Master's Thesis Academic year 2026

Control of an Underactuated Self-Balancing Robotic Platform with Variable Payloads and Dynamic Uncertainties for Human–Robot Interaction

Ángel Muñoz Rocha

(Student ID No.: 82523430)

Supervisor: Professor Toshiyuki Murakami
Co-supervisor: Professor Teresa Zielińska

September 2026

Keio University
Graduate School of Science and Technology
School of Integrated Design Engineering

Abstract. This thesis addresses the stabilization and control of underactuated self-balancing wheeled robots operating with variable payloads in physical human–robot collaboration scenarios. The research focuses on the Two-Wheeled Forklift Robot (TWFR), a differential-drive platform with a vertically actuated fork mechanism that introduces time-varying system parameters during payload manipulation. A comprehensive control framework is developed integrating Nonlinear Model Predictive Control (NMPC) with disturbance observation techniques. The NMPC, implemented through *acados* using Real-Time Iteration schemes, simultaneously regulates longitudinal velocity and pitch stabilization while enforcing operational constraints. Two disturbance observers—a Disturbance Observer (DOB) for external forces and a Reaction Disturbance Observer (RDOB) for payload-induced torques—provide real-time estimates of payload mass and center of gravity without physical sensors, continuously updating the NMPC internal model for adaptive control during manipulation. A novel collision detection mechanism analyzes prediction error derivatives to identify unexpected external interactions despite varying payloads, while a supervisory Finite State Machine coordinates operational modes across normal operation, payload handling, and collision response scenarios. Comprehensive simulations validate the proposed framework with payloads up to 10 kg under diverse conditions, including trajectory tracking, external disturbances, and collision events. Results demonstrate stable balancing, accurate tracking, effective disturbance rejection, reliable collision detection, and real-time computational feasibility, advancing self-balancing robot control by addressing underactuation, payload variability, and interaction safety within a unified predictive framework suitable for real-world collaborative environments.

Keywords: Self-balancing robot, wheeled inverted pendulum, underactuated control, nonlinear model predictive control, disturbance observer, variable payload, collision detection, human–robot interaction

Contents

List of Figures

List of Tables

Chapter 1 Introduction

In recent years, modern industrial environments have experienced a growing integration of robotic systems designed to operate in close proximity to humans. This trend is driven by demographic challenges such as declining birth rates and labor shortages, as well as by the suitability of robots for repetitive, physically demanding, or ergonomically harmful tasks. By delegating such activities to robotic platforms, human workers can focus on roles that require creativity, decision-making, and higher-level cognitive skills.

The expanding deployment of robots in human-centric environments has increased the demand for mobile platforms capable of operating safely and reliably under dynamic and uncertain conditions. As robotic systems move beyond isolated industrial cells into shared workspaces, homes, and public environments, physical human–robot collaboration becomes a central requirement. In these scenarios, robots must not only navigate and manipulate objects effectively but also adapt to unpredictable interactions with humans and changes in the surrounding environment. Within this context, ensuring stable and robust control of mobile robotic platforms subject to external disturbances and varying payloads is a fundamental challenge, particularly when safe physical interaction is required.

Self-balancing wheeled robots represent an inherently unstable class of mobile platforms that offer notable advantages over statically stable alternatives, including improved maneuverability, higher energy efficiency, and enhanced accessibility in constrained environments. However, these benefits come at the cost of increased control complexity due to their underactuated nature, strongly nonlinear dynamics, and sensitivity to external perturbations and payload variations. When such platforms are intended for physical human–robot collaboration, stable operation under uncertain dynamic conditions becomes essential to guarantee safety and reliability, especially during tasks involving load transportation.

This thesis focuses on the design, analysis, and implementation of a robust control framework for a two-wheeled differential-drive inverted pendulum robot equipped with a vertically actuated load manipulation mechanism, referred to as the Two-Wheeled Forklift Robot (TWFR), developed at the Keio Murakami Laboratory. The platform consists of a main chassis located at the wheel axis and a prismatic actuator that extends and retracts a fork-like end effector for payload handling. This configuration allows the robot to transport objects with varying mass and inertial properties, reflecting realistic use cases in collaborative logistics, personal assistance, and shared workspace scenarios where close physical interaction between humans and robots is required.

The control problem is further complicated by the platform's limited sensing capabilities. Available measurements are restricted to wheel encoders, a linear actuator position sensor, an angular sensor measuring the orientation of the fork joint, and an inertial measurement unit (IMU) providing the pitch angle. The resulting underactuated system dynamics, combined with time-varying parameters induced by payload changes and geometric reconfiguration during fork manipulation, pose significant challenges for controller design. Additionally, the control system must ensure robust performance in the presence of modeling uncertainties, external disturbances such as surface irregularities and human-induced forces, and abrupt dynamic changes caused by collisions or sudden load variations, while maintaining stability and operational safety during physical human–robot interaction.

This work contributes to the field of mobile robotics by proposing a comprehensive control methodology that explicitly addresses the challenges associated with self-balancing platforms operating under variable payload conditions and dynamic uncertainties in human-shared environments. The proposed approach integrates rigorous theoretical analysis with experimental validation, aiming to enhance system stability, adaptability, and safety in physical human–robot collaboration scenarios, thereby facilitating the deployment of such platforms in real-world applications.

The remainder of this thesis is organized as follows. Chapter 2 reviews the relevant literature on wheeled inverted pendulum robots, with particular emphasis on control strategies and their limitations in collaborative settings. Chapter 3 defines the objectives of this research, outlining the performance criteria and constraints of the control system. Chapter 4 presents the overall methodology adopted in this work. Chapter 5 details the system modeling and derivation of the dynamic equations used for control design. Chapter 6 describes the implemented control strategies, including nonlinear model predictive control and disturbance observer techniques. Chapter 7 presents the simulation setup and results, while Chapter 8 discusses the obtained findings and their implications. Finally, Chapter 9 concludes the thesis and outlines directions for future research.

Chapter 2 Literature Review

2.1 State of the Art

This section reviews the literature relevant to the development of control frameworks for self-balancing robotic platforms, with particular emphasis on two-wheeled inverted pendulum systems. The discussion covers model predictive control strategies, disturbance observation techniques, adaptive control approaches, and architectures for physical human–robot interaction. Emphasis is placed on the evolution of these methodologies and on identifying limitations that motivate further research.

2.1.1 Model Predictive Control for Wheeled Inverted Pendulum Systems

Model Predictive Control (MPC) has emerged as a powerful framework for stabilizing underactuated wheeled inverted pendulum robots due to its ability to explicitly handle system constraints and predict future behavior over finite horizons. Early implementations demonstrated the feasibility of real-time nonlinear MPC using Sequential Quadratic Programming to solve constrained optimization problems [1]. These approaches employed discrete-time nonlinear state-space models and Gauss–Newton Hessian approximations to achieve sampling rates on the order of 40 Hz, enabling autonomous swing-up and effective rejection of manual disturbances. Nevertheless, the reliance on deterministic models with fixed parameters limited robustness in the presence of stochastic uncertainties.

Comparative studies between Linear Quadratic Regulator and MPC strategies have highlighted the superior performance of MPC in terms of settling time, overshoot reduction, and control effort optimization [2]. Despite these advantages, such implementations typically assumed constant inertial parameters and assessed robustness primarily through impulse disturbances rather than continuous parametric variations. As a result, system behavior under sustained payload changes remains insufficiently characterized.

Continuous-time optimal MPC formulations based on orthonormal function parameterizations of control trajectories have demonstrated improvements over classical PID controllers [3]. However, these approaches continued to rely on linearized plant models and static payload assumptions. The absence of explicit mechanisms to accommodate mass and center-of-gravity variations represents a recurring limitation.

More recent work integrating MPC with reduced-order disturbance observers has addressed variable payloads and terrain irregularities through constrained quadratic programming [4]. By generating corrective actuator offsets based on estimated disturbances, these methods outperformed conventional linear quadratic regulators under asymmetric loading conditions. Nonetheless, their dependence on stationary equilibrium assumptions limited adaptability during highly dynamic operations involving simultaneous mobility and payload manipulation.

2.1.2 Adaptive Model Predictive Control and Parameter-Varying Formulations

Discrete payload variations have motivated the development of adaptive MPC strategies that explicitly update predictive models in response to changing system parameters. Dual-loop architectures combining Adaptive MPC with Linear Parameter-Varying formulations have employed mass-dependent scheduling of local linear models [5]. Time-varying Kalman filters ensured estimator consistency as system dynamics evolved, resulting in improved performance relative to PID and feedback linearization approaches under representative pick-and-place payload changes. However, the virtual link abstractions used to simplify multi-link dynamics assumed sufficiently fast internal dynamics, which may not hold during complex manipulation tasks.

Alternative formulations based on Predictive Functional Control using polynomial basis functions and coincidence points have reduced computational complexity compared to classical MPC [6]. Explicit torque constraint handling prioritized pitch stabilization during actuator saturation, while real-time internal model updates based on reaction torque estimation enabled robust operation under varying payloads. Although this approach outperformed conventional cascade controllers, it required careful phase compensation tuning and relied on specific assumptions regarding arm configurations, potentially limiting generalization.

These limitations become more pronounced in scenarios involving continuous payload manipulation, where platforms must simultaneously maintain balance, track trajectories, and respond to smoothly varying loads during physical human–robot interaction. Existing adaptive MPC formulations generally address discrete parameter changes or rely on offline identification, leaving seamless online adaptation during ongoing manipulation tasks largely unexplored.

2.1.3 Disturbance Observation and Real-Time Parameter Estimation

Disturbance observers play a central role in enhancing robustness for self-balancing platforms by estimating modeling errors, external forces, and parameter variations. The Synthesized Pitch Angle Disturbance Observer estimates integrated wheel and pitch disturbances, improving stability when combined with cascaded control architectures [7]. This approach effectively mitigates destabilizing effects arising from center-of-gravity shifts within fixed control structures.

Reaction Torque Observer techniques enable sensorless estimation of payload-induced torques through motor current measurements and inverse dynamics models [7, 8]. For platforms equipped with active arms or lifting mechanisms, such estimations support compensation strategies based on postural reference adjustment. While eliminating physical force sensors reduces system complexity, careful tuning is required to prevent oscillations during highly dynamic transitions.

The integration of multiple disturbance observers within predictive control frameworks has further enhanced robustness to dynamic uncertainties [6]. Combining pitch angle and reaction torque observers enables compensation for both ground-induced disturbances and payload variations, facilitating real-time internal model refinement. Despite these advances, the systematic integration of disturbance observation with nonlinear MPC for continuous parameter adaptation remains limited. Observer outputs are often used for fixed compensation or discrete model switching rather than direct updates of predictive models.

2.1.4 Adaptive Control Strategies for Variable Payloads

Beyond predictive control, various adaptive strategies have been proposed to handle variable payloads and uncertain dynamics. Adaptive nonlinear proportional–derivative controllers with time-varying gains have demonstrated improved rejection of external impulses compared to fixed-gain schemes [9]. Offline parameter optimization using genetic algorithms reduces tuning effort but generally assumes nominal system parameters, limiting effectiveness under significant parameter drift.

Neural-network-based adaptive controllers provide online estimation of unknown dynamics, including centrifugal, Coriolis, and gravitational effects [10]. While Lyapunov-based formulations ensure stability, these approaches typically assume slowly varying mechanical configurations and do not explicitly address time-varying center-of-mass shifts during manipulation.

Model Reference Adaptive Control schemes have also been applied to self-balancing robots, employing Lyapunov-based adaptation laws to handle nonlinearities and disturbances [11]. Although effective for balance stabilization, the use of simplified reference models and reduced-order dynamics may limit positioning accuracy during payload handling.

Machine learning approaches integrating payload classification with adaptive fuzzy control have enabled discrete payload management without physical load sensors [12]. However, the lack of analytical stability guarantees limits applicability in safety-critical human–robot interaction contexts. Overall, these adaptive strategies tend to focus on discrete classification or slow parameter adaptation rather than continuous integration with predictive control.

2.1.5 Physical Human–Robot Interaction and Compliance Control

Physical human–robot collaboration has driven the development of control architectures emphasizing safety, compliance, and adaptability. Position-based admittance control transforms external forces into reference trajectory adjustments via second-order filters, facilitating cooperative tasks involving mobile bases and manipulators [13]. However, limited control bandwidth introduces trade-offs between compliance and stabilization under high-frequency disturbances.

Repulsive Compliance Control offers an alternative approach by opposing estimated payload reaction torques to rebalance the robot through postural adjustments [7]. This strategy prevents excessive longitudinal acceleration caused by unmodeled center-of-mass offsets without requiring explicit payload parameter identification.

Model-based center-of-gravity compensation using reaction torque observation has also been proposed for platforms with active arms [8]. By statically computing pitch references that align the total center of gravity above the wheel axle, faster convergence is achieved compared to dynamic compliance methods. Nevertheless, reliance on small-angle assumptions and prior parameter identification may reduce robustness during aggressive maneuvers.

The integration of MPC with impedance control through Linear Parameter-Varying formulations has introduced Safety Control Barrier Functions as inequality constraints [14]. While enabling high-frequency convex optimization, these approaches typically rely on linearized dynamics and simplified human interaction models. Similar limitations arise in MPC-based force control frameworks for manipulation tasks [15], where small orientation assumptions restrict applicability to strongly nonlinear balancing scenarios.

2.1.6 Collision Detection and Safety-Critical Control

Collision detection for self-balancing platforms presents unique challenges due to underactuation and the presence of balancing-induced disturbances. Observer-based force estimation methods developed for fully actuated manipulators have limited direct applicability in this context.

Proprioceptive force estimation techniques based on virtual work principles and planar linkage models have been proposed to avoid external force sensors [16]. When combined with adaptive admittance control, these methods reduce oscillations during continuous environmental interaction. However, they are primarily designed for sustained contact rather than discrete collision events requiring rapid protective responses.

An open research gap lies in leveraging predictive control frameworks themselves for collision detection. Deviations between predicted and measured system behavior within MPC formulations may provide informative indicators of anomalous events, even under varying payload conditions and active manipulation.

