University of Cape Town

EEE5000W

FULL DISSERTATION: MSC(ENG) Electrical Engineering

Optimizing Dynamic Locomotion in Baleka II: From Simulation to Real-World Running

Author:
Zubair Martin
(MRTZUB001)
Supervisors:
Stacey Shield
Amir Patel

April 26, 2025

Abstract

In the field of legged locomotion, agility is a critical area of research in robotics due to its potential to enable versatile movement for various applications, including search and rescue missions. However, bipedal robots face significant challenges in achieving rapid movements, such as maintaining stability and agility. This dissertation presents the development of Baleka II, a bipedal robot designed to overcome these challenges by achieving rapid legged locomotion through open-loop control. Building upon its predecessor, this research seeks to evaluate the robot's capacity to perform agile tasks by incorporating trajectory optimization algorithms and conducting real-world experiments.

The study is structured around four primary objectives: improving the embedded system configuration, generating control trajectories using trajectory optimization, validating these solutions through simulations, and implementing them on the physical robot. The key locomotive tasks investigated include acceleration, deceleration (gait termination), and steady-state walking/running.

The control system was implemented using the Speedgoat Real-Time Target Machine, integrating Simulink Real-Time and Simscape Multibody for real-time execution. Trajectory optimization was accomplished using Pyomo and Interior Point Optimizer (IPOPT), producing solutions for walking (0.5 m/s), walk-to-run transitions (1.5 m/s), and maximum forward speeds (4.0 m/s). Simulations were used to verify these solutions, taking into account the robot's physical constraints. Despite the use of open-loop control, stability was maintained through proportional-derivative (PD) controllers for each motor.

The key findings of this research indicate that as the robot's speed increased, so did the actuation effort, peak torque, and GRFs, leading to velocity discrepancies and high deceleration upon ground contact. Nevertheless, Baleka II was able to accelerate into 3.2 m/s steady-state gait and decelerate in a stable manner, demonstrating competitive acceleration and deceleration rates relative to other bipedal robots. These results offer valuable insights into the use of open-loop optimal control for achieving rapid transitions in bipedal robots, with potential applications in search and rescue, industrial assistance, and entertainment.

Future work will focus on enhancing the robot's deceleration capabilities, integrating additional sensors, exploring advanced control techniques, and testing the robot on uneven terrain. These efforts will further expand the potential of Baleka II for real-world applications.

Acknowledgements

I would like to express my sincere gratitude to my supervisors, Dr. Amir Patel and Dr. Stacey Shield, for their thorough involvement, unwavering support, insightful guidance, and encouragement throughout the development of this dissertation. Their expertise in the field of robotics provided invaluable feedback, which significantly contributed to the success of this research. I am particularly grateful for the regular catch-up and check-up meetings, which offered motivational boosts during the course of the project.

I am also deeply thankful to the members of the African Robotics Unit for providing the necessary resources and knowledge to facilitate this research. Special thanks to Mr. Christopher Mailer and Mr. Daryn Bright for their unwavering support in troubleshooting and assisting with lab experiments, as well as their contributions to the development of the boom and Baleka II.

My heartfelt appreciation goes to my wife, Natheerah, my parents, Gamza and Zuleiga, and the rest of my family for their endless support, patience, and encouragement throughout this journey. Their unwavering belief in me and constant motivation were essential, especially during the most challenging moments.

Lastly, I would like to express my gratitude to the Masakh'iSizwe Bursary Programme for granting me the unique opportunity to pursue my studies and for funding my research. I sincerely thank all those who have supported me throughout this endeavour.

Declaration

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own. This thesis/dissertation has been submitted to the Turnitin module (or equivalent similarity and originality checking software) and I confirm that my supervisor has seen my report and any concerns revealed by such have been resolved with my supervisor.

1. I know that plagiarism is wrong. Plagiarism is to use another's work and pretend that it is one's own.
2. I have used the IEEE convention for citation and referencing. Each contribution to, and quotation in, this report from the work(s) of other people has been attributed, and has been cited and referenced.
3. This report is my own work.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as their own work or part thereof.

Zubair Martin
Signed by candidate

Signed by candidate

27 April 2025

Copyright Notice

Contents

List of Figures

List of Tables

Nomenclature

$\ddot{}$ Acceleration of variable

$\dot{}$ Velocity of variable

$\mathbf{\Delta_A}$ Manipulator equation: Deflection of foot A

$\mathbf{\Delta_B}$ Manipulator equation: Deflection of foot B

$\mathbf{\Delta}$ Hertz: Deformation position (foot)

$\mathbf{\dot{\Theta}_{NoLoad}}$ No load angular velocity

$\mathbf{\dot{\Theta}_{Trated}}$ Maximum angular velocity at the rated torque of the upper link

$\mathbf{\Lambda}$ Manipulator equation: External forces

$\mathbf{\Phi}$ Impedance control: Angle (between the virtual leg's radial plane [r] and vertical plane [Y])

$\mathbf{\Theta_{boom}}$ Manipulator equation: Boom angle between boom linkage relative to the ground

$\mathbf{\Theta_{KneeAngleLeftA}}$ Left knee angle of leg A

$\mathbf{\Theta_{KneeAngleLeftB}}$ Left knee angle of leg B

$\mathbf{\Theta_{KneeAngleRightA}}$ Right knee angle of leg A

$\mathbf{\Theta_{KneeAngleRightB}}$ Right knee angle of leg B

$\mathbf{\Theta_{KneeAngle}}$ Smallest maximum knee angle experienced

$\mathbf{\Theta_{LLA}}$ Manipulator equation: Motor angle: lower left link of leg A

$\mathbf{\Theta_{LLB}}$ Manipulator equation: Motor angle: lower left link of leg B

$\mathbf{\Theta_{LRA}}$ Manipulator equation: Motor angle: lower right link of leg A

$\mathbf{\Theta_{LRB}}$ Manipulator equation: Motor angle: lower right link of leg B

$\mathbf{\Theta_{Mes}}$ PD Controller: Measured angular position

$\mathbf{\Theta_{MKA}}$ Maximum knee angle

$\mathbf{\Theta_{ULA}}$ Manipulator equation: Motor angle: upper left link of leg A

$\mathbf{\Theta_{ULB}}$ Manipulator equation: Motor angle: upper left link of leg B

$\mathbf{\Theta_{URA}}$ Manipulator equation: Motor angle: upper right link of leg A

$\mathbf{\Theta_{URB}}$ Manipulator equation: Motor angle: upper right link of leg B

$\mathbf{a_t}$ Tangential acceleration at the CoM (of the combined boom-body system)

$\mathbf{A}$ Angular acceleration of the boom in the pitch axis

$\mathbf{A}$ Manipulator equation: Mapping of external forces to generalized coordinates

$\mathbf{B}$ Manipulator equation: Mapped the applied/control forces and torques generalized coordinates

$\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ Manipulator equation: Coriolis matrix

$\mathbf{CM}$ Butcher tableau coefficient matrix

$\mathbf{ConnectPoint_{LeftX}}$ Connection ankle point from left link in the X-direction

$\mathbf{ConnectPoint_{LeftY}}$ Connection ankle point from left link in the Y-direction

$\mathbf{ConnectPoint_{RightX}}$ Connection ankle point from right link in the X-direction

$\mathbf{ConnectPoint_{RightY}}$ Connection ankle point from right link in the Y-direction

$\mathbf{F_\Phi}$ Impedance control: angular force

$\mathbf{F_a}$ Force control: Actuator force command

$\mathbf{F_{normal}}$ Hertz: Normal force

$\mathbf{F_r}$ Impedance control: radial force

$\mathbf{F_y}$ Force Sensor: Force sensed in the Y-direction

$\mathbf{Fc_{XLA}}$ Connection Force: X-direction, left link of leg A

$\mathbf{Fc_{XLB}}$ Connection Force: X-direction, left link of leg B

$\mathbf{Fc_{XRA}}$ Connection Force: X-direction, right link of leg A

$\mathbf{Fc_{XRB}}$ Connection Force: X-direction, right link of leg B

$\mathbf{Fc_{YLA}}$ Connection Force: Y-direction, left link of leg A

$\mathbf{Fc_{YLB}}$ Connection Force: Y-direction, left link of leg B

$\mathbf{Fc_{YRA}}$ Connection Force: Y-direction, right link of leg A

$\mathbf{Fc_{YRB}}$ Connection Force: Y-direction, right link of leg B

$\mathbf{Fc}$ Manipulator equation: Connection force between the lower left and right links

$\mathbf{F}$ Force control: Force (desired or measured)

$\mathbf{GRF_{XA}}$ GRF: X-direction of leg A

$\mathbf{GRF_{XB}}$ GRF: X-direction of leg B

$\mathbf{GRF_{YA}}$ GRF: Y-direction of leg A

$\mathbf{GRF_{YB}}$ GRF: Y-direction of leg B

$\mathbf{G}$ Manipulator equation: Gravitational potential matrix

$\mathbf{i_m}$ Torque control: Motor current

$\mathbf{J}$ Jacobian

$\mathbf{M(q)}$ Manipulator equation: Mass matrix

$\mathbf{PF_Y}$ Foot position in the Y-direction

$\mathbf{q}$ Manipulator equation: generalized coordinates

$\mathbf{R}$ Manipulator equation: Effect of the damper

$\mathbf{r}$ Impedance control: Radial distance (from body centre to foot) - end effector position

$\mathbf{T_{Des}}$ Bounds of the desired torque limit

$\mathbf{T_{ff}}$ PD Controller: Reference torque into FOC controller

$\mathbf{T_m}$ Torque control: Motor torque

$\mathbf{T_{peak}}$ Desired peak torque limit

$\mathbf{T_P}$ Peak torque observed from Trajectory Optimization simulation

$\mathbf{T_{rated}}$ Rated Torque

$\mathbf{T_{ULA}}$ Torque: upper left link of leg A

$\mathbf{T_{ULB}}$ Torque: upper left link of leg B

$\mathbf{T_{URA}}$ Torque: upper right link of leg A

$\mathbf{T_{URB}}$ Torque: upper right link of leg B

$\mathbf{T_U}$ Torque actuator input

$\mathbf{u}$ Manipulator equation: Applied/control forces and torques

$\mathbf{V_{FootX}}$ Tangential foot velocity in the X-direction

$\mathbf{X_b}$ State: Translational movement body position in the X-direction

$\mathbf{Y_b}$ State: Translational movement body position in the Y-direction

μ Coefficient of friction

μ_d Coefficient of dynamic friction

μ_s Coefficient of static friction

$_{\text{DES}}$ Desired value of variable

$_{\text{MES}}$ Measured value of variable

ρ Weight scaling factor in cost function

D_{Foot} Hertz: Damping coefficient

FN Froude number

g Gravitational acceleration in the Y direction

GP Ground penalty variable

GRF Ground reaction forces

h Variable time-step

hm Scaled by the total number of element intervals

I_{rotor} Manipulator equation: Inertia of the motor's rotor

id_{ref} FOC: Direct current

iq_{ref} FOC: Quadrature current

J_{GP} Cost function of ground penalty

J_T Cost function of torque penalty

K Total number of collocation points

K_{Foot} Hertz: Stiffness coefficient

Kd PD Controller: Tunable differential gain for velocity loop

Kd_Φ Impedance control: Tunable differential gain in the angular direction

Kd_r Impedance control: Tunable differential gain in the radial direction

KE_{motors} Manipulator equation: Rotational kinetic energy of the motors

Kp Force control: Tunable gain force control gain

Kp PD Controller: Tunable proportional gain for the position loop

Kp_Φ Impedance control: Tunable proportional gain in the angular direction

Kp_r Impedance control: Tunable proportional gain in the radial direction

Kt Torque control: Torque constant

N Represents the total number of elements/nodes

n Specific node

N_{GR} Gear reduction ratio

p Collocation point

r_b Distance between the boom-base to body

r_{max} Froude Number: Maximum leg length of the biped

r_{wb} Distance between the boom-base to the CoM of the combined boom-robot

$t[N]$ Total simulation time

TM Total mass of Baleka

TT Desired total simulation time

BLDC Brushless DC

BWE Backward Euler

CAN Controller Area Network

CoM Centre of Mass

DF Diversity Factor

DoF Degrees of Freedom

EoM Equations of Motion

FN Froude Number

FOC Field Oriented Control

GRF Ground Reaction Forces

IPOPT Interior Point Optimizer

MoI Moment of Inertia

MPC Model Predictive Control

N Node Points

NLP Non-linear Programming

PD Proportional-Derivative

QDD Quasi-direct Drive

RL Reinforcement Learning

SLIP Spring-loaded Inverted Pendulum

TO Trajectory Optimization

1 Introduction

1.1 Overview

Advancements in robotics over recent decades have driven the development of new robots and control methods to address public service, industrial (e.g., construction), and specialized (e.g., emergency rescue) needs [1]. Ground locomotion in the field of robotics typically employs legged, wheeled, or tracked mechanisms. Among these, legged locomotion, inspired by biological evolution, excels in navigating uneven terrains and executing agile, manoeuvrable motions — capabilities that wheeled and tracked robots often lack [2]. Consequently, the feasibility of legged robots capable of versatile and dynamic manoeuvres, such as MIT Cheetah 3, MABEL, ATLAS, and Cassie, has been extensively studied [3], [4], [5], [6].

