Balance Stability Augmentation for Wheel-legged Biped Robot through Adaptive Control Strategies

Doctoral Thesis

Balance Stability Augmentation for Wheel-legged Biped Robot through Adaptive Control Strategies

Department of Robotics
Graduate School of Engineering,
TOHOKU UNIVERSITY

FAHAD RAZA

Advising Professor at Tohoku University	Prof. Mitsuhiro Hayashibe
Research Advisor at Tohoku University	–
Dissertation Committee Members Name marked with "○" is the Chair Examiner	○ Prof. Mitsuhiro Hayashibe 1 Prof. Kazuya Yoshida 2 Prof. Yasuhisa Hirata 3 Dr. Ahmed Chemori (LIRMM, University of Montpellier, CNRS)

TOHOKU UNIVERSITY

Graduate School of Engineering

Balance Stability Augmentation for Wheel-legged Biped Robot through Adaptive Control Strategies

(適応制御戦略に基づく車輪型二足歩行ロボットのバランス安定性の向上)

A dissertation submitted for the degree of

Doctor of Philosophy (Engineering)

Department of Robotics

by

Fahad RAZA

January 11, 2022

In memory of my dear father.

Balance Stability Augmentation for Wheel-legged Biped Robot through Adaptive Control Strategies

Fahad RAZA

Abstract

The rapid demographic changes worldwide are expected to increase the overall aging population significantly in the coming decades. These alterations in population indicate that more older people would require government support in the form of medical care, nursing facilities, social services, and other relief systems. This can further lead to a disproportionate transfer of money from the working group of the taxpayers to the elderly. At present, a shortage in labor supply owing to an aging population in developed nations, such as Japan and Italy, is hampering industrial efficiency, productivity, and economic growth. These socio-economic problems can be avoided by maintaining the necessary size of the labor force through intelligent robots.

In recent years, several organizations have adopted robots to replace humans in dangerous jobs, automate industrial processes, and overcome labor shortages. Particularly, mobile robots are used for infrastructure health monitoring, disaster response, tour guiding purposes, planetary exploration, and numerous other situations. As modern societies are designed and built for humans, humanoid robots (robots with a human-like shape) are considered to better assist humans than other types of robots. For instance, humanoid robots can use the same tools that are designed for humans, climb stairs, and assist people in nursing facilities and hospitals similar to humans. However, humanoid robots are bulky and slow owing to their multiple joints, high energy consumption, and highly complex motion control system.

Recently, wheel-legged robots have been increasingly researched in the robotics community because of their humanoid-shape factor, agility, and simple motion control system in comparison with humanoid robots. Wheel-legged robots have the

advantages of humanoid robots and are faster, energy-efficient, and simpler in design than humanoid robots. However, they are vulnerable to falls owing to the self-balancing, underactuation, and nonlinear coupling of the system states. To address the challenges of instability and robustness, this study presents the development of several balance controllers for wheel-legged robots that are underactuated and non-holonomic in nature. The objective is to begin with widely used linear motion controllers, such as a linear quadratic regulator (LQR) based on the linearized model of the robot and gradually augment this baseline controller with a nonlinear model-based or robust adaptive controller. This study performed different tests in the Gazebo simulator and on an actual robot to demonstrate the validity and effectiveness of these controllers.

This dissertation initially focuses on the development of a mathematical model of a wheel-legged robot (named Igor) using the Euler–Lagrange formulation. For simplification, the robot is assumed to be a two-wheeled inverted pendulum, wherein the center of mass (CoM) height can be changed by simultaneously varying the knee and hip joint angles. Based on this dynamic model, I developed an LQR controller to balance, steer, and move the robot in the horizontal plane. Particularly, I performed motion stability analyses of the wheel-legged robot under different conditions, such as system modeling errors, sensor noise, and external disturbances.

Subsequently, I developed a model-based nonlinear computed torque controller for the wheel-legged robot. However, uncertainties in the modeling parameters, such as friction coefficients, inertia matrices, and position of CoM, generally cause deteriorating effects on the performance of model-based motion controllers. To alleviate such degradation in the robot's performance, this study proposes combining the feedback LQR with the feedforward computed torque controller. Contrary to the simulation results, the actual experiments on the robot indicated a steady-state error in the translational position tracking of the robot regardless of the motion controller, which included LQR, computed torque, and the combination of LQR and computed torque.

To further improve the robustness of the robot's motion controller with respect to un-

known external disturbances and parameter uncertainties, the LQR controller of the underactuated wheel-legged robot was augmented with a state-of-the-art $\mathcal{L}_1$ adaptive controller. Various simulations and experiments that include external disturbances, uncertainties in system parameters, and changes in ground friction demonstrate the superior performance of the proposed augmented LQR adaptive controller compared to the baseline LQR.

Most previous studies on wheel-legged robots focused on lower body stabilization. Therefore, motivated by the human ability to maintain balance in laborious activities by actively articulating the arm, this study explored and analyzed the active arm control along with the wheel-legged system to assist in balance recovery during external pushes and disturbances. The centroidal moment pivot (CMP) is used as a key metric to quantitatively evaluate the effect of the active arm usage on the balance stability improvement of the robot. This concept forms the basis for a wheel-legged biped robot with an active arm for dual purposes; one for carrying objects and the other for increasing balance stability.

Contents

List of Figures

List of Tables

List of Symbols

m_c	Mass of the Robot body
m_w	Mass of the robot wheel
l	Height of the CoM from the origin of Σ_R
I_{xx}	Moment of inertia of the robot body in X
I_{yy}	Moment of inertia of the robot body in Y
I_{zz}	Moment of inertia of the robot body in Z
I_{wa}	Moment of inertia of the robot wheel about its axis
I_{wd}	Moment of inertia of the robot wheel about its diameter
I_{ma}	Moment of Inertia of the motor rotor about its axis
I_{md}	Moment of Inertia of the motor rotor about its diameter
g	Gravitational acceleration constant
r_w	Wheel radius
b	Distance between the frame origin Σ_R and the center of the wheel
γ	Gear reduction ratio
c_r, c_l	Viscosity coefficients of left and right wheels
I_1	Moment of inertia of the arm link-1
I_2	Moment of inertia of the arm link-2
l_1	Length of the arm link-1
l_2	Length of the arm link-2
l_{g1}	Distance of CoM of the arm link-1 from its origin
l_{g2}	Distance of CoM of the arm link-2 from its origin
m_1	Mass of the arm link-1
m_2	Mass of the arm link-2
T_c	Kinetic energy of the robot CoM
T_w	Kinetic energy of the robot wheels
U_w	Potential energy of the robot wheels
U_c	Potential energy of the robot CoM
$\mathbb{R}$	The set of real numbers
$\mathbb{R}^n$	The set of n dimensional vectors whose entries are real numbers
$\mathbb{R}^{n\times n}$	The set of $n \times n$ dimensional matrices whose entries are real numbers

Chapter 1

Introduction

1.1 Background

With technological advances in medical care, improved social welfare services from governments, and declining birthrates, particularly in developed countries, the aging population is increasing worldwide. This demographic change can cause a slump in productivity and economic growth, thus impacting society. According to the prediction of the United Nations, the number of elderly people older than 60 years of age is currently more than 1 billion worldwide, which is expected to double by 2050 [6]. The situation is extremely severe in Japan, wherein the official data from the Ministry of Internal Affairs estimated that the number of people aged 65 or more reached a record of 29.1% of the country's total population in 2021 [7]. Consequently, the global shortage of productive age groups is considered a serious socio-economic challenge of the 21^{st} century.

A 2018 United Nations' report on the ratio of the aged population to the working population states that Japan and Italy have the highest percentage of 46.2% and 37.8%, respectively, among the G20 countries (Figure 1.1) [1]. Another negative impact of the aging population is that a considerable portion of the tax money obtained from a shrinking working generation is utilized for elderly care programs because of the universal social security policies. In Japan, the proportion of social security benefits for the el-

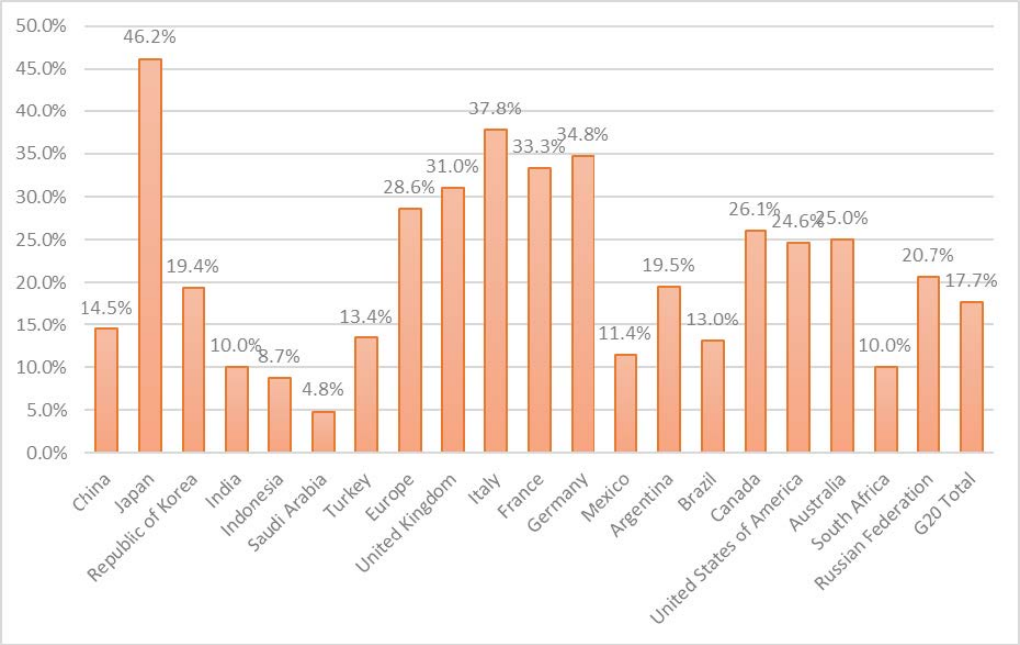
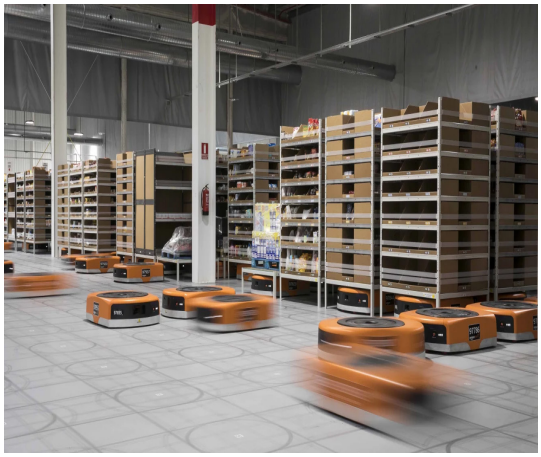


Figure 1.1 Ratio of aged population to working population in year 2015 [1].

derly is approximately two-thirds of the total social security benefits [8]. This trend of transferring a large sum of money from the working population to the elderly care may challenge the concept of joint social/inter-generational responsibility and undermine the socio-economic stability. Therefore, both developed and developing nations need to recognize and determine the consequences of the aging population. Furthermore, new policies should be devised to alleviate its harmful impacts on the economy and society.

Factory automation is one method that can overcome labor shortages; however, robots can be more useful in increasing work efficiency, helping humans in physically tough jobs, assisting the senior and disabled population, and interacting with people socially. In the last century, robotics significantly contributed to the improved efficiency of factories through industrial automation. Although industrial robots have been commonplace for decades, mobile robots are not yet widely accepted in modern society despite their potential benefits. Nevertheless, several companies, such as Amazon robotics (Figure 1.2a)[9] and JD logistics in China [10], have recently begun using autonomous mobile robots for warehouse automation.

The widespread use of the Internet has increased online shopping and food ordering,



(a) Robots in an Amazon warehouse [9].

(b) Amazon delivery drone [11].

(c) A Starship Technologies food delivery robot [12].

Figure 1.2 Autonomous mobile robots.

which involves logistic transportations. The last-mile delivery in logistic transportations is the *final step* in which the package arrives at the customer's door [13]. This is the most time-consuming and expensive part of the shipping process. Amazon has announced the use of delivery drones in the coming years for faster and efficient last-mile deliveries (Figure 1.2b) [11]. Several companies are testing autonomous ground vehicles (AGVs) to solve the efficiency problem of last-mile deliveries. Recently, Starship Technologies in the United States developed and tested a six-wheel robot for the last-mile delivery of groceries and food (Figure 1.2c) [12]. Similarly, a Japanese company, ZMP Inc., is developing an autonomous delivery robot called DeliRo that can carry a maximum load capacity of 50kg [14]. However, owing to local laws, privacy, safety, and reliability concerns, these robots have not been widely used.

Figure 1.3 ASIMO humanoid robot by Honda Motor.

In recent decades, commercial companies have introduced multi-purpose consumer robots, such as vacuum cleaning robots (Roomba from iRobot [15]), service robots (Pepper from SoftBank Robotics [16]), and pet robots (AIBO from Sony Corporation [17]). Additionally, humanoid robots have fascinated both researchers and the general public for a long time. Recently, the bipedal humanoid robot named Atlas (Boston Dynamics) [18] has acquired the ability to run and perform incredible parkours. These feats require significant computational power and highly customized hardware, such as hydraulic actuators. Despite decades of research, most bipedal humanoid robots remain bulky and slow, such as ASIMO by Honda (Figure 1.3) [19]. Alternatively, human-like mobile robots, which are referred to as wheel-legged robots with two independent driving wheels rather than feet, are increasingly investigated. Despite simple dynamics based on linear inverted pendulum (LIP) models, these robots are more agile and energy-efficient than the humanoid robots owing to the presence of wheels.

1.2 Challenges of Controlling Wheel-legged Robots

Underactuated robotic systems comprise fewer control inputs than degrees of freedom (DoFs). In other words, these systems include certain generalized coordinates that are not directly actuated [20]. However, these unactuated coordinates are indirectly controlled by actuated generalized coordinates owing to dynamic coupling. Wheel-legged robots are underactuated robotic systems controlled by only two wheels and three DoFs, namely the translational position, pitch angle, and yaw angle. In this case, the translational position and pitch angle are dynamically coupled; this coupling is nonlinear. The resulting dynamic constraints generated by the nonlinear coupling, also referred to as non-holonomic constraints, are non-integrable. Owing to these underactuated DoFs and non-holonomic constraints of wheel-legged robots, the implementation of conventional control methods is difficult.

It is well established that higher maneuverability in underactuated non-holonomic wheel-legged robots is obtained at the expense of instability and complex control systems. Fall prevention is vital for self-balancing robots because falls can damage these expensive systems and pose danger to nearby humans or other agents. Therefore, ensuring that the robot follows the desired reference commands despite the influence of external disturbances, parametric uncertainties, and modeling errors is a critical challenge for control designers of such systems. For instance, the viscosity coefficients of mechanical actuators change over time owing to the varying abrasion and friction between the wheels and floor based on the terrain. Another difficulty occurs when these self-balancing robots need to carry a payload from one point to another in real-world applications. As the mass and inertia of these payloads are often unknown, the robot must adapt and adjust its balance accordingly.

Typically, wheel-legged robots are intended to be deployed in multiple environments, such as search and rescue, smart warehouses, and homes to assist humans. Therefore, it is difficult to model human-robot interactions in various scenarios a priori, and it requires a robust control framework for safety-critical real-time systems.

1.3 Literature Review

A wheeled inverted pendulum (WIP) system is basically a robot with a base of two independently driving wheels and a body on top of it that acts as an inverted pendulum (see Figure 1.4). These WIP robots can be categorized into two major classes: with legs and without legs. A segway (Figure 1.5c) is an excellent commercial product of the latter type, which is commonly used to transport humans in the bustling city centers of the modern world. However, WIP robots with legs, also referred to here as wheel-legged robots, are relatively new. In 2017, Boston Dynamics, a robotics company in Massachusetts, USA, introduced its first wheel-legged robot called Handle for warehouse pick and place operations [21]. However, the control algorithms used by the company are largely unknown to the research community.

Early examples of pitch balance and trajectory control of a WIP robot were introduced by Yuta *et al.* [22][23]. Tani *et al.* used WIP robots in the cooperative transportation of objects while satisfying three main requirements: attitude stability of the robot against external disturbances, applying an appropriate force on the transported object, and following the given path [24].

Rufer *et al.* developed a prototype of a two-wheeled vehicle at EPFL, Lausanne and named it JOE (Figure 1.4) [2]. They implemented two independent state-space model-based controllers to balance the robot and control its orientation. The control interface of the vehicle consists of power-on and emergency switches. The robot is controlled by a human pilot who sends a desired translational velocity and turning rate through a radio control unit. Furthermore, the control unit is able to safely turn off the vehicle if the pitch angle becomes more than $100\ deg/s$.

Parent *et al.* proposed an intelligent two-wheeled vehicle that could provide on-demand urban taxi services [25]. The control system is designed to perceive the human passenger motion as an external disturbance to be rejected, unlike the motion control system of a Segway that generates a longitudinal acceleration when the rider leans forward,

Figure 1.4 JOE robot from EPFL [2].

thus preventing the passenger from falling. The equations of motion for the robot were derived using the Lagrangian method, model parameters such as friction coefficient of the robot wheels were identified, and lastly, an LQR was designed to control the vehicle.

In another study, Agrawal *et al.* investigated the partial feedback linearization and controllability properties of a WIP robot [26][27]. The study further proved that the three degrees of freedom (translational position, pitch angle, and yaw) nonlinear WIP robot system is strongly accessible (see [28] for details). After the partial feedback linearization, a two level velocity controller is designed to track a reference pitch angle and heading trajectory. However, the proposed control scheme is only validated in simulations and not on a real robot.

A team from Carnegie Mellon University developed a unicycle Ballbot robot capable of balancing itself on a single spherical ball (Figure 1.5a) [29]. One major advantage of such a robot over two-wheel platforms is that it does not need to turn before moving in a particular direction. To balance and control the robot an LQR controller is designed and

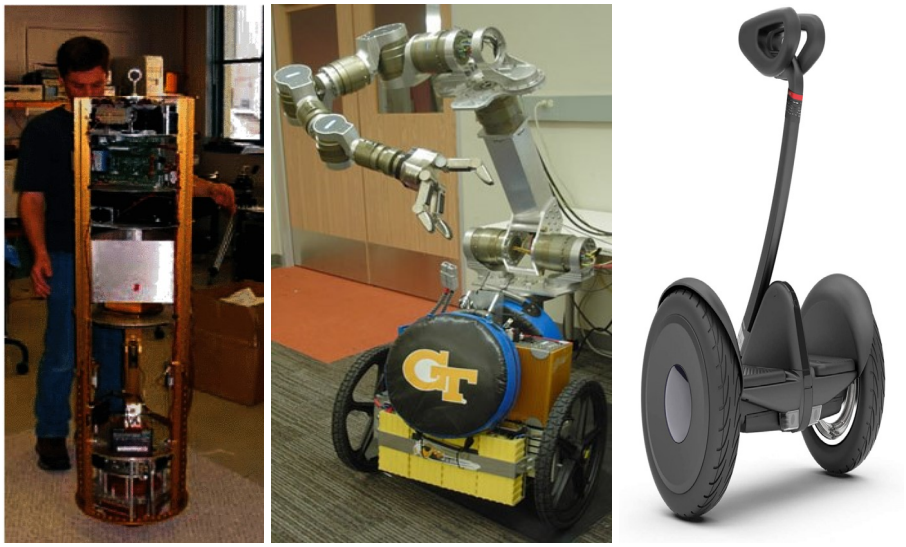
implemented. Nevertheless, this simple controller proved far from robust in the presence of disturbances likely due to the over simplification of the friction model.

A group of researchers at Fukushima University, Japan developed the first human assistant wheel-legged biped robot called I-PENTAR and applied a linear quadratic regulator (LQR) for its steering and pitch control [30] [31]. Furthermore, robot successfully performed sitting and standing motions while keeping its balance. In a later study, the authors developed an extended state observer to estimate and mitigate the external disturbances on the robot during task execution such as a standing up motion [32]. In this study, the robot has only two degrees of freedom (pitch control and translational motion) instead of three for the sake of simplicity.

Gloss *et al.* added a humanoid torso on top of a WIP for mobile manipulation (Figure 1.5b)[33]. The main objective of the robot is to manipulate objects and maneuver in indoor environments. The novel design enabled the robot to autonomously transform from a statically stable robot where its center of mass is within the support polygon of the base to a dynamically stable one standing on its two wheels. The robot is capable of lifting a maximum mass of 120 kg while balancing itself dynamically on two wheels. The motion control framework consists of different modules to control the subsystems of the robot. For example, balance and locomotion are achieved by a cascade of PID controllers with minimal dynamical modeling of the system. On the other hand, the torso and arms are controlled directly by controlling the actuator torques.

1.3.1 Sliding-mode Control of WIP

The application of nonlinear sliding-mode controllers to WIP robots has also been extensively studied and reported by the research community. For example, Sekiyama *et al.* implemented two sliding-mode controllers with a novel sliding surface to stabilize a WIP and remove steady-state tracking errors resulting from parameter uncertainties [34]. Lee *et al.* applied a novel integral sliding-mode controller (ISMC) on a two-wheeled inverted pendulum robot that successfully rejected the matched uncertainties and balanced



(a) Ballbot robot [29]. **(b)** Golem Krang robot [33]. **(c)** A modern segway.

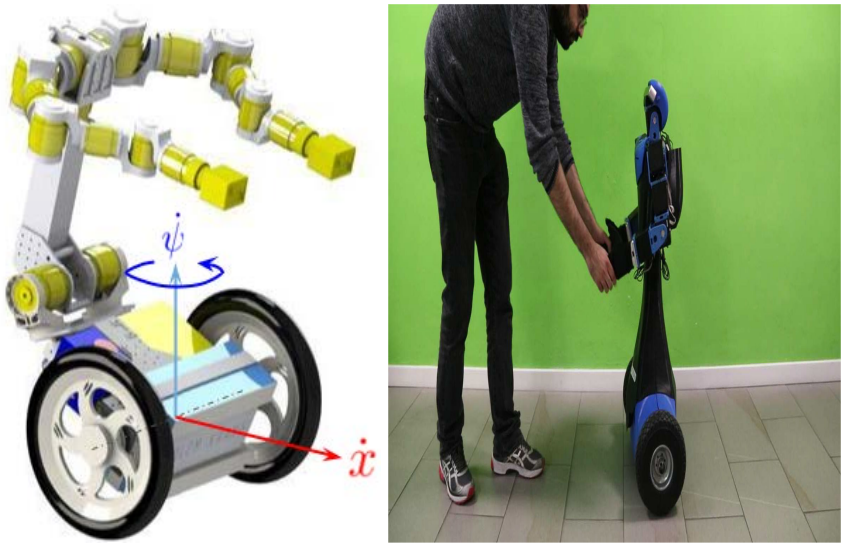
Figure 1.5 Different wheeled inverted pendulum systems.

itself on an inclined plane [35]. In the proposed ISMC, an integral-type sliding surface is defined and the controller is based on Lyapunov theory. ISMC has a nominal linear controller part, and a switching term that rejects matched uncertainties perfectly. Finally, the proposed controller is compared with existing control methods such as fuzzy traveling and position controller (FTPC) and conventional sliding-mode controller.

In another study, Liu *et al.* identified the friction parameters of the drive mechanism in a WIP robot and implemented sliding-mode control to manage the balancing and yaw motions [36]. Before dynamical modeling of the robot, the friction parameters of wheel joints are identified using the Stribeck friction and the Coulomb plus viscous friction models. It is reported that both friction models give similar results, and finally, the Coulomb plus viscous friction model is chosen for real-time experiments on a physical robot. Despite the nonlinear coupling between the two states, two sliding-mode controllers are designed to control the pitch and yaw angle of the robot independently.

1.3.2 Whole-body Control of WIP

Recently, whole-body control of WIP robots with a humanoid torso and manipulator has also become a subject of interest for the research community. These types of robots



(a) A WIP robot with torso [38]. **(b)** Alter-Ego WIP humanoid robot [39].

Figure 1.6 WIP robots with torso.

perform multiple tasks such as dynamic balancing, end-effector manipulation, and maintaining a specific pose in three-dimensional space simultaneously. Furthermore, joint angle and torque limits set constraints on the control system. The whole-body control framework enables these multiple tasks to be performed simultaneously while respecting the constraints of the physical system.

Christensen *et al.* introduced the whole-body control of a WIP humanoid with a manipulator [37]. They performed operational space control of the manipulator by isolating its dynamics from the wheelbase. This enabled the authors to treat the planar horizontal motion of the robot as any other task by the task prioritizing scheme used for the robot manipulator. The proposed whole-body controller is then validated on a 5 DoF planar robot in a simulation environment.

Theodorou *et al.* further worked on a hierarchical optimization for whole-body control of the WIP robot and successfully tested it in simulations [38]. The proposed scheme is based on a low-level controller for the robot's joints and a high-level controller that produces the center of mass trajectory for the low-level controller. Manipulator dynamics of the robot (shown in Figure 1.6a) have been isolated from the wheel-base dynamics as suggested in [37]. Finally, a quadratic programming (QP) optimization algorithm is used

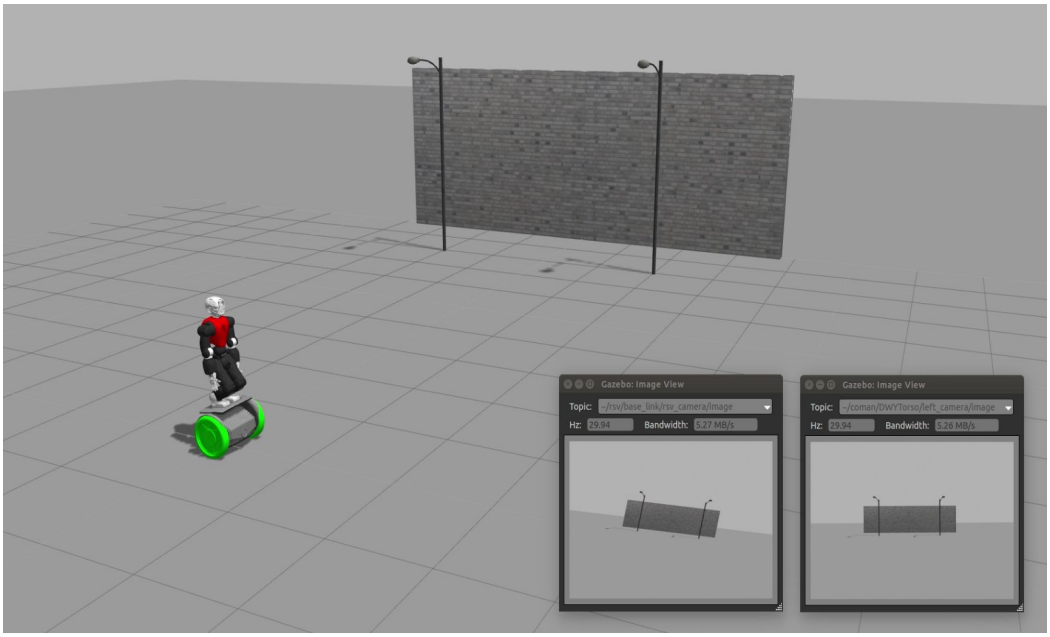


Figure 1.7 Humanoid robot on a Segway with views from the robot's cameras on right bottom [3].

for the low-level controller, while the high-level controller used model-predictive control (MPC) and differential dynamic programming (DDP) for the trajectory optimization of robot's center of mass.

Similarly, Yueyang *et al.* proposed a whole-body control of a WIP robot based on a distributed dynamic model that consisted of the torso model, wheel-leg model, and contact force constraint between the wheels and ground in simulation [40]. However, this robot did not include upper limb arm usage. The proposed control scheme included four sub-modules namely, Torso controller, Torque solver, Slippage predictor, and an online trajectory generator. It is believed that with the distributed whole-body control method, it would be easy to add and control arm manipulators or other limbs on the robot in future.