2.1.7 Cooperative Transportation and Force Control

Self-balancing platforms capable of exerting controlled external forces have been explored in cooperative transportation scenarios [17]. State-space formulations treating interaction forces as estimated states enable human-initiated motion through observer-based force control. Nonetheless, linearized models and assumptions regarding force application points limit robustness, particularly under significant payload-induced parameter changes. Low-frequency oscillations observed during transport further highlight the limitations of fixed-parameter linear models.

2.1.8 Summary and Positioning

The reviewed literature demonstrates significant progress in the control of wheeled inverted pendulum robots, with MPC emerging as a particularly effective framework due to its predictive and constraint-handling capabilities. Disturbance observers and adaptive control strategies have enhanced robustness to payload variations and external forces.

However, key challenges remain unresolved. Most MPC-based approaches rely on fixed-parameter models or discrete adaptation schemes that do not accommodate continuous payload variations during active manipulation. The integration of disturbance observers with nonlinear MPC for real-time model adaptation remains limited, and reliable collision detection under varying payload conditions is still an open problem.

These gaps motivate the development of integrated control frameworks that combine nonlinear MPC with real-time disturbance-based parameter estimation, specifically targeting self-balancing platforms with active payload manipulation capabilities. Such frameworks must operate robustly in human-centric environments and exploit the predictive structure of MPC for enhanced safety and interaction awareness. This objective defines the scope of the present work.

2.2 Previous Work

The research presented in this thesis builds upon prior work conducted within the same research group on the control of self-balancing wheeled robotic platforms. Earlier studies investigated the stabilization and trajectory tracking of two-wheeled inverted pendulum robots operating under nominal conditions and fixed mechanical configurations.

In this prior work, classical and model-based control strategies were employed to achieve stable balancing and trajectory tracking in structured environments. Disturbance observers and proportional–derivative controllers were implemented and validated in simulation, demonstrating the feasibility of reliable control for underactuated systems. Mass estimation techniques were developed, and trajectory tracking in both horizontal and vertical directions was successfully demonstrated.

Despite establishing a solid foundation, several limitations were identified. Human–robot interaction scenarios were not explicitly considered, validation was restricted to simulation without experimental verification on physical hardware, and the rigidity of the linear actuator mechanism reduced the effectiveness of observer-based mass estimation when applied to the real platform. Moreover, fixed-parameter control strategies did not account for dynamically changing system parameters or unexpected human interactions.

The present thesis extends this earlier work by explicitly addressing variable payload conditions and dynamic uncertainties within the control design. Building upon the established modeling and control framework, the research advances toward more adaptive, robust, and safety-oriented strategies suitable for real-world human-centric environments.

Chapter 3 Objectives

3.1 Research Motivation

The deployment of mobile robotic platforms in human-centric environments is a central objective in modern robotics, with applications in logistics, healthcare, and collaborative manufacturing. Within this domain, self-balancing wheeled platforms offer advantages in maneuverability, energy efficiency, and compact design. However, their inherent instability and underactuated dynamics introduce fundamental control challenges, which become particularly pronounced during physical human–robot interaction.

Although existing control strategies achieve reliable performance under nominal conditions, their effectiveness degrades when exposed to the dynamic uncertainties typical of real-world operation. In particular, payload manipulation tasks frequently involve objects with unknown or time-varying mass and inertial properties. Vertically actuated payload mechanisms induce continuous shifts in the system center of gravity and inertia, significantly altering the platform dynamics. Most existing approaches assume fixed parameters or discrete payload changes, limiting their applicability during continuous manipulation.

In addition, self-balancing platforms operating in shared environments are subject to modeling inaccuracies, surface irregularities, actuator nonlinearities, and external disturbances. Conventional robust control strategies often rely on conservative designs or high-gain feedback, which may compromise performance or introduce oscillatory behavior. This motivates control architectures capable of actively estimating and compensating disturbances while maintaining stability and responsiveness.

A further critical challenge arises from the need to safely handle unexpected human interactions such as collisions or unintended contact. For self-balancing systems, continuous balancing-induced forces must be distinguished from genuine external interactions, a task complicated by payload-dependent changes in system response. Existing collision detection methods developed for fully actuated systems are not directly applicable to underactuated balancing platforms with active payload manipulation.

These challenges are strongly interrelated. Payload variations influence disturbance rejection and collision detection, while safety responses must remain effective across different loading conditions. Addressing stabilization, payload adaptation, and interaction safety within a unified control framework therefore constitutes the primary motivation of this research.

3.2 Research Objectives

The objective of this thesis is to develop, implement, and validate a control framework for underactuated self-balancing wheeled platforms that enables robust stabilization, adaptability to variable payloads, and safe operation during close human interaction.

To this end, the research focuses first on the development of a parametric dynamic model of the Two-Wheeled Forklift Robot that captures the coupled dynamics of longitudinal motion, pitch behavior, and vertical fork actuation under varying payload conditions. The model balances physical fidelity with computational tractability in order to support real-time predictive control.

Building upon this model, a nonlinear Model Predictive Control strategy is designed to stabilize the platform around its unstable upright equilibrium while tracking operator-defined longitudinal velocity references. The controller explicitly accounts for system constraints and nonlinear dynamics, and operates robustly in the presence of modeling uncertainties and external disturbances at control rates suitable for fast mechanical dynamics.

To enhance robustness and adaptability, disturbance observer techniques are integrated to estimate external disturbances and payload-induced effects without relying on additional physical sensors. These observer outputs are exploited to improve the internal predictive model online, enabling adaptive behavior during continuous payload manipulation.

In parallel, mechanisms for detecting unexpected external interactions are developed, with particular emphasis on distinguishing genuine collisions from normal balancing dynamics under varying payload conditions. A supervisory safety logic is incorporated to manage appropriate response strategies, prioritizing stabilization and human safety upon detection of anomalous behavior.

The proposed framework is validated through extensive simulation studies covering nominal operation, variable payload scenarios, external disturbances, and collision events. In addition, the computational requirements of the overall control architecture are analyzed to ensure feasibility for real-time implementation on embedded hardware platforms.

3.3 Scope and Limitations

Chapter 4 Approach and methodology

This chapter presents the overall methodology adopted for the development of the control framework for the Two-Wheeled Forklift Robot operating under variable payload conditions and dynamic uncertainties. The approach systematically progresses from system modeling to control design, safety management, implementation, and validation, ensuring consistency with the research objectives and the practical constraints of underactuated self-balancing platforms.

4.1 Overall Research Approach

The methodology is structured into five interrelated phases: system analysis and modeling, control architecture design, safety and interaction management, implementation and computational framework, and validation and performance assessment. Each phase builds upon the outcomes of the preceding stages to ensure coherent development and traceability between theoretical design and validation results.

4.1.1 System Analysis and Modeling

The control framework is founded on an accurate representation of the platform dynamics. The complete multi-body dynamics of the system are derived using Lagrangian mechanics, capturing the coupled behavior of the wheeled base, main chassis, vertically actuated fork mechanism, and payload. The formulation explicitly accounts for non-slip wheel constraints, prismatic fork actuation, and time-varying parameters arising from payload manipulation and geometric reconfiguration.

To enable real-time predictive control, a reduced-order model is derived by approximating the system as an equivalent rigid body characterized by total mass, global center of gravity, and equivalent rotational inertia. This simplification is justified by the observation that the dominant dynamics relevant to balance and longitudinal motion occur at frequencies below those associated with the fork mechanism. The reduced model preserves the essential coupling between actuation and platform motion while maintaining computational tractability.

System parameters are identified through a combination of direct measurement, CAD-based extraction, and system identification procedures. Model validity is assessed by comparing predicted and simulated responses under representative operating conditions, with particular emphasis on pitch dynamics and the influence of fork position.

4.1.2 Control Architecture Design

The control architecture adopts a hierarchical structure centered on a nonlinear Model Predictive Controller responsible for simultaneous pitch stabilization and longitudinal velocity regulation. Operator-defined velocity references are tracked while maintaining the platform near the unstable upright equilibrium and respecting physical and safety-related constraints.

The optimal control problem is formulated in continuous time and discretized for numerical solution. Design choices include the selection of prediction horizon length, nonlinear least-squares cost formulation emphasizing pitch-related states, and explicit state and input constraints reflecting actuator limits and safe operating regions. Stabilization objectives are prioritized to ensure balance maintenance when conflicts arise between tracking performance and stability.

Real-time feasibility is achieved through the Real-Time Iteration (RTI) scheme, which performs a single Sequential Quadratic Programming iteration per control cycle. The nonlinear problem is approximated locally by a quadratic program, enabling efficient solution while retaining sensitivity to nonlinear dynamics.

Robustness is enhanced through the integration of two disturbance observers. A Disturbance Observer estimates external forces and modeling errors acting on the main platform dynamics, while a Reaction Disturbance Observer applied to the fork actuation estimates payload-induced torques and reaction forces. These observers reconstruct unmeasured disturbances using inverse dynamics and available sensor measurements, with filtering applied to balance noise rejection and disturbance tracking bandwidth.

Payload mass and center-of-gravity position are inferred from observer estimates using force and moment balance relationships. When the fork is in motion, vertical reaction forces enable mass estimation; when stationary, reaction torques provide center-of-gravity information. A switching logic selects the appropriate estimation mode based on fork motion. The estimated parameters are used to update the internal predictive model, enabling adaptation to varying payloads without dedicated payload sensors.

4.1.3 Safety and Interaction Management

Safe operation during close human interaction requires reliable detection of unexpected external contacts. Collision detection is achieved by exploiting the predictive nature of the NMPC framework. Deviations between predicted and measured states are monitored, and the rate of change of prediction errors is evaluated to emphasize abrupt disturbances characteristic of collisions.

This approach inherently accounts for expected system behavior, including payload-dependent dynamics, and is less sensitive to slow-varying modeling errors. The method does not require additional sensing infrastructure and remains compatible with the underactuated nature of the platform.

A supervisory Finite State Machine coordinates operational modes, including nominal tracking, payload manipulation, collision response, and fault handling. Mode transitions are triggered by events such as collision detection or task completion. The FSM modifies reference signals, cost function weights, and constraint sets to enforce appropriate behavior in each mode, prioritizing stabilization and safety during abnormal conditions.

4.1.4 Implementation and Computational Framework

The control framework is implemented using a combination of MATLAB and Simulink for development and simulation, with CasADi employed for symbolic formulation of the optimal control problem and automatic differentiation. The acados framework serves as the core solver, providing efficient real-time solution of the structured nonlinear optimal control problem using the RTI scheme.

The NMPC implementation employs Sequential Quadratic Programming with Gauss–Newton Hessian approximation, partial condensing, and interior-point methods for quadratic subproblems. System dynamics are discretized using an integration scheme suitable for stiff nonlinear systems. These design choices reflect a balance between numerical robustness, computational efficiency, and control performance.

A modular software architecture separates the NMPC, disturbance observers, collision detection, and supervisory logic into well-defined functional blocks, facilitating independent testing and future extensibility.

4.1.5 Validation and Performance Assessment

The proposed framework is validated through extensive simulation studies using a high-fidelity environment that incorporates the full multi-body dynamics as the plant, actuator saturation and friction effects, sensor noise, and sampling constraints representative of physical hardware. The reduced-order model used by the NMPC is intentionally mismatched to the simulated plant to assess robustness to modeling uncertainty.

Test scenarios include nominal operation with varying payloads, reference tracking with diverse velocity profiles, external disturbances such as impulse forces and sustained pushes, collision events under different loading conditions, and combined scenarios involving simultaneous payload manipulation and disturbances.

Performance is evaluated using quantitative metrics addressing stability, tracking accuracy, control effort, parameter estimation accuracy, collision detection reliability, and computational execution time. Comparative studies against baseline control strategies provide contextual assessment of the proposed approach.

4.2 Methodological Considerations

Several methodological considerations merit discussion. The use of a simplified predictive model while validating against a high-fidelity plant intentionally introduces model mismatch, reflecting practical deployment conditions. Although validation is performed primarily in simulation, the inclusion of realistic actuator and sensor effects provides a stringent test of robustness. Emphasis on computational efficiency ensures feasibility for embedded implementation. Finally, the integration of safety supervision and collision detection reflects a safety-oriented design philosophy consistent with operation in human-centric environments.

Chapter 5 Sytem Modeling

5.1 Problem Description

The robotic platform considered in this research is a two-wheeled differential-drive robot whose main body chassis is located at the wheel axis. Attached to the chassis is a linear actuator capable of extending and retracting, terminating in a fork-like end effector. This fork is connected via a revolute joint, allowing its orientation to be adjusted as required. The robot is designed to use this tool to transport loads of varying weights from one location to another.

The sensing capabilities of the system are limited and include wheel encoders, a position sensor for the linear actuator, and an angular sensor measuring the orientation of the fork joint. Despite this seemingly simple setup, the task is inherently challenging due to the underactuated nature of the system and the presence of dynamically varying, structured conditions induced by different payloads and configurations.

Furthermore, the robot is intended to operate in environments where human interaction may occur at any time. As a result, the control system must be capable of adapting to sudden changes in the system's dynamic behavior, such as those caused by collisions or unexpected interactions, while maintaining stability and safe operation.

5.2 Dynamic Model Derivation

This section presents the derivation of the dynamic model of the Two-Wheeled Forklift Robot using an energy-based formulation. The model captures the coupled dynamics of the wheeled base, main body, vertically actuated lift mechanism, and fork assembly, and serves as the foundation for control design.

5.3 Coordinate Frames and Generalized Coordinates

5.3.1 Reference Frames

An inertial world frame $\{W\}$ is defined with the x-axis aligned with the forward horizontal direction and the z-axis pointing upward. A body-fixed frame $\{B\}$ is attached to the main

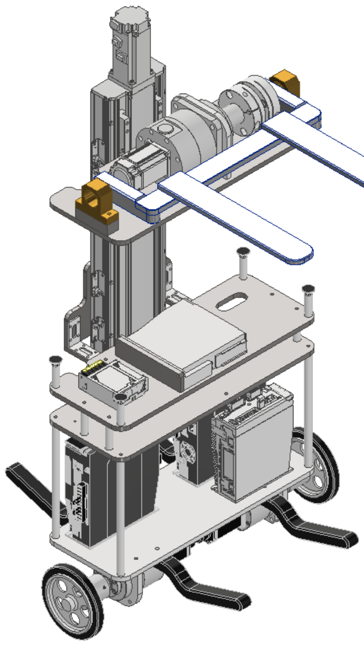


Figure 5.1: CAD model of the two-wheeled fork robot (TWFR).

body of the robot, with its origin located at the wheel–ground contact point and rotating with the pitch angle θ_p.