However, most legged robots focus on high-speed, steady-state motions rather than transient movements (e.g. acceleration and deceleration) due to the inherent complexity of their control. Legged robots are typically under-actuated, non-linear, and high-dimensional systems, making their control tasks computationally challenging [7]. Despite these challenges, studying transient motions is crucial to enhancing legged robots' capabilities, thereby advancing their utility across public, industrial, and specialized applications.

1.2 Motivation and Background

The evolutionary development of animal morphology and behaviour is shaped by constraints and functional requirements essential for species survival. Stability, agility, and manoeuvrability are fundamental to animal locomotion, playing a critical role in both catching prey and evading predators, and remain vital to survival."

Nature has long inspired engineering, but in recent years, robotics has shifted its focus toward computer science, mathematics, and mechanics rather than biology [8]. This shift is largely due to the strong theoretical foundations of these disciplines and the difficulty in deciphering the complex evolutionary mechanisms behind animal autonomy. According to [8], bio-mimicry fields seeks to replicate maximum animal features in robots, even if their functions are not fully understood, while bio-inspired robotics takes a more measured approach, incorporating well-understood biological traits to enhance robotic performance. Despite these advancements, modern bipedal robots still lack the agility demonstrated by legged animals.

Key limitations in bio-mimicry and bio-inspired robotics include their complexity of control systems, as well as constraints in actuator and mechanism technologies that define dynamic and kinematic capabilities [9]. Although these challenges hin-

der robots from achieving the agility and manoeuvrability of their natural counterparts, current legged robots demonstrate remarkable capabilities. For instance, ATLAS can navigate varied terrains [5], Cassie achieves running speeds exceeding 4.0 m/s [10], and the MIT Cheetah 3 can climb stairs [3]. These advancements rely on diverse control architectures, including Model Predictive Control (MPC), Reinforcement Learning (RL), Force and Torque Control, Trajectory Optimization (TO), and hybrid approaches. However, achieving agile manoeuvres remains a significant challenge.

Research on agile manoeuvres, particularly rapid acceleration and deceleration, remains limited. Some studies explore gradual changes in velocity or transitions between rest and motion [11], [12]. Others use RL and MPC for gait transitions in quadruped robots, albeit at low speeds [13], [14]. TO has recently gained attention for studying transient motions, identifying optimal parameters for bipedal and quadruped robots, and analysing appendage contributions in rapid gait transitions [15], [16]. However, these investigations are primarily simulation-based due to the open-loop nature of TO, highlighting the need for further exploration of TO-based controllers for rapid transient motions in bipedal robotics.

This study builds on the foundations of Baleka I, which focused on optimizing morphology and vertical agility in a bipedal robot [17]. This project aims to extend these findings by implementing steady-state running and rapid transient motions, advancing the capabilities of the initial design.

1.3 Aim and Objectives of Research

This research extends the foundational work of Mr. A. Blom's thesis, "Design of a Bipedal Robot for Rapid Acceleration and Braking Manoeuvres" [17]. Although the initial study focused on the construction of a bipedal robot capable of agile movements, specifically hopping, this research aims to develop an integrated framework to transition from simulated models to real-world controller testing on the physical robot. It further seeks to utilize TO to perform various locomotive tasks, including acceleration, deceleration, walking, walk-to-run transition, and running at the maximum capable forward velocity.

The research began with the development of Baleka II, an improved iteration of the Baleka robot. Mr. Daryn Bright, an MSc Mechanical Engineering graduate, was responsible for the design, modelling (SolidWorks), and procurement of components for the robot's body and legs. Concurrently, Mr. Christopher Mailer, also an MSc Mechanical Engineering graduate, designed, modelled, procured, and assembled the robot's boom. My contribution focuses on control implementation and embedded systems, including robot modelling and deploying control templates on the physical

platform using real-time operating systems. The following objectives outline the key milestones necessary to achieve the project's aim:

1. **Develop an Enhanced Embedded System Configuration**
   Design an improved embedded system to seamlessly execute control commands and monitor sensor data, enabling rapid control prototyping.
2. **Model and Optimize Locomotion Tasks Using TO**
   Solve optimal locomotion tasks focusing on acceleration, deceleration, and three steady-state gaits: walking, running (walk-to-run transition), and maximum forward velocity. Additionally, determine the shortest times for accelerating and decelerating to the maximum velocity.
3. **Validate Trajectories Through Simulations**
   Create a physical simulated model to validate and assess the feasibility of the optimal trajectories.
4. **Implement and Compare on Physical Robot**
   Deploy the optimal trajectories on the physical robotic platform and compare the real-time experimental results with the TO predictions and physical simulations.

The outcomes of this research will serve as a foundation for future iterations of the Baleka robot, with a focus on developing energy-efficient locomotion, enabling complex transient manoeuvres (such as turning), enhancing mechanical properties, increasing degrees of freedom (DoF), integrating appendages to improve stability, and advancing the development of both open- and closed-loop controllers. Additionally, this study will contribute to the field of rapid transient motion control, specifically in the implementation of such controllers on physical robotic platforms.

1.4 Scope and Limitations

The scope of this research is to evaluate the implementation of agile locomotive tasks on the bipedal robot Baleka II. It involves creating a pipeline from simulation (using a TO template and a physical model simulation template) to implementing controller templates on the physical robot.

The study is expected to be completed within 24 months, with strategies in place to expedite progress, including collaboration with other MSc students working on similar projects. Before implementing the controller templates, the robot and its support rig (boom) required detailed design and assembly.

The physical limitations of Baleka II stemmed from its design, which was based on the initial iteration of Baleka. The fixed length of the boom restricted lateral

translation and yaw and roll rotations, limiting the robot's movement to a spherical path centered at the boom's base. This constraint allowed motion only along the Y-axis (up and down) and X-axis (left and right), preventing operation outside the laboratory. Additionally, the robot lacked sensors to monitor its motion, relying solely on sensors within the motor controller and the boom to track the robot's vertical, horizontal, and angular position and velocity relative to the lateral plane. Although capable, including additional rotation to increase the DoF would have added further complexity to the project.

Software limitations were influenced by several factors, detailed in the sections below, including TO constraints, cost functions, parameters such as friction, number of nodes, motor model, and the modelling of dynamics and kinematics. Additionally, the physical model simulation employed variable time-steps instead of fixed time-step solvers to optimize solving time, addressing the limitation of processor speed.

Cost limitations were significant due to the motors being sourced from China, resulting in considerable import time and expenses. Prompt procurement at the project's outset mitigated this issue. Additionally, the motor drivers were nearing obsolescence, posing a substantial risk to the project's success if they failed. The robot's structure was primarily made of aluminium, chosen for its cost-effectiveness despite being heavier than carbon fibre. From a management perspective, the reliance on a single boom, one real-time operating machine, and a shared supply unit with two other robots required effective communication and scheduling to manage these constraints.

1.5 Evolution from Baleka to Baleka II

This section provides an overview of the mechanical improvements made from the first iteration of Baleka (which was outside the scope of this work). The objective was to preserve the optimized morphology while enhancing mechanical properties to enable agile locomotion. The robot's main design includes four torque actuators at the hips and a body connected to a boom. In addition to the onboard motor controller, the boom houses all necessary sensors to monitor robot data. The motors are powered by a fixed-location power supply unit. Figure 1 illustrates the initial and updated iterations of Baleka.

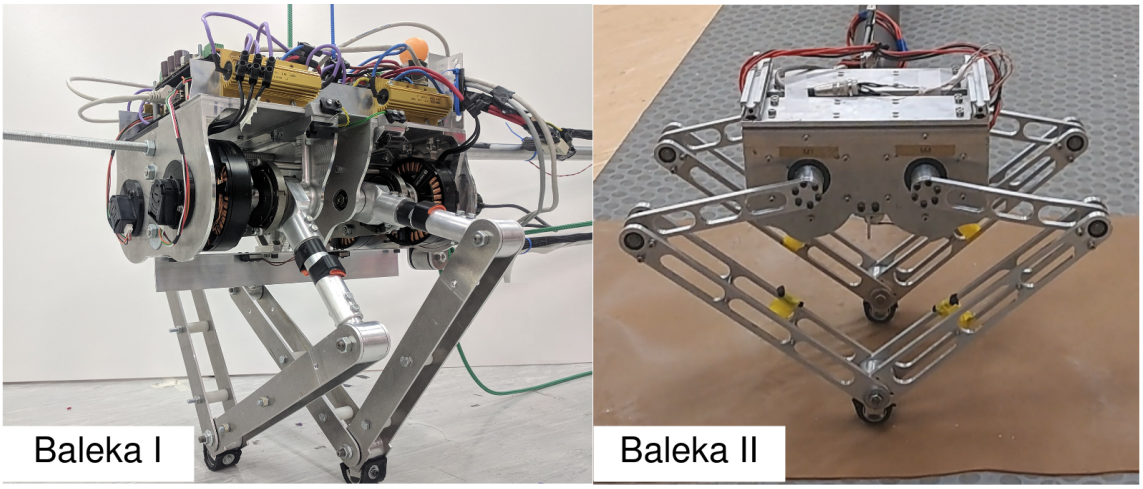

Figure 1: Baleka (left), the initial prototype, and Baleka II (right), the second iteration of the bipedal robot platform.

The force output was a considerable improvement factor in the robot's performance: as forward velocity increases, stride frequency also increases, resulting in higher contact forces over a short duration [18]. Although motor actuators face a trade-off between torque output and speed, it is preferable to achieve a high mass-specific force (force output per kilogram of robot mass) [17]. Reducing the overall mass lowers leg inertia, which benefits high-speed movement and reduces torque requirements. To enhance force output, the mass of Baleka II was reduced to 9.5 kg from the original 14.1 kg by minimizing the mass of the motor-driver and embedded system. Although [17] suggests that the total leg mass should be 10% of the body mass based on a spring-load inverted pendulum (SLIP) model, the legs were designed with increased thickness and mass to improve structural integrity, durability, and mass distribution for tasks involving high GRFs.

Another factor was active leg compliance, aimed at aiding ground impacts through torque actuation. Unlike the previous iteration, the use of series and parallel elastic actuators was not considered during the design, as no specific motions required the construction of an elastic component. The investigation of elastic components would require a separate study to determine their potential benefits or drawbacks for rapid manoeuvres.

As with the original Baleka, quasi-direct drive (QDD) motors were adopted. Direct drive motors, which operate without a gearbox, perform well in high-torque, low-speed environments but suffer from significant Joule heating and operate well below peak power output and efficiency, which are typically reached at no-load speeds [19]. In contrast, geared motors with high reduction ratios result in torque at the load being influenced by ${N_{GR}}^2$ of the motor's inertia, where N_{GR} is the gear reduction

ratio [17]. While these systems provide adequate power, motor-gearbox mechanisms are not ideal, as many control techniques rely on backdrivability.

QDD motors combine the principles of direct-drive and geared actuation by using a torque motor with a low gear reduction ratio (less than 10:1) [20]. This configuration enhances torque while minimizing backlash and maintaining high efficiency, enabling accurate torque measurement and effective backdriving [21]. QDD motors offer better efficiency and heat management compared to high-ratio geared systems, making them ideal for dynamic robotics. Notable examples include the MIT Mini Cheetah (6:1 gear reduction) and the Stanford Doggo (3:1 gear reduction), both benefiting from QDD's advantages. To facilitate active compliance, Baleka II incorporated four QDD brushless DC (BLDC) motors, each with a 9:1 gear reduction. These motors were improved over the initial iteration, reducing mass to 820 g (including driver boards) and increasing rated torque from 7.3 Nm to 15 Nm.

The following list highlights further improvements made from the first iteration:

1. Designed to operate on the boom without additional support cables.
2. Mechanical parts were cheaper to manufacture (using sheet metal instead of computer numerical control manufacturing).
3. Reduced exposed electronics/circuitry, with the driver boards and gearbox integrated within the motor casing.
4. Improved range of motion of the legs, with a wider stance, preventing body-leg collisions.
5. The endstops were omitted, eliminating the risk of body-leg collisions.
6. Modular design for easy component replacement when damaged.
7. The boom consisted of a 2.5 m long cylinder made from three 60 mm outer diameter carbon fibre tubes (1.5 mm wall thickness), compared to the original 2.1 m long 25 mm aluminium square tube with tensioned steel cables, weighing approximately 2.5 kg.