In another engaging simulation study, Tsagarakis *et al.* explored the question of whether a humanoid robot can ride a Segway type WIP robot [3]. To accomplish this difficult task, they applied quadratic optimization to generate whole-body joint torques for the humanoid robot to ride a WIP platform. On the other hand, to control the WIP base, its dynamics are decoupled in two independent subsystems; translational and pitch motion in the sagittal plane and yaw motion of the base in the transverse plane. Lastly, two

Figure 1.8 Ascento wheel-legged robot from ETH [4].

independent state-feedback controllers are designed to achieve self-balancing and steering of the base. Simulation results verified the balancing of a 29 DoF humanoid robot on top of a WIP base, as seen in Figure 1.7.

More recently, Siegwart *et al.* from ETH Zurich developed a WIP jumping robot (see Figure 1.8) that used a whole-body controller augmented with an LQR for its pitch balance [41]. The mechanical design of the Ascento robot results in open kinematic loops, and therefore its dynamical modeling is a non-trivial task. The authors derived the closed-form rigid-body dynamics of the wheeled bipedal robot by considering the kinematics loop open and then close it through appropriate dynamical constraints. Moreover, a rolling constraint was also added using rotation matrices to increase robot's robustness against curves and lateral disturbances. The roll angle reference is dynamically computed during curve driving in such a way that the zero moment point (ZMP) of the robot stays in the center of the line of support (LoS).

1.3.3 Nonlinear Control of WIP

Whole-body control of WIP robots uses online optimization tools to satisfy various constraints of the system, however, these tools are computationally intensive. To counter this problem, researchers have explored other nonlinear control methods that are compu-

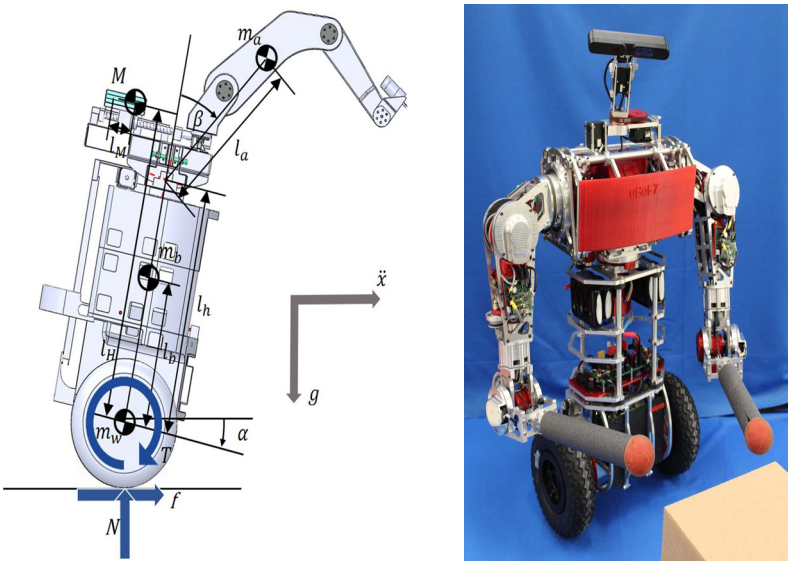
Figure 1.9 A wheel-legged robot from Tencent [5].

tationally simple yet more effective than the nominal linear feedback controllers such as LQR.

Caporale *et al.* demonstrated strong inertial coupling between the wheelbase and pitch dynamics of a humanoid WIP robot (shown in Figure 1.6b) and utilized a nonlinear computed-torque controller in quasi-velocities for stabilization [39]. Different disturbances such as static, dynamic, and opening/closing of a drawer were applied on the robot body without using any disturbance observer. The simulation results showed an improved performance of the proposed controller over a conventional LQR.

Kwon *et al.* presented a nonlinear optimal control design based on the state-dependent Riccati equation (SDRE) [42]. The proposed control framework is a nonlinear form of the LQR and demonstrated better performance than a conventional LQR in controlling the forward position, pitch, and yaw of a two-wheeled inverted pendulum mobile robot. The state-dependent coefficient (SDC) matrix is used as a design parameter that decides the performance, optimality, and stability of the closed-loop system.

Similarly in a more recent study, Ming *et al.* proposed a new SDRE controller design scheme for WIP robots [43]. The study analyzed the SDRE method from a computational perspective, improving the online computational performance by avoiding too many solvability inspections of the algebraic Riccati equations.



(a) Free-body diagram of J4.α [45]. **(b)** uBot-7 [46].

Figure 1.10 WIP robots with arm manipulators.

1.3.4 Machine Learning Methods

To overcome the weaknesses of model-based control methods, machine learning techniques are recently getting attention for balancing and control of wheel-legged robots. Model learning of a WIP robot where a simulation model is learned first and then the difference model to reduce the sim-to-real gap has been reported recently [44]. The robot has three states, pitch angle, pitch rate, and linear acceleration, whereas the control input is the wheel torques. To reduce the computational cost, a sparse Gaussian process (GP) method is used to learn the robot difference model as it requires less data than the typical GP method. After learning the robot model, an optimal control policy is obtained using a reinforcement learning algorithm.

In a very recent study, Jiang *et al.* proposed a novel data-driven value iteration algorithm that generates a balancing controller for a wheel-legged robot with a small amount of data (Figure 1.9) [5]. The given method solves the optimal balance control problem of WIP robots in the absence of accurate modeling using reinforcement learning and adaptive dynamic programming. Experiments on a real WIP robot validated the performance of the data-driven optimal adaptive controller that can compensate for the parameter vari-

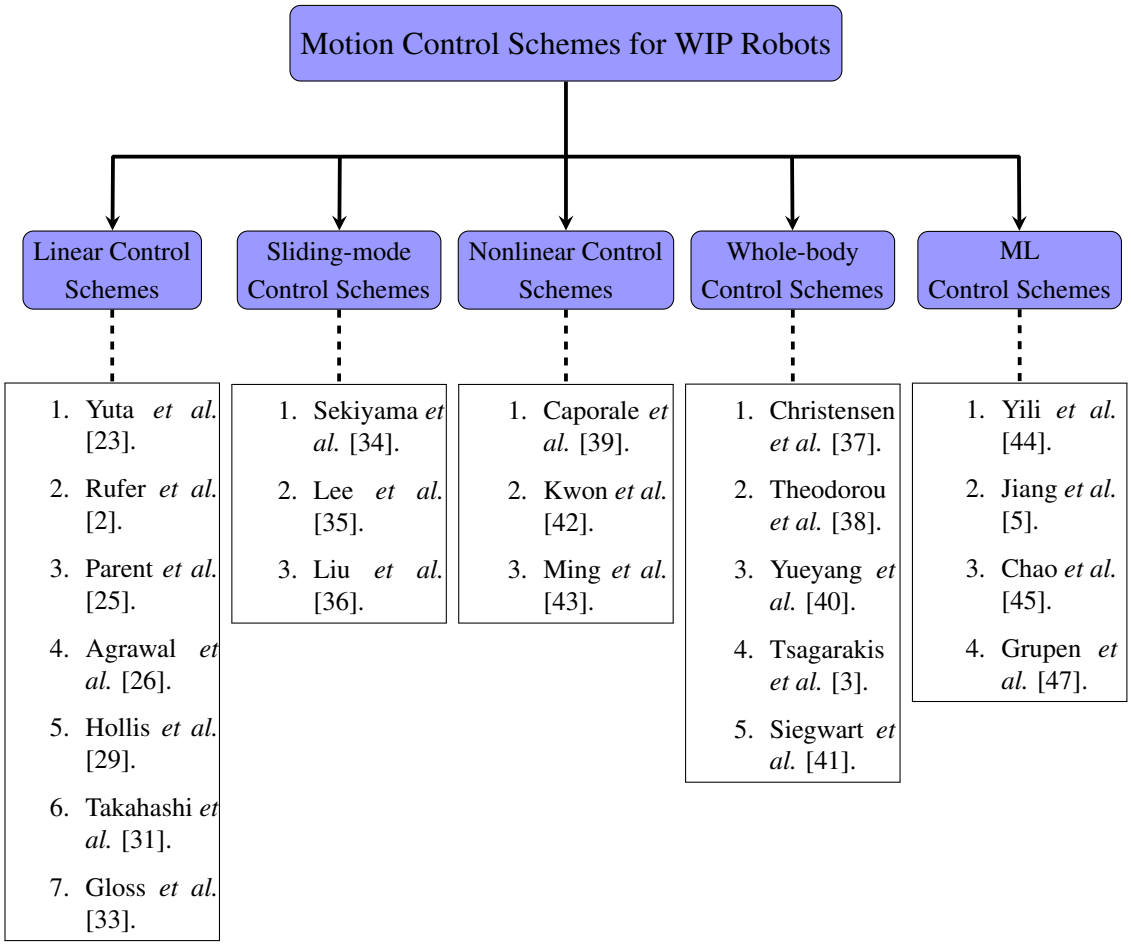


Figure 1.11 An overview of various control schemes for WIP Robots.

ations in real-time.

Chao *et al.* developed a self-balancing wheeled robot called J4.α that uses a deep convolutional neural network for socially compliant navigation in indoor environments (Figure 1.10a) [45]. The proposed navigation system has two modules; a topological localization module (TLM) to recognize the nodal locations along the trajectory and an auto-steering module (ASM) that determines specific tasks to execute at or between nodes. The robot is also equipped with a manipulator and a dynamic mass on top of it. During a manipulation task, the dynamic mass is employed to keep the robot in a steady place. A combination of different controllers including Q-learning is used in an 11 DoF self-stabilizing wheeled robot by Grupen *et al.* [47]. The robot can achieve four basic postures; a prone posture where the robot lies flat on the ground, a push-up posture where it changes

its arms configuration such that its both hands and wheels touch the ground, a 3-point posture where it removes one hand from the ground in order to transit from the push-up posture to the self-stabilizing standing posture, and finally a 2-point balancing where the robot is balanced on its two wheels. An LQR controller is used to stabilize the robot in standing posture, while Q-learning is used to perform knuckle-walking mobility mode. The group further improved the manipulation and mobility of the robot by adding three more degrees of freedom and using series elastic actuators (SEAs) (Figure 1.10b)[46].

1.4 Motivation

As discussed earlier, robots have traditionally been used to perform repetitive, dangerous, and unsanitary jobs. Although robots successfully perform a few of these jobs, predominantly in factories and warehouses, mobile robots are not yet broadly employed in the real world despite decades of research and development. This can be primarily attributed to the lack of reliability, robustness, and safety of mobile robots. The robustness problem generates from the presence of model uncertainties and external disturbances. For instance, unlike industrial robots, mobile robots must operate and interact with different environments; consequently, their control systems are vulnerable to unavoidable uncertainties and external forces.

Typical self-balancing wheel-legged robots provide higher maneuverability, faster speed, and a smaller footprint than bipedal humanoid robots. These unique characteristics make them appealing for indoor environments, such as hotel lobbies, banks, hospitals, restaurants, and warehouses. However, several ongoing research challenges exist in terms of system modeling, stability analysis, path planning, and motion control owing to their self-balancing nature and non-holonomic constraints of the differential drive. It is for these reasons that wheel-legged systems have attracted significant interest from academia and commercial sectors in recent years. A common commercial application of wheeled self-balancing robots is the modern Segway, which helps commuters avoid rush-hour city congestion.

Several challenges, such as the aging population driving labor shortages worldwide and the ever-increasing natural disasters caused by the rapid climate change, can be overcome by mobile robots. As explained earlier, coupling the immense potential of wheel-legged robots for indoor and outdoor environment applications, this thesis primarily explores various balance control approaches to ensure that the safety-critical underactuated wheel-legged robots are safer and robust in real-world scenarios.

1.5 Research Objectives

The objectives of this research are divided into general and specific objectives, as follows.

1.5.1 General Objectives

- To review and summarize the state of the art of wheeled inverted pendulum (WIP) robots
- To perform mathematical modeling and synthesize a robust motion controller for a three-DoF underactuated wheel-legged robot
- To perform stability and robustness analyses of the robot considering modeling uncertainty, sensor noise, and external disturbances

1.5.2 Specific Objectives

Chapter 2 : First Objective

We begin by laying the foundation for the development of a simulation model of the robot and further analyze its behavior under sensor noise, model uncertainties, and external disturbances. The primary objective is to evaluate the feasibility of a wheel-legged robot under different disturbances that can potentially occur when using the robot

around humans. Additionally, the limitations of the linear assumption can be compensated by combining the model-based nonlinear computed torque controller with LQR.

Chapter 3 : Second Objective

The primary objective of this chapter is to explore the possibilities of implementing a state-of-the-art adaptive controller for wheel-legged robots. It is clear that model-based linear and nonlinear controllers along with model-free machine learning approaches have been implemented in WIP robots; however, to the best of the author's knowledge, no adaptive controller has been designed for a bipedal wheel-legged robot thus far. Therefore, an $\mathcal{L}_1$ adaptive controller is augmented with a full state-feedback LQR owing to its fast adaptation ability and robustness against modeling uncertainties and external disturbances.

Chapter 4 : Third Objective

Inspired by humans and other animals, such as cats, who actively use limbs to maintain their balance, we use the arm manipulator of the self-balancing robot to enhance its stability. The purpose of this study is to develop a novel control strategy that elevates the balance recovery ability of the conventional lower-body controller of wheel-legged robots by actively using an upper-body arm manipulator. Furthermore, we adopt centroidal moment pivot (CMP), which is a key criterion for quantifying the stability of legged humanoid robots, to quantitatively analyze the effectiveness of the proposed control method.

1.6 Organization of the thesis

This thesis is structured as follows,

Chapter 1: Introduction

Chapter 1 introduces the study. It summarizes the shortage in labor supply, economic stagnation, and a possible reduction in productivity worldwide caused by the aging pop-

ulation. Furthermore, we discuss the potential applications of wheel-legged biped robots and the need for their robust control. A detailed literature review is also presented to provide a general idea of the state-of-the-art control strategies for WIP robots.

Chapter 2: Towards Robust Wheel-Legged Biped Robot System: Combining Feedforward and Feedback Control

Chapter 2 begins by deriving the mathematical model of a three-DoF wheel-legged robot and designing a full state-feedback controller for the system. Subsequently, an optimal LQR and a nonlinear computed torque method are combined in a newly proposed control framework for stabilizing the wheel-legged biped robot. All these control strategies are implemented in simulations and in an actual robot to demonstrate the proposed method improving the stability of the robot.

Chapter 3: An $\mathcal{L}_1$ Adaptive augmented LQR Control for Wheel-Legged Robots: Design and Experiments

This chapter provides some background on adaptive controllers and proposes an $\mathcal{L}_1$ adaptive augmented control for the wheel-legged robot. An $\mathcal{L}_1$ adaptive controller is designed for the pitch control of the self-balancing wheel-legged robot, and the proposed control framework is validated via simulations and by applying it to an actual robot. The combination of the $\mathcal{L}_1$ adaptive controller with LQR demonstrates the robot overcoming modeling uncertainties, such as errors in parameter values and external disturbances.

Chapter 4: Balance Stability Augmentation for Wheel-legged Biped Robot through Arm Acceleration Control

This chapter explains the design, mathematical modeling, and control strategy for the arm manipulator of the wheel-legged robot. The model-based resolved acceleration control method is used to control the position, velocity, and acceleration of the manipulator's end-effector in the Cartesian space. Additionally, a finite-state machine defines the behavior of the arm's end-effector based on the linear acceleration and rotational velocity of

the robot. The simulation results indicate that the active arm control strategy effectively increases the robustness of the robot against external forces.

Chapter 5: Conclusion and Future Work

Finally, Chapter 5 concludes the thesis with a summary of the results obtained in this study and further addresses the possible future research directions for wheel-legged biped robots.

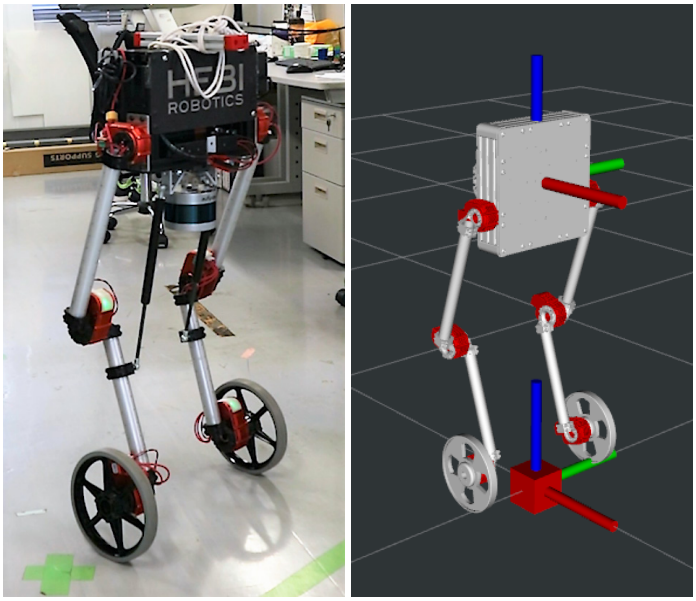
Although an effort has been made to maintain consistent mathematical notations throughout the thesis, notation conventions should be considered chapter-specific.

Chapter 2

Towards Robust Wheel-Legged Biped Robot System: Combining Feedforward and Feedback Control

2.1 Outline

In this chapter, we start by dynamic modeling of the wheel-legged robot which incorporates non-holonomic constraints due to the differential drive of the robot. After linearizing the equations of motion, we synthesize a linear state feedback controller and also an optimal linear quadratic regulator (LQR) to control the three degrees of freedom (DoF) of the robot. The performance of the two controllers under sensor noise and external disturbances is compared. Later, we present the design and implementation of a nonlinear feedforward controller together with feedback LQR to control the robot on a flat surface. Furthermore, an Extended Kalman Filter (EKF) is implemented to reduce the sensor noise, fuse the data from various sensors, and estimate the system states of the real robot. We perform motion stability analyses of the wheel-legged robot under different conditions such as system modeling errors, sensor noise, and external disturbances. Results validate that the linear quadratic regulator (LQR) combined with the nonlinear computed torque controller improves the overall stability of the robot.



(a) Real robot. **(b)** 3D model in ROS Rviz.

Figure 2.1 View of the Igor wheel-legged robot.

2.2 Robot Modeling

Before designing the controller, we present the system dynamic model of the wheel-legged robot (Figure 2.1). In our dynamic modeling of the robot, we assume a simplified inverted pendulum, where the body of the robot acts as a point mass. The origins of the robot-fixed frame Σ_R and body-fixed frame Σ_C are coincident and placed at the midpoint of the wheel axle. For pitch rotation β and wheel torques, the forward direction of the robot is considered positive; whereas for yaw rotation α, counterclockwise motion is assumed positive. The three-dimensional model of the robot is represented in Figure 2.2, whereas Table 2.1 shows all the modeling parameters of the system.

2.2.1 Equations of Motion

We use the Euler-Lagrange formulation to derive the equations of motion for the wheeled biped robot under the following important assumptions.

- There is no slip between the wheels of the robot and ground.
- The wheels are rigid nondeformable disks.

Parameters	Description	Value	Unit
m_c	Mass of the robot body	7.5	Kg
m_w	Mass of the robot wheel	0.35	Kg
l	Height of the CoM from the origin of Σ_R	0.5914	m
I_{xx}	Moment of inertia of the robot body in X	0.0878	Kg.m^2
I_{yy}	Moment of inertia of the robot body in Y	0.0347	Kg.m^2
I_{zz}	Moment of inertia of the robot body in Z	0.0632	Kg.m^2
I_{wa}	Moment of inertia of the robot wheel about the wheel axis	0.0018	Kg.m^2
I_{wd}	Moment of inertia of the robot wheel about its diameter	0.000929	Kg.m^2
I_{ma}	Moment of inertia of the motor rotor about its axis	$1.04E-7$	Kg.m^2
I_{md}	Moment of inertia of the motor rotor about its diameter	0.0	Kg.m^2
g	Gravitational acceleration constant	9.81	m/s^2
r_w	Wheel radius	0.1016	m
b	Distance between the origin of Σ_R and the center of the wheel	0.2150	m
γ	Gear reduction ratio	1	
c_r, c_l	Viscosity coefficients of left and right wheels	0.17	$\frac{Nm}{(rad/sec)}$
I_1	Moment of inertia of the arm link-1	0.018	Kg.m^2
I_2	Moment of inertia of the arm link-2	0.0339	Kg.m^2
l_1	Length of the arm link-1	0.3	m
l_2	Length of the arm link-2	0.3	m
l_{g1}	Distance of CoM of the arm link-1 from its origin	0.3	m
l_{g2}	Distance of CoM of the arm link-2 from its origin	0.3	m
m_1	Mass of the arm link-1	0.6	Kg
m_2	Mass of the arm link-2	1.13	Kg

Table 2.1: Robot Parameters.

- The structure of the robot is symmetrical.
- The legs of the robot are massless.

The first two assumptions provide the necessary conditions for the kinematic modeling of

the differential drive base of the robot. Nevertheless, the affects of these assumptions on robot's performance would be presented in our future study by comparing the simulation results with the real robot.

We now begin by writing equations for the position of the center of mass (CoM) in inertial frame Σ_I. A left subscript is used from here onwards to clearly specify the coordinate frame of the vector quantities. The equations are

$$\begin{aligned} {}_I x_c &= {}_I x_r + l\cos(\alpha)\sin(\beta), \\ {}_I y_c &= {}_I y_r + l\sin(\alpha)\sin(\beta), \\ {}_I z_c &= r_w + l\cos(\beta). \end{aligned} \tag{2.1}$$

We obtain the translational velocity vector of the CoM of the robot simply by taking the time derivative of the above equations:

$${}_I V_c = \begin{bmatrix} {}_I\dot{x}_r + l\left(\dot{\beta}\cos(\alpha)\cos(\beta) - \dot{\alpha}\sin(\alpha)\sin(\beta)\right) \\ {}_I\dot{y}_r + l\left(\dot{\beta}\sin(\alpha)\cos(\beta) + \dot{\alpha}\cos(\alpha)\sin(\beta)\right) \\ -l\dot{\beta}\sin(\beta) \end{bmatrix}. \tag{2.2}$$

The rotational velocity of the robot frame Σ_R with respect to inertial frame Σ_I is given as

$${}_I\Omega_R = \begin{bmatrix} 0 \\ 0 \\ \dot{\alpha} \end{bmatrix}. \tag{2.3}$$

We take the rotational velocity of the CoM in the body-fixed Σ_C frame to get a diagonal inertia matrix and simplify our kinetic energy equation:

$${}_C\Omega_c = \begin{bmatrix} -\dot{\alpha}\sin(\beta) \\ \dot{\beta} \\ \dot{\alpha}\cos(\beta) \end{bmatrix}. \tag{2.4}$$

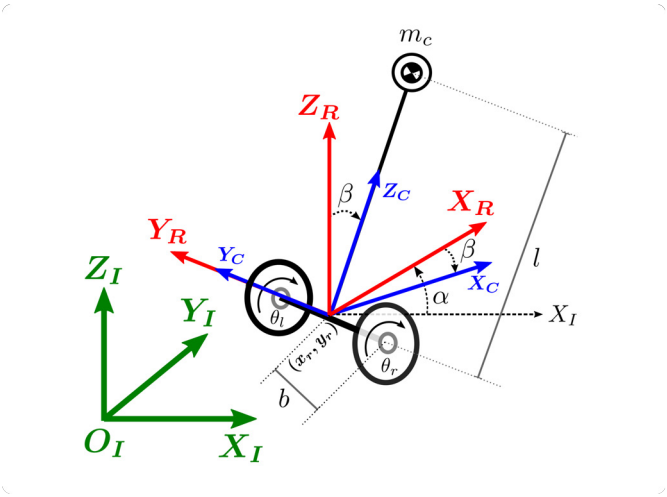


Figure 2.2 Model of the robot with generalized coordinates.

The translational and rotational kinetic energy of the CoM is calculated as

$$\begin{aligned} T_c &= \frac{1}{2} m_c \, {}_I V_c^T \, {}_I V_c + \frac{1}{2} \, {}_C\Omega_c^T \, I_c \, {}_C\Omega_c \\ &= \frac{1}{2} m_c \Big[{}_I\dot{x}_r^2 + 2 \, {}_I\dot{x}_r l \big(\dot{\beta} \cos(\alpha) \cos(\beta) - \dot{\alpha} \sin(\alpha) \sin(\beta) \big) \\ &\quad + {}_I\dot{y}_r^2 + 2 \, {}_I\dot{y}_r l \big(\dot{\beta} \sin(\alpha) \cos(\beta) + \dot{\alpha} \cos(\alpha) \sin(\beta) \big) \\ &\quad + (l\dot{\beta})^2 + l^2 \dot{\alpha}^2 \sin^2(\beta) \Big] \\ &\quad + \frac{1}{2} \big(I_{xx} \dot{\alpha}^2 \sin^2(\beta) + I_{yy} \dot{\beta}^2 + I_{zz} \dot{\alpha}^2 \cos^2(\beta) \big). \end{aligned} \tag{2.5}$$

The translational and rotational kinetic energy of the two wheels is given by

$$\begin{aligned} T_w &= \frac{1}{2} m_w r_w^2 \dot{\theta}_r^2 + \frac{1}{2} m_w r_w^2 \dot{\theta}_l^2 \\ &\quad + \frac{1}{2} \big(I_{wa} + I_{ma} \gamma^2 \big) \big(\dot{\theta}_r^2 + \dot{\theta}_l^2 \big) + \big(I_{wd} + I_{md} \big) \dot{\alpha}^2. \end{aligned} \tag{2.6}$$

The potential energy of the wheel is considered zero, as we assume the wheel remains in contact with the ground everywhere. Consequently,

$$U_w = 0. \tag{2.7}$$

The potential energy of the CoM is

$$U_c = m_c g l \cos(\beta), \tag{2.8}$$

where g is the gravitational acceleration constant.

Rayleigh's dissipation function relating to viscous forces is given by

$$F = \frac{1}{2} c_r \left(\dot{\theta}_r - \dot{\beta}\right)^2 + \frac{1}{2} c_l \left(\dot{\theta}_l - \dot{\beta}\right)^2. \tag{2.9}$$

The Lagrangian of a mechanical system is described by taking the difference between kinetic and potential energies of the system:

$$\begin{aligned} L &= K.E - P.E \\ &= T_c + T_w - U_c. \end{aligned}$$

The equations of motion are then calculated by solving

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} + \frac{\partial F}{\partial \dot{q}_i} = Q_i, \tag{2.10}$$

where q_i represents generalized coordinates and Q_i generalized forces.

For the wheel-legged robot, we choose the generalized coordinate vector $q = [{}_I x_r \; {}_I y_r \; \alpha \; \beta \; \theta_r \; \theta_l]^T$. After solving Eq. (2.10) by substituting for L and F, motion equations of the robot are expressed in familiar matrix form as

$$M(q)\ddot{q} + V\dot{q} + H(q, \dot{q}) + G = E\tau, \tag{2.11}$$

where $M(q) \in \mathbb{R}^{6\times6}$ is the inertia matrix, $V \in \mathbb{R}^{6\times6}$ is a square matrix of viscous friction terms, $H(q, \dot{q}) \in \mathbb{R}^6$ is a vector of Coriolis and centrifugal terms, $G \in \mathbb{R}^6$ is a vector of the gravitational force, $E \in \mathbb{R}^{6\times2}$ is the torque selection matrix, and $\tau \in \mathbb{R}^2$ is the vector

of control torques. These are expressed as

$$M(q) = \begin{bmatrix} m_c & 0 & -c_4 & c_5 & 0 & 0 \\ 0 & m_c & c_3 & c_6 & 0 & 0 \\ -c_4 & c_3 & c_1 & 0 & 0 & 0 \\ c_5 & c_6 & 0 & I_{yy} + m_c l^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & c_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & c_2 \end{bmatrix},$$

$$V = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & c_r + c_l & -c_r & -c_l \\ 0 & 0 & 0 & -c_r & c_r & 0 \\ 0 & 0 & 0 & -c_l & 0 & c_l \end{bmatrix},$$

$$H(q, \dot{q}) = \begin{bmatrix} -(c_3\dot{\alpha}^2 + c_3\dot{\beta}^2 + 2c_6\dot{\alpha}\dot{\beta}) \\ -(c_4\dot{\alpha}^2 + c_4\dot{\beta}^2 - 2c_5\dot{\alpha}\dot{\beta}) \\ c_7\dot{\alpha}\dot{\beta} \\ -\frac{c_7}{2}\dot{\alpha}^2 \\ 0 \\ 0 \end{bmatrix},$$

$$G = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -m_c g l sin(\beta) \\ 0 \\ 0 \end{bmatrix},$$

$$E = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -1 & -1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \tau = \begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix},$$

where $c_1, c_2, \ldots, c_7$ are

$$\begin{aligned} c_1 &= 2\big(I_{wd} + I_{md}\big) + I_{xx} + \big(I_{zz} - I_{xx} - m_c l^2\big)\cos^2(\beta) \\ &\quad + m_c l^2, \\ c_2 &= m_w r_w^2 + I_{wa} + I_{ma}\gamma^2, \;\; c_3 = m_c l \cos(\alpha)\sin(\beta), \\ c_4 &= m_c l \sin(\alpha)\sin(\beta), \;\; c_5 = m_c l \cos(\alpha)\cos(\beta), \\ c_6 &= m_c l \sin(\alpha)\cos(\beta), \;\; c_7 = \big(I_{xx} + m_c l^2 - I_{zz}\big)\sin(2\beta). \end{aligned}$$

2.2.2 Non-holonomic Constraints

It is well known that Eq. (2.11) is only valid for a system without non-holonomic constraints. Therefore, before any standard state-space based control design can be applied, an alternative approach is needed to represent the motion and constraints of the system [48].