5.3.2 Generalized Coordinates

The system dynamics are described by the generalized coordinate vector

$$\mathbf{q} = \begin{bmatrix} x & \theta_p & d_m & \theta_a \end{bmatrix}^T, \tag{5.1}$$

where $x = r_w \theta_w$ denotes the longitudinal position of the robot expressed through the wheel rotation angle θ_w, θ_p is the pitch angle of the main body, d_m is the vertical displacement of the lift mechanism, and θ_a represents the relative rotation of the fork with respect to the main body.

5.3.3 Geometric Parameters

The following geometric parameters define the locations of the subsystem centers of gravity (CoG):

- r_w: wheel radius,
- (l_{bx}, l_{bz}): CoG of the main body expressed in frame $\{B\}$,
- (l_{mx}, l_{mz}): CoG of the lift assembly,
- $(l_{ax}, l_{az} + d_m)$: CoG of the fork motor arm,

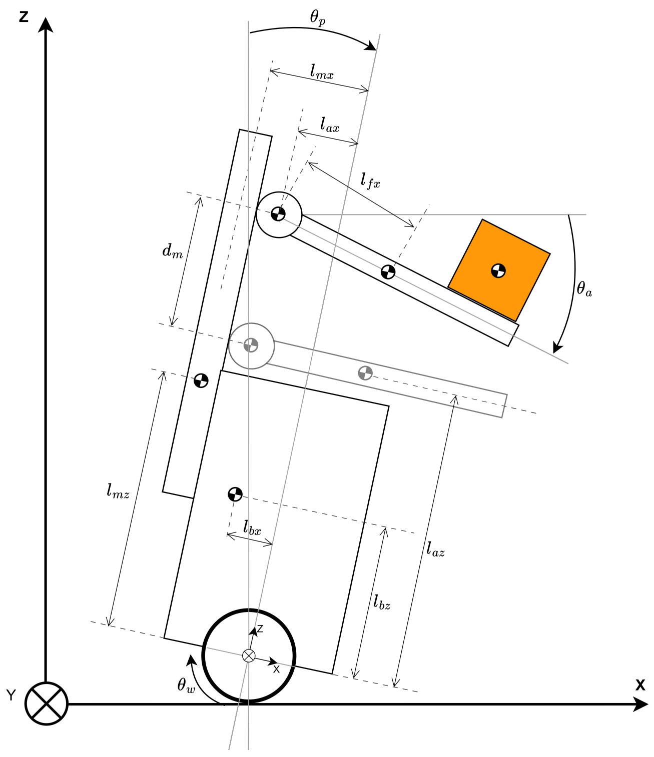


Figure 5.2: Parametric diagram of the two-wheeled fork robot (TWFR).

- $(l_{fx}, l_{fz} + d_m)$: CoG of the fork.

5.4 Energy-Based Modeling

The dynamic model is derived by computing the kinetic and potential energy contributions of each subsystem. The resulting expressions are grouped consistently to facilitate the Lagrangian formulation.

5.4.1 Wheel Subsystem

Position and Velocity

The wheel center translates horizontally according to

$$\dot{x}_w = \dot{x}, \qquad \dot{z}_w = 0. \tag{5.2}$$

Energies

The kinetic energy of each wheel consists of translational and rotational components. Using the rolling constraint $\dot{\theta}_w = \dot{x}/r_w$, the kinetic energy is given by

$$T_w = \frac{1}{2} m_w \dot{x}^2 + \frac{1}{2} I_w \left(\frac{\dot{x}}{r_w} \right)^2 . \tag{5.3}$$

The potential energy is defined with respect to the wheel axis and is therefore zero:

$$U_w = 0. \tag{5.4}$$

5.4.2 Main Body Subsystem

Position and Velocity

The position of the main body CoG expressed in the world frame is

$$x_b = x + l_{bx} \cos\theta_p + l_{bz} \sin\theta_p, \tag{5.5}$$
$$z_b = r_w - l_{bx} \sin\theta_p + l_{bz} \cos\theta_p. \tag{5.6}$$

Differentiation yields the linear velocity components

$$\dot{x}_b = \dot{x} - l_{bx}\dot{\theta}_p \sin\theta_p + l_{bz}\dot{\theta}_p \cos\theta_p, \tag{5.7}$$
$$\dot{z}_b = -l_{bx}\dot{\theta}_p \cos\theta_p - l_{bz}\dot{\theta}_p \sin\theta_p. \tag{5.8}$$

Energies

The kinetic and gravitational potential energies of the main body are given by

$$T_b = \frac{1}{2} m_b (\dot{x}_b^2 + \dot{z}_b^2) + \frac{1}{2} I_b \dot{\theta}_p^2, \tag{5.9}$$

$$U_b = m_b g(-l_{bx} \sin\theta_p + l_{bz} \cos\theta_p). \tag{5.10}$$

5.4.3 Lift Subsystem

Position and Velocity

The lift assembly CoG position is expressed as

$$x_m = x + l_{mx} \cos\theta_p + l_{mz} \sin\theta_p, \tag{5.11}$$
$$z_m = r_w - l_{mx} \sin\theta_p + l_{mz} \cos\theta_p, \tag{5.12}$$

with corresponding velocities

$$\dot{x}_m = \dot{x} - l_{mx}\dot{\theta}_p \sin\theta_p + l_{mz}\dot{\theta}_p \cos\theta_p, \tag{5.13}$$

$$\dot{z}_m = -l_{mx}\dot{\theta}_p \cos\theta_p - l_{mz}\dot{\theta}_p \sin\theta_p. \tag{5.14}$$

Energies

The kinetic and potential energies of the lift assembly are

$$T_m = \frac{1}{2}m_m(\dot{x}_m^2 + \dot{z}_m^2) + \frac{1}{2}I_m\dot{\theta}_p^2, \tag{5.15}$$

$$U_m = m_m g(-l_{mx}\sin\theta_p + l_{mz}\cos\theta_p). \tag{5.16}$$

5.4.4 Fork Motor Arm Subsystem

Position and Velocity

The CoG position of the fork motor arm is

$$x_a = x + l_{ax}\cos\theta_p + (l_{az} + d_m)\sin\theta_p, \tag{5.17}$$

$$z_a = r_w - l_{ax}\sin\theta_p + (l_{az} + d_m)\cos\theta_p, \tag{5.18}$$

with linear velocity components

$$\dot{x}_a = \dot{x} - l_{ax}\dot{\theta}_p \sin\theta_p + (l_{az} + d_m)\dot{\theta}_p \cos\theta_p + \dot{d}_m \sin\theta_p, \tag{5.19}$$

$$\dot{z}_a = -l_{ax}\dot{\theta}_p \cos\theta_p - (l_{az} + d_m)\dot{\theta}_p \sin\theta_p + \dot{d}_m \cos\theta_p. \tag{5.20}$$

Energies

The corresponding energy expressions are

$$T_a = \frac{1}{2}m_a(\dot{x}_a^2 + \dot{z}_a^2) + \frac{1}{2}I_a\dot{\theta}_p^2, \tag{5.21}$$

$$U_a = m_a g(-l_{ax}\sin\theta_p + (l_{az} + d_m)\cos\theta_p). \tag{5.22}$$

5.4.5 Fork Tool Subsystem

Position and Velocity

The fork CoG position is given by

$$x_f = x + l_{ax}\cos\theta_p + (l_{az} + d_m)\sin\theta_p + l_{fx}\cos(\theta_p + \theta_a), \tag{5.23}$$

$$z_f = r_w - l_{ax}\sin\theta_p + (l_{az} + d_m)\cos\theta_p - l_{fx}\sin(\theta_p + \theta_a). \tag{5.24}$$

The corresponding velocities are

$$\dot{x}_f = \dot{x} - l_{ax}\dot{\theta}_p\sin\theta_p + (l_{az} + d_m)\dot{\theta}_p\cos\theta_p + \dot{d}_m\sin\theta_p \tag{5.25}$$

$$- l_{fx}\sin(\theta_p + \theta_a)(\dot{\theta}_p + \dot{\theta}_a), \tag{5.26}$$

$$\dot{z}_f = -l_{ax}\dot{\theta}_p\cos\theta_p - (l_{az} + d_m)\dot{\theta}_p\sin\theta_p + \dot{d}_m\cos\theta_p \tag{5.27}$$

$$- l_{fx}\cos(\theta_p + \theta_a)(\dot{\theta}_p + \dot{\theta}_a). \tag{5.28}$$

Energies

The kinetic and potential energies of the fork are

$$T_f = \frac{1}{2}m_f(\dot{x}_f^2 + \dot{z}_f^2) + \frac{1}{2}I_f(\dot{\theta}_p + \dot{\theta}_a)^2, \tag{5.29}$$

$$U_f = m_f g\big(- l_{ax}\sin\theta_p + (l_{az} + d_m)\cos\theta_p - l_{fx}\sin(\theta_p + \theta_a)\big). \tag{5.30}$$

5.5 Lagrangian Formulation

5.5.1 Total Energies

The total kinetic and potential energies of the system are obtained by summing the individual contributions:

$$T = 2T_w + T_b + T_m + T_a + T_f, \tag{5.31}$$

$$U = 2U_w + U_b + U_m + U_a + U_f. \tag{5.32}$$

5.5.2 Equations of Motion

The Lagrangian is defined as

$$L = T - U. \tag{5.33}$$

The equations of motion follow from the Euler–Lagrange equations

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i, \qquad q_i \in \mathbf{q}, \tag{5.34}$$

where Q_i denotes the generalized forces.

5.5.3 Compact Matrix Representation

The system dynamics can be expressed in compact form as

$$M(q)\,\ddot{q} + C(q,\dot{q})\,\dot{q} + G(q) = \tau, \tag{5.35}$$

where $q \in \mathbb{R}^4$ is the vector of generalized coordinates. The inertia matrix $M(q) \in \mathbb{R}^{4\times4}$, Coriolis and centrifugal matrix $C(q,\dot{q}) \in \mathbb{R}^{4\times4}$, and gravity vector $G(q) \in \mathbb{R}^4$ depend on the system configuration and velocities.

The explicit expressions of $M(q)$, $C(q,\dot{q})$, and $G(q)$ are obtained through symbolic computation and are omitted for brevity.

5.6 System Simplification

To enable real-time optimal control, the full multibody structure of the robot is reduced to an equivalent planar rigid body. All rigid components of the platform, including the lift mechanism and the carried payload, are lumped into a single body characterized by an equivalent mass, a global center of gravity (CoG), and an equivalent rotational inertia. This reduction preserves the dominant inertial effects relevant to balance and longitudinal motion while significantly reducing model complexity.

5.6.1 Equivalent Mass and Center of Gravity

Let the system consist of $N = 4$ rigid components with masses m_i and centers of gravity located at (x_i, z_i) in the body-fixed frame. The payload is treated as a parameterized component with mass $\hat{m}$ and horizontal offset $\hat{d}$, while vertical positions depend on the lift displacement p_{lift}.

The total equivalent mass is defined as

$$M = \sum_{i=1}^{4} m_i. \tag{5.36}$$

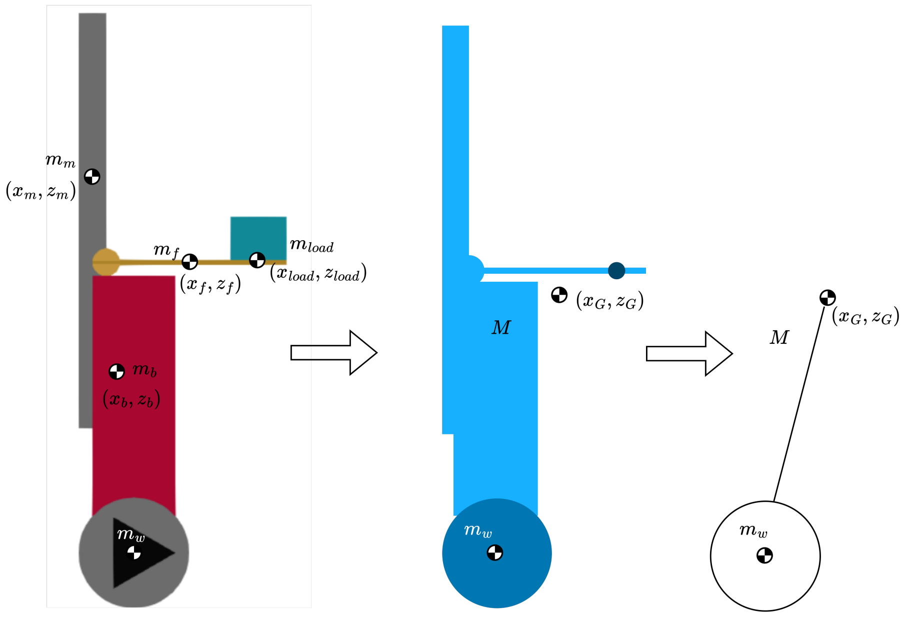


Figure 5.3: Simplified model of the two-wheeled fork robot (TWFR).

The coordinates of the equivalent center of gravity (x_G, z_G) are computed as

$$x_G = \frac{1}{M}\sum_{i=1}^{4} m_i x_i, \qquad z_G = \frac{1}{M}\sum_{i=1}^{4} m_i z_i. \tag{5.37}$$

5.6.2 Equivalent Rotational Inertia

The equivalent rotational inertia about the out-of-plane y axis and the global CoG is obtained using the parallel-axis theorem. Defining

$$\Delta x_i = x_i - x_G, \qquad \Delta z_i = z_i - z_G, \tag{5.38}$$

the equivalent inertia is given by

$$I_{\text{eq}} = \sum_{i=1}^{4}\left(I_{y,i} + m_i\left(\Delta x_i^2 + \Delta z_i^2\right)\right). \tag{5.39}$$

The parameters $(M, x_G, z_G, I_{\text{eq}})$ vary continuously as functions of payload mass, payload position, and lift displacement, enabling online adaptation within the control framework.

