.2 Comparison with the original training conditions . . . 82

10.3 Ascent–Descent Tracking and Vertical Error Characteristics 82
10.3.1 Phase-dependent Cartesian tracking 83
10.3.2 Vertical tracking error 83
10.3.3 Signed bias versus error magnitude 84
10.3.4 Relationship to the trajectory geometry 84
10.4 Interpretation of Intermediate-Mass Generalization 85
10.4.1 Generalization within a fixed transportation task 85
10.4.2 Relative roles of the nominal controller and learned policy 86
10.4.3 Does Policy B outperform Policy A? 86
10.4.4 Implications for policy selection and adaptation 87
10.4.5 Engineering significance 87

11 Discussion **88**
11.1 Physical Interpretation and Control Implications 88
11.1.1 Model-based control as the foundation, not the limitation 88
11.1.2 Conditional benefits of residual reinforcement learning 89
11.1.3 Why mismatch direction matters 89
11.1.4 Similar tracking accuracy can conceal different control behavior 90
11.1.5 Task-phase dependence and the role of Cartesian evaluation 90
11.1.6 Implications for controller reuse 91
11.2 Simulation-to-Real Deployment and Troubleshooting 91
11.2.1 Correct torque calculation does not guarantee correct physical motion . 92
11.2.2 Gravity compensation and stationary holding 92
11.2.3 Commanded torque, effective actuation, and timing 93
11.2.4 Diagnosing unintended vibration and oscillatory behavior 93
11.2.5 Distinguishing observed problems from potential hardware hazards . . . 94
11.3 Safety, Validation Boundaries, and Future Development 94
11.3.1 Simulation as a prerequisite for hardware commissioning 95
11.3.2 Limits of the physical model 95
11.3.3 Clearance and collision safety 96
11.3.4 Stability and learned-policy boundaries 96
11.3.5 Hardware deployment considerations 96
11.3.6 Limitations of the experimental evidence 97
11.3.7 Future research priorities 97
11.3.8 Overall engineering assessment 98

12 Conclusions **99**

References **102**

Code and Data Availability **103**

A Additional Modeling and Calculation Notes **104**

B Parameter and Configuration Inventory **105**

C Complete PD and MPC Screening Data **106**
C.1 Computed-torque PD: all 25 candidates 106
C.2 MPC: stage I (18 candidates) 106
C.3 MPC: stage II (12 candidates) 107

D ROS 2–PhysX Validation Records **108**

E Reinforcement-Learning Records **109**

F Bidirectional and Intermediate-Payload Records **110**

G Original Residual RL Report **111**

Chapter 1

Introduction

1.1 Industrial Motivation and Task-Oriented Robotic Manipulation

Industrial robotic manipulators are employed to perform physical tasks such as material handling, assembly, machining, inspection, and component transportation. Although these applications differ in their operational requirements, they share a fundamental engineering consideration: the robot must interact with its surrounding environment and accomplish a prescribed task while respecting its physical operating limitations. Successful operation therefore cannot be evaluated solely by whether the robot follows a commanded joint trajectory. The resulting motion must also be appropriate for the object being manipulated, the workspace in which the task is performed, and the physical loads generated during execution.

This distinction becomes particularly important when a manipulator transports an object through a three-dimensional workspace. Unexpected deviations from the intended path may cause a payload to approach nearby equipment, violate an intended clearance, or reach the destination in an unsuitable configuration. Excessive joint effort, unfavorable acceleration profiles, or inadequate compensation of external loading may also produce undesirable oscillations and transient motion. Depending on the application, these behaviors can lead to workpiece damage, equipment interference, loss of operational reliability, or conditions that require protective intervention. Consequently, maintaining a predictable dynamic response under the anticipated operating conditions is an essential part of engineering a reliable robotic transportation system.

These considerations are consistent with the broader principles of industrial robot safety engineering. ISO 10218-1:2025 addresses safety requirements for industrial robots, while ISO 10218-2:2025 addresses their integration and applications. Such requirements emphasize that safe robot operation involves more than the performance of an individual control algorithm. Robot capabilities, application-specific hazards, protective measures, and the surrounding operating environment must be considered together [1]. From a task-oriented perspective, the end effector provides the physical interface through which a manipulator performs its intended work. Its position, orientation, and motion determine how a transported object moves relative to the destination and the surrounding environment. Joint coordinates remain indispensable because the mechanical system is actuated through its joints; however, joint tracking and task accomplishment represent different levels of the control problem.

For example, a controller may maintain a small average joint-position error while the end effector exhibits a task-relevant displacement caused by the combined effects of joint errors and manipulator configuration. The relationship between these quantities is governed by the robot's forward kinematics and Jacobian. An identical joint error can produce different Cartesian displacements at different configurations. Consequently, a joint-space metric alone does not

fully characterize the accuracy of object transportation. The present study therefore adopts Cartesian end-effector trajectory tracking as the primary task-level performance measure, while retaining joint-position, velocity, and torque information for controller design and physical interpretation.

This choice reflects the engineering scope of the investigation. The objective is not to develop a new robotic mechanism or establish a fundamentally new motion-control algorithm. Rather, it is to examine how an existing industrial manipulator, its analytical model, and established control methods can be integrated to perform a physically demanding transportation task under changing operating conditions. In this context, Cartesian tracking is not considered inherently superior to joint-space tracking for every robotic application. Instead, it is selected because the location and motion of the transported object are directly related to the intended task.

1.2 Payload Variability and the Reusability of Model-Based Controllers

The dynamic behavior of a robotic manipulator depends on its mechanical configuration, joint velocities and accelerations, gravitational loading, and the inertial properties of any attached payload. For a rigid-body manipulator, the equations of motion can be expressed in the familiar form

$$M(\mathbf{q};\boldsymbol{\theta})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q},\dot{\mathbf{q}};\boldsymbol{\theta}) + \mathbf{g}(\mathbf{q};\boldsymbol{\theta}) = \boldsymbol{\tau}.$$

where (q), $\dot{q}$, and $\ddot{q}$ denote joint position, velocity, and acceleration; τ is the applied joint torque; and θ represents the dynamic parameters of the robot and attached payload. The matrices and vectors governing inertia, velocity-dependent effects, and gravitational loading are coupled to these parameters. A change in payload mass or center of mass therefore alters the torque required to execute the same commanded trajectory. Importantly, this change is generally dependent on robot configuration and motion. It cannot necessarily be represented by a uniform torque offset across all joints or throughout the entire transportation task. Model-based control methods exploit the structure of these dynamics. When the mathematical model adequately represents the controlled system, feedforward compensation can account for a substantial portion of the effort required to follow a prescribed motion. Feedback control then corrects deviations between commanded and measured states.

However, the practical usefulness of a model-based controller is influenced by how accurately its assumed dynamic parameters represent the operating system. In an industrial environment, a robot may be required to transport components that share similar geometries and destinations but differ in mass. Even a repeated operation may involve changes in the payload, tooling, fixtures, or material properties. A control system configured for one loading condition may therefore encounter different dynamic requirements when the task is repeated with another object. One approach is to identify the new payload parameters and update the controller model whenever the operating condition changes. This is technically reasonable and may be appropriate when accurate parameter identification is available. Nevertheless, repeated model identification, controller adjustment, and validation can introduce additional engineering effort and operational complexity.

The practical question motivating the present work is whether an established nominal controller can continue to support similar transportation tasks over a range of payload conditions without requiring an independently redesigned controller for every individual load. This question must be distinguished from a claim that model-based controllers cannot tolerate uncertainty. Conventional feedback, robust control, and adaptive control can provide important mechanisms for addressing modeling errors. The purpose here is not to reject those approaches but to

investigate the behavior of a fixed nominal dynamics model when the actual payload deviates from the assumed condition. Two aspects of this uncertainty are particularly relevant.

First, the magnitude of the payload discrepancy influences the difference between the predicted and required joint efforts. Second, the direction of the discrepancy matters: the controller may assume a payload lighter or heavier than the actual object. Because the required dynamic compensation depends on configuration, gravity, and acceleration, the consequences of these two cases need not be symmetric in the resulting closed-loop tracking response. An engineering evaluation should therefore extend beyond asking whether a controller can tolerate one selected payload discrepancy. It should also consider whether the direction of model error changes the required compensation and whether a correction developed for one condition remains beneficial under another.

1.3 Model-Based Control, Adaptive Methods, and Residual Reinforcement Learning

Robotic manipulator control has developed through several complementary approaches, including conventional feedback regulation, nonlinear model-based compensation, robust and adaptive control, and optimization-based predictive control. These methods differ in how they use mathematical models, account for uncertainty, handle operating constraints, and distribute the computational burden between design and execution. Spong [2] reviews the development of robotic manipulator control and highlights the role of the mathematical structure of robot dynamics in enabling nonlinear, robust, and adaptive control methods. This body of work establishes that controller design is closely related to the physical properties of the robot, rather than being reducible to generic input–output regulation.

The problem of uncertain manipulator and payload parameters has also been studied extensively through adaptive control. Slotine and Li [3], for example, developed an adaptive control formulation combining feedback and dynamics compensation with online estimation of unknown manipulator and payload parameters. Their work demonstrates an important alternative to controller redesign: rather than treating all uncertain loading conditions as unrelated control problems, the controller can exploit the mathematical structure of robot dynamics to update its compensation. Such methods provide a relevant foundation for the present investigation. They also clarify that reinforcement learning is not the only available response to payload uncertainty. Online parameter identification and adaptive model-based compensation remain meaningful alternatives, particularly when the uncertainty can be described by identifiable physical parameters.

A different class of approaches employs model predictive control (MPC), where a mathematical model is used to predict system behavior over a finite horizon and determine the control action through constrained optimization. MPC is attractive when actuator limitations, predicted future states, and competing control objectives must be considered explicitly. Its effectiveness, however, depends on the suitability of its prediction model, optimization formulation, and achievable computation time [4]. The choice between these approaches should consequently depend on the demands of the application. For a prescribed free-space transportation task, numerical tracking accuracy is an important performance indicator, but it is not the only relevant consideration. Computational efficiency, torque demand, implementation complexity, and predictable response under uncertainty are also relevant to the practical selection of a nominal controller.

In recent years, reinforcement learning has provided an additional approach for developing control policies from interaction with a dynamic environment. Its appeal lies partly in the possibility of learning behavior that is difficult to describe explicitly using a selected mathematical

model. Nevertheless, replacing an established model-based controller with a fully learned policy is not necessarily the most appropriate strategy when the fundamental manipulator dynamics are already understood. An alternative is residual reinforcement learning, in which the established controller continues to provide the principal control action while a learned policy generates an additional corrective input. Johannink et al. [5] demonstrated this approach in robot control by combining conventional feedback with learned residual actions. Their experimental work on contact-rich robotic assembly showed that a learned correction can complement a structured controller rather than replacing it.

The relevance of this approach extends beyond the particular assembly task investigated in that study. It establishes a general architectural principle: existing physical knowledge can remain embedded in the controller while learning is directed toward effects that the nominal formulation does not adequately address. For payload transportation, the corresponding concept is to retain a dynamics-based nominal joint-torque controller and introduce a bounded learned torque correction. The nominal controller remains responsible for executing the prescribed motion, while the residual policy attempts to compensate for discrepancies associated with the assumed and actual dynamics. Such a structure offers several potential engineering advantages. It preserves the nominal controller as an interpretable reference, separates the learned correction from the baseline effort calculation, and permits the magnitude of the additional command to be explicitly limited.

These characteristics, however, do not establish that residual learning will necessarily improve tracking or preserve stability. A learned correction may be beneficial under the conditions represented during training but inappropriate when the underlying model error changes. Furthermore, limiting the magnitude of the residual action is not equivalent to guaranteeing closed-loop stability, collision safety, or satisfactory behavior throughout the manipulator workspace. The effectiveness of residual reinforcement learning must therefore be evaluated as a property of the combined controller, physical task, and uncertainty conditions, rather than attributed to the learning algorithm alone.

1.4 Research Questions and Experimental Scope

The central objective of this study is to investigate whether bounded residual reinforcement learning can improve Cartesian transportation performance when the payload mass assumed by an established model-based controller differs from the mass represented in the physical simulation. The investigation is organized around three related questions.

First, how does an established nominal controller respond to payload-model uncertainty during a prescribed transportation task?

A meaningful learning-based comparison requires a nominal controller whose behavior has already been examined under conditions consistent with its mathematical model. Without this reference, an apparent benefit from learning may be difficult to distinguish from deficiencies in controller implementation, initialization, numerical integration, or the simulator interface. The investigation therefore begins with analytical modeling, controller selection, and physics-based validation before introducing residual learning.

Second, can learned torque compensation improve task-level tracking without replacing the nominal controller?

The role of the residual policy is deliberately restricted to supplementary joint-torque correction. This arrangement permits the nominal controller and the learned action to be examined separately and provides a basis for evaluating whether improved Cartesian tracking is associated with physically interpretable compensation. PPO, SAC, and TD3 are considered as alternative continuous-action reinforcement-learning methods within this common architecture.

Their comparison is intended to examine learning behavior and the resulting control actions under a shared task definition, rather than to establish a universal ranking among reinforcement-learning algorithms.

Third, how do the magnitude and direction of payload-model mismatch influence learned compensation and its applicability outside the training condition?

A policy trained when the actual payload is heavier than the assumed model may learn a correction that is inappropriate when the physical loading is reduced. Conversely, a policy trained under the opposite discrepancy may develop a different compensation pattern. The study therefore includes bidirectional payload-mismatch conditions and intermediate-mass cross-evaluation to investigate the extent to which learned corrections depend on the loading assumptions used during training. These questions are examined through a prescribed Cartesian transportation task using a six-degree-of-freedom UR5e manipulator. The task begins after the payload has been attached rigidly to the end effector. Grasp acquisition, frictional grasp stability, contact-rich manipulation, and release are intentionally excluded so that the principal investigation can focus on transportation dynamics and torque control. The study employs an analytical robot model, a numerical learning environment, and an independently executed ROS 2–NVIDIA Isaac Sim PhysX validation environment.

The role of the physics-based simulation is particularly important. A control algorithm that behaves correctly in a numerical dynamics model may exhibit different behavior when executed through a separate articulation, state-feedback interface, and torque-command pipeline. Controller timing, initialization, gravitational loading, and effective actuation behavior must therefore be considered in addition to the mathematical control law. The investigation remains a simulation-based engineering study. It does not claim hardware-level safety certification, universal robustness to arbitrary payload variations, or formal stability guarantees for the learning-augmented closed-loop system.

1.5 Engineering Methodology and Contributions

To examine these questions, the experimental environment was constructed from the physical characteristics and operating constraints of the UR5e rather than from an arbitrarily selected robot motion. The task-design process begins by examining the manipulator's nominal reach, allowable configurations, Jacobian properties, and payload-dependent joint effort. The objective is to construct a transportation task that is sufficiently demanding to reveal meaningful dynamic behavior while remaining within the evaluated feasibility conditions. In particular, the transportation endpoints are positioned to produce distinct manipulator configurations, including a relatively extended destination posture. This choice is intended to expose the configuration-dependent relationship between joint effort and Cartesian motion. The trajectory is also deliberately structured. Rather than relying exclusively on a conventional high-order polynomial chosen to maximize derivative smoothness, the transportation path employs two parabolic vertical segments with distinct ascending and descending characteristics.

The resulting path provides an analytically interpretable change in vertical curvature. Combined with an independently defined time parameterization, this enables examination of how the reference geometry and motion timing influence joint acceleration and inverse-dynamics torque. The use of a curvature transition is a deliberate experimental feature rather than a recommendation for discontinuous-acceleration commands in production applications. Its geometric and dynamic consequences are examined explicitly, including the distinction between continuity of position and velocity and continuity of acceleration. After the transportation task has been established, nominal controller candidates are evaluated with emphasis on their suitability for executing the physical task. The investigation considers computed-torque PD

and MPC as two contrasting model-based approaches. The comparison is not intended to determine the universally optimal robotic controller; it provides a controlled basis for selecting a dependable nominal system against which the contribution of residual learning can subsequently be assessed.

Payload-model uncertainty is then introduced without changing the reference transportation task. By independently specifying the payload mass used for nominal control and the mass represented in the simulated plant, the study examines how discrepancies in dynamic parameters influence closed-loop behavior. Finally, the learned residual actions are evaluated not only through aggregate Cartesian tracking error but also through joint-resolved torque compensation and motion-phase-dependent response. This distinction is necessary because comparable end-effector tracking errors do not necessarily imply identical control actions or dynamic behavior. The engineering contribution of the study is therefore organized around the integration of four elements:

1. A physically motivated transportation benchmark constructed from manipulator limitations, workspace configuration, and interpretable trajectory dynamics.
2. A model-based nominal control architecture selected through numerical and physics-based assessment rather than tracking accuracy alone.
3. A comparative investigation of bounded residual reinforcement-learning methods under controlled payload-model discrepancy.
4. An examination of how learned torque compensation changes with mismatch direction, intermediate payload conditions, and different phases of the transportation motion.

An additional objective is to demonstrate the importance of separating analytical controller performance from the behavior of the implemented physical simulation. This separation is relevant to engineering safety: mathematical feasibility and successful numerical tracking are valuable prerequisites, but they cannot substitute for evaluating the physical response of the controlled system. The following chapters develop the analytical UR5e model, establish the transportation benchmark, examine nominal controller candidates, and document the subsequent learning and physics-based validation. The study concludes by interpreting the observed behavior in terms of model uncertainty, control-system applicability, and the engineering considerations required before transferring such methods to physical robotic hardware.

Chapter 2

System Modeling

2.1 Physical Model and Kinematic Representation

The reliability of a model-based robotic controller depends not only on the mathematical formulation of the control law but also on the consistency between the physical robot, its analytical representation, and the system through which control commands are executed. A theoretically accurate controller may produce incorrect joint efforts if the underlying model contains inconsistencies in joint definitions, coordinate transformations, mass properties, or actuator interfaces. For this reason, the present study began by establishing a common mechanical and mathematical representation of the UR5e manipulator before introducing the transportation task or designing its controller. The modeling objective was not to develop an alternative description of the UR5e. Instead, the existing six-degree-of-freedom robot description was used to reconstruct a mathematical representation in which the geometry, kinematics, and dynamic parameters could be examined independently and compared with the NVIDIA Isaac Sim articulation.

This distinction is important. A robot model should not be considered validated merely because it produces plausible joint configurations or visually reasonable motion. Its coordinate conventions, transformation relationships, and inertial properties must correspond to the physical system being represented.

2.1.1 Robot configuration and coordinate frames

The UR5e is modeled as a six-revolute-joint serial manipulator. Its generalized position vector is defined as

$$\mathbf{q} = \begin{bmatrix} q_1 & q_2 & q_3 & q_4 & q_5 & q_6 \end{bmatrix}^T,$$

with corresponding joint velocity and acceleration vectors $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$.

The six actuated joints, in the adopted order, are shoulder pan, shoulder lift, elbow, wrist 1, wrist 2, and wrist 3. This ordering is preserved in the analytical model, Isaac Sim articulation, and ROS 2 joint-state and effort-command interfaces.

The homogeneous transformation from the robot base frame B to the tool frame T is expressed as

$$^{B}T_{T}(\mathbf{q}) = {}^{B}T_{1}(q_1)\,{}^{1}T_{2}(q_2)\cdots{}^{5}T_{T}(q_6),$$

where

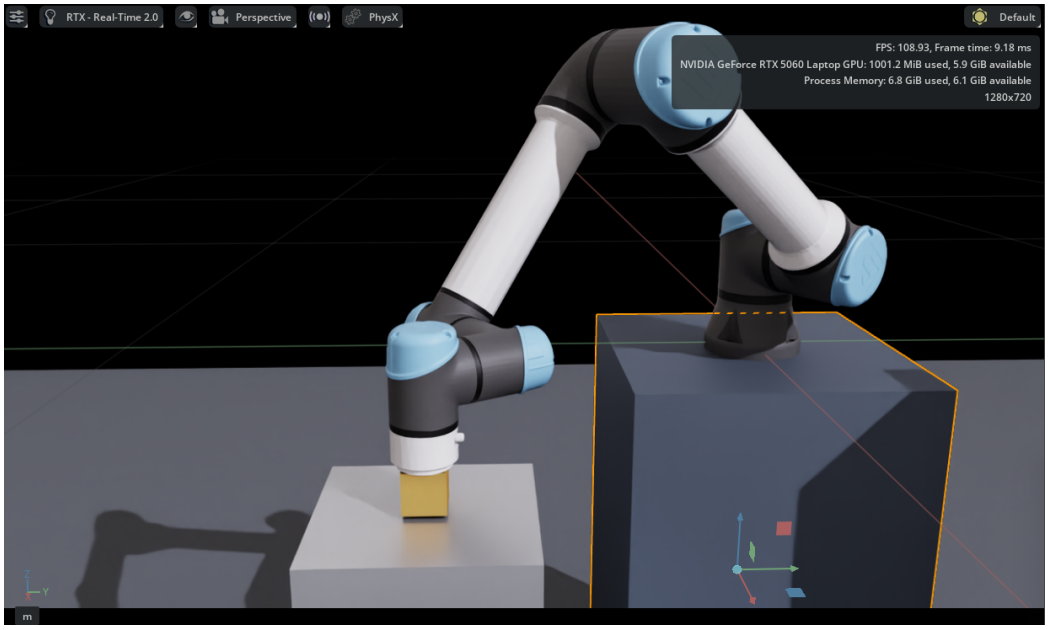


Figure 2.1: Validated UR5e cell and initial robot configuration (Isaac Sim screenshot).

$$ {}^{B}T_{T} = \begin{bmatrix} {}^{B}R_{T} & {}^{B}\mathbf{p}_{T} \\ \mathbf{0}^{T} & 1 \end{bmatrix}. $$

Here, ${}^{B}R_{T}$ represents the tool orientation relative to the base, while ${}^{B}\mathbf{p}_{T}$ represents its Cartesian position.

Each transformation incorporates the fixed geometric relationship between adjacent links and the rotation produced by the associated revolute joint. The transformation chain therefore provides a direct mapping from measured joint coordinates to the end-effector pose. The analytical coordinate convention was explicitly related to the Isaac Sim world frame. For the assembled simulation environment, the adopted base-to-world rotational mapping was

$$ {}^{W}R_{B} = \mathrm{diag}(-1, -1, 1). $$

The tool orientation selected for the transportation task is fixed relative to the world, with the local Tool0 $+Z$ axis pointing downward.

These conventions are not interchangeable. A frame transformation that is internally consistent but interpreted in the wrong reference frame can produce a correct-looking numerical result while representing the wrong physical pose. Such discrepancies are particularly important when gravitational loading and payload attachment are later introduced.

2.1.2 Forward kinematics and Cartesian task representation

Forward kinematics provides the relationship between joint coordinates and the physical location of the end effector:

$$ \mathbf{p}_{T} = f_{p}(\mathbf{q}), \qquad R_{T} = f_{R}(\mathbf{q}). $$

This mapping is central to the task-oriented formulation established in Chapter 1. Although the actuators operate in joint coordinates, the transportation objective is defined through

the end-effector position in Cartesian space. Therefore, the controller cannot be evaluated exclusively by individual joint-position errors. For small joint perturbations,

$$\Delta \mathbf{p}_T \approx J_v(\mathbf{q})\Delta \mathbf{q},$$

where J_v is the translational Jacobian.

This relationship demonstrates that the Cartesian consequence of a joint-position error depends on the manipulator configuration. For example, a given shoulder-joint angular error may produce a different end-effector displacement when the manipulator is extended than when the arm is folded closer to its base. Consequently, the magnitude of individual joint errors does not provide a configuration-independent measure of transportation accuracy. The forward-kinematics model is therefore used for two distinct purposes: generating and validating the transportation geometry, and reconstructing Cartesian tracking performance from the measured joint states during later control experiments. This second role is particularly important when interpreting the reported results. The Cartesian tracking errors obtained from the ROS 2 validation logs are analytical forward-kinematics estimates based on recorded joint coordinates. They should not be described as independent, directly measured end-effector positions from the PhysX engine.

2.1.3 Differential kinematics and Jacobian properties

The geometric Jacobian relates joint velocities to the translational and angular velocity of the tool:

$$\begin{bmatrix} \dot{\mathbf{p}}_T \\ \boldsymbol{\omega}_T \end{bmatrix} = J(\mathbf{q})\dot{\mathbf{q}},$$

with

$$J(\mathbf{q}) = \begin{bmatrix} J_v(\mathbf{q}) \\ J_\omega(\mathbf{q}) \end{bmatrix}.$$

For a revolute joint, its contribution to the translational Jacobian depends on the joint-axis direction and the moment arm between the joint origin and the point being considered. In a common coordinate frame, the corresponding column can be written as

$$J_{v,i} = \mathbf{z}_i \times (\mathbf{p}_T - \mathbf{p}_i),$$

where $\mathbf{z}_i$ is the axis direction and $\mathbf{p}_i$ is a point on that axis.

The angular-velocity Jacobian column is

$$J_{\omega,i} = \mathbf{z}_i.$$

These relationships provide an important physical interpretation of joint influence. The effect of a joint on Cartesian motion depends on both its axis direction and the current spatial arrangement of the robot.

The same geometric characteristics also influence the relationship between Cartesian loading and generalized joint effort. For an external wrench $\mathbf{w}$ expressed consistently with the Jacobian,

$$\boldsymbol{\tau}_{\text{ext}} = J^T(\mathbf{q})\mathbf{w}.$$

Thus, the Jacobian is not only a mathematical tool for converting velocities. It also describes how forces and moments applied at the tool contribute to joint torques. This dual role becomes important when transporting a payload through different robot configurations. The Cartesian acceleration is

$$\ddot{\mathbf{p}}_T = J_v(\mathbf{q})\ddot{\mathbf{q}} + \dot{J}_v(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}.$$

Therefore, the acceleration of the transported object depends not only on joint acceleration but also on configuration-dependent velocity coupling. These relationships form the connection between the Cartesian task introduced in Chapter 1 and the joint-space dynamics used for control.

2.1.4 Inverse kinematics and configuration consistency

A prescribed Cartesian trajectory does not uniquely determine the joint configuration of a six-degree-of-freedom manipulator. Multiple inverse-kinematics solutions may exist for the same end-effector pose, and different branches can exhibit substantially different joint motion and dynamic loading. For this reason, inverse kinematics was not treated as an isolated operation at each trajectory point. The adopted implementation uses sequential damped-least-squares inverse kinematics, initialized from the previously accepted joint configuration.

For a six-dimensional pose error $\mathbf{e}$, the iterative update has the form

$$\Delta\mathbf{q} = J^T \left(JJ^T + \lambda^2 I\right)^{-1} \mathbf{e},$$

where λ is a damping parameter.

The damping term regularizes the numerical inversion when the Jacobian becomes poorly conditioned. It does not eliminate physical singularities or guarantee a feasible solution in every workspace configuration. Initializing each solution from the preceding configuration helps preserve a consistent local inverse-kinematics branch and avoids accepting arbitrary alternative configurations that could introduce discontinuous joint motion. This approach is appropriate for the present study because the controller ultimately follows a continuous joint-space reference derived from the Cartesian transportation path. However, convergence of inverse kinematics alone does not establish physical feasibility. Joint limits, Jacobian conditioning, potential interference, velocity demand, and actuator effort must also be considered. Those checks are applied to the actual task in Chapter 3, after the workspace and transportation trajectory have been defined.

2.2 Rigid-Body Dynamics and Payload Representation

A kinematically feasible trajectory does not necessarily imply that the robot can execute it under its available joint effort. The torque required for a prescribed motion depends on the inertia of the links, velocity-dependent coupling, gravitational loading, and the additional properties of any attached object. A dynamics-based controller must therefore represent not only the geometric configuration of the manipulator but also the physical mechanisms that generate the required joint effort.

2.2.1 Manipulator dynamics

The six-degree-of-freedom UR5e is represented by the nonlinear rigid-body equation

$$M(\mathbf{q};\boldsymbol{\theta})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q},\dot{\mathbf{q}};\boldsymbol{\theta}) + \mathbf{g}(\mathbf{q};\boldsymbol{\theta}) = \boldsymbol{\tau},$$

where M is the joint-space inertia matrix, $\mathbf{c}$ is the vector of Coriolis and centrifugal generalized forces, $\mathbf{g}$ is the gravity generalized-force vector, and $\boldsymbol{\theta}$ denotes the model parameters.

The configuration-dependent inertia matrix describes the relationship between joint acceleration and the resulting inertial torque.

For a physically consistent rigid-body model, M is symmetric and positive definite over valid unconstrained configurations.

The velocity-dependent term accounts for nonlinear coupling between moving joints. Consequently, the effort required at one joint may depend on the velocities of several other joints. The gravity term depends on the position of each link's center of mass relative to the gravitational field and the corresponding joint moment arms. These properties explain why the torque demand of a multi-joint manipulator cannot generally be determined from payload mass and a single lever-arm calculation. The total required effort is a coupled function of the complete robot configuration and its motion. To understand how the model is constructed, consider the kinetic energy of the system:

$$T = \frac{1}{2}\dot{\mathbf{q}}^T M(\mathbf{q})\dot{\mathbf{q}},$$

and its potential energy $V(\mathbf{q})$.

The Lagrangian is

$$L(\mathbf{q},\dot{\mathbf{q}}) = T(\mathbf{q},\dot{\mathbf{q}}) - V(\mathbf{q}).$$

Applying the Euler–Lagrange equation,

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \tau_i,$$

provides the familiar coupled manipulator dynamics. The purpose of presenting this formulation is not to repeat a complete textbook derivation for six rigid links. It is to establish the physical meaning of the model terms subsequently used in inverse dynamics, computed-torque control, and payload-mismatch analysis. In particular, the controller must distinguish between the effort predicted by its assumed dynamic parameters and the effort required by the system carrying the actual payload.

2.2.2 Payload mass, center of mass, and inertia

The transported object is modeled as a rigid cube attached to the manipulator after grasp acquisition. The cube has side length

$$a = 0.05\text{m},$$

and its center of mass is defined relative to Tool0 as

$${}^{T}\mathbf{r}_p = \begin{bmatrix} 0 & 0 & 0.025 \end{bmatrix}^T \text{ m}$$

The position of the payload center of mass in the world frame is therefore

$${}^{W}\mathbf{p}_p = {}^{W}\mathbf{p}_T + {}^{W}R_T\,{}^{T}\mathbf{r}_p$$

Because the selected Tool0 local $+Z$ direction points downward, this offset places the cube center 25 mm below the tool origin during the prescribed fixed-orientation transportation task.

The payload's translational contribution to the manipulator inertia depends on the Jacobian evaluated at its center of mass.

For a fixed attachment offset $\mathbf{r} = {}^{W}R_{T}\,{}^{T}\mathbf{r}_{p}$, the corresponding translational Jacobian satisfies

$$J_{v,p} = J_{v,T} - [\mathbf{r}]_{\times} J_{\omega,T}$$

$$J_{v,p} = J_{v,T} - [\mathbf{r}]_{\times} J_{\omega,T},$$

where $[\mathbf{r}]_{\times}$ is the skew-symmetric cross-product matrix.

The translational payload inertia contribution is

$$M_{p,\text{trans}} = m_p J_{v,p}^{T} J_{v,p}.$$

This expression is significant because it shows that even when the payload mass is constant, its contribution to the joint-space inertia matrix changes with manipulator configuration. The position of the payload relative to the joint axes is therefore as important as its mass when analyzing dynamic loading. For a homogeneous rigid cube, the central rotational inertia tensor is

$$I_{p,c} = \frac{m_p a^2}{6} I_3.$$

For the initial 1 kg payload representation,

$$I_{p,c} = 4.166667 \times 10^{-4} I_3 \text{ kg m}^2$$

The rotational contribution to the joint-space inertia can be expressed as

$$M_{p,\text{rot}} = J_{\omega,p}^{T} R_p I_{p,c} R_p^{T} J_{\omega,p}.$$

The total attached-payload contribution to the inertia matrix is consequently

$$M_p = M_{p,\text{trans}} + M_{p,\text{rot}}.$$

The payload gravity contribution is related to its center-of-mass Jacobian:

$$\mathbf{g}_p(\mathbf{q}) = -J_{v,p}^{T} m_p \mathbf{g}_W$$

when $\mathbf{g}_W$ is defined as the physical gravitational acceleration vector and the left-hand gravity term represents the holding torque required to oppose gravity.

The sign depends on the chosen generalized-force convention; what matters is that the implementation follows one consistent convention throughout model construction and control. This formulation demonstrates that the attached payload affects several components of robot dynamics simultaneously. Additional mass changes the inertia, the gravity contribution, and the velocity-dependent terms derived from the configuration-dependent inertia. The magnitude and direction of these effects vary throughout the trajectory.

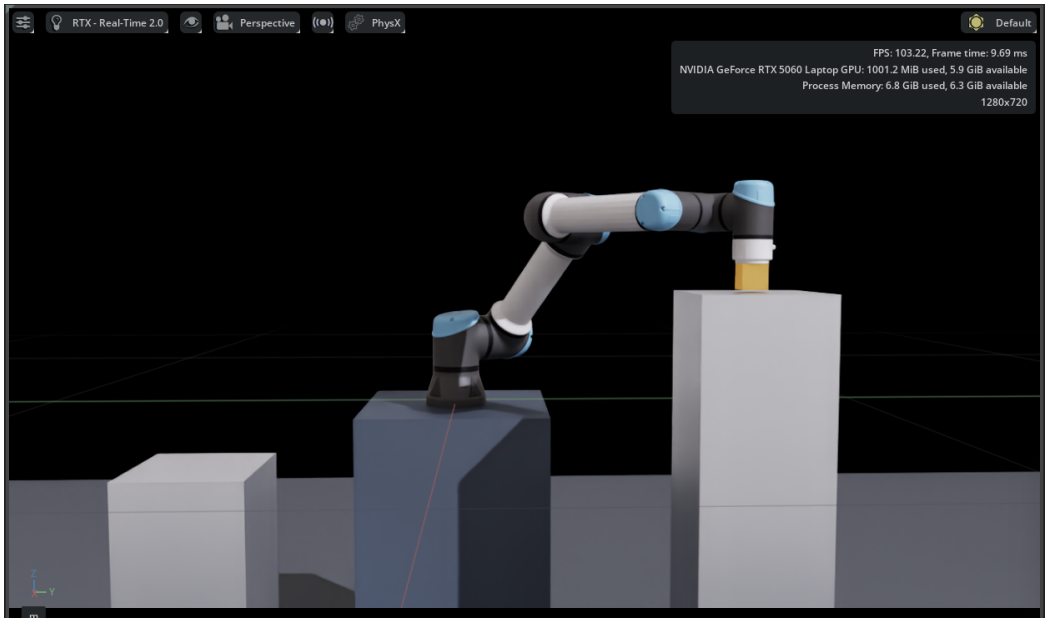


Figure 2.2: UR5e payload-transport setup at the place configuration (Isaac Sim screenshot).

2.2.3 Complete robot–payload dynamics

The loaded system is expressed as

$$M_{\text{total}} = M_{\text{robot}} + M_{\text{payload}},$$

with corresponding contributions to the velocity-dependent and gravitational terms. The dynamics can then be written as

$$M(\mathbf{q}; m_p)\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}; m_p) + \mathbf{g}(\mathbf{q}; m_p) = \boldsymbol{\tau}.$$

The distinction between the complete loaded model and an additional payload correction is important in the implementation. The previously established payload-dynamics routines return the dynamics of the robot with the configured payload already included. Their outputs must not be interpreted as payload-only increments and then added again to the unloaded robot dynamics. Doing so would count part of the inertia and gravitational loading twice, producing physically incorrect feedforward torque even if the numerical control algorithm were otherwise implemented correctly. Accordingly, the present work preserves the complete-model convention and explicitly distinguishes between total loaded dynamics and differences between two loaded models. The payload's rotational inertia was also considered separately during the preliminary reference-dynamics analysis.

Because the transportation reference maintains a fixed tool orientation, its reference angular velocity and angular acceleration are approximately zero apart from numerical differentiation effects. Under this particular reference motion, the additional torque associated with the cube's rotational inertia was correspondingly small in the recorded preliminary calculation. This observation should not be generalized to arbitrary manipulation tasks. A rapidly rotating payload, a different attachment geometry, or motion involving significant tool angular acceleration could produce substantially larger rotational-inertia contributions. The model therefore retains the physical meaning of the payload inertia even where its contribution to the specific reference trajectory is small.

2.2.4 Model uncertainty and the inverse-dynamics difference

The controller model and the simulated physical system are represented using different payload-mass variables:

$$m_{\text{model}} \qquad \text{and} \qquad m_{\text{actual}}.$$

The first is the mass assumed by the nominal dynamics calculation. The second is the mass assigned to the simulated plant.

Under a reference joint trajectory $\mathbf{q}_r(t)$, the inverse-dynamics torque predicted for a specified mass m is

$$\boldsymbol{\tau}_{\text{ID}}(t; m) = M(\mathbf{q}_r; m)\ddot{\mathbf{q}}_r + \mathbf{c}(\mathbf{q}_r, \dot{\mathbf{q}}_r; m) + \mathbf{g}(\mathbf{q}_r; m).$$

The theoretical torque discrepancy between two payload assumptions is therefore

$$\Delta\boldsymbol{\tau}_{\text{ID}}(t) = \boldsymbol{\tau}_{\text{ID}}(t; m_{\text{actual}}) - \boldsymbol{\tau}_{\text{ID}}(t; m_{\text{model}})$$

Expanding this difference gives

$$\Delta\boldsymbol{\tau}_{\text{ID}} = \Delta M\, \ddot{\mathbf{q}}_r + \Delta\mathbf{c} + \Delta\mathbf{g}$$

Here,

$$\Delta M = M(\mathbf{q}_r; m_{\text{actual}}) - M(\mathbf{q}_r; m_{\text{model}}),$$

with analogous definitions for $\Delta\mathbf{c}$ and $\Delta\mathbf{g}$.

For a rigid payload of unchanged geometry, attachment position, and inertia model per unit mass, these payload contributions are affine in the specified payload mass. The resulting reference inverse-dynamics difference therefore provides a physically interpretable relationship between mass discrepancy and torque demand. However, the torque discrepancy remains time-varying because the reference configuration, velocity, and acceleration change throughout the transport. This distinction becomes central to the subsequent reinforcement-learning analysis. The calculated inverse-dynamics difference is a reference-dependent physical quantity. It is not necessarily equal to the torque that a learned residual controller should apply during closed-loop operation. A real closed-loop controller acts on measured states that may differ from the reference. Its effort also depends on feedback errors, actuator limits, sampling, and the resulting plant response.