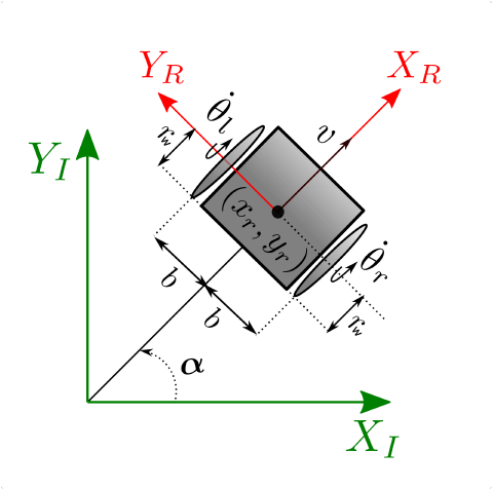


Figure 2.3 Top view of differential drive robot.

Under rolling and no-slip conditions that engender non-holonomic constraints, the kinematic equations of the differential drive robot shown in Figure 2.3 are [**?**]

$$ {}_I\dot{x}_r \sin(\alpha) - {}_I\dot{y}_r \cos(\alpha) = 0 \tag{2.12} $$

$$ {}_I\dot{x}_r \cos(\alpha) - {}_I\dot{y}_r \sin(\alpha) = \frac{r_w}{2}\left(\dot{\theta}_r + \dot{\theta}_l\right) \tag{2.13} $$

$$ \dot{\alpha} = \frac{r_w}{2b}\left(\dot{\theta}_r - \dot{\theta}_l\right) \tag{2.14} $$

The above three equations can be written in matrix form as $J(q)\dot{q} = 0$, where

$$ J(q) = \begin{bmatrix} \sin(\alpha) & -\cos(\alpha) & 0 & 0 & 0 & 0 \\ \cos(\alpha) & \sin(\alpha) & b & 0 & -r_w & 0 \\ \cos(\alpha) & \sin(\alpha) & -b & 0 & 0 & -r_w \end{bmatrix}. \tag{2.15} $$

Once we have equations for the constraints of the system, Eq. (2.11) is modified to account for k kinematic constraints as

$$ M(q)\ddot{q} + V\dot{q} + H(q,\dot{q}) + G = E\mathcal{T} + J(q)^T\lambda, \tag{2.16} $$

where $\lambda \in \mathbb{R}^k$ is a vector of Lagrange multipliers. The term $J(q)^T\lambda$ represents the vector of reaction forces at the generalized coordinate level.

We now use the standard method to remove the Lagrange multipliers from Eq. (2.16) as elaborated in [49]. We first define a full-rank matrix $S(q) \in \mathbb{R}^{n\times m}$ that lies in the null-space of matrix $J(q)$; here, $m = n - k$, and n is the number of generalized coordinates. It is noted that the choice of matrix $S(q)$ is not unique.

$$S(q) = \begin{bmatrix} \cos(\alpha) & 0 & 0 \\ \sin(\alpha) & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ \frac{1}{r_w} & \frac{b}{r_w} & 0 \\ \frac{1}{r_w} & \frac{-b}{r_w} & 0 \end{bmatrix}. \tag{2.17}$$

A new vector $u \in \mathbb{R}^m$ of *pseudo-velocities* is defined such that

$$\dot{q} = S(q)u. \tag{2.18}$$

In the literature, Eq. (2.18) is mentioned as the kinematic model of the constrained mechanical system. In this study, we select $u = [v\ \dot{\alpha}\ \dot{\beta}]^T$ so that the controller design can be achieved rather simply. Differentiation of the above equation leads to

$$\ddot{q} = \dot{S}(q)u + S(q)\dot{u}. \tag{2.19}$$

The substitution of Eqs. (2.18) and (2.19) into Eq. (2.16) yields

$$\begin{aligned} M(q)\dot{S}(q)u + M(q)S(q)\dot{u} + VS(q)u + H(q,\dot{q}) + G \\ = E\mathcal{T} + J(q)^T\lambda. \end{aligned} \tag{2.20}$$

Finally, the Lagrange multipliers can be eliminated by premultiplying both sides of Eq. (2.20) by $S(q)^T$ for the reason that $S(q)J(q) = 0$, and we get the reduced dynamic model

of m differential equations:

$$\hat{M}(q_r)\dot{u} + \hat{V}u + \hat{H}(q_r, u) + \hat{G} = \hat{E}\tau, \tag{2.21}$$

where $\hat{M}(q_r) = S(q)^T M(q) S(q)$ is positive definite, $\hat{V} = S(q)^T V S(q)$, $\hat{H}(u) = S(q)^T \big[M(q)\dot{S}(q)u + H(q, \dot{q})\big]$, $\hat{G} = S(q)^T G$, $q_r = [p\ \alpha\ \beta]^T$, and $\hat{E} = S(q)^T E$.

Equation (2.21) is a set of nonlinear equations that cannot be used in designing a linear controller, and we thus have to linearize Eq. (2.21) about the equilibrium point ($\beta = 0$). Making a small-angle approximation results in

$$\begin{aligned} \sin(*) &\simeq (*), \\ \sin^2(*) &\simeq 0, \\ \cos(*) &\simeq 1, \\ (\dot{*})^2 &\simeq 0, \\ (\dot{*})(*) &\simeq 0. \end{aligned}$$

By virtue of the above conditions, $\hat{H}$ is eliminated from Eq. (2.21) and we finally get

$$\hat{M}\dot{u} + \hat{V}u + \hat{G} = \hat{E}\tau, \tag{2.22}$$

where $\hat{M}$ and $\hat{G}$ have linearized elements. Furthermore, the above equation implies that

$$\begin{aligned} \dot{u} &= \hat{M}^{-1}\big[\hat{E}\tau - \hat{V}u - \hat{G}\big] \\ &= -\hat{M}^{-1}\big(\hat{V}u + \hat{G}\big) + \hat{M}^{-1}\hat{E}\tau. \end{aligned} \tag{2.23}$$

It is noted that θ_r and θ_l are completely decoupled from other state variables. Additionally we find $\dot{\theta}_r$ and $\dot{\theta}_l$ given the forward velocity $v = \frac{r_w}{2}(\dot{\theta}_r + \dot{\theta}_l)$ and Eq. (2.14). We can therefore now reduce the order of the system and define a configuration vector $q_r = [p\ \alpha\ \beta]^T$ as an actual control variable, where $p = {}_I x_r \cos(\alpha) + {}_I y_r \sin(\alpha)$.

2.2.3 State-space Representation

State-space modeling is generally adopted to convert N^{th}-order system dynamics to a system of N first-order differential equations. We can formulate the state-space model as we have already obtained linearized equations of motion for the wheel-legged robot. A general linear-time-invariant state-space model is represented as

$$\begin{aligned} \dot{X} &= AX + BU \\ Y &= CX + DU, \end{aligned} \tag{2.24}$$

where $X \in \mathbb{R}^n$ is called the state vector, $Y \in \mathbb{R}^i$ is the output vector of the system, and $U \in \mathbb{R}^p$ is the system input. The constant matrices A, B, C, and D are respectively called dynamics, input, output, and feedforward matrices. In this study, we define the state vector as

$$X = \begin{bmatrix} q_r \\ u \end{bmatrix}_{6\times 1}$$

and the control input as $U = \tau$. The state-space model of the robot is therefore given by

$$\begin{aligned} \dot{X} = \begin{bmatrix} \dot{q}_r \\ \dot{u} \end{bmatrix} = &\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{H_1}{D_1} & \frac{-H_2}{D_1} & \frac{H_3}{D_1} & \frac{H_4}{D_1} \\ 0 & 0 & 0 & b\frac{a_5}{D_2} & -b^2\frac{a_4}{D_2} & -br_w\frac{a_5}{D_2} \\ 0 & 0 & \frac{H_5}{D_3} & \frac{H_6}{D_3} & \frac{-H_7}{D_3} & \frac{-H_8}{D_3} \end{bmatrix} X \\ &+ \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ e_1 & e_1 \\ e_2 & -e_2 \\ e_3 & e_3 \end{bmatrix} \tau, \end{aligned} \tag{2.25}$$

where

$$
\begin{aligned}
a_1 &= I_{wa} + I_{ra}\gamma^2 + m_w r_w^2, \\
a_2 &= I_{rd} + I_{wd} + 0.5 I_{zz} + m_w b^2, \\
a_3 &= I_{wa} + I_{ra}\gamma^2, \quad a_4 = c_r + c_l, \\
a_5 &= c_l - c_r, \quad a_6 = I_{yy} + m_c l^2 + m_c l r_w, \\
a_7 &= 2a_3 + r_w^2(m_c + 2m_w) + m_c l r_w, \\
e_1 &= \frac{a_6 r_w}{D_1}, \quad e_2 = \frac{b r_w}{D_2}, \quad e_3 = -\frac{a_7 r_w}{D_3}, \\
D_1 &= 2a_1\big(I_{yy} + m_c l^2\big) + I_{yy} m_c r_w^2, \\
D_2 &= 2\big(a_2 r_w^2 + a_3 b^2\big), \\
D_3 &= 2r_w\big[I_{yy}(a_1 + 0.5 m_c r_w^2) + a_1 m_c l^2\big], \\
H_1 &= -g(m_c l r_w)^2, \;\; H_2 = a_4 a_6, \\
H_3 &= a_5 a_6 b, \;\; H_4 = a_4 a_6 r_w, \\
H_5 &= g m_c l r_w[2a_3 + r_w^2(m_c + 2m_w)], \;\; H_6 = a_4 a_7, \\
H_7 &= a_5 a_7 b, \;\; H_8 = a_4 a_7 r_w.
\end{aligned}
$$

For a full-state-feedback system, the matrix C is an identity matrix and $D = 0$ if the system has no feedforward.

2.3 State Feedback Control

After defining the state space model, our objective is to stabilize the robot in vertical equilibrium position ($\beta = 0$) along with controlling its translational motion p. First, we use linear state feedback controller to control our robotic system. State feedback controller is a powerful MIMO (Multiple Input Multiple Output) control method that uses system's state vector X to compute the necessary control action. It can be formulated as,

$$U = ref - K_{sf} X, \; K_{sf} \in \mathbb{R}^{p \times n} \tag{2.26}$$

Where ref is the reference signal, U is the control input, and K_{sf} is the state feedback gain vector. If $ref = 0$, the state feedback controller is usually called a state feedback regulator.

The state feedback controller manipulates the system dynamics matrix A by using gain matrix K_{sf} to stabilize the system. After replacing U in (2.24) by equation (2.26) we obtain,

$$\begin{aligned} \dot{X} &= AX + B(ref - K_{sf}X) \\ &= (A - BK_{sf})X + B(ref) = A_{cl}X + B(ref) \end{aligned} \tag{2.27}$$

Here A_{cl} represents the dynamics matrix of the closed-loop system. Since we can manipulate the closed-loop dynamics matrix A_{cl} by an appropriate K_{sf}, we can change the pole positions of our system to desired locations in the s-plane.

For a system with n number of poles, we find the appropriate gain vector K_{sf} by solving equation (2.28) for K_{sf}.

$$det(sI - A_{cl}) = (s + a_1)(s + a_2)\dots(s + a_n) \tag{2.28}$$

Where $a_1, a_2, \dots, a_n$ are the desired system poles locations in the complex s-plane. This technique is also known as pole placement method.

To determine feedback gain K_{sf} for which our system is stabilized, we develop the state-space model in *MATLAB*. After putting the parameters of the robot (Table-2.1) in *MATLAB* model, we compute the poles of the open-loop system. All the four poles lie on the imaginary-axis of the s-plane, therefore making the system marginally stable. For stabilizing our system, we choose new poles at $[-12.8775, -0.5127, -0.7705 + 0.4525i, -0.7705 - 0.4525i]$ in the s-plane. We use *MATLAB*'s *place* command to find the feedback gain K_{sf} corresponding to the newly chosen poles.

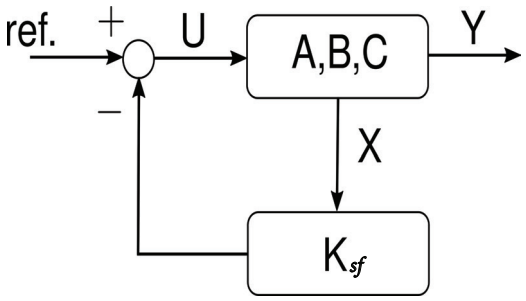


Figure 2.4 State-feedback Control Scheme.

2.3.1 Linear Quadratic Regulator (LQR)

In a simple state-feedback controller, we choose the closed-loop poles of a system by trial and error; however, these pole locations usually do not provide the optimal control performance. To overcome this problem, we design an LQR controller for our system, which allows us to find the optimal feedback gain vector K_{lqr}. The LQR places the poles in such a way that the closed-loop system optimizes the cost function

$$J_{lqr} = \int_0^\infty [X(t)^T Q X(t) + U(t)^T R U(t)]\, dt, \tag{2.29}$$

where $X^T Q X$ is the state cost with weight Q, while $U^T R U$ is the control cost with weight R [50]. The optimal state-feedback law is then given as

$$U = ref - K_{lqr} X. \tag{2.30}$$

Where as before, ref is the reference signal. Here the necessary condition for optimality is

$$K_{lqr} = R^{-1} B^T P, \tag{2.31}$$

where P is calculated by solving the algebraic Riccati equation (ARE) [50]

$$0 = A^T P + P A + Q - P B R^{-1} B^T P. \tag{2.32}$$

This leads to the following closed-loop system,

$$\dot{X} = \big(A - BR^{-1}B^T P\big)X \tag{2.33}$$

Furthermore, the following theorem has been proved.

Theorem 2.1

If there exists a symmetric solution P to the ARE (2.32) *for which* $A - BR^{-1}B^T P$ *is a stable matrix, then the feedback control law* $U = ref - K_{lqr}X, \forall t \geq 0$ *minimizes the LQR cost function,*

$$J_{lqr} = \int_0^\infty [X(t)^T Q X(t) + U(t)^T R U(t)]\, dt.$$

It is also important to choose Q and R matrices carefully to get the best possible results. We use the well-known Bryson's method [51], which gives a simple and reasonable choice to determine Q and R as

$$Q = \begin{bmatrix} \frac{{w_1}^2}{(x_{11})^2_{max}} & 0 & \dots & 0 \\ 0 & \frac{{w_2}^2}{(x_{22})^2_{max}} & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \dots & \frac{{w_n}^2}{(x_{nn})^2_{max}} \end{bmatrix}, \tag{2.34}$$

where $(x_{ii})_{max}$ denotes the largest desired response for that component of the state vector. R is determined as

$$R = \rho \begin{bmatrix} \frac{{b_1}^2}{(u_{11})^2_{max}} & 0 & \dots & 0 \\ 0 & \frac{{b_2}^2}{(u_{22})^2_{max}} & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \dots & \frac{{b_j}^2}{(u_{jj})^2_{max}} \end{bmatrix}. \tag{2.35}$$

Here, $(u_{jj})_{max}$ is the maximum desired control input for the system and ρ is used as the last relative weighting between the control and state penalties. Furthermore, w_i and b_i are respectively the weights for each state and control input. We define w_i and b_i according to

$$\sum_{i=1}^{n} {w_i}^2 = 1,$$
$$\sum_{i=1}^{j} {b_i}^2 = 1.$$

Even though Bryson's method usually yields satisfactory results, it is often just the beginning of a trial-and-error iterative method of obtaining the desired closed-loop system response. Using Bryson's rule and then making alterations, we have $Q = diag([0.78\ 6.37\ 39.06\ 0.25\ 0.19\ 11.11])$, $\rho = 1$, and $R = diag([0.03\ 0.03])$ which leads to the satisfactory reference tracking of the system states. Finally, the optimal feedback gain

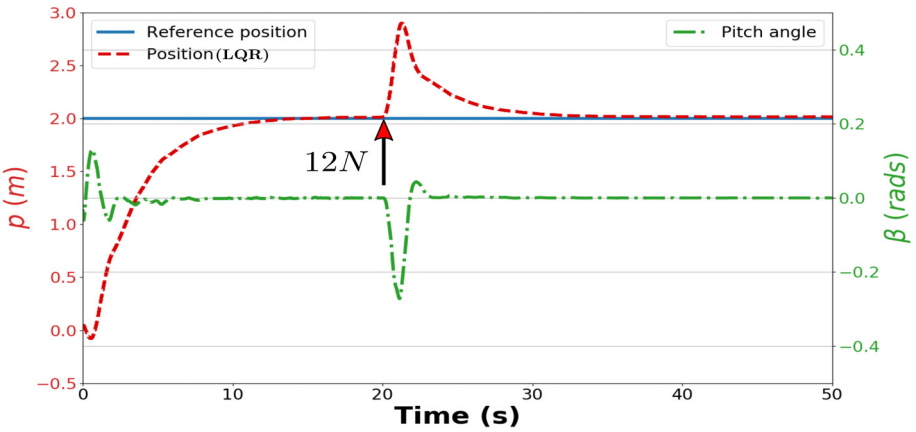
$$K_{lqr} = \begin{bmatrix} -2.8284 & 1.4142 & -19.4797 & -4.8849 & 0.4032 & -4.7839 \\ -2.8284 & -1.4142 & -19.4797 & -4.8849 & -0.4032 & -4.7839 \end{bmatrix}.$$

is acquired using lqr command in *MATLAB (Mathworks Inc., Natick, MA, USA)* with system parameters listed in Table 2.1. As the robot steers its heading using a differential drive mechanism, we note that the corresponding gain values for α and $\dot{\alpha}$ states in K_{lqr} have opposite signs.

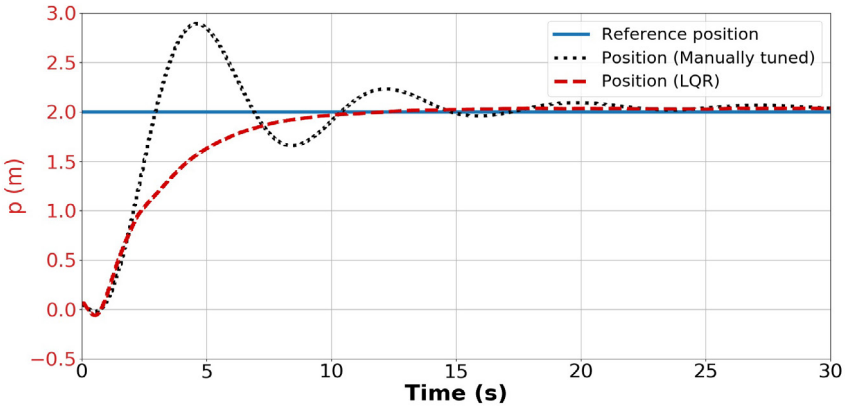
2.3.2 Numerical Simulation Results

Simulation Setup

With the purpose of simulating our wheeled biped robot in the *Gazebo* simulator [52], we modeled our robot in URDF (Universal Robotic Description Format) using the dimensions of the Igor robot as shown in Figure 2.1. The URDF model includes all the physical constraints of the real robot, such as joint limits, static friction, and damping coefficients, and the actuator's torque-speed characteristics. *Gazebo* is an open-source simulator that



(a) LQR step response and translational push of 12 N.



(b) Manually tuned controller vs. LQR.

Figure 2.5 Translational position step response.

uses *ODE* [53] as its physics engine. ROS (Robot Operating System) is used to implement the control algorithm in C++, thus controlling the robot in *Gazebo*. The *ROS-Gazebo* simulation runs at a frequency of 500 Hz with a motion controller steering the wheeled biped robot in the desired direction while keeping the robot in an upright position at the same time.

Results and Discussion

A series of tests are conducted to validate the effectiveness of the designed controller. Simple reference trajectories comprising step and sinusoidal signals are chosen for the robot. Given that an off-the-shelf robot employs a simple proportional-integral-derivative (PID) controller for balancing, we considered it necessary to compare the designed LQR

Mass Change	**Noise** $\sigma = 0.0$			**Noise** $\sigma = 0.02$		
	p_{err}	α_{err}	β_{err}	p_{err}	α_{err}	β_{err}
-20%	13.72E-3	19.33E-5	73.77E-5	17.84E-3	9.11E-5	74.98E-5
-10%	25.25E-3	18.94E-5	23.91E-5	28.27E-3	21.7E-5	48.87E-5
0%	36.54E-3	22.49E-5	18.11E-5	36.63E-3	18.6E-5	39.15E-5
+10%	42.07E-3	25.04E-5	54.38E-5	44.97E-3	21.28E-5	68.07E-5
+20%	50.11E-3	28.74E-5	86.19E-5	52.19E-3	29.02E-5	92.32E-5

Table 2.2: Steady-state errors in the presence of model uncertainties and Gaussian noise.

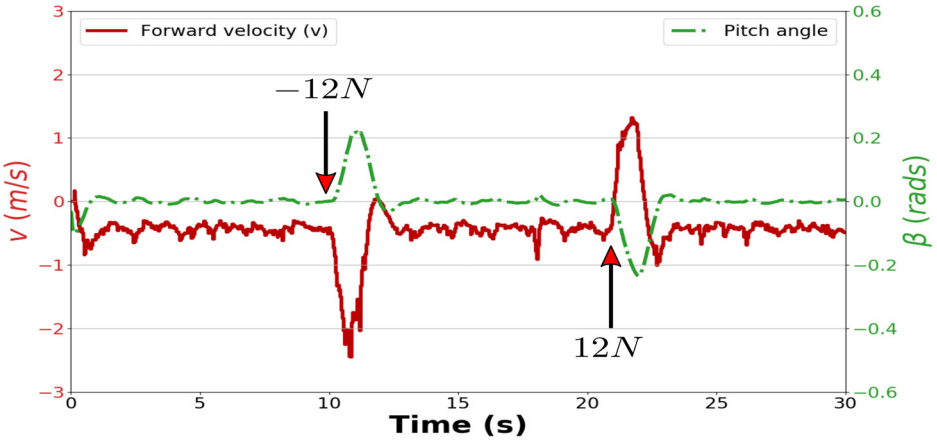
with a manually tuned state-feedback controller of gain

$$K_{sf} = \begin{bmatrix} -1.09 & 0.51 & -21.37 & -0.87 & 0.03 & -8.26 \\ -1.09 & -0.51 & -21.37 & -0.87 & -0.03 & -8.26 \end{bmatrix}.$$

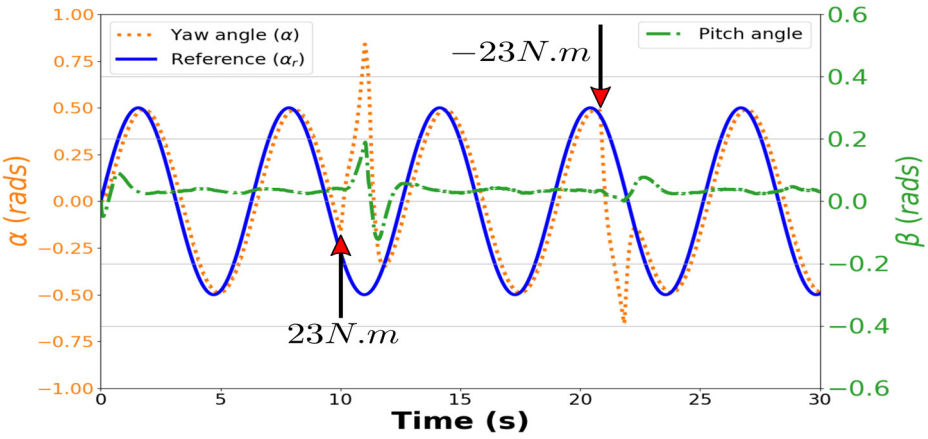
This is an important point of the study since the off-the-shelf robot is not using the LQR controller, thus by this comparison, we can check how much the disturbance rejection capability of the current robot can be potentially improved for the future use. Results of *ROS-Gazebo* simulations are presented in Figures 2.5 and 2.6.

Figure 2.5a shows the translational reference position, step response of the LQR, pitch angle, and system's recovery from a force of 12 N applied in the frontal plane at the CoM of the robot body. One can see that the robot leaned backwards because of the strong push but the controller brought the robot back to the reference point smoothly while keeping the pitch angle below 0.25 rad.

In another test, the results of which are shown in Figure 2.5b, we compare the translational position step response of the optimal controller with the manually tuned state-feedback control method. It is clear from the simulation results that the LQR controller has a shorter settling time than the state-feedback controller. Most importantly, the LQR has no overshoot or oscillations, which is essential for human-robot interaction. Although state-feedback and LQR control laws have the same theoretical basis and one can argue that the state-feedback controller would perform better with some further fine tuning, the sole purpose of the comparison here is to determine how well our designed LQR performs



(a) Forward and backward push during translational motion.



(b) Counterclockwise and clockwise moment during yaw tracking.

Figure 2.6 LQR tracking and disturbance rejection.

against non-optimal off-the-shelf robot controllers.

To ensure the robustness of the LQR controller against the model uncertainties and sensor noise, various tests are performed and the results are summarized in Table 2.2. To account for model uncertainties, we change the masses of the body m_c and wheel m_w along with the corresponding moment of inertia by $\pm 10\%$ and $\pm 20\%$. Two series of tests are conducted, namely tests without sensor noise and tests including Gaussian white noise with standard deviation $\sigma = 0.02$. Finally, steady-state errors for reference positions $p_r = 2$ meters, $\alpha_r = 0.78$ rad, and $\beta_r = 0$ rad are determined.

The results clearly show that regardless of sensor noise, the steady-state error for the translational position tracking decreases as the mass of the system is reduced. This

External Force	**Mass Change = 0%**				**Mass Change = -20%**			
	LQR		**Manual Tuning**		**LQR**		**Manual Tuning**	
	$\int\|e_v\|dt$	t_r	$\int\|e_v\|dt$	t_r	$\int\|e_v\|dt$	t_r	$\int\|e_v\|dt$	t_r
12 N	1.660	2.8	4.117	5	1.597	3	4.782	6
8 N	1.038	2.6	2.664	4	1.169	3	3.093	5.9
4 N	0.549	2.4	1.439	3.4	0.620	2	1.517	4.8
External Torque	$\int\|e_\alpha\|dt$	t_r	$\int\|e_\alpha\|dt$	t_r	$\int\|e_\alpha\|dt$	t_r	$\int\|e_\alpha\|dt$	t_r
21 N.m	0.578	1.6	N/A	N/A	0.582	1.6	N/A	N/A
14 N.m	0.362	1.5	4.873	4	0.356	1.5	N/A	N/A
7 N.m	0.206	1.2	2.772	2.8	0.206	1.3	2.924	2.8

Table 2.3: Disturbance rejection comparison of the LQR and manually tuned controller. Forces and moments are applied at the CoM of the body during forward motion and sinusoidal yaw tracking, respectively. e_v, e_α, and t_r are respectively the linear velocity error, yaw angle error, and recovery time. N/A indicates that the robot fell as a result of an external disturbance.

should not come as surprise because it is conventional behavior of a proportional feedback control method. We also find that the pitch angle (β) tracking is sensitive to the model irregularities and the steady-state error can increase by a factor of 4 with a mere change of $\pm 20\%$ in the system mass. Meanwhile, we observe that yaw tracking of the robot improves as we reduce the mass and inertia of the system; however, this difference is not large.