5.6.3 Simplified Dynamic Model

Using the equivalent parameters defined above, the robot is modeled as a planar rigid body mounted on two identical wheels. The dynamics are described using the generalized coordinate vector

$$q_s = \begin{bmatrix} q_x & \theta_p \end{bmatrix}^T, \tag{5.40}$$

where q_x denotes the horizontal position of the wheel axis and θ_p is the pitch angle of the equivalent body.

Kinematics

Let (l_{bx}, l_{bz}) denote the position of the equivalent CoG relative to the wheel axis in the body-fixed frame. The CoG position in the inertial frame is

$$x_b = q_x + l_{bx}\cos\theta_p + l_{bz}\sin\theta_p, \tag{5.41}$$

$$z_b = r_w - l_{bx}\sin\theta_p + l_{bz}\cos\theta_p. \tag{5.42}$$

The corresponding velocities are

$$\dot{x}_b = \dot{q}_x - l_{bx}\dot{\theta}_p\sin\theta_p + l_{bz}\dot{\theta}_p\cos\theta_p, \tag{5.43}$$

$$\dot{z}_b = -l_{bx}\dot{\theta}_p\cos\theta_p - l_{bz}\dot{\theta}_p\sin\theta_p. \tag{5.44}$$

Energy Expressions

Assuming pure rolling without slip, the kinetic energy of one wheel is

$$T_w = \frac{1}{2}m_w\dot{q}_x^2 + \frac{1}{2}I_w\left(\frac{\dot{q}_x}{r_w}\right)^2, \tag{5.45}$$

with zero potential energy.

The kinetic energy of the equivalent body is

$$T_b = \frac{1}{2}M\left(\dot{x}_b^2 + \dot{z}_b^2\right) + \frac{1}{2}I_{\text{eq}}\dot{\theta}_p^2, \tag{5.46}$$

and the gravitational potential energy is

$$U_b = Mg\left(-l_{bx}\sin\theta_p + l_{bz}\cos\theta_p\right). \tag{5.47}$$

The total energies are therefore

$$T = 2T_w + T_b, \qquad U = U_b. \tag{5.48}$$

Equations of Motion

The Lagrangian is defined as

$$L = T - U. \tag{5.49}$$

Applying the Euler–Lagrange equations yields

$$M_s(q_s)\ddot{q}_s + h_s(q_s, \dot{q}_s) = \boldsymbol{\tau}_s, \tag{5.50}$$

where $M_s(q_s) \in \mathbb{R}^{2\times 2}$ is the inertia matrix and $h_s(q_s, \dot{q}_s)$ collects Coriolis, centrifugal, and gravitational terms.

Generalized Forces

The system is actuated through torques applied at the wheels. Let τ_L and τ_R denote the left and right wheel torques. The resulting generalized force vector is

$$\boldsymbol{\tau}_s = \begin{bmatrix} (\tau_L + \tau_R)/r_w \\ 0 \end{bmatrix}. \tag{5.51}$$

State-Space Representation

For control design, the dynamics are expressed in first-order form using the state vector

$$x_s = \begin{bmatrix} q_s^T & \dot{q}_s^T \end{bmatrix}^T = \begin{bmatrix} q_x & \theta_p & \dot{q}_x & \dot{\theta}_p \end{bmatrix}^T. \tag{5.52}$$

The continuous-time state equations are

$$\dot{x}_s = \begin{bmatrix} \dot{q}_s \\ \ddot{q}_s \end{bmatrix}, \qquad \ddot{q}_s = M_s(q_s)^{-1}\left(\boldsymbol{\tau}_s - h_s(q_s, \dot{q}_s)\right), \tag{5.53}$$

which corresponds to the model implemented symbolically and used within the NMPC framework.

5.7 Payload Mass and Center-of-Mass Estimation

This section presents a quasi-static estimation method for the payload mass and its horizontal distance from the arm joint axis. The payload is modeled as a point mass, and the estimation relies on reaction force and torque measurements obtained from the lift linear actuator and the arm motor.

5.7.1 Assumptions

The following assumptions are made:

- The payload is rigidly attached to the fork and can be approximated as a point mass.
- The system operates under quasi-static conditions.

- Reaction estimates are available from the lift actuator and the arm motor.

5.7.2 Available Measurements

The estimator uses the following measured or estimated quantities:

- $\hat{f}_m^{\text{reac}}$: estimated reaction force at the lift actuator
- $\hat{\tau}_a^{\text{reac}}$: estimated reaction torque at the arm motor
- θ_p^{res}: measured pitch angle
- d_m^{res}: measured lift position
- θ_a^{res}: measured arm motor angle

The parameters to be estimated are:

- $\hat{m}$: payload mass
- $\hat{d}$: horizontal distance of the payload center of mass from the arm joint axis

5.7.3 Estimation Equations

When the lift mechanism is active, the payload mass can be directly estimated from the lift reaction force as

$$\hat{m}_{\text{lift}} = \frac{\hat{f}_m^{\text{reac}}}{g\cos\left(\theta_p^{\text{res}}\right)}, \tag{5.54}$$

where g denotes the gravitational acceleration.

The payload mass can alternatively be estimated from the arm reaction torque according to

$$\hat{m}_{\text{arm}} = \frac{\hat{\tau}_a^{\text{reac}}}{g\cos\left(\theta_a^{\text{res}} + \theta_p^{\text{res}}\right)\,\hat{d}}. \tag{5.55}$$

The distance of the payload center of mass from the arm joint axis is estimated as

$$\hat{d} = \begin{cases} \dfrac{\hat{\tau}_a^{\text{reac}}}{g\cos\left(\theta_a^{\text{res}} + \theta_p^{\text{res}}\right)\,\hat{m}}, & \text{if } d_m^{\text{res}} > 0, \\ d_{\text{fork}}^{\text{nom}}, & \text{otherwise,} \end{cases} \tag{5.56}$$

where $d_{\text{fork}}^{\text{nom}}$ denotes the nominal center-of-mass distance of the empty fork.

5.7.4 Estimation Logic

The final payload mass estimate is selected based on the lift position:

$$\hat{m} = \begin{cases} \hat{m}_{\text{lift}}, & \text{if } d_m^{\text{res}} > 0, \\ \hat{m}_{\text{arm}}, & \text{otherwise.} \end{cases} \tag{5.57}$$

5.7.5 Implementation Remark

The estimation logic is implemented using Simulink blocks and logical switching. By combining force-based and torque-based measurements, the estimator remains operational even when one of the reaction signals is unavailable, ensuring robustness across different operating conditions.

Chapter 6 Control

This chapter presents the control architecture adopted for the TWFR, which combines a centralized nonlinear model predictive controller with disturbance observer-based techniques. The overall strategy is designed to address the strongly nonlinear and underactuated nature of the system while ensuring robust and safe operation during human–robot interaction. A single nonlinear model predictive control (NMPC) scheme is employed to simultaneously regulate the longitudinal motion of the robot along the x-axis and stabilize its pitch angle, which is essential for maintaining dynamic balance. By explicitly incorporating the nonlinear system dynamics and physical constraints, NMPC provides anticipative control actions and naturally acts as a prediction-based safety layer, enabling early detection of abnormal behaviors and appropriate reactions to external interactions. To further enhance robustness and situational awareness, disturbance and reaction disturbance observers (DOB/RDOB) are integrated as virtual sensing mechanisms. These observers provide real-time estimates of external disturbances and payload-induced uncertainties without relying on additional physical sensors, thereby complementing the NMPC framework and improving control performance under varying operating conditions.

6.1 Introduction to Nonlinear Model Predictive Control

Nonlinear Model Predictive Control (NMPC) is an advanced control strategy that has attracted considerable interest in embedded optimization and robotic applications. Unlike linear MPC, NMPC explicitly accounts for nonlinear system dynamics, constraints, and objective functions, making it particularly well suited for mechanically complex and underactuated systems. This capability is essential for self-balancing robotic platforms, where strong nonlinearities, coupling effects, and state-dependent constraints fundamentally influence the system behavior.

6.1.1 NMPC Problem Formulation

The continuous-time nonlinear optimal control problem underlying NMPC can be formulated as

$$\begin{aligned}
\min_{x(\cdot),z(\cdot),u(\cdot)} \quad & \int_0^T \ell(x(t), z(t), u(t))\,\mathrm{d}t + M(x(T)) \\
\text{subject to} \quad & x(0) = \bar{x}_0, \\
& 0 = f(\dot{x}(t), x(t), z(t), u(t)), \quad t \in [0,T], \\
& 0 \geq g(x(t), z(t), u(t)), \quad t \in [0,T],
\end{aligned} \tag{6.1}$$

where $x : \mathbb{R} \longrightarrow \mathbb{R}^{n_x}$ denotes the differential state vector, $z : \mathbb{R} \longrightarrow \mathbb{R}^{n_z}$ represents the algebraic variables, and $u : \mathbb{R} \longrightarrow \mathbb{R}^{n_u}$ is the control input. The function $\ell(\cdot)$ defines the running cost, while $M(\cdot)$ denotes the terminal cost. The implicit differential-algebraic equations (DAEs) describing the system dynamics are captured by $f(\cdot)$, and $g(\cdot)$ represents nonlinear path constraints.

6.1.2 Multiple Shooting Discretization

To enable the numerical solution of the continuous-time optimal control problem, a multiple shooting discretization is employed. The prediction horizon $[0,T]$ is divided into N intervals defined by the grid points $t_0, t_1, \ldots, t_N$. This yields the following discrete-time formulation:

$$\begin{aligned}
\min_{x_0,\ldots,x_N, z_0,\ldots,z_{N-1}, u_0,\ldots,u_{N-1}} \quad & \sum_{k=0}^{N-1} (t_{k+1} - t_k)\, \ell(x_k, z_k, u_k) + M(x_N) \\
\text{subject to} \quad & x_0 = \bar{x}_0, \\
& \begin{bmatrix} x_{k+1} \\ z_k \end{bmatrix} = \varphi_k(x_k, u_k), \quad k = 0, \ldots, N-1, \\
& 0 \geq g_k(x_k, z_k, u_k), \quad k = 0, \ldots, N-1,
\end{aligned} \tag{6.2}$$

where $\varphi_k(\cdot)$ denotes the numerical integration operator that propagates the system dynamics over each discretization interval.

6.2 The *acados* Software Framework

6.2.1 Overview of *acados*

acados is a modular open-source framework for fast embedded optimal control, designed to achieve high computational performance while maintaining modeling flexibility. It provides efficient numerical solvers tailored to real-time NMPC applications and supports seamless integration with symbolic modeling tools. Key features of *acados* include efficient C-based implementations of optimal control algorithms, high-performance linear algebra routines based on the BLASFEO library, interfaces to MATLAB and Python, compatibility with the CasADi modeling framework, and deployability on embedded hardware platforms.

6.2.2 Sequential Quadratic Programming in *acados*

Nonlinear optimal control problems in *acados* are solved using Sequential Quadratic Programming (SQP). At each SQP iteration, the original nonlinear problem is locally approximated by a quadratic programming (QP) subproblem obtained through first-order linearization of the system dynamics and constraints, together with a second-order approximation of the cost function.

Standard QP Subproblem

At SQP iteration k, the resulting QP is formulated in terms of state and input increments $\Delta x_k = x_k - \bar{x}_k$ and $\Delta u_k = u_k - \bar{u}_k$, and reads

$$\begin{aligned} \min_{\Delta x_k, \Delta u_k} \quad & \frac{1}{2} \begin{bmatrix} \Delta x_k \\ \Delta u_k \end{bmatrix}^T H_k \begin{bmatrix} \Delta x_k \\ \Delta u_k \end{bmatrix} + g_k^T \begin{bmatrix} \Delta x_k \\ \Delta u_k \end{bmatrix} \\ \text{subject to} \quad & \Delta x_{k+1} = A_k \Delta x_k + B_k \Delta u_k + d_k, \\ & \underline{c}_k \leq C_k \begin{bmatrix} \Delta x_k \\ \Delta u_k \end{bmatrix} \leq \bar{c}_k, \end{aligned} \tag{6.3}$$

where the optimization variables represent corrections to the current trajectory estimate rather than absolute state and input values.

Interpretation of the SQP Decision Variables and QP Components

At each SQP iteration, the quadratic program is solved in terms of incremental decision variables Δx_k and Δu_k, which represent correction steps with respect to the current trajectory estimate. The updated solution is obtained as

$$x_k^{\text{new}} = x_k^{\text{current}} + \Delta x_k, \qquad u_k^{\text{new}} = u_k^{\text{current}} + \Delta u_k. \tag{6.4}$$

This incremental formulation allows the solver to iteratively refine the solution by computing how the current estimate should be adjusted to reduce the cost while satisfying the constraints.

Cost Function Approximation The matrix H_k denotes the Hessian approximation of the Lagrangian and captures the local curvature of the cost function around the current iterate. In *acados*, a Gauss–Newton approximation is commonly employed,

$$H_k \approx J_k^T W J_k, \tag{6.5}$$

where J_k is the Jacobian of the least-squares residual and W is a user-defined weighting matrix. This approximation avoids the computation of second-order derivatives, significantly reducing computational complexity while maintaining favorable numerical properties for tracking and regulation problems.

The vector g_k represents the gradient of the cost function evaluated at the current trajectory estimate and indicates the direction of steepest descent in the local quadratic model.

Linearized Dynamics and Constraints The matrices A_k and B_k result from first-order linearization of the nonlinear system dynamics

$$\dot{x} = f(x, u), \tag{6.6}$$

and are defined as

$$A_k = \frac{\partial f}{\partial x}, \qquad B_k = \frac{\partial f}{\partial u}. \tag{6.7}$$

They describe the sensitivity of the future system evolution with respect to variations in the current state and control input, respectively. Through this linearization, the nonlinear dynamics are locally approximated by an affine model suitable for efficient QP-based optimization.

The term d_k denotes the dynamic residual introduced by linearization and represents the mismatch between the nonlinear dynamics evaluated at the current iterate and their linear approximation. The QP solution seeks to reduce this residual, thereby enforcing consistency with the underlying physical model across successive SQP iterations.