Therefore, direct agreement between a learned residual torque and the analytical inverse-dynamics difference is neither required nor sufficient to establish optimality. Instead, the analytical difference provides an important diagnostic reference for interpreting the direction and distribution of learned joint-torque corrections.

2.3 Model Reconstruction and Cross-Validation in Isaac Sim

A mathematical model is useful only when its representation of the controlled system is sufficiently consistent for the intended analysis. This requirement becomes particularly important when the numerical model and the physics simulator are implemented through different software components. In the present study, the UR5e analytical representation and NVIDIA Isaac Sim articulation were compared explicitly before the transportation and learning experiments were accepted. The purpose of this stage was to distinguish model-definition errors from genuine differences in controller performance.

2.3.1 Joint definitions and articulation consistency

The Isaac Sim UR5e articulation was inspected to establish the number and identity of controllable degrees of freedom. The final articulation contained six revolute degrees of freedom corresponding to the six UR5e joints. The robot asset also included fixed relationships in the articulation structure, which must not be confused with additional controllable joints. The numerical model and physics runtime therefore share a six-dimensional generalized coordinate representation. Joint order and sign conventions were retained consistently when transferring states between the analytical routines, simulation articulation, and ROS 2 interfaces. This is an elementary but essential verification step. If two joint coordinates are exchanged, or if the assumed positive rotation direction differs from the articulation convention, the controller may generate physically incorrect torque commands even though its matrix dimensions and software interfaces remain valid.

The original robot description was consequently treated as the reference for joint identity, axis configuration, and physical geometry.

2.3.2 Forward-kinematics cross-validation

The analytical forward-kinematics implementation was compared with poses derived from the Isaac Sim robot representation. The recorded validation included a representative six-joint configuration:

$$\mathbf{q} = \begin{bmatrix} -0.00556 \\ 0.41064 \\ -0.35299 \\ -0.28383 \\ 0.11242 \\ 0.22618 \end{bmatrix} \text{rad}.$$

Under the adopted frame transformation, the analytical base-to-tool position for this configuration was recorded as approximately

$${}^{B}\mathbf{p}_{T} = \begin{bmatrix} -0.815740 \\ -0.227741 \\ -0.124416 \end{bmatrix} \text{m}.$$

The corresponding reported analytical-to-Isaac position discrepancy was approximately

$$\|\Delta \mathbf{p}_{T}\| = 0.000206 \text{ mm}$$

This extremely small discrepancy is evidence of numerical agreement for the tested pose and coordinate conventions. It should not be interpreted as physical end-effector positioning accuracy or an estimate of real-world measurement uncertainty. Both quantities arise from numerical representations of the robot. More importantly, this comparison validates the transformation implementation rather than the complete dynamics model. Agreement in forward kinematics does not automatically establish agreement in link inertia, friction, joint drives, contact modeling, or closed-loop actuation. These aspects require separate consideration.

2.3.3 Payload attachment and physical-model consistency

The transported cube is represented in Isaac Sim as a rigid body connected to the manipulator through a fixed joint. The fixed connection is an intentional simplification: the study begins

after grasp acquisition, and the payload is assumed to remain rigidly attached throughout the motion. The attachment frame was inspected to confirm the relationship between Tool0 and the payload center of mass. This verification was necessary because the cube geometry was represented using a scaled USD primitive. The relationship between local attachment coordinates, scale, and world coordinates could otherwise lead to an incorrect payload offset. A local-anchor interpretation issue was identified and corrected during the model setup. The accepted attachment was then rechecked against the intended Tool0-to-payload geometry. The recorded discrepancy in the payload-center position relative to Tool0 was approximately

$$1.3 \times 10^{-5}\text{mm}.$$

As with the forward-kinematics comparison, this is a numerical consistency check for the evaluated configuration, not a measurement of physical grasp rigidity or contact performance. The physical meaning of the payload representation also extends beyond its position. Mass and inertia properties must be interpreted according to their effective simulation behavior. For example, a zero-valued inertia field authored in a USD property does not necessarily establish that PhysX uses zero rigid-body inertia, since the engine may infer effective inertia from mass and collision geometry. For the analytical calculation, the homogeneous-cube inertia is defined explicitly. Where exact equivalence between analytical and simulated inertial properties is required, the corresponding effective PhysX values would need independent inspection. This distinction prevents model fidelity from being overstated.

2.3.4 From model agreement to physical execution

The model-validation process was organized around several distinct forms of evidence. Kinematic agreement establishes consistency in the mapping between joint coordinates and end-effector poses. Dynamic calculations establish the predicted torque required for a specified motion under the assumed physical parameters. Geometric and inverse-kinematics checks examine whether the intended task configurations are admissible under the adopted constraints. Closed-loop PhysX validation then examines how the controlled articulation actually responds when effort commands are applied through the simulation interface. These stages address related but different engineering questions. For example, a controller may compute the correct gravitational holding torque according to the analytical model while the simulated manipulator nevertheless moves unexpectedly because the applied command is delayed, an internal drive acts simultaneously, or the controller has not yet been initialized.

Such behavior is not necessarily evidence that the analytical rigid-body equations are incorrect. Similarly, a robot may appear motionless while resting in an unintended equilibrium configuration. Low measured joint velocity does not establish that the intended initial pose has been reached. These distinctions became particularly relevant during the later ROS 2–PhysX controller validation, where gravitational sag, startup control, and effort-command timing required explicit treatment. The corresponding runtime observations and corrective procedures are presented in Chapter 5, while their implications for physical robot deployment and safety are discussed in Chapter 11. The role of Chapter 2 is therefore to establish a traceable correspondence between the physical system being represented and the quantities subsequently used for controller design.

The validated kinematic representation provides the connection between joint motion and Cartesian task performance. The dynamic representation explains how robot configuration and payload parameters influence required effort. The simulator cross-checks establish that these calculations refer to a consistent mechanical system within the documented validation scope. Together, they provide the physical and mathematical foundation for the task-design process in Chapter 3.

Chapter 3

Task Design

3.1 Physical Constraints and Workspace Design

The transportation task was developed from the physical capabilities and limitations of the UR5e manipulator rather than from an arbitrarily selected set of pick-and-place coordinates. This distinction represents a fundamental consideration in task-oriented robotic engineering: a geometrically reachable motion is not necessarily dynamically feasible, and a dynamically feasible reference does not automatically guarantee satisfactory closed-loop execution. For an industrial transportation application, the robot must move the attached object through the prescribed workspace while maintaining sufficient control authority, respecting its operating limits, and avoiding unintended motion that could compromise the task or the surrounding equipment. Accordingly, the experimental design began with an examination of the UR5e's physical operating envelope. The intention was to establish a demanding but feasible transportation task whose behavior could be explained through manipulator geometry, nonlinear dynamics, and payload-dependent joint effort.

The objective was not to locate the mathematically most demanding configuration over the entire UR5e workspace. Such a claim would require a separate constrained optimization over robot configurations, payload properties, velocities, and accelerations. Instead, the study deliberately selected extended configurations and a dynamically informative trajectory while evaluating their feasibility through analytical calculations and simulation-based geometric checks.

3.1.1 Manufacturer specifications and physical operating constraints

The UR5e is a six-degree-of-freedom industrial collaborative manipulator. According to the manufacturer's November 2023 technical specification, the robot has a nominal reach of 850 mm and a rated maximum payload of 5 kg. The same document specifies a maximum joint speed of $\pm 180^\circ/\mathrm{s}$ for all six axes and a nominal control frequency of 500 Hz. These specifications provide useful physical references for defining the experimental workspace and motion requirements.

Table 3.1 — Selected manufacturer specifications for the UR5e

Parameter	Manufacturer specification
Degrees of freedom	6 revolute joints
Nominal reach	850 mm
Rated maximum payload	5 kg

Joint working range	±360° for each axis
Maximum joint speed	±180°/s for each axis
Typical TCP speed	1 m/s
Nominal control frequency	500 Hz
Pose repeatability	±0.03 mm under ISO 9283

These specifications must be interpreted according to their intended meaning. The nominal reach describes a geometric capability; it does not imply that every configuration within that reach has identical dynamic performance or available joint effort. Similarly, the rated payload capacity is not independent of its center-of-mass location and operating conditions. The manufacturer provides a payload curve describing the relationship between allowable payload and center-of-gravity offset. Consequently, selecting a payload at or below the nominal 5 kg rating is an initial design consideration rather than proof that every commanded motion is admissible. The manufacturer-defined pose repeatability and maximum joint speed also describe different aspects of robot performance. Repeatability is not equivalent to dynamic Cartesian tracking accuracy, while a velocity requirement below the rated maximum does not establish that the corresponding acceleration and torque requirements can be achieved under every payload condition.

The manufacturer's 500 Hz control frequency was used as a relevant reference for the simulation architecture, but it must not be confused with the physical integration frequency of Isaac Sim or the execution rate of the research controller. Those frequencies describe different computational operations and are examined in Chapter 5. In addition to the manufacturer specifications, the adopted Isaac Sim robot asset contained joint-effort limits of

$$\boldsymbol{\tau}_{\max} = [150, 150, 150, 28, 28, 28]^T \text{ Nm}$$

These are the effort constraints configured in the simulation environment. They are not independently verified manufacturer continuous or peak motor-torque ratings. The manufacturer specifications, simulation settings, and calculated dynamic requirements are therefore maintained as separate categories of evidence throughout this study.

3.1.2 Configuration-dependent feasibility

The operating limitations of a serial manipulator cannot be described solely by its reach and rated payload. The relationship between joint motion and end-effector velocity is expressed by the geometric Jacobian.

$$\dot{\mathbf{x}} = J(\mathbf{q})\dot{\mathbf{q}}, \qquad \boldsymbol{\tau}_{\text{ext}} = J^T(\mathbf{q})\mathbf{w}$$

The total effort required for dynamic motion includes inertial, velocity-dependent, and gravitational components:

$$\boldsymbol{\tau} = M(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q})$$

3.1.3 Workspace configuration and endpoint selection

The simulation environment consists of a fixed-base UR5e manipulator, a pick table, an elevated place table, and a cubic payload rigidly attached to the tool. The pick and place locations were

intentionally separated to produce different robot configurations and different gravitational and inertial loading characteristics. The horizontal distance from the base to the pick target was approximately 0.680 m, corresponding to approximately 80% of the UR5e's nominal reach. The corresponding horizontal distance to the place target was approximately 0.807 m, or approximately 95% of nominal reach. These ratios were used as workspace-design references. They are based on horizontal target distance and should not be interpreted as exact normalized measures of the complete three-dimensional reachable workspace or as percentages of actuator capacity.

The accepted target coordinates are summarized in Table 3.2.

Table 3.2 — Accepted Cartesian transportation targets

Parameter	Pick	Place
Tool0 X (m)	0.4648	0.5520
Tool0 Y (m)	−0.4959	0.5889
Tool0 Z (m)	0.0050	0.3550
Tabletop Z (m)	−0.050	0.300
Horizontal distance from base (m)	Approximately 0.680	Approximately 0.807
Ratio to nominal reach	Approximately 0.80	Approximately 0.95

The place table was positioned 0.350 m above the pick table, introducing a meaningful vertical component to the transportation task. The tool orientation was held constant, with Tool0 local positive (Z) directed downward. The payload center was positioned 25 mm below Tool0, so that the bottom of the 50 mm cube had a nominal clearance of approximately 5 mm above each tabletop at the corresponding target configuration. The purpose of the extended destination configuration was to examine transportation in a region where manipulator geometry significantly influences the relationship between Cartesian motion and joint loading.

However, approaching the nominal reach limit is not equivalent to approaching a kinematic singularity. The actual conditioning of the Jacobian must be evaluated. Accordingly, the accepted Cartesian path was converted into joint configurations using sequential inverse kinematics. All 121 prescribed waypoint solutions converged. The maximum recorded condition number of the six-dimensional Jacobian was approximately 15.53, while the minimum recorded singular value was approximately 0.1237 under the numerical Jacobian convention used in the original verification. These results indicate that the evaluated waypoint sequence did not encounter a detected kinematic singularity under the adopted numerical criteria. The robot geometry was also reviewed for possible interference. A total of 481 configurations were sampled, including the original 121 waypoints and three intermediate joint-space configurations between each adjacent pair.

Potential broad-phase overlap candidates were examined with more detailed mesh-intersection tests. No triangle-mesh intersections were found among the evaluated candidates. This supports the sampled geometric feasibility of the prescribed path. It does not constitute a continuous swept-volume collision guarantee, nor does it establish collision-free behavior for every possible tracking deviation during closed-loop execution.

The selected workspace therefore represents a deliberately demanding but geometrically feasible transportation arrangement. Its significance lies not merely in the distance between the two workstations but in the changing relationship between manipulator configuration, Cartesian motion, payload loading, and required joint effort.

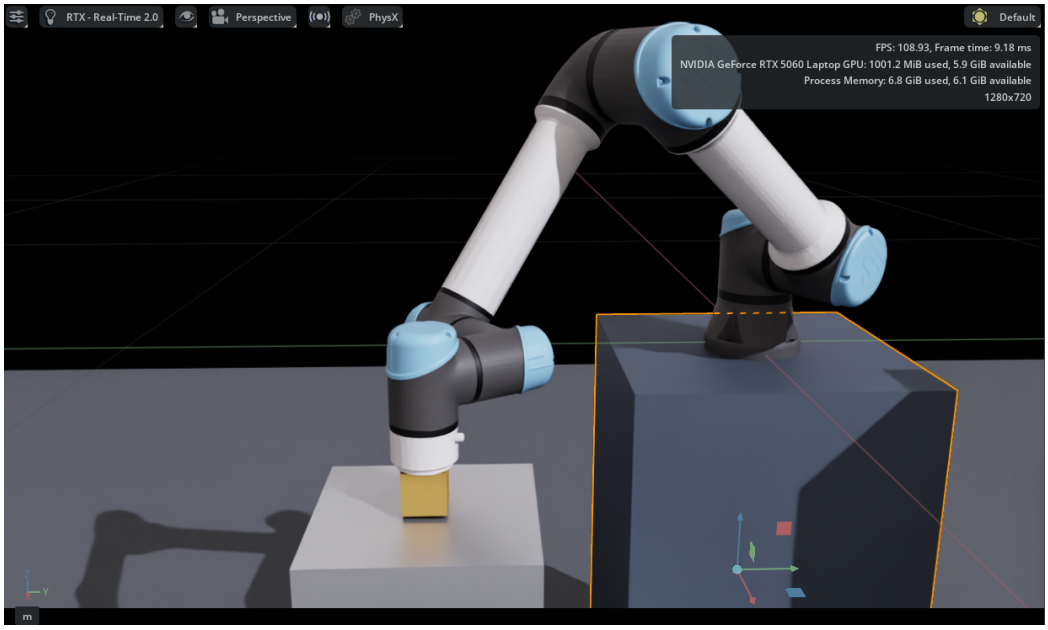


Figure 3.1: Accepted pick configuration in Isaac Sim.

3.2 Cartesian Trajectory and Time Parameterization

Once the transportation endpoints were established, the next design question was how the end effector should move between them. A conventional trajectory-generation objective is to connect the prescribed endpoints with smooth position, velocity, and acceleration profiles. Higher-order polynomial interpolation is commonly employed for this purpose, particularly when continuity of acceleration or jerk reduction is required. Such methods are appropriate for many robotic applications. However, the present investigation was not intended simply to identify the smoothest possible trajectory. The objective was to construct a physically interpretable transportation reference containing distinct ascending and descending phases, while deliberately introducing a change in vertical curvature whose effect on dynamic loading and controller response could be examined. For this reason, the Cartesian path was constructed using two parabolic segments rather than a single higher-order polynomial.

3.2.1 Geometric construction of the transportation path

Let $s \in [0, 1]$ represent normalized progress along the prescribed transportation path. The horizontal coordinates are defined as

$$x(s) = 0.4648 + 0.0872s, \qquad y(s) = -0.4959 + 1.0848s$$

$$z(s) = \begin{cases} 0.500 - 0.495(1 - s/s_j)^2, & 0 \le s \le s_j, \\ 0.500 - 0.145((s - s_j)/(1 - s_j))^2, & s_j < s \le 1, \end{cases}$$

$$s_j = \frac{2}{3}, \quad z(0) = 0.005 \text{ m}, \quad z(s_j) = 0.500 \text{ m}, \quad z(1) = 0.355 \text{ m}$$

3.2.2 Curvature transition and its dynamic significance

The first derivative of the vertical path is

$$z_s = \begin{cases} \dfrac{0.990}{s_j}(1 - s/s_j), & s \le s_j, \\ -\dfrac{0.290}{(1 - s_j)^2}(s - s_j), & s > s_j, \end{cases}$$

At the transition,

0.

]

Thus, the vertical trajectory is continuous in both position and first derivative. However, the second derivative differs between the two segments:

$$z_{ss} = \begin{cases} -2.2275 \text{ m}, & s < s_j, \\ -2.6100 \text{ m}, & s > s_j, \end{cases}$$

The change in the vertical second derivative is therefore -0.3825 m.

$$\Delta z_{ss} = -0.3825 \text{ m}$$

$$\ddot{\mathbf{p}} = \mathbf{p}_{ss}\dot{s}^2 + \mathbf{p}_s\ddot{s}$$

The first term depends on geometric curvature and the square of the path-progress rate. The second depends on changes in the traversal rate. At the parabolic transition, the geometric second derivative has different one-sided values. Consequently, when the path is traversed at nonzero speed, the corresponding acceleration can also exhibit a finite discontinuity. For the recorded transition speed of approximately 0.24196 s^{-1}, the curvature-related change in vertical acceleration is approximately

$$\ddot{\mathbf{p}} = \mathbf{p}_{ss}\dot{s}^2 + \mathbf{p}_s\ddot{s}$$

$$\Delta z_{ss}\dot{s}^2 \approx -0.0224 \text{ m/s}^2$$

This calculation isolates the geometric curvature contribution under a continuous time law. It is not a direct measurement of the robot's actual acceleration during PhysX execution. The effect on generalized joint acceleration and torque is more complex because it depends on the inverse-kinematics relationship, Jacobian variation, and complete nonlinear dynamics. The selected trajectory therefore provides a controlled opportunity to examine the relationship between reference geometry, acceleration demand, and the resulting torque compensation. The deliberate curvature change should not be confused with a manipulator singularity. A Jacobian singularity concerns the loss of independent Cartesian motion directions; the parabolic transition is a property of the commanded reference. Furthermore, the inclusion of an acceleration discontinuity is not presented as a recommended production trajectory for general industrial transportation.

In a physical implementation, jerk limitations, actuator bandwidth, structural flexibility, and payload stability may motivate additional trajectory smoothing. Here, the curvature transition was retained because the purpose was to observe and interpret differences in dynamic response under controlled simulation conditions.

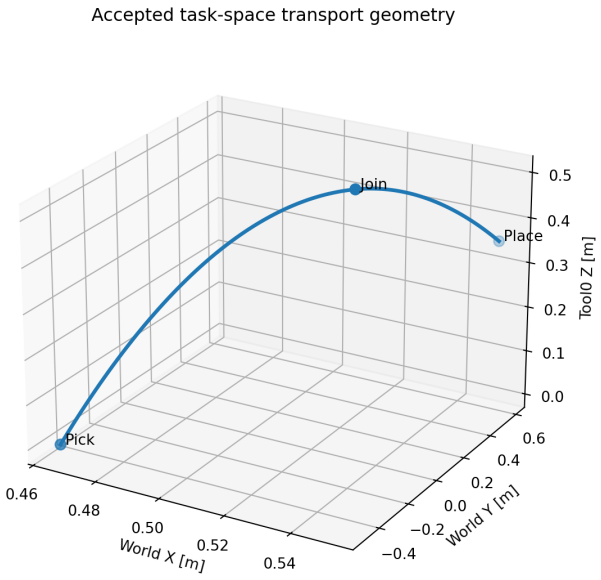


Figure 3.2: Accepted Cartesian transportation reference.

3.2.3 Time parameterization and transport duration

A geometric path does not uniquely define the dynamic effort required for execution. The same spatial trajectory can be traversed at different speeds, producing different joint velocities, accelerations, and inertial torques. The transportation duration was therefore treated as a separate design variable. For the accepted reference, the total duration was fixed at

$$T = 3.0 \text{ s}, \qquad \mathbf{q} = \mathbf{q}(s), \quad s = s(t)$$

$$\dot{\mathbf{q}} = \mathbf{q}_s \dot{s}$$

while the joint acceleration is

$$\ddot{\mathbf{q}} = \mathbf{q}_{ss} \dot{s}^2 + \mathbf{q}_s \ddot{s}$$

This decomposition is useful because it distinguishes acceleration due to the geometry of the path from acceleration associated with the selected time law. To obtain zero path-progress speed at the beginning and end of the transportation task, the time parameterization uses

$$s = \frac{1 - \cos\theta}{2}$$

However, the selected time law does not impose a complete stop at the join. It therefore mitigates the acceleration change without mathematically eliminating it. For the accepted three-second reference, the transition occurs at approximately

$$\dot{\theta} = K \left[1 - 0.55 \exp\left(-\frac{1}{2} \left(\frac{s - 2/3}{0.055} \right)^2 \right) \right]$$

For this reason, the trajectory was subjected to inverse-dynamics analysis before being used for control experiments.

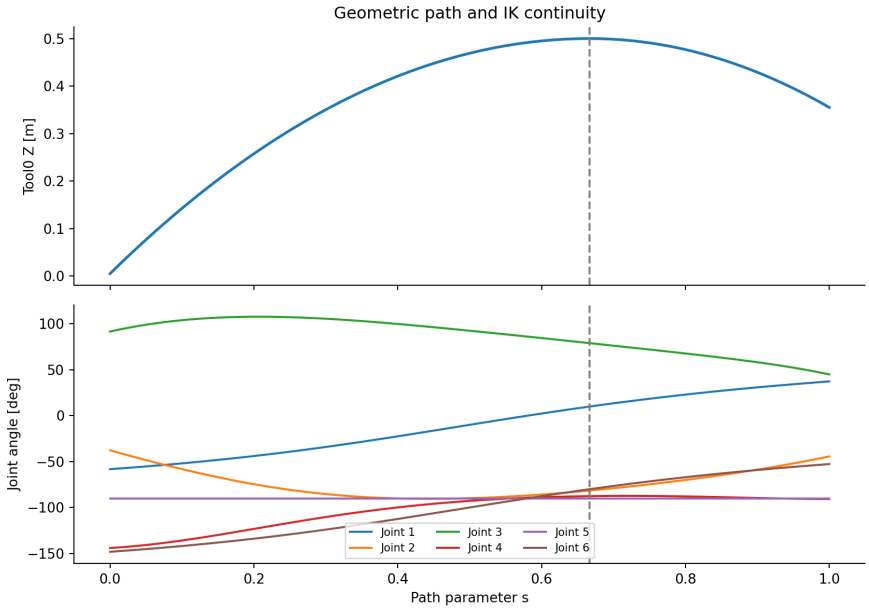


Figure 3.3: Reference path and associated joint profiles.

3.2.4 Reference resolution versus execution frequency

The accepted trajectory contains 121 reference configurations distributed over the three-second motion. These configurations define the geometric and time-parameterized reference. Their number does not determine the feedback-controller execution frequency. The reference contains nonuniform time intervals as a consequence of the selected time law. Controller implementations must therefore retrieve the appropriate reference states at their own execution times rather than treating each recorded waypoint as a fixed-period control update. The subsequent physics-based control architecture uses a 500 Hz PhysX integration rate and a 50 Hz learning-augmented torque-controller update rate. These frequencies serve different purposes. Physics integration advances the simulated robot dynamics, while the controller computes and updates the commanded effort from the available state and reference information. The selection and implementation of these frequencies, together with command holding, time synchronization, and their influence on closed-loop performance, are examined in Chapter 5.

The reference geometry and transportation duration are kept independent of the controller frequency so that subsequent controllers can be compared against the same prescribed task.

3.3 Payload Modeling and Dynamic Feasibility

The physical feasibility of the transportation task was assessed through the inverse dynamics of the UR5e and its attached payload. The purpose of this assessment was not merely to verify that the robot could reach the prescribed targets. It was to estimate the effort required to execute the selected motion and identify the dominant relationships between manipulator configuration, gravity, acceleration, and payload loading. This distinction becomes particularly important in the present study because payload variation is subsequently introduced as the principal source of model uncertainty.

3.3.1 Rigid payload representation

The payload is modeled as a homogeneous rigid cube with a side length of

$$a = 0.05 \text{ m}, \qquad {}^{T}\mathbf{r}_p = [0, 0, 0.025]^T \text{ m}$$

$${}^{T}\mathbf{r}_p = \begin{bmatrix} 0 & 0 & 0.025 \end{bmatrix}^T \text{ m}$$

The tool orientation is held constant along the reference trajectory, with the local positive (Z) direction pointing downward in the world frame. The payload is rigidly connected to the manipulator in Isaac Sim through a fixed joint. This idealization isolates the transportation problem from grasp acquisition, finger-contact behavior, and release mechanics. It allows changes in the simulated response to be interpreted primarily through robot dynamics, payload mass, reference motion, and the control architecture. The simplification does not imply that grasp stability is unimportant for physical deployment. Rather, grasp mechanics are excluded from the present experimental scope so that the effects of payload-model uncertainty on torque-controlled transportation can be examined separately. For a uniform solid cube of mass (m_p), the inertia tensor about its center of mass is

$$I_{p,c} = \frac{m_p a^2}{6} I_3$$

The corresponding contributions to the robot's joint-space inertia and gravitational loading were established in Chapter 2. The center-of-mass offset used here is defined relative to Tool0. It should not automatically be treated as the offset from the manufacturer's tool flange used in the published payload-capacity curve, because the relevant frame origins may differ. Therefore, while the selected experimental payload masses are at or below the datasheet's nominal 5 kg rating, complete hardware payload admissibility would require evaluating the effective center-of-gravity position relative to the appropriate manufacturer reference frame and the actual application conditions.

3.3.2 Inverse-dynamics assessment

For a prescribed joint-space reference $\mathbf{q}_r(t)$ and payload mass m_p, the theoretical torque requirement is expressed as

$$\boldsymbol{\tau}_{\text{ID}} = M(\mathbf{q}_r; m_p)\ddot{\mathbf{q}}_r + \mathbf{c}(\mathbf{q}_r, \dot{\mathbf{q}}_r; m_p) + \mathbf{g}(\mathbf{q}_r; m_p)$$

This preliminary mass must be distinguished from the 1 kg actual-payload condition introduced later in the controller-mismatch experiments. The former belongs to task development; the latter belongs to closed-loop robustness evaluation. For the accepted three-second, 1 kg reference, the recorded peak absolute joint torques are shown in Table 3.3.

Table 3.3 — Preliminary inverse-dynamics effort requirements for the 1 kg payload

Joint	Peak absolute torque (Nm)	Simulation effort limit (Nm)	Peak utilization
Shoulder pan	3.779	150	2.52%
Shoulder lift	62.435	150	41.62%
Elbow	26.437	150	17.62%
Wrist 1	2.976	28	10.63%
Wrist 2	0.276	28	0.99%
Wrist 3	Less than 10^{-7}	28	Approximately 0%

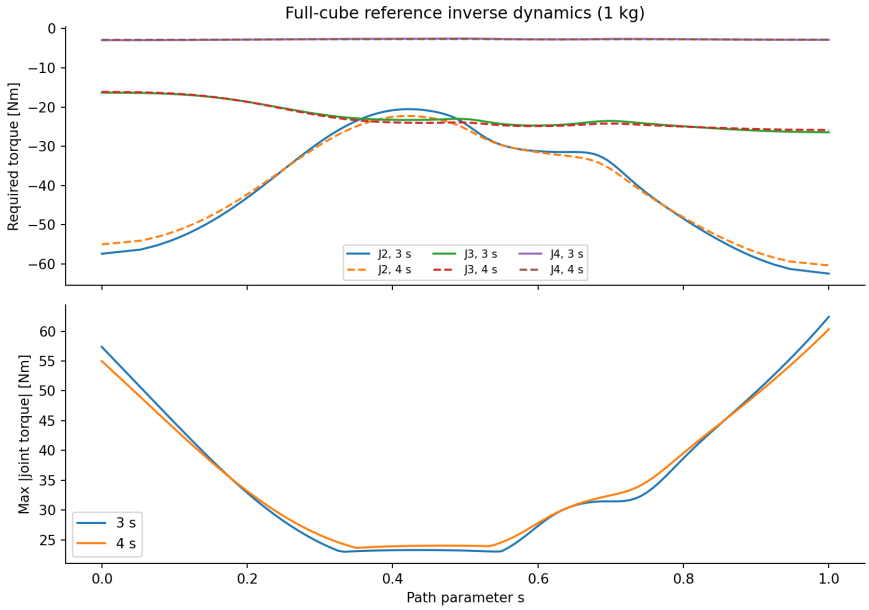


Figure 3.4: Reference inverse-dynamics effort profiles.

The torque requirements are derived from the accepted 1 kg analytical reference. The limits are values configured in the Isaac Sim model and are not independently verified hardware motor-torque ratings.

The shoulder-lift joint exhibits the largest predicted torque demand. This behavior is physically consistent with the importance of gravitational loading and the moment arms created by the manipulator configuration. However, the peak torque cannot be interpreted as a purely static gravitational load. The inverse-dynamics calculation includes acceleration-dependent inertia and velocity-dependent coupling, and the time law does not necessarily impose zero joint acceleration at the transportation endpoints. The distinction becomes apparent when the traversal duration is changed while preserving the same geometric path. In the preliminary analysis, increasing the transportation duration from 3 s to 4 s reduced the maximum shoulder-lift torque from approximately 62.435 Nm to 60.340 Nm. The peak joint acceleration decreased more strongly than the peak torque. This indicates that slowing the motion reduces its inertial demand but does not eliminate the configuration- and gravity-dependent components of the required effort.

Consequently, simply extending the motion duration does not necessarily reduce the total joint-torque demand in proportion to the reduction in acceleration. This result reinforces the need to evaluate manipulator dynamics together with trajectory timing.

3.3.3 Physical interpretation of payload-dependent effort

Changing the payload mass alters the dynamic parameters of the complete robot–payload system. For the same reference trajectory, the difference in required inverse-dynamics effort between two payload assumptions is

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}} = \boldsymbol{\tau}_{\mathrm{ID}}(t; m_b) - \boldsymbol{\tau}_{\mathrm{ID}}(t; m_a)$$

Expanding this difference yields

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}} = \Delta M\ddot{\mathbf{q}}_r + \Delta\mathbf{c} + \Delta\mathbf{g}$$

The discrepancy is time-varying because the joint configuration, velocities, and accelerations change during the transportation motion. The difference is also joint-dependent. A change in payload mass does not require an identical additional torque at every joint. For the prescribed fixed-orientation transportation trajectory, the shoulder and elbow configurations have substantial influence on the gravitational and inertial effects of the attached payload. Nevertheless, the exact torque discrepancy depends on the complete analytical model and the specified payload parameters, rather than on a general assumption that one joint must always dominate. These relationships are important for the later residual reinforcement-learning investigation. The learned policy is not asked to transport the object without a dynamics model. The nominal controller continues to provide the principal joint effort, while the residual policy attempts to compensate for effects associated with the mismatch between the assumed and actual dynamics.

The inverse-dynamics discrepancy provides a physically interpretable reference for examining learned torque corrections. However, the theoretical reference discrepancy is not identical to the optimal residual action in a closed-loop system with finite tracking errors, actuator constraints, and discrete-time execution. The distinction is maintained in the subsequent analysis of learning behavior.

3.4 Experimental Conditions and Payload-Mismatch Design

The final experimental conditions were selected to investigate a practical industrial control problem: whether a robotic controller developed for one payload configuration can remain useful when the same transportation task is performed with different loads. In manufacturing and material-handling applications, it is possible for similar operations to involve objects of different masses. One approach is to identify the payload parameters and redesign or retune the controller for each condition. However, repeated controller adjustment may increase the engineering and commissioning effort associated with deploying the same robot across related tasks. The present investigation therefore examines an alternative possibility: maintaining an established nominal controller while introducing an additional learned correction to compensate for selected differences between the assumed and actual payload dynamics.

This experimental question is not equivalent to claiming that a single controller can safely accommodate every load. The goal is to determine how the nominal control structure and learned compensation behave under explicitly defined payload conditions.

3.4.1 Payload selection and model discrepancy

The manufacturer's November 2023 technical specification identifies 5 kg as the rated maximum payload for the UR5e. The experimental study considers actual payload masses of 1, 3, 4, and 5 kg. These values establish a set of operating conditions within the manufacturer's nominal payload-mass range while allowing lighter, matched, heavier, and intermediate cases to be examined. The selected values should not be interpreted as a complete sampling of the robot's certified payload operating envelope. Hardware payload capacity also depends on the center-of-gravity offset and other application-specific conditions. Two independent mass parameters are defined:

$$m_{\mathrm{actual}} \in \{1, 3, 4, 5\} \text{ kg}$$

When $\Delta m = 0$, the nominal controller and simulated plant use matching payload masses. When $\Delta m > 0$, the physical payload is heavier than the assumed model. When $\Delta m < 0$, the physical payload is lighter than the assumed model. These conditions lead to different dynamic-compensation requirements. For a heavier-than-modeled payload, the nominal inverse dynamics generally underrepresents the additional inertia and gravitational loading associated with the attached object. For a lighter-than-modeled payload, the model may predict compensation corresponding to an object heavier than the one actually transported. The effects are not necessarily symmetric in the resulting closed-loop Cartesian tracking performance, because the compensation discrepancy interacts with reference acceleration, gravity, feedback response, and manipulator configuration. The direction of uncertainty was therefore included as a deliberate experimental variable rather than treated as an incidental change in payload mass.

$$\Delta m = m_{\text{actual}} - m_{\text{model}}$$

3.4.2 Experimental comparison cases

The payload-mismatch study was organized around the conditions listed in Table 3.4.

Table 3.4 — Payload-mismatch experimental design

Model mass	Actual mass	Condition	Purpose
3 kg	3 kg	Matched	Establish nominal tracking behavior
3 kg	5 kg	Heavier than modeled	Investigate nominal-model error and initial residual learning
3 kg	1 kg	Lighter than modeled	Examine policy behavior under the opposite mismatch direction
5 kg	3 kg	Reverse mismatch	Investigate separately learned compensation in the opposite direction
3 kg	4 kg	Intermediate mismatch	Cross-evaluate the policy trained for 3→5 kg
5 kg	4 kg	Intermediate mismatch	Cross-evaluate the policy trained for 5→3 kg

The matched 3 kg condition establishes a reference for evaluating the nominal controller under the payload parameters used in its dynamics model. The 3 kg model / 5 kg actual condition introduces a positive mass discrepancy and serves as the primary environment for the initial residual-learning comparison involving PPO, SAC, and TD3. The 3 kg model / 1 kg actual condition retains the same nominal model but reverses the direction of the mass discrepancy. This case examines whether policies developed under heavier-than-modeled loading remain suitable when the actual payload is lighter than the model assumes. The reverse 5 kg model / 3 kg actual condition investigates whether a separate learning process can generate compensation appropriate to the opposite discrepancy direction.

Finally, the actual 4 kg condition is used to cross-evaluate the two independently trained PPO policies at a mass between their respective model and training payload values. This intermediate experiment is especially relevant to controller reuse. A policy that performs well only at the exact mass used during training may have limited practical usefulness when the payload varies. However, a successful result at one intermediate mass does not establish continuous generalization throughout an entire loading range. The experimental interpretation must remain tied to the conditions actually evaluated.

3.4.3 Maintaining a consistent transportation task

To compare the effects of payload-model discrepancy, the underlying transportation task was kept unchanged across the relevant experiments. The fixed experimental elements include the pick and place coordinates, the tool orientation, the geometric path, the transportation duration, and the accepted reference trajectory. The payload retains its geometric dimensions and nominal attachment relationship to Tool0. The primary experimental changes are introduced through the nominal-model payload mass, the simulated actual payload mass, and the presence or absence of

learned residual torque compensation. This approach reduces the risk of attributing performance differences to changes in reference geometry or task duration rather than to the intended model discrepancy. Importantly, changing the assumed model mass requires consistent recomputation of the corresponding model-based compensation. A controller cannot be described as using a different nominal payload merely by changing a label while retaining feedforward data calculated for the original mass.

Similarly, numerical learning results and ROS 2–PhysX experiments must be distinguished. The numerical environment and physics-based runtime are not necessarily identical in integration, actuator implementation, or state-update timing, even when they refer to the same UR5e geometry. The experimental process therefore follows a hierarchy of evidence:

1. Analytical modeling establishes the relationship between task geometry, payload properties, and predicted dynamic requirements.
2. Numerical controller evaluation examines tracking performance under explicitly defined model assumptions.
3. Residual reinforcement-learning experiments investigate torque correction under prescribed mass discrepancies.
4. ROS 2–PhysX validation evaluates the response of the independently executed rigid-body articulation to the combined torque controller.

These stages are designed to provide complementary evidence rather than interchangeable measurements.

3.4.4 Performance measures and task relevance

The principal task-level performance measure is Cartesian end-effector trajectory tracking. Let

$$\mathbf{p}_r(t), \qquad \hat{\mathbf{p}}(t) = f_p(\mathbf{q}_{\text{measured}}(t))$$

$$\mathbf{e}_p(t) = \hat{\mathbf{p}}(t) - \mathbf{p}_r(t)$$

The aggregate sampled position error is expressed as

$$\text{RMSE}_p = \left(\frac{1}{N} \sum_{k=1}^{N} \|\mathbf{e}_p(t_k)\|^2 \right)^{1/2}$$

This quantity provides a consistent scalar measure of Cartesian tracking accuracy along the evaluated task. However, minimizing this quantity is not the sole engineering objective. The intended transportation task requires an adequately controlled trajectory, acceptable joint effort, predictable dynamic behavior, and practical controller execution. A lower aggregate RMSE does not necessarily imply that the controller exhibits better behavior in every motion phase. For this reason, joint-position errors, velocity errors, commanded torque, residual effort, and signed Cartesian errors are also retained for subsequent analysis. Particular attention is given to the ascending and descending phases, since the two-parabola construction introduces different vertical motion characteristics. The separate evaluation of these phases is intended to identify dynamic effects that could be obscured by a single trajectory-wide RMSE value.