Figure 2.6 depicts the robot's recovery from rather strong external disturbances during translational and rotational movements. Figure 2.6a shows the application of two external forces on the body while the robot is moving at constant linear velocity $v = -0.5$ m/s. The first force is applied in the robot's direction of motion while the second force is applied in the direction opposing the robot' s direction of motion. Similarly, Figure 2.6b shows the application of external torques about the z-axis of the robot body as rotational disturbance. Different disturbances are applied on the system and we find that the peak translational and rotational push that the robot can sustain with the current LQR is 12 N and 23 N.m, respectively.

The values in Table 2.3 are used to quantitatively demonstrate the disturbance rejec-

Figure 2.7 Igor performing a slalom in the *Gazebo* simulator.

tion of the LQR and compare it with the disturbance rejection of the manually tuned state-feedback controller. All errors are taken as the area under the curve from the time a disturbance is applied till the moment the robot recovers to its steady state. The table shows that the LQR has smaller errors and a shorter recovery time than the state-feedback controller. Additionally, the LQR endures much higher external torques as compared with the manually tuned controller. Furthermore, we have also tested the system's disturbance rejection capability in case of overestimation of the mass and inertia values during controller design process. As system becomes lighter, it gets more prone to external disturbances. Simulation results point out that the error, and recovery time increase slightly in case of translational disturbances for the LQR as compared to the manually tuned controller, where the change is more significant. On the other hand, alterations in the mass and moment of inertia did not effect the performance of the controllers during rotational disturbances.

Finally, to check whether the LQR controller can handle the translational and rotational accelerations simultaneously, we perform a slalom maneuver as shown in Figure 2.7. We observe that the robot can carry out the slalom easily at peak forward velocity of 1 m/s.

2.4 Feedforward-Feedback Controller

In this section, we propose a simple control framework for the wheel-legged robot that combines nonlinear Computed Torque control with the linear LQR feedback controller, as shown in Figure 2.8. In the real world, not only actuators have different friction coefficients but they also change over time because of abrasion and other factors that cannot be modeled beforehand. More importantly, a slight mismatch of these parameters in a differential drive robot severely deteriorates its performance.

Our main motivation is to implement a computationally simple yet robust controller that may compensate for such model uncertainties and external disturbances. Through combining the model-based nonlinear computed torque controller with the linear LQR, the limitation of linear assumption can be compensated. Its performance was verified in a physics-engine based Gazebo simulator and also with the real robot experiments to reveal the characteristics of the controllers: feedback LQR, feedforward Computed Torque, and the combination.

2.4.1 Computed Troque Control

The Computed Torque method is the simple model-based nonlinear control law that computes input torques based on the manipulator dynamics equation of the system [54]. We rewrite nonlinear Eq. (2.21) as

$$\tau = \hat{E}^{-1}\big(\hat{M}(q_r)\dot{u} + \hat{V}u + \hat{H}(q_r, u) + \hat{G}(q_r)\big). \tag{2.36}$$

To get the desired configuration vector $q_r = [p\ \alpha\ \beta]^T$, we design the acceleration $\dot{u}$ as

$$\dot{u} = \dot{u}_d + K_v(u_d - u) + K_p(q_{r_d} - q_r), \tag{2.37}$$

where $\dot{u}_d$, u_d, q_{r_d} are the desired acceleration, velocity, and position vectors respectively, whereas K_v and K_p are the diagonal gain matrices for the respective velocities and positions.

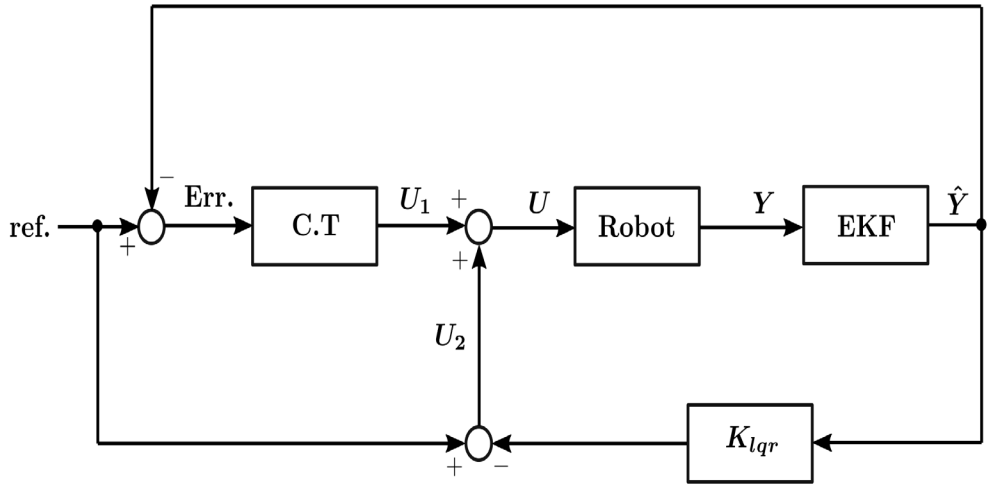


Figure 2.8 Combined LQR and Computed Torque control scheme.

By putting Eq. (2.37) back in Eq. (2.36) we get

$$\begin{aligned} U_1 = \tau = \hat{E}^{-1}\big[\hat{M}(q_r)\big(\dot{u}_d + K_v(u_d - u) + K_p(q_{r_d} - q_r)\big) \\ + \hat{V}u + \hat{H}(q_r, u) + \hat{G}(q_r)\big]. \end{aligned} \tag{2.38}$$

With some experimentation we find the position gains

$$K_p = \begin{bmatrix} -9.1 & 0 & 0 \\ 0 & -25 & 0 \\ 0 & 0 & -61.75 \end{bmatrix},$$

and the velocity gains

$$K_v = \begin{bmatrix} -2.65 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & -13 \end{bmatrix}.$$

2.4.2 Robot Localization

To localize the real robot and estimate its pose, we use the kinematic model of the differential drive to obtain wheel odometry, and inertial measurement unit (IMU) sensors for linear accelerations and angular velocities. Each actuator of the real robot shown in Figure 2.1a includes an IMU, however, we only use two IMUs of the left and right hip actuators for the localization purposes. Finally, linear accelerations and angular velocities from the IMUs are fused with the wheel odometry data through an extended Kalman filter to obtain the pose of the robot.

Wheel Odometry

In this section, we derive the forward kinematics equation of the differential wheel drive of the robot shown in Figure 2.3. We rewrite the rolling and no-sliding constraint equations from section 2.2.2 in matrix form. Rolling constraints are given by,

$$\begin{bmatrix} 1 & 0 & b \\ 1 & 0 & -b \end{bmatrix} \begin{bmatrix} {}_I\dot{x}_r \\ {}_I\dot{y}_r \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} r_w & 0 \\ 0 & r_w \end{bmatrix} \begin{bmatrix} \dot{\theta}_r \\ \dot{\theta}_l \end{bmatrix} \tag{2.39}$$

No-sliding constraints equations are given as below,

$$\begin{bmatrix} 0 & -1 & 0 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} {}_I\dot{x}_r \\ {}_I\dot{y}_r \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{2.40}$$

Now we combine Eq. (2.39) and (2.40) in the form,

$$\begin{bmatrix} 1 & 0 & b \\ 1 & 0 & -b \\ 0 & -1 & 0 \\ 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} {}_I\dot{x}_r \\ {}_I\dot{y}_r \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} r_w & 0 \\ 0 & r_w \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_r \\ \dot{\theta}_l \end{bmatrix} \tag{2.41}$$

To find the robot's linear velocities and yaw angle on the horizontal plane, we rewrite Eq. (2.41) as,

$$\begin{bmatrix} {}_I\dot{x}_r \\ {}_I\dot{y}_r \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} 1 & 0 & b \\ 1 & 0 & -b \\ 0 & -1 & 0 \\ 0 & -1 & 0 \end{bmatrix}^{\dagger} \begin{bmatrix} r_w & 0 \\ 0 & r_w \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_r \\ \dot{\theta}_l \end{bmatrix} \tag{2.42}$$

where † represents the pseudo inverse of the matrix. Solving Eq. (2.42) leads to the final result for the forward differential kinematics as,

$$\begin{bmatrix} {}_I\dot{x}_r \\ {}_I\dot{y}_r \\ \dot{\alpha} \end{bmatrix} = \begin{bmatrix} \frac{r_w}{2} & \frac{r_w}{2} \\ 0 & 0 \\ \frac{r_w}{2b} & -\frac{r_w}{2b} \end{bmatrix} \begin{bmatrix} \dot{\theta}_r \\ \dot{\theta}_l \end{bmatrix} \tag{2.43}$$

Therefore, forward velocity of the robot is,

$${}_I\dot{x}_r = r_w \frac{(\dot{\theta}_r + \dot{\theta}_l)}{2},$$

and no-sliding

$${}_I\dot{y}_r = 0,$$

and finally yaw velocity is

$$\dot{\alpha} = r_w \frac{(\dot{\theta}_r - \dot{\theta}_l)}{2b}.$$

Extended Kalman Filter (EKF)

To filter and fuse the wheel odometry with the multiple IMUs signals, we use an Extended Kalman Filter (EKF). The conventional Kalman filter assumes that the system is described in linear state-space form and the measurements are also linear functions

of the system states. However, most of the real world systems operate in a nonlinear environment, and hence the linear assumptions of the Kalman filter become invalid. EKF is the extension of the Kalman filter that models the environment by a set of nonlinear differential equations or

$$\dot{x} = g(x) + w \tag{2.44}$$

where x is a vector of the system states, $g(x)$ is a nonlinear state transition function, and w is the Gaussian process noise. The vector x consists of 12 states i.e. robot's 3D location, 3D orientation, and their respective velocities. Eq. (2.44) can be written in discrete time form as follows,

$$x_k = g(x_{k-1}) + w_{k-1} \tag{2.45}$$

The process noise covariance matrix is given by

$$Q_p = E(ww^T)$$

where E is the expectation operator. Additionally, the nonlinear measurement equation is given by

$$z = h(x) + v \tag{2.46}$$

where, $h(x)$ is a nonlinear function that maps states into measurement space and v is the Gaussian measurement noise. Eq. (2.46) can be given in discrete time as

$$z_k = h(x_k) + v_k. \tag{2.47}$$

The measurement noise covariance matrix is then defined as

$$R_m = E(vv^T)$$

R_m is a matrix of variances of each measurement noise source.

Since the functions $g(x)$ and $h(x)$ are nonlinear, a first-order approximation is used. Now, the prediction step of the Kalman filter algorithm in discrete time form is given as,

$$\hat{x}_k = g(x_{k-1}) \tag{2.48}$$

$$\hat{P}_k = GP_{k-1}G^T + Q_p \tag{2.49}$$

where P_k is the covariance matrix of the estimate errors and G is the Jacobian of the nonlinear function $g(x)$,

$$G = \frac{\partial g(x)}{\partial x}\Big|_{x=\hat{x}}$$

Next the correction step of the Kalman filter algorithm is given by,

$$K = \hat{P}_k H^T \big(H\hat{P}_k H^T + R_m\big)^{-1} \tag{2.50}$$

$$x_k = \hat{x}_k + K(z_k - H\hat{x}_k) \tag{2.51}$$

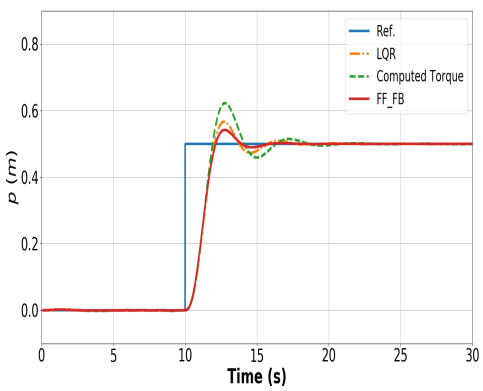
$$P_k = \big(\mathbb{I} - KH\big)\hat{P}_k\big(\mathbb{I} - KH\big)^T + KR_mK^T \tag{2.52}$$

where H is the Jacobian of $h(x)$ that is given as,

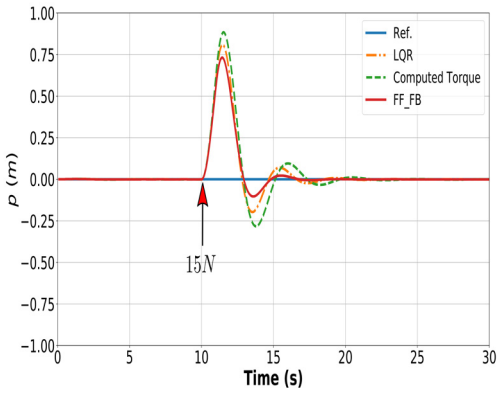
$$H = \frac{\partial h(x)}{\partial x}\Big|_{x=\hat{x}}$$

2.4.3 Numerical Simulations and Real-Time Experiments

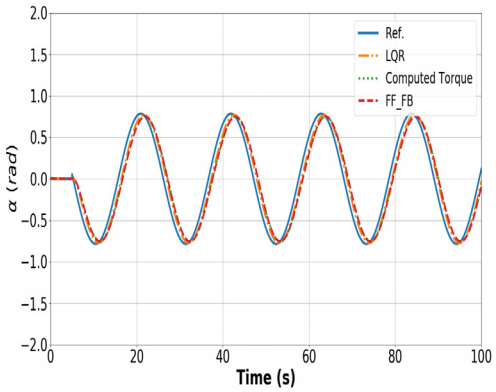
In this section, we report the results of three different tests to characterize the motion controllers for the self-balancing wheel-legged robot. First and foremost is the forward position step response of the robot. Another essential factor to consider for indoor robots is the human-robot interaction, therefore the second test is used to check the external push rejection of the system. Finally, the third test is applied to inspect the sinusoidal yaw tracking of the robot. To validate and assess all of the three controllers, these tests have



(a) Translational position step response.



(b) Translational position evolution against an external push.



(c) Yaw angle tracking of the three controllers.

Figure 2.9 ROS-Gazebo simulation results.

been performed first in the simulation and then on a real wheel-legged robot. Figure 2.8 illustrates the motion control framework that combines both; the nonlinear feedforward Computed Torque and the linear feedback LQR controller. The motion control loop in simulation as well on the real robot runs at the rate of 500 Hz. It is also important to mention that all these tests are carried out on a flat and non-slippery surface to keep the modeling assumptions valid as far as possible.

Numerical Simulation Results

First, we started with the forward position step response for all the three controllers which can be seen in Figure 2.9a. A step of $0.5m$ is applied at the $10th$ second of the

simulation. It is clear from the results that the combined FF-FB controller outperforms the other two controllers in overshoot and settling time. Although, the difference between the step responses of LQR and FF-FB is negligible.

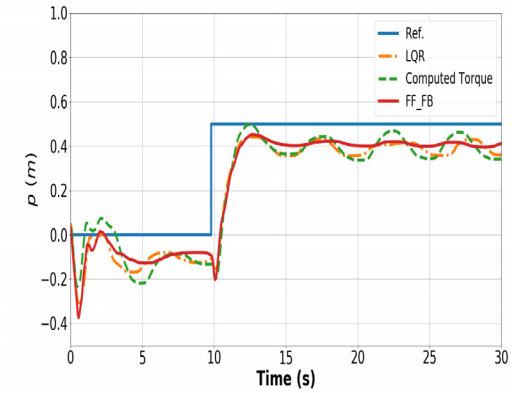
In the second test, we applied a linear force of $15N$ for the entirety of 1 second on the body of the robot as shown in Figure 2.9b. In spite of the fact that the FF-FB controller can uphold an external push of up to $20N$ in the simulator, we chose a force that can be sustained by all of the three controllers under consideration in this study. This test also validates that the FF-FB controller handles this external disturbance better than the other two controllers. Moreover, it is evident that the Computed Torque controller gives the highest oscillatory reaction to the disturbance.

The third and final test is carried out to observe the yaw tracking performance of the robot. As seen in Figure 2.9c, a sinusoidal reference signal of $\pi/4\ rad$ amplitude is applied to the system. However, it is clear that all three controllers have identical results. This implies that the inertial effect by the body rotation task is smaller compared to the inertial effect by the body translation task.

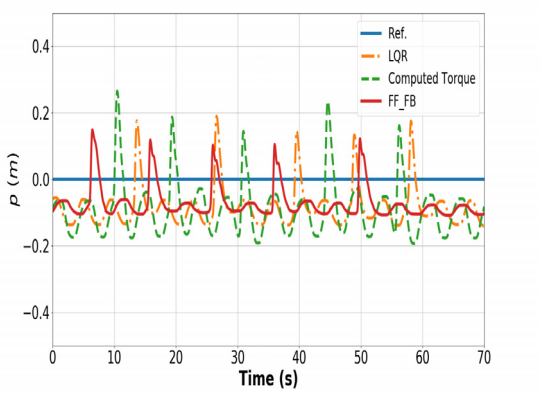
Real-Time Experiments

For real robot experiments, a wheel-legged biped robot from Hebi Robotics company [55] is employed, which is shown in Figure 2.1a. The robot's series elastic actuators are equipped with IMUs, and hence it eliminates the need of using any external IMU on the robot. Furthermore, reading and writing commands to the actuators can be sent from any computer using a Wi-Fi connection. To fuse the data from multiple sensors and have a better estimate of the robot's states, we used the Extended Kalman Filter (EKF). Finally, forward position (p), yaw angle (α), pitch angle (β), and their corresponding rates of change are estimated successfully using the wheel odometry and IMU data from the two hip actuators of the robot.

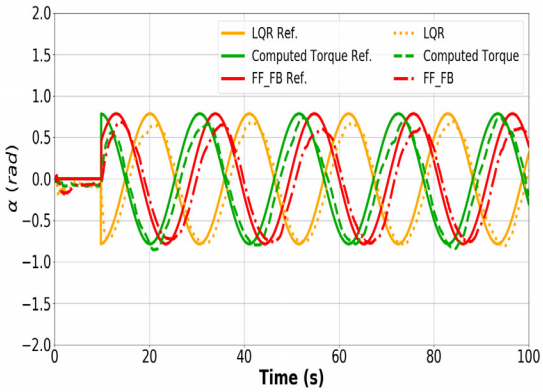
All three tests that are carried out in simulations are repeated on the real robot as well and these results are presented in Figure 2.10. The forward position step response is



(a) Translational position step response.



(b) Translational position evolution against external pushes.



(c) Yaw angle tracking of the three controllers.

Figure 2.10 Real robot experimental results.

shown in Figure 2.10a, where the robot moves from a standstill position to $0.5m$ in the forward direction. It can be seen that all of the three controllers give a small steady state error, however, the FF-FB controller evidently gives a more stable response.

Since the real robot lacks force sensors, we have applied manual force on the robot body multiple times to find out the average response of the control system against external pushes. During these experiments, we tried to apply the disturbances of the same magnitude; however, it must be noted that all these pushes were applied by a human and are not repeatable. It is also clear from the results in Figure 2.10b that the combined FF-FB control law is better in rejecting these disturbances.

In Figure 2.10c, yaw angle (α) tracking of the real robot is presented. All the three

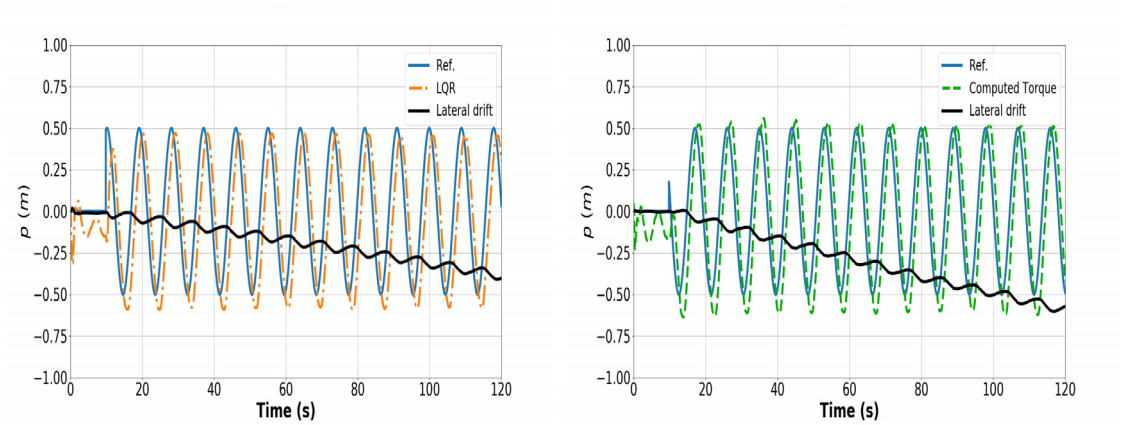
controllers have different sinusoidal reference tracking signals, which start at 10 seconds. It is apparent from the results that all the three controllers perform very similar when it comes to yaw tracking on the real robot as well.

In the final test, we give a sinusoidal reference trajectory for the translational motion p of the robot, and the results are presented in Figure 2.11. Lateral drift is commonly observed when mobile robots transverse over a long period of time without any absolute localization sensor such as a GPS. This lateral drift usually cannot be noticed just by the step response of the system, therefore, we observed the robot's translational motion for about 2 minutes. It is clear from the results that the robot started drifting away in all three cases as time passes; however, in the case of the model-based computed torque controller the lateral drift is about 0.6 meters in 110 seconds, more than the other two controllers. Whereas, this lateral drift in the case of the LQR and the proposed feedforward-feedback controller is about 0.4 meters for the same time period, as shown in Figures 2.11a and 2.11c, respectively. It is clear that the feedback nature of LQR compensated lateral drift better than the model-based computed torque controller that heavily depends on the system modeling.

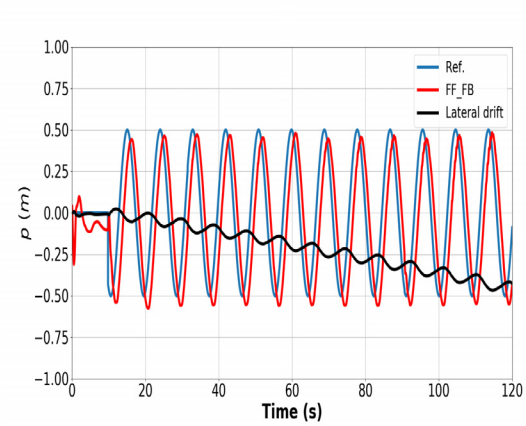
2.4.4 Discussion

This section deliberates over the results of the feedforward-feedback controller and the LQR obtained in various experiments. It becomes clear from the experimental results that all three control methods achieve satisfactory performance during trajectory tracking and external disturbance rejections. However, the combination of the linear feedback LQR and nonlinear Computed Torque control method surpasses the other two controllers in most of the measured metrics as expected.

While there are no steady state errors during the forward position step response in the simulations, yet the real robot experimental results expose this behavior for all of the three control methods. These steady state errors of about less than $0.1m$ arise because of the obvious discrepancy between the mathematical model of the robot and the real sys-



(a) Translational trajectory following with LQR. **(b)** Translational trajectory following with Computed Torque controller.



(c) Translational trajectory following with FF-FB controller.

Figure 2.11 Real robot translational trajectory following.

tem. Furthermore, steady-state errors in multi-input-multi-output (MIMO) underactuated systems occur due to the nonlinear coupling of different states and may require advanced control strategies such as partial feedback linearization to overcome such problems. One naive approach to overcome this problem is to raise the corresponding translational position gains of the motion controllers higher, however, due to the fact that the robot's pitch angle (β) is highly coupled with its translational position (p), the robot becomes unstable.

During the second test (translational pushes) on the real robot, it is particularly clear that not only the proposed feedforward-feedback controller keeps the translational position error smaller than the other two control methods but also the pitch angle β as well.

During these tests, the maximum pitch angle observed for the combined feedforward-feedback controller is 0.111 rads, while it is 0.135 and 0.132 rads for the Computed Torque and LQR, respectively.

Another merit of the combined feedforward-feedback controller becomes clear during the translational position sinusoidal trajectory tracking of the real robot. Given that the viscous friction coefficients of the real actuators cannot be equal, a lateral drift is observed during the straight translational motion of the robot. As expected, the feedforward Computed Torque controller does the least to compensate for these modeling errors and drifts about $0.6m$ in 2 minutes. On the other hand, both the feedback LQR and the combined feedforward-feedback controller keeps the lateral drift less than $0.4m$ during the same time period.

All in all, the differences between the simulation and the real robot results have revealed that the model-based nonlinear Computed Torque controller or the linear feedback LQR is inadequate to compensate for the modeling errors and other uncertainties that cannot be modeled. However, the combination of the two control methods perfectly complements the overall performance without any extra mathematical intricacies and computational resources.

2.5 Conclusion

In this chapter, we performed dynamic modeling of a self-balancing robot using the Euler-Lagrange method to control the translation, rotation, and pitch of the robot. Furthermore, non-holonomic constraints due to the differential-drive wheel system were integrated into the mathematical model of the robot. A linear quadratic regulator based on the mathematical model was then designed for the motion control of the robot. Instead of relying on differential-equation based *MATLAB* simulations to verify our controller of choice, we used the physics-engine based *Gazebo* simulator because it includes rigid-body dynamics, collision detection, and friction and thus provides a better approximation

of the real physical system. One advantage of our approach is the immediate application of it on the off-the-shelf Igor robot. Several simulation test runs were carried out in the presence of model uncertainties, external disturbances, and sensor noise to quantitatively investigate the robustness of the motion controller in detail. Later, a manually tuned state-feedback controller was designed and compared with the LQR. As a result, it became clear that the robot with the LQR endures almost twofold stronger external torques than with the simpler manually tuned controller. Thanks to the LQR controller, the robot can track reference trajectories within a short time period. Also, the robot performs extremely well under the sensor noise, and external translational and rotational perturbations of up to 12 N and 21 N.m, respectively.

To further improve the performance of the robot, we finally presented a computationally simple yet robust control of the three degrees of freedom namely translational position (p), steering or yaw angle (α), and pitch angle (β). From the nonlinear equations of motion of the robot, a nonlinear Computed Torque controller is derived. It is demonstrated through real robot experiments that modeling discrepancies and uncertainties deteriorate the performance of model-based nonlinear feedforward controller as well as the linear feedback LQR, and hence a combination of both controllers performed better in the real world.

Chapter 3

An $\mathcal{L}_1$ Adaptive augmented LQR Control for Wheel-Legged Robots: Design and Experiments

3.1 Outline

In this chapter, we propose the augmentation of an $\mathcal{L}_1$ adaptive controller with a feedback Linear Quadratic Regulator (LQR) to control a wheel-legged biped robot. The performance of linearized model-based controllers, such as LQR, depends on the accurate knowledge of model parameters, a priori information about input and output disturbances, and other unforeseen conditions. However, as robots are required to operate in real-world dynamic environments, robust adaptive control strategies need to be adopted in order to compensate for parameter uncertainties, modeling errors, and external disturbances. We propose a hybrid scheme where an $\mathcal{L}_1$ adaptive controller is combined with LQR to compensate for matched uncertainties and other disturbances related to the environment change such as friction conditions of the floor. The proposed control scheme is able to keep the robot stable under model uncertainties and external disturbances through a series of validation scenarios including simulations and real-time experiments.

3.2 Introduction

In the previous chapter, we developed a hybrid control scheme that combines the LQR controller with the nonlinear computed torque controller. However, both the nonlinear computed torque and the LQR controllers depend on the system model and hence, underperform in real life where the robot parameters and operating environments are not known perfectly. To further improve the stability of the wheel-legged biped robot, we introduce a new control framework that combines LQR with the state-of-the-art $\mathcal{L}_1$ adaptive controller. One significant advantage of the $\mathcal{L}_1$ adaptive controller over the other adaptive control schemes such as Model Reference Adaptive Controller (MRAC) is that it decouples the adaptation from the robustness of the controller and therefore enables high adaptation gains.