Multi-Stage SQP Formulation

In the multi-stage setting adopted by *acados*, the QP over the prediction horizon is written as

$$\begin{aligned}
\min_{\substack{\Delta x_0,\ldots,\Delta x_N\\ \Delta u_0,\ldots,\Delta u_{N-1}\\ s_0,\ldots,s_N}} \quad & \sum_{k=0}^{N-1} \left(\frac{1}{2} \begin{bmatrix} \Delta x_k \\ \Delta u_k \end{bmatrix}^T H_k \begin{bmatrix} \Delta x_k \\ \Delta u_k \end{bmatrix} + \begin{bmatrix} q_k \\ r_k \end{bmatrix}^T \begin{bmatrix} \Delta x_k \\ \Delta u_k \end{bmatrix} \right) \\
& + \frac{1}{2} \Delta x_N^T Q_N \Delta x_N + q_N^T \Delta x_N + \sum_{k=0}^{N} \left(s_k^T P_k s_k + p_k^T s_k \right) \\
\text{subject to} \quad & \Delta x_{k+1} = A_k \Delta x_k + B_k \Delta u_k + d_k, \\
& G_k^x \Delta x_k + G_k^u \Delta u_k - J_{s,k} s_k \le \bar{g}_k, \\
& s_k \ge 0,
\end{aligned} \tag{6.8}$$

where slack variables s_k are introduced to soften constraints and improve numerical robustness.

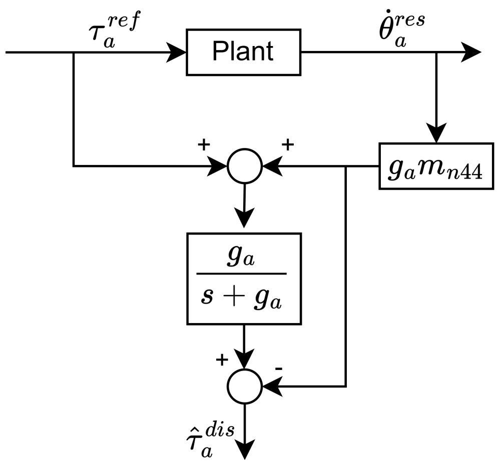


Figure 6.1: Block diagram of the fork disturbance observer (FDOB) implementation.

6.2.3 Real-Time Iteration Scheme

For real-time control applications, *acados* employs the Real-Time Iteration (RTI) scheme. Instead of iterating the SQP algorithm until convergence at each sampling instant, RTI performs a single SQP iteration per control cycle. This iteration is divided into a preparation phase, which computes the linearization of the dynamics and constraints, and a feedback phase, which solves the resulting QP using the updated initial state. This strategy enables a favorable trade-off between computational efficiency and control performance, making it suitable for embedded implementations with strict real-time constraints.

6.3 Disturbance Observers and Reaction Force Estimation

In addition to the NMPC framework, disturbance observer–based techniques are employed to enhance robustness and to estimate interaction forces without relying on additional physical sensors. Disturbance observers (DOBs) are used to compensate for unmodeled dynamics, coupling effects, and external perturbations, while reaction disturbance observers enable the estimation of load-dependent forces and torques arising from physical interaction with the environment.

6.3.1 Fork Disturbance Observer (FDOB)

To compensate for unmodeled dynamics and external effects acting on the fork joint, a disturbance observer is introduced. The observer estimates the total disturbance torque affecting the fork angle, including dynamic coupling with the remaining subsystems, parametric uncertainties, and external torques.

The rotational dynamics associated with the fork angle are expressed as

$$m_{41}\ddot{\theta}_w + m_{42}\ddot{\theta}_p + m_{43}\ddot{d}_m + m_{44}\ddot{\theta}_a + h_4 + g_4 = n_a\tau_a - T_{la}. \tag{6.9}$$

By introducing a nominal inertia parameter m_{n44}, the fork dynamics are rewritten in nominal form as

$$m_{n44}\ddot{\theta}_a^{res} = n_a\tau_a^{ref} - \tilde{\tau}_a^{dist}, \tag{6.10}$$

where $\tilde{\tau}_a^{dist}$ denotes the lumped disturbance torque. This term can be expanded as

$$\tilde{\tau}_a^{dist} = m_{41}\ddot{\theta}_w^{res} + m_{42}\ddot{\theta}_p^{res} + m_{43}\ddot{d}_m^{res} + (m_{44} - m_{n44})\ddot{\theta}_a^{res} + h_4 + g_4 + T_{la}. \tag{6.11}$$

The disturbance torque is estimated using a pseudo-differentiation method applied to the measured fork angular velocity. The resulting FDOB estimate is given by

$$\hat{\tau}_a^{dist} = \frac{g_a}{s + g_a}\left(n_a\tau_a^{ref} + g_a m_{n44}\dot{\theta}_a^{res}\right) - g_a m_{n44}\dot{\theta}_a^{res}. \tag{6.12}$$

The block diagram of the FDOB implementation within the control architecture is shown in Fig. 6.3.1.

6.3.2 Fork Angle Control

The fork angle is regulated using a PD controller augmented with the disturbance estimate provided by the FDOB. The resulting control law is defined as

$$\tau_a^{ref} = K_{pa}(\theta_a^{cmd} - \theta_a^{res}) + K_{da}(\dot{\theta}_a^{cmd} - \dot{\theta}_a^{res}) + \hat{\tau}_a^{dist}. \tag{6.13}$$

The complete FDOB–PD control architecture is illustrated in Fig. 6.4.1.

6.3.3 Fork Reaction Torque Observer (FRTOB)

Beyond disturbance compensation, a reaction torque observer is formulated to estimate the torque generated by an external load acting on the fork. The load-dependent torque is modeled as

$$T_{la} = \tilde{\tau}_a^{ext} + \tilde{\tau}_a^{fric}. \tag{6.14}$$

By isolating gravity and friction effects from the total disturbance, the reaction torque is expressed as

$$\tilde{\tau}_a^{reac} = m_{41}\ddot{\theta}_w^{res} + m_{42}\ddot{\theta}_p^{res} + m_{43}\ddot{d}_m^{res} + (m_{44} - m_{n44})\ddot{\theta}_a^{res} + h_4 + \tilde{\tau}_a^{ext}. \tag{6.15}$$

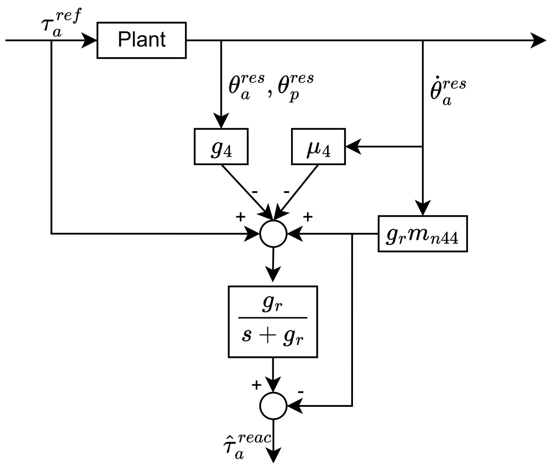


Figure 6.2: Block diagram of the fork reaction torque observer (FRTOB) implementation.

The nominal fork dynamics can then be written as

$$m_{n44}\ddot{\theta}_a^{res} = n_a \tau_a^{ref} - g_4 - \tilde{\tau}_a^{fric} - \tilde{\tau}_a^{reac}. \tag{6.16}$$

The reaction torque is estimated using

$$\hat{\tau}_a^{reac} = \frac{g_r}{s+g_r}\left(n_a \tau_a^{ref} + g_r m_{n44}\dot{\theta}_a^{res} - g_4 - \tilde{\tau}_a^{fric}\right) - g_r m_{n44}\dot{\theta}_a^{res}. \tag{6.17}$$

The FRTOB implementation is depicted in Fig. 6.5.1. The estimated reaction torque is related to the load-dependent kinematic term t_l as

$$t_l = -\frac{\hat{\tau}_a^{reac}}{g m_l}. \tag{6.18}$$

6.3.4 Lift Disturbance Observer (LDOB)

An analogous disturbance observer is designed for the lift subsystem to estimate the total disturbance force acting on the vertical actuator. The lift dynamics are described by

$$m_{31}\ddot{\theta}_w + m_{32}\ddot{\theta}_p + m_{33}\ddot{d}_m + m_{34}\ddot{\theta}_a + h_3 + g_3 = f_m - F_{lm}. \tag{6.19}$$

Introducing a nominal mass parameter m_{n33}, the lift dynamics are rewritten as

$$m_{n33}\ddot{d}_m^{res} = f_m^{ref} - \tilde{f}_m^{dist}, \tag{6.20}$$

where the disturbance force is defined as

$$\tilde{f}_m^{dist} = m_{31}\ddot{\theta}_w^{res} + m_{32}\ddot{\theta}_p^{res} + (m_{33} - m_{n33})\ddot{d}_m^{res} + m_{34}\ddot{\theta}_a^{res} + h_3 + g_3 + F_{lm}. \tag{6.21}$$

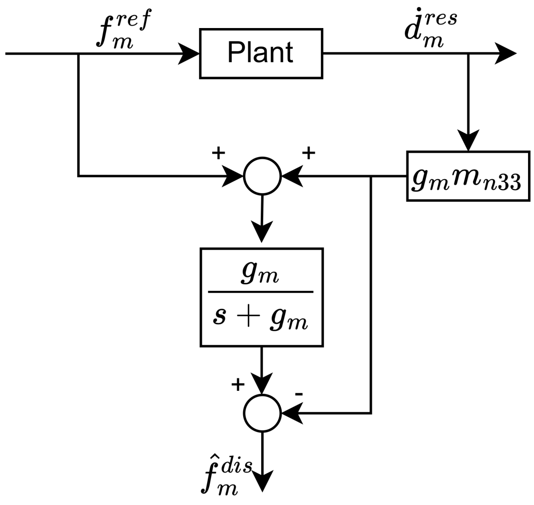


Figure 6.3: Block diagram of the lift disturbance observer (LDOB) implementation.

The disturbance force estimate is obtained using a pseudo-differentiation method,

$$\hat{f}_m^{dist} = \frac{g_m}{s + g_m}\left(f_m^{ref} + g_m m_{n33}\dot{d}_m^{res}\right) - g_m m_{n33}\dot{d}_m^{res}. \tag{6.22}$$

The LDOB implementation is shown in Fig. 6.6.1.

6.3.5 Lift Displacement Control

The lift displacement is regulated using a PD controller augmented with the disturbance force estimate,

$$f_m^{ref} = K_{pm}(d_m^{cmd} - d_m^{res}) + K_{dm}(\dot{d}_m^{cmd} - \dot{d}_m^{res}) + \hat{f}_m^{dist}. \tag{6.23}$$

The complete LDOB–PD control architecture is illustrated in Fig. 6.7.1.

6.3.6 Lift Reaction Force Observer (LRFOB)

The external load force acting on the lift mechanism is modeled as

$$F_{lm} = \tilde{f}_m^{ext} + \tilde{f}_m^{fric}. \tag{6.24}$$

By excluding gravity and friction contributions from the disturbance formulation, the reaction force estimate is obtained as

$$\hat{f}_m^{reac} = \frac{g_r}{s + g_r}\left(f_m^{ref} + g_r m_{n33}\dot{d}_m^{res} - g_3 - \tilde{f}_m^{fric}\right) - g_r m_{n33}\dot{d}_m^{res}. \tag{6.25}$$

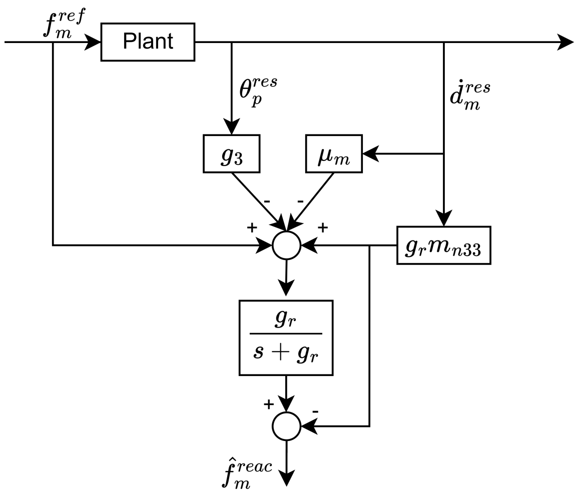


Figure 6.4: Block diagram of the lift reaction force observer (LRFOB) implementation.

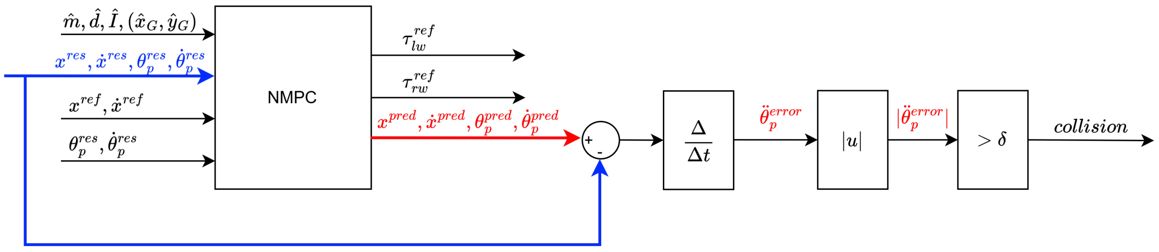

$\hat{m}, \hat{d}, \hat{I}, (\hat{x}_G, \hat{y}_G)$
$x^{res}, \dot{x}^{res}, \theta_p^{res}, \dot{\theta}_p^{res}$
$x^{ref}, \dot{x}^{ref}$
$\theta_p^{res}, \dot{\theta}_p^{res}$
NMPC
τ_{lw}^{ref}
τ_{rw}^{ref}
$x^{pred}, \dot{x}^{pred}, \theta_p^{pred}, \dot{\theta}_p^{pred}$
$\frac{\Delta}{\Delta t}$
$\ddot{\theta}_p^{error}$
$|u|$
$|\ddot{\theta}_p^{error}|$
$> \delta$
collision

Figure 6.5: Control architecture with collision handling

The LRFOB block diagram is shown in Fig. 6.8.1. The estimated reaction force is related to the load mass parameter used in the kinematic model as

$$m_l = \frac{\hat{f}_m^{reac}}{g\cos(\theta_p)}. \tag{6.26}$$

6.4 Collision Detection Mechanism

This section describes the collision detection mechanism implemented within the proposed control architecture. Unlike force-based approaches relying directly on external sensors or disturbance estimates, the adopted method exploits the prediction capabilities of NMPC to identify abnormal system behavior caused by external contacts.