At the same time, an observed reduction in Cartesian RMSE cannot independently establish collision safety, stable grasping, hardware-level actuator compliance, or formal closed-loop stability. Those requirements remain separate engineering considerations and are addressed in the later discussion of physical deployment.

3.4.5 Engineering significance of the experimental design

The transportation benchmark was established through a sequence of connected engineering decisions. The physical characteristics and limitations of the UR5e were examined before defining the workstations. The workspace was then configured to produce meaningful changes in manipulator posture while remaining within the evaluated kinematic constraints. A two-parabola Cartesian trajectory was selected to retain analytical interpretability while introducing distinct ascending, descending, and curvature-transition behavior. The motion was assigned a three-second duration and checked against joint-speed requirements and inverse-dynamics torque demand. Finally, the payload mass was introduced as a controlled variable, allowing the consequences of model uncertainty to be investigated without simultaneously changing the underlying transportation task. The resulting experiment is therefore not simply a robot motion demonstration. It is a physically motivated transportation benchmark intended to examine the relationship between manipulator dynamics, model-based control, and learned compensation under changing payload conditions.

The next chapter considers the selection of a suitable nominal controller for this task. The comparison is guided not by tracking accuracy alone, but by the ability to execute the prescribed motion with interpretable joint effort, practical computation, and a consistent baseline for investigating residual reinforcement learning.

Chapter 4

Nominal Controller Design

4.1 Control Strategy Selection for Task-Oriented Payload Transport

The selection of a robotic controller should begin with the requirements of the intended task rather than with the assumption that a more sophisticated control algorithm will necessarily produce a better engineering solution. Industrial manipulators have been controlled using a broad range of methods, including conventional PID feedback, nonlinear dynamics compensation, optimal state feedback, adaptive and robust control, and model predictive control (MPC). These approaches differ in their use of mathematical models, treatment of uncertainty, computational requirements, and ability to accommodate operational constraints. The historical development of robotic manipulator control demonstrates how the mathematical structure of robot dynamics enabled increasingly sophisticated model-based, robust, and adaptive methods. Among these approaches, computed-torque control has a long-established theoretical and experimental foundation. Real-time studies of nonlinear feedback and feedforward dynamics compensation have also highlighted the importance of incorporating the coupled manipulator inertia matrix into controller design.

MPC represents a different design philosophy. Rather than relying only on the current tracking error and a prescribed feedback law, MPC uses a prediction model to determine control actions through finite-horizon optimization. Its principal attraction is the ability to incorporate multiple performance objectives and operating constraints into a common optimization framework. However, predictive control also introduces computational requirements and sensitivity to the selected prediction model, horizon, and objective-function parameters. The existence of these alternatives does not imply that they should all be evaluated in every robotic application. Controller selection is an engineering decision involving the task, available physical information, expected disturbances, implementation platform, and required performance.

4.1.1 Control objectives of the transportation task

The task developed in Chapter 3 requires the UR5e to transport a rigidly attached payload along a specified three-second Cartesian trajectory. The robot must execute coordinated six-joint motion while accommodating the configuration-dependent inertia, gravitational loading, and acceleration demands associated with the transportation path. The primary objective is successful and predictable transportation with acceptable Cartesian tracking, rather than the pursuit of the smallest achievable numerical error irrespective of implementation cost. This distinction should not be interpreted as suggesting that precision is unimportant. Tracking accuracy remains a principal quantitative measure because unexpected deviations may affect

the transported object's position relative to its intended path and surrounding equipment. Nevertheless, transportation differs from tasks such as precision insertion, where a small positioning error can immediately determine success or failure. For the free-space transportation phase investigated here, controller quality must be evaluated using a broader set of engineering considerations.

These include whether the controller can follow the required motion, maintain joint effort within the adopted simulation limits, avoid undesirable terminal motion, and execute reliably through the available software and physics interfaces. The nominal controller must also provide a suitable foundation for the subsequent study of payload uncertainty. This final requirement is particularly important. The purpose of the broader investigation is not to determine which controller produces the lowest error under one accurately modeled payload condition. It is to investigate whether a learned torque correction can extend the usefulness of an established model-based controller when the actual payload differs from its assumed dynamic model. The nominal controller should therefore have sufficient task-level accuracy and a sufficiently transparent structure to distinguish its own dynamics compensation from the additional learned action.

4.1.2 Why computed-torque PD and MPC?

Two controller families were selected for detailed study: computed-torque PD and linearized MPC. Computed-torque PD was considered because the analytical UR5e dynamics had already been reconstructed and validated in the preceding modeling stage. The controller can therefore exploit the available inertia, velocity-dependent, and gravitational terms directly. Its principal advantage for the present investigation is the explicit relationship between the assumed dynamics, the tracking-error feedback, and the resulting joint torque. The method does not require repeated numerical optimization at every feedback update, and its torque components can be examined separately. This transparency is particularly useful when an additional residual policy is introduced. The learned torque can be distinguished from the model-based feedforward and feedback components, making its physical behavior easier to analyze.

MPC was selected as a contrasting approach because it explicitly considers predicted future tracking behavior and imposed effort constraints. For payload transportation, both properties are relevant. The required torque changes throughout the trajectory, and excessive commands may exceed the available control authority. A predictive optimization framework is therefore a technically meaningful alternative to a fixed computed-torque feedback law. The comparison was intended to examine whether the additional prediction and optimization machinery provided a task-relevant advantage that justified its computational and implementation requirements. It was not intended to establish a universal performance ranking between computed-torque control and MPC.

4.1.3 Relationship to the preceding UR5e control study

The selection of these candidates was informed by the preceding UR5e control-analysis project, which had already examined PID, computed-torque control, LQR, and constrained MPC. That earlier benchmark used a four-second quintic joint-space reference and investigated the controllers under common analytical and Isaac Sim conditions. The present investigation addresses a different problem. Its reference is a three-second Cartesian payload-transportation trajectory, converted into coordinated joint motion and evaluated under payload-dependent dynamics. The earlier controller results therefore cannot be transferred directly as a performance ranking for the present task.

However, they provided practical knowledge of controller formulation, numerical integration, torque-command implementation, and simulator synchronization. Rather than repeat an

exhaustive comparison of established controller families, the present study narrowed the nominal-controller investigation to two methods offering different balances between nonlinear dynamic compensation and constrained predictive optimization. The exclusion of PID, LQR, robust control, and adaptive control from the new comparison should not be interpreted as evidence that those methods are unsuitable. In particular, adaptive control provides an established approach to uncertain manipulator and payload parameters, as discussed in Chapter 1. A direct comparison against online parameter adaptation would address a related but distinct research question. The objective here is to evaluate a fixed nominal dynamics model supplemented by learned residual torque, rather than to compare all available methods of model identification and uncertainty compensation.

The candidate set was therefore intentionally restricted to support a controlled, interpretable comparison without expanding the project into a general survey of robot control algorithms.

4.2 Computed-Torque PD: Formulation and Gain Selection

Computed-torque control uses the nonlinear equations of motion to compensate for the predicted dynamics of the manipulator. For an assumed payload mass (m), the model is

$$M(\mathbf{q}; m)\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}; m) + \mathbf{g}(\mathbf{q}; m) = \boldsymbol{\tau}$$

The prescribed Cartesian transportation trajectory is converted into a joint-space reference

$$\mathbf{q}_r(t), \quad \dot{\mathbf{q}}_r(t), \quad \ddot{\mathbf{q}}_r(t)$$

$$\boldsymbol{\tau}_{\mathrm{PD}} = \mathrm{sat}_{\boldsymbol{\tau}_{\max}}[\boldsymbol{\tau}_{\mathrm{ff}}(t; m) + M(\mathbf{q}; m)(K_p\mathbf{e} + K_d\dot{\mathbf{e}})]$$

Here (K_p) and (K_d) are the proportional and derivative gain matrices, represented in the tested implementation using common diagonal gain values across the six joints. The saturation operator applies the adopted joint-effort limits componentwise. The feedforward term compensates for the dynamics predicted along the reference trajectory. The feedback term responds to deviations between the measured and commanded joint states. Multiplication by the configuration-dependent inertia matrix provides a model-based relationship between the acceleration-like feedback correction and the corresponding generalized torque. This arrangement retains the distinction between the reference-dependent model effort and the state-dependent error correction. It should not be confused with the classical inverse-dynamics expression evaluated entirely at the measured state with the commanded reference acceleration included inside the same inertia multiplication. The implemented formulation uses reference-state inverse dynamics for feedforward and the current-state inertia for the feedback contribution.

This distinction matters when interpreting the consequences of payload-model uncertainty.

4.2.1 Nominal error dynamics and modeling assumptions

Under idealized conditions, computed-torque compensation can simplify the coupled nonlinear tracking problem. If the assumed dynamics accurately represent the controlled system, the relevant actuator commands are applied without delay, and the commanded torque remains within the admissible limits, the feedback gains can be interpreted through the behavior of the resulting tracking-error dynamics. For a conventional exact inverse-dynamics PD law, the ideal error equation takes the form

$$\ddot{\mathbf{e}} + K_d\dot{\mathbf{e}} + K_p\mathbf{e} = \mathbf{0}$$

This equation is useful for understanding gain selection, but it should not be presented as an exact global closed-loop description of the implemented controller under all operating conditions. The present implementation uses reference-state feedforward, sampled feedback updates, effort saturation, and an independently executed physics environment. These factors can introduce differences from the ideal continuous-time formulation. In particular, the interaction between reference feedforward and the current-state inertia means that the complete nonlinear error dynamics need not reduce exactly to the simplified expression away from the reference trajectory. Nevertheless, the second-order error-system interpretation provides a systematic starting point for selecting proportional and derivative gains.

4.2.2 Gain parameterization

Rather than selecting (K_p) and (K_d) independently over an arbitrary grid, the preliminary PD search used the parameterization

$$K_p = \omega_n^2, \qquad K_d = 2\zeta\omega_n$$

$$\Delta\mathbf{q}_0 = \begin{bmatrix} 0 & 0.5 & -0.5 & 0 & 0 & 0 \end{bmatrix}^T \text{ deg}$$

4.2.3 Evaluation criteria

The PD candidates were not ranked solely by a single Cartesian RMSE value. The search recorded Cartesian Tool0 RMSE, joint-position RMSE, joint-velocity error, terminal tool-position error, terminal joint-velocity norm, maximum commanded torque, and occurrences of command saturation. These criteria represent different aspects of transportation performance. Cartesian RMSE evaluates how closely the end effector follows the prescribed path. Joint tracking metrics reveal control behavior that may be hidden by a single Cartesian scalar. Terminal position and velocity deserve separate attention because the final state matters when a robot approaches its destination. A controller that produces a slightly lower trajectory-wide RMSE but leaves substantial residual joint motion at the endpoint may not be preferable for a practical transportation task. Similarly, a controller that achieves lower error only by repeatedly reaching the imposed torque limits may offer a less desirable operating margin than its tracking result alone suggests.

The purpose of the screening was therefore to identify candidates with sufficiently good overall transportation behavior for independent physics-based evaluation.

4.2.4 Results of the PD screening

The 25-candidate search produced several useful configurations, including G013, G022, and G025.

Table 4.1 summarizes their recorded performance in the common 3 kg numerical environment.

Table 4.1 — Representative computed-torque PD configurations

Candidate	K_p	K_d	Cartesian RMSE (mm)	Joint RMSE (deg)	Terminal speed norm (deg/s)	Peak torque (Nm)
G013	144	24.0	1.286	0.1268	1.234	82.408
G022	256	27.2	1.099	0.0951	2.091	84.530
G025	256	41.6	1.127	0.0931	1.443	84.530

The lowest Cartesian RMSE among the tested configurations was obtained with G022. However, its minimum trajectory-wide error did not make it unambiguously preferable for

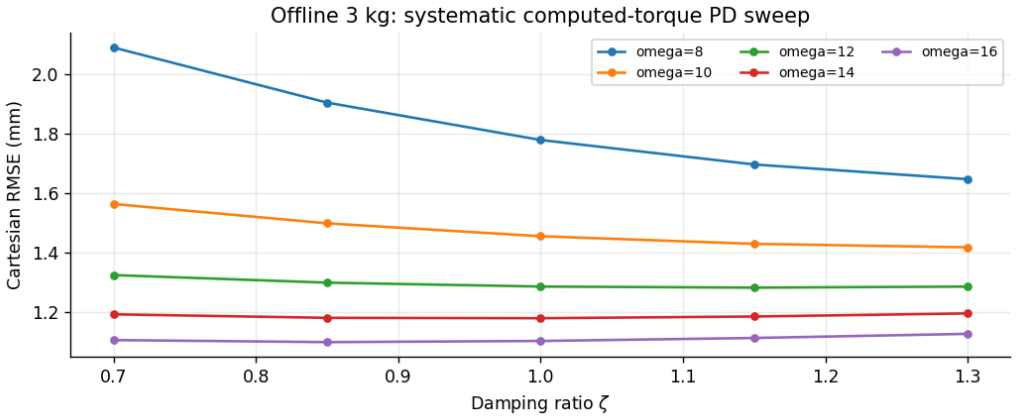


Figure 4.1: Offline computed-torque PD candidate screening.

transportation. G025 used the same proportional gain as G022 but a larger derivative gain. It produced a slightly higher Cartesian RMSE, with a difference of approximately 0.028 mm, while reducing the terminal joint-velocity norm from 2.091 to 1.443 deg/s and slightly improving the aggregate joint-position RMSE. This distinction is important because the objective of the project was not to optimize one numerical measure in isolation. The stronger damping of G025 offered a potentially useful compromise between tracking accuracy and residual motion near the destination. G013 was retained as a lower-gain reference with a clearly distinguishable nominal feedback response.

All 25 offline PD candidates completed the evaluated three-second trajectory without recorded command-limit saturation under the tested conditions. This establishes the behavior of the selected numerical experiments. It does not prove that the corresponding configurations are free from actuator saturation or other limitations under arbitrary loading conditions or physical hardware operation.

The screening therefore produced a shortlist for subsequent physics validation rather than an immediately final controller decision.

4.3 Model Predictive Control: Formulation and Comparative Evaluation

MPC was developed as a genuine competing nominal-controller approach, rather than being excluded on the assumption that a simpler controller would necessarily be more practical. Unlike computed-torque PD, MPC determines its control action through repeated optimization over a future prediction horizon. For the transportation task, this approach is attractive because the robot must execute coordinated joint motion while respecting finite torque limits. MPC can represent the trade-off between tracking error and control effort explicitly and account for the predicted consequences of its present action. The extent of these advantages depends, however, on the actual prediction model and optimization formulation used in the implementation.

4.3.1 Prediction model

The tested controller was based on a local linearization of the nonlinear UR5e dynamics around the prescribed reference trajectory. Define the tracking-error state as

$$\mathbf{x}_k = [\mathbf{q}_k - \mathbf{q}_{r,k},\ \dot{\mathbf{q}}_k - \dot{\mathbf{q}}_{r,k}]^T, \quad \mathbf{x}_{k+1} \approx A_k \mathbf{x}_k + B_k \mathbf{u}_k + \mathbf{d}_k$$

Thus, the prediction model was not reduced to a gravity-only approximation. However, it remained a locally linearized model and should not be described as full nonlinear MPC. The approximation is most reliable in the neighborhood for which its local representation is sufficiently accurate. Deviations from the reference or changes in payload parameters may introduce prediction discrepancies that influence the optimizer's decisions.

4.3.2 Cost function and input constraints

The finite-horizon objective can be written as

$$\min_{\mathbf{u}_0,\ldots,\mathbf{u}_{N-1}} \sum_{k=0}^{N-1} (\mathbf{x}_k^T Q \mathbf{x}_k + \mathbf{u}_k^T R \mathbf{u}_k) + \mathbf{x}_N^T Q_f \mathbf{x}_N$$

However, the constraints included in this implementation are not equivalent to a complete industrial robot safety envelope. Explicit torque limits do not automatically establish collision avoidance, joint-velocity compliance, contact-force safety, or robustness to all modeling uncertainties.

4.3.3 Initial MPC parameter search

The first MPC screening investigated three design quantities:

- Prediction horizon (N).
- State-tracking weight scale (Q).
- Correction-effort weight scale (R).

$$N \in \{5, 10, 15\}, \quad Q_{\text{scale}} \in \{0.5, 1, 2\}, \quad R_{\text{scale}} \in \{0.25, 1\}$$

However, the best result occurred near the edge of the evaluated weighting range. It would therefore have been premature to conclude that the MPC approach was inferior based solely on the first sweep. A further search was justified because the initial parameter grid had not adequately explored the region suggested by its own results.

4.3.4 Second-stage MPC refinement

The second search narrowed the horizon candidates while extending the weighting range toward the region identified by the first sweep. The refinement used

$$N \in \{5, 10\}, \quad Q_{\text{scale}} \in \{2, 4, 8\}, \quad R_{\text{scale}} \in \{0.10, 0.25\}$$

This behavior illustrates the distinction between increasing the numerical importance of an objective and improving the actual closed-loop response. A larger state penalty changes the optimizer's preferred correction, but the resulting response still depends on the approximate prediction model, finite horizon, discrete-time implementation, and control-effort constraint. The second search therefore identified a favorable region within the tested parameter combinations, not a mathematically proven global optimum. Overlapping settings between the two searches produced consistent RMSE values, providing a useful internal check of the numerical screening process.

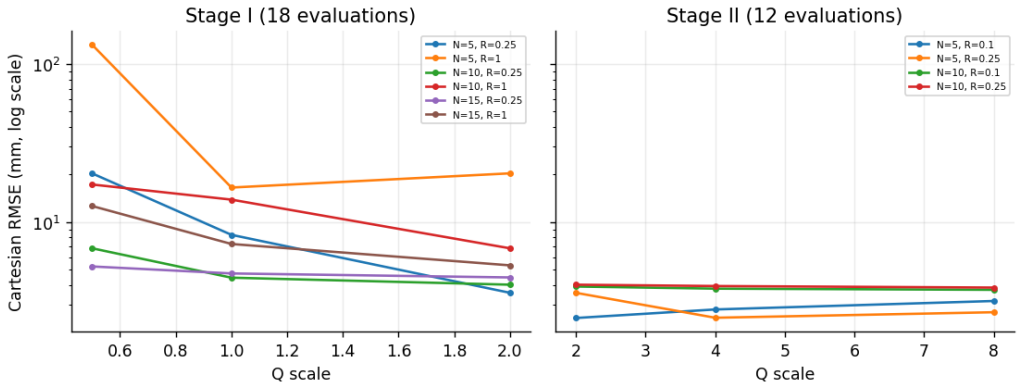


Figure 4.2: Two-stage MPC parameter search.

4.3.5 Comparing MPC candidates by engineering objective

The best MPC configuration depends on the chosen evaluation criterion.

Table 4.2 summarizes the initial winner, the minimum-RMSE refined candidate, and an alternative candidate that produced a slightly different balance of trajectory and terminal behavior.

Table 4.2 — Representative MPC configurations

Candidate	Horizon (N)	(Q) scale	(R) scale	Cartesian RMSE (mm)	Joint RMSE (deg)	Terminal speed norm (deg/s)	Terminal error (mm)
Stage I M005	5	2	0.25	3.566	0.211	0.876	1.314
Stage II M001	5	2	0.10	2.470	0.416	0.852	1.079
Stage II M004	5	4	0.25	2.479	0.337	0.853	1.043

The second-stage M001 configuration achieved the lowest aggregate Cartesian RMSE. However, M004 produced a terminal position error of approximately 1.043 mm compared with 1.079 mm for M001, while its joint-position RMSE was lower. The difference in aggregate Cartesian RMSE between the two candidates was only 0.009 mm. These results demonstrate why a single tracking metric is not sufficient to describe the relative strengths of the tested controllers. For a transportation task, terminal motion and position may be important because the object must approach the destination in a controlled manner. A controller that minimizes trajectory-wide position error does not necessarily minimize residual joint motion or the final Cartesian error. The MPC study was therefore interpreted using the same task-oriented perspective applied to the PD candidates.

4.3.6 Computation time and real-time feasibility

MPC introduces an additional requirement that is less prominent in computed-torque PD: an optimization problem must be solved during controller execution. The relevant question is not simply whether the optimizer converges, but whether the combined controller computation, state acquisition, command publication, and simulator or hardware execution can meet the selected update schedule. For the tested M004 configuration, the recorded quadratic-program computation time averaged approximately 0.779 ms and reached approximately 0.986 ms in the offline screening. These values are below the nominal 20 ms controller-update interval used in the numerical experiments. Consequently, the available evidence does not support describing MPC as computationally infeasible for the task. Indeed, all 30 tested MPC configurations completed the three-second offline trajectory without recorded command-limit saturation.

However, the measured QP solve time is only one component of complete controller execution. A deployed implementation must also account for state retrieval, reference interpolation,

prediction-model preparation, communication, scheduling, and interaction with the physical simulation or actual robot interface. The offline timing measurements therefore establish solver feasibility within the tested environment, not a verified end-to-end hard real-time guarantee. More importantly, the comparison must consider whether the additional prediction model and repeated optimization provide a meaningful practical advantage for this specific transportation task. This is a question of engineering value relative to implementation complexity, not simply processor speed.

4.4 Comparative Assessment and Selection of G025

The computed-torque PD and MPC studies were performed under the same nominal 3 kg payload and prescribed three-second transportation task. Both approaches completed the evaluated reference in the numerical experiments without recorded effort-limit saturation. However, their performance characteristics differed. The lowest recorded offline Cartesian RMSE among the PD candidates was 1.099 mm, obtained by G022. The lowest recorded MPC Cartesian RMSE was 2.470 mm, obtained by the refined M001 configuration. Under the tested conditions, the computed-torque PD implementation therefore achieved lower aggregate Cartesian trajectory error. This result does not imply that computed-torque PD is universally more accurate than MPC. The MPC formulation was locally linearized and evaluated within a finite parameter-search space. Different prediction models, horizons, objective functions, or constraint formulations could produce different results.

The comparison also identified advantages of the tested MPC implementation in terminal-state behavior. For example, the refined MPC candidates produced lower terminal joint-velocity norms than the representative PD candidates. These findings support a narrower conclusion: the two approaches exhibited different trade-offs for the prescribed transportation task, and the additional complexity of the tested MPC did not yield a sufficient aggregate tracking advantage to justify selecting it as the nominal controller for the subsequent learning study.

4.4.1 Selection criteria

The final controller decision was guided by four engineering considerations.

Task-level performance. The nominal controller must execute the prescribed Cartesian transportation trajectory with sufficiently small error for meaningful comparison under changing payload conditions.

Effort and runtime feasibility. The controller must operate within the adopted simulation effort bounds and remain computationally practical for the selected implementation.

Interpretability. The relationship between the assumed dynamics, nominal effort, feedback correction, and resulting robot motion must be understandable from the recorded signals.

Suitability for residual learning. The baseline should permit an additional learned torque to be introduced without obscuring the distinction between model-based compensation and learning-based correction.

These criteria were used to interpret the measured performance. They should not be represented as a previously established numerical pass/fail specification where no such threshold was formally defined.

4.4.2 Why G025 rather than the lowest-RMSE G022?

The offline PD study identified G022 as the minimum-Cartesian-RMSE configuration. However, G025 produced lower terminal joint motion with only a small penalty in aggregate tracking

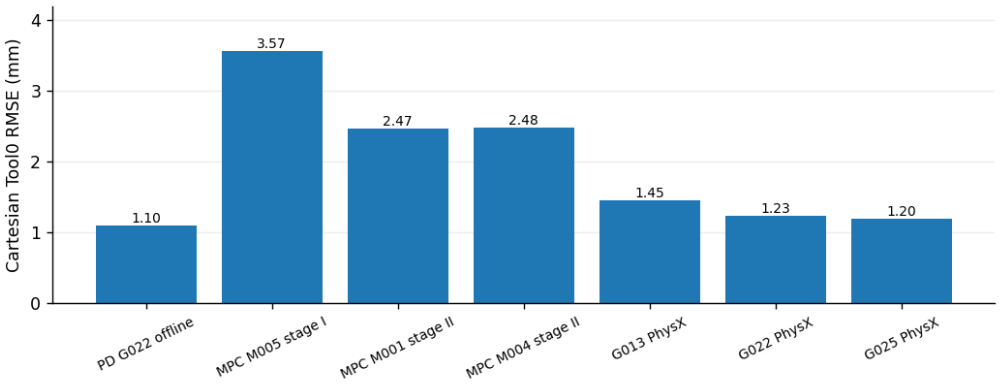


Figure 4.3: Nominal-controller numerical and PhysX performance overview.

accuracy. Both candidates were therefore retained for independent ROS 2–PhysX evaluation, together with the lower-gain G013 reference. This additional validation step was necessary because numerical controller ranking may change when the control law is executed through an independently implemented rigid-body articulation. PhysX introduces differences associated with the simulation integrator, actuator interface, state timing, and effective dynamics. Consequently, the final controller choice should not rely solely on the analytical or offline numerical plant. The three selected PD configurations were transferred to the ROS 2–Isaac Sim effort-control environment without changing the underlying transportation task. Their recorded Cartesian RMSE values were:

Table 4.3 — Nominal 3 kg ROS 2–PhysX PD comparison

Candidate	Cartesian RMSE (mm)	Selection relevance
G013	1.454	Lower-gain reference
G022	1.233	Offline minimum-RMSE candidate
G025	1.195	Best PhysX Cartesian RMSE among the three tested candidates

The ordering differed from that of the offline screening. G022 produced the lowest Cartesian RMSE in the numerical model, but G025 produced the lowest recorded Cartesian RMSE in the PhysX experiments. Because G022 and G025 share the same proportional gain, their principal tuning difference is the derivative gain. The result is consistent with stronger damping being useful when the controller is implemented through a separate discrete-time physics environment. However, the observed improvement cannot be attributed exclusively to damping without independently separating the effects of state timing, integration, friction, initialization, and other model–simulation differences. The evidence supports the practical selection of G025 under the evaluated implementation, rather than a general theoretical claim about its superiority. The actual ROS 2–PhysX execution architecture, stabilization sequence, torque-command timing, and runtime observations are examined in Chapter 5.

4.4.3 Final nominal controller

The selected nominal controller is computed-torque PD with

$$K_p = 256, \qquad K_d = 41.6, \qquad m_{\text{model}} = 3 \text{ kg}$$

The complete results of the 25 computed-torque PD configurations and 30 MPC configurations, including gain parameters, Cartesian and joint-space tracking errors, terminal-state metrics,

computational performance, and torque-constraint evaluations, are provided in Appendix C. The full screening records are retained to support independent examination of the controller-selection process rather than reporting only the selected candidates. The choice also preserves an important methodological distinction. The subsequent reinforcement-learning experiments do not begin with a deliberately inadequate controller that must be rescued by learning. They begin with a nominal controller that has already demonstrated accurate transportation when its model assumptions are consistent with the simulated plant. The question is then whether a bounded learned correction improves the combined system when the actual payload mass differs from the assumed model.

4.4.4 Implications for the subsequent learning study

The selected architecture provides three distinguishable sources of joint effort:

1. Reference inverse-dynamics feedforward calculated from the assumed payload model.
2. State-feedback effort generated by the computed-torque PD controller.
3. Additional residual torque generated by the learned policy.

This decomposition allows subsequent experiments to examine whether learned compensation is associated with the discrepancy between assumed and actual payload dynamics. It also permits evaluation of circumstances in which the learned correction becomes unnecessary or counterproductive. The nominal controller remains active in every learning-augmented experiment. Residual reinforcement learning is therefore investigated as an additional control mechanism rather than a replacement for the established model-based controller. The resulting architecture is consistent with the broader engineering objective: to investigate whether an existing robotic control system can perform related transportation tasks under changing payload conditions without independently redesigning its nominal controller for each load. Chapter 5 establishes how this nominal controller was deployed through ROS 2 and Isaac Sim PhysX, including the practical importance of initialization, gravitational loading, effort-command timing, and synchronization between the control loop and the physical simulation.

Only after those implementation effects have been examined is it meaningful to assess the contribution of residual reinforcement learning.

Chapter 5

ROS 2–Isaac Sim PhysX Validation

5.1 From Analytical Control to an Executed Physics System

The preceding chapters established the kinematic and dynamic representation of the UR5e, developed a physically motivated Cartesian transportation task, and selected computed-torque PD configuration G025 as the nominal controller. These results provided a mathematical basis for controlling the manipulator, but they did not by themselves establish that the same controller would behave as intended when executed through a separate physics engine. This distinction is fundamental to the experimental methodology. A numerical controller can produce mathematically consistent torque commands while the controlled articulation exhibits an unexpected response. The discrepancy may result from the assumed dynamics, but it may also originate from initialization, coordinate conventions, actuation settings, command timing, or the interaction between the controller and simulation runtime.

Consequently, transferring a controller from an analytical environment to Isaac Sim should not be treated as a procedural step in which an existing script is connected to a robot model and assumed to work. It is an independent validation problem. The present study employed ROS 2 Humble and NVIDIA Isaac Sim 6.1 to examine the closed-loop behavior of the UR5e under directly commanded joint effort. The objective was to establish an independently executed, physics-based reference system before introducing reinforcement-learning compensation. This process was particularly important because subsequent payload-mismatch experiments would deliberately alter the dynamic properties of the simulated plant. Without a validated nominal implementation, errors associated with command delivery, startup transients, or conflicting internal drives could be mistaken for the effects of payload uncertainty.

5.1.1 Simulation and ROS 2 architecture

The accepted Isaac Sim scene contained the UR5e articulation, a rigidly attached cubic payload, the transportation workspace, and an existing ROS 2 Action Graph. The Action Graph published joint states through the topic

/joint_states

and received joint-effort commands through

/joint_command.

Both topics used the *sensor_msgs/JointState* message type. For incoming commands, the six joint names were retained in the established articulation order. The position and velocity fields were left empty, while the effort field contained the six commanded joint torques. The effort values were connected to the Isaac Sim Articulation Controller's effort-command input. This implementation was intentionally different from sending target joint positions to an internal servo

or trajectory-following drive. The objective was to evaluate the computed-torque controller itself, including its gravity compensation and error-feedback behavior. Accordingly, the angular-drive stiffness and damping settings were explicitly set to zero for the torque-controlled experiment. This prevented an unaccounted internal position-servo action from supplying additional holding or corrective effort that could obscure the contribution of the external controller.

The distinction is important when interpreting the observed robot motion. If an internal position drive remains active, apparent tracking performance may arise from a combination of the nominal controller and the simulator's internal actuator model. In that situation, the recorded external torque alone would not adequately describe the control system. The present implementation therefore sought to make the relationship between externally calculated joint effort and simulated joint response explicit. However, disabling the internal position drive does not make the simulator equivalent to a physical UR5e servo system. The numerical articulation still follows the physics engine's integration, constraint, and actuation conventions. The resulting validation demonstrates the behavior of the selected torque-control architecture in PhysX, not the complete dynamics of the manufacturer's proprietary motor-control system.

5.1.2 Controller and physics time scales

The architecture contained several time scales that must be distinguished. The nominal PhysX integration interval was

$$\Delta t_{\text{phys}} = 0.002 \text{ s}, \quad f_{\text{phys}} = 500 \text{ Hz}, \quad \Delta t_{\text{ctrl}} = 0.020 \text{ s}, \quad f_{\text{ctrl}} = 50 \text{ Hz}$$

$$\boldsymbol{\tau}(t) = \boldsymbol{\tau}_k, \qquad t_k \leq t < t_{k+1}$$

However, this relationship describes the intended multirate control structure, not an experimentally verified guarantee that exactly ten PhysX steps occurred between every ROS callback. The implemented ROS timer and Isaac Sim physics progression were not driven by one shared deterministic scheduling clock. The original validation records show that the joint-state topic arrived at approximately 200–300 Hz, depending on the run. ROS callbacks and simulator time could advance at different effective rates, particularly under variable computational load. Accordingly, the 50 Hz value represents the configured ROS timer frequency. It must not be presented as proof that controller commands were applied at exactly 50 updates per second of simulated time.

Likewise, the manufacturer-specified 500 Hz control frequency and the configured 500 Hz PhysX integration frequency describe different mechanisms. Their numerical equality is a useful contextual reference, but it does not establish equivalence between simulated rigid-body integration and the industrial robot's internal servo controller. The controller frequency was adopted as part of the experimental implementation rather than established through an exhaustive frequency-optimization study. A systematic claim that 50 Hz is optimal would require additional experiments comparing tracking, latency, stability margin, command variation, and computational load across multiple update rates. No such frequency-sensitivity study is claimed here.

5.1.3 Why the controller frequency matters

The effect of discrete control updates can be understood from the relationship between the nominal commanded torque and the actual continuously evolving joint state. The computed-torque command depends on the measured state:

$$\boldsymbol{\tau}_{\mathrm{G025},k} = \boldsymbol{\tau}_{\mathrm{ff}}(t_k; 3) + M(\mathbf{q}_k; 3)[K_p(\mathbf{q}_{r,k} - \mathbf{q}_k) + K_d(\dot{\mathbf{q}}_{r,k} - \dot{\mathbf{q}}_k)]$$

The controller therefore uses the state available at its calculation instant. During the interval before the next update, the physical simulation continues to evolve while the most recently issued command remains in effect according to the actuation interface. If the reference or actual state changes significantly during that interval, the held torque no longer corresponds to the instantaneous feedback correction that would be calculated by a continuous-time controller. The difference becomes especially relevant during rapid changes in reference acceleration, gravitational loading, or tracking error. A smaller controller interval may reduce this discrepancy, but it also increases command-generation and communication requirements. A larger interval reduces the update burden while increasing the time between feedback corrections. This is the engineering trade-off that motivates separating physics integration from feedback-controller execution.

It is also why the numerical learning environment and the ROS 2–PhysX runtime are treated as distinct validation environments even when they share the same nominal integration and controller intervals. In the numerical environment, the specified step sequence can be executed deterministically. In ROS 2–PhysX, scheduling and state arrival introduce additional timing considerations. The distinction must remain visible when comparing tracking results.

5.1.4 Command generation and actuation evidence

A ROS topic being available does not demonstrate that its values are being applied to the articulation as intended. The software interface must be verified at the level of actual motion and effort response. The validation therefore distinguished between the calculated controller command, the value published to the ROS topic, and the physical response of the simulated joints. In the accepted configuration, the six-element effort command was delivered to the articulation effort input, and the robot exhibited a closed-loop holding response that changed with the configured drive conditions. This provided practical evidence that the external torque command influenced the articulation. Nevertheless, the study did not establish independent measurement of actual motor current or manufacturer-defined applied actuator torque.

The recorded signal represents commanded generalized effort and the observed simulated response. Differences between the command and the effective simulated actuation remain possible through timing, internal physics behavior, and the articulation interface. Consequently, subsequent controller evaluation was based on the combination of torque records and joint-state motion rather than on the numerical correctness of the torque equation alone.

5.2 Startup Gravity Sag and Deterministic Controller Initialization

The most significant commissioning problem encountered during the initial physics-based implementation was unexpected movement before effective external torque control had been established. When the simulation was started with torque-mode drives disabled and no useful holding effort yet applied, the manipulator moved under gravitational loading. In an initial trial, the robot departed substantially from the intended configuration, with joint-position deviations exceeding 100 degrees. This behavior was not a minor numerical discrepancy. It represented a failure to establish a valid initial condition for the transportation experiment. The underlying issue was physical and procedural. The manipulator had gravitational loads acting on its links and payload, but the controller intended to counteract those loads was not yet active. Once the robot had moved away from its initial configuration, a subsequent controller command could

not reasonably be evaluated as though the transportation task had begun from the prescribed starting state.

5.2.1 Why passive settling was insufficient

One apparent solution was to wait until the robot stopped moving before beginning the trajectory. This did not address the underlying problem. A small measured joint velocity only indicates that the robot is moving slowly at that instant. It does not establish that the intended joint configuration has been recovered. A manipulator may settle into an unintended equilibrium because of contact, joint constraints, numerical damping, or other effects. The development record also showed that temporarily enabling a position-holding arrangement could produce a visibly sagged configuration rather than the intended initial pose. More importantly, waiting for the manipulator to return to its commanded initial configuration without applying corrective torque creates a circular dependency: the desired condition cannot be reached because the mechanism needed to produce it has not been activated.

The solution therefore required a controlled initialization sequence, not a longer arbitrary waiting period.

5.2.2 Gravity preload and active return

The corrected procedure began before the physics simulation entered its running state. The rigid payload mass was assigned consistently to the accepted 3 kg nominal validation configuration, and the six initial joint positions were authored in the USD stage. The ROS 2 controller node was launched before starting Isaac Sim playback. At startup, before valid joint-state feedback had arrived, the controller published an initial gravity-compensation effort based on the intended starting configuration. This preload was intended to prevent the uncompensated manipulator from immediately accelerating under gravity while the feedback interface was becoming available. Upon receiving the first valid six-joint state, the controller transitioned to full computed-torque PD correction toward the designated initial configuration. This was an active recovery phase. It was not treated as part of the scored transportation trajectory.

The distinction is important because startup recovery and task tracking serve different purposes. The former establishes a reproducible initial condition, while the latter evaluates the controller's response to the prescribed three-second reference. The corrected procedure therefore separated recovery from experimental evaluation.

5.2.3 Stability gate

After the manipulator returned toward its intended starting configuration, trajectory tracking remained disabled until a defined stabilization condition had been satisfied. The recorded initialization logic required an explicit start request and a continuous stability interval of two seconds. The adopted gate conditions are summarized in Table 5.1.

Table 5.1 — Accepted startup-stability conditions

Quantity	Required condition
Maximum absolute initial joint-position deviation	0.05°
Maximum absolute joint speed	0.30°/s
Tool0 height deviation from the intended initial offset pose	1.0 mm
Tool0 horizontal deviation from the intended initial offset pose	2.0 mm
Continuous stable duration	2.0 s of simulator time
Joint-state freshness	Within 0.15 s of receipt, host time

All geometric deviations in the gate refer to the defined initial offset pose, not to the unperturbed reference trajectory. This qualification is important because the controller experiments used a deliberate initial joint-position perturbation:

$$\Delta \mathbf{q}_0 = [0, 0.5, -0.5, 0, 0, 0]^T \text{ deg}$$

Indeed, the deliberate joint perturbation placed the recorded Cartesian Tool0 position approximately 2.9 mm below the original 5 mm reference Tool0 height when the tracking log began. This offset was part of the test condition and should not be mistaken for a failure of the return controller. At the same time, it illustrates why geometric clearance cannot be inferred solely from satisfaction of a joint-error or relative forward-kinematics gate. The direct distance between the rigid payload and tabletop was not recorded as a separate clearance measurement during these validation runs. Consequently, the logged initialization criteria establish repeatability relative to the specified starting configuration, not an independently verified collision-free payload clearance.