1.6 Plan of Development

This dissertation begins with a review of existing literature on legged locomotion, focusing on acceleration, deceleration, and steady-state gait cycles. It then presents an overview of common control techniques used in legged locomotion, emphasizing impedance control and TO, and highlights notable legged robots and their control methodologies.

The approach to achieving the dissertation objectives are then outlined, including tasks to investigate, the effects of the motor model and impedance control gains, and the modelling approach for the TO controller and physical simulator.

The subsequent section analyses the motor and impedance controller, detailing the experiments conducted and their outcomes. This is followed by an in-depth discussion of the TO model, covering constraints, parameters, solvers, and validation tests. The next chapter focuses on the robotic platform, detailing the physical simulator, implementation strategies for the physical robot, and results from locomotive experiments. The final chapter concludes by summarizing the results, addressing faults, proposing improvements, and suggesting future work.

2 Literature Review

In the field of locomotion, the environment is considered a fixed entity, while the natural or mechanical system applies forces to the environment, resulting in either acceleration or deceleration [22]. Significant efforts are dedicated to understanding the actuators and mechanisms that generate these forces, which in turn determine the dynamic and kinematic properties of the system. Engineers and researchers strive to investigate the optimal morphology, gait patterns, and power efficiency across various gait cycles to generate and study these forces.

Historically, engineers have drawn inspiration from nature's species in the development of robotic systems. However, in recent years, the focus of robotics has shifted towards more established fields such as computer science, mathematics, and mechanics, rather than biology [8]. The primary reason for this shift is the complexity of animal structure and function—fully replicating these aspects in a robot (bio-mimicry) is not feasible. Additionally, the foundational knowledge in these well-defined fields is more thoroughly understood. Consequently, this research prioritizes using the first iteration of Baleka's optimal design morphology, rather than attempting to incorporate the full range of animal features into the robot's design.

Although significant progress has been made in legged locomotion, dynamic locomotion remains incompletely characterized, and further contributions are necessary. In particular, when focusing on rapid transient motions, few existing robots possess the capability to accelerate from rest into steady-state motion and decelerate effectively. Thus, while this study is focused on rapid transient motions constrained to the lateral plane and conducted within a laboratory environment, it serves as a foundational step towards enabling future bipedal robots to perform dynamic locomotion in real-world settings. This research will examine the process of incorporating simulated control templates and implementing them on the physical bipedal robot (Baleka II). Various locomotive tasks will be explored, including walking, running, and rapid transient motions such as acceleration and deceleration.

2.1 Legged Locomotion in Robotics

Biology serves as inspiration for designing robots capable of operating in hazardous, unstructured, and complex environments, to operate in surveillance, planet exploration, search and rescue, endoscopic surgery, and prosthetic control applications [8], [23].

In designing robots, it is essential to compare their morphology to that of animals suited for the desired application. This is particularly important for agile and mobile robots operating in unstructured environments, where selecting the optimal locomo-

tion mode is crucial for performance [24]. Common categories of terrestrial locomotion include two-wheeled, three/four-wheeled, bipedal, quadrupedal, wheeled stair-climbing, and tracked robots [25]. Although wheeled robots perform well on even ground, they struggle on uneven terrain and stairs due to slipping [25], [24]. In contrast, bipedal robots excel in narrow spaces, rough terrains, and obstacle avoidance, making them ideal for environments with gaps, slopes, or stairs [26], [25]. Additionally, legged robots are more energy-efficient on soft surfaces due to discrete ground deformation, whereas wheeled robots cause continuous deformation [24]. Legged robots also benefit from adjustable ground pressure through sole width selection, unlike wheeled robots with a small contact area [24].

Legged locomotion is a key focus in bio-inspired robotics, as it enables navigation of complex, unstructured terrains [27]. Minimizing the number of legs reduces the actuator count and robot weight, making two-legged robots preferred for agility, cost-effectiveness, and weight reduction [24]. Slow, stable robots often have more than four legs to maintain static stability, while two-legged robots are favoured for dynamic stability and locomotion.

Humanoid robots (bipedal) must avoid causing harm when interacting with the environment, which requires flexible locomotion with the ability to launch (accelerate) and stop abruptly (decelerate). Bipedal robots have superior braking power compared to wheeled robots [25], enabling agile manoeuvres. Real-time solutions, such as Zero Moment Point (ZMP) modification, allow walking robots to stop in a stable manner [28], [29]. Legged robots also offer better manoeuvrability on rugged terrain due to discrete ground contact points [27]. However, achieving high manoeuvrability requires complex control to maintain stability and sufficient DoF in the legs to generate forces in various directions.

For fast-moving legged animals and robots, various locomotion patterns (gaits) emerge, including walking, running, trotting, bounding, and galloping [13]. This study focuses on bipedal robots, specifically Baleka II, to explore methods for stable walking and running gaits, gait initiation, and termination, contributing to the stability of bipedal manoeuvres and the success of robots in dynamic environments.

2.1.1 Walking and Running

A significant body of research has focused on the steady-state locomotion of legged robotics, particularly walking and running [4], [10]. Studies show that gait changes optimize energy expenditure at various speeds, with running being more energy-efficient than walking at higher speeds [30]. For bipeds, walking is a stable gait where at least one foot is in contact with the ground at all times, while running involves flight phases where no foot is on the ground.

The forward velocity of a gait depends on the step length and cadence (steps per minute). Transitioning from walking to running is governed by the walk-to-run transition factor, which is linked to the Froude Number (FN) — a dimensionless ratio of inertial to gravitational force. During locomotion, centripetal force must equal or exceed the gravitational force to keep the centre of mass (CoM) within the circular arc of travel. When the centripetal force exceeds the gravitational force, the robot enters the flight phase [31]. The FN, which incorporates leg length and velocity, further explains the walk-to-run transition, as shown in Equation 1 [31].

$$FN = \frac{\dot{\mathbf{X}}_{\mathbf{b}}^2}{\mathbf{g} \times \mathbf{r_{max}}} \tag{1}$$

The FN is calculated using Equation 1, which incorporates $\dot{\mathbf{X}}_{\mathbf{b}}$, the forward velocity of the body in the X direction, $\mathbf{g}$, the gravitational acceleration, and $\mathbf{r_{max}}$, the maximum leg length of the bipedal robot. Typically, a FN greater than 0.5 indicates a transition from walking to running in bipeds [31]. Therefore, this study will examine walking and running locomotion for Baleka II, using the FN as a reference to categorize the locomotion task.

2.1.2 Gait Transitions

Steady-state locomotion in terrestrial animals and humans involves manoeuvres such as starts, stops, and turns. The ability to perform rapid, agile manoeuvres is essential for survival, such as escaping predators, catching prey, and avoiding obstacles [32]. These manoeuvres include acceleration, deceleration, direction changes, and jumping/hopping. Studying the traits of animals performing such manoeuvres is challenging due to the diversity in morphology across species. Manoeuvres are influenced by the interaction between environment, instinct, and morphology [32]. Integrating biomechanics findings promotes advancements in bio-inspired motor control models [33], fostering the development of robotics beyond steady-state motion.

Manoeuvrability, defined as the ability to change whole-body motion (direction or position), is characterized by changes in velocity vectors, including direction, translational velocity, or angular velocity [34]. It focuses on controlled instabilities that enable starting, changing direction, and stopping [35]. Furthermore, manoeuvres can be classified as either simple (rotation or translation) or complex (combining both) [34]. This study focuses on acceleration from rest and deceleration to rest.

A study of cheetahs, tracked with GPS and inertial measurement units over 17 months, found that successful hunts relied on accelerations exceeding 13 m/s^2 and decelerations of -7.5 m/s^2, due to high friction coefficients between the feet and the

ground [36]. Despite numerous studies on rapid transitions in nature, legged robots have yet to master such capabilities.

The difficulty in developing rapid gait transitions stems from the complexity of discontinuous mechanics, such as aperiodic foot contact orders. Simplified models (e.g. massless legs, point mass bodies, and infinite friction) are intractable in real-world systems [16].

2.1.3 Acceleration: Gait Initialization

Despite limited studies on rapid transition motions for bipedal legged robots, recent research has focused on optimizing acceleration methods for such robots [16]. Key approaches include:

1. Impedance control by incrementally increasing the desired velocity [11],
2. TO to determine the optimal motion for accelerating a legged robot from rest [16],
3. RL and MPC to study if gait-switching behaviours naturally emerge during learning [13]

Although acceleration was not a primary focus in the original Raibert bipedal development, it was incorporated through incremental increases in the desired forward body velocity, with dynamic stability as the main goal [11]. This method, however, struggles with high-speed transitions from rest due to its reliance on a SLIP model, not specifically designed for acceleration.

The TO approach involved modelling both bipedal and quadrupedal robots to optimize gaits for acceleration, steady-state, and deceleration motions over a long time horizon [16]. These motions were then stitched together into a complete cycle from rest to rest, with slip effects and a sliding mass template emerging in velocity analysis.

Another study combined RL and MPC to optimize gait transition contact scheduling in quadrupeds, focusing on energy efficiency. This approach, validated on a Unitree A1 robot, successfully demonstrated gait transitioning at various speeds and provided energy-efficient solutions [13].

Figure 2 shows the velocity curves for these approaches. Notably, the TO approach achieved faster response times, even at higher speeds. A useful metric for evaluating acceleration in legged robots is the time taken to reach the desired velocity.

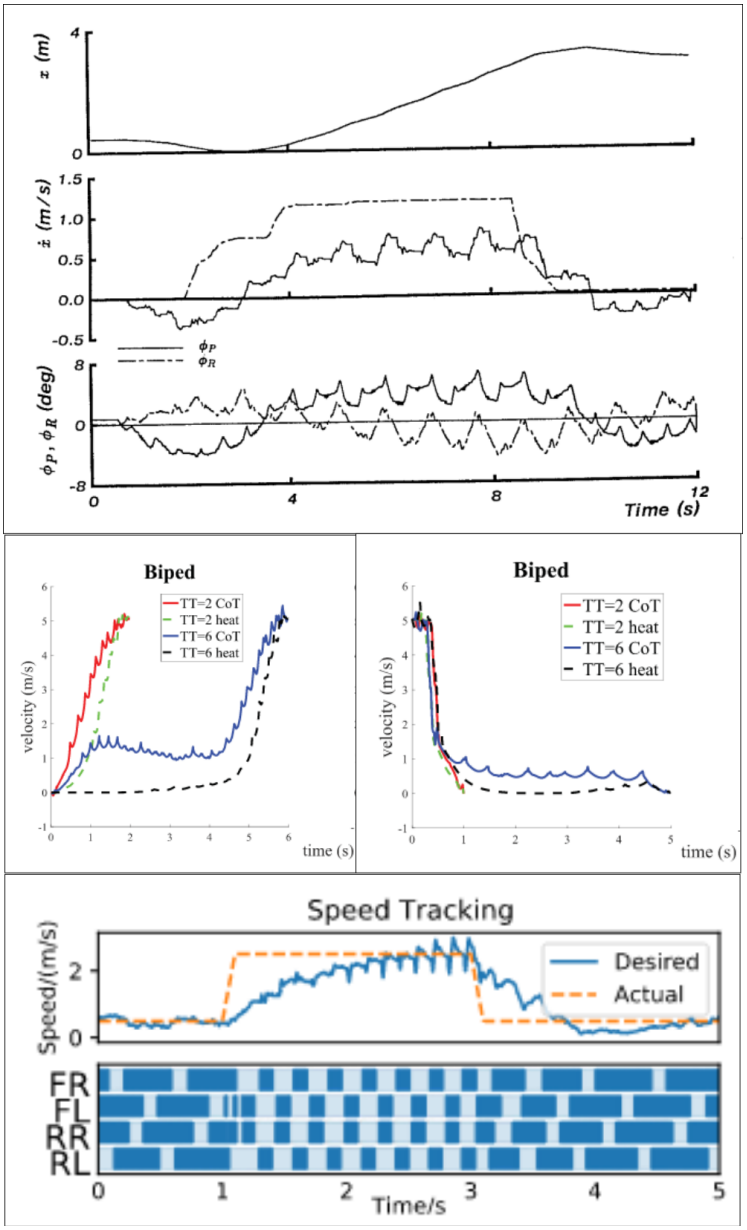

Figure 2: The figure compares Raibert's quadruped (top, dashed curve: desired body velocity) [11], velocity curves for acceleration states under various time constraints using TO (middle) [16], and the RL-MPC approach for gait transitioning with velocity changes (bottom).

2.1.4 Deceleration: Gait Termination

Another agile manoeuvre under investigation is gait termination, a common but under-explored aspect of bipedal locomotion [37], [38]. Understanding its dynamics

and kinematics is crucial for humans and robots, to prevent injuries and to avoid obstacles.

Raibert's robots addressed deceleration by placing the foot ahead of the neutral point, where no net forward acceleration occurs [11]. In human physiology, studies have shown that body control strategies correlate with feedback rules used by Raibert's robots [39]. Alternatively, TO techniques have been applied to identify optimal deceleration strategies for both bipedal and quadrupedal systems [16].