To design a motion control system for autonomous robots, which are capable of performing challenging maneuvers under uncertain conditions, has gained a lot of interest recently. In real-world conditions, extra sensor placements and the addition of the payload changes the center of mass (CoM) position of the robot, which may result in the overall instability of the robot. Furthermore, the environmental conditions change; for example, an autonomous car has to track the road lane markings in nominal conditions as well as in sand or snowstorm when visibility becomes poor. Similarly, indoor mobile robots have to adapt to different floor and illumination conditions. For these reasons, adaptive controllers play an important role to avoid significant degradation in the system's performance by compensating for the external disturbances through self-tuning.

Model reference adaptive control (MRAC) architecture remained the theoretical basis of the earlier adaptive controllers. One of the major advantages of this architecture is that it does not depend on the system dynamics model. DesForges *et al.* designed an MRAC for a 3-DoF robotic manipulator [56]. The control scheme consisted of a PD controller with online adjustable gains, and the coupling between the manipulator joints was ignored. The proposed adaptive controller achieved excellent performance despite geometric nonlinearities of the manipulator and different characteristics of the manipulated

objects. In a later study, Tomizuka *et al.* proposed an MRAC controller for mechanical manipulators that compensated for the nonlinear dynamics and coupling among the joints [57]. However, the proposed controller could not be implemented on real-time applications due to its large number of estimation parameters.

In another study, Slotine *et al.* suggested a new adaptive control algorithm for robot manipulators that consists of a PD feedback loop and a full dynamics feedforward part [58]. The controller estimated the manipulator and payload parameters online without knowing the joint accelerations. Other advantage of the proposed algorithm was that it was computationally simple and could be applied directly in Cartesian space. A major drawback of the proposed algorithm was to obtain an optimal adaptation gain matrix to avoid excitation of high frequency dynamics. The author further extended this method to accurately control the attitude of a spacecraft carrying some unknown mass [59].

Annaswamy *et al.* introduced an adaptive controller for the plants with unknown parameters and demonstrated the robustness of the controller in the presence of bounded external disturbances through various simulations [60][61]. It was also observed that the high adaptation gain of the controller deteriorates the system stability; on the other hand, a small adaptation gain reduces the rate of convergence.

Adaptive control methods are significantly popular among the underwater robotics community because these robots are constantly exposed to the inherent nonlinearities of hydrodynamics which are difficult to model. For example, Fossen *et al.* proposed a hybrid control scheme that combined an adaptive controller with a sliding mode controller to control a six DoF underwater mobile robot [62].The uncertainties in the input matrix term due to the nonlinear thruster hydrodynamics were compensated by the switching term of the sliding mode controller. In a separate study, Antonelli *et al.* proposed a new adaptive controller for underwater mobile robots that compensated for the generalized restoring forces and the ocean currents [63].

It is clear from previous studies that MRAC architecture couples the control loop with

the estimation loop, and hence a high adaptation gain brings the closed-loop system towards instability. It happens mainly because the adaptive gain in MRAC controller architecture behaves as a feedback gain and therefore, a trade-off has to be made between the rate of adaptation and the robustness of the closed-loop system. To address the coupling problem between the adaptation rate and robustness, Naira *et al.* introduced a low-pass filter in MRAC architecture to attenuate high-frequency components of the control signal that arise due to the high adaptation gain and named it the $\mathcal{L}_1$ adaptive controller [64].

In recent years, the $\mathcal{L}_1$ adaptive controller has gained significant interest among the robotics and control communities due to its advantages over the conventional MRAC. For example, Bennehar *et al.* proposed an $\mathcal{L}_1$ adaptive controller for a parallel kinematic manipulator with 4-DoF [65]. To enhance the performance of the $\mathcal{L}_1$ adaptive controller, they also added a model-based adaptive feedforward term based on the system's nonlinear dynamics. Where $\mathcal{L}_1$ adaptive controller term was responsible for compensating uncertainties such as frictions, external disturbances, and other residual nonlinearities, whereas, model-based adaptive feedforward term compensated modeling uncertainties.

Maalouf *et al.* proposed an $\mathcal{L}_1$ adaptive controller to control the pitch and depth of an underwater robot for the first time (Figure 3.1a) [66]. The proposed $\mathcal{L}_1$ adaptive control scheme was compared with a well-proven Adaptive Nonlinear State Feedback (ANSF) controller. Three different experimental scenarios were devised namely control in nominal conditions, punctual external disturbance rejection, and robustness towards parameter uncertainty. It was observed that both controllers performed in a similar manner during nominal conditions. In the external disturbance experiments where a push was applied to cause a depth error, the $\mathcal{L}_1$ adaptive controller performed better in stabilizing the depth of the robot, on the other hand, the ANSF controller did well to keep the pitch error small. In last, during the parameter uncertainty experiment, both controllers demonstrated robust performance, however, the ANSF took more time than the $\mathcal{L}_1$ adaptive controller to adapt to these parameter uncertainties.

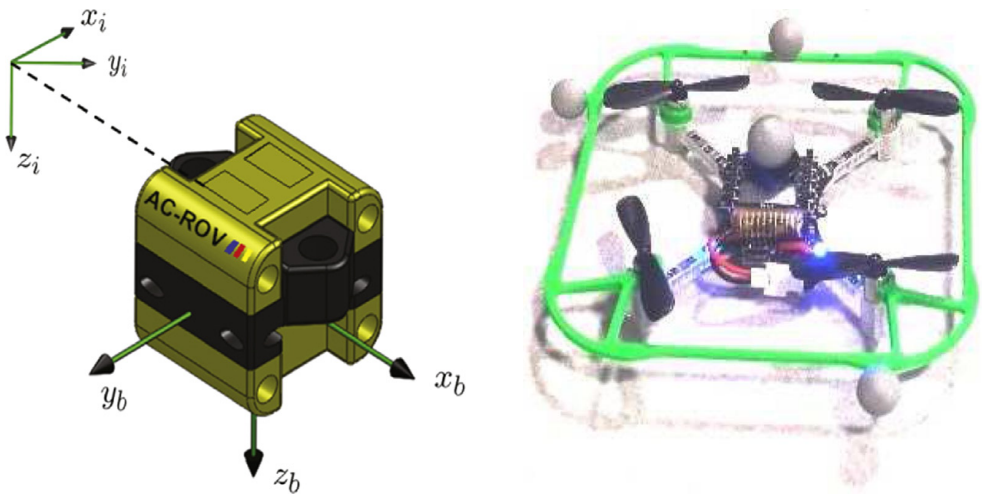
Another study by Xuan *et al.* proposed a cascaded PID and an $\mathcal{L}_1$ adaptive balancing

controller for a three-axis spacecraft simulator [67]. First modeling of the automatic balancing system is performed and a cascaded PID is designed with the help of the Ziegler-Nichols method. Also, an $\mathcal{L}_1$ adaptive controller is developed and implemented in MATLAB's Simulink environment. After nominal case testing, mass and moment of inertia of the system are changed to observe the robustness of the controllers. Simulation results validated the better performance of the $\mathcal{L}_1$ adaptive controller under uncertainties and external disturbances.

Singh *et al.* used an $\mathcal{L}_1$ adaptive autopilot to control the depth and pitch angle of a submarine using only its bow and stern hydroplanes [68]. A detailed mathematical model of the submarine that includes hydrodynamic nonlinearities is provided, later the $\mathcal{L}_1$ adaptive controller is designed. Various simulation tests were conducted to validate and compare the proposed controller against other control methods. These experiments demonstrated the effectiveness of the $\mathcal{L}_1$ adaptive controller under random disturbances, parameter uncertainties, and model nonlinearities.

A multi-rate output-feedback $\mathcal{L}_1$ adaptive controller has been proposed for the security of MIMO cyber-physical systems [69]. A sufficient condition on the sampling time of the digital controller is derived that guarantees stability of the overall closed-loop system. The controller can detect and mitigate attacks on the actuators. The proposed scheme is tested on a small quadrotor, and experimental results demonstrated that the controller successfully recovered the system stability in case of zero-dynamics actuator attacks.

More recently, Lee *et al.* proposed an $\mathcal{L}_1$ adaptive output feedback controller for underactuated multi-input multi-output (MIMO) missile system [70]. The proposed controller worked as an augmentation to an existing baseline three-loop autopilot for the missile systems and compensated for the system nonlinearities to achieve nominal performance. For the first time, Schoellig *et al.* combined an $\mathcal{L}_1$ adaptive controller with an iterative learning control (ILC) to achieve high trajectory tracking for a quadrotor under unknown dynamic disturbances [71]. While the $\mathcal{L}_1$ adaptive part of the control framework achieved the nominal performance under uncertainties and disturbances, the ILC part im-



(a) Underwater ROV [66]

(b) CrazyFlie quadrotor with visual markers [72].

Figure 3.1 Mobile robots with $\mathcal{L}_1$ adaptive controller onboard.

proved the trajectory tracking further by learning from previous iterations over time. The proposed framework was validated on a real quadrotor, and results demonstrated significant performance improvements compared to stand-alone ILC and non-adaptive PD-ILC methods.

3.3 Motivation

It is clear from the previous discussion that both linear and nonlinear motion controllers have been designed to balance and control WIP robots. Commonly used model-based controllers depend on system parameters which may change over time, and other assumptions to simplify the system for modeling purposes. Moreover, linear motion controllers such as LQR are mainly based on linearized models of the system and hence relevant for only a small region around the operating point. We believe an adaptive control mechanism is essential for the wheel-legged system under the assumed circumstances. Accordingly, we propose a new control scheme based on the combination of an LQR feedback controller and an $\mathcal{L}_1$ adaptive controller to compensate for modeling errors, external disturbances, and other eventual uncertainties. To the best of the authors' knowledge, it is the first time that an adaptive control scheme has been proposed for a wheel-legged robot

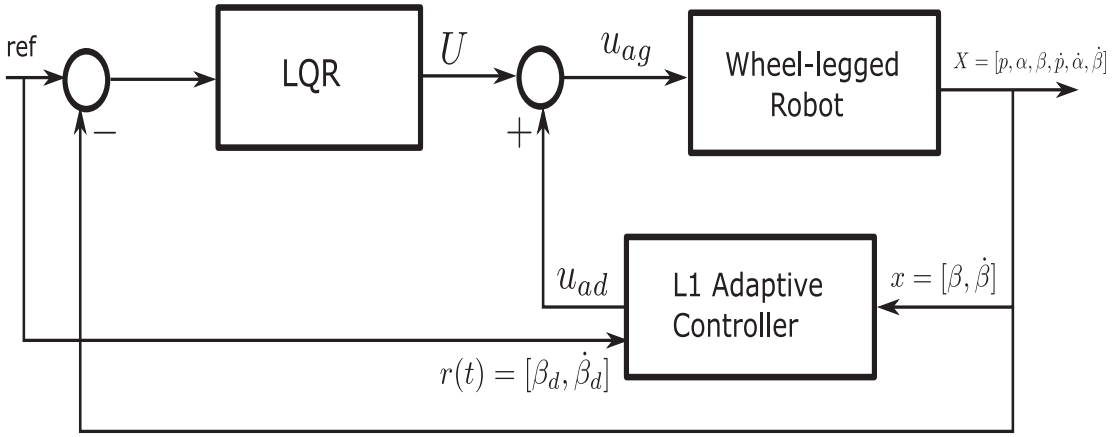


Figure 3.2 Structure of the proposed augmented $\mathcal{L}_1$ adaptive controller.

with experimental validation on a real system.

3.4 Proposed Control Scheme

This section introduces the proposed control method to stabilize the wheel-legged robot that is inherently unstable in nature. The proposed $\mathcal{L}_1$ adaptive augmented control framework combines the LQR controller with the state-of-the-art $\mathcal{L}_1$ adaptive control method, as represented in Figure 3.2. For the sake of completeness, please read the section about LQR that has been presented in detail in chapter 2.

3.4.1 Background on $\mathcal{L}_1$ Adaptive Control

This section describes the design of an adaptive control system for the pitch angle (β) stabilization of the wheel-legged robot. For a safety-critical system like a self-balancing robot, keeping it in the upright position is of utmost importance to avoid any accident, and damage of the robot and its environment. Accordingly, we introduce the method where an $\mathcal{L}_1$ adaptive controller is augmented with an LQR controller in order to keep the robot stable in unforeseen circumstances.

In adaptive control, the estimates of unknown parameters are adjusted in a way to minimize the error between the reference system output and the actual system. In literature, there are two commonly used methods to estimate the parameters in adaptive control,

namely, direct method and indirect method. The implementation of these methods is different, however, both the approaches give the same error dynamics if provided with the same initial conditions. In the indirect method, parameters are estimated implicitly and can be compared with online system identification, on the other hand, the direct method explicitly estimates the controller parameters.

It is well known that the closed-loop system's performance depends on the actuator bandwidth and the dynamics of the plant. However, mechanical systems commonly have slow dynamics and for this reason, high-frequency signals to the plant affect the overall system stability and robustness. To overcome this problem, the $\mathcal{L}_1$ adaptive control uses a low-pass filter (first-order or second-order) to limit its frequency response and meet robustness requirements. The low-pass filter must be designed and tuned so that its frequency response is compatible with the actuator's frequency response.

The $\mathcal{L}_1$ adaptive controller follows the indirect adaptive control architecture. The decoupling between adaptation and robustness ensured by the $\mathcal{L}_1$ adaptive control architecture makes it an ideal adaptive controller for real-time applications. High adaptation gains for achieving fast convergence can be used without introducing a high frequency signal in the control input. The control scheme proposed in this section is based on the $\mathcal{L}_1$ adaptive control theory for systems with time-varying parameters and disturbances along with uncertain system input gain [73].

The $\mathcal{L}_1$ adaptive control consists of an adaptation and a prediction stage as illustrated in Figure 3.3. The adaptation phase is used to predict the unknown and/or time-varying parameters and other uncertainties including external disturbances, whereas the prediction stage is used to get the ideal required performance of the system. Furthermore, a low pass filter is incorporated in the closed-loop to remove high frequencies from the control signal that may occur due to high adaptation gains, as explained earlier.

Let us consider the inverted pendulum model that is extracted from (2.16) in the form

$$\begin{aligned} \dot{x}(t) &= A_p x(t) + B_p(\Omega u_{ad}(t) + \theta^T(t)x(t) + \sigma(t)), \\ y(t) &= C_p x(t) \end{aligned} \tag{3.1}$$

where A_p is the known $\mathbb{R}^{2\times 2}$ matrix that describes the linear dynamics of the inverted pendulum, $x(t) = [\beta\ \dot{\beta}]^T \in \mathbb{R}^2$ is the state vector, $B_p \in \mathbb{R}^2$ and $C_p \in \mathbb{R}^{2\times 2}$ known matrices, $\theta(t) \in \mathbb{R}^2$ vector of time-varying unknown parameters, $\sigma(t) \in \mathbb{R}$ models eventual disturbances and unmodeled dynamics, $\Omega \in \mathbb{R}$ is an unknown positive constant and $u_{ad}(t) \in \mathbb{R}$ is the adaptive control input.

It is assumed that the unknown parameters θ, σ, and Ω meet the following three conditions,

- $\theta(t) \in \Theta$, $|\sigma(t)| \leq \delta_0$, $\forall\ t \geq 0$, where Θ is a known convex compact set and $\delta_0 \in \mathbb{R}^+$ is a known conservative bound of $\sigma(t)$. This assumption is named as the uniform boundedness of unknown parameters.

- $\theta(t)$ and $\sigma(t)$ are continuously differentiable with bounded derivatives i.e. $\left\|\dot{\theta}(t)\right\| \leq d_\theta < \infty,\ |\dot{\sigma}(t)| \leq d_\sigma < \infty,\ \forall\ t \geq 0.$

- Partial knowledge of uncertain system input gain Ω, $\Omega \in \mathbf{\Omega_0} \triangleq [\Omega_{l_0}, \Omega_{u_0}]$, where $0 < \Omega_{l_0} < \Omega_{u_0}$ are known lower and upper bounds on Ω.

State Predictor

To develop a full-state feedback controller so that the system output $y(t)$ tracks the reference signal $r(t)$, we consider a state predictor of the form

$$\begin{aligned} \dot{\hat{x}}(t) &= A_m\hat{x}(t) + B_p(\hat{\Omega}(t)u_{ad}(t) + \hat{\theta}^T x(t) + \hat{\sigma}(t)), \\ \hat{y}(t) &= C_p\hat{x}(t) \end{aligned} \tag{3.2}$$

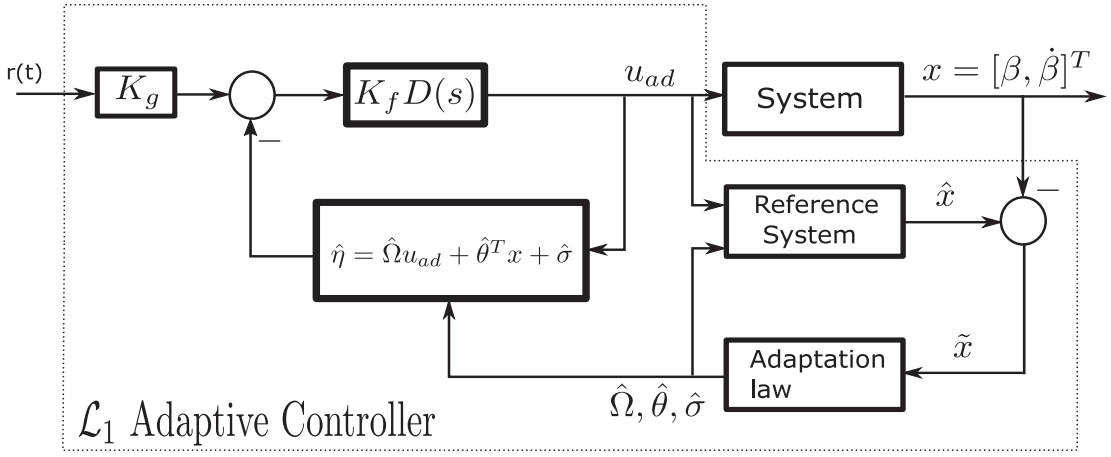


Figure 3.3 Block diagram of $\mathcal{L}_1$ Adaptive Control.

here $A_m = A_p - B_p K_m \in \mathbb{R}^{2\times 2}$ is a user defined Hurwitz matrix, whereas $\hat{\theta}$, $\hat{\sigma}$, and $\hat{\Omega}$ are the estimates of θ, σ, and Ω, respectively. In this research, we chose

$$A_m = \begin{bmatrix} 0 & 1 \\ -150 & -430 \end{bmatrix}, \tag{3.3}$$

and B_p is,

$$B_p = \begin{bmatrix} 0 \\ -0.4 \end{bmatrix}$$

Adaptation Law

The estimations of the parameters are governed by the following projection-based adaptive laws

$$\begin{aligned} \dot{\hat{\theta}}(t) &= Proj(\hat{\theta}(t), -\Gamma_\theta \tilde{x}^T(t) P_a B_p x(t)),\ \hat{\theta}(0) = \hat{\theta}_0, \\ \dot{\hat{\sigma}}(t) &= Proj(\hat{\sigma}(t), -\Gamma_\sigma \tilde{x}^T(t) P_a B_p),\ \hat{\sigma}(0) = \hat{\sigma}_0, \\ \dot{\hat{\Omega}}(t) &= Proj(\hat{\Omega}(t), -\Gamma_\Omega \tilde{x}^T(t) P_a B_p u_{ad}(t)),\ \hat{\Omega}(0) = 1, \end{aligned} \tag{3.4}$$

where $\Gamma_\Omega > 0$, $\Gamma_\sigma > 0$, and $\Gamma_\theta > 0$ are the adaptation gains, $\tilde{x}(t) = \hat{x}(t) - x(t)$ is the prediction error, and $P_a = P_a^T > 0$ is the solution of Lyapunov equation $A_m^T P_a + P_a A_m = -Q_a$ for an arbitrary symmetric matrix $Q_a = Q_a^T > 0$. The above given projection-

based adaptive laws guarantee that the adaptation remains within the feasible region of parameter space. Using (3.3) and

$$Q_a = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

we obtain the solution of above mentioned Lyapunov equation as,

$$P_a = \begin{bmatrix} 1.6089 & 0.0033 \\ 0.0033 & 0.0012 \end{bmatrix}.$$

Furthermore for fast convergence of the estimation parameters, the adaptation gains are selected as $\Gamma_\theta = 5000$, $\Gamma_\sigma = 10000$, and $\Gamma_\Omega = 10000$.

Projection Operator

A projection operator is used for updating the parameters $\hat{\theta}, \hat{\sigma}$, and $\hat{\Omega}$ smoothly and confining them within the required set [68]. This helps in faster adaptation while guaranteeing the closed-loop stability of the system at the same time. Furthermore, the projection operator $Proj(z, \phi)$ ensures that z stays locally Lipchitz continuous even if ϕ is piecewise continuous.

The algorithm of the projection operator $Proj(z, \phi)$, used in (3.4) for a parameter z, is described as follows,

---

Algorithm : Projection Operator

Inputs : ϵ, z, ϕ, z_{max}, z_{min}
1 : compute $f_d = (z_{max} - z_{min})^2$;
2 : compute $f_z = \frac{-4*(z_{min}-z)*(z_{max}-z)}{\epsilon * f_d}$;
3 : compute $f_{\dot{z}} = \frac{4*(z_{min}+z_{max}-2*z)}{\epsilon * f_d}$;
4 : define $output = \phi$;
5 : ***if*** $(f_z <= 0 \; and \; f_{\dot{z}} * \phi < 0)$ ***then***
$output = \phi * (f_z + 1)$;
6 : ***return*** output;

---

here, $0 < \epsilon < 1$ is a constant that sets the steepness of the curve of the projection at the

maximum values. The parameter ϵ should be chosen in a way that the parabolic curve of the projection operator is steep enough to capture the highest expected error [74]. z_{max} and z_{min} are respectively the maximum and minimum values delimiting the admissible range of the parameter z.

Adaptive Control Law

According to the block diagram of Figure 3.3, the adaptive control input u_{ad} for the inverted pendulum system (3.1) is given in Laplace domain as follows;

$$u_{ad}(s) = -K_f D(s)\big(\hat{\eta}(s) - K_g r(s)\big), \tag{3.5}$$

where K_g is the feedforward gain, $r(s) = [\beta_d, \dot{\beta}_d]^T$ is the reference signal, $K_f > 0$ is the feedback gain, and $D(s)$ is a user defined low-pass filter that has a strictly proper transfer function such that

$$C(s) = \frac{\Omega K_f D(s)}{1 + \Omega K_f D(s)}, \tag{3.6}$$

is strictly proper stable transfer function with DC gain $C(0) = 1$. Furthermore,

$$\hat{\eta}(t) = \hat{\Omega} u_{ad}(t) + \hat{\theta}^T(t) x(t) + \hat{\sigma}(t).$$

To ensure the stability of the resulting closed-loop system, the design of the feedback gain K_f and the low-pass filter $D(s)$ should satisfy the following $\mathcal{L}_1$-norm condition

$$\|G(s)\|_{\mathcal{L}_1} L < 1, \tag{3.7}$$

where $G(s) = H(s)\big(1 - C(s)\big)$, $H(s) = \big(s\mathbb{I} - A_m\big)^{-1} B_p$, and $L = \max_{\theta \in \Theta} \|\theta\|_1$, here Θ is a known convex compact set. In this study, we chose $K_g = -375$, $K_f = 1$ and D(s) as a

first-order filter,

$$D(s) = \frac{50}{s + 50}$$

The particular structure of the adaptive controller that is given in (3.5) decouples the control loop from the estimation loop of the adaptive controller and hence allowing for arbitrarily high adaptation gains without compromising the closed-loop robustness. The reader is encouraged to refer to [73] for detailed proofs of stability and performance analysis.

The feedback gain K_f is very influential in the whole $\mathcal{L}_1$ control architecture and must be set low enough to block high-frequency signals to the actuators, on the other hand, high enough to meet the $\mathcal{L}_1$ norm condition given in (3.7).

Another real challenge of the $\mathcal{L}_1$ adaptive controller is to implement its low-pass filter D(s) in discrete time on a real micro-controller. It is well noted that failures of early adaptive controllers happened because of a poor understanding of closed-loop system robustness. Furthermore, identification of the parameters needs to be faster than the plant variation time scale [75]. The control loop for our robot runs at 500 Hz, which is much faster than the robot dynamics. We use a digital Bi-Quad filter to implement the low-pass filter in discrete time with sampling time of $T_s = 0.002$ secs.

Bi-Quad Filter

A digital bi-quad filter is a recursive second-order filter that provides an accurate and straightforward way to implement a low-pass filter. It uses a finite impulse response (FIR) and an infinite impulse response (IIR) and thus has four memory blocks in total. It is well known that higher order IIR filters are very sensitive to the quantization errors that can easily lead to instability. In practice, this problem is resolved by implementing higher order filters as a series of cascaded bi-quad filter blocks. To ensure the stability of the overall filter, the two poles of the bi-quad filter must be inside the unit circle in the Z-domain.

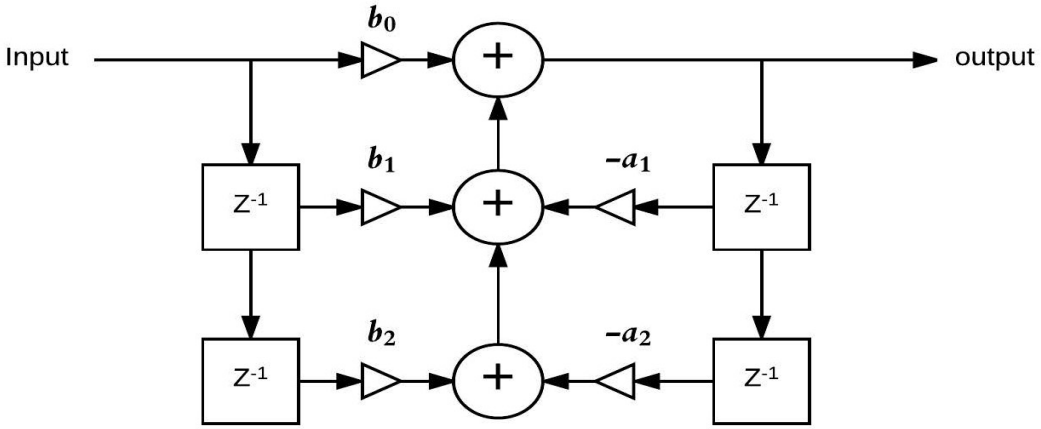


Figure 3.4 Architecture of Bi-Quad filter in Direct Form I.