6.5 Collision

Figure 6.5 illustrates the signal flow used for collision detection and handling.

6.5.1 Prediction Error-Based Collision Detection

At each control cycle, the NMPC provides a one-step-ahead prediction of the system state based on the previously applied control input and the internal model. Let x_k denote the measured state at the current sampling instant, and $\hat{x}_{k|k-1}$ the state predicted at the previous iteration for the same time instant.

The prediction error is computed as

$$e_k = x_k - \hat{x}_{k|k-1}. \tag{6.27}$$

To emphasize sudden deviations caused by impulsive external interactions, the absolute value of the prediction error is considered and numerically differentiated:

$$\dot{e}_k = \frac{\mathrm{d}}{\mathrm{d}t} \left(|e_k| \right). \tag{6.28}$$

This operation highlights rapid changes in the system response that cannot be explained by the nominal model dynamics or by smooth load variations.

6.5.2 State Selection and Thresholding

Among the available state components, the derivative of the pitch-rate prediction error, $\dot{e}_{\dot{\beta},k}$, is selected as the collision indicator. This choice is motivated by the strong sensitivity of the pitch dynamics to external contacts in self-balancing systems.

A collision is detected when the following condition is satisfied:

$$|\dot{e}_{\dot{\beta},k}| > \gamma_{\text{coll}}, \tag{6.29}$$

where γ_{coll} is a predefined threshold determined empirically. This threshold allows reliable discrimination between nominal disturbances and abrupt collision events.

6.5.3 Integration with the Control Architecture

Once the collision condition is met, a collision flag is raised and forwarded to the supervisory finite state machine (FSM). The FSM then triggers the appropriate control reconfiguration, as shown in Figure 6.5, enabling a safe reaction while maintaining continuity of operation.

Since the robot is not autonomous, this mechanism does not aim to cancel user commands but rather to detect unsafe interactions and adapt the control behavior accordingly. By leveraging NMPC predictions, the proposed approach achieves fast and reliable collision detection without requiring additional physical sensors.

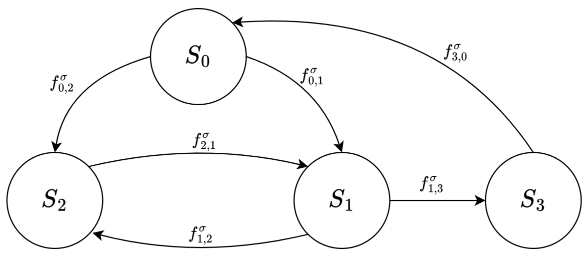


Figure 6.6: Block diagram of the finite state machine (FSM) implementation.

6.6 Finite State Machine

Distinguishing between disturbances caused by charging or discharging operations and those arising from physical contact is non-trivial, as both scenarios produce similar sensor signatures. To address this challenge, a Finite State Machine (FSM) is proposed to robustly discriminate between these situations and to enforce appropriate system behavior.

6.6.1 FSM Inputs

The FSM operates based on the following boolean input signals:

- **Start**: Enables the FSM operation.
- **Move**: Commands the robot to move according to the provided reference.
- **Load**: Indicates that a loading or unloading operation is in progress.
- **Collision**: Signals the detection of an external disturbance or collision.

6.6.2 FSM States

The FSM is composed of four discrete states, each associated with a specific robot behavior:

- S_0 **– Idle:** The robot remains stationary with no active motion or task execution.
- S_1 **– Moving:** The robot moves while tracking the given motion references.
- S_2 **– Loading:** The robot remains stationary and deliberately ignores disturbance signals to allow safe loading or unloading.
- S_3 **– Collision:** Upon detecting a collision, the robot performs a short backward motion and subsequently stops.

6.6.3 State Transitions

The state transitions are governed by the following transition functions:

- $f^{\sigma}_{0,1}$: cmd(Move) = 1
- $f^{\sigma}_{0,2}$: cmd(Load) = 1
- $f^{\sigma}_{1,2}$: cmd(Load) = 1
- $f^{\sigma}_{1,3}$: cmd(Collision) = 1
- $f^{\sigma}_{2,1}$: cmd(Move) = 1
- $f^{\sigma}_{3,0}$: cmd(Start) = 1

The resulting FSM diagram is illustrated in Figure.

State	Behavior	Key Feature
S_0 – Idle	Stationary, no motion execution	System entry/exit point
S_1 – Moving	Tracks motion references while monitoring disturbances	Active navigation mode
S_2 – Loading	Stationary with **disturbance signals ignored**	Safe human-robot interaction during load/unload
S_3 – Collision	Short backward motion → full stop	Automatic safety recovery

Table 6.1: FSM operational states

Transition	Condition	Description
$f^{\sigma}_{0,1}$	cmd(Move)	Start navigation from idle
$f^{\sigma}_{0,2}$	cmd(Load)	Initiate loading procedure
$f^{\sigma}_{1,2}$	cmd(Load)	Pause motion for loading
$f^{\sigma}_{1,3}$	cmd(Collision)	Emergency response during motion
$f^{\sigma}_{2,1}$	cmd(Move)	Resume navigation after loading
$f^{\sigma}_{3,0}$	cmd(Start)	System reset after collision

Table 6.2: FSM state transition logic

6.7 Global Control Strategy and Architecture

This section summarizes the overall control strategy adopted in this work and clarifies the interaction between the different control and estimation modules introduced previously. The proposed architecture is designed for a non-autonomous self-balancing robot, where high-level decisions are driven by a human operator, while low-level control and safety are handled automatically.

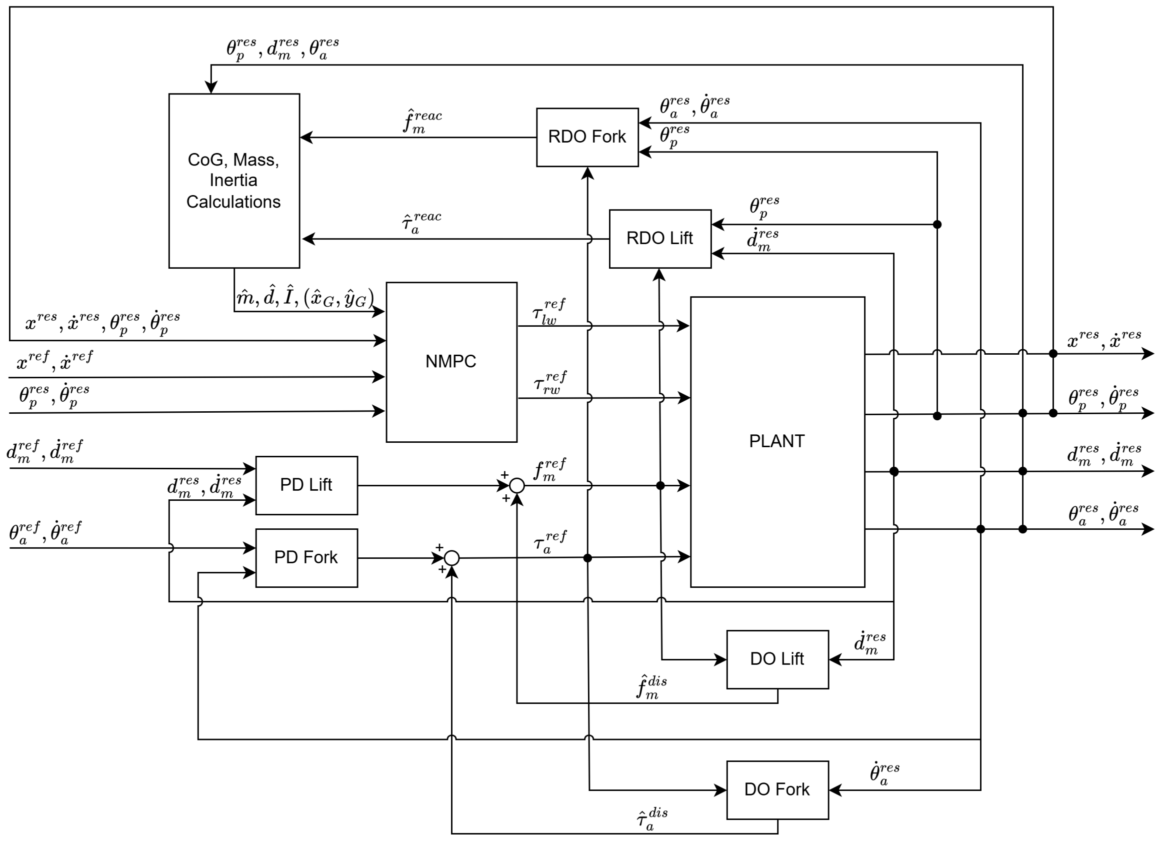
At the core of the architecture lies a Nonlinear Model Predictive Controller (NMPC), which is responsible for stabilizing the robot and regulating its motion. The NMPC simultaneously enforces dynamic balance, longitudinal motion control, and actuator constraints by explicitly accounting for the nonlinear system dynamics. Rather than generating autonomous behaviors, the NMPC tracks references that are derived from human commands or higher-level supervisory logic, ensuring that operator intentions are executed in a dynamically feasible and safe manner.

To enhance robustness against modeling uncertainties and external interactions, Disturbance Observers (DOB) and Reaction Disturbance Observers (RDOB) are integrated into the control loop. These observers act as virtual sensors, providing real-time estimates of external forces, payload variations, and interaction-induced disturbances. Their outputs are not used to override the operator commands, but instead to improve the reliability of the NMPC predictions and to enable informed safety decisions when abnormal conditions are detected.

A Finite State Machine (FSM) supervises the overall system behavior and governs the transitions between different operational modes. The FSM monitors key variables such as estimated disturbances, reaction forces, and system states, and determines whether the robot operates in nominal control, interaction-aware, or safety-related modes. In this way, the FSM provides a structured and transparent mechanism for handling events such as human contact, payload changes, or constraint violations, without compromising the continuous control performance of the NMPC.

Figure 6.7 illustrates the global control architecture. Human inputs are processed at the supervisory level and translated into references for the NMPC. The NMPC computes optimal control actions based on the current state estimates and predicted system evolution. In parallel, the DOB and RDOB estimate external disturbances and feed this information to both the NMPC and the FSM. The FSM, acting as a high-level safety and logic layer, can adapt control parameters or switch operational modes in response to detected events, ensuring safe and predictable human–robot interaction.

Overall, this layered control strategy combines prediction-based optimal control, disturbance-aware estimation, and discrete supervisory logic. This integration allows the robot to remain stable and responsive under human operation, while providing a safety-oriented framework capable of handling external interactions and uncertainties in a principled manner.



$\theta_p^{res}, d_m^{res}, \theta_a^{res}$
CoG, Mass, Inertia Calculations
$\hat{f}_m^{reac}$
RDO Fork
$\theta_a^{res}, \dot{\theta}_a^{res}$
θ_p^{res}
$\hat{\tau}_a^{reac}$
RDO Lift
θ_p^{res}
$\dot{d}_m^{res}$
$\hat{m}, \hat{d}, \hat{I}, (\hat{x}_G, \hat{y}_G)$
$x^{res}, \dot{x}^{res}, \theta_p^{res}, \dot{\theta}_p^{res}$
NMPC
τ_{lw}^{ref}
$x^{ref}, \dot{x}^{ref}$
$\theta_p^{res}, \dot{\theta}_p^{res}$
τ_{rw}^{ref}
PLANT
$x^{res}, \dot{x}^{res}$
$\theta_p^{res}, \dot{\theta}_p^{res}$
$d_m^{ref}, \dot{d}_m^{ref}$
$d_m^{res}, \dot{d}_m^{res}$
PD Lift
f_m^{ref}
$d_m^{res}, \dot{d}_m^{res}$
$\theta_a^{ref}, \dot{\theta}_a^{ref}$
PD Fork
τ_a^{ref}
$\theta_a^{res}, \dot{\theta}_a^{res}$
DO Lift
$\dot{d}_m^{res}$
$\hat{f}_m^{dis}$
DO Fork
$\dot{\theta}_a^{res}$
$\hat{\tau}_a^{dis}$

Figure 6.7: Block diagram of the global control architecture.

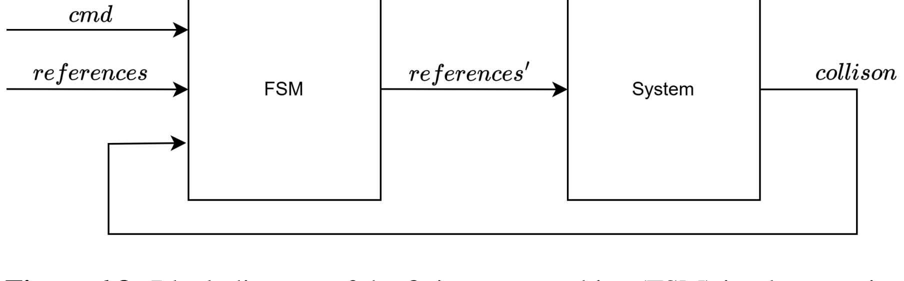


Figure 6.8: Block diagram of the finite state machine (FSM) implementation.

Chapter 7 Simulation and Results

7.1 NMPC Solver Configuration and Weight Selection

The NMPC controller is implemented using the *acados* optimal control framework. The prediction horizon is defined with $N = 40$ discretization steps over a horizon length of $T = 1$ s, resulting in a sampling time of 25 ms. This configuration represents a compromise between prediction accuracy and computational complexity, allowing real-time execution.

The system state is defined as

$$x = \begin{bmatrix} x & \beta & \dot{x} & \dot{\beta} \end{bmatrix}^T,$$

while the control input consists of the left and right wheel torques.

Cost Function A nonlinear least-squares formulation (NONLINEAR_LS) is employed for the initial and path cost stages. This formulation provides flexibility in weighting states and inputs while remaining compatible with a Gauss–Newton Hessian approximation.