Research indicates that ground reaction forces (GRFs) during braking in bipeds exceed those generated during acceleration [38], [40]. Additionally, the forward-placed leg in gait termination is angled to produce a dominant horizontal force, contributing to instability. Similar to acceleration, key metrics for evaluation include deceleration speed and the body's ability to maintain post-manoeuvre stability.

The stability of a system refers to its ability to self-correct in response to disturbances, thereby maintaining the desired posture. In legged robots, stability can be categorized into two types: static and dynamic. Static stability is achieved when the CoM lies within the support polygon, which is the geometric area enclosed by the ground contact points of the robot's body [41].

Dynamic stability, on the other hand, applies to systems such as bipedal robots. These robots must actively prevent tipping during motion, as their support polygon is limited to a rectangle (defined by the sole size) or even a point during single-leg support. The CoM in dynamically stable bipedal locomotion is challenging to predict due to its continuous shifting. However, during the deceleration phase of the gait cycle, Baleka II should maintain stability by ensuring that its CoM remains within the support polygon.

2.2 Review of Legged Robotic Platforms and Control Methodologies

Engineers draw inspiration from animal morphology, features, and dynamics when designing and controlling legged robots. The integration of morphology, sensors, actuators, materials, and platforms is crucial in robot design, driven by application needs. Robotics, studied since the 18th century, initially used mechanical linkages to enable locomotion, though rigid walking patterns limited the adaptability and terrain navigation [42]. From the 1960s, research into active control led to the development of adaptive legged machines.

In the early 1960s, General Electric Walking Truck, a 12-DoF quadrupedal robot, was hydraulically powered and manually controlled [42]. Shortly thereafter, automated machines, such as the Ohio State University hexapod and WL-9DR robot,

emerged, paving the way for adaptable walking robots [42], [43]. Together, these global advancements laid the foundation for adaptable walking robots. However, the emergence of novel designs for dynamic legged machines soon challenged the prevailing mindset of the time.

Prior to the 1980s, robotic designs focused on static stability by increasing the number of limbs or expanding foot areas. However, Marc Raibert's pioneering work in dynamic locomotion during the 1980s significantly advanced the field. By employing a SLIP model with pneumatic actuation, Raibert's robots incorporated balancing masses and constrained lateral motion. These robots demonstrated the ability to manage disturbances, perform hopping, and execute speed, position, and body attitude control. Dynamic behaviours were achieved by aligning the hip joints with the CoM and designing a high-inertia torso, which minimized the impact of leg movements on the body and prevented undesired moments on the torso caused by leg impulses [42], [11]

Since Raibert's advancements, legged robotics observed numerous developments, from humanoid robots assisting with daily tasks to agile robots running. Despite diverse designs, control systems, and sensors, these robots share the common goal of achieving or surpassing the natural agility of animals and humans, with some emphasizing energy-efficient techniques. This collective effort continues to expand the possibilities of legged robots.

While the goal is shared, different control algorithms enable legged robots to perform dynamic manoeuvres. This section will explore Model-Based, Learning-Based, Bio-Inspired, and Heuristic-Based control methods in existing robots.

2.2.1 Raibert Control

Raibert's control approach for legged robots is centred around dynamic balance, utilizing a state machine to manage different phases of motion. The approach distinguishes between hopping, forward speed, and body attitude control, each implemented through separate control techniques [11]. A key feature of this approach is the concept of a neutral point, which defines the foot placement that results in zero net horizontal acceleration. To increase forward speed, the foot is placed ahead of the neutral point, whereas placing the foot behind it serves to reduce speed. The method also incorporates impedance control to regulate the body pitch and forward speed, ensuring these values meet the desired targets. The control focuses on leg thrust, swing control, leg placement, body attitude, and the associated velocities.

The robot's movement is categorized into four distinct phases: foot touchdown, lowest body height during motion, foot lift-off, and body apex during motion. The key control states associated with these phases are as follows:

1. **Compression State** (between touchdown and lowest body height): This state is activated when the leg shortens, exerting no actuation, except for adjusting the body via the hip.
2. **Thrust State** (between bottom and lift-off): Activated when the leg reaches the lowest compression length or when the body's vertical velocity is zero. This state exerts actuation on both legs for launch and adjusts the body via the hip.
3. **Unloading State** (at lift-off): Occurs when the leg stretches to its full length, exerting no actuation.
4. **Flight State** (between lift-off and touchdown): In this state, the foot is off the ground, and the leg is actuated to position it for landing.

Raibert developed both monopodal and bipedal robots using pneumatic cylinder actuation (see Figure 3). The primary distinction between the two robots lies in the biped's additional control states, which are necessary to coordinate the retraction and extension of each leg.

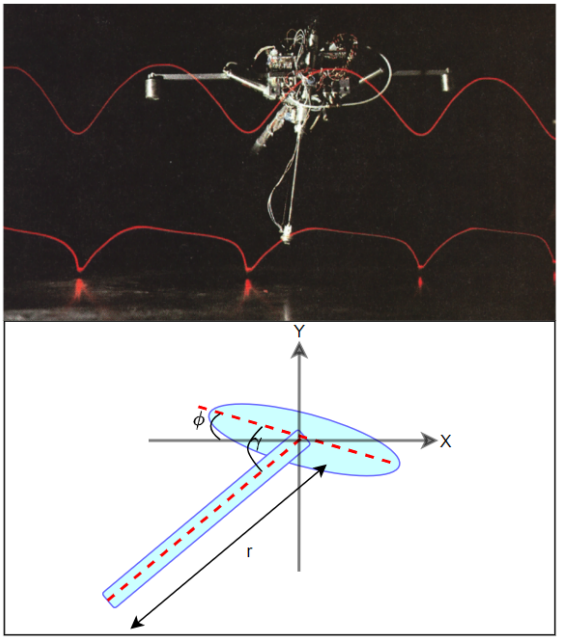

Figure 3: Display image of Raibert Monopod Hopper of both the experiment and model [11].

While this approach is effective for dynamic motion and provides stable feedback, it has several limitations [11]:

1. The approach prioritizes balanced motions over optimal control.
2. The equations for forward speed are suited for low-speed applications with nearly vertical, stiff legs, becoming less accurate at higher speeds. The **compression state** reduces speed before acceleration.
3. In the **compression state**, the foot remains motionless, failing to exert a net negative acceleration to launch the body.
4. These limitations hinder rapid transitions from rest, as large leg angles and lack of acceleration reduce launch capability, as shown in Figure 2.

2.2.2 Force Control and Torque Control

Force control applies forces and torques to the environment, based on sensor data from force/torque sensors or encoders tracking joint states [44]. The actuators adjust to achieve the desired force output, as described by the proportional controller in Equation 2, where $\mathbf{F_{Mes}}$ and $\mathbf{F_{Des}}$ represent measured and desired forces, respectively. The actuator command $\mathbf{F_a}$ is controlled by a tunable gain Kp. When using motors, force is typically not directly measured; instead, the torque ($\mathbf{T_m}$) is estimated from motor current ($\mathbf{i_m}$) and torque constant (Kt) using Equation 3, mapping the torque through the Jacobian $\mathbf{J^{-T}}$.

$$\mathbf{F_a} = -Kp(\mathbf{F_{Mes}} - \mathbf{F_{Des}}) \tag{2}$$

$$\begin{aligned} \mathbf{T_m} &= \mathbf{i_m} Kt \\ \mathbf{F_a} &= \mathbf{T_m} \mathbf{J^{-T}} \end{aligned} \tag{3}$$

Legged robots like the Stanford Doggo, MIT Cheetah 3, and MABEL have adopted force control for various gaits (see Figure 4). The Stanford Doggo uses QDD actuators and virtual compliance, focusing on vertical agility and using the leg Jacobian for force estimation [20]. The MIT Cheetah 3 employs a Balance Controller with proportional-derivative (PD) control to align foot forces with friction constraints, optimizing leg force distribution for robust locomotion [3]. On the other hand, the MABEL bipedal robot uses a force controller with hybrid zero dynamics during stance for dynamic locomotion. It integrates virtual compliance and springs in its drivetrain to absorb shocks and enhance agility during running [4], [45].

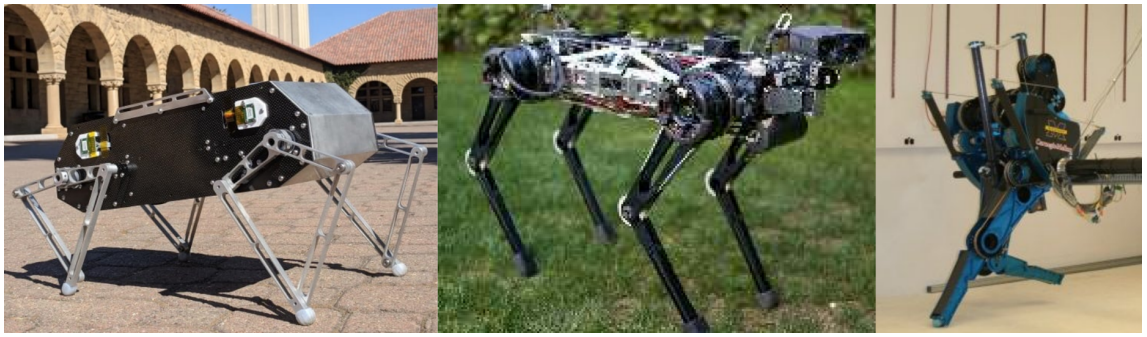

Figure 4: Legged robots which used Force control techniques to obtain desired locomotive tasks. The Stanford Doggo is on the left [20], MIT Mini Cheetah in the centre [46], and MABEL on the right [45].

2.2.3 Impedance Control

Impedance control regulates the relationship between force and motion to manage a robot's dynamic behaviour, unlike force control [44]. Instead of relying on end-effector force–torque sensors, it uses encoders, tachometers, and accelerometers to measure joint positions, velocities, and accelerations. This avoids instability from inaccurately modelled dynamics between actuators and sensors [47]. This data is used to apply force through controlled joint torques.

Equation 4 illustrates the actuation force, which depends on position and velocity errors, where $\mathbf{r_{Mes}}$ and $\mathbf{r_{Des}}$ are the measured and desired end-effector positions, and $\dot{\mathbf{r}}_{\mathbf{Mes}}$ and $\dot{\mathbf{r}}_{\mathbf{Des}}$ are their respective velocities. Kp_r and Kd_r are the position and velocity gains.

$$\mathbf{F_a} = -Kp_r(\mathbf{r_{Mes}} - \mathbf{r_{Des}}) - Kd_r(\dot{\mathbf{r}}_{\mathbf{Mes}} - \dot{\mathbf{r}}_{\mathbf{Des}}) \tag{4}$$

An impedance-controlled leg prototype for the MIT Cheetah demonstrated the approach's effectiveness in dynamic legged locomotion [47]. The robot focused on improving torque density and reducing dynamic effects from mass and friction for high-speed running. Although it only incorporated the proportional term of Equation 4 (with torque-dependent Coulomb friction), it enabled high-force impacts and fast running.

2.2.4 Model Predictive Control (MPC)

Model Predictive Control (MPC) is an online feedback control strategy that leverage a system's model to predict and optimize control inputs over a predefined horizon [48]. By minimizing a weighted cost function, MPC selects the optimal control

inputs while adhering to system constraints, classified as either soft (violable) or hard (inviolable) constraints. This adaptive controller recalculates solutions at each time step, offering benefits such as preview capability, multi-variable control, and constraint handling. While it excels in dynamic environments, its computational demand increases with the inclusion of full system dynamics and extended prediction horizons.

Linear MPC uses linearized models and solves convex quadratic online programming problems, making it computationally efficient for approximately linear systems. Conversely, non-linear MPC handles true system dynamics but requires solving non-convex optimization problems, resulting in higher computational costs [49].

MPC is widely applied in legged robots (see Figure 5) due to its ability to adapt to real-time data, reject disturbances through high control loop sample rates, and handle error accumulation where offline trajectory-generated solutions often fall short [50]. The following overview highlights notable robots that leverage MPC control for various applications.

The **ANYmal** quadrupedal robot employs a collision-free MPC strategy to navigate cluttered environments. The optimizer incorporates collision-free spaces, kinematic constraints, and dynamic feasibility into a relaxed barrier function within the cost function [51]. Experimental results showed its capacity to avoid obstacles by trotting in place for fast-moving objects (>0.5 m/s) or surpassing slower ones (<0.2 m/s). However, rapid gait transitions may enhance its responsiveness to faster objects.