The output of the bi-quad filter shown in Figure 3.4 is

$$y = b_0x + b_1xz^{-1} + b_2xz^{-2} - a_1yz^{-1} - a_2yz^{-2} \tag{3.8}$$

here x is the filter input, y is the filter output, $b_0,\ b_1,\ b_2$ are the coefficients determine the position of the zeros, $a_1,\ a_2$ are the coefficients determine the position of the poles, and z is the Z-domain variable. Eq. (3.8) results into a discrete form normalized transfer function as,

$$H(z) = \frac{b_0 + b_1z^{-1} + b_2z^{-2}}{1 + a_1z^{-1} + a_2z^{-2}} \tag{3.9}$$

This form of the bi-quad filter is called as direct form I and is the most straightforward architecture. A continuous-time model or transfer function of the filter is converted into the Z-domain by bilinear transformation in form of (3.9) first, and then implemented as a bi-quad filter. It is clear from (3.8) that this form of the filter uses four memory blocks, two for the input and two for the output. However, by rearranging a few terms we can reduce the memory blocks to two without losing any information. The new form is called direct form II and has two equations,

$$y = b_0w + b_1wz^{-1} + b_2wz^{-2} \tag{3.10}$$

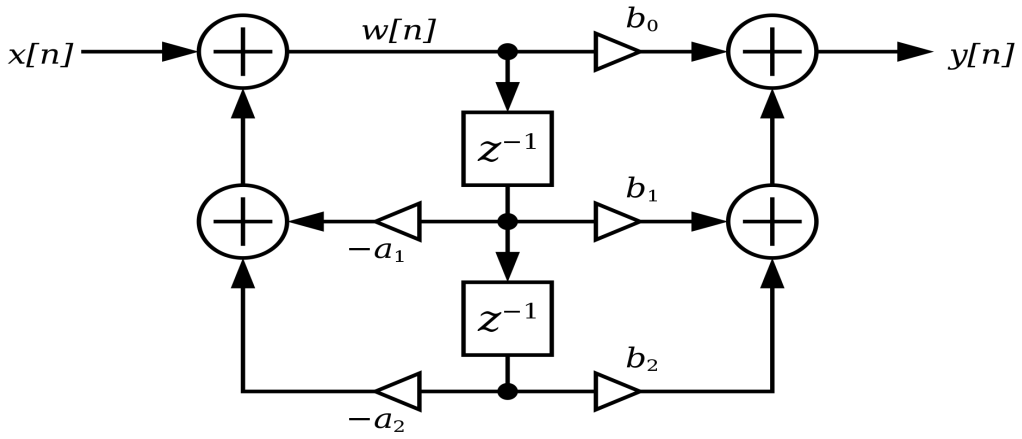


Figure 3.5 Architecture of Bi-Quad filter in Direct Form II.

where

$$w = x - a_1 w z^{-1} - a_2 w z^{-2}$$

The block diagram of the direct form II is given in Figure 3.5. In this research, we simply used MATLAB to do the continuous to the discrete time transformation of D(s) and an open-source C++ library of bi-quad filters by Tom Lankhorst that uses the direct form II at the back end [76].

3.5 Simulation Results

This section provides an overview of the simulation setup used in this study as well as the corresponding obtained results. The wheel-legged robot Igor was defined in the Unified Robotic Description Format (URDF) and simulated in Gazebo simulator using the Robotic Operatic System (ROS). We used C++ programming language to implement the proposed motion control algorithm for better portability and efficiency. The control loop runs at a frequency of 500Hz. The nominal parameters of the system are summarized in TABLE 2.1. To validate the proposed augmented adaptive control scheme qualitatively as well as quantitatively, we conducted two different tests and results are shown in Figure 3.6. The main motivation behind these scenarios is to compare the input torques of both

controllers, and the pitch angle (β) and pitch rate ($\dot{\beta}$) subject to external disturbances and model uncertainties.

In the first scenario, we used the nominal parameters of the robot and compare the augmented control scheme against the model-based LQR. The obtained results from this simulation are depicted in Figure 3.6a. We applied a linear force of $20N$ on the robot body at $t = 10s$ for a duration of $1s$, and hence producing a significant linear acceleration. As a result, the robot is pushed back from its reference position but slowly returned back while maintaining its balance. The given plots clearly indicate that the augmented controller keeps the pitch angle (β) and rate ($\dot{\beta}$) smaller than those of the LQR. More specifically, the error in the pitch angle (β) over this $20s$ period is about 50% less compared to the LQR controller. It is further noted that the settling time of the pitch angle for the proposed controller is about $2.5s$ against $6s$ for LQR. We observe that the faster response of the augmented controller is due to the fast adaptation of the $\mathcal{L}_1$ adaptive controller to external disturbances.

In the second scenario, to introduce a modeling uncertainty we propose to change the value of the viscous friction coefficient of the wheel actuators from its nominal value $c_r = c_l = 0.17$ to $c_r = c_l = 0.34$ (i.e. $+100\%$). The change in the friction coefficients also changes the dynamical behavior of the system, and for this reason model-based controllers are expected to perform poorly under such circumstances. Furthermore, we know that determining these friction coefficients in the real world is near impossible as they change over time due to abrasion and other factors. In case of the model-based LQR as shown in Figure 3.6b, it is evident that the impact of inaccurate friction coefficients can be highly risky for a self-balancing robot as it could not recover from the translational push of $20N$. On the other hand, the proposed augmented controller successfully kept the robot around its vertical upright position. It becomes clear from the estimated parameters graph in Figure 3.6 that the unknown input gain Ω and the unknown parameter σ have greater influence on the $\mathcal{L}_1$ adaptive control input.

TABLE 3.1 summarizes the quantitative difference between the LQR and the proposed

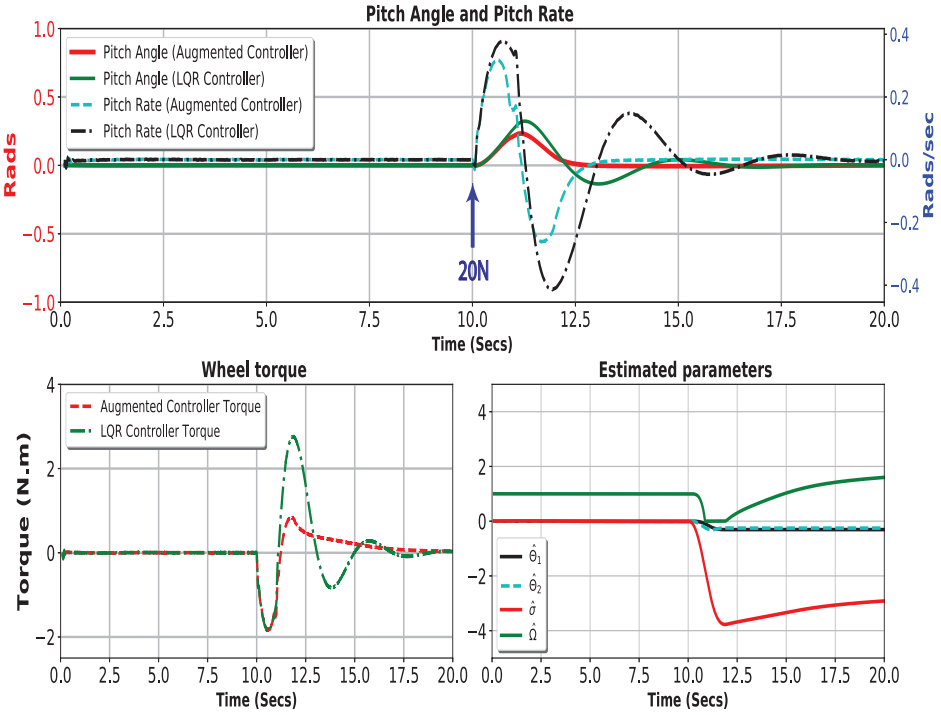
		Simulations		
		LQR	**Augmented Controller**	**Percent Change**
Nominal case	$\int\|\beta\|dt$	0.6157	0.3070	50.1%
	$\int\|\mathcal{T}\|dt$	6.3441	3.6355	42.7%
Uncertain case	$\int\|\beta\|dt$	11.5773	0.5006	95.7%
	$\int\|\mathcal{T}\|dt$	18.0505	16.2275	10.0%
		Real-time Experiments		
		LQR	**Augmented Controller**	**Percent Change**
Tile floor	$\int\|\beta\|dt$	0.6624	0.2700	59.2%
	$\int\|\mathcal{T}\|dt$	13.5188	8.8440	34.6%
Carpet floor	$\int\|\beta\|dt$	0.5160	0.3283	36.3%
	$\int\|\mathcal{T}\|dt$	10.9857	10.9037	0.75%

Table 3.1: We use area under the curve of the pitch angle and the wheel torque to quantify the performance of the given controllers. The smaller these values are the better the corresponding controller performs in terms of settling time, overshoot, and energy consumption. The percent change indicates the percentage reduction of these areas in the case of augmented controller w.r.t the LQR.

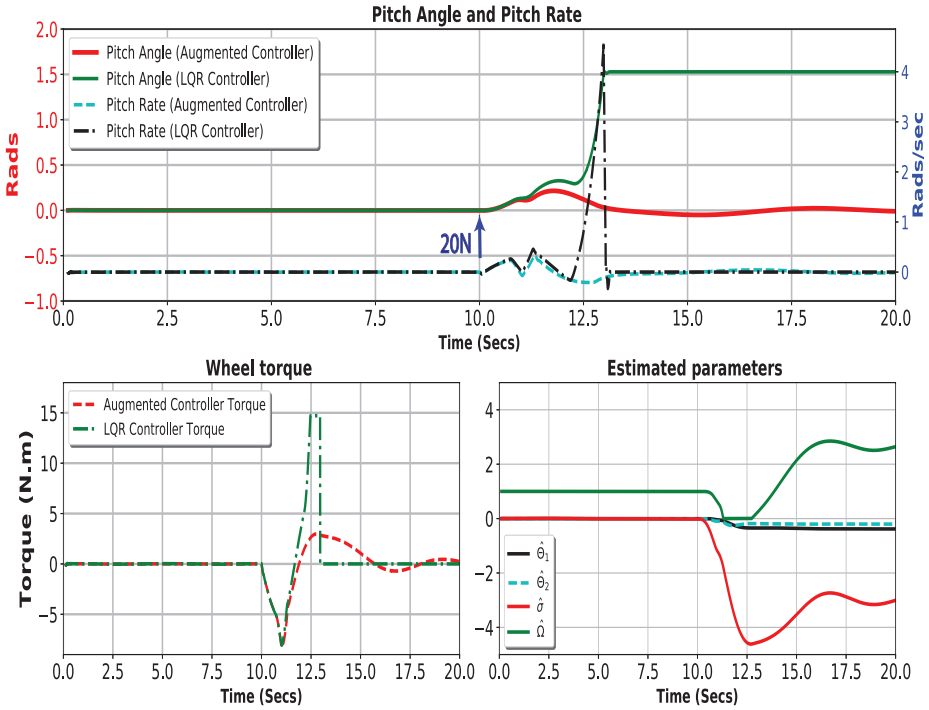
augmented controller in different settings. By integrating the torques over time, which is also known as angular impulse, we found that in the nominal case, it is 6.3441 N.m.s against 3.6355 N.m.s for the LQR and the proposed augmented controller, respectively. It is clear that the augmented controller uses upto 42.7% less torque and hence less energy than the LQR in keeping the robot balanced in the nominal case. Besides it is worth to note that the proposed augmented control scheme can endure a linear push of up to $30N$ i.e. 50% more force than the maximum of $20N$ for the LQR.

3.6 Real-Time Experiments And Results

For the real-time control of the Igor robot, we used ROS with C++ to run the motion control algorithms. The control loop runs on an intel core i7 microprocessor at a frequency of 500Hz and the torque commands are transmitted to the robot actuators through a wifi connection. Besides, for the robot localization we used an extended kalman filter (EKF) to filter and fuse the data from multiple IMUs and the wheel odometry.



(a) Nominal case.



(b) Uncertain case, $\Delta c_r = \Delta c_l = +100\%$.

Figure 3.6 Validation in simulation: Temporal evolution of pitch angle, pitch rate, wheel torque, and the estimated parameters of the augmented $\mathcal{L}_1$ adaptive controller during a push of 20N force.

We performed a couple of experiments on the real Igor robot to compare the stability performance and energy efficiency of the two controllers in different operating conditions. The controllers have the regulation task to keep the robot balanced $\beta \approx 0$ in the presence of the external pushes. Since the real robot lacks a force sensor, we applied manual pushes repeated three times to take the average result. The first experiment is performed on a normal tiled floor with less friction between the floor and the wheels of the robot, while the second experiment is performed on a carpet floor with a better friction between the robot wheels and the ground. Also a lidar sensor of mass 1Kg is attached to the bottom of the Igor body for emulating modeling uncertainties. This extra mass is not included in the dynamic model of the robot and the LQR controller. The obtained results of these real-time experiments are displayed in Figure 3.7 and further summarized quantitatively in TABLE 3.1.

Figure 3.7a provides the evolution of the pitch angle β, pitch rate $\dot{\beta}$, wheel torque τ, and the estimated parameters of the augmented $\mathcal{L}_1$ adaptive controller versus time for the tiled floor test. It is clear that the pitch angle remains significantly smaller (about 59%) for the proposed augmented controller case compared to the LQR case. Furthermore, this enhancement in the robot stability in case of the proposed controller also comes with the benefit of a reduced wheel torque of about 34.6%. However, it is worth to note that even with model uncertainties, the LQR feedback controller successfully achieves the nominal performance by balancing the robot.

For comparison, we repeated the same tests on a carpet surface in the second experiment as shown in Figure 3.7b. It is noted that with similar pushes, the deviation of the pitch angle from the given reference point stays smaller for the proposed augmented controller than for the LQR. However, this difference is reduced from 59.2% to 36.3% in the case of carpet floor. Furthermore, due to high friction between the carpet floor and the robot wheels, both controllers give the similar torque profiles to keep the robot balanced against the applied external pushes.

3.7 Discussion

To demonstrate the effectiveness of the $\mathcal{L}_1$ adaptive controller, various experiments are performed both in simulations and on the real robot. The overall reduction in the pitch angle error of the robot, which is more profound in the tiled floor case proves the advantage of the augmentation of an $\mathcal{L}_1$ adaptive controller with the baseline LQR. However, there remain many other situations that are not taken into account in this chapter and can be the subject of future studies. For example, the effect of time delay can be of catastrophic nature for safety-critical systems and therefore must be part of closed-loop stability analyses.

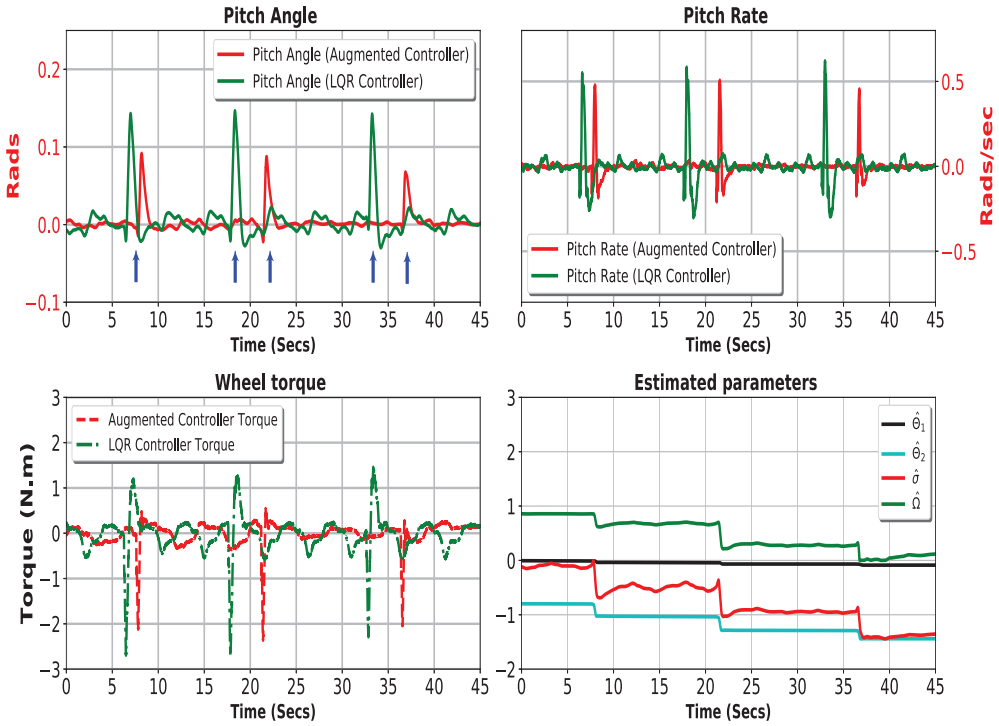
In this study, we only applied $\mathcal{L}_1$ adaptive controller to stabilize the pitch angle, whereas the LQR is used to control all the 3 DoFs of the robot. The main reason is that $\mathcal{L}_1$ adaptive control theory has not been fully developed for controlling the underactuated MIMO (multi-input-multi-output) systems. We know from the equations of motion of the robot that it is an underactuated system with its pitch angle (β) and translational position highly coupled with each other. It is expected that an $\mathcal{L}_1$ adaptive controller design to control all the three DoFs of the robot will handle the coupling of states implicitly and hence improving the overall robustness of the system.

Lastly, choosing an optimal low-pass filter systematically for the $\mathcal{L}_1$ adaptive controller is still an open research question [73]. It is clear from (3.7) that the closed-loop system stability depends on the choice of C(s), which decides the trade-off between robustness and performance. The low-pass filter design of $\mathcal{L}_1$ adaptive controller can be formed as an $\mathcal{L}_1$-norm optimization problem. Kevin *et al.* designed an optimal low-pass filter for the $\mathcal{L}_1$ adaptive controller by forming the trade-off between the performance improvement and the maximization of time-delay margin as a convex optimization problem [77]. A linear matrix inequality approach was used to optimize one objective function while keeping the desired bound on the other. In another study, Naira *et al.* proposed the D-K iteration method to find an optimal filter for the $\mathcal{L}_1$ adaptive controller [78]. More recently, quadratic programming is used to solve the optimal filter problem as a convex

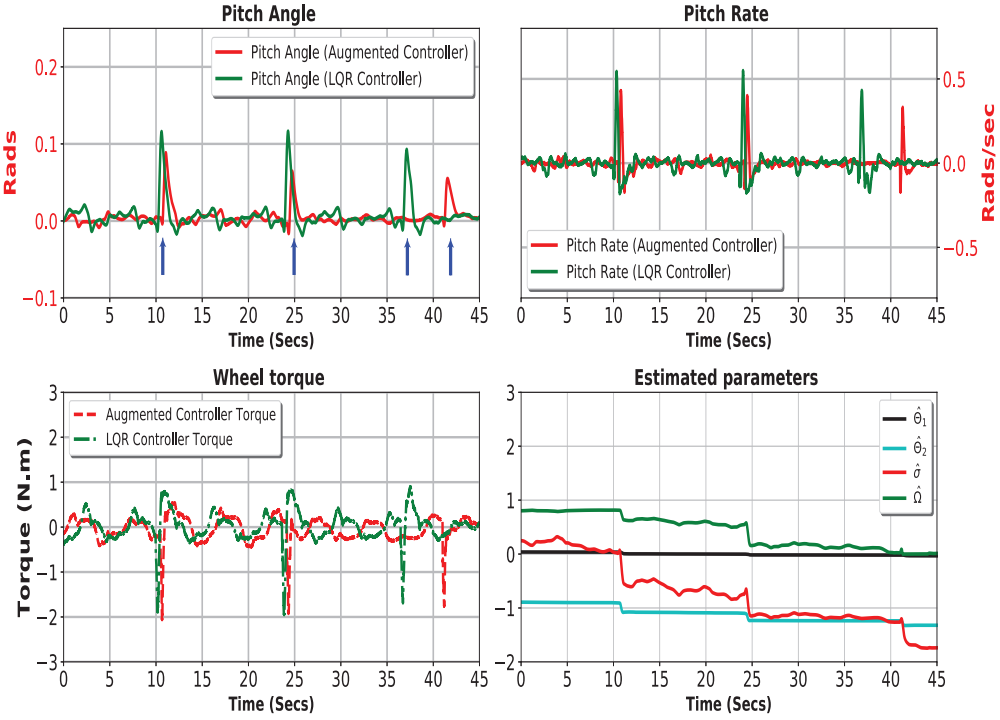
optimization [72]. The trade-off between the performance and the robustness is solved by optimizing a mixed $\mathcal{L}_1/\mathcal{H}_2$-norm problem. The optimal filter is obtained in discrete-time first and then converted into continuous-time domain. The obtained $\mathcal{L}_1$ adaptive controller with optimized filter is implemented and tested on a small quadrotor (Figure 3.1b) for precision trajectory tracking. However, one drawback of this method is that because of small time-steps in the discretization, high-order filters are obtained. Lee *et al.* proposed the filter design based on the $\mathcal{H}_\infty$ optimization framework by taking the stability specifications of the system in frequency domain [79].

3.8 Conclusion

This chapter shows the development of a linear quadratic regulator augmented with an $\mathcal{L}_1$ adaptive controller to stabilize a wheel-legged robot. First, a state-of-the-art $\mathcal{L}_1$ adaptive controller is derived in detail for stabilizing the pitch angle β and pitch rate $\dot{\beta}$ of the wheel-legged robot. Furthermore, an algorithm to keep the estimation parameters of the adaptive controller within the required set is also provided. It is shown through different simulations and real-time experiments that the proposed augmented controller was successfully able to compensate for the model uncertainties and external disturbances and clearly outperforms the model-based LQR controller.



(a) Tile floor case.



(b) Carpet floor case.

Figure 3.7 Validation in real-time experiments: Temporal evolution of pitch angle, pitch rate, wheel torque, and the estimated parameters of the augmented $\mathcal{L}_1$ adaptive controller subject to external pushes.

Chapter 4

Balance Stability Augmentation for Wheel-legged Biped Robot through Arm Acceleration Control

4.1 Outline

A self-balancing wheel-legged robot provides higher maneuverability and mobility than legged biped robots. For this reason, wheel-legged systems have attracted enormous interest from academia and commercial sectors in recent years. Most of the past works in this field including ours mainly focused on lower body stabilization. Motivated by the human ability to maintain balance in laborious activities by articulating the arm actively, we explore and analyze the active arm control on top of the wheel-legged system to assist in its balance recovery during external pushes and disturbances. This chapter presents a control framework to improve the stability and robustness of an underactuated self-balancing wheel-legged robot using its upper limb arm. Furthermore, we use the centroidal moment pivot (CMP) as a key metric to quantitatively evaluate the effect of the active arm usage on the balance stability improvement of the robot in the ROS-Gazebo environment. The difference from the case of nonusage of arm is verified to clarify the impact of the active arm quantitatively. This concept would lead to the wheel-legged biped robot with an

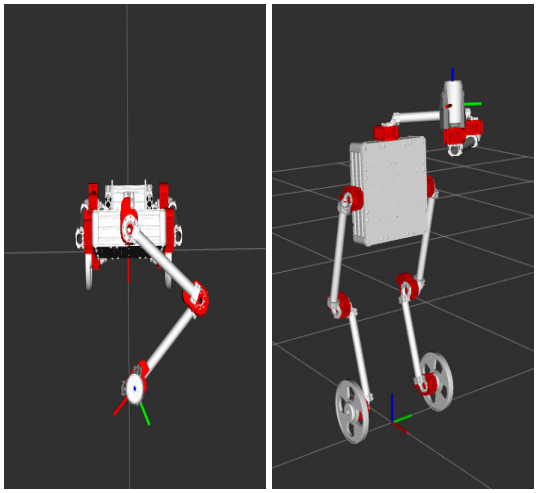
active arm for dual purposes, one is for carrying objects, another is for increasing the balance stability. This point is important for future application in a real-world environment with human-robot interactions.

4.2 Motivation

In the previous chapters, we discussed the importance of robust stability for the self-balancing wheel-legged robots and proposed different controllers including a state-of-the-art adaptive control framework. Although these motion controllers are capable enough to stabilize the robot in different scenarios, we can further improve the overall stability of the robot by the use of its arm manipulator that is shown in Figure 4.1.

Recent studies suggest that humans extensively use upper-body movements in addition to the ankle and hip joints to dynamically balance during challenging tasks [80]. It is revealed that at the time of performing difficult balancing tasks, humans intuitively use their upper body more than their lower body to stay balanced. Similarly, many vertebrate animals use tails to control their equilibrium, for example, a cat swings its tail to change its center of gravity [81]. Inspired from such studies, Minamizawa *et al.* from Keio University designed a robotic tail for assistive rehabilitation applications [82]. This anthropomorphic tail continuously estimates the user's center of gravity and provides active stability by swinging the tail. Furthermore, it has a weight adjustable design so that it can be used by people of different weights.

Balancing of WIP robots is of foremost importance for the safety of the agents around the robot and smooth human-robot interaction. It is clear from the previous discussions that most of the past work in this field focused only on the lower body stabilization of WIP systems. In this study, we propose a novel balance recovery method for a wheel-legged robot with active usage of an arm manipulator on top of the lower limb. Furthermore, we evaluate the real effect of the upper limb motion of the robot on its stability during external disturbances. For this reason, the centroidal moment pivot (CMP), a key criterion



(a) Top View. **(b)** Perspective View.

Figure 4.1 Wheel-legged robot with manipulator arm.

for measuring the stability of humans (also known as the zero rate of angular momentum) is applied to this study [83][84].

4.3 Arm Manipulator Modeling and Control

The manipulator arm is decoupled from the rest of the robot body and modeled separately from the lower body of the robot. We used the resolved acceleration control method to manipulate the end-effector position and acceleration in the Cartesian plane [85].

A schematic of the manipulator is shown in Figure 4.2. For simplification purposes, we assumed that the masses of the links act as point masses and are located at the end of each link. Furthermore, the CoM of each link was located at the far end of the corresponding link. In addition, friction was considered negligible. The rigid-body dynamics equation for a two-link arm can easily be derived using the Lagrangian formulation and is generally presented in the following form:

$$\tau_m = M(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \boldsymbol{h}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + \boldsymbol{g}(\boldsymbol{\theta}), \tag{4.1}$$

where τ_m is the vector of the joint torques τ_1 and τ_2, $\boldsymbol{\theta}$ is the vector of the joint angles θ_1

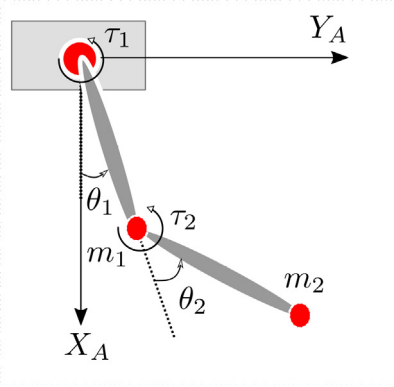


Figure 4.2 Schematic of the arm manipulator on top of the lower limb of the wheel-legged robot.

and θ_2, $\boldsymbol{h}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ is the vector of the Coriolis and centrifugal forces, $\boldsymbol{g}(\boldsymbol{\theta})$ is the vector of the gravitational forces, and $M(\boldsymbol{\theta})$ is the inertia matrix that is symmetric, positive definite, and always invertible.

$$M(\boldsymbol{\theta}) = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix}$$

where

$$\begin{aligned} M_{11} &= I_1 + I_2 + m_1 l_{g1}^2 \\ &\quad + m_2\big(l_1^2 + l_{g2}^2 + 2l_1 l_{g2} + 2l_1 l_{g2}\cos(\theta_2)\big) \\ M_{12} &= M_{21} = I_2 + m_2\big(l_{g2}^2 + l_1 l_{g2}\cos(\theta_2)\big) \\ M_{22} &= I_2 + m_2 l_{g2}^2 \end{aligned}$$

The manipulator works in the horizontal plane, so the gravity term is exactly equal to zero and (4.1) reduces to:

$$\tau_m = M(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \boldsymbol{h}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}), \tag{4.2}$$

The vector of the Coriolis and centrifugal forces is:

$$\boldsymbol{h}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = \begin{bmatrix} -m_2 l_1 l_{g2}\dot{\theta}_2\big(2\dot{\theta}_1 + \dot{\theta}_2\big)\sin(\theta_2) \\ m_2 l_1 l_{g2}\dot{\theta}_1^2 \sin(\theta_2) \end{bmatrix}$$

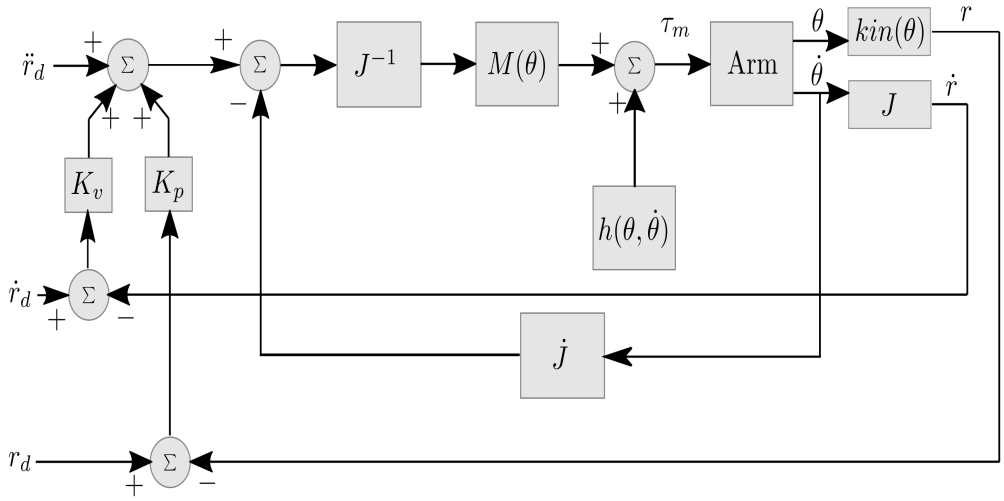


Figure 4.3 Arm Manipulator control scheme.