5.2.4 Experimental state machine

The corrected initialization and execution process was represented by the state sequence

$$\text{PRELOAD} \rightarrow \text{RETURN} \rightarrow \text{STABILIZE} \rightarrow \text{TRACKING} \rightarrow \text{FINAL_HOLD}$$

Finally, FINAL_HOLD maintains control at the terminal configuration without extending the interval used for trajectory-performance calculation. This separation makes the execution sequence reproducible and prevents variable startup and terminal-hold durations from contaminating tracking metrics.

The significance of this correction extends beyond eliminating an inconvenient startup transient. A controller should not be evaluated from an uncontrolled initial state when the purpose of the experiment is to compare its response to a prescribed task. The corrected initialization sequence establishes the starting condition as an explicitly controlled experimental variable. This allows subsequent differences in tracking to be interpreted with greater confidence.

5.3 Closed-Loop Validation and Quantitative Results

After the startup procedure and effort-command interface had been established, the shortlisted computed-torque PD configurations were evaluated in ROS 2–PhysX. The purpose of this stage was to determine whether the controller ranking observed in the offline numerical model remained valid when the same nominal dynamics model was applied through an independent simulation runtime. Three candidates were retained:

- G013, representing a lower-gain computed-torque PD controller.

- G022, which achieved the lowest Cartesian RMSE in the offline PD screening.
- G025, which provided stronger damping and favorable terminal-state characteristics.

All three used the same 3 kg nominal payload model and 3 kg simulated actual payload. The transportation path, time parameterization, initial perturbation, ROS topics, effort constraints, and startup conditions were maintained. Only the controller gain settings were changed. This consistency was essential because the objective was to examine the consequences of controller tuning, not simultaneous changes in task definition or simulated loading.

5.3.1 Tracking measurements

The recorded ROS joint states were used to reconstruct the end-effector position through the validated analytical forward-kinematics model. The reconstructed position is defined by

$$\hat{\mathbf{p}}(t) = f_p(\mathbf{q}_{\mathrm{ROS}}(t))$$

$$\mathbf{e}_p(t) = \hat{\mathbf{p}}(t) - \mathbf{p}_r(t)$$

Because the recorded simulator-time samples were not uniformly spaced, a time-weighted Cartesian RMSE was used:

$$\mathrm{RMSE}_p = \left(\frac{1}{T} \int_0^T \|\mathbf{e}_p(t)\|^2 \, dt \right)^{1/2}$$

The numerical implementation evaluates this expression using the recorded simulator timestamps. This metric must be distinguished from an unweighted sample average:

$$\mathrm{RMSE}_{p,\mathrm{sample}} = \left(\frac{1}{N} \sum_{k=1}^{N} \|\mathbf{e}_p(t_k)\|^2 \right)^{1/2}$$

5.3.2 Validation results

The recorded results are summarized in Table 5.2.

Table 5.2 — ROS 2–PhysX validation of the shortlisted PD controllers, nominal 3 kg condition

Metric	G013	G022	G025
K_p	144	256	256
K_d	24.0	27.2	41.6
Cartesian RMSE (mm)	1.4539	1.2335	1.1951
Joint-position RMSE (deg)	0.170	0.125	0.115
Peak Cartesian error (mm)	7.380	7.134	6.823
Terminal Cartesian error (mm)	2.0661	1.7668	1.8619
Terminal joint-speed norm (deg/s)	1.2726	2.1807	1.5132
Peak commanded torque (Nm)	85.966	89.397	89.095
Logged command-limit saturation count	0	0	0

CSV records, including terminal sample	302	286	295

These results demonstrate that all three selected controllers successfully executed the evaluated three-second nominal transportation reference within the adopted simulation command limits. G025 produced the lowest Cartesian RMSE and joint-position RMSE among the tested candidates. Relative to G013, its Cartesian RMSE was lower by approximately 17.8%. Relative to G022, the difference was smaller but still favored G025 in aggregate trajectory accuracy. The result is significant because G022 had achieved the lowest Cartesian RMSE in the offline numerical screening. The controller ranking therefore changed when the computation was transferred to the ROS 2–PhysX execution environment. This observation supports the need for independent physics validation rather than selecting the nominal controller solely from offline optimization.

However, it does not establish why the ranking changed. Differences in state timing, integration, effective rigid-body dynamics, feedback update behavior, or initialization could influence the relative performance of the controllers. The stronger derivative gain of G025 is consistent with improved damping of tracking transients, but the experiment did not isolate damping as the unique causal mechanism behind the observed improvement.

5.3.3 Interpreting the tracking error

The maximum Cartesian errors occurred near the beginning of the recorded tracking interval. The peak deviations of approximately 6.8–7.4 mm are consistent with the deliberate initial joint perturbation and the transition from the stabilized offset configuration to the unperturbed reference. These maximum values should therefore not be interpreted as steady-state tracking error over the entire transportation task. The central portion of the reference produced substantially smaller tracking deviations, generally on the millimeter scale. Localized changes in error were also observed around the parabolic curvature transition and during the final approach. These observations are physically relevant because the prescribed trajectory contains changing acceleration requirements and configuration-dependent loading. Nevertheless, a local change in tracking error near the curvature transition does not by itself demonstrate actuator resonance or an unstable control mode.

Identifying resonance would require appropriate dynamic analysis, including time-aligned torque and joint-motion records and, where necessary, frequency-domain evidence. Similarly, oscillatory behavior during startup may be caused by missing gravity compensation or inconsistent actuation rather than by inadequate tuning of the nominal tracking gains. The physical interpretation must therefore follow the actual experimental condition rather than attributing every observed deviation to the controller algorithm.

5.3.4 Terminal-state trade-offs

G025 did not achieve the best value for every measured quantity. G022 produced a smaller terminal Cartesian position error: 1.7668 mm,

$$e_{p,T}^{\mathrm{G022}} = (1.7668\ \mathrm{mm}), \qquad e_{p,T}^{\mathrm{G025}} = (1.8619\ \mathrm{mm})$$

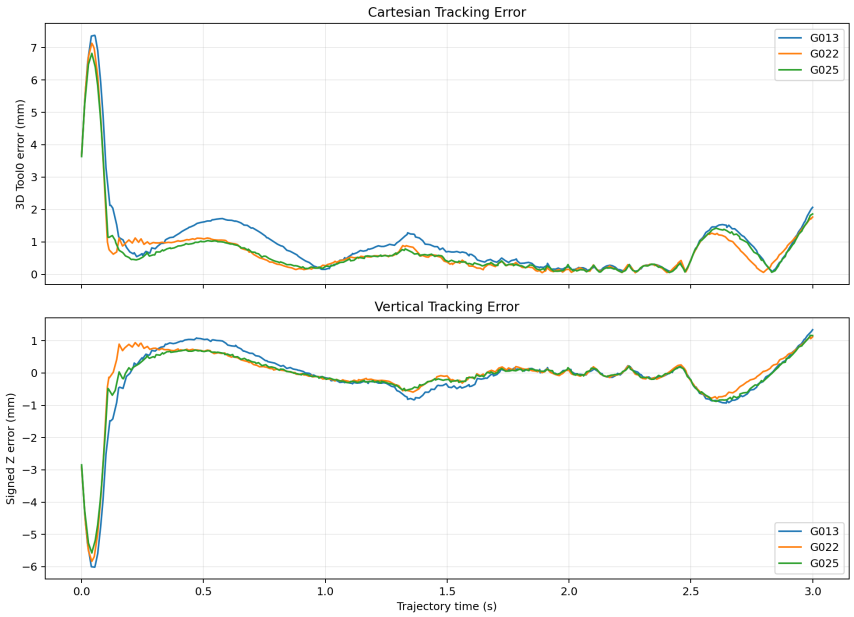


Figure 5.1: PhysX Cartesian tracking errors of nominal controllers.

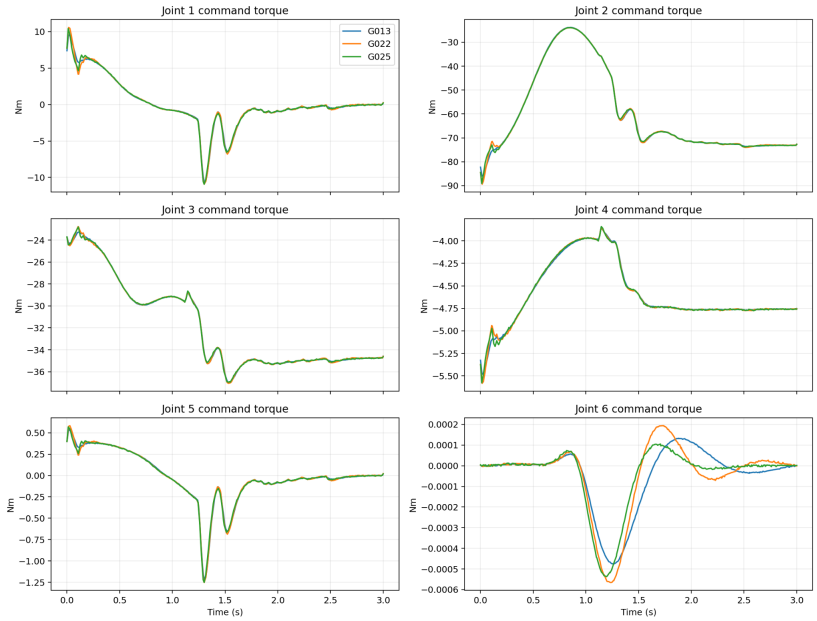


Figure 5.2: PhysX joint-effort histories.

5.4 Validation Boundaries and Implications for Learning-Augmented Control

The physics-based validation established an important prerequisite for the residual-learning investigation: the selected nominal controller could execute the transportation task under a matched 3 kg loading condition when applied through the ROS 2–Isaac Sim effort-control interface. It also exposed practical implementation issues that were not visible in the analytical controller search. The observed gravity sag, startup recovery difficulty, and dependence on the timing of effective torque application demonstrate why physically executed simulation is necessary even when the mathematical dynamics are known.

5.4.1 Distinguishing modeling errors from execution errors

Suppose a simulated manipulator deviates from its reference even though its controller computes an apparently correct inverse-dynamics torque. Several explanations are possible. The dynamic parameters may not represent the articulation accurately. The controller may use a joint order or sign convention inconsistent with the simulated model. The torque command may arrive after the relevant physics update. The robot may begin from an uncontrolled configuration. Alternatively, an unintended internal drive may modify the response. These mechanisms can produce similar external symptoms but require different engineering corrections. Increasing the PD gains without identifying the underlying mechanism may hide an implementation problem or introduce further oscillation. Likewise, applying reinforcement learning to a system whose torque interface is not understood creates a risk that the learned policy will compensate for incidental software behavior rather than for the intended payload-model uncertainty.

The nominal ROS 2–PhysX validation therefore served as a diagnostic separation step. It established the effort-command path, the initialization sequence, and the observable tracking response before additional residual torque was introduced. This does not mean that all model discrepancies were eliminated. The numerical rigid-body model and PhysX remain distinct representations, and their dynamic equivalence was not demonstrated over all possible states. However, the validated baseline provides a controlled operating point from which intentional payload discrepancies can be investigated.

5.4.2 Evidence required for a meaningful learning comparison

The subsequent learning experiments retain the nominal G025 structure:

$$\boldsymbol{\tau}_{\mathrm{nom}} = \boldsymbol{\tau}_{\mathrm{ff}} + \boldsymbol{\tau}_{\mathrm{fb}}$$

A learned residual policy contributes an additional effort:

$$\boldsymbol{\tau}_{\mathrm{cmd}} = \mathrm{sat}_{\boldsymbol{\tau}_{\max}}(\boldsymbol{\tau}_{\mathrm{nom}} + \Delta\boldsymbol{\tau}_{\pi})$$

To evaluate the benefit of the residual term, the comparison must preserve the same task geometry, timing definition, payload assumptions, and nominal-controller configuration. Furthermore, numerical policy-training results cannot be substituted for ROS 2–PhysX validation. A policy trained in the numerical environment may improve the mathematical plant but behave differently after transfer to the independently executed articulation. The resulting behavior depends on both the learned correction and the system through which it is applied. For this reason, the residual-learning study separates training and numerical evaluation from physics-based deployment. The analytical model defines the hypothesized payload-dependent effort discrepancy. The numerical learning environment is used to develop candidate corrections.

PhysX subsequently tests whether those corrections remain beneficial under a different execution mechanism.

5.4.3 Reproducibility and data interpretation

The accepted controller implementation used a defined experimental lifecycle, rather than collecting data continuously before, during, and after the transportation task. The TRACKING state recorded the relevant reference and actual joint states, reconstructed Cartesian positions, effort components, saturation indicators, and simulator timestamps. The final boundary sample at

$$T = 3.0 \text{ s}$$

The complete nominal PhysX comparison and associated state and effort records are retained in Appendix D.

5.4.4 Engineering interpretation

The significance of this validation stage is not confined to the numerical improvement obtained by G025. Its broader contribution is the demonstration that controller development must account for the physical execution process. Analytical modeling identifies the effort a robot should require under specified assumptions. Numerical controller testing examines the response of a chosen mathematical plant. Physics-based execution introduces additional questions concerning initial conditions, effective actuation, feedback timing, and implementation consistency. A dependable control experiment must distinguish these levels of evidence. For the present project, the corrected initialization procedure and executed nominal-controller validation established a meaningful foundation for investigating payload-mass uncertainty.

The next chapter introduces the residual reinforcement-learning problem using G025 as the continuing nominal controller. Learning is directed toward bounded additional joint effort rather than replacing the model-based control architecture that has already been validated.

Chapter 6

Residual Reinforcement Learning Formulation

6.1 Learning-Augmented Control Under Payload-Model Uncertainty

The nominal-control study established that a computed-torque PD controller can execute the prescribed three-second UR5e transportation task with small Cartesian tracking error when the assumed payload model is consistent with the simulated plant. This result provides the starting point for the reinforcement-learning investigation. The motivation for introducing learning is not that the nominal controller is fundamentally incapable of controlling the robot. Rather, it is that the torque compensation calculated from a fixed dynamics model may become inaccurate when the robot transports a payload with different inertial properties. In practical manipulation systems, an existing controller may be configured for a nominal loading condition while the actual object changes from one task to another. Reconstructing or adapting the model for each condition is one possible solution, but it is not the only approach.

The present study investigates whether a learned correction can supplement the fixed nominal controller under a controlled payload discrepancy. The distinction between replacing the controller and augmenting it is central to the research design. A fully learned torque policy would be responsible for generating the complete control action required to support the robot, compensate for gravity, and follow the prescribed reference. Such a policy would need to learn behaviors that are already represented by the established analytical model. A residual policy instead operates alongside the nominal controller. The nominal controller retains responsibility for the principal rigid-body dynamics compensation and tracking-error feedback. Reinforcement learning is restricted to an additional bounded torque command. This architectural separation preserves the model-based controller as an explicit physical reference and permits the learned effort to be examined independently.

6.1.1 Nominal and actual dynamics

The nominal model is parameterized using the assumed payload mass

$$m_{\text{model}} = 3 \text{ kg}, \qquad m_{\text{actual}} = 5 \text{ kg}, \qquad \Delta m = +2 \text{ kg}$$

$$\boldsymbol{\tau}_{\text{G025}} = \boldsymbol{\tau}_{\text{ff}} + \boldsymbol{\tau}_{\text{fb}}$$

The numerical coefficients 256 and 41.6 are the selected G025 feedback gains established through the controller-search and physics-validation process. The feedforward torque for the 3 kg model is recomputed using the corresponding loaded dynamics. It is not obtained by reusing the original 1 kg feasibility-study torque records. This is necessary because the modeled inertia, gravitational loading, and velocity-dependent dynamics change with payload mass.

6.1.2 Physical interpretation of model mismatch

The torque required to follow the reference with the actual payload differs from that predicted by the nominal model. For the same reference states, the analytical inverse-dynamics discrepancy is

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}}(t) = \boldsymbol{\tau}_{\mathrm{ID}}(t; m_{\mathrm{actual}}) - \boldsymbol{\tau}_{\mathrm{ID}}(t; m_{\mathrm{model}})$$

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}} = \Delta M\ddot{\mathbf{q}}_r + \Delta\mathbf{c} + \Delta\mathbf{g}$$

This discrepancy is not a constant torque offset. The inertia contribution depends on the acceleration reference and robot configuration. The velocity-dependent contribution depends on the coupled joint motion. The gravitational contribution depends on the changing configuration of the robot and payload. Consequently, the physical compensation required for a given mass discrepancy varies throughout the transportation task. This is particularly relevant for the two-parabola reference, where the manipulator experiences different configurations and acceleration characteristics during ascending and descending motion. The analytical discrepancy provides a physically meaningful comparison for the subsequent learned residual actions.

However, it is not equivalent to the torque that the policy must reproduce. The nominal feedback controller is already active. Once tracking errors develop, its corrective effort changes, and the actual robot states no longer necessarily coincide with the reference states used in the inverse-dynamics calculation. An effective residual policy may therefore generate commands different from the reference inverse-dynamics discrepancy while still improving closed-loop Cartesian tracking. Conversely, a residual action resembling the analytical discrepancy is not sufficient evidence that the resulting combined controller is stable or effective. The learned action must ultimately be evaluated through the behavior of the complete controlled system.

6.1.3 Residual control architecture

$$\pi_\phi(\mathbf{o}_t)$$

$$\mathbf{a}_t = \pi_\phi(\mathbf{o}_t), \qquad \mathbf{a}_t \in [-1, 1]^6$$

The action is converted into a physical residual torque:

$$\Delta\boldsymbol{\tau}_{\pi,t} = \boldsymbol{\tau}_{\mathrm{res,max}} \odot \mathbf{a}_t$$

where $\odot$ denotes elementwise multiplication. The final commanded joint torque is

$$\boldsymbol{\tau}_{\mathrm{cmd},t} = \mathrm{clip}_{\boldsymbol{\tau}_{\max}}(\boldsymbol{\tau}_{\mathrm{G025},t} + \Delta\boldsymbol{\tau}_{\pi,t})$$

with the adopted full-command bounds

$$\boldsymbol{\tau}_{\max} = [150, 150, 150, 28, 28, 28]^T \ \mathrm{Nm}$$

These values are the configured simulation effort limits, not independently verified UR5e motor ratings. The residual-policy action is additive. It does not replace the nominal gravity compensation, reference feedforward, or tracking-error feedback. This distinction gives the architecture an important practical property: when the residual action is zero, the commanded control law reduces to the selected nominal controller, subject to the same effort saturation. That reference behavior provides a direct experimental baseline for evaluating whether the learned correction offers an additional benefit. However, a zero residual does not imply zero model mismatch. It simply means the policy is not adding torque beyond the nominal controller. Similarly, the architecture does not automatically include a mechanism that recognizes when the learned residual should be deactivated under an unfamiliar payload condition.

Determining when residual compensation remains beneficial is a central issue for the later bidirectional and cross-mass experiments.

6.2 Observation, Action, and Reward Design

The performance of a reinforcement-learning controller depends strongly on how the control problem is formulated. The choice of learning algorithm is only one part of the design. The state information provided to the policy, the action representation, the reward function, and the environment dynamics determine what behavior the policy can learn and what trade-offs it is encouraged to make. For the present task, the objective was to train a bounded corrective controller while preserving the established nominal dynamics-based control structure. The observation and reward definitions were therefore designed around the variables already used in the transportation controller.

6.2.1 Observation vector

The policy receives a 49-dimensional observation constructed from eight six-dimensional vectors and one scalar time variable. The observation is

$$\mathbf{o}_t = [\mathbf{q}_t, \dot{\mathbf{q}}_t, \mathbf{q}_{r,t}, \dot{\mathbf{q}}_{r,t}, \mathbf{e}_{q,t}, \mathbf{e}_{\dot{q},t}, \mathbf{a}_{t-1}, \bar{\boldsymbol{\tau}}_{\mathrm{nom},t-1}, \bar{t}]^T \in \mathbb{R}^{49}$$

$$\mathbf{e}_{q,t} = \mathbf{q}_{r,t} - \mathbf{q}_t$$

$$\mathbf{e}_{\dot{q},t} = \dot{\mathbf{q}}_{r,t} - \dot{\mathbf{q}}_t$$

The previous action and previous nominal torque are represented using the normalization conventions of the numerical environment, and

$$\bar{t} = t/T, \qquad 8 \times 6 + 1 = 49$$

Table 6.1 — Residual-policy observation structure

Component	Dimension	Physical or control significance
Actual joint position $\mathbf{q}_t$	6	Current robot configuration
Actual joint velocity $\dot{\mathbf{q}}_t$	6	Current joint motion
Reference joint position $\mathbf{q}_{r,t}$	6	Desired configuration
Reference joint velocity $\dot{\mathbf{q}}_{r,t}$	6	Desired joint motion

Position error $\mathbf{e}_{q,t}$	6	Configuration tracking discrepancy
Velocity error $\mathbf{e}_{\dot{q},t}$	6	Motion tracking discrepancy
Previous normalized residual action	6	Recent policy command
Previous normalized nominal torque	6	Recent nominal control demand
Normalized task time	1	Progress along prescribed transport
Total	**49**	

The inclusion of both measured and reference states allows the policy to distinguish the robot's current motion from the intended motion. Position and velocity errors make tracking deviations explicit. The previous residual action provides information about the policy's immediately preceding correction and supports the reward penalty on action changes. The previous nominal torque provides context about the effort being requested by the model-based controller. Normalized task time identifies progress along the fixed transportation reference, including the different geometric and dynamic phases established in Chapter 3. This last component is appropriate for the present deterministic trajectory, but it also limits the generality of the formulation. A policy trained with an explicit normalized-time input may learn compensation associated with particular phases of the prescribed motion. Good performance on that trajectory does not establish that the same policy will behave appropriately for a different path or timing schedule.

Similarly, the observation does not include the true payload mass as a direct input. The policy must learn a correction from the resulting dynamic state, reference information, and nominal-control context. The training objective therefore concerns compensation for a specified payload-model discrepancy, not explicit supervised identification of the unknown payload mass.

6.2.2 Residual torque action

The policy output is a normalized continuous six-dimensional vector:

$$\mathbf{a}_t \in [-1, 1]^6$$

$$\boldsymbol{\tau}_{\text{res,max}} = [5, 5, 5, 0.5, 0.5, 0.01]^T \text{ Nm}$$

The resulting learned torque is

$$\Delta\boldsymbol{\tau}_{\pi,t} = \boldsymbol{\tau}_{\text{res,max}} \odot \mathbf{a}_t$$

These limits are intentionally much smaller than the corresponding full simulation effort limits. The purpose is to restrict reinforcement learning to a supplementary control role. The numerical values are also joint dependent. The first three joints are permitted corrections of up to 5 Nm in either direction, wrist 1 and wrist 2 are limited to 0.5 Nm, and wrist 3 is limited to 0.01 Nm. The adopted bounds were not selected through a proof of optimal control authority. During initial exploratory rollouts, larger random actions produced unsuccessful numerical episodes. The final residual limits were reduced to prevent such exploratory commands from dominating the dynamics, particularly at the lightly inertial final joint. All three reinforcement-learning algorithms were evaluated using the same final action bounds.

This history matters because the selected limits influence which compensation behaviors are physically expressible by the policy. If the inverse-dynamics discrepancy at a joint exceeds

its residual bound, the policy cannot reproduce that discrepancy using residual torque alone. However, this does not necessarily imply that the complete controller cannot track the task. The nominal PD feedback remains active and may supply additional effort in response to tracking error. The bounded residual policy therefore modifies the combined closed-loop response rather than acting as an independent inverse-dynamics replacement.

6.2.3 Residual saturation versus total effort saturation

Two different forms of limiting occur in the architecture. The first restricts the learned residual action:

$$|\Delta\tau_{\pi,i}| \leq \tau_{\text{res,max},i}, \qquad |\tau_{\text{cmd},i}| \leq \tau_{\text{max},i}$$

6.2.4 Reward design

The reward function was constructed to prioritize Cartesian trajectory accuracy while retaining weaker penalties for joint-space error, residual effort, and changes in the learned action. The implemented single-step reward is represented by

$$r_t = -\left[\left(\frac{\|\mathbf{p}(\mathbf{q}_t) - \mathbf{p}_r(t)\|}{0.01\ \text{m}}\right)^2 + 0.1\,\text{mean}(\mathbf{e}_{q,t}^2) + 0.01\,\text{mean}(\mathbf{a}_t^2) + 0.01\,\text{mean}((\mathbf{a}_t - \mathbf{a}_{t-1})^2)\right]$$

The third term penalizes the magnitude of the normalized residual action. This discourages the policy from applying unnecessarily large corrections when the nominal controller already provides adequate tracking. The fourth term penalizes changes between consecutive normalized residual actions. Its purpose is to discourage abrupt policy-command variation that could generate undesirable changes in joint effort. However, this term is a soft optimization penalty, not a hard bound on torque slew rate or a guarantee of smooth physical actuation. The weighting coefficients also should not be interpreted as dimensionless universal preferences independent of variable scaling. The relative contribution of each term depends on the normalization and numerical range of its associated variable. The reward was implemented for the specific numerical training environment. Its suitability must therefore be judged through the resulting control behavior rather than by assuming that the selected coefficients guarantee a desired physical response.

6.2.5 Why Cartesian accuracy receives the dominant weight

The reward structure follows directly from the task definition. The UR5e transports an object between two specified spatial locations. The central task requirement is therefore that the attached payload follow the intended Cartesian path. Joint tracking remains important, but it is a means of accomplishing the spatial motion. A reward based only on joint-position error could encourage accurate reproduction of the inverse-kinematics reference without directly representing the Cartesian displacement that matters to the transportation task. Conversely, a reward based only on Cartesian position could allow joint behavior that appears acceptable at the tool level but introduces excessive joint motion or control effort. The selected reward addresses this distinction by making Cartesian position error the principal term while retaining secondary joint and action penalties.

The formulation is deliberately limited to the task investigated. It does not contain a complete collision-avoidance objective, contact-force penalty, grasp-stability condition, or verified hardware

safety constraint. Consequently, maximizing this reward is not equivalent to establishing safe industrial operation.

6.3 Numerical Learning Environment and Algorithm-Screening Protocol

The residual-learning policies were developed in a numerical environment rather than trained directly through the ROS 2–PhysX runtime. This was a deliberate experimental separation. Policy training requires repeated interaction with a dynamic system. In the early stages, exploration can produce large or poorly coordinated actions, even when the action magnitude is bounded. A numerical environment provides a controlled mechanism for evaluating candidate policies, resetting initial conditions, and recording their behavior before exposing the independently executed physics system to learned corrections. The numerical environment was designed to use the same UR5e dynamic formulation, nominal-controller parameters, reference trajectory, and payload-model discrepancy as the intended learning experiment. Nevertheless, it was not assumed to be dynamically identical to PhysX. The distinction between training and independently executed physics validation remains essential to interpreting the results.

6.3.1 Numerical integration and control updates

The numerical plant was integrated with a time step of

$$\Delta t_{\text{plant}} = 0.002 \text{ s}, \qquad \Delta t_{\text{control}} = 0.020 \text{ s}$$

6.3.2 Episode duration and initialization

The prescribed Cartesian transportation trajectory lasts three seconds. At a 20 ms controller interval, each complete evaluation episode contains

$$N = 3.0/0.020 = 150 \text{ steps}$$

A prematurely terminated rollout must not be compared with a complete 150-step episode using only the RMSE computed over its shorter duration, because the two measurements would describe different portions of the transportation task.

6.3.3 PPO, SAC, and TD3 as candidate methods

Three established continuous-action reinforcement-learning algorithms were selected:

- Proximal Policy Optimization (PPO).
- Soft Actor-Critic (SAC).
- Twin Delayed Deep Deterministic Policy Gradient (TD3).

These algorithms were chosen because they offer different learning and policy-update structures while supporting continuous action spaces. PPO is an on-policy policy-gradient method that constrains the magnitude of policy changes through a clipped surrogate objective. Its optimization process uses data collected by the current policy or a closely related rollout policy. SAC is an off-policy actor–critic algorithm based on entropy-regularized reinforcement learning.

Its objective includes both expected reward and policy entropy, encouraging exploration while learning from replayed transitions. TD3 is an off-policy deterministic actor–critic method using twin critics, delayed policy updates, and target-policy smoothing to mitigate certain value-estimation and policy-update instabilities. These differences can influence how efficiently the algorithms learn a bounded corrective action and how their resulting policies distribute torque among the six joints.

However, algorithmic descriptions alone cannot determine which approach is most suitable for the prescribed transportation task. The appropriate comparison is empirical and must use an explicit common evaluation protocol.

6.3.4 Training-budget design

Each algorithm was allocated 30,000 environment interactions using random seed 42. Deterministic evaluation was performed every 3,000 interactions. The purpose of this arrangement was to compare learning progress under a common number of plant interactions. It does not imply that PPO, SAC, and TD3 received equal computational budgets in every other sense. On-policy and off-policy algorithms differ in how frequently their parameters are updated, how they reuse recorded transitions, and how much optimization work occurs between environment interactions. Consequently, equal interaction counts do not imply equal numbers of gradient updates, equal total computation, or equal elapsed training time. The recorded training configurations are summarized below.

Table 6.2 — Initial reinforcement-learning configurations

Parameter	PPO	SAC	TD3
Environment interactions	30,000	30,000	30,000
Random seed	42	42	42
Evaluation interval	3,000 steps	3,000 steps	3,000 steps
Policy optimization device	CPU	CPU	CPU
Learning rate	3×10^{-4}	3×10^{-4}	10^{-3}
Rollout length	300	—	—
Minibatch size	60	256	256
Optimization epochs	5	—	—
Replay capacity	—	30,000	30,000
Learning starts	—	600	600
Entropy coefficient	—	Automatic	—
Policy delay	—	—	2
Exploration noise standard deviation	—	Policy-dependent	0.1, normalized action units

The algorithm configurations were retained throughout the initial comparison so that differences in learning behavior could be interpreted within one defined experimental protocol. The initial experiment involved one random seed per algorithm. Therefore, its results cannot establish statistically robust algorithm rankings or performance variance across independent training runs. The intention was to characterize the observed learning behavior and identify useful candidate policies for subsequent physical-simulator validation. The complete convergence records and selected checkpoint metadata are retained in Appendix E.

6.3.5 Evaluation and policy selection

At each scheduled evaluation point, the current policy was tested deterministically on the prescribed transportation task. The policy was assessed over the full three-second trajectory.

A checkpoint was selected according to its recorded Cartesian tracking performance. This produces an important distinction between the training objective and the reported evaluation metric. During learning, the policy is optimized using the multi-term reward that penalizes Cartesian error, joint tracking deviation, residual effort, and action variation. During checkpoint evaluation, Cartesian RMSE provides the principal task-level performance measure. These quantities are related but not identical. A policy producing a more favorable cumulative reward does not necessarily minimize Cartesian RMSE if it makes different trade-offs among the reward terms. Likewise, a policy with the lowest RMSE does not automatically exhibit the smallest residual effort or smoothest action history.

Therefore, checkpoint selection was accompanied by examination of joint-resolved residual torque and complete trajectory behavior. The best-checkpoint evaluations were performed on the same prescribed reference used during training. They should not be characterized as independent generalization tests. Independent assessment requires evaluation under conditions not used to choose the policy, such as a different payload mass, reversed mismatch direction, or the independently executed PhysX environment. Those tests are addressed in the subsequent chapters.

6.4 Scope, Identifiability, and Physical Interpretation of Learned Compensation

Residual reinforcement learning provides an additional control mechanism, but its meaning must be interpreted carefully. A policy that improves Cartesian tracking under one payload mismatch has not necessarily identified the unknown payload mass or reconstructed the missing dynamics. It may instead have learned a combination of task-specific feedforward correction, feedback-dependent adjustment, and phase-dependent behavior that is effective in the tested closed-loop environment. Understanding this distinction is essential to the subsequent analysis.

6.4.1 Residual torque is not direct parameter identification

The theoretical payload-dependent inverse-dynamics discrepancy can be calculated from the analytical model for a known mass difference:

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}} = \Delta M\ddot{\mathbf{q}}_r + \Delta\mathbf{c} + \Delta\mathbf{g}$$

For a controller carrying a heavier actual payload than assumed, the nominal feedforward effort may be insufficient in some joints or phases of the motion. A residual action can partially address this discrepancy. However, the actual closed-loop effort is

$$\boldsymbol{\tau}_{\mathrm{cmd}} = \boldsymbol{\tau}_{\mathrm{ff}} + \boldsymbol{\tau}_{\mathrm{fb}} + \Delta\boldsymbol{\tau}_{\pi}$$

subject to final effort saturation. Because the nominal feedback already reacts to tracking deviations, the residual policy does not have sole responsibility for compensating the mismatch. The policy is rewarded for improving task performance, not for matching the analytical inverse-dynamics difference. Consequently, a learned action can differ substantially from the theoretical torque discrepancy without being intrinsically incorrect. Conversely, agreement with that discrepancy does not establish closed-loop robustness. The physically meaningful question is whether the additional action improves the behavior of the combined controller under the defined task and loading conditions.

6.4.2 Action limits and compensation authority

The selected residual limits provide an explicit boundary on the policy's contribution. This is valuable because the learned policy cannot issue unrestricted additional torque. Nevertheless, the residual limits do not establish a formal safety guarantee. The learned action is added to an active nominal controller, and the complete system must still satisfy the final effort limits and exhibit acceptable motion. Furthermore, a policy can remain at a residual bound for an extended interval. Such behavior may indicate that the preferred correction exceeds the available residual authority, but it may also reflect the interaction between reward shaping, the nominal feedback controller, and the finite training process. The later analysis therefore distinguishes three quantities:

1. The theoretical inverse-dynamics discrepancy caused by changing the payload model.
2. The torque generated by the learned residual policy.
3. The final torque commanded by the combined controller.

These quantities should be compared through time-aligned records rather than treated as interchangeable descriptions of the same physical effect.

6.4.3 Fixed-mass learning and conditional applicability

The initial numerical learning condition uses a 3 kg nominal model and a 5 kg actual payload. The trained policies are therefore exposed to one specific direction and magnitude of mass discrepancy. Although they observe the robot's state and tracking error, they are not initially trained over a randomized distribution of payloads. This has direct implications for their applicability. A policy may learn a correction appropriate to the added gravitational and inertial loading of the heavier payload. If the actual mass later becomes smaller than the assumed mass, the sign and magnitude of the missing dynamic compensation can change. A correction that was beneficial under the original loading condition may then become unnecessary or counterproductive. The residual architecture does not automatically resolve this issue.

The policy's actual behavior under reversed or intermediate mismatches must be measured. This motivates the bidirectional and cross-evaluation experiments developed later in the study.

6.4.4 Numerical training versus PhysX deployment

The numerical learning environment is useful for controlled policy development, but it is not a substitute for independently executed physics validation. The numerical model and PhysX may differ in their effective inertial properties, integration implementation, joint-drive behavior, state timing, and command application. Furthermore, the numerical environment advances through a prescribed sequence of ten integration steps per controller action, while ROS 2–PhysX execution involves independently scheduled communication and physics updates. A policy that improves tracking in the numerical environment may therefore behave differently when transferred to PhysX. This difference should not immediately be interpreted as failure of the learning algorithm. It may reflect a mismatch between the training environment and the executed physical simulation.

The correct engineering procedure is to preserve the nominal controller and reference conditions, introduce the learned residual under the same defined loading, and examine the resulting joint states, Cartesian tracking, and commanded effort. Only then can the additional contribution of the policy be assessed in the independently executed environment.

6.4.5 Evaluation beyond aggregate RMSE

Cartesian RMSE remains the principal task-level measure because the object is transported through a prescribed spatial path. However, a lower trajectory-wide RMSE is not sufficient to characterize the complete behavior of the learned controller. The policy may improve one portion of the trajectory while degrading another. It may reduce vertical error while increasing horizontal deviation. It may also achieve similar task-level accuracy to another policy using a different distribution of joint torque. For this reason, the analysis retains time-resolved Cartesian error, joint tracking, residual effort, and motion-phase information. The different phases of the two-parabola trajectory are particularly relevant because the loading and acceleration characteristics vary between ascent and descent. A physically interpretable comparison should therefore determine not merely whether the policy improves the overall number, but where the improvement occurs and what corrective actions accompany it.

The selected reward function and action limits support this examination, but they do not guarantee a favorable result. The possibility that residual learning degrades a previously accurate nominal controller remains an essential part of the research question.

6.4.6 Engineering significance of the formulation

The reinforcement-learning problem was constructed around an established model-based controller rather than replacing the robot's analytical dynamics with an unrestricted learned policy. The 49-dimensional observation provides the policy with current and reference motion, tracking errors, recent control information, and trajectory progress. The six-dimensional action represents bounded additional joint torque. The reward gives primary importance to Cartesian task performance while retaining secondary penalties intended to discourage unnecessary joint error, excessive residual effort, and abrupt action variation. The training environment preserves the nominal task and uses deterministic integration and controller updates to permit controlled algorithm comparisons. At the same time, the formulation has explicit limits. It represents a fixed trajectory and a specified training mismatch. It does not identify payload parameters explicitly, does not impose formal stability guarantees, and does not certify collision-safe operation.

Its engineering value must therefore be established experimentally, with attention to the behavior of the combined controller under both the training condition and independently evaluated loading conditions. Chapter 7 examines the numerical learning behavior of PPO, SAC, and TD3, including their convergence histories, checkpoint selection, and joint-resolved residual-torque characteristics. The subsequent physics and cross-mass experiments then evaluate whether the learned corrections remain beneficial beyond the numerical environment in which they were developed.