The hydraulically actuated **IIT HyQ** quadruped employs a non-linear MPC approach based on a Single Rigid Body Dynamics model with a two-second prediction horizon to manage non-linear system dynamics and constraints [50]. The cost function is designed to maximize a mobility factor related to hip-to-foot positioning, facilitating stable locomotion on rough terrains. However, the model simplifies the robot's dynamics by neglecting leg inertia, making it more suitable for quadrupeds, whose mass and inertia are concentrated at the base, rather than bipedal robots. The robot achieved a maximum speed of 0.1 m/s on uneven terrain.

The **Mini Cheetah** employs convex MPC to calculate GRFs over a gait cycle, avoiding the complexity of non-linear optimization by using simplified robot dynamics as described in Equation 3 [46]. These forces are then used to compute motor torques, enabling the robot to achieve rapid trotting speeds of up to 2.45 m/s with accelerations reaching 5 m/s^2.

Atlas, a humanoid robot, uses MPC to optimize motions over a short horizon. Its cost function evaluates stability and pose, while the model maps joint-level dynamics, including momentum and forces, to assess the feasibility of motions across all links.

This enables complex tasks like parkour and object manipulation but requires costly high-performance hardware [52].

MPC is most effective for systems with fewer DoF due to its computational demands. For systems with many DoF or requiring rapid motions, solving MPC problems in real-time can become prohibitively slow, with execution times reaching up to 50 ms. [53]

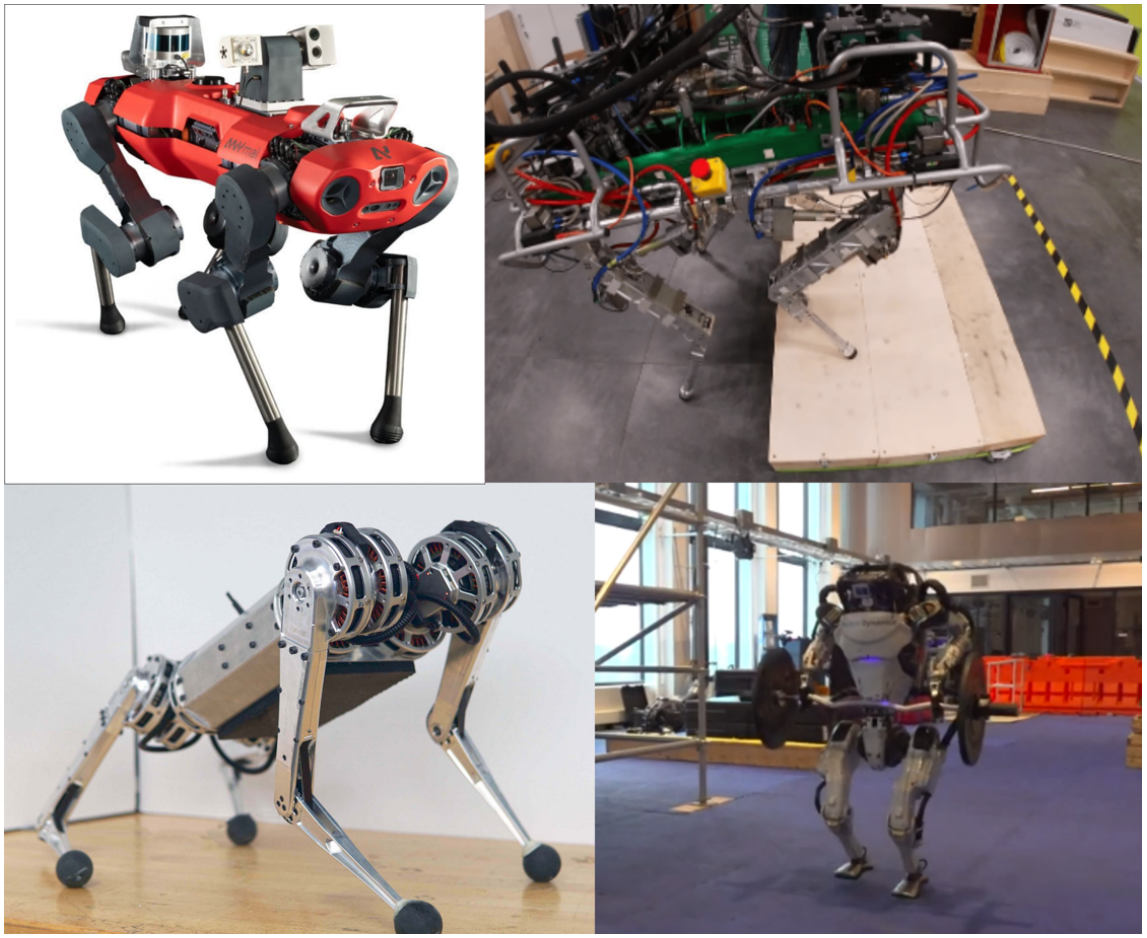

Figure 5: Display of legged robots adopting MPC, top left: ANYmal used to capture the interactions with the environment [51], top right: IIT's HyQ traversing a pallet using NMPC [50], bottom left: Mini Cheetah [46], and bottom right: Atlas lifting dumbbells [53].

2.2.5 Reinforcement Learning (RL)

Reinforcement learning (RL) is an advanced optimization tool in legged robotics, which leverage agents and Monte Carlo roll-outs to maximize reward functions [54], [55]. RL employs an exploration-exploitation paradigm to train policies for energy

efficiency and precise tracking. By utilizing full dynamic models, it can derive optimal feedback control policies based on sensor data and target states. A crucial step in achieving optimal policies involves gradually guiding deep neural networks from simple to complex tasks to address non-linearities. RL is particularly effective in locomotion, manipulation, and navigation, enabling robots to adapt to dynamic environments where traditional models often fail. For instance, RL aids in navigating varied terrains, improving robotic grasping using vision data, and enabling navigation in unknown, dynamic settings [54].

Despite its potential, relatively few platforms validate RL models on real robots (some shown in Figure 6). One example is a study involving rapid gait transitions on the quadrupedal robot Unitree A1 [13]. This study integrated optimal control (convex MPC) with RL to separate gait generation from motor control. RL developed a gait policy defining key parameters, which the convex MPC used to compute optimal motor commands, enabling a top speed of 2.5 m/s. Although limited to slow accelerations, the system handled abrupt changes effectively.

The bipedal robot Cassie adopted a control approach integrating RL for dynamic walking and running locomotion. Using a sim-to-real training framework with an actor-critic PPO algorithm, the RL controller was implemented directly on the robot [10]. Positive rewards were assigned based on gait properties, such as leg swing rate, stride frequency, and zero velocity of the foot in stance. The controller generated joint position and velocity references, executed with a PD controller operating at 2 kHz. This approach enabled Cassie to achieve a world record for the fastest 100 m dash by a bipedal robot, reaching a speed of 4 m/s. Other robots, such as the KAIST RAPTOR (12.7 m/s on a treadmill), and Unitree's H1 humanoid (3.3 m/s), have also achieved notable speed benchmarks [56], [57].

Implementing RL in robotics poses risks, particularly instability during policy exploration. Sim-to-real policy transfer is common to mitigate the risk of damaging hardware, yet this approach has limitations. Simulations often fail to represent real-world challenges accurately, and fine-tuning simulation environments is computationally expensive for complex systems. Furthermore, developing robust benchmarks for RL algorithms is challenging, as it requires careful specification of reward functions, observation spaces, and action spaces. To address model uncertainties, actuator data must be incorporated into neural networks, further increasing computational demands [55], [22].

Figure 6: Cassie [10] (left) and Unitree [13] (right) use RL to achieve their desired locomotive tasks.

2.2.6 Trajectory Optimization (TO)

In classical control theory, controller development is based on an iterative trial-and-error approach, where parameters are selected to ensure system acceptability. Acceptability is determined by characteristics such as rise time, overshoot, settling time, bandwidth, and gain and phase margins. These methods, which focus on frequency response, are effective for single-input, single-output systems. However, they face challenges in handling constraints on state or control inputs and complex multi-input, multi-output systems, such as robotics [58]. In contrast, optimal control theory, emerging from state-space control, addresses these limitations by adjusting the parameters at each time step to account for future states [59]. It determines control signals to minimize a performance criterion while satisfying physical constraints. To formulate an optimal control problem, a mathematical model, constraints, bounds, performance criteria, and desired outcomes are required [59].

TO is an offline, numerical open-loop optimal control method used for motion planning in dynamic systems, particularly in non-linear, high-dimensional, and unstable systems such as legged robots, where analytical solutions are difficult to obtain [60], [61]. This approach addresses the limitations of classical control methods and efficiently handles complex behaviours for simple cost functions [62]. The goal of TO is to minimize a cost function by determining the input control parameters and system states at each time interval, considering constraints, bounds, and initialization parameters. Constraints define the acceptable trajectory, while bounds and initialization parameters define the solution search space.

A solution is deemed feasible when all constraints and bounds are met, with optimal

solutions being those that minimize the cost function [63]. These solutions are found by adjusting decision variables, such as states, control inputs, and time, within specified bounds while adhering to constraints. The system setup includes boundary conditions, constraints, and dynamics, narrowing the search space. Dynamics are defined by the equations of motion (EoM) for Baleka II, while path constraints ensure adherence to rules like avoiding ground penetration or respecting torque-speed limits. Boundary conditions specify limitations on initial or terminal decision variables.

TO has limited hardware validation due to its offline nature, though it has been tested on platforms like Cassie, which uses TO for steady-state locomotion on various terrains by using the full body dynamics [6]. This approach models compliant multi-link mechanisms for dynamic control and feeds trajectories into a PD controller. Similarly, Atlas's previous revision used TO for step-up tasks, by specifying a foot contact order and optimizing the centroidal dynamics of the robot to reduce maximum knee joint torque by up to 20% [64].

While TO is effective for rapid locomotion in legged robots, it has several drawbacks. It is computationally expensive, relies on bounds close to optimal, and may prioritize computational efficiency over model accuracy [65]. A major limitation is the lack of feedback, leading to increasing errors during real-time implementation. Despite this, TO provides feasible motions without requiring a pose sequence or actuation profile, optimizing motion based on a desired cost function [66].

To reduce computational complexity, some methods prioritize efficiency over accuracy. For example, solving half-gait patterns for symmetric legged robots, which can be recycled for the second half of the gait, reduces solver time. Using absolute angles instead of relative angles for orientation coordinates also improves solver efficiency. Additionally, including more gait information (e.g., initialization, contact orders, termination, or specifying desired states) and reducing trajectory time (instead of using long time horizons) can further enhance solving time. Assumptions such as 2D systems, infinite friction, and actuation also help [67]. The choice of numerical solver impacts both solving time and accuracy. Given the limited use of TO in high-speed bipedal locomotion with rapid gait transitions, this study will adopt the TO approach.

The TO approach addresses complex, non-linear tasks using numerical methods. These include direct and indirect approaches [61]. Direct methods discretize the optimization problem (state and control variables) and numerically solve for optimal values, while indirect methods first construct analytical conditions, then discretize and solve them numerically.

Direct methods can be further classified into shooting and collocation methods. Di-

rect shooting methods transform the problem into a non-linear programming (NLP) formulation, where variables are determined as subsets of initial, parameter, and final conditions. This process is iteratively repeated until the selected inputs yield a feasible trajectory that satisfies the specified constraints and bounds [68]. These methods reduce the coupling between decision variables and can be either single or multiple.

The single shooting method represents the entire trajectory as a single simulation, ideal for simple control and few constraints (e.g. space flight) [63]. However, it becomes unstable with increased system complexity due to limited initialization options, making it unsuitable for legged systems requiring precise state and control initialization [69].

In contrast, direct multiple shooting methods divide the trajectory into smaller segments, treating each as a separate simulation. This approach reduces non-linearity and is more robust for complex models [63]. However, it struggles to converge due to the non-linear relationship between constraints and variables and cannot incorporate path constraints, as state variables are not decision variables.

A method known as orthogonal collocation offers an alternative approach to TO. It typically utilizes higher-order polynomials to represent system dynamics, which are constrained by non-linear conditions. This method involves discretizing the simulation interval into subintervals, transforming the continuous problem into a set of algebraic equations. A feasible solution is first identified before attempting to find a locally optimal solution. The decision variables are approximated using polynomial functions at specified collocation points, which are the roots of orthogonal polynomials such as Legendre or Chebyshev polynomials [63]. Compared to the previously discussed methods, state initialization and path constraints are more easily incorporated. This approach provides high accuracy, particularly when the number of segments is increased. In the context of legged locomotion, both multiple shooting and orthogonal collocation have been applied, with the availability of free software and introductory tutorials [63]

TO is a powerful tool for generating locally optimal trajectories for both linear and non-linear dynamic systems. Non-linear dynamical systems, such as legged robots, often perform tasks involving locomotion, which require ground impact. These collisions, characterized by the making and breaking of contact with the ground, introduce complexity in solving the optimal trajectory due to their discontinuous nature (rapid changes in velocity and large impulse forces). These collisions could be modelled as continuous reaction forces (e.g. using a spring-damper system); however, this would increase complexity by requiring small step sizes due to stiff differential equations. Alternatively, a through-contact numerical approach could be used,

employing time-steps to solve contact constraints with linear or non-linear complementarity [70].