4.3.1 Resolved Acceleration Control

To control the end-effector position, velocity, and acceleration in the Cartesian plane, we must transform (4.2) from joint-space variables to Cartesian space variables. This is achieved by using the Jacobian matrix that relates the joint velocities of the manipulator to the Cartesian velocities of the end-effector, as follows:

$$\dot{\boldsymbol{r}} = \begin{bmatrix} {}_A\dot{x}_{ee} \\ {}_A\dot{y}_{ee} \end{bmatrix} = J \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix}$$

where the closed-form Jacobian matrix is denoted as:

$$J = \begin{bmatrix} -l_1 \sin(\theta_1) - l_2 \sin(\theta_1 + \theta_2) & -l_2 \sin(\theta_1 + \theta_2) \\ l_1 \cos(\theta_1) + l_2 \cos(\theta_1 + \theta_2) & l_2 \cos(\theta_1 + \theta_2) \end{bmatrix}$$

Equation (4.2) then becomes:

$$\tau_m = M(\boldsymbol{\theta}) J^{-1} \big(\ddot{\boldsymbol{r}} - \dot{J}\dot{\boldsymbol{\theta}} \big) + \boldsymbol{h}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \tag{4.3}$$

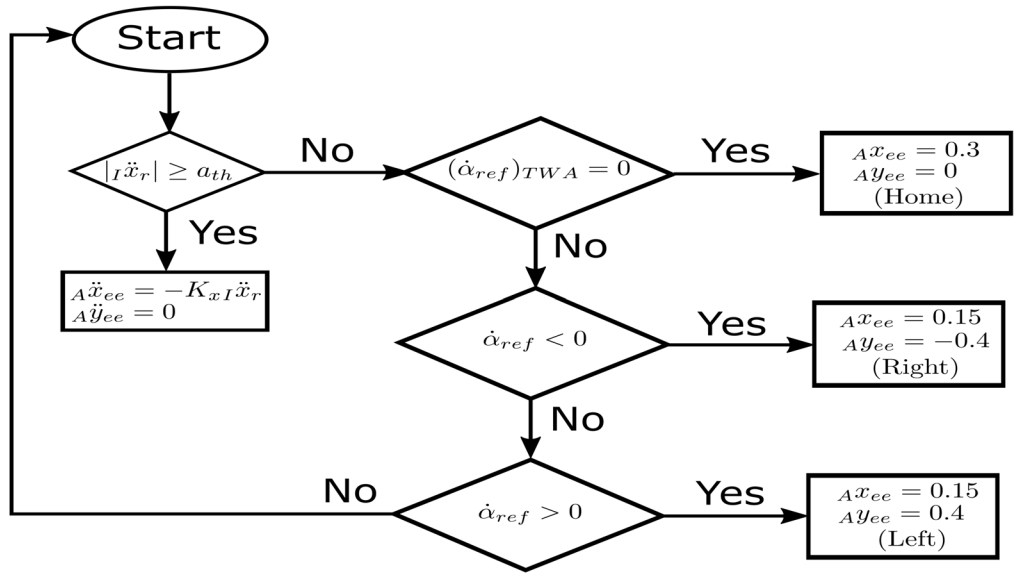


Figure 4.4 State machine flowchart for active arm usage.

Equation (4.3) can be used to design a simple controller that reduces the error between the desired and actual position, velocity, and acceleration of the end-effector. Here, we choose a simple proportional derivative (PD) control scheme that results in:

$$\begin{aligned}\mathcal{T}_m = M(\boldsymbol{\theta})J^{-1}\big[\ddot{\boldsymbol{r}}_d + \boldsymbol{K_v}(\dot{\boldsymbol{r}}_d - \dot{\boldsymbol{r}}) + \boldsymbol{K_p}(\boldsymbol{r}_d - \boldsymbol{r}) - \dot{J}\dot{\boldsymbol{\theta}}\big] \\ + \boldsymbol{h}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\end{aligned} \tag{4.4}$$

where $\ddot{\boldsymbol{r}}_d$ is the vector of desired accelerations in the Cartesian plane, $\dot{\boldsymbol{r}}_d$ is the vector of the desired velocities, $\boldsymbol{r}_d$ is the vector of the desired positions, and $\boldsymbol{K_p}$ and $\boldsymbol{K_v}$ are the PD controller gains for the position and velocity, respectively. Suitable PD controller gains were obtained through trial and error in the simulations. Figure 4.3 presents the overall controller scheme for the manipulator arm of the robot.

4.4 Finite-State Machine for Active Arm Usage

In simple words, a finite-state machine (FSM) is a computational model to execute sequential logic. Due to their flexibility, low computation requirements, and simplicity, FSMs are commonly used in games, traffic light systems, and other artificial intelligence

applications.

To control the end-effector acceleration and position to cancel out or mitigate the external disturbances on the robot, we design a finite-state machine, as shown in Figure 4.4. First, we check the translational acceleration of the robot, and if it is greater than a certain threshold a_{th}, then the end-effector accelerates in the opposite direction of the robot's translational acceleration with some proportional gain K_x.

Second, when the robot traverses a curve, it rotates the end-effector toward the center of the curvature to deal with outward inertial forces such as centrifugal force. However, to distinguish whether the robot is traversing a curve or it is merely a small rotation, we take the time-weighted average of the reference yaw velocity $(\dot{\alpha}_{ref})_{TWA}$ over the period of a half-second. If $(\dot{\alpha}_{ref})_{TWA}$ is zero, which indicates that the robot is not traversing a curve, then the end-effector remains at its home position.

In this manner, if the robot rotates in the clockwise direction, the end-effector moves toward the right side of the robot, and when the robot rotates in the counterclockwise direction, the end-effector shifts toward the left side.

4.5 Centroidal Moment Pivot

Goswami and Herr independently introduced the centroidal moment pivot (CMP) as a useful criterion for the analysis and posture stability measure of biped robots [86][87]. CMP is derived from the rate of change of the angular momentum $\dot{\boldsymbol{H}}_{\boldsymbol{G}}$, which is applicable only when a robot is walking on a planar surface. The loss of balance in biped robots simply means that $\dot{\boldsymbol{H}}_{\boldsymbol{G}}$ is non-zero. We used this key stability criterion for the first time for a biped wheel-legged robot.

CMP is defined as the robot's CoM projection point on the ground along the resultant ground reaction force [88][89]. In other words, it is the point where the reaction force needs to be applied to result in zero $\dot{\boldsymbol{H}}_{\boldsymbol{G}}$. Thus, the amount of CMP deviation from the body center can indicate the intensity of the whole-body balance instability, and hence

it is used to evaluate the resultant balancing stability in this study. CMP ($\boldsymbol{p_f}$) is derived from the position of the CoM of the robot, ground reaction force vector, and a normal vector, as follows:

$$\boldsymbol{p_f} = \frac{(\boldsymbol{p_0} - \boldsymbol{c}) \cdot \boldsymbol{n}}{\boldsymbol{f} \cdot \boldsymbol{n}} \, \boldsymbol{f} + \boldsymbol{c} \tag{4.5}$$

where **c** is the position of the CoM, **n** is the normal vector to the ground plane, **f** is the vector of the ground reaction forces, and $\boldsymbol{p_0}$ is the ground level.

There are no sensors available to measure the ground reaction force directly from the robot, so we approximately calculate $\boldsymbol{f}$ as follows:

$$\boldsymbol{f} = m(\ddot{\boldsymbol{c}} - \boldsymbol{g}) \tag{4.6}$$

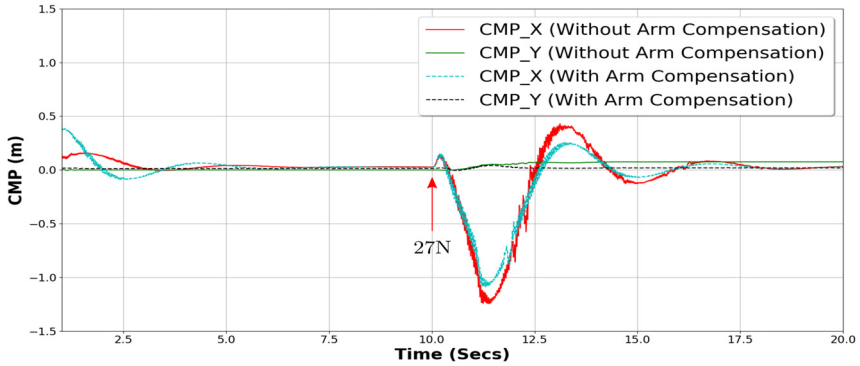
where $\ddot{\boldsymbol{c}}$ is the acceleration of the CoM, m is the total mass of the robot, and $\boldsymbol{g}$ is the vector of gravitational acceleration.
For a planar surface, (4.5) can be reduced to:

$$\boldsymbol{p_f} = \frac{p_0 - c_z}{f_z} \, \boldsymbol{f} + \boldsymbol{c} \tag{4.7}$$

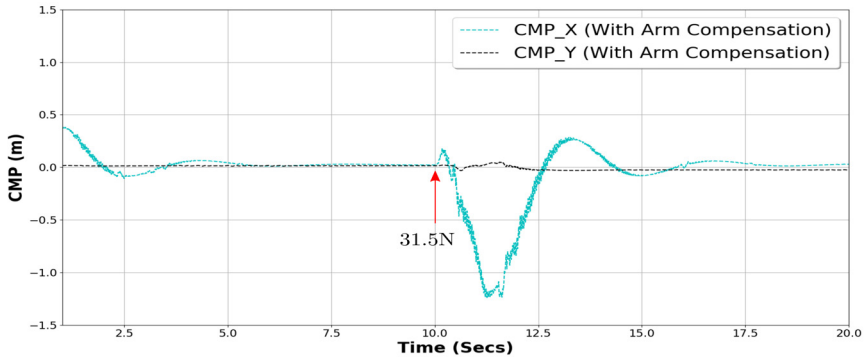
where c_z and f_z are the vertical components of the CoM position and ground reaction force, respectively.

4.6 Simulation Validation and Results

In this section, we present the details of the simulation setup and experimental results. The bipedal wheel-legged robot was designed and simulated using the Gazebo simulator, and the robot operating system (ROS) was used to implement the control framework in C++ [52][90]. The lower body and manipulator controls of the robot were decoupled from each other, so they were implemented separately. In all experiments, the robot carried a can of cola in its manipulator hand to simulate a real-world example, as shown in Figure



(a) 27N force push.



(b) 31.5N force push.

Figure 4.5 Temporal variation of the centroidal moment pivot (CMP) point under the translational disturbances while the robot is holding a typical can.

4.1b. The stretched arm of the robot was 0.6 m in length, and in all experiments the initial arm posture remained at the home configuration $({}_{A}x_{ee} = 0.3, {}_{A}y_{ee} = 0)$ to hold an object in front of the robot body and to allow ample room for arm motion.

4.6.1 Translational Disturbance

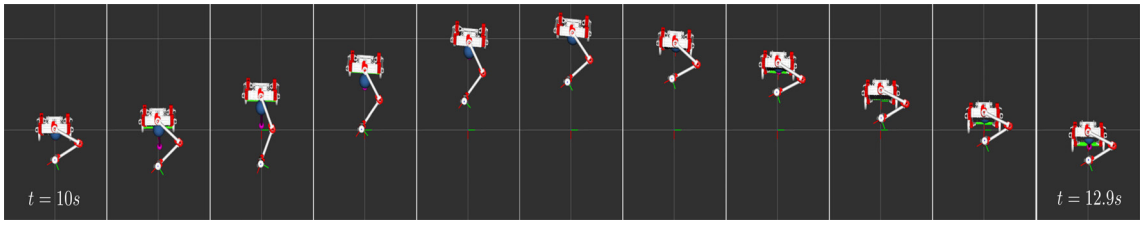
We began by applying translational pushes to the robot in the $-X_C$ direction, and the arm is actively used to cancel these external disturbances as shown in Figure 4.6. Multiple experiments were performed to characterize the effects of arm acceleration on the CMP stability of the robot. Additionally, to statistically improve the results, each experiment was performed five times and the mean errors and standard deviations were computed. The results are presented in Table 4.1 that compares the fixed arm cases and the active arm cases together with percentage differences information. As force was applied to the

sagittal plane of the robot, errors were measured as time integration in its translational direction to measure the overall amount of instability over the time. Furthermore, to observe the implications of object mass on system stability, we divided the experiments into two categories. The first set of tests was performed with endpoint mass of 360g assuming that the robot is holding a typical canned drink. In the second set of experiments, we doubled the endpoint mass to 720g and its respective moment of inertia.

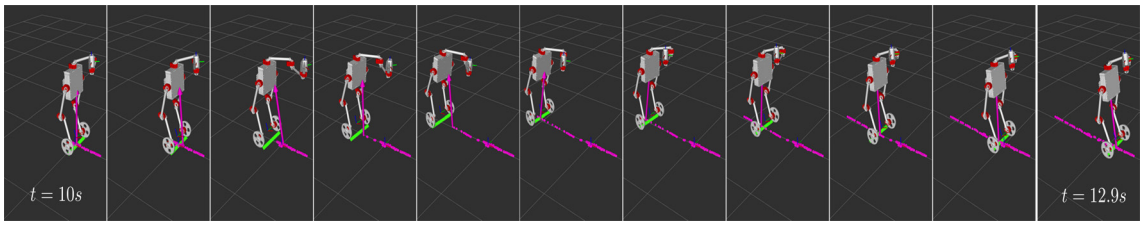
In the first experiment, we used the threshold force of 20 N that could activate the arm acceleration to cancel the external translational disturbances. The linear force was applied to the robot body at time $t = 10$s of the experiment, first in the fixed arm posture and then with the end-effector acceleration control. The results demonstrate that the arm acceleration to cancel the effects of the external push reduced the CMP mean error by more than 12% over the fixed arm configuration. This difference is more pronounced in the case of the heavy can, where the CMP error was reduced by over 18%.

In the second experiment, the external force applied was 23 N. When the robot was carrying a typical can the CMP error decreased from 2.2044 m.s to 1.8885 m.s thanks to the active arm usage. Likewise, when the can mass was doubled, the CMP error decreased further from 2.2098 m.s to 1.7901 m.s. In the typical and doubled examples, the errors were reduced by 14.33% and 18.99%, respectively.

Likewise, the same experiments were repeated for 25 and 27 N forces. The results of the 27 N force experiment, in which the robot is holding a typical can, are shown in Figure 4.5a for both the fixed arm case and arm acceleration method. It is clear that the robot successfully maintains a small CMP error while using its arm compared to when the arm is fixed. We found that without using the end-effector acceleration(the fixed arm configuration) the robot could only sustain the maximum translational push of 27 N. However, the end-effector acceleration could increase its ability to maintain balance for a push of up to 31.5 N for a typical can and 32.5 N when the can mass was doubled. Figure 4.5b demonstrates the CMP of the robot when a force of 31.5 N was applied to the front of its body.



(a) Top view of the robot during frontal push.



(b) Perspective view of the robot during frontal push.

Figure 4.6 Wheel-legged robot reacting to the frontal push of 27N with active arm acceleration control. In the first scene, the disturbance is applied to the robot's front side. The arm starts to make extension action as it is reacting actively to cancel this disturbance. The blue sphere shows the total robot CoM and the pink sphere shows the CMP point. Just after the push, the mismatch between the CMP and the ground projected CoM is large due to the external disturbance. However, after making the arm extension action, this mismatch is disappeared in a few scenes, which demonstrates the balance recovery.

4.6.2 Rotational Disturbance

We define rotational disturbance as an external moment applied about the robot's vertical axis. During human-robot interactions, such type of strong external forces can destabilize the robot and pose the risk to humans and/or other nearby objects. For the purpose of measuring the effectiveness of the arm compensation to mitigate external moments, we applied a 16 Nm torque about the Z-axis of the robot body, as seen in Figure 4.7. To compensate for the sudden rotation as a result of external torque, the robot moves its end-effector in the opposite direction of the robot's rotation to generate a counter angular acceleration.

It is clear from the plot that the yaw angle stays small when active arm compensation is used. Furthermore, the settling time for the yaw angle is also shorter than the without arm compensation case. However, these differences are negligible and may not make significant difference in real-world situations. A major reason for this behavior could be

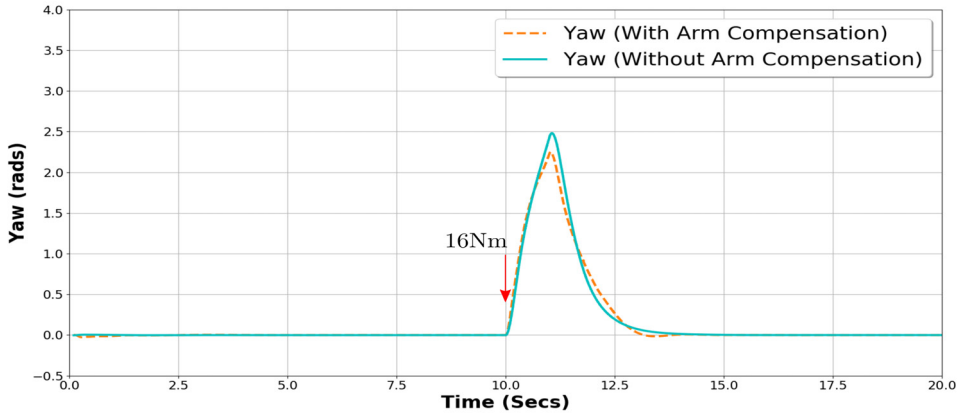


Figure 4.7 Rotational disturbance.

that we put limitations on the arm joints i.e. $-\pi/2 \leq \theta_1 \leq \pi/2$, this limits the arm's ability to generate adequate angular motion.

4.6.3 Lateral Disturbance

When a body moves around a fixed point in a circular path, an inertial force acts radially outwards on the body, called the centrifugal force. This force, along with external lateral forces, can cause rollover accidents on roads if vehicles try to navigate sharp curved paths at higher speeds. However, this phenomenon is not limited to road vehicles, as modern mobile robots are equally susceptible to these forces. To counter the effects of centrifugal force on a wheel-legged robot in circular trajectories, Siegwart *et al.* proposed regulating the roll angle of the robot using its knee joints as a function of the zero-moment point [41].

When humans ride a bike over a sharp curvature, they anticipate the potential centrifugal force in order to increase the stability margin. Inspired by this fact, we propose active arm usage for traversing curved trajectories by the wheel-legged robot. In this section, we present the results of the proposed method in which the robot uses its manipulator arm to move its CoM and to shift the balance point toward the center of the circular path, thereby improving its robustness against lateral forces.

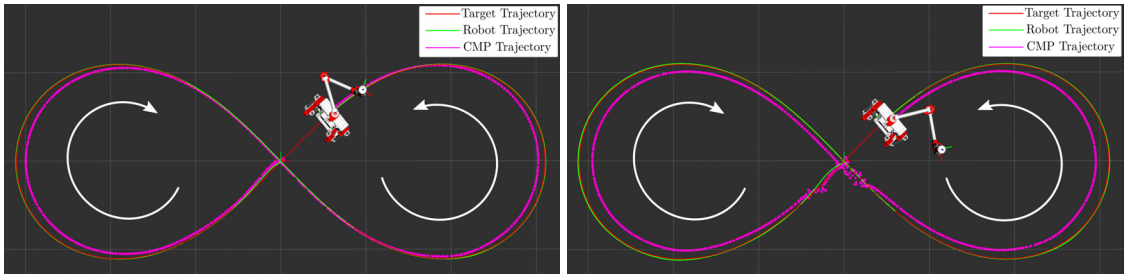
Figure 4.8 shows the experiment in which the robot traverses an infinity-loop shape

External Force	Endpoint Mass $360g$		
	Fixed Arm	**Active Arm**	**Percent Change**
	$\int \lvert e_x \rvert dt$	$\int \lvert e_x \rvert dt$	
20N	1.9582±0.0062	1.7076±0.0042	12.7948%
23N	2.2044±0.0161	1.8885±0.0019	14.3302%
25N	2.3469±0.0083	2.0213±0.0102	13.8746%
27N	2.4931±0.0226	2.0539±0.0164	17.6137%
31.5N	N/A	2.2887±0.0221	-
	Endpoint Mass $720g$		
	Fixed Arm	**Active Arm**	**Percent Change**
	$\int \lvert e_x \rvert dt$	$\int \lvert e_x \rvert dt$	
20N	1.9778±0.0073	1.6207±0.0008	18.0554%
23N	2.2098±0.0120	1.7901±0.0067	18.9938%
25N	2.3529±0.0078	1.9139±0.0072	18.6565%
27N	2.4983±0.0045	2.0401±0.0123	18.3414%
32.5N	N/A	2.2160±0.0136	-

Table 4.1: Balance stability indicated by the time integrated CMP errors as a result of the forces applied to the robot body, along with the standard deviations. The smaller CMP errors indicate higher balance stability. e_x denotes the CMP error in the robot's sagittal plane. In contrast, N/A implies that the robot falls over as a result of the corresponding external force. For all cases, the active arm usage improves the balance stability as indicated by the percentage change compared to the fixed arm case.

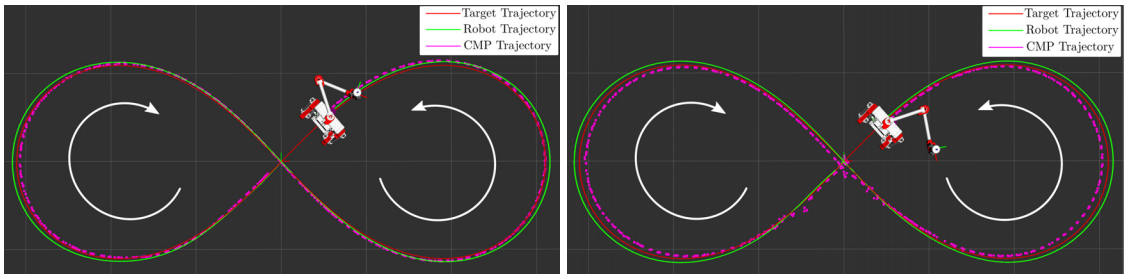
trajectory at different speeds. We chose the infinity-loop shaped trajectory because it provides a good combination of linear and circular paths. First, we start with the normal speed in which the maximum translational velocity is $0.3\ m/s$ and maximum yaw velocity is $0.3\ rads/s$. Figures 4.8a and 4.8b show the experiments in a ROS-Gazebo environment for the fixed arm and active arm cases, respectively. We can clearly see that the robot follows the target trajectory completely in both cases when it is moving rather slowly. However, it is evident from Figure 4.8b that by moving the arm towards the center of the curvature, the CMP not only remains inside the loop but also has a bigger margin from the target trajectory than in the fixed arm case (Figure 4.8a).

In the second experiment, we enhanced the maximum translational velocity of the robot to $0.7\ m/s$ and the maximum yaw velocity to $0.6\ rads/s$. The inertial forces on



(a) Fixed arm at $v_{max} = 0.3$ m/s and $\dot{\alpha}_{max} = 0.3$ rads/s.

(b) Active arm at $v_{max} = 0.3$ m/s and $\dot{\alpha}_{max} = 0.3$ rads/s.



(c) Fixed arm at $v_{max} = 0.7$ m/s and $\dot{\alpha}_{max} = 0.6$ rads/s.

(d) Active arm at $v_{max} = 0.7$ m/s and $\dot{\alpha}_{max} = 0.6$ rads/s.

Figure 4.8 Infinity-loop trajectory tracking with the wheel-legged robot. By increasing motion speed, we observe the balance point indicated by CMP is going outward against the target trajectory for the fixed arm case. In contrast, by articulating the arm, this balance point can be manipulated toward the inside even at high speeds, which allows increasing balance stability margin even if there is outward centrifugal force during a curved trajectory tracking.

the robot increased as the robot speed increased on the curves. Additionally, the CMP moved radially outwards, as seen in Figures 4.8c and 4.8d. Furthermore, in both cases it is clear that the robot does not closely follow the target trajectory and its path diverges from the goal trajectory because of the increased speed and consequential inertial forces. Nevertheless, Figure 4.8d suggests that the robot has a higher stability margin for the lateral disturbances as the CMP remains inside the target trajectory and closer toward the center of the curvature.

4.7 Discussion

The simulation results clearly demonstrate that the proposed arm acceleration control improves the overall translational and lateral stability of the wheel-legged self-balancing

robot. However, this approach depends on the mass and inertia of the object carried by the end-effector and the initial configuration of the end-effector itself. The video demonstration of the presented experiments is available at `https://youtu.be/KQ1mbQeLFtc`

Figure 4.6 shows the motion of the robot arm during a frontal push of 27 N. We can observe that the robot starts to accelerate its end-effector towards the forward position as soon as the force is applied, and once it returns to its reference position the end-effector also returns to the home configuration. The first part of Table 4.1 shows how the arm acceleration with holding a typical can reduces the errors as the external force increases on the robot. However, it is also clear from the second part of the table that when the can mass is doubled the errors cannot be reduced after a certain level. This ascertains a fundamental limit on reducing the effects of external forces through the use of a manipulator arm. Indeed, this is a similar situation to the human movement. Humans use their arms to increase balance stability, but there is a certain limit related to the mechanical dynamics condition where the arm mass portion is limited against the total whole-body mass.

In addition, the use of the arm can increase the robot's maximum tolerance to the external linear push from 27 to 31.5 N with the typical can carrying task and 32.5 N when the mass of the can is doubled, an increase of 16.67% and 20.37% compared to the fixed arm case, respectively. This makes a significant contribution to the overall system stability. This concept would lead to the wheel-legged biped robot with an active arm for dual purposes, one is for carrying objects, another is for increasing the balance stability. This point is important for the future application of wheel-legged biped robots in a real-world environment with complex interactions.

4.8 Conclusion

In this chapter, we proposed and validated a novel scheme to improve the stability of a wheel-legged biped robot with active usage of its arm manipulator. To assess the system stability, we also used the CMP as an evaluation metric, which is commonly used

for humanoid robots but had not been previously used for wheel-legged biped robots. Through several experiments, it was shown that the proposed method improved stability against translational disturbances and centrifugal and lateral forces when traversing curves at high speeds.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This dissertation addresses the stability and robustness problem of self-balancing wheel-legged robots that are underactuated and nonlinear in nature. Several motion control strategies are developed and tested to control a three degrees of freedom (DoF) wheel-legged robot called Igor on a horizontal plane.

In chapter 2, we derived the equations of motion of the robot using the famous Lagrange-Euler method by assuming it as a wheeled inverted pendulum. During the mathematical modeling of the system, we also considered the viscous friction of the wheel joints and non-holonomic constraints due to the differential drive. The nonlinear equations of motion of the robot are then linearized about the equilibrium point of the pitch angle ($\beta \approx 0$). After state-space modeling of the robot, we developed a linear state-feedback controller using the pole-placement method and a linear quadratic regulator (LQR). Both controllers were tested in the Gazebo simulator under sensor noise and external translational and rotational disturbances. These results confirmed the advantages of the optimal LQR over a manually tuned state-feedback controller. Lastly, we developed a nonlinear model-based Computed Torque controller and combined it with the LQR. The simulations and experimental results on the real robot suggested that the combined controller is more robust

against the model uncertainties and outperforms the linear LQR and the nonlinear Computed Torque controller.