Chapter 7

Comparative Evaluation of PPO, SAC, and TD3

7.1 Experimental Conditions and Learning Convergence

The residual reinforcement-learning investigation was conducted after establishing the nominal G025 controller and validating its execution through ROS 2–PhysX. The objective was to examine whether a learned torque correction could improve Cartesian payload transportation when the nominal dynamics model underestimated the actual payload mass. The initial training condition used a 3 kg nominal payload model and a 5 kg numerical plant. The same three-second transportation trajectory, computed-torque controller, observation structure, reward definition, and joint-dependent residual action limits were maintained for PPO, SAC, and TD3. Each algorithm received 30,000 environment interactions using random seed 42. Deterministic policy evaluation was performed every 3,000 interactions, with each evaluation covering the complete 150-step transportation episode. This arrangement controlled the number of environment interactions and the task definition. It did not equalize the total number of gradient updates, optimization operations, or computational resources consumed by the algorithms.

The comparison therefore addresses learning performance under a common interaction budget, rather than establishing an algorithm-independent measure of computational efficiency.

7.1.1 Why compare learning progression rather than only final checkpoints?

A reinforcement-learning algorithm may achieve a favorable final score while requiring substantially more interaction to reach useful behavior. Another algorithm may learn rapidly but exhibit later oscillations or sensitivity to additional updates. These differences are relevant when learning is considered as an engineering method rather than an isolated numerical optimization exercise. In the present study, all three algorithms were intended to produce bounded torque corrections to an already functional model-based controller. The desired learning behavior was therefore not simply a reduction in one final error metric; it was progressive identification of useful corrective actions without introducing unnecessary or destabilizing control effort. The deterministic evaluation records are summarized in Table 7.1.

Table 7.1 — Deterministic offline Cartesian RMSE during training

Environment interactions	PPO (mm)	SAC (mm)	TD3 (mm)
3,000	9.870	9.951	8.882

6,000	8.577	9.329	6.744
9,000	7.709	8.053	6.642
12,000	6.969	7.996	6.285
15,000	6.898	7.266	6.121
18,000	6.727	6.943	6.164
21,000	6.464	7.184	6.112
24,000	6.194	7.009	6.231
27,000	6.094	6.934	6.122
30,000	6.105	6.965	6.169

The zero-residual G025 baseline under the same offline 3 kg model / 5 kg plant condition had a Cartesian RMSE of approximately 10.636 mm. Each learned policy eventually improved upon this numerical baseline, but the progression differed. TD3 reduced Cartesian RMSE to 6.744 mm after 6,000 interactions and approximately 6.285 mm by 12,000 interactions. It reached a favorable region substantially earlier than PPO. PPO showed a more progressive reduction over the interaction budget, continuing to improve until approximately 27,000 interactions. SAC also reduced tracking error, but its evaluation curve remained above the best PPO and TD3 results under the selected training allocation. This behavior should be interpreted as observed learning progression, not proof of asymptotic convergence.

A finite sequence of evaluations over 30,000 interactions cannot establish that any algorithm has reached a global or local optimum, nor can it show that further training would be beneficial. The appropriate conclusion is that TD3 achieved favorable tracking with fewer observed environment interactions, while PPO reached the lowest recorded checkpoint RMSE within the available training budget.

7.1.2 Best-checkpoint evaluation

The best saved checkpoints were selected from the scheduled deterministic evaluations. The resulting measurements are shown in Table 7.2.

Table 7.2 — Best-checkpoint offline evaluation, model 3 kg / actual 5 kg

Metric	PPO	SAC	TD3
Selected checkpoint	27,000	27,000	21,000
Cartesian RMSE (mm)	6.0939	6.9339	6.1122
Peak Cartesian error (mm)	7.956	8.488	7.589
Terminal Cartesian error (mm)	5.884	6.285	5.918
Terminal vertical error (mm)	−5.124	−6.254	−5.275
Residual torque RMS (Nm)	2.900	2.916	3.450

The relative reduction in Cartesian RMSE compared with the 10.6358 mm offline baseline was approximately 42.70% for PPO, 34.81% for SAC, and 42.52% for TD3. PPO achieved the lowest observed aggregate RMSE, but its advantage over TD3 was approximately 0.0183 mm. This difference is small relative to the absolute tracking errors and should not be used to claim statistically significant superiority. Only one seed was evaluated, and the checkpoints were selected using the same prescribed trajectory on which the policies had been trained.

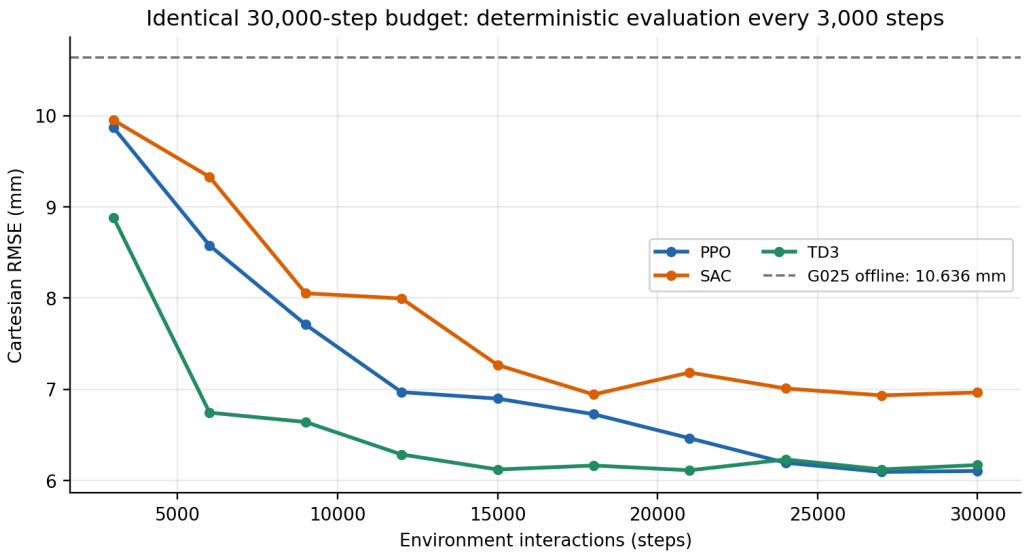


Figure 7.1: PPO, SAC, and TD3 deterministic evaluation curves.

Furthermore, the evaluation metrics reveal that the algorithms were not ordered identically on every criterion. TD3 produced a smaller peak Cartesian error than PPO, whereas PPO produced a slightly smaller aggregate and terminal error. These differences illustrate why one scalar result cannot completely characterize learned control behavior.

The particularly important question is whether similar Cartesian tracking performance was achieved through similar physical torque corrections. The joint-resolved evaluation addresses this question.

7.2 Joint-Resolved Residual Torque and Physical Interpretation

The residual-policy action consists of six additional joint torques, each subject to an independent bound. The nominal G025 controller remains responsible for most of the generalized effort required by the manipulator. Consequently, the learned residual should be interpreted in relation to both the actual payload-model discrepancy and the existing nominal feedback action. A comparison based exclusively on Cartesian RMSE may conceal substantial differences in how the algorithms distribute corrective effort among the robot joints.

7.2.1 Residual torque distributions

The joint-wise RMS residual effort calculated over the 150-step best-checkpoint evaluations is summarized in Table 7.3.

Table 7.3 — Joint-resolved residual torque RMS, offline evaluation

Joint	PPO (Nm)	SAC (Nm)	TD3 (Nm)
J1 — Shoulder pan	0.397	3.790	4.632
J2 — Shoulder lift	5.000	3.620	5.000
J3 — Elbow	5.000	4.824	4.962
J4 — Wrist 1	0.500	0.311	0.496

J5 — Wrist 2	0.195	0.431	0.309
J6 — Wrist 3	0.003	0.006	0.008

The corresponding mean residual torques were:

Table 7.4 — Joint-resolved mean residual torque, offline evaluation

Joint	PPO (Nm)	SAC (Nm)	TD3 (Nm)
J1	+0.150	−1.357	−1.842
J2	−5.000	−2.697	−5.000
J3	−5.000	−4.667	−4.951
J4	−0.500	−0.250	−0.495
J5	−0.051	−0.384	+0.033
J6	+0.002	Approximately 0	+0.002

The similarity between PPO and TD3 in overall Cartesian accuracy does not imply similarity in their learned actions. The most pronounced difference occurs at J1. PPO requests a relatively small shoulder-pan correction, while SAC and TD3 generate much larger J1 residuals. By contrast, J2 and J3 frequently receive large negative corrections across the policies, particularly PPO and TD3. J4 also exhibits substantial negative residual action relative to its smaller allowable range. This pattern invites a physical interpretation based on the dynamic consequences of carrying a payload heavier than the nominal model assumes.

7.2.2 Residual-bound occupancy

The adopted policy action bounds were the configured joint-dependent residual limits.

$$\boldsymbol{\tau}_{\text{res,max}} = [5, 5, 5, 0.5, 0.5, 0.01]^T \text{ Nm}$$

Table 7.5 — Residual action-bound occupancy

Joint	PPO	SAC	TD3
J1	0.0%	23.3%	74.0%
J2	100.0%	35.3%	100.0%
J3	100.0%	90.7%	97.3%
J4	100.0%	0.0%	96.7%
J5	0.0%	24.0%	1.3%
J6	0.0%	8.0%	56.0%

PPO remains at the negative limits of J2, J3, and J4 throughout the deterministic evaluation. TD3 exhibits a closely related pattern at those joints but also approaches the J1 bound frequently. SAC distributes its residual actions differently and does not remain at the J2 and J4 limits for the complete episode. These observations establish differences in learned action allocation. They do not by themselves identify which action distribution is dynamically

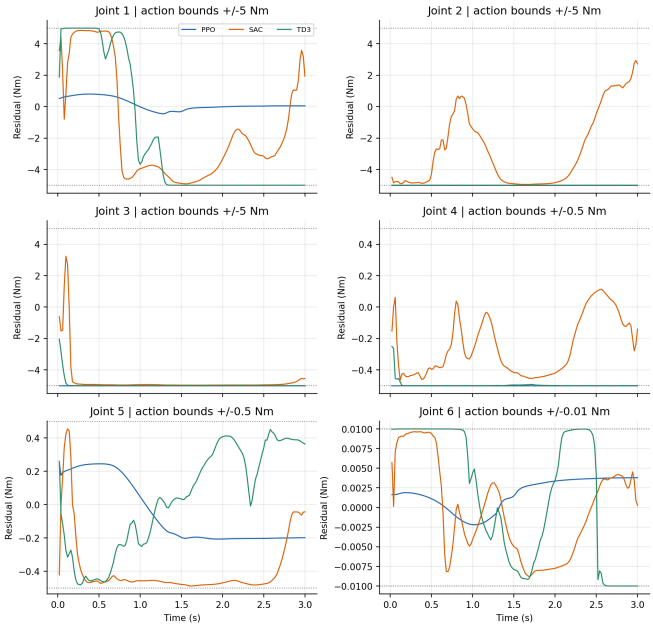


Figure 7.2: Joint-resolved residual effort during offline evaluation.

optimal. A policy may approach an action bound because the physically desirable correction exceeds its authority, because the nominal controller interacts with the policy in a particular way, or because the training objective has developed a saturated solution. The relationship between these possibilities must be examined against the analytical payload-dependent torque discrepancy.

It is also essential to distinguish residual-bound occupancy from final torque-command saturation. The residual action may equal -5 Nm at J2 while the full commanded torque remains below the adopted 150 Nm simulation limit. Thus, persistent residual action at its bound does not necessarily indicate that the simulated actuator has exhausted its complete torque authority.

7.2.3 Analytical comparison with payload-model torque mismatch

The theoretical inverse-dynamics discrepancy between a 5 kg payload and a 3 kg payload can be calculated along the unchanged reference trajectory:

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}}(t) = \boldsymbol{\tau}_{\mathrm{ID}}(t;5) - \boldsymbol{\tau}_{\mathrm{ID}}(t;3)$$

The corresponding expression separates inertial, velocity-dependent, and gravitational differences.

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}} = [M_5 - M_3]\ddot{\mathbf{q}}_r + [\mathbf{c}_5 - \mathbf{c}_3] + [\mathbf{g}_5 - \mathbf{g}_3]$$

Table 7.6 — Analytical 5 kg minus 3 kg reference inverse-dynamics discrepancy

Joint	Mean (Nm)	RMS (Nm)	Range (Nm)	Residual limit (Nm)
J1	-0.007	0.873	-2.815 to $+1.558$	5.0
J2	-13.384	13.653	-15.712 to -8.079	5.0

J3	−8.860	8.924	−10.097 to −6.782	5.0
J4	−1.864	1.869	−2.080 to −1.593	0.5
J5	−0.013	0.168	−0.619 to +0.233	0.5
J6	Approximately 0	Approximately 0	Approximately 0	0.01

These results provide a physical basis for interpreting the dominant residual actions. At J2, J3, and J4, the theoretical inverse-dynamics difference is negative throughout the evaluated reference and its magnitude exceeds the available residual action range. It is therefore physically plausible that policies attempting to compensate the heavier-than-modeled payload would request persistent negative corrections at these joints. The PPO action pattern is consistent with this expected direction. However, the agreement is qualitative rather than a demonstration that the policy has learned the exact missing feedforward torque. The analytical discrepancy was calculated on the reference trajectory. The learned residual is applied to a feedback-controlled system whose measured states deviate from that reference. Moreover, the nominal PD controller already contributes corrective torque in response to tracking error.

Consequently, the residual policy cannot be expected to reproduce the entire theoretical discrepancy. The difference between the theoretical J2 torque deficit and the maximum available −5 Nm residual correction illustrates the importance of this limitation. If the controller required the learned residual alone to reproduce the analytical mismatch, the adopted bound would be insufficient. But the complete closed-loop torque includes feedback compensation, and the actual task-level result depends on the interaction of these effort sources. The relationship is even more revealing at J1. The analytical discrepancy has a mean close to zero and changes sign along the trajectory. Its gravity-only component is zero in the reported calculation, while its total RMS discrepancy is approximately 0.873 Nm. PPO uses relatively little J1 residual action. TD3 and SAC, in contrast, generate much larger shoulder-pan corrections.

Those larger actions cannot be explained as compensation for a persistent J1 gravitational torque deficit. They may reflect feedback interaction, trajectory-coupled dynamics, reward shaping, or a different learned control strategy. Without a dedicated counterfactual experiment, it would be inappropriate to label those actions either physically unnecessary or intrinsically incorrect. Nevertheless, their magnitude provides an important reason to evaluate the policies in an independent physics environment. Two policies that achieve nearly identical numerical Cartesian RMSE while applying substantially different torques may respond differently when model and actuation details change.

7.3 Algorithm Assessment and Selection for PhysX Validation

The offline evaluation established that all three algorithms improved Cartesian tracking over the zero-residual numerical baseline under the tested heavier-than-modeled condition. However, the algorithms differed in learning progression, joint effort allocation, residual-bound occupancy, and secondary tracking metrics. An engineering decision about which policy to deploy first should therefore consider more than the minimum saved-checkpoint RMSE. PPO achieved the lowest recorded offline aggregate error and the smallest overall residual RMS among the three policies. TD3 achieved nearly the same RMSE with fewer environment interactions and showed a different, more aggressive distribution of J1 residual effort. SAC also improved tracking but did not match the lowest PPO and TD3 errors under the 30,000-interaction budget.

At the completion of this offline stage, TD3 was prioritized for initial ROS 2–PhysX evaluation because its learning curve reached a favorable tracking region earlier and its deterministic policy

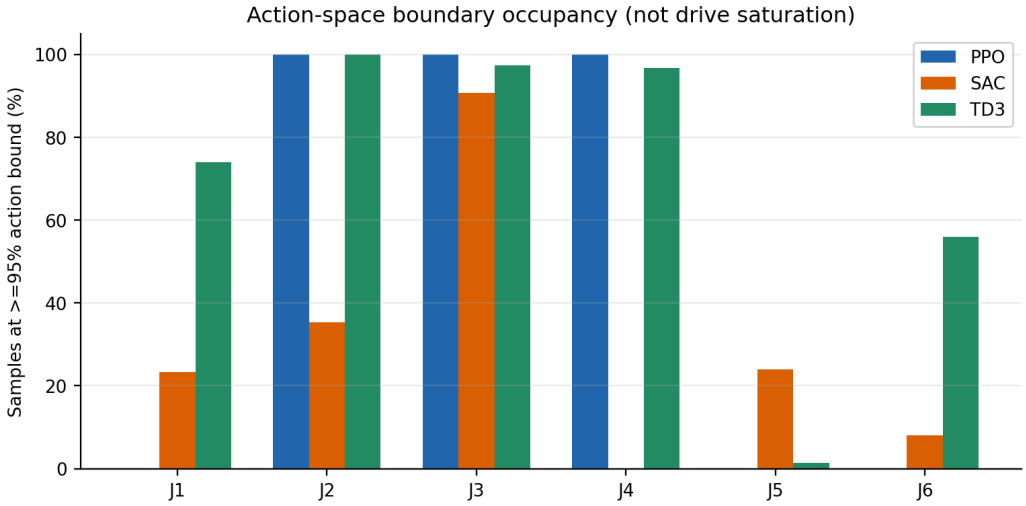


Figure 7.3: Residual action-bound occupancy by joint and algorithm.

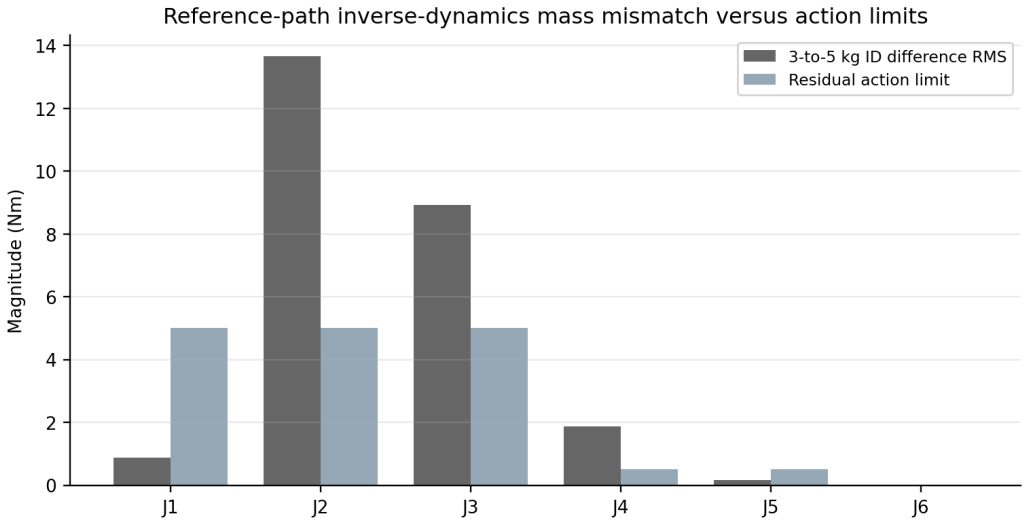


Figure 7.4: Reference inverse-dynamics mismatch and residual action authority.

offered a straightforward repeatable inference path. PPO was retained as a direct comparator. This decision should be interpreted as a provisional experimental selection, not a final conclusion that TD3 was the best controller. The numerical environment provided one level of evidence; the independently executed PhysX system provided another. A later change in policy preference would therefore be consistent with the original research methodology if supported by the physics-based results.

7.3.1 Limits of the offline comparison

Several limitations constrain the conclusions that can be drawn from this stage. First, only one training seed was used for each algorithm. The observed differences do not establish statistically reliable rankings across repeated independent training runs. Second, the checkpoints were selected and evaluated using the same prescribed trajectory. These measurements describe the trained task and do not establish generalization to different trajectories. Third, the initial learning condition contained one fixed payload mismatch. No randomized distribution of payload masses was used during the algorithm-screening experiment. Fourth, the equal environment-interaction budget did not equalize computation time or the number of policy-optimization updates.

Finally, the analytical numerical environment and ROS 2–PhysX do not represent identical execution mechanisms. These limitations do not negate the observed improvements. They determine the scope of the conclusions. The offline results establish that bounded residual torque can improve the numerical transportation response under one specified model discrepancy. They do not yet establish that those policies will provide the same benefit under independent physics execution or under different payload masses. The complete convergence records, joint-resolved residual histories, and saved-checkpoint evaluations are retained in Appendix E. The next chapter evaluates the learned policies in ROS 2–PhysX and examines whether their benefits are preserved when the actual payload mass changes.

Chapter 8

Payload-Mismatch Experiments in ROS 2–PhysX

8.1 Physics-Based Evaluation and Controlled Payload Conditions

The numerical reinforcement-learning study established that bounded residual torque could improve Cartesian tracking under a prescribed heavier-than-modeled payload condition. However, the learning environment used deterministic numerical integration and a controlled sequence of feedback updates. The next experimental question was whether the resulting policies would remain effective when executed through the independently implemented ROS 2–Isaac Sim PhysX system. This validation was necessary for two reasons. First, a policy trained with one numerical dynamics implementation may rely on details of that implementation, including integration behavior, state timing, and the relationship between commanded torque and resulting joint motion.

Second, the practical objective of the research concerns controller reuse under changing payload conditions. A learned correction that improves one training condition may be inappropriate when the nominal model is correct or when the actual load changes in the opposite direction. These considerations motivated a physics-based evaluation using the same G025 nominal controller and the policies developed in Chapter 7.

8.1.1 Maintaining the nominal control architecture

The nominal controller remained computed-torque PD with

$$K_p = 256, \qquad K_d = 41.6, \qquad m_{\text{model}} = 3 \text{ kg}$$

$$\boldsymbol{\tau}_{\text{cmd}} = \text{clip}_{\boldsymbol{\tau}_{\max}}(\boldsymbol{\tau}_{\text{G025}} + \Delta\boldsymbol{\tau}_{\pi})$$

The same prescribed Cartesian trajectory, three-second reference duration, fixed tool orientation, and rigid payload geometry were retained. The purpose was to change the actual payload mass while preserving the model-based control structure and learned-policy parameters. The evaluated conditions were:

$$m_{\text{actual}} \in \{1, 3, 5\} \text{ kg}$$

8.1.2 Measurement consistency

As in the nominal-controller validation, Cartesian tracking was evaluated using the end-effector position reconstructed from the recorded ROS 2 joint states. The time-weighted Cartesian RMSE was

$$\mathrm{RMSE}_p = \left(\frac{1}{T} \int_0^T \|\hat{\mathbf{p}}(t) - \mathbf{p}_r(t)\|^2 dt \right)^{1/2}$$

This maintains the distinction between physics-runtime tracking results and the fixed-step numerical evaluation used for reinforcement learning. Although both environments represent the same transportation task, their RMSE values should not be treated as repeated measurements from one identical simulation implementation. The physics-based comparisons are therefore performed within the ROS 2–PhysX dataset. The principal baseline is the original G025 controller with the learned residual disabled. This comparison directly tests the incremental effect of enabling the saved residual policy without redesigning the nominal controller. The resulting measurements are summarized in Table 8.1.

8.2 Tracking Results Across 1, 3, and 5 kg Payloads

Table 8.1 — PhysX Cartesian RMSE with a fixed 3 kg nominal model

Controller	Actual 1 kg	Actual 3 kg	Actual 5 kg
G025, no residual	**10.920 mm**	**1.195 mm**	10.560 mm
PPO residual	15.508 mm	5.009 mm	**6.061 mm**
TD3 residual	15.270 mm	5.048 mm	6.103 mm
SAC residual	14.818 mm	4.884 mm	6.935 mm

The results reveal a central characteristic of learning-augmented control: the effect of a trained residual policy depends strongly on the operating condition. Under the heavier-than-modeled payload condition used during learning, all three policies reduced Cartesian RMSE relative to the nominal G025 baseline. However, under the matched-mass and lighter-than-modeled conditions, all three policies increased tracking error. The same learned control mechanism was therefore beneficial in one region of the tested payload conditions and detrimental in others. This pattern cannot be adequately summarized by stating that residual reinforcement learning improves model-based control. The more precise conclusion is that the evaluated residual policies learned useful compensation for a particular direction of payload-model discrepancy, but their corrections did not remain beneficial when that discrepancy changed or disappeared.

8.2.1 Heavier-than-modeled payload: 3 kg model, 5 kg actual

When the simulated plant carried a 5 kg payload while the nominal controller retained its 3 kg dynamics model, G025 produced a Cartesian RMSE of 10.560 mm. This represented a substantial degradation relative to the matched 3 kg nominal result. The degradation is physically consistent with the fact that the nominal model no longer represents the actual payload inertia and gravitational loading. For the same reference trajectory, the missing generalized effort varies with joint configuration and motion. The nominal feedback controller can respond to the resulting tracking errors, but its feedback action does not make the original

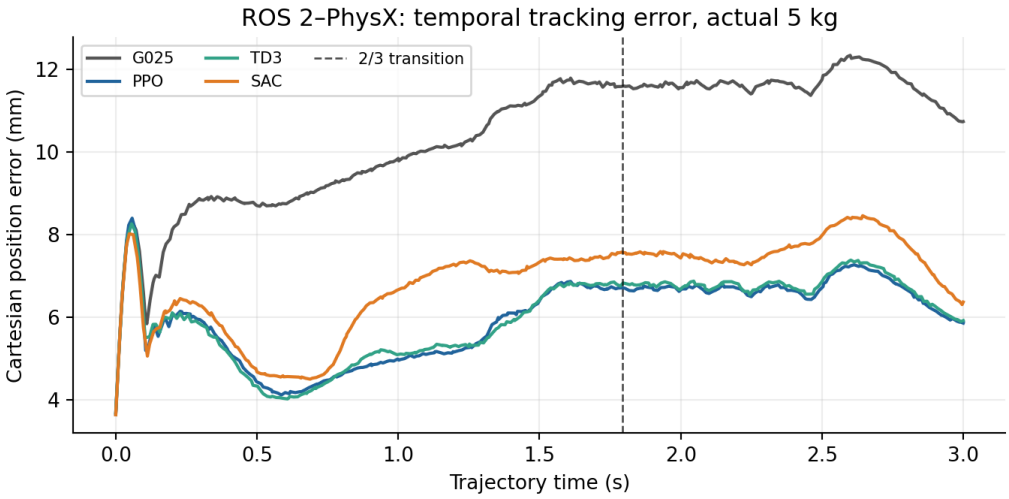


Figure 8.1: PhysX Cartesian error histories for the 5 kg actual payload.

feedforward prediction correct. Introducing the learned residual policies reduced the reported tracking errors are summarized below:

$$\mathrm{RMSE}_{\mathrm{PPO}} = 6.061 \text{ mm}, \ \mathrm{RMSE}_{\mathrm{TD3}} = 6.103 \text{ mm}, \ \mathrm{RMSE}_{\mathrm{SAC}} = 6.935 \text{ mm}$$

These improvements establish that the learned corrections had a beneficial effect under the evaluated heavier-than-modeled condition. However, the remaining tracking errors are not negligible in the context of the original matched-mass performance. PPO reduced the mismatch-induced error substantially, but it did not recover the approximately 1.195 mm RMSE of the original matched 3 kg G025 experiment. This distinction is important. Residual learning was effective as partial compensation under the tested condition, not as a complete restoration of ideal model-consistent tracking. The remaining error may involve insufficient residual action authority, interaction with nominal feedback, numerical-to-PhysX differences, or other factors not separately identified by this experiment. The results do not support assigning the full remaining discrepancy to one cause.

8.2.2 Matched payload: 3 kg model, 3 kg actual

The matched-mass condition provides a particularly informative counterexample to the assumption that adding a learned controller is always beneficial. With the nominal model and actual payload both set to 3 kg, G025 achieved a Cartesian RMSE of 1.195 mm. This was the condition under which the model-based controller had already demonstrated accurate tracking. The learned residual policies, however, produced the following errors are summarized below:

$$\mathrm{RMSE}_{\mathrm{PPO}} = 5.009 \text{ mm}, \ \mathrm{RMSE}_{\mathrm{TD3}} = 5.048 \text{ mm}, \ \mathrm{RMSE}_{\mathrm{SAC}} = 4.884 \text{ mm}$$

The recorded results demonstrate this loss of performance. They do not establish that the policy has no capacity to recognize all changes in loading. Rather, they show that the particular trained policies, observation structure, and learning conditions did not yield appropriate suppression of residual actions in the matched case. This distinction has direct implications for controller deployment. The presence of a trained policy should not be treated as sufficient justification for enabling it continuously. When the nominal model is accurate, the best observed choice among the tested controllers was to leave the learned residual inactive.

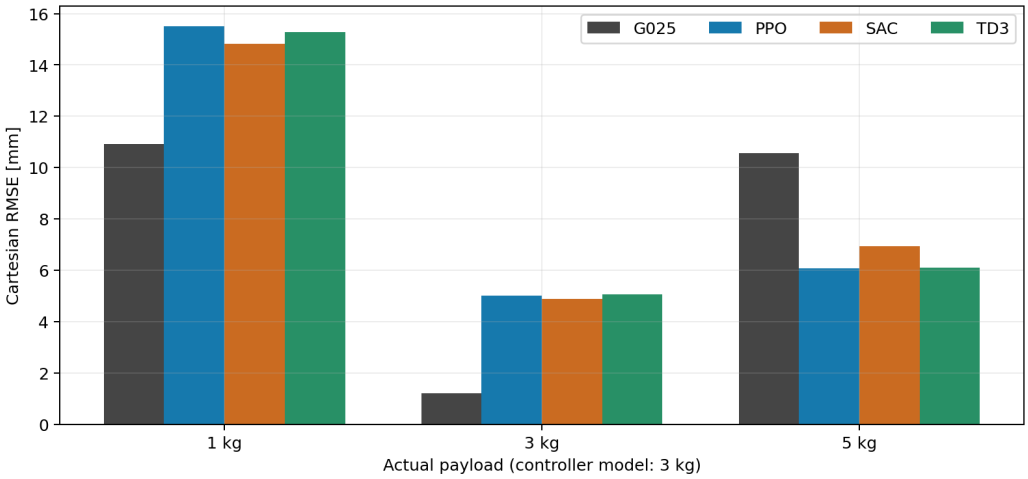


Figure 8.2: PhysX Cartesian RMSE across three actual payload masses.

8.2.3 Lighter-than-modeled payload: 3 kg model, 1 kg actual

The 1 kg actual-payload condition reverses the direction of the mismatch relative to the training environment. The nominal model still assumes a 3 kg payload, but the simulated plant carries only 1 kg. The zero-residual G025 controller produced a Cartesian RMSE of 10.920 mm. The learned policies produced larger errors are summarized below:

$$\mathrm{RMSE}_{\mathrm{PPO}} = 15.508 \text{ mm}, \ \mathrm{RMSE}_{\mathrm{TD3}} = 15.270 \text{ mm}, \ \mathrm{RMSE}_{\mathrm{SAC}} = 14.818 \text{ mm}$$

A policy trained for positive discrepancy may apply corrections in directions that are unsuitable for the negative-discrepancy condition. The increased Cartesian error is consistent with this mechanism. However, the experiment does not provide a complete causal decomposition of the tracking degradation. Establishing the exact contribution of each residual joint action would require time-aligned effort analysis or controlled joint-wise intervention. The measured result nevertheless supports a clear engineering conclusion: equal absolute mass differences do not imply that the same learned compensation will remain effective when the discrepancy changes sign.

8.3 Physical Interpretation and Controller Applicability

The principal value of the payload-sensitivity experiment lies in its contrast between three operating conditions. The nominal G025 controller performs best when its model matches the simulated plant. The learned policies improve the response when the actual payload is heavier than the model used during control. The same policies degrade performance when the payload is matched or lighter than the model. This behavior illustrates the distinction between compensation effectiveness and compensation applicability.

8.3.1 Compensation as a load-dependent action

The theoretical payload-dependent torque difference has the form

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}} = \Delta M \ddot{\mathbf{q}}_r + \Delta\mathbf{c} + \Delta\mathbf{g}$$

For a fixed reference trajectory and unchanged payload geometry, reversing the signed mass discrepancy reverses the corresponding ideal payload contribution difference under the linear rigid-payload model. However, the resulting closed-loop tracking errors do not need to be symmetric. The robot is controlled by nonlinear feedback and a saturated effort command. Its measured states, timing, and effective actuation determine the final motion. Consequently, a +2 kg discrepancy and a −2 kg discrepancy can produce different Cartesian error magnitudes even when the same nominal controller is used. The observed G025 RMSE values of 10.560 mm and 10.920 mm are consistent with comparable but not identical degradation under opposite signed discrepancies. The learned policies alter this relationship further because their compensation is not defined by direct analytical subtraction of the two payload models.

Instead, it emerges from optimizing the reward under one training condition. The policy therefore has no inherent obligation to reverse its output when the actual payload mass changes sign relative to the nominal model. The experimental results demonstrate why this limitation must be evaluated explicitly.

8.3.2 Why the matched-mass result matters

The matched-mass result provides evidence against treating residual learning as a universal improvement mechanism. The nominal controller was already accurate under that condition. A residual policy that remains active without correctly assessing whether additional compensation is needed may introduce unnecessary effort. The resulting tracking degradation is not a contradiction of the original learning experiment. The policy was trained to improve a different operating condition. Its behavior under a matched payload represents a change in the control problem rather than a repetition of the original training task. From an engineering perspective, the appropriate question is therefore not whether the learned controller can be activated, but whether its training assumptions remain applicable. This distinction becomes particularly important in industrial operations where a robot may repeatedly transport objects with changing but related physical properties.

A controller architecture intended for such an environment may ultimately require a mechanism to estimate the relevant loading condition, select a suitable compensation policy, or reduce the residual action when model uncertainty is small. The present experiments do not implement such a mechanism. They establish the need to investigate it.

8.3.3 Why PhysX validation changed the policy preference

During the offline stage, TD3 was prioritized for initial physics evaluation because it improved tracking rapidly and achieved an aggregate RMSE very close to PPO. The subsequent PhysX results introduced an additional selection criterion: performance of the saved policy under an independently executed simulation environment. In the 5 kg actual-payload case, PPO achieved a slightly lower Cartesian RMSE than TD3. The difference was modest, but PPO also exhibited a substantially smaller J1 residual action in the offline joint-resolved comparison. This offered a physically interpretable reason to prefer PPO for the continuation of the investigation, while recognizing that the numerical and physics measurements represent different evidence levels. The choice was not based on an assertion that lower residual magnitude is always better.

Rather, given similar task-level tracking, the PPO policy offered a comparatively restrained shoulder-pan correction and the lowest observed PhysX RMSE under the principal training mismatch. The complete physical behavior of the wrist joints and other secondary signals remains relevant to this decision. The result supports selection of PPO for the subsequent reverse-mismatch and cross-evaluation stages; it does not establish that PPO is universally superior to SAC or TD3.

8.3.4 Limits of the payload-transfer experiment

The experiment evaluates three actual payload masses for a single fixed nominal 3 kg model and a single prescribed transportation task. It does not establish a continuous valid operating interval for any learned policy. The results also do not independently verify real-world actuator torque, collision clearance, or hardware stability. The tested policies were trained with one random seed and one fixed trajectory. Their behavior under changes in path geometry, timing, attachment properties, or external disturbances remains uncharacterized. These limits define the next research question. Rather than increasing the training budget for a policy already shown to be condition dependent, the investigation turns to the direction of the underlying dynamic mismatch. If a policy trained for a heavier-than-modeled payload becomes counterproductive when the actual payload is lighter, can a separately trained residual policy compensate effectively for the reverse condition?

This question is addressed in Chapter 9.

Chapter 9

Bidirectional Payload-Mismatch Learning

9.1 Motivation for Reverse-Mismatch Learning

The payload-transfer results in Chapter 8 demonstrated that the benefit of residual reinforcement learning was conditional on the relationship between the nominal payload model and the actual simulated load. The original PPO policy was trained using a 3 kg nominal model and a 5 kg actual payload. Under that condition, the learned residual reduced Cartesian tracking error relative to the zero-residual nominal controller. However, the same policy degraded performance when the actual payload was lighter than the assumed model. This result suggests that a correction learned under one direction of dynamic uncertainty should not automatically be considered appropriate under the opposite direction.

The next investigation therefore examined whether reversing the payload-model discrepancy and training a separate policy would produce useful compensation for the new operating condition. The objective was not to demonstrate that two independently trained policies solve the general problem of variable-payload control. Rather, the experiment was intended to isolate the physical importance of mismatch direction within the same nominal-controller and residual-learning architecture.

9.1.1 Signed payload-model discrepancy

The signed discrepancy is defined as the actual payload mass minus the nominal model mass.

$$\Delta m = m_{\text{actual}} - m_{\text{model}}$$

However, equal discrepancy magnitude does not imply symmetric closed-loop tracking errors. The two nominal controllers use different payload-mass parameters in their inverse-dynamics calculations, while the feedback system and simulated plant produce configuration-dependent responses. The experiment therefore compares two related but not dynamically identical closed-loop conditions.

9.1.2 Preserving the learning architecture

The reverse-mismatch experiment retained the selected computed-torque PD gains:

$$K_p = 256, \qquad K_d = 41.6$$

The previously trained heavier-mismatch PPO policy was retained rather than overwritten. This preserved two independently learned corrective policies for subsequent comparison. For clarity, the policies are denoted:

$$\pi_A : (m_{\text{model}}, m_{\text{actual,train}}) = (3, 5) \text{ kg}; \quad \pi_B : (5, 3) \text{ kg}$$

9.2 Reverse-Mismatch PhysX Validation

The reverse-mismatch study was evaluated in the ROS 2–PhysX environment using the same prescribed transportation task. The principal comparison was between the 5 kg nominal G025 controller without a learned residual and the same controller augmented by the separately trained reverse-mismatch PPO policy. The actual simulated payload mass was 3 kg. The results are summarized in Table 9.1.

Table 9.1 — Reverse-mismatch ROS 2–PhysX evaluation

Condition	Model mass	Actual mass	Cartesian RMSE
G025, no residual	5 kg	3 kg	8.039 mm
G025 + reverse PPO	5 kg	3 kg	4.585 mm

The learned reverse-mismatch policy reduced the Cartesian RMSE by approximately 43.0% relative to the corresponding zero-residual baseline. This result demonstrates that bounded residual learning can also provide useful compensation when the nominal controller assumes a heavier payload than the simulated plant carries. The improvement is not restricted to the original heavier-than-modeled condition. However, it is conditional on training the policy for the appropriate direction of discrepancy. The earlier heavier-mismatch policies had already shown that correction developed for positive mass discrepancy could be detrimental when applied to a lighter-than-modeled plant. The reverse experiment therefore does not contradict those findings. Instead, it strengthens the interpretation that residual compensation must be evaluated in relation to the sign and physical characteristics of the underlying model error.