A key aspect of trajectory generation involves ground contacts. Specifically, the motion and sequence of the legged robot require intermittent contact phases to successfully complete a gait. One advantage of using an NLP solver is that it can determine gait cycles without requiring a predefined contact order. This approach leads to an unbiased, optimal, and energy-efficient solution (if energy is included in the cost function), while ensuring that constraints and bounds are met, compared to methods that specify the contact order [71]. However, a limitation of this approach is that the solve time may increase significantly due to the large search space and the potential for poor initialization. In this study, we leverage both methods to determine the optimal walking and running gait cycle.

2.3 Summary

This study will examine three core locomotive tasks: acceleration, deceleration, and steady-state gait cycles, specifically walking and running, at both the walk-to-run transition velocity and maximum forward velocity, for a bipedal robot (Baleka II). The walking task will be investigated for steady-state motion, using the FN to identify the transition point from walking to running. Additionally, the maximum speed of the Baleka II robot will be evaluated and compared to the current world record. The control method employed in this study is TO, a relatively under-explored approach for rapid locomotive tasks. A stitching technique will be used, applying this method to each locomotion task and combining them for the overall analysis.

3 Methodology

This section outlines the approach taken to achieve the objectives of this research. First, the validation testing methods used to justify the model parameter values will be reviewed. Next, an overview of the TO approach will be provided, along with its integration into the study. Following that, the validation process on both a physical model simulator and the real-time robot will be described. Finally, key metrics for investigation and evaluation will be discussed. The overarching aim is to create a seamless pipeline from simulation to real-time implementation. This study adopts three executable phases, namely: Pyomo, Simscape Multibody, and Simulink Real-Time. The Pyomo platform was used to solve trajectories using the Interior Point Optimizer solver, while Simscape was adopted to simulate a physical model of Baleka II, and finally, Simulink was used to execute the trajectories on the physical robot.

3.1 Parameter Validation Testing Method

3.1.1 Foot Deflection Modelling

During agile manoeuvres, the contact between a legged robot's feet and the ground must maintain traction. When a robot slips, it becomes unstable, leading to errors in dynamic modelling [72]. To ensure legged robots maintain traction during contact, high coefficients of friction are required [17]. Therefore, this study will provide justification for the materials used for the feet and ground, emphasizing the coefficient of friction to be applied.

During contact, large contact forces develop over a short duration - with high energy loss and rapid accelerations causing the foot/feet and ground to deform. In this study - the normal forces are modelled using the Hertz contact force model which makes use of a spring-damper system [73]. Equation 5 represents the resulting normal force $\mathbf{F_{normal}}$ being modelled as a spring-damper system with K_{Foot} and D_{Foot} being the stiffness and damping coefficient, respectively. Furthermore, $\mathbf{\Delta}$ and $\dot{\mathbf{\Delta}}$ represents the deformation position and velocity (which occurs during contact), respectively.

$$\mathbf{F_{normal}} = K_{Foot}\mathbf{\Delta} + D_{Foot}\dot{\mathbf{\Delta}} \tag{5}$$

In order to determine K_{Foot} and D_{Foot}, the robot was placed on a force sensor (with a selected mat of specified material between the robot and sensor plate). Thereafter, 200 N was applied to the robot by the researcher's pushing force. This test was done in the robot's rest state (knees were at 0°) - therefore, any changes to the body height in the vertical direction was due to the applied force (foot deflection). A multiple

linear regression model was applied to fit the change in force to determine K_{Foot} and D_{Foot}.

3.1.2 Motor Step Testing

Another crucial aspect to the design was the motor model. Motor step tests together with manufacturer specifications are generally used to model the motor characteristics in simulation. Specifically, modelling the dynamic behaviour such as back EMF, startup torque, motor inertia and friction ensure the robust transition from simulation to real time implementation.

To compare the simulated motor model and the real time motor - the motors were modelled in Simscape. Although the above-mentioned factors are crucial, this study only investigates the motor inertia and internal PD controller within the selected motors. Each motor is equipped with an internal PD controller, which adjusts the motor's position and velocity by minimizing the state error using specified gain values. The friction and back EMF modelling was out of scope as it would have required detailed modelling and parameter estimation, which would divert attention from the core objectives of this study. Nonetheless, motor friction is often mitigated through lubrication, high-precision manufacturing, and the built in PD controller to compensate for minor frictional losses. Therefore, modelling friction was not required for the level of accuracy required in this study.

The motor model designed in Pyomo and Simscape was based on the measured moment of inertia (MoI) of the rotor and modelling a PD controller. Specifically, the rotor was accurately modelled in Solidworks - by disassembling the motor, measuring the rotor's dimensions and mass. Furthermore, a simple PD controller was incorporated taking into account the boundaries of the torque-speed curve represented on the motors' data sheets.

To validate the model, several step tests were conducted. The first step test involved stepping the motor position from 0 radians to $\frac{\pi}{3}$ rad, while the motor velocity and torque were fixed to zero. The gains for the PD control were set to half the maximum allowable limits. This test studies the response speed, settling time and overshoot experienced by the motors.

The second step test conducted has the same requirements as the first - the only difference was that the gains for the PD controller were set to the maximum allowable limits. This test would then be compared to the first to evaluate the affect of setting the gain limits on the tracking capability and power/torque requirements.

The third test conducted replicates the second test, with the exception of tracking a spline curve from 0 to $\frac{\pi}{3}$, rather than a step. The key aspect of this test was to

assess the capability of the motors to track a smooth curve. This was done because the trajectories generated from the TO solver are interpolated spline curves.

Lastly, the final step test focused on the torque input. Specifically, all gains for the PD controller were set to zero (controller not influenced by input position nor velocity commands) - while the torque command value was stepped. The test provided insight on the startup torque required for the motor to begin its motion in the absence of position and velocity commands.

The results would consist of four curves: the command, two Simscape curves and the actual motor states. The two Simscape curves were derived from the use of a variable time-step and a fixed time step. The variable time-step was used to validate the feasibility of the input (assess whether the motor can ever achieve the desired input). On the other hand, the fixed time-step aimed to replicate the actual motor output as real time testing depend on a fixed time step.

3.2 The Baleka II Model

Three different modelling approaches were taken to observe the robot's dynamics. Firstly, the robot was modelled in CAD using Solidworks software - which mainly focused around a detailed 3D design of the robot (components and assemblies). This is shown in Figure 7. Secondly, the robot's dynamics were modelled using the manipulator equation approach to achieve the EoM - used in Pyomo. Lastly, the robot was modelled in Simscape which focused on physical body simulations and controller requirements (implicitly determining the EoM).

Baleka II's design was derived from its predecessor, Baleka (a 12 DoF system). The robot consisted of two legs (each made up of four linkages but made use of a five link configuration), a body (housing four electrical BLDC motors), and the boom. Although the robot was capable of body rotation, implementing this feature within the project's timeline was not feasible. As a result, this simplified the physical implementation by reducing the model complexity when solving for trajectories.

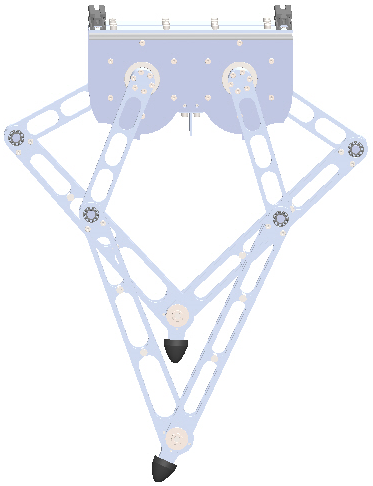

Figure 7: Solidworks model of Baleka II, containing all relevant mass properties of the robot.

3.3 Trajectory Optimization Simulation Setup

TO was the optimal control method used in this research. The manipulator equation was used to model the EoM within the TO formulation. Using these equations: bounds, constraints, system dynamics and the minimization or maximization of a cost function was used to guide the solver in determining an optimal solution [63].

The EoM depended on energies (links, boom, and motors), external forces (ground reaction, and spring-damper), constrained forces (link connections), and input torques. They were derived based on the system's DoF, detailing the motion of all links and foot deflection. Model parameters (MoI, mass, length, and CoM) were sourced from the motor datasheet, Solidworks model, and verified with measurements. A key aspect was the torque-speed characteristic curve, which defined the torque and speed limits. Further details are discussed in Chapter 5.

The EoM were solved using the direct transcription method, which discretizes the TO problem into a NLP with N node points. The NLP solver Interior Point Optimizer (IPOPT) was used for whole-body dynamic motion planning [61, 63]. While indirect methods are more accurate, they have drawbacks compared to direct methods, such as a smaller region of convergence, requiring good initialization, adjoint variable initialization, and higher computational demands [63].

This study focused on solving TO via collocation, which leverage the availability of generalized NLP solvers and introductory tutorials [63]. These methods are categorized by their numerical integration approach: implicit or explicit. Implicit methods use both the current and future states to solve the future state, while explicit meth-

ods rely only on the current state [66]. Although explicit methods are faster and simpler, they can become unstable for non-linear systems. Therefore, implicit integration methods were adopted in this study to ensure stability and accuracy, incorporating both the Backward Euler (BWE) and Radau IIA (implicit Runge-Kutta) approaches.

To both aid the solver and ensure the dynamics of the system avoids singularities - bounding of parameters were incorporated. This includes bounding of the variable time-step, foot deflection, connection force between ankle joint links, GRFs, ground penalties, knee angles and the body height. Furthermore, constraints were applied to maintain the desired dynamics of the system - including, the motor model, ground constraints (contact,friction and slipping), and body-foot constraints (ensuring foot is always below body).

Moreover, a through-contact approach was adopted to model the dynamics of the contacts. Specifically, contact constraints were formulated using complementarity conditions to ensure continuous foot-to-ground contact. These conditions include: the foot's velocity is zero during contact, GRFs are active only when the foot is in contact with the ground, and a friction cone model is maintained. These complementarity conditions were penalized within the cost function.

The cost function is a crucial component of the TO approach, serving as a baseline that guides the solver toward the desired solution. In this study, solution feasibility (minimize ground complementarity penalties) were studied while incorporating methods to minimize the effort of the actuators. To improve solving time, a warm start approach was used. Specifically, running the solver only using the ground penalty to achieve a feasible solution, and thereafter solve the same solution with both the ground penalty and torque. Solving both penalties at once may prove challenging for the solver.

To validate the model, three different testing methods were applied: high drop, low drop, and leg rotation tests. These tests assess whether the system's EoM obey the laws of physics, assess the ground contact model, and assess the actuator/motor model, respectively. Once all three tests were accepted, acceleration, steady-state gait cycle and deceleration trajectories were solved and *stitched* together similarly to what was done in a previous study [16]. This was done for three different forward velocities: walking, running at walk-to-run transition velocity, and running at maximum forward velocity.

The FN was used to determine both the walking and walk-to-run transition velocities (Equation 1). The maximum leg length was set to avoid singularities (knee angles exceeding 180°). A FN of 0.05 was chosen for walking to simulate low-speed locomotion for stability and low impact forces. The walk-to-run transition velocity

corresponded to a FN of 0.5. For maximum velocity, simulations were run with progressively increasing speeds until the solver could no longer find an optimal solution.

For steady-state locomotion, the goal was to obtain a gait cycle at each desired forward velocity. Due to the symmetry of Baleka II, only half of the gait cycle needed solving, as the second half mirrors the first with leg positions and velocities, swapped. Periodic constraints were applied to ensure the initial states of the first leg matched the final states of the second leg, and vice versa. Additionally, a body velocity constraint was used to ensure the robot's average forward velocity matched the desired forward velocity.

When solving locomotion tasks, foot contact order aids the solver but limits the search space. To address this, five simulations were run without a contact order using the faster BWE integration approach. The contact order from the *best solution* was then applied to Radau IIA integration for stability and accuracy. To further expand the search space, simulations were run five times with different seed values, and the *best solution* was implemented on the robot. Criteria for selecting the *best solution* are discussed in Chapter 5.

Once the *best solution* for each gait cycle was found, the acceleration and deceleration trajectories were solved using Radau IIA integration. The acceleration simulation started from the standby state and the termination conditions were set to transitioned into the gait cycle, while the deceleration simulation began from the gait cycle and ended in the standby state. Chapter 6.4 defines the standby state. For both the acceleration and deceleration solutions, the initialization, termination, foot contact order, and seeding parameters differ.

3.4 Simscape Multibody Simulation and Implementation

After solving the *stitched* solutions, the next objectives were to verify the trajectories on a physical model simulator and implement the controller on Baleka II. This section provides a brief overview of the approach used to assess the feasibility of the solved solutions and implement the optimal trajectories on Baleka II.