The state weighting matrix is defined as

$$W_x = \operatorname{diag}\left(10^{-9},\ 10^{1},\ 10^{2},\ 10^{3}\right),$$

placing dominant emphasis on the pitch angle and pitch rate to prioritize balance stabilization. The longitudinal position is assigned a negligible weight to avoid unnecessary regulation of absolute position during balance recovery. The control input weighting matrix is selected as

$$W_u = 10^{-1} I_2,$$

penalizing excessive control effort while preserving sufficient authority for fast corrective actions. The resulting stage cost matrix is given by $W = \operatorname{blkdiag}(W_x, W_u)$, with a zero reference corresponding to regulation around the upright equilibrium.

Constraints Box constraints are imposed on both control inputs and states to reflect physical limitations and ensure safe operation. Wheel torques are bounded within ±10 Nm. State constraints limit the longitudinal position, pitch angle, and their derivatives, with the pitch angle restricted to $\pm 45^\circ$ to prevent loss of balance.

The initial state is enforced as a hard constraint, ensuring consistency between the measured system state and the NMPC prediction at each control cycle.

Solver Settings The nonlinear program is solved using a Sequential Quadratic Programming Real-Time Iteration (SQP–RTI) scheme, enabling a single SQP iteration per control step and ensuring real-time feasibility. A Gauss–Newton approximation is used for the Hessian of the cost function, which significantly reduces computational complexity for least-squares objectives.

System dynamics are integrated using the GNSF integrator, and the resulting quadratic programs are solved using the HPIPM solver with partial condensing. Warm-starting is enabled to accelerate convergence across successive control cycles. Overall, the selected solver configuration allows real-time NMPC execution while maintaining robust balance control and effective constraint handling.

7.2 Simulation Results

This section presents simulation results for a representative and challenging scenario involving a sudden payload variation, commanded motion, and an external collision. The objective of this simulation is to evaluate the closed-loop behavior of the proposed control and estimation architecture under combined disturbances, while assessing stability, estimation performance, and computational feasibility.

The simulation begins with the robot operating under nominal conditions. At a given time instant, a payload is suddenly applied to the system, inducing a change in the overall mass and center-of-mass location. Simultaneously, longitudinal motion is commanded by the operator, and an external collision is introduced as an impulsive disturbance. This scenario is designed to stress both the prediction-based controller and the disturbance-aware estimation layers.

Figure 7.1 shows the longitudinal velocity profile during the disturbance event. Despite the abrupt changes in system dynamics and the external interaction, the NMPC maintains bounded velocity behavior and smoothly rejects the disturbance. This demonstrates the controller's ability to preserve dynamic stability while tracking operator-driven motion commands.

The estimated payload mass is shown in Figure 7.2. Following the sudden load application, the estimator rapidly converges to the new payload mass value. The transient response remains well damped, indicating that the reaction-based estimation strategy is effective under quasi-static conditions even in the presence of motion and disturbances.

Figure 7.3 illustrates the estimated horizontal distance of the payload center of mass from the arm joint axis. The estimate adapts consistently with the applied load, converging toward

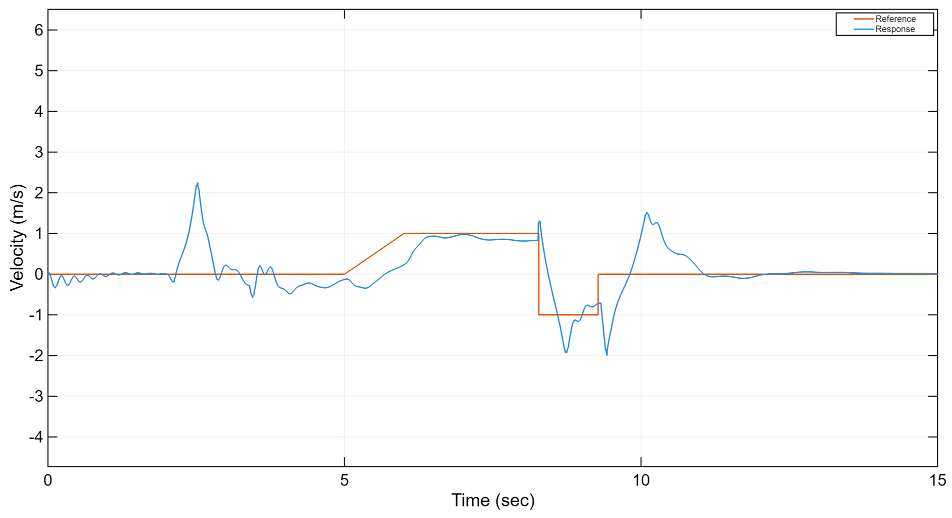


Figure 7.1: Simulation results showing the velocity profile during a disturbance event

a stable value after the disturbance. This result confirms that the proposed estimation logic can reliably infer geometric payload properties using reaction torque information.

The absolute value of the derivative of the prediction error is reported in Figure 7.4. Peaks correspond to the instants of payload application and collision, reflecting abrupt deviations between predicted and measured behavior. These events are correctly detected by the observer framework, providing a meaningful signal for interaction awareness and safety supervision through the FSM.

Finally, Figure 7.5 shows the computational time per control iteration. The CPU time remains within bounded limits throughout the simulation, including during disturbance events. This confirms that the NMPC configuration and solver settings are compatible with real-time execution requirements.

Overall, the simulation results demonstrate that the proposed control architecture successfully combines nonlinear predictive control, disturbance-aware estimation, and supervisory logic. The system remains stable, responsive, and computationally efficient under simultaneous payload changes, motion commands, and external collisions, supporting its suitability for safe human-operated scenarios.

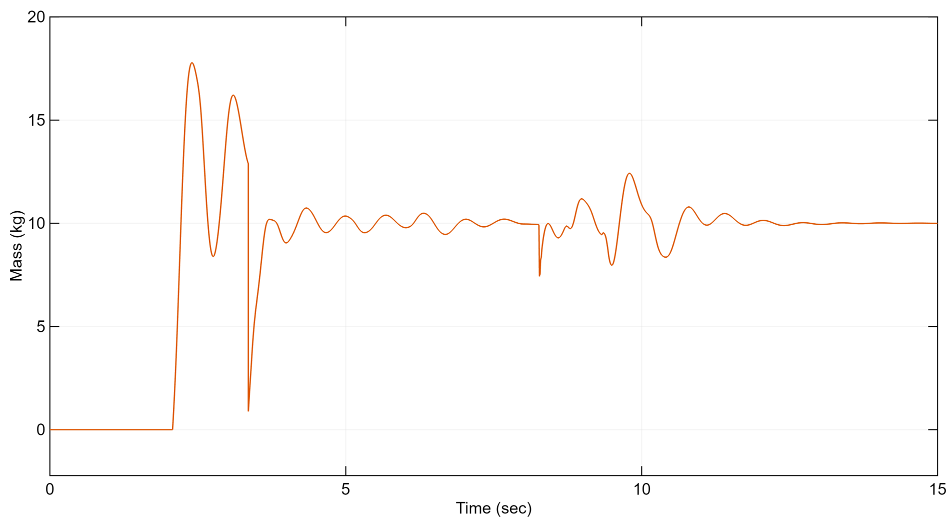


Figure 7.2: Simulation results showing mass estimation

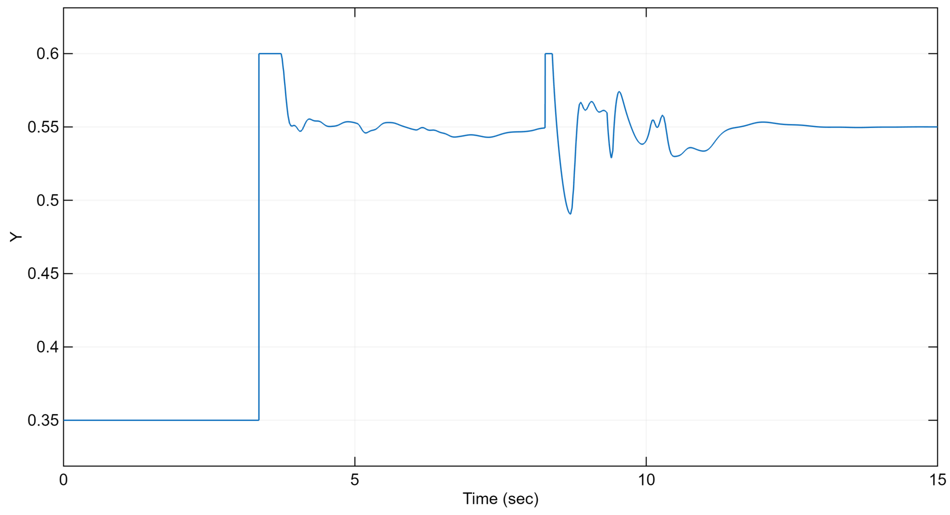


Figure 7.3: Simulation results showing distance of load from arm axis estimation

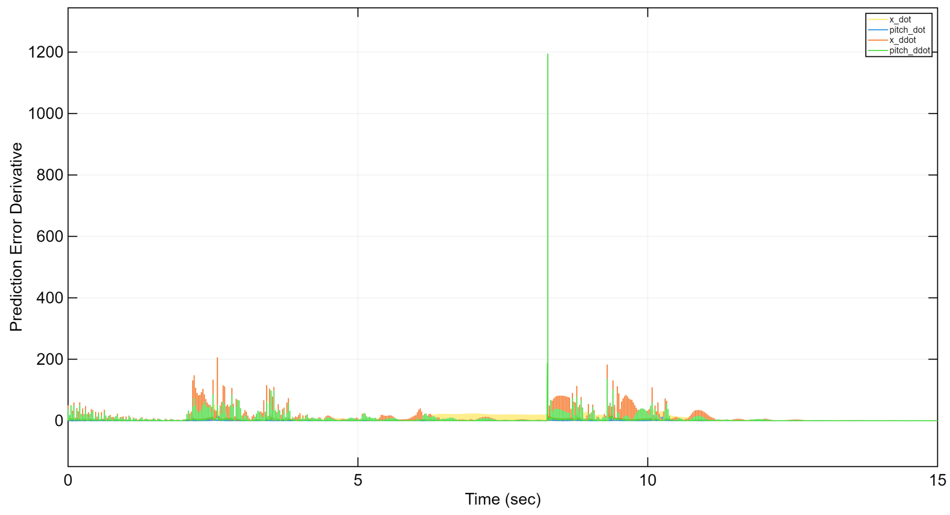


Figure 7.4: Simulation results showing the absolute value of the derivative of the prediction error

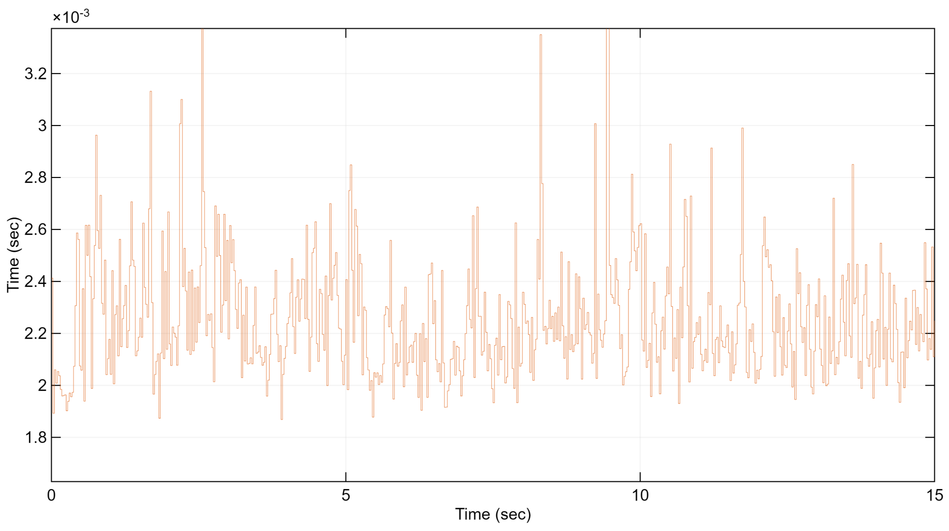


Figure 7.5: Simulation results showing CPU time

Chapter 8 Discussion

Chapter 9 Conclusions

Chapter 10 Equations

10.1 Visualize equations

10.2 Summary of Model Equations

$$\mathbf{q} = \begin{bmatrix} x & \theta_p & d_m & \theta_a \end{bmatrix}^T \tag{10.1}$$

$$x = r_w \theta_w, \qquad \dot{\theta}_w = \frac{\dot{x}}{r_w} \tag{10.2}$$

$$\dot{x}_w = \dot{x}, \qquad \dot{z}_w = 0 \tag{10.3}$$

$$T_w = \frac{1}{2} m_w \dot{x}^2 + \frac{1}{2} I_w \left(\frac{\dot{x}}{r_w} \right)^2, \qquad U_w = 0 \tag{10.4}$$

$$x_b = x + l_{bx} \cos\theta_p + l_{bz} \sin\theta_p \tag{10.5}$$

$$z_b = r_w - l_{bx} \sin\theta_p + l_{bz} \cos\theta_p \tag{10.6}$$

$$\dot{x}_b = \dot{x} - l_{bx} \dot{\theta}_p \sin\theta_p + l_{bz} \dot{\theta}_p \cos\theta_p \tag{10.7}$$

$$\dot{z}_b = -l_{bx} \dot{\theta}_p \cos\theta_p - l_{bz} \dot{\theta}_p \sin\theta_p \tag{10.8}$$

$$T_b = \frac{1}{2} m_b (\dot{x}_b^2 + \dot{z}_b^2) + \frac{1}{2} I_b \dot{\theta}_p^2 \tag{10.9}$$

$$U_b = m_b g (-l_{bx} \sin\theta_p + l_{bz} \cos\theta_p) \tag{10.10}$$

$$x_m = x + l_{mx}\cos\theta_p + l_{mz}\sin\theta_p \tag{10.11}$$
$$z_m = r_w - l_{mx}\sin\theta_p + l_{mz}\cos\theta_p \tag{10.12}$$

$$\dot{x}_m = \dot{x} - l_{mx}\dot{\theta}_p\sin\theta_p + l_{mz}\dot{\theta}_p\cos\theta_p \tag{10.13}$$
$$\dot{z}_m = -l_{mx}\dot{\theta}_p\cos\theta_p - l_{mz}\dot{\theta}_p\sin\theta_p \tag{10.14}$$