9.2.1 Interpretation of the tracking improvement

The nominal 5 kg controller calculates feedforward torque corresponding to a payload heavier than the actual 3 kg object. The resulting dynamic mismatch affects gravitational and motion-dependent effort requirements. Because the controller is closed loop, tracking errors induce feedback corrections. Nevertheless, the model-based feedforward remains based on the 5 kg assumption. The residual policy provides an additional control input that can modify the resulting response. The reduction from 8.039 mm to 4.585 mm demonstrates improved aggregate Cartesian tracking under the evaluated reverse-mismatch condition. It does not demonstrate exact reconstruction of the correct 3 kg dynamics. Nor does it establish that the combined controller would remain superior across all lighter payloads. The observed improvement must therefore be understood as a task- and condition-specific closed-loop result.

The experiment retains particular value because the same nominal-controller architecture, learning algorithm, action structure, and prescribed reference were used in both directions. The difference in policy behavior can thus be examined in relation to the reversed model–plant loading relationship.

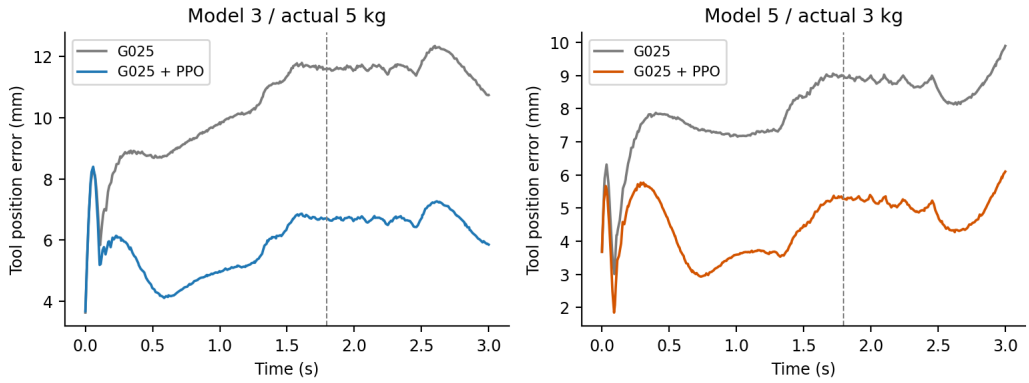


Figure 9.1: Reverse payload-mismatch PhysX tracking results.

9.3 Direction of Learned Torque Compensation

The reverse-mismatch experiment provides an opportunity to examine whether the learned residual torque changes sign in the joints most affected by payload loading. The theoretical reference inverse-dynamics difference for the reverse condition is

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}}^{5\to3}(t) = \boldsymbol{\tau}_{\mathrm{ID}}(t;3) - \boldsymbol{\tau}_{\mathrm{ID}}(t;5)$$

For the original heavier-mismatch condition,

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}}^{3\to5}(t) = \boldsymbol{\tau}_{\mathrm{ID}}(t;5) - \boldsymbol{\tau}_{\mathrm{ID}}(t;3)$$

Therefore,

•

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}}^{5\to3}(t) = -\Delta\boldsymbol{\tau}_{\mathrm{ID}}^{3\to5}(t)$$

when both differences are evaluated using the same reference trajectory and consistent physical parameter definitions. This identity follows directly from reversing the order of the two inverse-dynamics models. It is an analytical property of the reference torque difference, not a claim that the full closed-loop trajectories or learned policy actions must be exact negatives of one another.

9.3.1 Shoulder, elbow, and wrist compensation

The original 3→5 kg analytical comparison identified substantial negative inverse-dynamics differences at J2, J3, and J4. The recorded mean differences were approximately the values shown below:

$$\overline{\Delta\tau}_{J2} = +13.384,\ \overline{\Delta\tau}_{J3} = +8.860,\ \overline{\Delta\tau}_{J4} = +1.864\ \mathrm{Nm}$$

It optimized the tracking reward under the reverse-mismatch numerical environment. The agreement in correction direction therefore supports a physically meaningful interpretation of the learned action, but it does not establish that the network identified the payload mass or reconstructed the exact dynamic parameters.

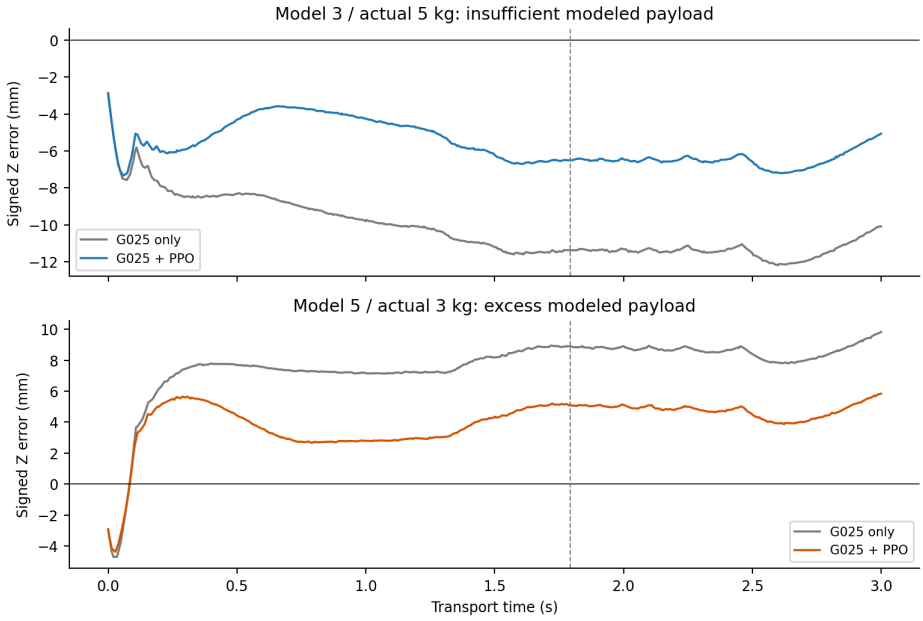


Figure 9.2: Signed vertical-error histories under opposite mismatch directions.

9.3.2 Why learned actions need not be exact negatives

Although the reference inverse-dynamics differences are exact negatives under reversal, the learned residual policies need not satisfy

•

$$\Delta\boldsymbol{\tau}_{\pi_B}(t) \neq -\Delta\boldsymbol{\tau}_{\pi_A}(t) \quad \text{in general}$$

There are several reasons. First, each policy was trained independently. Reinforcement-learning optimization can produce different action distributions even when related control problems have symmetric analytical characteristics. Second, the policies respond to measured joint states and tracking errors, not only to the reference time. Third, the nominal controller itself differs between the two conditions because its assumed payload mass changes. Fourth, the final control action is subject to residual action bounds and full-command saturation. Finally, the numerical and physics-based trajectories may differ because the two closed-loop systems develop different errors. Therefore, exact antisymmetry of learned residual action is neither expected nor necessary for improved task performance. The more meaningful observation is that the dominant correction direction changes in a manner consistent with the reversed payload-dependent torque discrepancy.

9.3.3 The significance of residual action bounds

The analytical mismatch magnitudes at J2, J3, and J4 exceed their respective residual limits in the original 3→5 kg reference calculation. The corresponding reverse differences also exceed those absolute limits. Therefore, the learned residual alone cannot reproduce the entire theoretical reference inverse-dynamics difference at these joints. This does not invalidate the observed improvement. The nominal controller continues to provide error-dependent feedback effort, while the residual modifies the combined control response. The result demonstrates that even a bounded additional action can improve tracking without independently supplying the full

analytical feedforward correction. At the same time, it remains possible that different residual limits would change the achievable performance. The present experiment does not establish optimal residual action bounds, nor does it identify the largest safe correction range.

These would require separate sensitivity and stability investigations.

9.4 Engineering Implications and Transition to Cross-Evaluation

The bidirectional learning results provide two complementary observations. First, a policy trained for a heavier-than-modeled payload improved PhysX tracking under its intended loading condition but degraded performance under other tested masses. Second, a separately trained policy improved tracking under the reverse, lighter-than-modeled condition. These findings support the interpretation that learned compensation is sensitive to the direction of payload-model uncertainty. The relationship is physically meaningful because reversing the mass discrepancy reverses the reference inverse-dynamics torque difference. However, the experiments do not establish that any one policy should be applied continuously across both directions. Instead, they suggest that the applicability of a learned correction is an engineering design variable requiring explicit consideration.

9.4.1 Controller reuse does not imply policy universality

The original motivation for residual learning was to investigate whether an established nominal controller could remain useful when payload conditions change. The bidirectional experiments demonstrate one possible mechanism: retain the G025 control architecture and introduce an additional learned torque policy appropriate to a selected mismatch condition. However, using the same nominal-controller structure is not equivalent to using the same complete controller configuration for every load. In the reverse experiment, the nominal dynamic model was changed to 5 kg and a new PPO residual policy was trained. The result should therefore be described as evidence that the residual-learning architecture can be applied to opposite mismatch directions, not as proof that one unchanged policy generalizes automatically between them. This distinction matters because the broader practical objective is controller reuse with reduced redesign effort.

The present results show the potential usefulness of an additive compensation architecture, while also identifying a limitation that must be addressed before general operation under unknown payload conditions. A future controller may require a means of recognizing the current loading regime and selecting, interpolating, adapting, or suppressing the residual action accordingly. Such mechanisms remain future research directions rather than demonstrated capabilities of the present implementation.

9.4.2 Why evaluate an intermediate payload?

The heavier-mismatch and reverse-mismatch policies were trained under specific actual masses. The next question is whether their behavior remains meaningful at a mass not used as the target plant during either training condition. An intermediate actual payload mass of 4 kg was therefore selected for cross-evaluation. The two policies retain their respective assumed nominal models:

$$m_{\text{actual}} = 4 \text{ kg}, \quad \Delta m_A = +1 \text{ kg}, \quad \Delta m_B = -1 \text{ kg}$$

Because both policies operate through the same transportation reference but use different nominal models and learned corrections, their tracking behavior may differ in ways that are

not captured by one aggregate Cartesian RMSE value. The cross-evaluation is presented in Chapter 10.

9.4.3 Experimental boundaries

The reverse-mismatch result is based on a specified three-second trajectory and an independently trained PPO policy. It establishes an observed improvement in aggregate PhysX Cartesian tracking under that condition. It does not establish repeated-trial statistical reliability, hardware safety, or generalization across arbitrary payloads and trajectories. Likewise, the directional interpretation is supported by the analytical inverse-dynamics relationship and the observed signs of dominant residual corrections. It should not be described as a formal identification of unknown inertial parameters. The engineering conclusion is therefore conditional but important: **the direction of payload-model uncertainty affects the corrective action required, and a residual policy learned for one direction should not be assumed applicable to the opposite direction.** The complete reverse-mismatch validation records and associated joint-effort histories are retained in Appendix F.

Chapter 10

Cross-Evaluation Under an Intermediate Payload

10.1 Experimental Rationale and Cross-Evaluation Design

The preceding experiments established two distinct findings. A residual policy trained for a payload heavier than the nominal model improved Cartesian tracking under its intended loading condition, but degraded performance when the payload discrepancy changed or disappeared. Separately, a PPO policy trained for the reverse discrepancy improved tracking when the nominal controller assumed a heavier payload than the simulated plant actually carried. Together, these results indicate that the effectiveness of learned torque compensation depends on the relationship between the assumed and actual payload dynamics. The remaining question is whether the independently trained policies exhibit useful behavior at a payload mass different from their respective training conditions.

This question is especially relevant to the original engineering motivation of the study. If residual learning is intended to extend the usefulness of existing model-based controllers across related transportation tasks, its applicability cannot be assessed solely by repeating the conditions under which it was trained. An intermediate-mass experiment was therefore introduced to examine whether learned compensation retains useful characteristics away from the original training masses. The selected actual payload mass was

$$m_{\text{actual}} = 4 \text{ kg}$$

10.1.1 Two independently trained policies

The original heavier-mismatch PPO policy is denoted Policy A:

$$\pi_A : (m_{\text{model}}, m_{\text{actual}}) = (3, 4) \text{ kg}, \quad \pi_B : (5, 4) \text{ kg}$$

Policy B was trained to compensate for a plant lighter than its nominal model, and it is now evaluated with a smaller negative discrepancy. In both cases, the discrepancy magnitude is reduced from the original 2 kg training difference to 1 kg. However, the two evaluations are not identical control problems. The nominal dynamics models differ, which changes the model-based feedforward and feedback contributions. The policies were also trained independently and need not generate symmetric residual actions. The comparison therefore examines the performance of two complete learning-augmented controllers at the same actual payload mass, rather than isolating the learned policy as the only changing variable. This distinction is necessary for a valid interpretation of the results.

10.1.2 Common task and measurement conditions

The prescribed transportation task was retained from the preceding chapters. The end effector followed the same three-second Cartesian reference constructed from two parabolic vertical segments, with the same pick and place targets and fixed tool orientation. The nominal controller retained the selected G025 gains,

$$K_p = 256, \quad K_d = 41.6, \quad \boldsymbol{\tau}_{\text{res,max}} = [5, 5, 5, 0.5, 0.5, 0.01]^T \text{ Nm}$$

However, the experiment also examined separate ascent and descent intervals and their vertical tracking characteristics. This phase-resolved analysis was particularly relevant because the reference trajectory was intentionally designed to contain distinct vertical curvature properties. The two motion phases therefore represent different combinations of manipulator configuration, acceleration demand, and gravitational loading. A policy may respond differently to these conditions even when its aggregate tracking performance appears comparable.

10.2 Overall Cartesian Tracking Results

The recorded intermediate-mass PhysX results are summarized in Table 10.1.

Table 10.1 — Intermediate 4 kg cross-evaluation results

Configuration	Nominal model	Actual payload	Mass discrepancy	Cartesian RMSE
Policy A — PPO trained for 3→5 kg	3 kg	4 kg	+1 kg	1.931 mm
Policy B — PPO trained for 5→3 kg	5 kg	4 kg	−1 kg	1.763 mm

Both learning-augmented controllers produced millimeter-scale Cartesian tracking errors under the intermediate actual payload condition. Policy B achieved the lower recorded aggregate RMSE in this evaluation:

$$\text{RMSE}_A = 1.931 \text{ mm}, \quad \text{RMSE}_B = 1.763 \text{ mm}$$

The combined effects of model-based compensation and learned residual action determine the final tracking response.

10.2.1 What the aggregate RMSE establishes

The overall RMSE provides a compact measure of the end-effector deviation from the reference trajectory. Its value is useful for comparing controllers under a common task, but it does not describe how tracking error is distributed along the motion. For example, one policy may produce relatively larger error during ascent and smaller error during descent, while another may exhibit the opposite pattern. These two behaviors can yield similar aggregate Cartesian RMSE values despite different physical responses. The issue is especially relevant when the trajectory contains a deliberate change in vertical curvature. The ascent and descent phases are not simply duplicated motions in opposite directions. They occur at different robot configurations and involve different Cartesian displacements, geometric curvature, joint accelerations, and gravitational moment arms.

Consequently, a single aggregate error is insufficient to determine whether one policy provides uniformly better compensation throughout the transportation task. The cross-evaluation therefore included phase-dependent analysis.

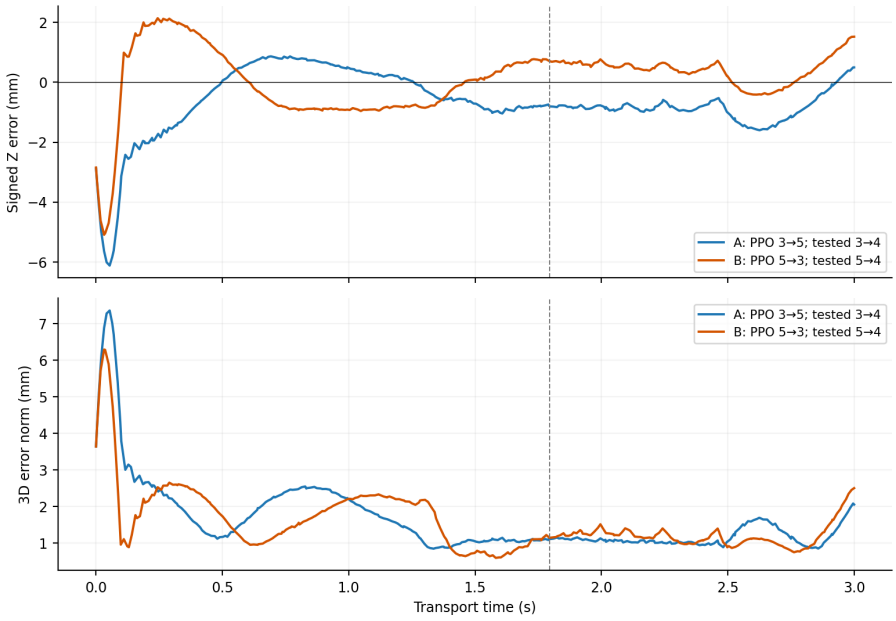


Figure 10.1: Intermediate 4 kg cross-evaluation Cartesian error histories.

10.2.2 Comparison with the original training conditions

The original Policy A training condition involved a 2 kg positive discrepancy, while its intermediate evaluation involved a 1 kg positive discrepancy. Similarly, Policy B was trained for a 2 kg negative discrepancy and evaluated at a 1 kg negative discrepancy. In this sense, the intermediate tests examine policy behavior under a reduced discrepancy magnitude without reversing the sign experienced during training. The recorded tracking errors were substantially smaller than those observed in the corresponding original 2 kg-mismatch PhysX experiments. However, this reduction should not be attributed entirely to policy generalization. A smaller mismatch also reduces the difference between the nominal dynamics and the actual payload dynamics. The baseline model-based controller would therefore be expected to experience a different level of compensation error even without a learned residual.

A complete analysis of the incremental contribution of the policy at 4 kg would require the corresponding zero-residual G025 baseline under each nominal-model condition. The available Chapter 10 comparison establishes the relative performance of the two complete learning-augmented configurations. It should not be presented as proof that either policy improves upon its own zero-residual 4 kg baseline unless that baseline is explicitly evaluated. This distinction is essential when interpreting the practical meaning of the cross-evaluation.

10.3 Ascent–Descent Tracking and Vertical Error Characteristics

The transportation reference was designed using two parabolic vertical segments joined at normalized progress

$$s_j = \frac{2}{3}, \qquad t_j \approx 1.7944 \text{ s}$$

10.3.1 Phase-dependent Cartesian tracking

The recorded phase-dependent Cartesian results are summarized in Table 10.2.

Table 10.2 — Cartesian tracking metrics by transportation phase, actual 4 kg

Metric	Policy A	Policy B
Ascent Cartesian error metric (mm)	1.806	1.740
Descent Cartesian error metric (mm)	**1.189**	1.236

Policy B produced the lower recorded Cartesian error metric during ascent. The difference was 0.066 mm.

$$\Delta\text{RMSE}_{\text{ascent}} = 0.066 \text{ mm}, \quad \Delta\text{RMSE}_{\text{descent}} = 0.047 \text{ mm}$$

The current experiment does not assign such application-specific weights to the phases. It demonstrates that the choice of performance aggregation can influence how the controller comparison is interpreted. The phase-dependent values are retained as their recorded experimental metrics. They should not be recombined into the overall RMSE without verifying the original sample partitions, weighting method, and aggregation definitions.

10.3.2 Vertical tracking error

The Cartesian norm provides no information about the direction of the tracking deviation. For payload transportation, vertical deviation deserves separate attention because the tool and payload must follow a reference that includes both ascent and descent relative to the worktables. Define the signed vertical error as

$$e_z(t) = \hat{z}(t) - z_r(t)$$

where $\hat{z}(t)$ is the forward-kinematics-derived tool height and (z_r(t)) is the prescribed vertical reference. A negative value means that the reconstructed Tool0 position is below the reference, while a positive value means it is above the reference. The original phase-dependent vertical error magnitudes are summarized in Table 10.3.

Table 10.3 — Recorded Z-axis tracking metrics, actual 4 kg

Metric	Policy A	Policy B
Ascent Z-axis error metric (mm)	**0.923**	1.023
Descent Z-axis error metric (mm)	0.946	**0.588**

The vertical-axis comparison reveals a different pattern from the full Cartesian error metrics. During ascent, Policy A exhibited the smaller recorded vertical error magnitude, despite Policy B having the lower overall Cartesian phase metric. During descent, Policy B produced the smaller vertical error magnitude. The ascent difference was 0.100 mm, as summarized below.

$$\Delta\text{RMSE}_{z,\text{ascent}} = 0.100 \text{ mm}, \quad \Delta\text{RMSE}_{z,\text{descent}} = 0.358 \text{ mm}$$

$$\|\mathbf{e}_p\| = \sqrt{e_x^2 + e_y^2 + e_z^2}$$

A reduction in (|e_z|) can therefore be offset by increased deviations in (x) or (y). This is consistent with the observed phase-dependent trade-offs. However, the available aggregate component metrics do not establish the complete time-dependent coupling among axes. That interpretation requires the original signed (x), (y), and (z) error histories.

10.3.3 Signed bias versus error magnitude

An important distinction must be maintained between the magnitude of vertical error and its signed bias. The Z-axis values in Table 10.3 quantify recorded phase-specific error magnitudes; they are not themselves signed mean offsets. A small Z-axis error metric does not reveal whether the tool tends to remain above or below the prescribed reference. Nor does a larger vertical error necessarily imply that the same signed deviation persists throughout the phase. To examine the directional behavior, the signed mean vertical error can be defined as

$$\bar{e}_{z,\mathcal{P}} = \frac{1}{N_{\mathcal{P}}} \sum_{k \in \mathcal{P}} e_z(t_k)$$

where $\mathcal{P}$ denotes the selected ascending or descending evaluation samples. The sign of this measure indicates the average direction of displacement relative to the reference. By contrast, a vertical RMSE is

$$\mathrm{RMSE}_{z,\mathcal{P}} = \left(\frac{1}{N_{\mathcal{P}}} \sum_{k \in \mathcal{P}} e_z(t_k)^2 \right)^{1/2}$$

The latter loses directional information. The two quantities should therefore be reported separately when the original time histories are available. This distinction is particularly relevant in the present study because payload-model mismatch can produce a systematic vertical tracking bias associated with gravitational compensation. A policy trained under positive mass discrepancy may develop a correction that reduces downward deviation, while another trained under negative discrepancy may produce a different vertical response. The recorded phase-specific error magnitudes support a comparison of tracking accuracy, but they do not alone prove a specific signed-bias mechanism. Consequently, the interpretation of signed vertical bias remains dependent on the actual time-aligned (e_z(t)) records.

10.3.4 Relationship to the trajectory geometry

The phase-dependent tracking differences are relevant to the deliberate trajectory construction described in Chapter 3. The ascending segment moves the tool toward a maximum reference height of 0.500 m while the horizontal coordinates change continuously. The descending segment then moves toward the elevated place target. These two portions have different geometric curvature and occur at different joint configurations. The required generalized effort is therefore determined by the full nonlinear manipulator dynamics:

$$\boldsymbol{\tau}_{\mathrm{req}} = M(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q})$$

$$\ddot{\mathbf{q}}_r = \mathbf{q}_{ss}\dot{s}^2 + \mathbf{q}_s\ddot{s}$$

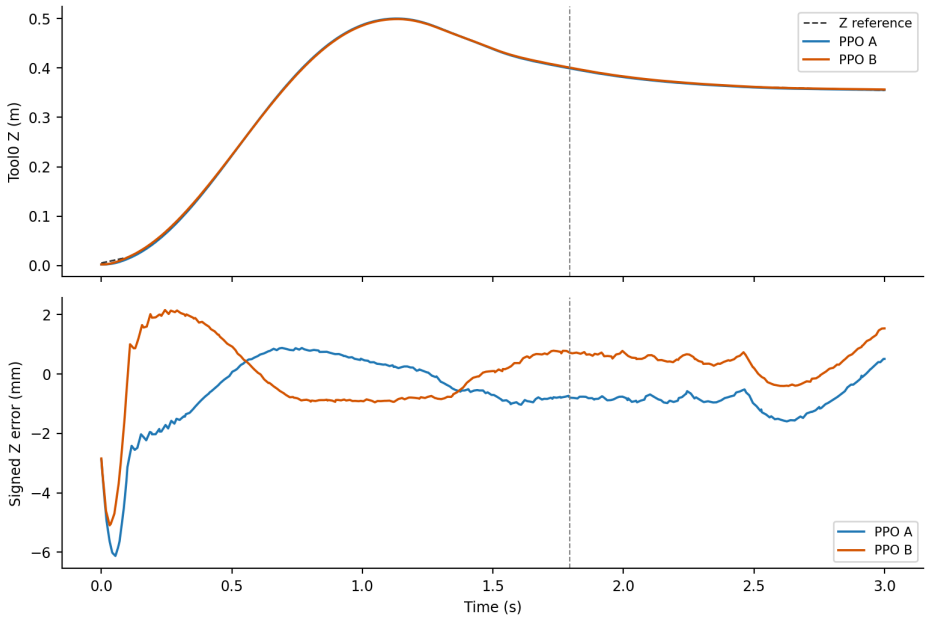


Figure 10.2: Ascent–descent tracking metrics at actual 4 kg.

The difference in vertical geometric curvature can therefore influence the acceleration reference and the associated inverse-dynamics effort. At the same time, the position and configuration of the manipulator change substantially between the two phases, altering the gravitational and inertial relationships among the joints. These properties provide a physical basis for expecting different control demands during ascent and descent. However, the experiment does not independently isolate curvature, gravity, Jacobian conditioning, or residual-policy action as the unique explanation for the observed differences. Those mechanisms act together. A valid causal attribution would require additional controlled experiments, such as changing the trajectory curvature while preserving other task characteristics, or evaluating selected residual joint actions independently. The present results support a narrower but useful conclusion: two policies trained for opposite mismatch directions show different phase-dependent tracking characteristics when evaluated under the same intermediate actual payload.

10.4 Interpretation of Intermediate-Mass Generalization

The intermediate-mass evaluation provides an important extension beyond the original policy-training conditions. Both policies were trained for an absolute payload discrepancy of 2 kg. At the 4 kg evaluation mass, each experienced a discrepancy of only 1 kg while retaining the sign of its original training condition. The resulting Cartesian tracking errors were approximately 1.8–1.9 mm. These observations show that the trained controllers remained capable of executing the prescribed transportation task under a previously untested actual payload mass. However, the meaning of generalization must be defined carefully.

10.4.1 Generalization within a fixed transportation task

The experiment changed the payload condition but retained the same reference trajectory and controller architecture. Therefore, it evaluates a limited form of mass-condition transfer. It does not establish generalization to arbitrary Cartesian paths, different transport durations, tool orientations, initial configurations, or payload attachment locations. The normalized task-time

input to the learned observation also provides information about the fixed reference phase. Consequently, a policy may learn phase-specific compensation that performs well along the accepted trajectory but would not automatically apply to a differently timed or geometrically altered motion. The intermediate-mass result should therefore be interpreted as evidence of behavior away from the original training mass within the fixed task, not as broad task or workspace generalization.

10.4.2 Relative roles of the nominal controller and learned policy

The intermediate comparison also illustrates an important limitation of interpreting the result solely as reinforcement-learning generalization. Policy A uses a nominal controller based on 3 kg dynamics. Policy B uses a nominal controller based on 5 kg dynamics. At an actual mass of 4 kg, the two nominal feedforward calculations have opposite signed model discrepancies. Each learned policy is then added to its corresponding nominal controller. The final response is therefore determined by

$$\boldsymbol{\tau}_{\mathrm{cmd}} = \boldsymbol{\tau}_{\mathrm{ff}}(m_{\mathrm{model}}) + \boldsymbol{\tau}_{\mathrm{fb}}(m_{\mathrm{model}}) + \Delta\boldsymbol{\tau}_{\pi}$$

subject to effort saturation. The difference between Policy A and Policy B cannot be attributed to the neural-network correction alone. To isolate the incremental benefit of learning for each configuration, the appropriate comparison would be

$$\Delta\mathrm{RMSE}_{\pi_A} = \mathrm{RMSE}_{\mathrm{G025},3\to4} - \mathrm{RMSE}_{\pi_A,3\to4}$$

and

$$\Delta\mathrm{RMSE}_{\pi_B} = \mathrm{RMSE}_{\mathrm{G025},5\to4} - \mathrm{RMSE}_{\pi_B,5\to4}$$

These differences quantify the reduction, if any, produced by enabling the residual policy while retaining the corresponding nominal model. The necessary zero-residual values should be taken from the original matched-protocol 4 kg records if available. They should not be inferred from the 3 kg or 5 kg validation results. In the present chapter, the confirmed numerical comparison concerns the two complete learning-augmented controller configurations. This distinction prevents the results from being overstated.

10.4.3 Does Policy B outperform Policy A?

Under the evaluated intermediate-mass condition, Policy B achieved a lower overall Cartesian RMSE. However, Policy A produced a smaller descent Cartesian error metric and a smaller ascent Z-axis error metric. Policy B produced smaller ascent Cartesian and descent Z-axis error metrics. Thus, the data do not support a universal claim that one policy is superior for all physically relevant measures. The preference depends on the evaluation objective. If the sole criterion is aggregate Cartesian RMSE over the prescribed trajectory, Policy B is preferable in the recorded experiment. If particular motion phases or directions of error are more important for an application, a different selection may be justified. For example, a transportation task with restricted overhead clearance might require close attention to the sign and peak magnitude of vertical error during ascent.

A task involving placement near a work surface might place greater emphasis on terminal vertical and horizontal tracking. The present study does not impose such application-specific success thresholds after observing the data. Instead, it demonstrates why the metrics used to evaluate learning-augmented control should correspond to the physical requirements of the intended task.

10.4.4 Implications for policy selection and adaptation

The combined results from Chapters 8–10 suggest that a fixed residual policy should not be enabled indiscriminately across changing payload conditions. The original heavier-mismatch policies improved tracking under their intended condition but degraded performance when the nominal model was matched or the actual payload became lighter. A separately trained reverse-mismatch policy improved tracking under the opposite discrepancy. At an intermediate 4 kg payload, both bidirectional PPO controllers exhibited relatively small aggregate errors but different phase-dependent characteristics. These observations motivate several possible future approaches. One is to identify the approximate payload condition and select a residual policy trained for the corresponding mismatch regime. Another is to develop a policy that receives an estimate of payload parameters or a measure of model uncertainty.

A third is to use a supervisory mechanism that reduces or disables the residual action when nominal model-based tracking is already sufficiently accurate. Such mechanisms would address an important limitation identified in the experiments: a learned correction can improve tracking when its assumptions are appropriate but become counterproductive when they are not. However, these approaches remain proposals for future investigation. The current work does not implement an automatic payload estimator, a policy-switching controller, or a formally verified residual-disable mechanism.

10.4.5 Engineering significance

The intermediate-mass experiment provides a more informative conclusion than a simple comparison of two RMSE values. It demonstrates that learned torque compensation should be evaluated as part of a complete task-specific control system, with explicit attention to the nominal dynamic model, the direction of payload mismatch, and the physical characteristics of different trajectory phases. The result also reinforces the decision to use a task-oriented Cartesian evaluation. An aggregate RMSE provides a useful summary of tracking accuracy, but it does not describe the distribution or direction of errors. The separate ascent, descent, and vertical tracking metrics reveal distinctions that could remain hidden in the overall score. At the same time, the study maintains a clear boundary between measured behavior and physical explanation.

The results establish that the two learned controllers exhibit different performance patterns under the intermediate loading condition. They do not establish a complete causal model of those differences, a guaranteed operating payload range, or the safety of transferring the policies to real hardware. The detailed intermediate-mass records and phase-dependent metrics are retained in Appendix F. The following chapter examines the broader engineering implications of the entire study, including physical interpretation of learned compensation, the practical role of simulation before hardware deployment, and the limitations that must be addressed before such a controller could be considered for industrial use.

Chapter 11

Discussion

11.1 Physical Interpretation and Control Implications

The preceding experiments examined how an established dynamics-based controller responds to changes in payload mass and whether bounded residual reinforcement learning can improve Cartesian transportation performance without replacing the nominal control structure. The results provide several important insights into the relationship between manipulator dynamics, model uncertainty, and learned torque compensation. First, accurate model-based control remains effective when the assumed dynamic parameters are consistent with the simulated plant. Second, systematic payload-model discrepancies can produce substantial task-level tracking errors even when the same controller continues to operate correctly according to its mathematical formulation. Third, learned residual torque can improve performance under a selected discrepancy, but its benefit does not necessarily extend to different loading conditions. These observations should be interpreted together rather than as independent comparisons of numerical tracking accuracy.

11.1.1 Model-based control as the foundation, not the limitation

The nominal G025 controller achieved a Cartesian RMSE of approximately 1.195 mm under the matched 3 kg ROS 2–PhysX condition. This result establishes that the computed-torque PD architecture was capable of accurately executing the prescribed transportation task within the tested simulation environment. The controller combined reference inverse-dynamics compensation with joint-position and velocity feedback. Its behavior could therefore be related directly to the manipulator's modeled inertia, velocity-dependent forces, gravitational loading, and tracking errors. When the actual payload changed while the assumed model remained fixed, the underlying plant dynamics changed. For a prescribed reference trajectory, the corresponding inverse-dynamics difference can be written as

$$\Delta\boldsymbol{\tau}_{\mathrm{ID}} = \Delta M(\mathbf{q}_r)\ddot{\mathbf{q}}_r + \Delta\mathbf{c}(\mathbf{q}_r, \dot{\mathbf{q}}_r) + \Delta\mathbf{g}(\mathbf{q}_r)$$

The discrepancy is not equivalent to an arbitrary external disturbance or a constant torque offset. It depends on the configuration and motion of the complete manipulator–payload system. The nominal controller responds through feedback when tracking errors develop, but its original feedforward torque remains based on the assumed payload model. Consequently, the deterioration observed under a mismatched payload is not evidence that computed-torque control is inherently unsuitable for industrial manipulation. Rather, it demonstrates the practical importance of dynamic-model consistency. The model-based controller performed accurately under its nominal operating condition. Changing the payload introduced a predictable source

of model uncertainty that could no longer be fully accounted for by the original feedforward calculation. The purpose of residual learning was to investigate whether this limitation could be mitigated while retaining the useful properties of the established controller.

This distinction matters because the engineering objective was never to replace an effective controller simply because a learning-based method was available. The objective was to examine whether the established control system could remain useful when the task was repeated under changed loading conditions.

11.1.2 Conditional benefits of residual reinforcement learning

The primary heavier-than-modeled PhysX experiment used a 3 kg nominal model and a 5 kg actual payload. Under this condition, the zero-residual G025 controller produced a Cartesian RMSE of 10.560 mm. The PPO residual policy reduced the error to 6.061 mm, an improvement of approximately 42.6%. TD3 and SAC also improved tracking, although their final errors and joint-level compensation patterns differed. This result demonstrates that learning-based torque compensation can provide measurable value when the nominal dynamics model contains a systematic payload error. However, the corresponding matched- and lighter-payload evaluations revealed an important limitation. When the actual mass agreed with the 3 kg nominal model, G025 achieved much lower tracking error than any of the learned residual policies. When the actual payload was reduced to 1 kg, all three policies also performed worse than the zero-residual nominal controller.

The original policies had been trained under a heavier-than-modeled condition. Their learned corrections were not guaranteed to remain appropriate when the physical loading became lighter or when the model discrepancy disappeared. Therefore, the evidence does not support the general statement that adding residual reinforcement learning improves a model-based controller. A more accurate interpretation is that residual learning improved the combined controller under a particular model discrepancy but degraded its performance under other tested conditions. The engineering significance lies in identifying the conditions under which compensation is useful. An additional control mechanism should not be regarded as beneficial merely because it is capable of generating corrective effort. Its contribution must be evaluated in relation to the operating condition and the behavior of the controller already in place.

11.1.3 Why mismatch direction matters

The experiments also demonstrate the importance of distinguishing the magnitude of a payload discrepancy from its direction. A nominal 3 kg controller carrying 5 kg experiences a positive mass discrepancy:

$$\Delta m = +2 \text{ kg} \quad \text{versus} \quad \Delta m = -2 \text{ kg}$$

•

$$\Delta \boldsymbol{\tau}_{\text{ID}}^{5\to3} = -\Delta \boldsymbol{\tau}_{\text{ID}}^{3\to5}$$

This identity provides a physical explanation for the opposite directions of the dominant torque corrections. In the original heavier-than-modeled condition, the analytical discrepancy was predominantly negative at the shoulder-lift, elbow, and wrist-1 joints under the adopted joint-axis conventions. The PPO policy also generated persistent negative residual actions at these joints. When the nominal model was changed to 5 kg and the physical payload reduced to 3 kg, the separately trained reverse-mismatch PPO policy generated positive compensating

actions at the corresponding joints. The reverse policy reduced the PhysX Cartesian RMSE from approximately 8.039 mm to 4.585 mm. These observations support a physical relationship between the sign of the model discrepancy and the direction of learned compensation.

However, the learned residual should not be interpreted as an exact identification of the missing inverse-dynamics torque. The complete controller remains closed loop:

$$\boldsymbol{\tau}_{\mathrm{cmd}} = \boldsymbol{\tau}_{\mathrm{ff}} + \boldsymbol{\tau}_{\mathrm{fb}} + \Delta\boldsymbol{\tau}_{\pi}$$

subject to the adopted effort limits. The nominal feedback term changes in response to measured tracking errors. The learned policy also responds to joint states, errors, previous actions, and trajectory progress. Accordingly, the residual action represents a learned contribution to closed-loop performance, not an independently measured physical parameter. This distinction is particularly important when interpreting the frequent saturation of the residual action bounds. At J2, J3, and J4, the analytical torque discrepancy could exceed the allowed residual magnitude. Nevertheless, the complete controller improved tracking because the feedback and residual contributions acted together. The remaining Cartesian error cannot be attributed exclusively to residual action limits without further controlled experiments.

11.1.4 Similar tracking accuracy can conceal different control behavior

The PPO and TD3 policies achieved very similar offline Cartesian RMSE values under the initial heavier-than-modeled condition. However, their joint-resolved residual torque patterns differed substantially. PPO used relatively little additional shoulder-pan torque, whereas TD3 generated much larger J1 corrections. At J2, J3, and J4, both policies frequently approached their negative action bounds. This demonstrates that similar task-level tracking errors can be produced by different joint-level control actions. The difference matters because the dynamics of a six-degree-of-freedom manipulator are coupled. A corrective action at one joint may change the response of the complete mechanism and alter the errors that must subsequently be corrected by other joints. Consequently, one cannot evaluate the physical quality of a learned controller solely by comparing total Cartesian RMSE.