In chapter 3, we developed a state-of-the-art $\mathcal{L}_1$ adaptive controller to stabilize the robot's pitch angle. The $\mathcal{L}_1$ adaptive controller is adopted because it gives a predictable transient response and guarantees the robustness of the closed-loop system. Furthermore, it decouples the estimation loop from the control loop and hence enabling high adaptation gains without compromising system stability. The proposed control scheme combines the $\mathcal{L}_1$ adaptive controller with the full state-feedback LQR to improve the robot's robustness against the system modeling errors, parameter uncertainties, and unknown external disturbances. Simulation results showed that the proposed $\mathcal{L}_1$ augmented LQR controller kept the robot balanced against a big translational push even when the viscous friction coefficients of the wheel joints were set inaccurate. Similarly, the proposed control scheme and the standalone LQR were tested on the real robot under external disturbances and different ground friction scenarios. The real-time experiments also validated the effectiveness of the proposed $\mathcal{L}_1$ augmented LQR controller that achieved accurate control under uncertain conditions due to its fast adaptation.

In the end, inspired by the humans' and other animals' ability to use their limbs to improve their body balance, chapter 4 proposed a novel control scheme that uses an arm manipulator on top of the wheel-legged robot to mitigate external disturbances and enhance the overall stability. Furthermore, we used the centroidal moment pivot (CMP)-a key criterion to determine stability in humans for a wheel-legged robot. The two-link arm manipulator is controlled by the resolved acceleration control method while the lower body is stabilized by the LQR. A finite-state machine is designed to determine the behavior of the arm's end-effector when the robot is under the influence of certain linear or rotational accelerations. Several simulation tests proved that the robot successfully attenuated the effects of external forces and also improved its stability while traversing curved paths by the use of an active arm.

5.2 Future Work

Due to higher speed, maneuverability, and agility, wheel-legged robots have the potential to be used in a variety of environments such as homes, smart warehouses, hospitals, hotel lobbies, etc. However, these robots are prone to fall over due to their dynamic balancing and underactuated nature. In this study, we developed different control strategies to improve the stability and robustness of these robots, nonetheless, there remain many challenges to be solved in the future. For example, in this study we assumed a flat ground on which the robot is operating, however, in reality, this is seldom the case. One potential solution to overcome this assumption is to use impedance control for the knee and hip joints. It will mitigate the effects of uneven surfaces as the robot will adjust its legs accordingly instead of keeping the leg configuration rigid.

Second, we augmented an $\mathcal{L}_1$ adaptive controller with the LQR, since the $\mathcal{L}_1$ adaptive control theory is lacking for the underactuated MIMO systems, for this reason, we designed $\mathcal{L}_1$ adaptive control to stabilize the pitch angle of the robot only. Future work can be extended to develop an $\mathcal{L}_1$ adaptive controller to stabilize and control all three degrees of freedom of the robot. Furthermore, as discussed in section 3.7, the design of the low-pass filter of the $\mathcal{L}_1$ adaptive controller determines the performance of the closed-loop system. A systematic approach to design an optimal low-pass filter for the $\mathcal{L}_1$ adaptive controller is still required.

We believe machine learning approaches will be critical in improving the stability and robustness of the underactuated wheel-legged robots. As yet these data-driven methods have not been widely explored and implemented on real wheel-legged robots $\big($[5] being the exception$\big)$ because of the challenges of sim-to-real transfer and underactuation in the sense of input-output of these robots. Therefore, another possible future research direction is to develop learning-based control methods for nonlinear wheel-legged robots.

Appendix A

Experimental Wheel-legged Robot Igor

In this appendix, we present the details of the experimental wheel-legged robot Igor used in this research. Igor as seen in Figure A.1 is a modular wheel-legged robot with 14 degrees of freedom designed by Hebi Robotics, a spin-off company from Carnegie Mellon University in the United States [55]. The primary advantage of this robot is that it is modular and its configuration can be adapted according to one's needs. Since in this research we are only concerned with the lower body dynamics and motion control, robot arms are not used. Excluding the robot's arms that are usually used for object manipulation, its degrees of freedom are reduced to six, three on each leg.

The robot's maximum height is about 1 meter, and simple hollow steel tubes are used as its legs. Each leg consists of two parts i.e. upper leg (0.35 meters in length) and the lower leg (0.30 meters in length). Also, the distance between the centers of the two wheels is about 0.50 meters. Igor comes with a PD controller, and therefore can be controlled via its joystick or mobile phone app. Even though the battery life of the robot is $1-2$ hours but it has the ability to hot-swap the batteries for longer usage.

A.1 Robot Modules

The Igor robot is consist of five main modules,

Figure A.1 A fully assembled 14 DoF Igor robot.

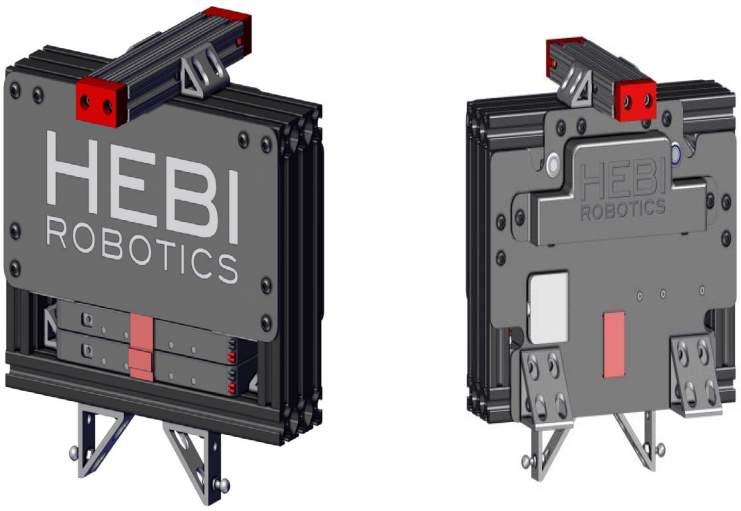
- Main chassis that includes an Intel computer that runs a Linux-based Ubuntu operating system, an Ethernet switch, and a WiFi module for wireless connectivity.

- Two Grin Tech's LiGo batteries of 98 Wh capacity each to power the whole robot.

- Hip, knee, and wheel actuators.

- Two gas springs to support the load on robot's knees.

- 8-inch diameter wheels.



(a) Front of the chassis. **(b)** Back of the chassis.

Figure A.2 Main chassis of Igor robot.

A.1.1 Robot Chassis

Igor has a squared 12×12 inches chassis that includes a computer and batteries as shown in Figure A.2. The Intel NUC computer works as the central processing unit of the robot and all the actuators are connected to it. Intel NUC is equipped with a Core i3 processor that can clock up to 3.60GHz with 4M of cache memory, 8GB of RAM, and 256GB of SSD. Furthermore, to connect other peripheral devices such as a monitor, keyboard, Lidar, etc. with the computer, it has an open Ethernet port, four USB ports, and an HDMI port. A power converter is used to convert the 36 volts of the batteries to 12 volts for the NUC computer board.

A.1.2 Actuators

The Hebi robotics' X-series actuator modules are smart series elastic actuators (SEAs) that combine BLDC motor, spring, geartrain, and electronic control unit in a compact casing. Every actuator module is equipped with an Ethernet port for high-speed communication and multiple sensors like IMU, torque sensor, encoder, etc. Furthermore, each actuator module runs a real-time operating system (RTOS) to collect and filter the data from these sensors. These actuator modules provide simultaneous control of position, ve-

Figure A.3 Hebi Robotics' X-Series actuator module.

locity, and torque at a rate of up to 1 kHz. Three different types of actuators are employed for hips, knees, and wheel actuation based on the joint's torque requirements.

- The X5-9 model is used for hip joints because of its relatively little torque needs. This module is capable of generating about 9 N.m of torque at a lower speed.
- Robot knee joints bear most of its load and hence require powerful actuators. The X8-16 model is therefore deployed for knee joints. This actuator module can produce up to 16 N.m of torque at lower speeds.
- To balance the robot, its wheels need to have a higher speed. For this reason, the X8-3 model is used for wheel joints which can run up to 80 RPM at lower torques.

Bibliography

[1] Naoyuki Yoshino, Chul Ju Kim, and Pitchaya Sirivunnabood., "Aging population and its impacts on fiscal sustainability." [Online]. Available: https://t20japan.org/policy-brief-aging-population-impacts-fiscal-sustainability/

[2] F. Grasser, A. D'Arrigo, S. Colombi, and A. Rufer, "Joe: a mobile, inverted pendulum," *IEEE Transactions on Industrial Electronics*, vol. 49, no. 1, pp. 107–114, 2002.

[3] S. Xin, Y. You, C. Zhou, C. Fang, and N. Tsagarakis, "A torque-controlled humanoid robot riding on a two-wheeled mobile platform," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017.

[4] V. Klemm, A. Morra, C. Salzmann, F. Tschopp, K. Bodie, L. Gulich, N. Küng, D. Mannhart, C. Pfister, M. Vierneisel, F. Weber, R. Deuber, and R. Siegwart, "Ascento: A two-wheeled jumping robot," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7515–7521.

[5] L. Cui, S. Wang, J. Zhang, D. Zhang, J. Lai, Y. Zheng, Z. Zhang, and Z.-P. Jiang, "Learning-based balance control of wheel-legged robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7667–7674, 2021.

[6] Department of Economics and Social Affairs, United Nations., "World population prospects 2019." [Online]. Available: https://population.un.org/wpp/Graphs/Probabilistic/POP/60plus/900

[7] Statistics Bureau of Japan., "Statistical handbook of japan 2021." [Online]. Available: https://www.stat.go.jp/english/data/handbook/c0117.html

[8] Masanobu Masuda, and Katsuhisa Kojima, "Japanese social security for the elderly from a viewpoint of life cycles," *Review of Population and Social Policy*, vol. 10, pp. 37 – 54, 2001.

[9] Amazon., "Amazon warehouse robots." [Online]. Available: https://www.aboutamazon.com/news/innovation-at-amazon/bots-by-the-numbers-facts-and-figures-about-robotics-at-amazon

[10] The Robot Report., "Jd logistics robots." [Online]. Available: https://www.therobotreport.com/chinese-startup-syrius-robotics-builds-warehouse-robots-with-jd-logistics/

[11] Amazon., "Amazon air prime." [Online]. Available: https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011

[12] Starship technologies., "Starship technologies delivery robot." [Online]. Available: https://www.starship.xyz/

[13] Business Insider., "Last mile delivery challenges." [Online]. Available: https://www.businessinsider.com/last-mile-delivery-shipping-explained

[14] ZMP Inc. Japan., "DeliRo delivery robot." [Online]. Available: https://www.zmp.co.jp/en/products/lrb/deliro

[15] iRobot., "Roomba vacuum cleaning robot." [Online]. Available: https://www.irobot.com/roomba

[16] Softbank Robotics., "Pepper humanoid robot." [Online]. Available: https://www.softbankrobotics.com/emea/en/pepper

[17] Sony Corporation., "Aibo dog robot." [Online]. Available: https://en.wikipedia.org/wiki/AIBO

[18] Boston Dynamics., "Atlas humanoid robot." [Online]. Available: https://www.bostondynamics.com/atlas

[19] Honda., "Asimo." [Online]. Available: https://en.wikipedia.org/wiki/ASIMO

[20] Z. Li, C. Yang, and L. Fan, *Advanced Control of Wheeled Inverted Pendulum Systems*. Springer, 2013.

[21] Boston Dynamics., "Introducing handle." [Online]. Available: https://www.youtube.com/watch?v=-7xvqQeoA8c

[22] E. Koyanagi, S. Iida, K. Kimoto, and S. Yuta, "A wheeled inverse pendulum type self-contained mobile robot and its two-dimensional trajectory control," in *2nd International symposium, Measurement and control in robotics*. JIRA, 1992, pp. 891–898.

[23] Y. Ha and S. Yuta, "Trajectory tracking control for navigation of self-contained mobile inverse pendulum," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1994.

[24] N. Shiroma, O. Matsumoto, and K. Tani, "Cooperative behavior of a mechanically unstable mobile robot for object transportation," *JSME International Journal Series C*, vol. 42, no. 4, pp. 965–973, 1999.

[25] M. Baloh and M. Parent, "Modeling and model verification of an intelligent self-balancing two-wheeled vehicle for an autonomous urban transportation system," in *The Conference on Computational Intelligence, Robotics, and Autonomous Systems*, 2003.

[26] K. Pathak, J. Franch, and S. K. Agrawal, "Velocity control of a wheeled inverted pendulum by partial feedback linearization," in *IEEE Conference on Decision and Control (CDC)*, 2004.

[27] Kaustubh Pathak, Jaume Franch, and Sunil K. Agrawal, "Velocity and position control of a wheel inverted pendulum by partial feedback linearization," *IEEE Transactions on Robotics*, vol. 21, no. 3, p. 505-513, 2005.

[28] H. Nijmeijer and A. J. van der Schaft, *Nonlinear Dynamical Control Systems*. Springer-Verlag, 1990.

[29] T. B. Lauwers, G. A. Kantor, and R. L. Hollis, "A dynamically stable single-wheeled mobile robot with inverse mouse-ball drive," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2006, pp. 2884–2889.

[30] S. Jeong and T. Takahashi, "Wheeled inverted pendulum type assistant robot: inverted mobile, standing, and sitting motions," in *IEEE International Conference on Intelligent Robots and Systems (IROS)*, 2007, pp. 1932–1937.

[31] Seonghee Jeong, and Takayuki Takahashi, "Wheeled inverted pendulum type assistant robot: design concept and mobile control," *Intelligent Service Robotics*, vol. 1, p. 313-320, 2008.

[32] L. Canete, and Takayuki Takahashi, "Modeling, analysis and compensation of disturbances during task execution of a wheeled inverted pendulum type assistant robot using a unified controller," *Advanced Robotics*, vol. 29, no. 22, p. 1453-1462, 2015.

[33] M. Stilman, J. Olson, and W. Gloss, "Golem krang: Dynamically stable humanoid robot for mobile manipulation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2010, pp. 3304–3309.

[34] Jian Huang, Zhi-Hong Guan, Takayuki Matsuno, Toshio Fukuda, and Kosuke Sekiyama, "Sliding-mode velocity control of mobile-wheeled inverted-pendulum systems," *IEEE Transactions on Robotics*, vol. 26, no. 4, p. 750-758, 2010.

[35] Jian-Xin Xu, Zhao-Qin Guo, and Tong Heng Lee, "Design and implementation of integral sliding-mode control on an underactuated two-wheeled mobile robot," *IEEE Transactions on Industrial Electronics*, vol. 61, no. 7, p. 3671-3681, 2014.

[36] Fuquan Dai, Xueshan Gao, Shigong Jiang, Wenzeng Guo, and Yubai Liu, "A two-wheeled inverted pendulum robot with friction compensation," *Mechatronics*, vol. 30, p. 116‑125, 2015.

[37] M. Zafar and H. I. Christensen, "Whole body control of a wheeled inverted pendulum humanoid," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2016, pp. 89–94.

[38] M. Zafar, S. Hutchinson, and E. A. Theodorou, "Hierarchical optimization for whole-body control of wheeled inverted pendulum humanoids," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 7535–7542.

[39] G. Zambella, G. Lentini, M. Garabini, G. Grioli, M. G. Catalano, A. Palleschi, L. Pallottino, A. Bicchi, A. Settimi, and D. Caporale, "Dynamic whole-body control of unstable wheeled humanoid robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, p. 3489‑3496, 2019.

[40] Yaxian Xin, Hui Chai, Yibin Li, Xuewen Rong, Bin Li, and Yueyang Li, "Speed and acceleration control for a two wheel-leg robot based on distributed dynamic model and whole-body control," *IEEE Access*, vol. 7, pp. 180 630 – 180 639, 2019.

[41] V. Klemm, A. Morra, L. Gulich, D. Mannhart, D. Rohr, M. Kamel, Y. de Viragh, and R. Siegwart, "LQR-assisted whole-body control of a wheeled bipedal robot with kinematic loops," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, p. 3745‑3752, 2020.

[42] Sangtae Kim and SangJoo Kwon, "Nonlinear optimal control design for underactuated two-wheeled inverted pendulum mobile platform," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 6, pp. 2803 – 2808, 2017.

[43] L.-G. Lin and M. Xin, "Nonlinear control of two-wheeled robot based on novel analysis and design of SDRE scheme," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 1140 – 1148, 2020.

[44] E. Li, H. Feng, H. Zhou, X. Li, Y. Zhai, S. Zhang, and Y. Fu, "Model learning for two-wheeled robot self-balance control," in *IEEE International Conference on Intelligent Robotics and Biomimetics*, 2019, pp. 1582–1587.

[45] Chih-Hung Gilbert Li, Long-Ping Zhou, and Yu-Hua Chao, "Self-balancing two-wheeled robot featuring intelligent end-to-end deep visual-steering," *IEEE Transactions on Mechatronicsl*, vol. 26, no. 5, p. 2263‑2273, 2021.

[46] D. Ruiken, J. P. Cummings, U. R. Savaria, F. C. S. IV, and R. A. Grupen, "uBot-7: A dynamically balancing mobile manipulator with series elastic actuators," in *IEEE International Conference on Humanoid Robotics (Humanoids)*, 2017, pp. 676–682.

[47] S. R. Kuindersma, E. Hannigan, D. Ruiken, and R. A. Grupen, "Dexterous mobility with the uBot-5 mobile manipulator," in *International Conference on Advanced Robotics*, 2009, pp. 1–7.

[48] Nilanjan Sarkar, Xiaoping Yun, and R. Vijay Kumar, "Control of mechanical systems with rolling constraints: Application to dynamic control of mobile robots," *The International Journal of Robotics Research*, vol. 13, no. 1, p. 55‑69, 1994.

[49] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Springer, 2009.

[50] K. J. Astrom and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2008.

[51] A. E. Bryson, *Applied Optimal Control: Optimization, Estimation and Control*. Hemisphere Publishing Corporation, 1975.

[52] Open Source Robotics Foundation., "Gazebo." [Online]. Available: http://gazebosim.org/

[53] S. Russell, "ODE ‑open dynamics engine." [Online]. Available: http://www.ode.org

[54] J. Craig, *Introduction to Robotics: Mechanics and Control*. Pearson, 2017.

[55] Hebi Robotics., "Igor." [Online]. Available: https://www.hebirobotics.com/robotic-kits

[56] S. Dubowsky and D. T. DesForges, "The application of model-referenced adaptive control to robotic manipulators," *Journal of Dynamic Systems, Measurement, and Control*, vol. 101, no. 3, p. 193-200, 1979.

[57] Roberto Horowitz and Masayoshi Tomizuka, "An adaptive control scheme for mechanical manipulators−compensation of nonlinearity and decoupling control," *Journal of Dynamic Systems, Measurement, and Control*, vol. 108, no. 2, p. 127-135, 1986.

[58] Jean-Jacques E. Slotine and Weiping Li, "On the adaptive control of robot manipulators," *The International Journal of Robotics Research*, vol. 6, no. 3, p. 49-59, 1987.

[59] Jean-Jacques E. Slotine and M.D. Di Benedetto, "Hamiltonian adaptive control of spacecraft," *IEEE Transactions on Automatic Control*, vol. 35, no. 7, pp. 848–852, 1990.

[60] K. S. Narendra and A. M. Annaswamy, "A new adaptive law for robust adaptation without persistent excitation," *IEEE Transactions on Automatic Control*, vol. 32, no. 2, p. 134-145, 1987.

[61] K. Narendra and A. Annaswamy, "Robust adaptive control in the presence of bounded disturbances," *IEEE Transactions on Automatic Control*, vol. 31, no. 4, pp. 306–315, 1986.

[62] T. I. Fossen and S. I. Sagatun, "Adaptive control of nonlinear underwater robotic systems," in *IEEE International Conference on Robotics and Automation (ICRA)*, 1991, pp. 1687–1694.

[63] G. Antonelli, F. Caccavale, S. Chiaverini, and G. Fusco, "A novel adaptive control law for autonomous underwater vehicles," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2001, pp. 447–452.

[64] C. Cao and N. Hovakimyan, "Design and analysis of a novel L1 adaptive controller, part i: control signal and asymptotic stability," in *American Control Conference (ACC)*, 2006, pp. 3397–3402.

[65] M. Bennehar, A. Chemori, F. Pierrot, and V. Creuze, "Extended model-based feedforward compensation in L1 adaptive control for mechanical manipulators: Design and Experiments," *Frontiers in Robotics and AI*, vol. 2, 2015.

[66] D. Maalouf, A. Chemori, and V. Creuze, "L1 adaptive depth and pitch control of an underwater vehicle with real-time experiments," *Ocean Engineering*, vol. 98, pp. 66–77, 2015.

[67] Hung Truong Xuan, Ahmed Chemori, Tuan Pham Anh, Huy Le Xuan, Thu Phan Hoai, and Phuong Vu Viet, "From PID to L1 adaptive control for automatic balancing of a spacecraft three-axis simulator," *International Journal of Emerging Technology and Advanced Engineering*, vol. 6, no. 1, pp. 77–86, 2016.

[68] K. W. Lee and S. N. Singh, "Multi-input submarine control via L1 adaptive feedback despite uncertainties," *Journal of Systems and Control Engineering*, vol. 228, no. 5, pp. 330–347, 2014.

[69] H. Jafarnejadsani, H. Lee, N. Hovakimyan, and P. Voulgaris, "A multirate adaptive control for mimo systems with application to cyber-physical security," in *IEEE Conference on Decision and Control (CDC)*, 2018, pp. 6620–6625.

[70] H. Lee, S. Synder, and N. Hovakimyan, "L1 adaptive output feedback augmentation for missile systems," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 2, pp. 680 – 692, 2018.

[71] K. Pereida, R. Duivenvoorden, and A. P. Schoellig, "High-precision trajectory tracking in changing environments through L1 adaptive feedback and iterative learning," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 344–350.

[72] H. Jafarnejadsani, D. Sun, H. Lee, and N. Hovakimyan, "Optimized L1 adaptive controller for trajectory tracking of an indoor quadrotor," *Journal of Guidance, Control, and Dynamics*, vol. 40, no. 6, pp. 1415 – 1427, 2017.

[73] N. Hovakimyan and C. Cao, *L1 Adaptive Control Theory: Guaranteed Robustness with Fast Adaptation*. Society for Industrial and Applied Mathematics, 2010.

[74] R. G. Beall, "Engineering of fast and robust adaptive control for fixed-wing unmanned aircraft," Master's thesis, Naval Postgraduate School, June 2017.

[75] B. D. O. Anderson, "Failures of adaptive control theory and their resolution," *Communications in Information and Systems*, vol. 5, no. 1, pp. 1 – 20, 2005.

[76] T. Lankhorst, "Biquad library," https://github.com/tomlankhorst/biquad, 2016.

[77] D. Li, N. Hovakimyan, C. Cao, and K. A. Wise, "Filter design for feedback-loop trade-off of L1 adaptive controller: A linear matrix inequality approach," in *AIAA Guidance, Navigation and Control Conference*, 2008, p. 6280.

[78] M. Naghnaeian, P. G. Voulgaris, and N. Hovakimyan, "On robustness of L1 adaptive control with time varying perturbations & filter design," in *American Control Conference (ACC)*, 2012, pp. 1937–1942.

[79] H. Lee, "L1 adaptive control for nonlinear and non-square multivariable systems," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2017.

[80] K. J. Boström, T. Dirksen, K. Zentgraf, and H. Wagner, "The contribution of upper body movements to dynamic balance regulation during challenged locomotion," *Frontiers in Human Neuroscience*, vol. 12, no. 8, 2018.

[81] C. Walker, C. J. Vierck, and L. A. Ritz, "Balance in the cat: role of the tail and effects of sacrocaudal transection," *Behavioural Brain Research*, vol. 91, no. 2, pp. 41 – 47, 1998.

[82] J. Nabeshima, M. Y. Saraiji, and K. Minamizawa, "Arque: Artificial biomimicry-inspired tail for extending innate body functions," in *ACM SIGGRAPH 2019 Posters*, ser. SIGGRAPH '19, 2019.

[83] Marko B. Popovic, Ambarish Goswami, and Hugh Herr, "Ground reference points in legged locomotion: Definitions, biological trajectories and control implications," *The International Journal of Robotics Research*, vol. 24, no. 12, pp. 1013–1032, 2005.

[84] M. Hayashibe, A. González, and M. Tournier, "Personalized balance and fall risk visualization with kinect two," in *IEEE Engineering in Medicine and Biology Society (EMBC)*, 2020.

[85] J. Luh, M. Walker, and R. Paul, "Resolved-acceleration control of mechanical manipulators," *IEEE Transactions on Automatic Control*, vol. 25, no. 3, p. 468-474, 1980.

[86] A. Goswami and V. Kallem, "Rate of change of angular momentum and balance maintenance of biped robots," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2004, pp. 3785–3790.

[87] M. Popovic, A. Hofmann, and H. Herr, "Angular momentum regulation during human walking: biomechanics and control," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2004, pp. 2405–2411.

[88] D. Orin, "Interactive control of a six-legged vehicle with optimization of both stability and energy," Ph.D. dissertation, The Ohio State University, 1976.

[89] B. S. Lin and S. M. Song, "Dynamic modeling, stability and energy efficiency of a quadrupedal walking machine," in *IEEE International Conference on Robotics and Automation (ICRA)*, 1993, pp. 367–373.

[90] Open Source Robotics Foundation., "ROS ‑robot operating system." [Online]. Available: https://www.ros.org/

List of Publication

Journal paper

1. **F. Raza**, W. Zhu, M. Hayashibe: "Balance Stability Augmentation for Wheel-Legged Biped Robot Through Arm Acceleration Control," in *IEEE Access, vol. 9, pp. 54022-54031, 2021*.
   This paper is related to Chapter 4.

Peer-reviewed International conferences

1. **F. Raza**, A. Chemori, M. Hayashibe: "A new Augmented $\mathcal{L}_1$ Adaptive Control for Wheel-Legged Robots: Design and Experiments," *2022 American Control Conference, 2022*, Accepted.
   This paper is related to Chapter 3.

2. W. Zhu, **F. Raza**, M. Hayashibe: "Reinforcement Learning Based Hierarchical Control for Path Tracking of a Wheeled Bipedal Robot with Sim-To-Real Framework," *2022 IEEE/SICE International Symposium on System Integrations (SII), 2022*, Accepted.

3. **F. Raza**, M. Hayashibe: "Towards Robust Wheel-Legged Biped Robot System: Combining Feedforward and Feedback Control," *2021 IEEE/SICE International*

Symposium on System Integrations (SII), Iwaki, Fukushima, Japan, 2021, *pp.* 606–612.

This paper is related to Chapter 2.

4. **F. Raza**, D. Owaki, M. Hayashibe: "Modeling and Control of a Hybrid Wheeled Legged Robot: Disturbance Analysis," *2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM), 2020*, *pp.* 466–473.

   This paper is related to Chapter 2.

Acknowledgments

I am deeply grateful to my advisor Prof. Mitsuhiro Hayashibe for his guidance, inspiration, and support throughout my Ph.D. studies. His supervision has promoted my adherence to high academic standards and high-quality work. I would like to express my gratitude to the committee members Prof. Kazuya Yoshida, Prof. Yasuhisa Hirata, and Prof. Ahmed Chemori for their helpful comments to improve my dissertation. I am also grateful to Prof. Chemori for his guideline and invaluable feedback during my research on the $\mathcal{L}_1$ adaptive controller for the wheel-legged robot.

My thanks also go to our research group alumni Keli Shen for discussing academic subjects with great pleasure. My sincere thanks to Muhammad Jehanzeb Khan for the thoughtful discussions that have been a significant aid in developing the ideas of my dissertation. I am also thankful to my lab-mate Zhu Wei for working together on reinforcement learning based path tracking for the wheel-legged robot. In addition, friendly support from all the other lab-mates is greatly acknowledged.

Last but not least, heartfelt appreciation and gratitude go towards my late father. He taught me all the important lessons of life with love, kindness, and patience. He made me believe in myself and inculcated me with the ideals of hard work, persistence, and integrity. I am indebted to my loving mother for her sacrifices, unbounded love, and patience during the course of my graduate studies. This work could never have been completed without having her blessings. I am also profoundly grateful and indebted to my brothers Hammad Khan and Jawad Hassan, who took care of our parents while I stayed away from home in the pursuit of my dreams. This dissertation would not have been possible without the unconditional love and support from my whole family.