$$T_m = \frac{1}{2}m_m(\dot{x}_m^2 + \dot{z}_m^2) + \frac{1}{2}I_m\dot{\theta}_p^2 \tag{10.15}$$

$$U_m = m_m g(-l_{mx}\sin\theta_p + l_{mz}\cos\theta_p) \tag{10.16}$$

$$x_a = x + l_{ax}\cos\theta_p + (l_{az} + d_m)\sin\theta_p \tag{10.17}$$
$$z_a = r_w - l_{ax}\sin\theta_p + (l_{az} + d_m)\cos\theta_p \tag{10.18}$$

$$\dot{x}_a = \dot{x} - l_{ax}\dot{\theta}_p\sin\theta_p + (l_{az} + d_m)\dot{\theta}_p\cos\theta_p + \dot{d}_m\sin\theta_p \tag{10.19}$$
$$\dot{z}_a = -l_{ax}\dot{\theta}_p\cos\theta_p - (l_{az} + d_m)\dot{\theta}_p\sin\theta_p + \dot{d}_m\cos\theta_p \tag{10.20}$$

$$T_a = \frac{1}{2}m_a(\dot{x}_a^2 + \dot{z}_a^2) + \frac{1}{2}I_a\dot{\theta}_p^2 \tag{10.21}$$

$$U_a = m_a g(-l_{ax}\sin\theta_p + (l_{az} + d_m)\cos\theta_p) \tag{10.22}$$

$$x_f = x + l_{ax}\cos\theta_p + (l_{az} + d_m)\sin\theta_p + l_{fx}\cos(\theta_p + \theta_a) \tag{10.23}$$
$$z_f = r_w - l_{ax}\sin\theta_p + (l_{az} + d_m)\cos\theta_p - l_{fx}\sin(\theta_p + \theta_a) \tag{10.24}$$

$$\dot{x}_f = \dot{x} - l_{ax}\dot{\theta}_p \sin\theta_p + (l_{az} + d_m)\dot{\theta}_p \cos\theta_p + \dot{d}_m \sin\theta_p \tag{10.25}$$

$$- l_{fx} \sin(\theta_p + \theta_a)(\dot{\theta}_p + \dot{\theta}_a) \tag{10.26}$$

$$\dot{z}_f = -l_{ax}\dot{\theta}_p \cos\theta_p - (l_{az} + d_m)\dot{\theta}_p \sin\theta_p + \dot{d}_m \cos\theta_p \tag{10.27}$$

$$- l_{fx} \cos(\theta_p + \theta_a)(\dot{\theta}_p + \dot{\theta}_a) \tag{10.28}$$

$$T_f = \frac{1}{2}m_f(\dot{x}_f^2 + \dot{z}_f^2) + \frac{1}{2}I_f(\dot{\theta}_p + \dot{\theta}_a)^2 \tag{10.29}$$

$$U_f = m_f g\big(- l_{ax} \sin\theta_p + (l_{az} + d_m) \cos\theta_p - l_{fx} \sin(\theta_p + \theta_a)\big) \tag{10.30}$$

$$T = 2T_w + T_b + T_m + T_a + T_f \tag{10.31}$$

$$U = 2U_w + U_b + U_m + U_a + U_f \tag{10.32}$$

$$L = T - U \tag{10.33}$$

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i \tag{10.34}$$

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau \tag{10.35}$$

10.3 Simplified

$$M = \sum_{i=1}^{4} m_i \tag{10.36}$$

$$x_G = \frac{1}{M}\sum_{i=1}^{4} m_i x_i, \qquad z_G = \frac{1}{M}\sum_{i=1}^{4} m_i z_i \tag{10.37}$$

$$\Delta x_i = x_i - x_G, \qquad \Delta z_i = z_i - z_G \tag{10.38}$$

$$I_{\text{eq}} = \sum_{i=1}^{4} \left(I_{y,i} + m_i(\Delta x_i^2 + \Delta z_i^2)\right) \tag{10.39}$$

$$q_s = \begin{bmatrix} x & \theta_p \end{bmatrix}^T \tag{10.40}$$

$$x_b = x + l_{bx} \cos\theta_p + l_{bz} \sin\theta_p \tag{10.41}$$

$$z_b = r_w - l_{bx} \sin\theta_p + l_{bz} \cos\theta_p \tag{10.42}$$

$$\dot{x}_b = \dot{x} - l_{bx}\dot{\theta}_p \sin\theta_p + l_{bz}\dot{\theta}_p \cos\theta_p \tag{10.43}$$

$$\dot{z}_b = -l_{bx}\dot{\theta}_p \cos\theta_p - l_{bz}\dot{\theta}_p \sin\theta_p \tag{10.44}$$

$$T_w = \frac{1}{2} m_w \dot{x}^2 + \frac{1}{2} I_w \left(\frac{\dot{x}}{r_w}\right)^2 \tag{10.45}$$

$$T_b = \frac{1}{2} M \left(\dot{x}_b^2 + \dot{z}_b^2\right) + \frac{1}{2} I_{\text{eq}} \dot{\theta}_p^2 \tag{10.46}$$

$$U_b = Mg\left(-l_{bx} \sin\theta_p + l_{bz} \cos\theta_p\right) \tag{10.47}$$

$$T = 2T_w + T_b \tag{10.48}$$

$$U = U_b \tag{10.49}$$

$$L = T - U \tag{10.50}$$

$$M_s(q_s)\ddot{q}_s + h_s(q_s, \dot{q}_s) = \boldsymbol{\tau}_s \tag{10.51}$$

$$\boldsymbol{\tau}_s = \begin{bmatrix} (\tau_L + \tau_R)/r_w \\ 0 \end{bmatrix} \tag{10.52}$$

$$x_s = \begin{bmatrix} x & \theta_p & \dot{x} & \dot{\theta}_p \end{bmatrix}^T \tag{10.53}$$

$$\dot{x}_s = \begin{bmatrix} \dot{x} \\ \dot{\theta}_p \\ \ddot{x} \\ \ddot{\theta}_p \end{bmatrix} \tag{10.54}$$

$$\ddot{q}_s = M_s(q_s)^{-1}\left(\boldsymbol{\tau}_s - h_s(q_s, \dot{q}_s)\right) \tag{10.55}$$

- N: The prediction horizon length
- T: The total prediction horizon time
- $x_k \in \mathbb{R}^{n_x}$: The differential state vector at time step k
- $u_k \in \mathbb{R}^{n_u}$: The control input vector
- z_k: The algebraic variables
- $\ell(x_k, z_k, u_k)$: The stage cost (or running cost)
- $M(x_N)$: The terminal cost
- $x_0 = \bar{x}_0$: The initial state constraint
- $\varphi_k(x_k, u_k)$: The numerical integrator (discretized dynamics)
- $g_k(\cdot) \leq 0$: The inequality constraints

$\Delta x_k, \Delta u_k$: corrections ($x^+ = x + \Delta x,\ u^+ = u + \Delta u$)
$H_k \approx J_k^\top W J_k$: Gauss–Newton Hessian
g_k: cost gradient
A_k, B_k: Jacobians $\partial f/\partial x, \partial f/\partial u$
d_k: dynamics residual
$C_k, \underline{c}_k, \bar{c}_k$: linearized constraints
s_k: slack variables

$$\hat{m}_{\text{lift}} = \frac{\hat{f}_m^{\text{reac}}}{g\cos\left(\theta_p^{\text{res}}\right)} \tag{10.56}$$

$$\hat{m}_{\text{arm}} = \frac{\hat{\tau}_a^{\text{reac}}}{g\cos\left(\theta_a^{\text{res}} + \theta_p^{\text{res}}\right)\ \hat{d}} \tag{10.57}$$

$$\hat{d} = \begin{cases} \dfrac{\hat{\tau}_a^{\text{reac}}}{g\cos\left(\theta_a^{\text{res}} + \theta_p^{\text{res}}\right)\ \hat{m}}, & \text{if } d_m^{\text{res}} > 0, \\ d_{\text{fork}}^{\text{nom}}, & \text{otherwise} \end{cases} \tag{10.58}$$

$$\hat{m} = \begin{cases} \hat{m}_{\text{lift}}, & \text{if } d_m^{\text{res}} > 0, \\ \hat{m}_{\text{arm}}, & \text{otherwise} \end{cases} \tag{10.59}$$

References

[1] Adam Mills, Adrian Wills, and Brett Ninness. “Nonlinear Model Predictive Control of an Inverted Pendulum”. In: *2009 American Control Conference*. 2009 American Control Conference. St. Louis, MO, USA: IEEE, 2009, pp. 2335–2340. DOI: 10.1109/ACC.2009.5160391.

[2] Ashutosh Mishra and Kritika Bansal. “Control of Two-Wheel Self-Balancing Robot: LQR and MPC Performance Analysis”. In: *2024 IEEE International Students’ Conference on Electrical, Electronics and Computer Science (SCEECS)*. 2024 IEEE International Students’ Conference on Electrical, Electronics and Computer Science (SCEECS). Bhopal, India: IEEE, Feb. 24, 2024, pp. 1–6. ISBN: 979-8-3503-4846-0. DOI: 10.1109/SCEECS61402.2024.10481856.

[3] Haris Sheh Zad et al. “Optimal Controller Design for Self-Balancing Two-Wheeled Robot System”. In: *2016 International Conference on Frontiers of Information Technology (FIT)*. 2016 International Conference on Frontiers of Information Technology (FIT). Islamabad, Pakistan: IEEE, Dec. 2016, pp. 11–16. DOI: 10.1109/FIT.2016.011.

[4] David Kim and Dongil Choi. “Control of a Longitudinally Extended Two-Wheeled Inverted Pendulum Robot with a Sliding Mechanism”. In: *Journal of Computational Design and Engineering* 12.10 (Oct. 11, 2025), pp. 118–132. DOI: 10.1093/jcde/qwaf098.

[5] Mert Önkol and Coşku Kasnakoğlu. “Adaptive Model Predictive Control of a Two-wheeled Robot Manipulator with Varying Mass”. In: *Measurement and Control* 51.1–2 (Mar. 2018), pp. 38–56. DOI: 10.1177/0020294018758527.

[6] Jin Ito and Toshiyuki Murakami. “Underactuated Control for Two-Wheeled Mobile Robot with an Arm Using Torque Constraint Conditions and Disturbance Observer”. In: *2023 IEEE 32nd International Symposium on Industrial Electronics (ISIE)*. 2023 IEEE 32nd International Symposium on Industrial Electronics (ISIE). Helsinki, Finland: IEEE, June 19, 2023, pp. 1–6. ISBN: 979-8-3503-9971-4. DOI: 10.1109/ISIE51358.2023.10228153.

[7] Hikaru Yajima et al. “Posture Stabilization Control Compensating Variation of Body Center of Gravity in Underactuated System”. In: *2023 IEEE International Conference on Mechatronics (ICM)*. 2023 IEEE International Conference on Mechatronics (ICM). Loughborough, United Kingdom: IEEE, Mar. 15, 2023, pp. 1–6. ISBN: 978-1-6654-6661-5. DOI: 10.1109/ICM54990.2023.10101996.

[8] Hirotaka Kanazawa et al. “Model-Based Pitch Angle Compensation for Center of Gravity Variation in Underactuated System with an Arm”. In: *2023 IEEE 32nd International Symposium on Industrial Electronics (ISIE)*. 2023 IEEE 32nd International Symposium on Industrial Electronics (ISIE). Helsinki, Finland: IEEE, June 19, 2023, pp. 1–6. ISBN: 979-8-3503-9971-4. DOI: 10.1109/ISIE51358.2023.10228009.

[9] Van-Truong Nguyen et al. “Adaptive Nonlinear PD Controller of Two-Wheeled Self-Balancing Robot with External Force”. In: *Computers, Materials & Continua* 81.2 (2024), pp. 2337–2356. DOI: 10.32604/cmc.2024.055412.

[10] Vo Ba Viet Nghia et al. “Adaptive Neural Sliding Mode Control for Two Wheel Self Balancing Robot”. In: *International Journal of Dynamics and Control* 10.3 (June 2022), pp. 771–784. DOI: 10.1007/s40435-021-00832-1.

[11] Feni Isdaryani and Rudi Salam. “Design and Implementation of Two-Wheeled Robot Control Using MRAC”. In: ().

[12] Ali Unluturk and Omer Aydogdu. “Machine Learning Based Self-Balancing and Motion Control of the Underactuated Mobile Inverted Pendulum With Variable Load”. In: *IEEE Access* 10 (2022), pp. 104706–104718. DOI: 10.1109/ACCESS.2022.3210540.

[13] Jae Kook Ahn and Seul Jung. “Development of a Two-Wheel Mobile Manipulator: Balancing and Interaction Control”. In: *Robotica* 32.7 (Oct. 2014), pp. 1135–1152. DOI: 10.1017/S026357471300129X.

[14] Julian M. Salt Ducaju, Björn Olofsson, and Rolf Johansson. “Model-Based Predictive Impedance Variation for Obstacle Avoidance in Safe Human–Robot Collaboration”. In: *IEEE Transactions on Automation Science and Engineering* 22 (2025), pp. 9571–9583. DOI: 10.1109/TASE.2024.3508718.

[15] Zhipeng Yu and Lei Wang. “Impedance Control for Industrial Robots Based on Model Predictive Control”. In: *2025 44th Chinese Control Conference (CCC)*. 2025 44th Chinese Control Conference (CCC). Chongqing, China: IEEE, July 28, 2025, pp. 280–285. ISBN: 978-988-75816-1-1. DOI: 10.23919/CCC64809.2025.11179306.

[16] Tianyu Wang et al. *Head Stabilization for Wheeled Bipedal Robots via Force-Estimation-Based Admittance Control*. Nov. 24, 2025. DOI: 10.48550/arXiv.2511.18712. arXiv: 2511.18712 [cs]. Pre-published.

[17] Naoji Shiroma et al. “Cooperative Behavior of a Wheeled Inverted Pendulum for Object Transportation”. In: *Proceedings of the 1996 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS ’96)*. IEEE, 1996, pp. 396–401.