Residual effort magnitude, torque direction, action variation, final-command saturation, and phase-dependent tracking all provide additional information. At the same time, smaller residual torque is not automatically evidence of a better controller. A larger correction may be physically necessary under a given operating condition, while a smaller correction may be insufficient. The appropriate question is whether the resulting effort improves the intended task without introducing unacceptable behavior or violating the adopted constraints. In the present study, PPO ultimately offered the lowest observed PhysX Cartesian RMSE under the principal heavier-than-modeled condition, together with a comparatively restrained J1 action pattern. This supported its selection for the subsequent experiments. It did not establish general superiority over TD3 or SAC. The comparison involved limited training conditions and one random seed per algorithm, and the observed differences may change under additional tasks or training runs.

11.1.5 Task-phase dependence and the role of Cartesian evaluation

The intermediate 4 kg experiments further demonstrated the importance of examining the physical task rather than relying exclusively on one aggregate performance measure. The two PPO policies were trained under opposite mismatch directions and then evaluated with the same actual 4 kg payload. Policy A retained a 3 kg nominal model, while Policy B retained a 5 kg nominal model. Their overall Cartesian RMSE values were approximately 1.931 mm and 1.763 mm, respectively. Although Policy B achieved lower aggregate error, the phase-dependent

analysis revealed that neither policy was uniformly preferable across all reported measures. Policy B showed a lower ascent Cartesian error metric, while Policy A showed a slightly lower descent Cartesian error metric. The Z-axis results showed another pattern: Policy A exhibited a smaller ascent vertical error metric, while Policy B exhibited a smaller descent vertical error metric.

These differences are relevant to the deliberate two-parabola reference trajectory. The ascending and descending segments involve different path curvature, acceleration demand, joint configuration, and gravitational loading. Therefore, different compensation behavior across the two phases is physically plausible. However, the available results do not establish which of these mechanisms was the dominant cause of the observed differences. The experiment did not independently vary curvature, gravitational loading, Jacobian conditioning, or individual residual joint actions. The phase-dependent analysis therefore identifies where the controllers behave differently, while the full causal explanation remains subject to further investigation. The result also demonstrates why Cartesian error should be examined through both magnitude and direction. For a transported object, deviations in height can have different practical consequences from deviations in horizontal position.

An aggregate Cartesian RMSE combines these components but does not preserve their signs. A complete task-level assessment should therefore retain signed Cartesian error histories, particularly when evaluating operations near work surfaces or other obstacles. The present study uses these observations to characterize controller behavior, not to claim that the reported tracking errors satisfy a predefined industrial safety or placement requirement. No independent collision-clearance guarantee was established from the tracking metrics.

11.1.6 Implications for controller reuse

The principal engineering question motivating the project was whether an existing nominal controller could remain useful under changing payload conditions without requiring an independently redesigned control system for every load. The results provide a qualified answer. The G025 architecture remained the foundation of every learning-augmented experiment. The residual policy supplied bounded additional torque rather than replacing the analytical dynamics model. Under selected payload discrepancies, the learned correction improved tracking substantially. However, the correction was condition dependent. The experiments therefore support the feasibility of augmenting an existing controller for particular dynamic uncertainties, but they do not establish that one fixed policy is appropriate across the full operating range. This finding suggests that practical controller reuse may require a higher-level decision mechanism that determines whether additional compensation is necessary and which correction is appropriate.

Such a mechanism could use estimated payload properties, observed model discrepancy, or a verified policy-validity region. Nevertheless, these approaches remain future work. The current implementation does not include automatic payload identification, verified policy switching, or a supervisory controller that guarantees preservation of nominal performance. The important distinction is between demonstrating useful compensation under controlled uncertainty and demonstrating a complete variable-payload industrial control solution. The present study establishes the former while identifying requirements for the latter.

11.2 Simulation-to-Real Deployment and Troubleshooting

Simulation-based evaluation was an intentional engineering component of this study. The decision to model and validate the controller before applying it to physical hardware was not simply a consequence of hardware availability. For industrial manipulators, unexpected motion may damage the robot, attached tooling, transported objects, or nearby equipment. Depending

on the application and protective arrangements, it may also create hazards for personnel. A controller intended for physical deployment should therefore be examined analytically and through suitable simulation before being commissioned on an actual robotic system. The purpose of these stages is to identify foreseeable dynamic and implementation-related problems under controlled conditions, reduce unnecessary physical risk, and establish evidence for subsequent supervised hardware testing. Simulation does not eliminate the need for physical validation. It provides an earlier layer of engineering verification.

The distinction becomes particularly important when working with learning-augmented controllers, because learned policies may produce actions that were not anticipated by the original nominal-controller design.

11.2.1 Correct torque calculation does not guarantee correct physical motion

A central lesson from the UR5e development process is that a controller should not be evaluated solely by the torque it calculates. The relevant outcome is the physical response produced when that torque is applied to the robotic system. This distinction emerged clearly during the initial PhysX implementation. The analytical controller could calculate gravitational and tracking compensation, but the simulated manipulator did not initially maintain the intended configuration after the physics simulation started. The robot moved under gravity before useful external effort control had been established. In an early trial, the configuration departed substantially from the intended pose, producing joint-position deviations exceeding 100 degrees. The problem was not resolved by waiting for the robot to stop moving.

A stationary or slowly moving articulation is not necessarily at the intended configuration, and a controller that has not been activated cannot be expected to restore the desired pose. The corrected implementation used gravity preload, active return, a defined stabilization gate, and a separate trajectory-arming sequence. This procedure illustrates a general engineering principle: the initial state of the controlled system must be established before the controller's task performance can be meaningfully measured. Otherwise, the experimental result may be dominated by startup behavior that is unrelated to the intended trajectory-tracking problem. The same consideration would be relevant to physical hardware. A real robot can experience unexpected motion if its controller, gravity compensation, brake release, drive enabling, or motion command sequence is inconsistent with the mechanical loading.

The precise physical implementation differs from Isaac Sim, but the need for controlled initialization remains.

11.2.2 Gravity compensation and stationary holding

A robotic manipulator holding a payload must generate sufficient joint effort to balance the gravitational loading associated with its configuration. For an ideal static condition,

$$\dot{\mathbf{q}} = \mathbf{0}, \qquad \ddot{\mathbf{q}} = \mathbf{0}$$

$$\boldsymbol{\tau}_{\text{hold}} = \mathbf{g}(\mathbf{q}; m_p)$$

under the adopted sign convention and absent additional external forces. This equation explains why a robot may move even when no trajectory has been commanded. A zero-velocity reference does not imply that the required motor torque is zero. In the present simulation, disabling the internal position drives while failing to supply external gravity-compensating effort allowed the manipulator to move under its own weight and the attached payload. The implementation therefore required an explicit torque-controlled holding strategy. The accepted

computed-torque controller supplied the relevant model-based effort together with corrective feedback.

However, even when the calculated gravity compensation is correct according to the analytical model, actual holding performance depends on whether the corresponding command reaches the simulated articulation and is applied consistently. This is why the gravity-preload procedure was validated through observed joint motion rather than accepted solely from the calculated torque values. In physical hardware, additional considerations include brake behavior, actuator dynamics, friction, structural flexibility, and the manufacturer's internal control architecture. Those effects were not modeled or independently validated in the present study. They would need to be assessed before transferring an external torque-control strategy to an actual UR5e.

11.2.3 Commanded torque, effective actuation, and timing

The distinction between commanded and effectively applied torque is another important source of control-system uncertainty. A controller may compute a desired generalized effort at time (t_k), but that effort may not influence the plant at the same instant. State acquisition, controller computation, message transmission, actuator-interface processing, and physics integration all introduce ordering and timing requirements. In a discrete-time system, a command calculated from the available state can be written as

$$\boldsymbol{\tau}_k = \mathcal{K}(\mathbf{q}_k, \dot{\mathbf{q}}_k, \mathbf{q}_{r,k}, \dot{\mathbf{q}}_{r,k}, \ddot{\mathbf{q}}_{r,k})$$

The physical plant then evolves under the effort effectively applied during the following interval. If the applied command corresponds to an earlier state, the controller may act on stale information. The resulting error depends on the plant dynamics, feedback gains, reference motion, and delay. This issue is especially relevant for motion containing rapid changes in acceleration or configuration-dependent loading. The adopted ROS 2–PhysX architecture used a nominal 500 Hz physics integration frequency and a requested 50 Hz controller update rate. The ten-to-one ratio provides a practical multirate structure, but the ROS callbacks and PhysX progression were not enforced by one deterministic scheduling mechanism. The measured joint-state topic arrived at rates that varied between runs, and the logged trajectory intervals contained unequal sample spacing.

Therefore, the reported controller update frequency should be interpreted as a configured execution rate, not proof of an exact number of control updates per simulated second. Earlier UR5e validation work also showed that the timing of torque application relative to the physics update could materially change tracking performance even when the underlying controller formulation was unchanged. In that preceding experiment, correcting the timing of applied effort produced a substantial reduction in tracking error without requiring a corresponding redesign of the control law. This experience supports the need to treat execution timing as part of the controlled system.

However, the earlier four-second joint-space benchmark and the present three-second payload-transportation study are different experiments. Their numerical tracking results should not be combined or interpreted as repeated measurements of the same controller. The general lesson is that the timing of effective actuation can be as important as the mathematical expression used to calculate the effort.

11.2.4 Diagnosing unintended vibration and oscillatory behavior

Unexpected oscillation or vibration is particularly important in industrial payload transportation because it can degrade tracking, disturb the transported object, and increase mechanical loading. However, the presence of oscillatory motion does not establish that the control algorithm

itself is unstable. Several distinct mechanisms can produce superficially similar behavior. An insufficient or incorrectly signed gravity-compensation command may allow the manipulator to sag or move away from its intended configuration. Inappropriate feedback gains may produce underdamped responses or excessive control action. Delays between state measurement and effective torque application may reduce the damping provided by the controller. An incorrect payload mass, center-of-mass offset, or inertia tensor may generate systematic feedforward error. The simulator's internal drive settings may also interact with externally commanded effort if the actuation architecture is not configured consistently.

These mechanisms require different corrective actions. For example, increasing derivative gain may reduce a tracking transient caused by insufficient damping, but it does not correct an incorrectly defined gravity direction or joint-axis convention. Likewise, increasing proportional gain cannot guarantee that a command arrives before the relevant physics integration step. Therefore, troubleshooting should proceed by identifying the physical or computational mechanism responsible for the observed motion. A defensible diagnostic sequence begins with verifying the mechanical model and frame conventions. It then examines the gravity-compensated holding condition, the relationship between commanded effort and joint response, the control and physics time bases, and the influence of feedback gains. Only after those elements have been examined should unexplained motion be attributed to the learned policy.

This sequence is especially important for residual reinforcement learning. A learned controller may compensate for implementation artifacts if they are consistently present in its training environment. Such compensation can improve training performance without addressing the underlying engineering problem. If the implementation artifact changes during transfer to another simulator or actual hardware, the learned correction may no longer be appropriate. The present study therefore established the nominal ROS 2–PhysX implementation before evaluating residual policies. That decision reduced the risk of interpreting unresolved startup or effort-delivery problems as evidence of payload-dependent learning behavior.

11.2.5 Distinguishing observed problems from potential hardware hazards

The study directly observed gravity-induced startup motion and the importance of controlled initialization in PhysX. It also recorded implementation-dependent tracking behavior and the effects of executing the controller through a distinct physics environment. These are experimental observations. By contrast, equipment collision, payload damage, actuator overheating, loss of grasp stability, and personnel injury are potential consequences that must be considered in real-world risk assessment. They were not observed as measured outcomes of this simulation study. The distinction is important because engineering reports should not present plausible hazards as though they had occurred during the experiment. The value of the simulation is that it allows potential mechanisms leading to such hazards to be investigated before hardware deployment. This includes examining unexpected motion, effort limits, workspace constraints, startup sequencing, and controller behavior outside nominal conditions.

Nevertheless, the absence of an observed problem in simulation does not prove that the corresponding real-world hazard has been eliminated.

11.3 Safety, Validation Boundaries, and Future Development

The preceding chapters demonstrate a systematic progression from analytical robot modeling to constrained task design, nominal control, physics-based validation, residual learning, and payload-condition transfer experiments. This progression provides a technically grounded basis for discussing the next steps required before physical deployment. The objective of the present section is not to claim that the learning-augmented controller is already suitable for unrestricted

industrial operation. Rather, it is to distinguish the capabilities established by the experiments from the additional evidence required for reliable and safe implementation.

11.3.1 Simulation as a prerequisite for hardware commissioning

An engineering validation process should begin by identifying the intended task, the physical limits of the robot, and the foreseeable consequences of control failure. In the present investigation, this meant examining the UR5e's manufacturer specifications, workspace constraints, inverse-kinematics feasibility, Jacobian conditioning, reference velocities, and predicted torque requirements before executing the transportation trajectory. The task was then evaluated in a numerical dynamics environment and subsequently through an independently executed PhysX articulation. These stages were not substitutes for engineering judgment. They were tools used to test assumptions, expose implementation errors, and determine whether the observed behavior was consistent with the intended physical system. The startup gravity-sag problem illustrates the practical value of this process. If the controller had been applied directly to physical hardware without adequate initialization and protective measures, a corresponding uncontrolled-motion mechanism could have created a hazardous condition.

The simulation made the problem observable in a controlled environment and allowed the startup procedure to be redesigned before any hardware implementation was attempted. This is precisely the type of issue that simulation-based pre-commissioning should identify. However, physical hardware remains necessary for validating real actuator behavior, structural compliance, transmission effects, sensor characteristics, thermal limits, and actual interaction with the surrounding environment. The appropriate engineering conclusion is therefore neither that simulation guarantees safety nor that simulation is insufficient to be useful. Simulation is an important verification stage within a broader hardware commissioning process.

11.3.2 Limits of the physical model

The analytical UR5e model and PhysX articulation were constructed to represent the same six-degree-of-freedom mechanism, but they do not encompass every relevant physical effect. The transportation payload was represented as a rigid cube with a fixed attachment. Real grasped objects may exhibit compliance, relative motion, uncertain center-of-mass location, or imperfect attachment. Joint friction and actuator dynamics may differ from the assumptions used in the analytical equations. The physical robot's internal servo controller may also contain proprietary or implementation-specific behavior not represented by the external effort-control setup. In addition, the adopted simulation effort limits were derived from the configured robot asset rather than independently verified manufacturer continuous and peak torque specifications. The UR5e technical specification provides useful manufacturer information, including nominal reach, maximum payload, joint-speed limits, and payload capacity as a function of center-of-gravity offset.

However, those specifications do not directly establish the admissibility of every simulated torque command. Hardware validation would require checking the corresponding manufacturer-defined operating limits for the complete robot, payload, and motion profile. The numerical tracking errors reported in this study also require careful interpretation. Cartesian positions were reconstructed from joint states using forward kinematics. They were not independently measured using an external motion-tracking system. Consequently, these results do not quantify unmodeled structural deflection, elastic transmission behavior, payload slip, or absolute tool-position accuracy on physical hardware. The manufacturer-specified pose repeatability is a separate performance characteristic and should not be equated with the dynamic Cartesian RMSE obtained from the simulation.

11.3.3 Clearance and collision safety

The prescribed transportation task includes nominal clearance above the pick and place work surfaces. However, a nominal geometric clearance is not a guarantee of collision-free closed-loop execution. Tracking error may shift the payload relative to the intended path. The original initial-position perturbation also produced a measurable offset between the actual starting Tool0 position and the unperturbed reference. The ROS 2–PhysX validation did not record a separate continuous minimum distance between the payload collision geometry and the surrounding work surfaces. Therefore, the reported Cartesian RMSE cannot be used to establish that the payload maintained a specified physical clearance throughout the task. A hardware-oriented validation process would require independent assessment of the swept geometry of the complete robot, payload, and tooling. The assessment should consider tracking uncertainty, possible overshoot, emergency stopping motion, and the consequences of controller or communication failure.

For close-clearance operations, the relevant measure is the minimum physical separation between collision surfaces under the actual executed motion, not solely the position error of the Tool0 origin. This issue would become particularly important if the task were extended to grasp acquisition, insertion, contact-rich placement, or operation near personnel. Those applications introduce constraints and hazards not represented by the current rigidly attached free-space transportation model.

11.3.4 Stability and learned-policy boundaries

The residual action was explicitly bounded:

$$|\Delta\tau_{\pi,i}| \leq \tau_{\mathrm{res,max},i}$$

Possible mechanisms include monitoring payload estimates, tracking-error trends, model discrepancy indicators, and the magnitude or rate of change of residual actions. However, merely detecting a large tracking error is not equivalent to knowing which correction should be applied. A supervisory mechanism would need to be designed and evaluated as a separate control component. The system should also define how residual compensation is disabled and how the nominal controller continues to operate when the learned policy becomes unavailable or unreliable. For a physical robot, these decisions would have to be integrated with the manufacturer's protective functions and the application-specific risk assessment. The present study does not implement or validate such a supervisory safety system.

11.3.5 Hardware deployment considerations

Before applying the developed control architecture to a physical UR5e, several additional engineering steps would be necessary. First, the actual robot and payload parameters would need to be checked against the manufacturer's permitted operating envelope, including the relevant center-of-gravity offset and complete tooling mass. Second, the controller's interface to the physical robot would need to be established using an appropriate supported control mode. The external torque-command structure used in Isaac Sim should not be assumed to correspond directly to an accessible or permissible hardware interface. Third, startup, holding, and stopping behavior would require supervised verification under controlled conditions. The robot should not begin a transportation task until its initial configuration, payload state, communication status, and protective arrangements have been verified. Fourth, commanded effort should be distinguished from the effective actuator response.

Where accessible, relevant drive and state diagnostics would be required to evaluate latency, saturation, and unexpected deviations. Fifth, the trajectory should be commissioned progres-

sively, beginning with reduced operating demands and protective controls appropriate to the assessed risks. This would permit the physical system's response to be compared with analytical predictions and simulation results before moving toward the intended operating condition. Finally, the application would require an independent safety assessment covering mechanical interference, payload retention, protective stopping, personnel exposure, and foreseeable failures. A favorable Cartesian tracking result cannot replace these steps. The role of the present research is to establish a technically interpretable control and simulation basis from which such validation could proceed.

11.3.6 Limitations of the experimental evidence

The conclusions of the present study are constrained by its experimental design. The initial PPO, SAC, and TD3 comparison used one random seed and a fixed transportation trajectory. Best checkpoints were selected using evaluations on the same prescribed task, rather than independent trajectories. The subsequent payload studies tested selected actual masses and nominal-model configurations, not a continuous distribution of all possible payload conditions. The 4 kg cross-evaluation examined two complete learning-augmented controllers with different nominal models. It therefore did not isolate the effect of the neural-network residual alone without corresponding zero-residual baselines for both configurations. Furthermore, the numerical training environment and PhysX runtime were not proven dynamically equivalent over the complete state space. These limitations do not invalidate the observed results.

They restrict the extent to which those results can be generalized. The evidence demonstrates that bounded learned compensation can improve Cartesian tracking under selected payload-model discrepancies and that its applicability depends on the direction and magnitude of the mismatch. It does not establish universal robustness, an optimal learning algorithm, guaranteed collision safety, or suitability for unrestricted hardware operation.

11.3.7 Future research priorities

The results suggest that further progress should focus on the applicability and supervision of learned compensation rather than simply extending the original training duration. A first priority is the development of a payload-aware compensation strategy. This could involve estimating the current payload parameters and using the estimate to select an appropriate residual policy or adjust the nominal dynamics model. A second priority is the development of a mechanism for suppressing unnecessary residual actions. The matched-payload experiments demonstrated that activating a policy trained for another condition can degrade the performance of an already accurate nominal controller. A practical learning-augmented architecture should therefore preserve nominal-controller performance when additional compensation is not needed. A third priority is broader validation over task and model variation.

The current policies were trained for a fixed transportation reference. Their behavior under different workspace configurations, transport durations, tool orientations, and payload attachment conditions remains unknown. Controlled experiments varying these conditions would help distinguish compensation of general dynamic uncertainty from behavior learned for one trajectory. A fourth priority is a more complete dynamic analysis of learned actions. Joint-resolved residual records already reveal important differences among PPO, SAC, and TD3. Additional time-aligned comparisons with analytical torque discrepancy, nominal feedback effort, effective actuation, and phase-dependent errors would clarify which learned actions contribute most directly to improved task performance. A fifth priority is the integration of explicit operational safeguards before physical commissioning. This includes validated initialization, controlled fallback behavior, appropriate velocity and effort monitoring, and independent workspace-clearance assessment.

Formal closed-loop stability analysis or verified safety constraints would require additional controller development beyond the present experiments.

11.3.8 Overall engineering assessment

The study demonstrates that the value of learning-augmented control cannot be determined independently of the physical task, nominal dynamics model, and operating condition. Accurate model-based control remains highly effective when its assumptions are appropriate. Residual reinforcement learning can provide valuable additional compensation when systematic model discrepancies degrade performance. However, the learned correction may become counterproductive when its training assumptions no longer match the actual system. The bidirectional experiments establish that compensation direction matters, while the intermediate-mass results show that similar aggregate tracking accuracy can coexist with different phase-dependent behavior. The ROS 2–PhysX validation also demonstrates that controller performance depends on more than the correctness of its mathematical formulation. Initialization, gravity compensation, command timing, effective actuation, and the consistency of the simulated physical system all influence the observed response.

These findings point toward a practical engineering principle. The objective is not to maximize the contribution of the learned controller, nor to replace an established controller simply because a learning method is available. The objective is to preserve dependable nominal behavior while introducing additional compensation only when its contribution is justified by the operating condition and supported by validation evidence. A controller should therefore be evaluated not only by its calculated torque or aggregate tracking score, but also by the physical response it produces, the assumptions under which it operates, and the consequences when those assumptions change. The following chapter concludes the study by returning to its original practical motivation: reliable Cartesian payload transportation under varying dynamic conditions, using an established model-based controller supplemented by carefully evaluated learned compensation.

Chapter 12

Conclusions

This study investigated the integration of model-based control and residual reinforcement learning for Cartesian payload transportation using a six-degree-of-freedom UR5e manipulator. The research was motivated by a practical engineering question: whether an established robotic control system could remain useful when performing similar transportation tasks under changing payload conditions, without requiring an independently redesigned controller for every load. The investigation was not intended to introduce a fundamentally new manipulator control algorithm or demonstrate that reinforcement learning is inherently superior to conventional control. Instead, it examined how an analytically established controller, a physically motivated transportation task, and a bounded learned torque correction could be combined and evaluated under explicitly defined dynamic uncertainties.

The transportation benchmark was developed from the physical characteristics of the UR5e rather than from arbitrary Cartesian targets. Manufacturer specifications, workspace reach, joint configuration, Jacobian conditioning, reference velocities, and inverse-dynamics requirements were considered before the final task was accepted. The resulting three-second trajectory employed two parabolic vertical segments, providing distinct ascending and descending motion characteristics while preserving a consistent reference for controller comparison. This task-oriented formulation was important because the purpose of the manipulator was to transport an object through a prescribed three-dimensional path. Joint tracking remained essential to executing that motion, but Cartesian end-effector tracking was adopted as the principal measure of task performance.

The nominal-controller study compared computed-torque PD and constrained MPC under the prescribed transportation conditions. Twenty-five computed-torque PD configurations and thirty MPC configurations were examined to identify a controller offering an appropriate balance of tracking accuracy, computational practicality, effort constraints, and physical interpretability. The selected G025 computed-torque PD controller achieved a Cartesian RMSE of 1.195 mm in ROS 2–PhysX validation with a matched 3 kg nominal and actual payload. Its feedforward–feedback decomposition provided a dependable reference for subsequent learning experiments. This result confirmed the usefulness of an accurately parameterized model-based controller. It also established an important experimental requirement: the contribution of reinforcement learning should be assessed against a functioning nominal system, rather than against a controller whose implementation or physical-model consistency remains unresolved.

The independent ROS 2–PhysX validation further demonstrated that analytical correctness alone does not guarantee the expected physical response. During implementation, gravitational loading and controller initialization produced substantial unintended motion before effective holding torque had been established. Resolving this behavior required gravity preload, active return to the intended initial configuration, a defined stabilization gate, and controlled activation of trajectory tracking. These observations reinforced the role of physics-based simulation as

an engineering verification stage preceding potential hardware commissioning. Correct control equations must be accompanied by consistent actuation, initialization, timing, and state-feedback behavior. Such validation can identify foreseeable problems before physical equipment is exposed to them, although it cannot replace hardware-specific testing or establish safety certification.

With the nominal controller validated, payload-model discrepancy was introduced while retaining the same transportation task. When G025 assumed a 3 kg payload but the PhysX plant carried 5 kg, Cartesian RMSE increased to 10.560 mm. This deterioration demonstrated the practical consequences of using a fixed nominal dynamics model when the actual payload changes. Residual reinforcement learning was then investigated as an additional torque-compensation mechanism. PPO, SAC, and TD3 were trained under a common numerical interaction budget using six-dimensional bounded residual actions added to the existing G025 controller. The numerical learning experiments demonstrated that all three algorithms could reduce the tracking error associated with the heavier-than-modeled payload condition. However, the learned policies exhibited different convergence characteristics and joint-level torque distributions, even when their aggregate Cartesian tracking errors were similar.

In particular, PPO and TD3 achieved closely comparable numerical tracking performance while applying substantially different shoulder-pan residual torques. This demonstrated that task-level accuracy alone does not fully characterize the physical behavior of a learned controller. The subsequent ROS 2–PhysX experiments provided a more decisive assessment of practical applicability. Under the 3 kg nominal model and 5 kg actual payload condition, PPO reduced Cartesian RMSE from 10.560 mm to 6.061 mm, corresponding to an improvement of approximately 42.6%. This result established that bounded residual learning could improve the performance of the existing nominal controller under a specific systematic payload-model discrepancy.

However, the improvement was conditional. When the nominal and actual payload masses were both 3 kg, the original G025 controller achieved substantially better tracking than the learned residual policies. Similarly, when the actual payload was reduced to 1 kg while the nominal model remained at 3 kg, the learned policies increased tracking error relative to the zero-residual controller. These findings demonstrated that a learned correction beneficial under one loading condition may become unnecessary or counterproductive when the model discrepancy changes or disappears. The direction of payload-model uncertainty was therefore examined through a separate reverse-mismatch experiment. A PPO policy was trained using a 5 kg nominal model and a 3 kg actual payload. In ROS 2–PhysX validation, the reverse policy reduced Cartesian RMSE from approximately 8.039 mm to 4.585 mm, an improvement of approximately 43.0%.

The dominant learned torque corrections also changed direction at joints associated with substantial payload-dependent loading. This behavior was consistent with the analytical reversal of the reference inverse-dynamics discrepancy when the nominal and actual payload masses were exchanged. Nevertheless, the learned residual actions should not be interpreted as exact identification of the missing dynamics. The nominal feedback controller remained active, and the policy optimized closed-loop task performance rather than direct agreement with the analytical torque difference. The reverse-mismatch result therefore established the usefulness of separately learned compensation under the opposite uncertainty direction, not the ability of one policy to operate universally across both conditions. The intermediate 4 kg cross-evaluation extended this investigation beyond the exact masses used during training. The two independently trained PPO controllers achieved overall Cartesian RMSE values of 1.931 mm and 1.763 mm, respectively.

Although the controller based on the 5 kg nominal model produced the lower aggregate error, phase-dependent analysis revealed different relative performance during ascent and descent. The vertical tracking metrics also exhibited differences that were not fully represented by the aggregate Cartesian score. These observations reinforced the importance of evaluating learned

control according to the physical requirements of the task. Similar overall errors may arise from different combinations of horizontal and vertical deviations, joint-level compensation, and phase-dependent responses. At the same time, the 4 kg results should be interpreted as a limited intermediate-mass cross-evaluation. Because the two policies retained different nominal models, the comparison did not isolate the residual-network contribution independently of the underlying model-based controllers. Nor did one intermediate loading condition establish generalization throughout the complete payload range.

Taken together, the experiments demonstrate three principal engineering findings. First, model-based control remains an effective foundation for robotic transportation when its assumed physical parameters adequately represent the system. Reinforcement learning should not be introduced under the assumption that an established controller must be replaced. Second, bounded residual learning can improve task-level tracking under selected dynamic uncertainties while preserving the nominal controller's underlying structure. Its corrective actions can also be interpreted in relation to the manipulator's configuration-dependent inertia and gravitational loading. Third, the applicability of learned compensation is conditional. The direction and magnitude of payload-model discrepancy influence whether the learned action improves or degrades closed-loop performance. A policy trained for one operating condition cannot automatically be assumed appropriate for another.

The broader engineering implication is that controller reuse under changing payload conditions requires more than a capable nominal controller and an accurate learned residual policy. It also requires a means of determining when additional compensation is necessary and when the nominal controller should operate without it. Potential future developments include payload-aware policy selection, online estimation of relevant dynamic parameters, adaptive residual scaling, and supervisory mechanisms that suppress learned actions when their operating assumptions are not satisfied. Broader evaluation across payload masses, trajectories, initial conditions, and independent training runs would also be necessary to assess robustness and generalization. Before physical hardware deployment, further verification would be required for actuator behavior, controller-interface compatibility, collision clearance, payload retention, operational safeguards, and the robot's manufacturer-defined limits. The present simulation results do not establish formal closed-loop stability, guaranteed collision safety, or unrestricted industrial applicability.

The contribution of this research is therefore not a claim that reinforcement learning universally outperforms conventional robotic control. It is the systematic examination of how a physically interpretable nominal controller and bounded learned compensation behave when the assumptions of the dynamic model change. The engineering value of learning-augmented control lies not in replacing an effective nominal controller, but in understanding when additional compensation is necessary, when it is beneficial, and when it should remain inactive.

References

[1] International Organization for Standardization, *ISO 10218-1:2025, Robotics — Safety Requirements — Part 1: Industrial Robots*, and *ISO 10218-2:2025, Part 2: Industrial Robot Applications and Robot Cells*, 2025.

[2] M. W. Spong, "An Historical Perspective on the Control of Robotic Manipulators," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, 2022. doi:10.1146/annurev-control-042920-094829.

[3] J.-J. E. Slotine and W. Li, "On the Adaptive Control of Robot Manipulators," *The International Journal of Robotics Research*, vol. 6, no. 3, pp. 49–59, 1987. doi:10.1177/027836498700600303.

[4] T. Johannink et al., "Residual Reinforcement Learning for Robot Control," *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 6023–6029, 2019. doi:10.1109/ICRA.2019.8794127.

[5] Universal Robots A/S, *UR5e Technical Specification*, November 2023.

[6] P. K. Khosla and T. Kanade, "Experimental Evaluation of Nonlinear Feedback and Feedforward Control Schemes for Manipulators," *The International Journal of Robotics Research*, vol. 7, no. 1, 1988.

Code and Data Availability

The implementation source, ROS 2–Isaac Sim validation code, controller experiments, and supporting datasets can be found at

https://github.com/greatsyong/ur5e-control-analysis

The archived CSV files accompanying this Overleaf project are retained in the `data/` directory. The repository provides the project source for interpreting and reproducing the appendix results.

Appendix A

Additional Modeling and Calculation Notes

The analytical definitions and derivations in Chapters 2–3 establish the primary calculation conventions. Supplementary modeling references and original source files are retained in the project data directory. No additional physical parameter values have been inferred for this appendix.

Appendix B

Parameter and Configuration Inventory

Item	Value
UR5e joints	6 revolute
Reference duration	3.0 s
Nominal PhysX step	0.002 s
Requested ROS controller interval	0.020 s
Nominal controller	G025, $K_p = 256$, $K_d = 41.6$
Residual action	$[5, 5, 5, 0.5, 0.5, 0.01]$ Nm
Simulation effort limits	$[150, 150, 150, 28, 28, 28]$ Nm

Table B.1: Selected implementation parameters (simulation, not manufacturer torque ratings).

Appendix C

Complete PD and MPC Screening Data

The following tables reproduce the candidate-level results from the original 25-run PD screening and the 18+12-run two-stage MPC screening. The source CSV files, including additional logged measurements, are supplied in the project data directory.

C.1 Computed-torque PD: all 25 candidates

Candidate	K_p	K_d	Tool RMSE	Joint RMSE	Final speed	Peak Nm	Clips
G001	64	11.2	2.089	0.259	1.687	80.89	0
G002	64	13.6	1.904	0.235	1.466	80.84	0
G003	64	16.0	1.778	0.218	1.299	80.80	0
G004	64	18.4	1.696	0.205	1.165	80.75	0
G005	64	20.8	1.646	0.196	1.054	80.70	0
G006	100	14.0	1.563	0.177	1.402	81.78	0
G007	100	17.0	1.498	0.167	1.209	81.68	0
G008	100	20.0	1.455	0.159	1.064	81.58	0
G009	100	23.0	1.429	0.153	0.948	81.48	0
G010	100	26.0	1.417	0.148	0.854	81.37	0
G011	144	16.8	1.324	0.136	1.723	82.78	0
G012	144	20.4	1.299	0.131	1.441	82.60	0
G013	144	24.0	1.286	0.127	1.234	82.41	0
G014	144	27.6	1.282	0.124	1.074	82.26	0
G015	144	31.2	1.286	0.121	0.947	82.26	0
G016	196	19.6	1.192	0.112	2.116	83.85	0
G017	196	23.8	1.181	0.109	1.794	83.54	0
G018	196	28.0	1.179	0.107	1.545	83.31	0
G019	196	32.2	1.185	0.105	1.347	83.31	0
G020	196	36.4	1.195	0.104	1.185	83.31	0
G021	256	22.4	1.106	0.097	2.393	84.94	0
G022	256	27.2	1.099	0.095	2.091	84.53	0
G023	256	32.0	1.103	0.094	1.836	84.53	0
G024	256	36.8	1.113	0.093	1.623	84.53	0
G025	256	41.6	1.127	0.093	1.443	84.53	0

C.2 MPC: stage I (18 candidates)

Candidate	N	Q	R	Tool RMSE	Joint RMSE	QP ms	Clips
M001	5	0.50	0.25	20.337	0.768	0.861	0
M002	5	0.50	1.00	132.662	3.440	0.794	0
M003	5	1.00	0.25	8.305	0.333	0.784	0
M004	5	1.00	1.00	16.547	1.054	0.763	0
M005	5	2.00	0.25	3.566	0.211	0.770	0
M006	5	2.00	1.00	20.353	0.769	0.772	0
M007	10	0.50	0.25	6.825	0.613	1.082	0
M008	10	0.50	1.00	17.294	0.717	1.054	0
M009	10	1.00	0.25	4.447	0.744	1.086	0
M010	10	1.00	1.00	13.874	0.639	1.050	0
M011	10	2.00	0.25	4.014	0.903	1.106	0
M012	10	2.00	1.00	6.822	0.612	1.100	0
M013	15	0.50	0.25	5.237	0.873	1.609	0
M014	15	0.50	1.00	12.637	0.718	1.589	0
M015	15	1.00	0.25	4.729	1.017	1.621	0
M016	15	1.00	1.00	7.268	0.758	1.636	0
M017	15	2.00	0.25	4.463	1.099	1.651	0
M018	15	2.00	1.00	5.322	0.893	1.638	0

C.3 MPC: stage II (12 candidates)

Candidate	N	Q	R	Tool RMSE	Joint RMSE	QP ms	Clips
M001	5	2.00	0.10	2.470	0.416	0.800	0
M002	5	2.00	0.25	3.566	0.211	0.780	0
M003	5	4.00	0.10	2.798	0.649	0.789	0
M004	5	4.00	0.25	2.479	0.337	0.779	0
M005	5	8.00	0.10	3.165	0.828	0.781	0
M006	5	8.00	0.25	2.687	0.585	0.786	0
M007	10	2.00	0.10	3.909	1.044	1.149	0
M008	10	2.00	0.25	4.014	0.903	1.078	0
M009	10	4.00	0.10	3.789	1.093	1.147	0
M010	10	4.00	0.25	3.937	1.018	1.123	0
M011	10	8.00	0.10	3.726	1.130	1.186	0
M012	10	8.00	0.25	3.849	1.086	1.136	0

Appendix D

ROS 2–PhysX Validation Records

All recorded trajectory and effort CSVs retained from the prior engineering project are included under `data/`. They distinguish commanded effort, reconstructed Cartesian motion, and timestamped tracking windows. The tables in Chapters 5 and 8 remain the authoritative summaries for the discussed comparisons.

Appendix E

Reinforcement-Learning Records

The original learning curves, evaluation traces, and policy-specific source data are retained under `data/rl/`. The standalone early-stage RL report is reproduced in Appendix G.

Appendix F

Bidirectional and Intermediate-Payload Records

Reverse-mismatch and intermediate-mass CSV records are provided under `data/rl/reverse_5to3/` and `data/rl/heldout_4kg/`. Metrics that cannot be inferred from the available original records have not been fabricated.

Appendix G

Original Residual RL Report

The following report is reproduced as a historical experiment-stage document. It predates some of the subsequent ROS 2–PhysX policy comparisons and should be interpreted in that original evidentiary context.

UR5e Heavy-Payload Transport

Residual Reinforcement Learning Under Payload Mass Mismatch

PPO, SAC, and TD3: training convergence, six-joint correction, and physical interpretation

Sooyong Kim

11 October 2026 | ROS 2 Humble | Isaac Sim 6.1 | Python / Stable-Baselines3

Abstract

Following the nominal-controller study documented in *UR5e Heavy-Payload Transport: Geometry, Dynamics, Controller Selection, and ROS 2–PhysX Validation* (10 October 2026), this report addresses systematic payload-model mismatch. A six-degree-of-freedom UR5e tracks the unchanged three-second Cartesian transport trajectory under a computed-torque G025 controller parameterized by a 3 kg payload, while the offline plant carries 5 kg. PPO, SAC, and TD3 learn six-joint bounded residual torques under an identical 30,000-interaction training budget (seed 42). Each algorithm completes a 150-control-step deterministic evaluation. Best-checkpoint Cartesian RMSE is 6.094 mm (PPO, 27,000 steps), 6.934 mm (SAC, 27,000 steps), and 6.112 mm (TD3, 21,000 steps), versus 10.636 mm for the zero-residual G025 offline baseline. Detailed time histories reveal significant policy differences, especially in shoulder-pan torque, while common saturation of residual action bounds on shoulder-lift, elbow, and wrist-1 is consistent with a separately derived 3-to-5 kg inverse-dynamics mismatch. TD3 demonstrates more rapid improvement in the measured learning curve and is prioritized for later ROS 2–PhysX evaluation; neither algorithm ranking across random seeds nor learned-policy PhysX performance is claimed here.