Baleka II has 12 DOFs, made fully of aluminium parts with motors as actuators. A Speedgoat Real-Time Target machine was used to communicate with the motors via CAN protocol. The controller model was compiled on the target machine using Matlab Simulink modelling tools which were seamlessly compatible with Simscape physical modelling. This physical robot model incorporated all aspects of the robot - from link dimensions and mass properties to motor models.

To ensure that Baleka II remained in a standby state, an impedance controller was employed to facilitate the transition from rest to standby. The controller gains were

critical in determining response time, overshoot, and overall stability. Given that stability was the primary focus of this controller implementation, the gain values must be sufficiently high to maintain the standby state. However, excessively high gains could lead to motor chattering (rapid torque variations), potentially compromising system stability [44], [74].

To determine the gain limitations, the motors were decoupled from the robot, reducing inertia and creating a less damped system with increased vibrations. The gain values were then systematically varied across a defined range. When motor chattering occurred, the corresponding gain values were recorded as boundary limits. Following the identification of these boundaries, the optimal gain values were selected.

The solved trajectories from Pyomo were interpolated and implemented in the Simscape physical model to evaluate command tracking. The model utilized a variable time-step for feasibility assessment rather than precision. Prior to this implementation, validation tests (high drop, low drop, and leg rotation) were conducted to confirm that the dynamic model was correct.

Once the Simscape model successfully simulated the trajectories, they were implemented on Baleka II. The results from the walking, walk-to-run transition, and maximum forward velocity scenarios will be compared and discussed.

3.5 Metrics to be Investigated

The *best solution* is defined according to the desired category of interest, as outlined in Chapter 5.3.1. When assessing the results from the Simscape simulator and the actual Baleka II implementation, steady-state trajectory tracking is crucial. Therefore, state tracking comparisons will be made among the TO solver (Pyomo), the physical model simulator (Simscape), and the physical robot (Simulink).

3.6 Summary

This chapter outlined the methodological framework used to achieve a seamless transition from simulation to real-time implementation of Baleka II. The modelling approach began with parameter validation, where foot-ground contact was captured using a spring-damper system based on the Hertz model to reflect realistic deformation. Motor dynamics were partially modelled by including inertia and the internal PD controller, while friction and back EMF were excluded due to scope limitations.

Three modelling environments were employed: Solidworks was used for the CAD design and to obtain physical properties such as mass and inertia; Simscape provided

a platform for real-time simulation and control; and Pyomo was used for TO solving, with system dynamics expressed in manipulator form.

The TO problem was formulated as a NLP and solved using IPOPT with a direct collocation method. To ensure numerical stability, implicit integration schemes—specifically BWE and Radau IIA—were used. Complementarity conditions modelled footground contact, and a warm-start strategy was implemented to reduce solver time. Optimized trajectories for walking, transition, and running were solved and stitched together using validated system dynamics. These trajectories were first verified in Simscape and then implemented on the physical robot using a Speedgoat Real-Time Target machine.

An impedance controller maintained a stable standby pose, with gain limits determined through motor chattering experiments. Finally, trajectory performance was evaluated based on state tracking accuracy across the solver, simulator, and real robot.

The methodology presented in this chapter provides a structured and scalable approach that can be adapted for other legged robotic platforms, enabling efficient development and deployment of TO and control strategies across different morphologies and actuation schemes.

4 Parameter Validation Testing

This section provides an overview of the data obtained when modelling both the foot deflection and motors adopted in this study.

4.1 Foot Deflection Modelling

For high-speed applications, robots need high friction between their feet and the ground to prevent slipping. Baleka II uses the same tread rubber feet (3 mm width) as its predecessor, designed to withstand high impacts without oscillation [17]. Additionally, natural rubber mats were placed around the lab for high traction. A previous study showed that at a load of 150 N, the friction coefficient for rubber is 0.8 [72]. Since the expected load will be at least double the robot's weight (approximately 150 N), a static and dynamic friction coefficient of 0.8 was set for all experiments.

The sampled data in Table 1 was obtained by placing the robot on the force sensor (with a rubber mat between the robot's feet and sensor plate). Specifically, $\mathbf{F_y}$, $\mathbf{Y_b}$, and $\dot{\mathbf{Y}}_\mathbf{b}$ represents the force sensed in the Y-direction, body position in the Y-direction and body velocity in the Y-direction, respectively.

The data obtained in Figure 8 represents a multiple linear regression approach to fit each rapid change in force. Specifically, 10 different multiple linear regression batches were calculated for each row of three in Table 1 - as each batch provides a rapid change in force in the same direction.

Table 1: Extracted data table from deflection curve (Figure 8).

Time (s)	$\mathbf{Y_b}$ (m)	$\dot{\mathbf{Y}}_\mathbf{b}$ (m/s)	$\mathbf{F_y}$ (N)
28	0.2023	0	-132.2
28.5	0.202	-0.009	-137.4
28.522	0.2014	-0.016	-140.5
28.564	0.2012	-0.07	-145.5
28.606	0.2013	0	-150.9
30.474	0.2014	0	-300
30.484	0.201	-0.007	-301.3
30.505	0.2004	-0.016	-304.4
30.549	0.2002	-0.007	-311.2
30.586	0.2003	0	-314.9
32.27	0.2004	0.001	-221.5
32.28	0.2007	0.009	-215
32.3	0.2012	0.017	-202.1
32.35	0.2015	0.007	-166.6
32.399	0.2014	-0.001	-138.2
38.45	0.2014	0	-323.2
38.47	0.2011	-0.005	-326
38.5	0.2004	-0.016	-327
38.54	0.2002	-0.006	-323.7
38.61	0.2004	0	-335.3
40.87	0.2005	0.003	-253.3
40.88	0.201	0.011	-236.1
40.9	0.2013	0.019	-209.9
40.94	0.2012	0.006	-129.7
40.98	0.2014	-0.005	-105.8

The linear regression curves were used to determine K_{Foot} and D_{Foot} gains for each batch as shown in Figure 9. The resulting gains were averaged, where: $K_{Foot} = 45018.89$ N/m and $D_{Foot} = 1818.57$ Ns/m. The coefficients were set to $K_{Foot} = 10^4$ N/m and $D_{Foot} = 10^3$ Ns/m to levitate some of the approximations made. Table 1 further highlights that the maximum foot deflection is 0.01 m from beginning to end of deformation.

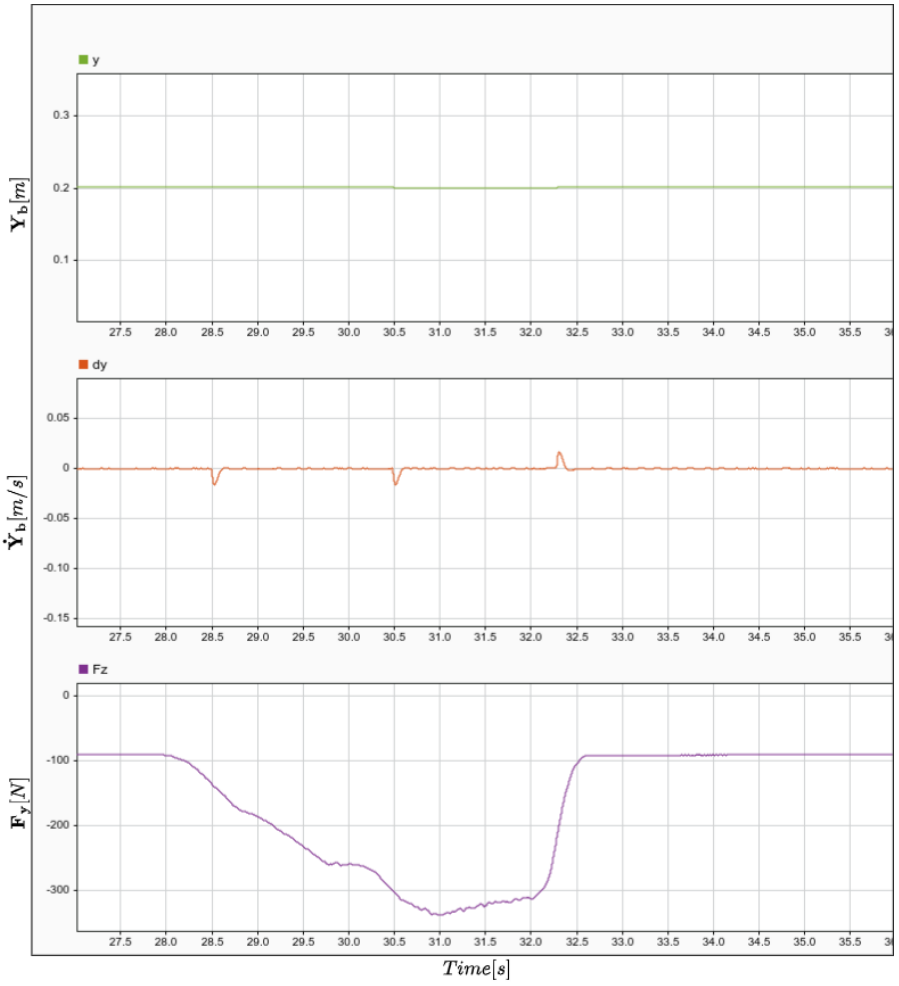

Figure 8: Results obtained from the deflection test between the rubber feet of the robot and the rubber mat.

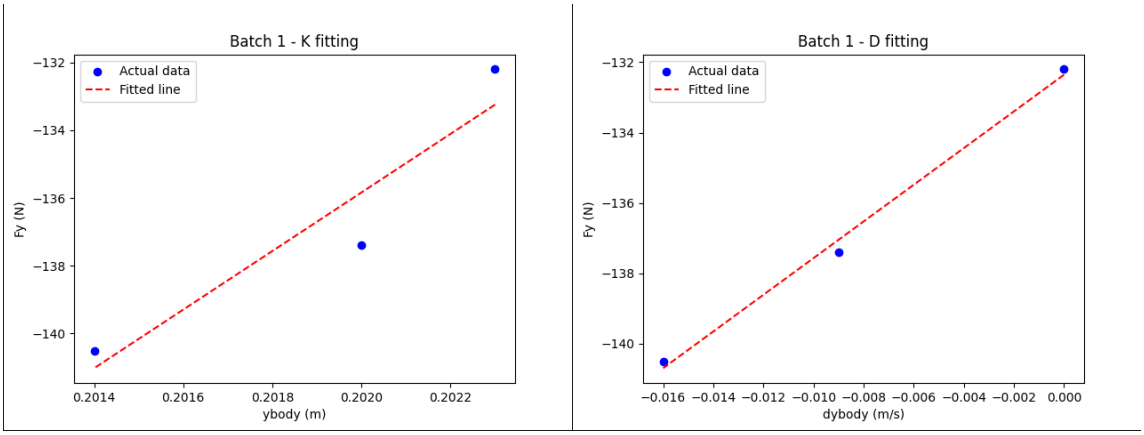

Figure 9: Linear Regression of deflection testing. The left and right curve estimates a coefficient for K_{Foot} and D_{Foot}, respectively.

4.2 Motor Modelling

Four TMotor AK10-9 V1.1 KV 100 QDD BLDC motors were used. These motors were best suited for locomotion given it contained: high torque, high speed, low mass, small diameter, low gear reduction (allowing for backdrivability), and consisted of a built in driver board. Table 2 provides an overview of all the motors' parameters while Figure 20 displays the approximated torque-speed curve. When setting up the motor parameters, a serial debugging interface via an R-Link for encoder calibration was used, assigning CAN IDs per motor (1-4). This allows: setting the speed, torque and position limits. However, during operation, it makes use of CAN protocol with a maximum communication rate of 1 MHz.

Table 2: Specification Table for AK10-9 V1.1 KV100 Tmotors.

Parameter	**Value**
Weight (g)	820
Voltage (V)	24
Rated Torque (Nm)	15
Peak Torque (Nm)	38
Maximum Speed @ Rated Torque (rpm)	222 (Output)
Rated Current (A)	18.5
Peak Current (A)	50
Kt (Nm/A)	0.095
Kv (RPM/V)	100
Ke (V/rpm)	0.01
Km (Nm/$\sqrt{\text{W}}$)	0.3
Number of Pole-Pair	21
Resistance Phase to Phase (Ω)	90 ± 5
Inductance Phase to Phase (uH)	331 ± 10
OD (mm)	Φ98
Height (mm)	62.5
Maximum Torque Weight Ratio (Nm/kg)	65
Reduction Ratio	9:1

The motor makes use of Field Oriented Control (FOC) algorithm to accurately control the angular position, angular velocity and torque. These motors use electronically controlled commutators, three phase winding's, and a permanent magnet rotor (as seen in Figure 10). FOC allows for BLDC motors to operate efficiently and smoothly by reducing torque ripples which are ideal for systems with quick dynamic responses to load and speed changes [75].