1 Research context and experimental question

The preceding 19-page engineering report established a geometrically feasible transport cell, a fixed 3 s reference, analytical UR5e dynamics, a 25-candidate computed-torque PD search, a 30-candidate offline MPC search, and nominal ROS 2–PhysX validation. Its selected controller, G025, uses $K_p = 256$ and $K_d = 41.6$ and achieved 1.195 mm Cartesian RMSE in the *matched* 3 kg PhysX case. This report does not repeat those foundations: it tests the additional control benefit of a residual learning policy when the controller's payload-mass assumption is wrong.

The physical question is: can a bounded torque correction compensate for a fixed but unknown extra payload mass without replacing a nominal model-based controller? The task is the same rigid, post-grasp transport with a fixed 50 mm cube, Tool0-relative payload center $[0, 0, 0.025]^T$ m, and reference duration 3 s. Grasping, release, contact-force tracking, and collision-clearance guarantees are outside this test. Pick/place tables are visual scene elements in the executed control tests; their geometry is not used to infer reaction forces.

1.1 Evidence levels and comparison boundaries

- **Earlier nominal PhysX validation:** model 3 kg, physical 3 kg, G025 RMSE 1.195 mm, with ROS 2 command and state timing.
- **Earlier mismatched PhysX baseline:** model 3 kg, physical 5 kg, G025 RMSE 10.560 mm; model 5 kg, physical 3 kg, G025 RMSE 8.034 mm. These are physics-system baseline observations, not RL tests.
- **Present offline controlled RL comparison:** model 3 kg, offline plant 5 kg, zero-residual G025 RMSE 10.6358 mm, with exactly the same integration and reference conventions for PPO, SAC, TD3. The similarity of 10.560 and 10.6358 mm is evidence of agreement in this particular case, not a global proof of dynamical equivalence.

Offline evaluations have prescribed numerical integration (2 ms) and controller updates (20 ms); Isaac/ROS feedback timing is asynchronous. Do not merge offline and PhysX data as if they were repeated measurements in the same environment.

2 Residual learning formulation

2.1 Nominal model-based controller

The load-dependent manipulator model and reference feedforward are

$$M(q; m)\ddot{q} + c(q, \dot{q}; m) + g(q; m) = \tau, \tag{1}$$

$$\tau_{\text{ff}}(t; m) = M(q_r; m)\ddot{q}_r + c(q_r, \dot{q}_r; m) + g(q_r; m), \tag{2}$$

$$\tau_{\text{G025}} = \tau_{\text{ff}}(t; 3) + M(q; 3)\big[256(q_r - q) + 41.6(\dot{q}_r - \dot{q})\big]. \tag{3}$$

The learned-policy command is

$$\boxed{\tau_{\text{cmd}} = \text{clip}_{\tau_{\max}}[\tau_{\text{G025}} + \Delta\tau_\pi(o_t)]\,, \qquad \tau_{\max} = [150, 150, 150, 28, 28, 28]^T \text{ Nm.}} \tag{4}$$

The above are the authored USD simulation effort settings, not verified physical UR5e actuator ratings. Feedforward is *recomputed* for the 3 kg assumed mass rather than reused from the original 1 kg trajectory CSV.

2.2 Observation, action, and objective

The 49-dimensional observation concatenates six-vectors $q, \dot{q}, q_r, \dot{q}_r, (q_r - q), (\dot{q}_r - \dot{q})$, previous normalized residual (6), previous normalized nominal torque (6), and normalized time (1). Thus $6 \times 8 + 1 = 49$. Policies output a normalized six-dimensional continuous action, subsequently scaled by

$$\Delta\tau_{\max} = [5, 5, 5, 0.5, 0.5, 0.01]^T \text{ Nm}, \qquad \Delta\tau_\pi = \Delta\tau_{\max} \odot a_\pi, \quad a_\pi \in [-1, 1]^6. \tag{5}$$

These are *residual* limits, distinct from the full torque limits above. The action bounds were deliberately reduced after initial random-action rollout failures, especially for the lightly inertial final joint; all algorithm comparisons shown use the same final action specification. The evaluation episode has 150 steps of 20 ms; a held torque is

integrated over ten 2 ms numerical plant steps. The objective penalizes Cartesian position error, smaller joint position error, normalized residual magnitude, and action change. In the implementation the single-step reward has the form

$$r_t = -\left[\left(\frac{\|p(q) - p_r\|}{0.01\ \text{m}}\right)^2 + 0.1\ \text{mean}(e_q^2) + 0.01\ \text{mean}(a_t^2) + 0.01\ \text{mean}((a_t - a_{t-1})^2)\right]. \tag{6}$$

The first term directly emphasizes Tool0 Cartesian trajectory accuracy. The joint/error regularizers are numerically weaker and do not impose hard closed-loop robustness or torque-slew guarantees. Policy inputs and rewards are dimensioned or normalized as in the actual offline environment; no unseen trajectory or randomized payload is part of this first-screening experiment.

2.3 Algorithm-screening design

PPO is an on-policy clipped policy-gradient method; SAC is an entropy-regularized off-policy actor–critic method; TD3 uses a deterministic actor with twin critics, delayed actor updates, and target-policy smoothing. Their relative suitability cannot be inferred from these descriptions alone; it is measured here under matched *environment interactions* rather than matched gradient operations. All runs use seed 42, CPU policy optimization, 30,000 environment interactions, and deterministic evaluation every 3,000 interactions. PPO uses 300 rollout steps, 60-sample minibatches, 5 epochs, learning rate 3×10^{-4}. SAC uses replay buffer 30,000, learning starts at 600, batch 256, learning rate 3×10^{-4}, automated entropy coefficient. TD3 uses the same replay capacity/batch/learning starts, learning rate 10^{-3}, policy delay two, and exploration noise standard deviation 0.1 in normalized action units. Offline numerical dynamics dominate measured runtime; wall time is not isolated inference latency.

3 Training convergence and selection

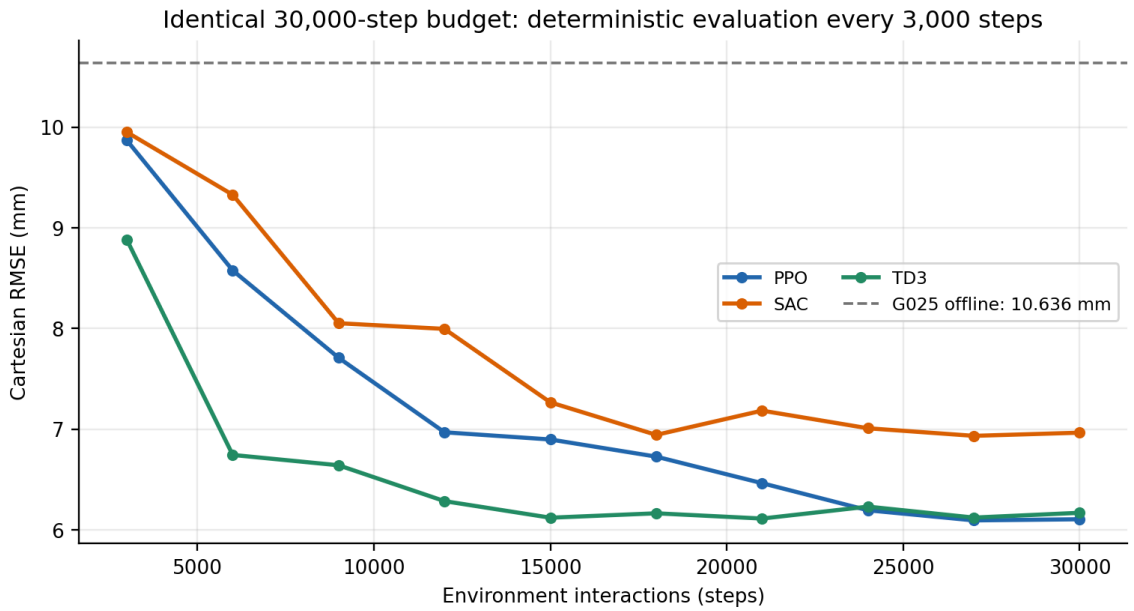


Figure 1: Deterministic Cartesian RMSE under an equal 30,000-interaction budget, evaluated every 3,000 steps. The dashed reference is zero-residual offline G025. Best-checkpoint selection is based on this same reference trajectory; it is not a generalization test.

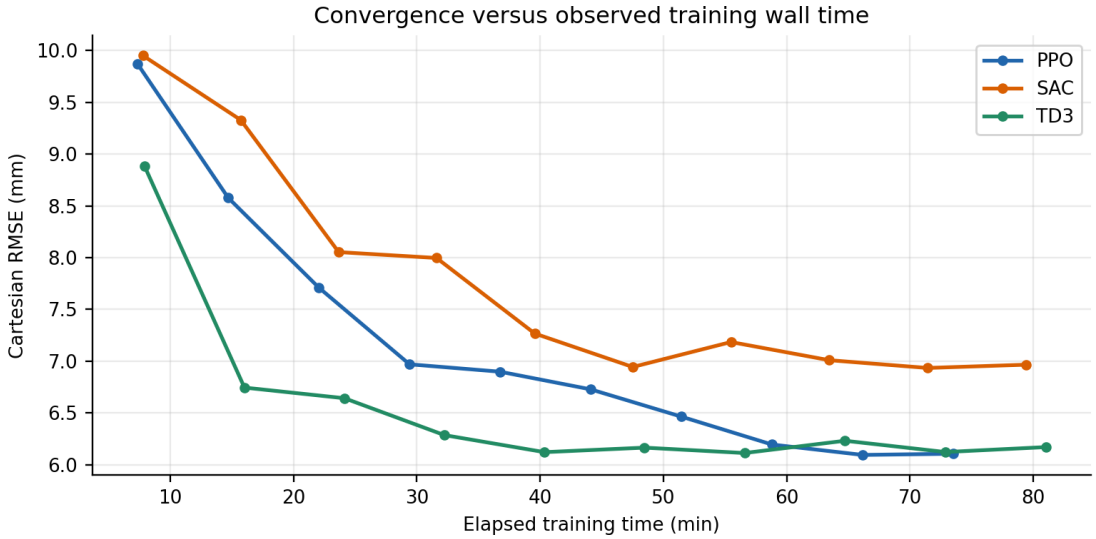


Figure 2: The same deterministic evaluations as a function of measured total training time; differences include optimization overhead and model-integration work, and are not pure neural-network training benchmarks.

Table 1: Convergence records from original evaluation CSVs; all values are mm.

Interactions	PPO	SAC	TD3
3,000	9.870	9.951	8.882
6,000	8.577	9.329	6.744
9,000	7.709	8.053	6.642
12,000	6.969	7.996	6.285
15,000	6.898	7.266	6.121
18,000	6.727	6.943	6.164
21,000	6.464	7.184	6.112
24,000	6.194	7.009	6.231
27,000	6.094	6.934	6.122
30,000	6.105	6.965	6.169

TD3 reached 6.744 mm by 6,000 interactions and 6.285 mm by 12,000; PPO first fell below 6.3 mm at 24,000. TD3 therefore approached the best measured error with fewer environment interactions. SAC reached 6.943 mm at 18,000 and attained its minimum 6.934 mm at 27,000, fluctuating near 7 mm late in training. This is a practical late-stage plateau, not a formal convergence guarantee. TD3's best checkpoint at 21,000 steps occurred before PPO's best at 27,000; PPO's absolute RMSE advantage is only 0.0183 mm (approximately 0.30% relative to TD3).

Table 2: All 150 evaluation steps completed. Best policy metrics are computed from saved trajectory CSVs; whole-vector residual RMS is the RMS over six joint channels and 150 times.

Algorithm	Best step	RMSE [mm]	Peak error [mm]	Final error [mm]	Final e_z [mm]	Residual RMS [Nm]
PPO	27,000	6.0939	7.956	5.884	-5.124	2.900
SAC	27,000	6.9339	8.488	6.285	-6.254	2.916
TD3	21,000	6.1122	7.589	5.918	-5.275	3.450

The zero-residual offline baseline is 10.6358 mm. Best-checkpoint relative reduction is 42.70% (PPO), 34.81% (SAC), and 42.52% (TD3). This is a *single-seed, single-trajectory* controlled comparison. Complete full-length episodes were required for these RMSE values; shortened/failed rollouts must not be ranked using partial-trajectory RMSE.

4 Cartesian transport accuracy and failure mode

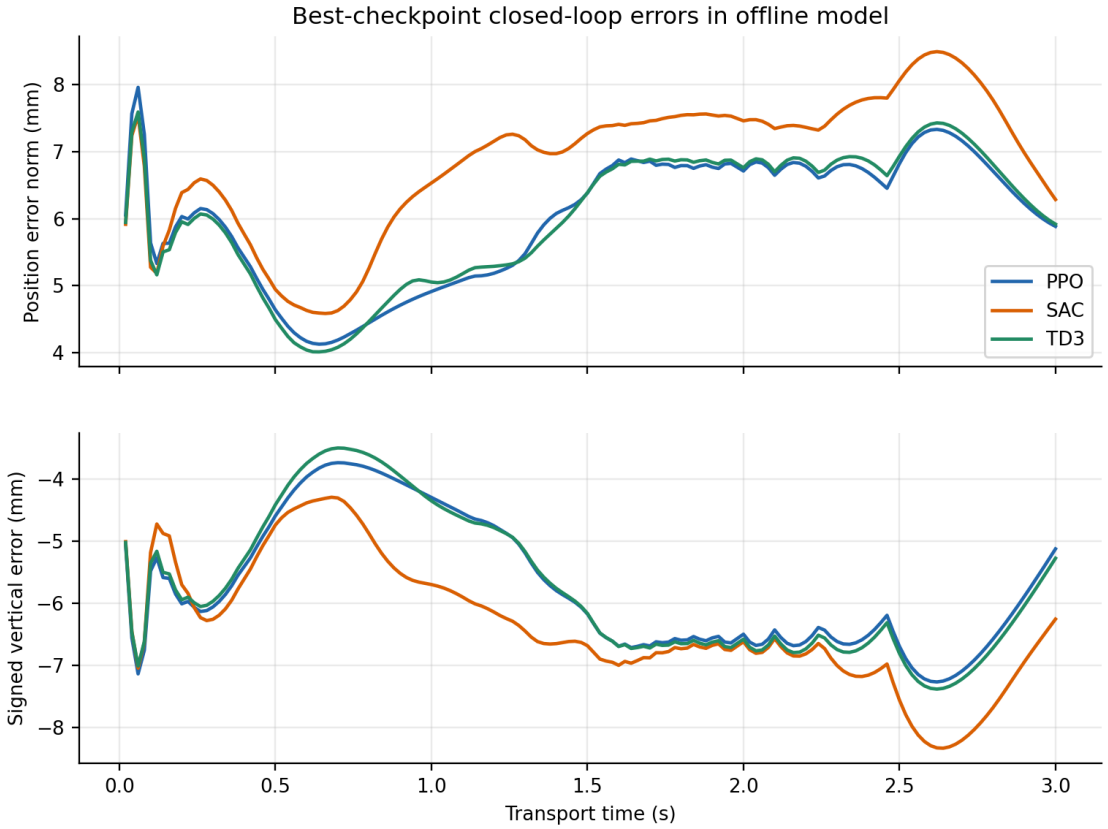


Figure 3: Best-checkpoint Cartesian error norm and signed Tool0 vertical error across all 150 matched control times. Negative signed vertical error indicates the tool remains below the prescribed reference.

All three policies reduce the model-mismatch-induced downward trajectory bias but do not eliminate it. For the 3-to-5 kg mismatch, the expected residual gravitational deficit produces a physically interpretable systematic vertical error. At $t = 3$ s, the best-policy signed vertical errors are PPO -5.124 mm, SAC -6.254 mm, and TD3 -5.275 mm. The remaining 5–6 mm vertical deficit should not be treated as proven table penetration: table collision/reaction forces are not part of the offline trajectory comparison, and the payload-to-table clearance was not independently measured.

5 Joint-resolved residual action: three-policy comparison

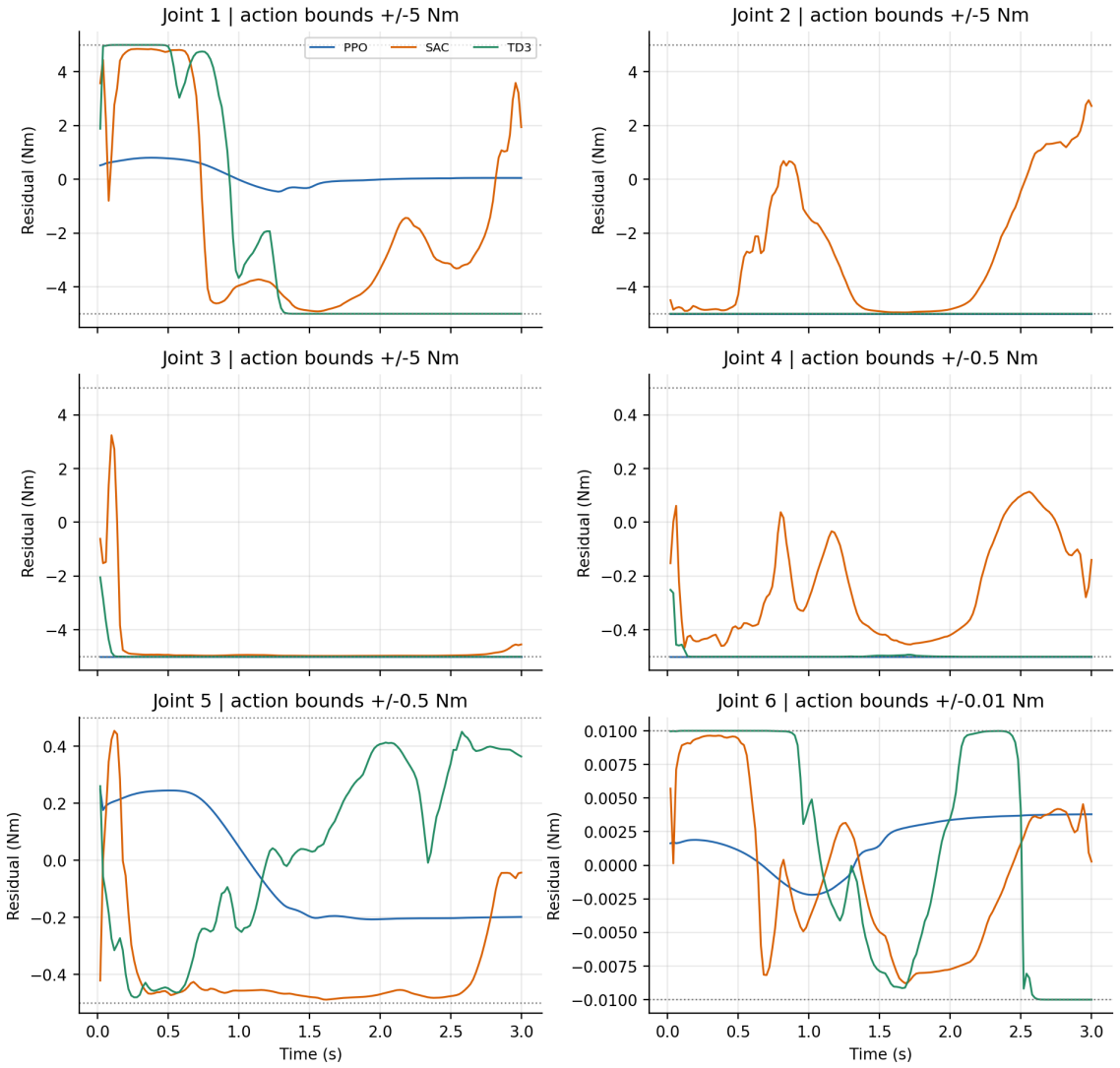


Figure 4: Six residual-torque time histories, each with its own action limits (dotted lines). All three algorithms are shown; the behavior of J1 sharply distinguishes PPO from TD3 and SAC.

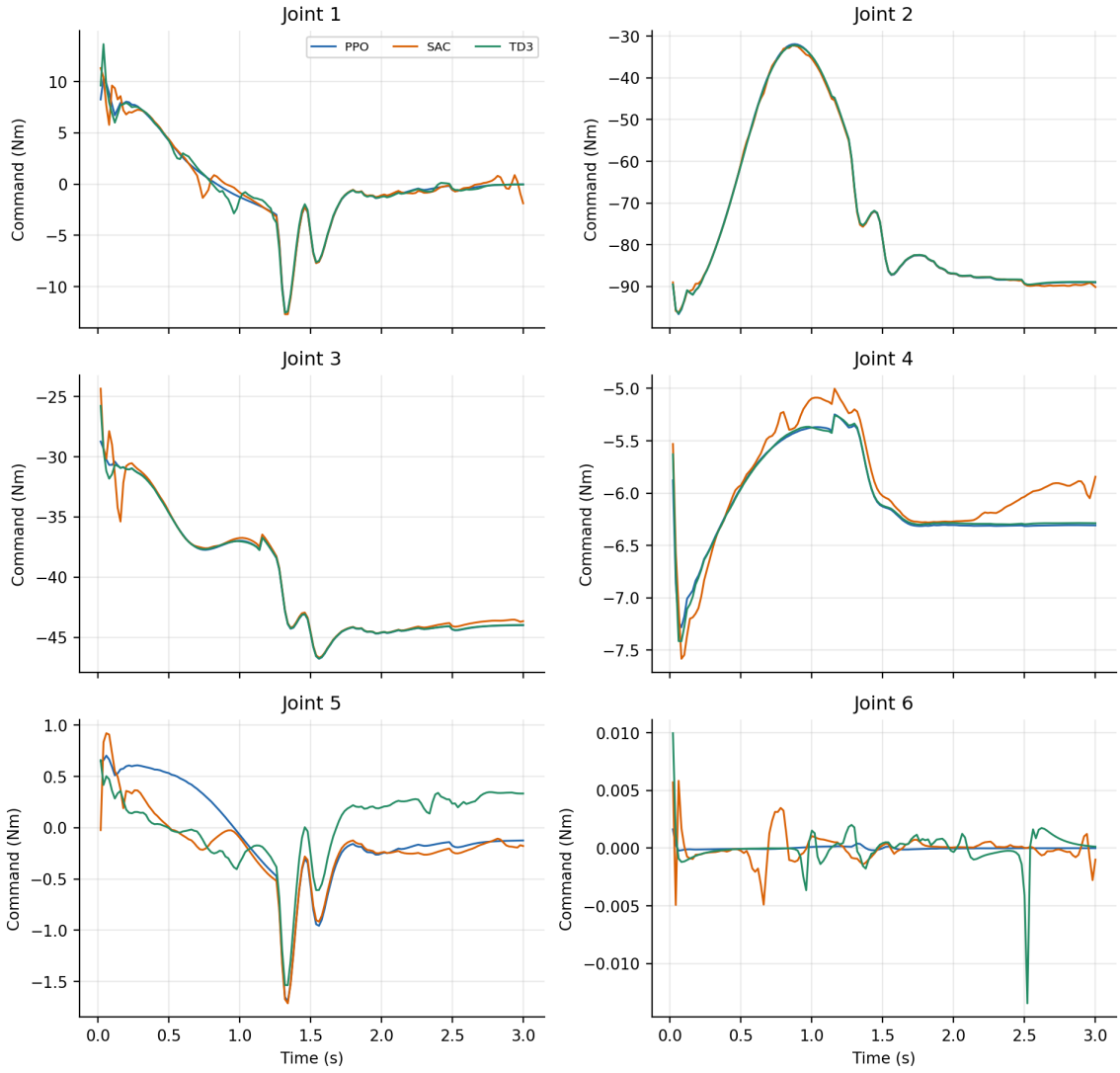


Figure 5: Full commanded joint efforts (nominal G025 plus residual) for all three best checkpoints. These are simulated commands, not measured motor currents or independently confirmed applied physical efforts.

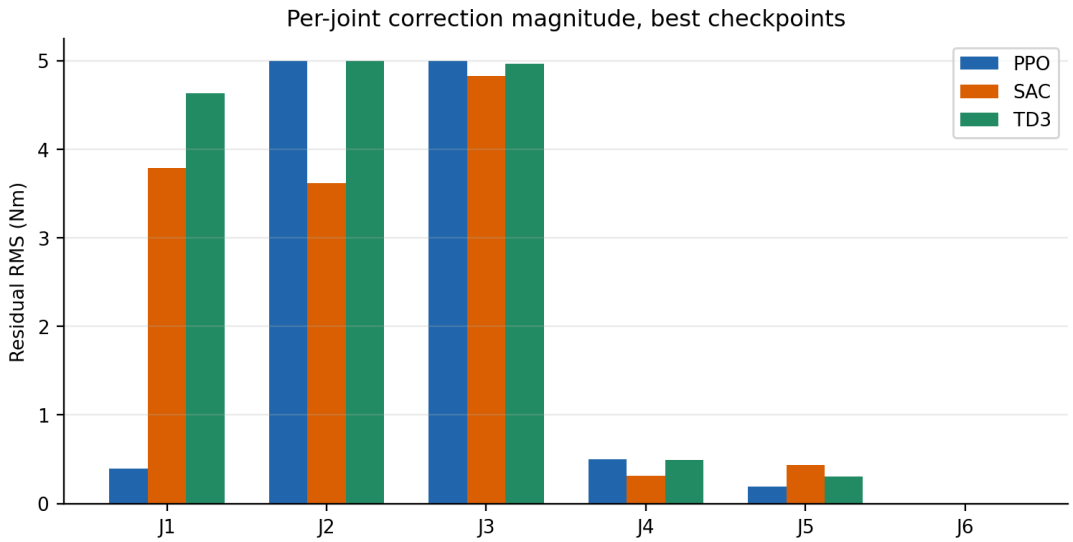


Figure 6: Joint-wise RMS residual effort. PPO uses markedly less J1 compensation than the other methods; the shoulder-lift and elbow outputs are strongly constrained by residual action bounds.

Table 3: Joint-resolved RMS residual torque (Nm), 150 samples.

Joint	PPO	SAC	TD3
J1	0.397	3.790	4.632
J2	5.000	3.620	5.000
J3	5.000	4.824	4.962
J4	0.500	0.311	0.496
J5	0.195	0.431	0.309
J6	0.003	0.006	0.008

Joint	PPO mean	SAC mean	TD3 mean
J1	+0.150	-1.357	-1.842
J2	-5.000	-2.697	-5.000
J3	-5.000	-4.667	-4.951
J4	-0.500	-0.250	-0.495
J5	-0.051	-0.384	+0.033
J6	+0.002	-0.000	+0.002

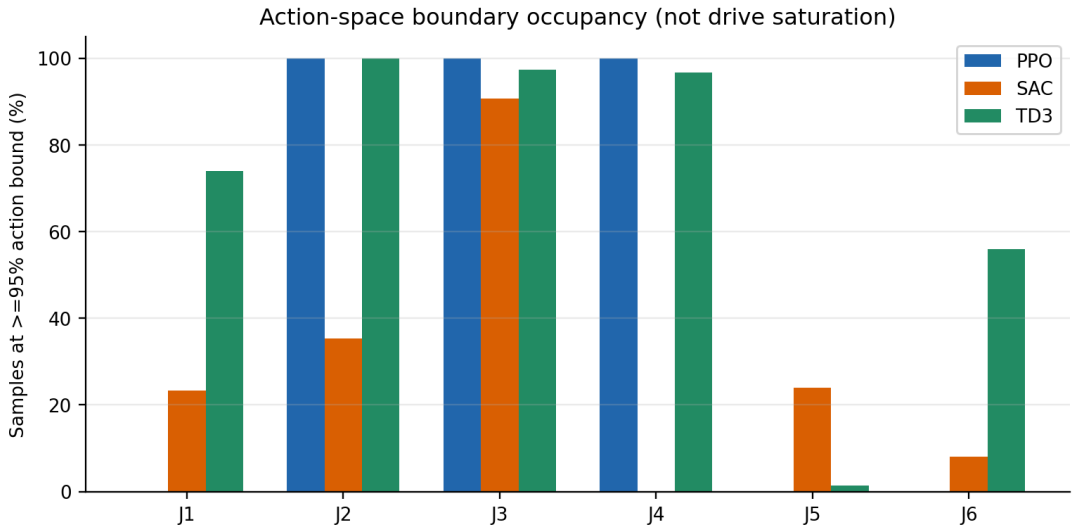


Figure 7: Percentage of best-checkpoint time samples with $|\Delta\tau_j| \geq 0.95\Delta\tau_{\max,j}$. This is *residual action-bound occupancy*, not full motor-command saturation.

Table 4: Residual action-bound occupancy (%).

Joint	PPO	SAC	TD3
J1	0.0	23.3	74.0
J2	100.0	35.3	100.0
J3	100.0	90.7	97.3
J4	100.0	0.0	96.7
J5	0.0	24.0	1.3
J6	0.0	8.0	56.0

The shoulder-lift (J2), elbow (J3), and wrist-1 (J4) actions frequently sit at negative limits. Importantly, zero logged *command-limit* saturation is consistent with high residual action-bound occupancy because these are two different constraints. The most discriminating difference is J1: PPO attains nearly the same Cartesian accuracy as TD3 while requesting a much smaller J1 correction. A larger residual norm is neither automatic evidence of a bad controller nor proof of physically necessary compensation; a joint-specific analytical comparison is warranted.

6 Physical interpretation: analytical mass mismatch

At the unchanged reference $q_r(t), \dot{q}_r(t), \ddot{q}_r(t)$, the theoretical 3-to-5 kg inverse-dynamics difference is

$$\Delta\tau_{\mathrm{ID}}(t) = \tau_{\mathrm{ID}}(q_r, \dot{q}_r, \ddot{q}_r; 5) - \tau_{\mathrm{ID}}(q_r, \dot{q}_r, \ddot{q}_r; 3) \tag{7}$$

$$= [M_5 - M_3]\ddot{q}_r + [c_5 - c_3] + [g_5 - g_3]. \tag{8}$$

The independent 150-sample calculation from `payload_dynamics.py` used the same 20 ms reference times, point-mass payload CoM $[0, 0, 0.025]^T$ m, and accepted trajectory. The resulting aggregate results, reproduced below, were supplied as numerical terminal output. The full per-time-step theoretical torque CSV was *not* uploaded, so no time-resolved policy-versus-theory correlation is asserted.

Table 5: Independent analytical torque difference, 5 kg minus 3 kg, relative to residual action amplitude (Nm).

Joint	Mean	RMS	Min	Max	Gravity RMS	Action bound
J1	-0.007	0.873	-2.815	+1.558	0.000	5.00
J2	-13.384	13.653	-15.712	-8.079	13.428	5.00
J3	-8.860	8.924	-10.097	-6.782	9.191	5.00
J4	-1.864	1.869	-2.080	-1.593	1.956	0.50
J5	-0.013	0.168	-0.619	+0.233	0.000	0.50
J6	+0.000	0.000	+0.000	+0.000	0.000	0.01

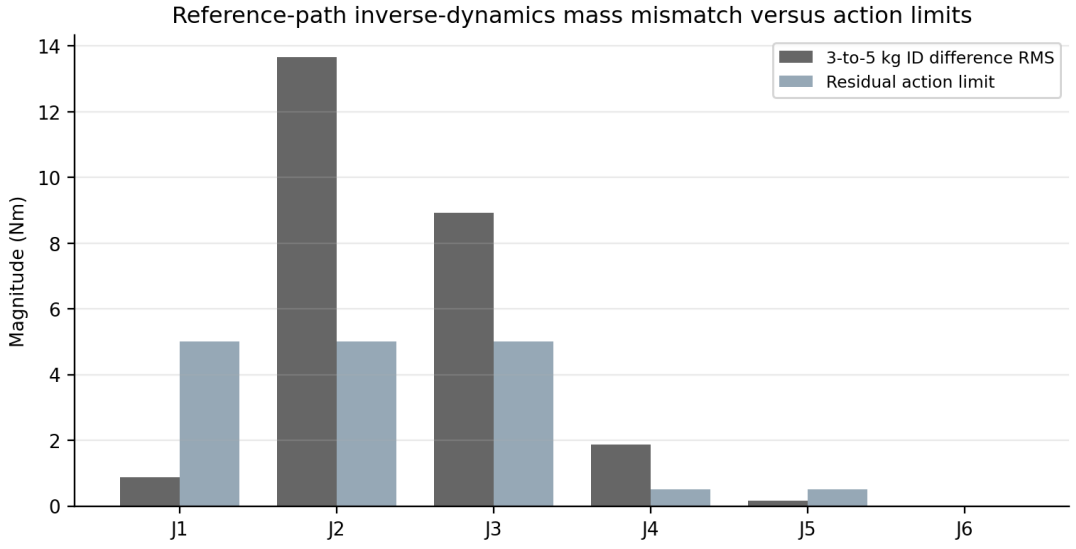


Figure 8: Aggregate inverse-dynamics torque discrepancy compared with the fixed residual action amplitude. The discrepancy is a reference-path feedforward difference, not the exact residual torque required by a closed-loop policy.

For J2 the analytical deficit ranges from −15.712 to −8.079 Nm, exceeding the available −5 Nm action throughout the reference. J3 requires −10.097 to −6.782 Nm, also exceeding its −5 Nm bound across the trajectory; J4 spans −2.080 to −1.594 Nm against a −0.5 Nm bound. This supports the *hypothesis* that residual action limits constrain achievable compensation, but does not prove the remaining Cartesian RMSE is solely due to these bounds. G025 feedback torque already reacts to tracking error; its interaction with the learned residual is essential.

J1 has almost zero mean analytical difference (−0.007 Nm), RMS 0.873 Nm, and a sign-changing range $[-2.815, 1.558]$ Nm; its gravity-only difference is zero. Large J1 residual corrections are therefore not explained by a sustained missing gravitational feedforward load. The additional J1 behavior in TD3/SAC could arise from interaction of tracking-state errors, cost shaping, coupled dynamics, and policy structure; characterizing it as *incorrect* requires counterfactual closed-loop testing or time-aligned theoretical torque histories. J5 mass-mismatch torque is small, and J6 is approximately zero in this constant-orientation reference.

7 Algorithm assessment, limitations, and next validation

7.1 Selection argument

PPO gives the lowest observed offline Cartesian RMSE and lower overall residual RMS. TD3 reaches comparable accuracy earlier in the interaction budget and has a deterministic actor appropriate for repeatable online residual inference. SAC improves on G025 but produces higher best-checkpoint RMSE within the same 30,000-step allocation. Because the PPO–TD3 RMSE gap is 0.0183 mm and only one training seed was tested, these data do *not* establish a statistically significant general ranking. For practical project progression, TD3 is selected as the first policy to attempt in ROS 2–Isaac Sim, with PPO retained as a fallback comparator.

7.2 Why no immediate additional offline optimization?

The analytical discrepancy establishes how reference-path feedforward torque should shift with mass and highlights residual bounds. Enlarging those bounds might improve the offline score; however, it changes the original training problem and may reintroduce unstable exploratory episodes. At this stage the more valuable question is whether a policy trained in a 2 ms numerical plant can improve the *actual ROS 2–PhysX* mismatched-load trial. Existing trained policies and their bounds are preserved for that transfer test.

7.3 Controlled Isaac Sim follow-up (not yet performed)

Use the stored best TD3 policy, the original G025 node/state sequencing, and the same physical 5 kg / assumed 3 kg condition. Maintain PRELOAD → RETURN → 2 s STABILIZE → TRACKING → FINAL_HOLD and activate residual actions only during TRACKING. Reconstruct the exact 49-feature vector (joint order, previous actions, nominal effort scaling, time normalization) before applying the 6D action. Preserve the reference, initial perturbation, full torque clipping, and evaluation window. Log Cartesian and joint tracking, six nominal/residual/final torques, timestamps, action clipping, step-to-step effort changes, and successful episode completion. If TD3 fails to transfer or causes inferior physics behavior, apply the saved PPO best checkpoint under the same controls. Future 5-to-3 kg and held-out 4 kg tests are not evidence in this report.

7.4 Threats to interpretation

- A single seed and one deterministic fixed reference do not establish variance, robustness, or out-of-distribution mass generalization.
- Best checkpoints were selected and assessed using the same reference trajectory; selection and independent performance validation are separate questions.
- Equality of environment interactions does not equalize gradient updates, runtime, or computational budget across PPO/SAC/TD3.
- Offline model and PhysX share geometry but differ in runtime integration and possibly effective rigid-body inertia, friction, state timing, and ROS communication; current RL gains are not yet PhysX gains.
- The theoretical mass-mismatch inverse dynamics is a reference-feedforward quantity, not the exact optimal residual for an already feedback-controlled off-reference robot.

- No hard collision/clearance guarantee, hardware actuator validation, or repeated success-rate estimate is inferred.

8 Conclusion

The G025 baseline established in the previous report was degraded by the 3-to-5 kg payload-model discrepancy. Under a matched offline training budget, PPO, SAC, and TD3 reduced its 10.636 mm Cartesian RMSE to 6.094, 6.934, and 6.112 mm respectively at their saved best checkpoints. Joint-by-joint records show that PPO and TD3 reach similar trajectory accuracy using different residual torque distributions, especially about shoulder pan. The independently computed inverse-dynamics discrepancy explains why J2, J3, and J4 often request action-bound corrections, while caution is required in interpreting the residual as direct identification of missing feedforward. TD3 offers the fastest observed early reduction and is therefore the policy chosen for the first ROS 2–PhysX transfer experiment, retaining PPO as a controlled alternative. *The principal next result is physical-simulator transfer, not further tuning of the offline score.*

A Reproducibility inventory

The accompanying Overleaf ZIP contains `main.tex`, original learning-curve CSVs, best-checkpoint evaluation CSVs for all three algorithms, their run summaries, matched/mismatched G025 PhysX baseline CSVs, and rendered source-data figures. The previous 10 October nominal-control report is a distinct earlier-stage document, not silently altered or republished here. Analytical mass-mismatch summary in this document was transcribed from the supplied terminal output; the actual `payload_torque_difference.csv` was not provided. Numerical values are case-specific and should not be interpreted as experimental uncertainties.