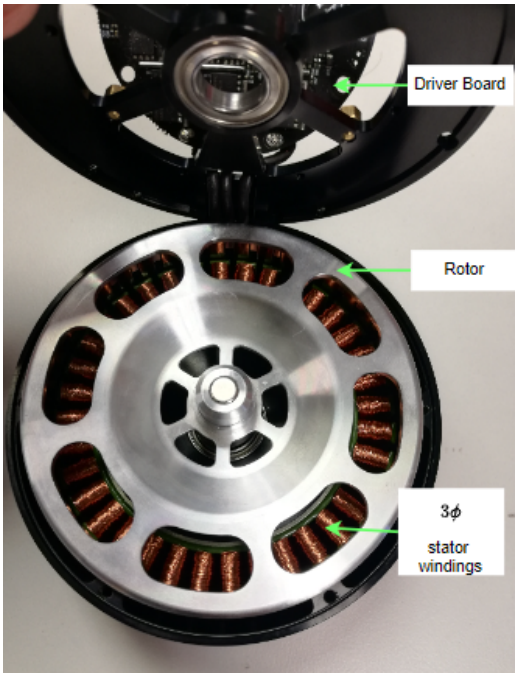

Figure 10: Internal structure of the AK10-9 motor, highlighting the rotor, stator, and motor driver board.

The motor controller makes use of PD control for the position and velocity control loops. The motor has an integrated driver board containing a single loop absolute encoder 14 BIT to accurately measure the position ($\mathbf{\Theta_{Mes}}$). Furthermore, the measured velocity ($\mathbf{\dot{\Theta}_{Mes}}$) is determined by differentiating the measured position data (with respect to time). The desired position ($\mathbf{\Theta_{Des}}$) and velocity ($\mathbf{\dot{\Theta}_{Des}}$) is specified by the user. This is shown in Figure 11 - where Kp is the proportional gain for the position loop and Kd is the proportional gain for the velocity loop. The maximum allowable value for $Kp = 500$ and $Kd = 5$. The resulting control signals are then added to a reference torque value ($\mathbf{T_{ff}}$). Thereafter, the torque is converted to the current and limited within the set bounds. This current is then fed into the FOC.

FOC aims to ensure the stator vector field leads the rotor vector field by 90° to maximize the torque. The stator field is adjusted by splitting it up into it's direct component (parallel to rotor vector field) and quadrature component (perpendicular to rotor vector field). Therefore, minimizing the direct component to zero allows maximizing the quadrature component. The stator vector field is manipulated with the three phase currents to obtain direct (id_{ref}) and quadrature (iq_{ref}) current values (using Clarke/Park transforms). As shown in Figure 11, the id_{ref} is equal to the control current resulting from the PD control, while $iq_{ref} = 0$.

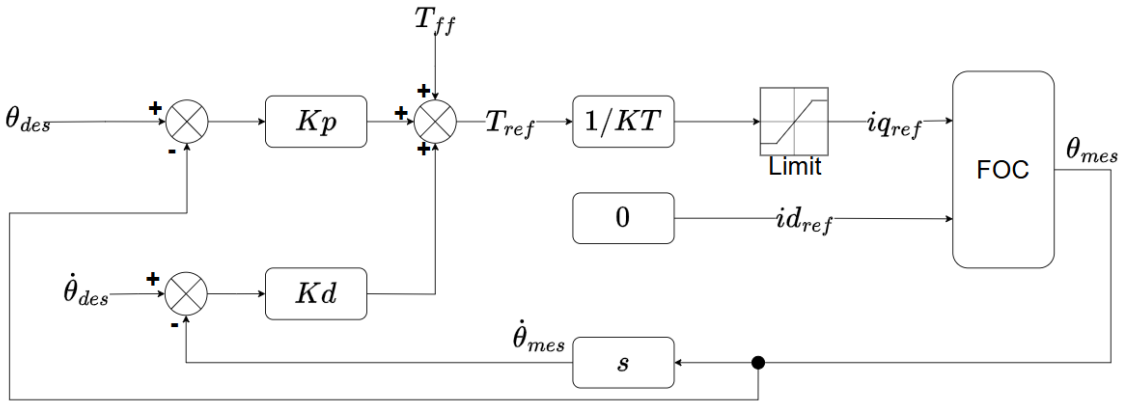

Figure 11: Inner control loop within each motor - includes a PD control for position and velocity commands, which is then fed into a FOC to obtain the desired set-points using current.

Figure 12 presents an overview of Simulink motor model - taking into account the torque-speed curve, rotor inertia, and PD controller. The following values were used (by using Solidworks and measuring the motor dimensions):

1. Length between Teeth = 0.0060 m
2. Ring Radius = 0.0360 m
3. Ring Width = 0.0045 m
4. Sun Radius = 0.0045 m
5. Planet Radius = 0.0158 m
6. Gear Carrier Length = 0.0203 m
7. Rotor Interia = 181877 g.mm^2

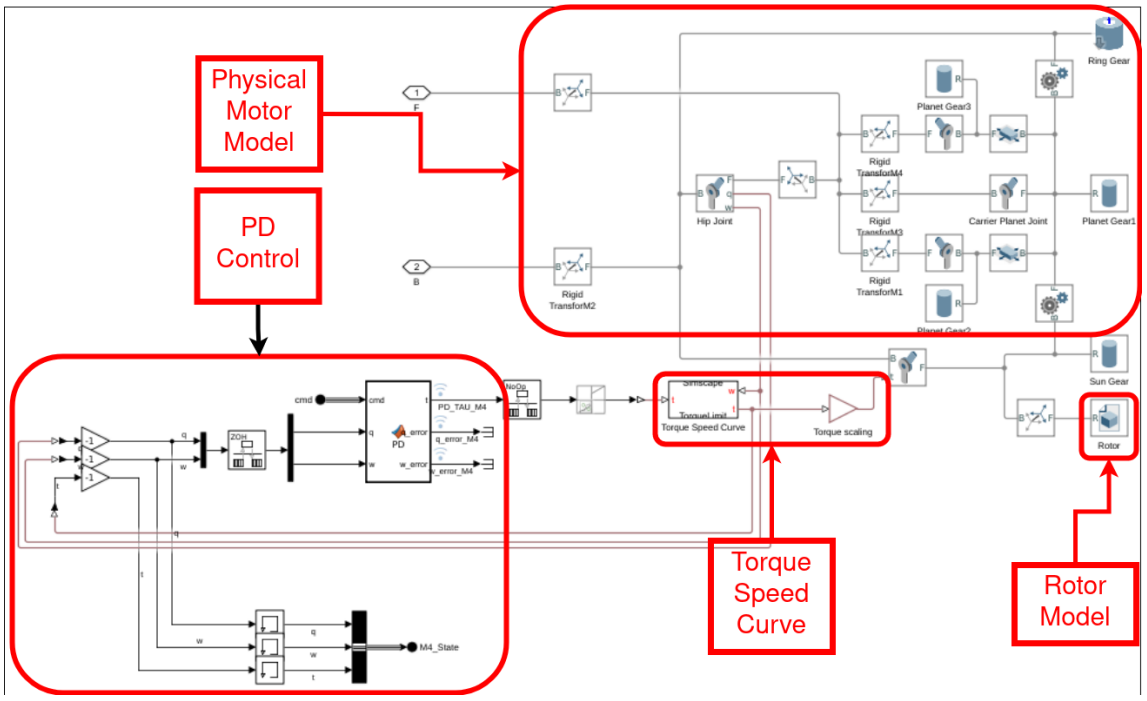

Figure 12: Overview of the Simscape Multibody model incorporating the rotor inertia, torque-speed curve, PD controllerm, and the rest of the physical motor model (gears).

The below results were obtained from the four step tests highlighted in the methodology (Figure 13, 14, 15, and 16). Pyomo, Simscape, and Simulink refers to the TO commands, physical simulator model response, and actual motor response, respectively. The outcomes were as follows:

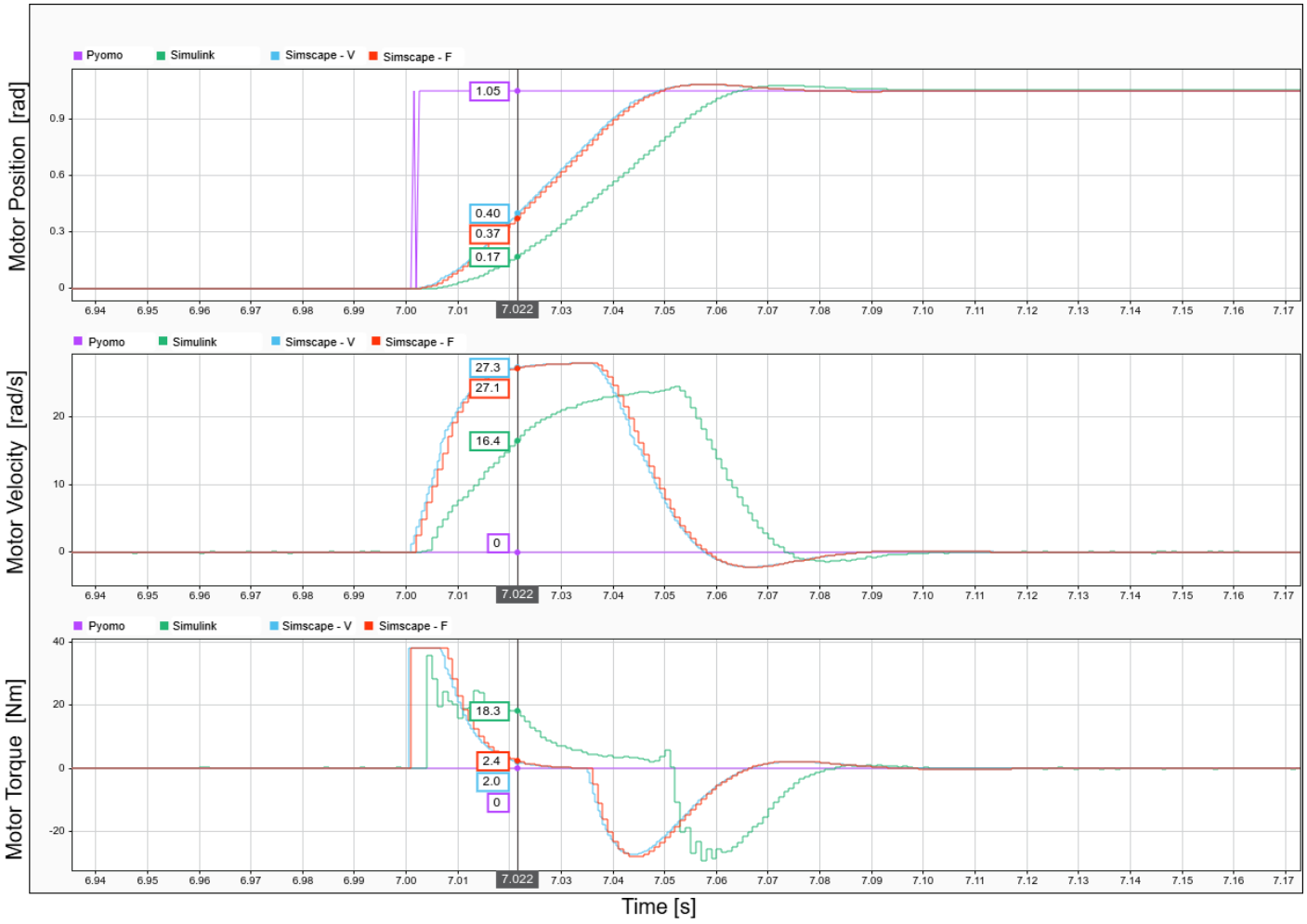

Figure 13: Step Test 1: Results of four step tests are shown for a single motor. Each plot consists of three subplots: position, velocity, and torque. Step 'Pyomo', 'Simulink', 'Simscape-F', and 'Simscape-V' represents the desired command, actual motor states, and fixed and variable time-step Simscape results, respectively.

Step Test 1:

$\boldsymbol{\Theta}_{\mathbf{Des}}$ was stepped from 0 to $\frac{\pi}{3}$ while $\dot{\boldsymbol{\Theta}}_{\mathbf{Des}}$ and $\mathbf{T}_{\mathbf{ff}}$ were fixed to zero. In addition, the PD controller gains were set to $Kp = 250$ and $Kd = 2$. It's observed that the Simscape fixed time-step (at 1 kHz), variable time-step, and actual motor curve (Simulink) solutions are close in value. However, the Simscape solutions has a faster response time than Simulink. Moreover, the Simscape position overshoot Simscape by 0.07 radians while the Simscape's peak velocity experiences a 3.4 rad/s higher velocity than Simulink. This is most likely due to the mismatch in mass property modelling in Simscape. In addition, the torque curves display that the Simscape

model reach the torque limit while the Simulink result have not. It is also observed that the Sumulink result applies more torque over a longer period compared to the simulated models - indicating to the absence of friction in the model.

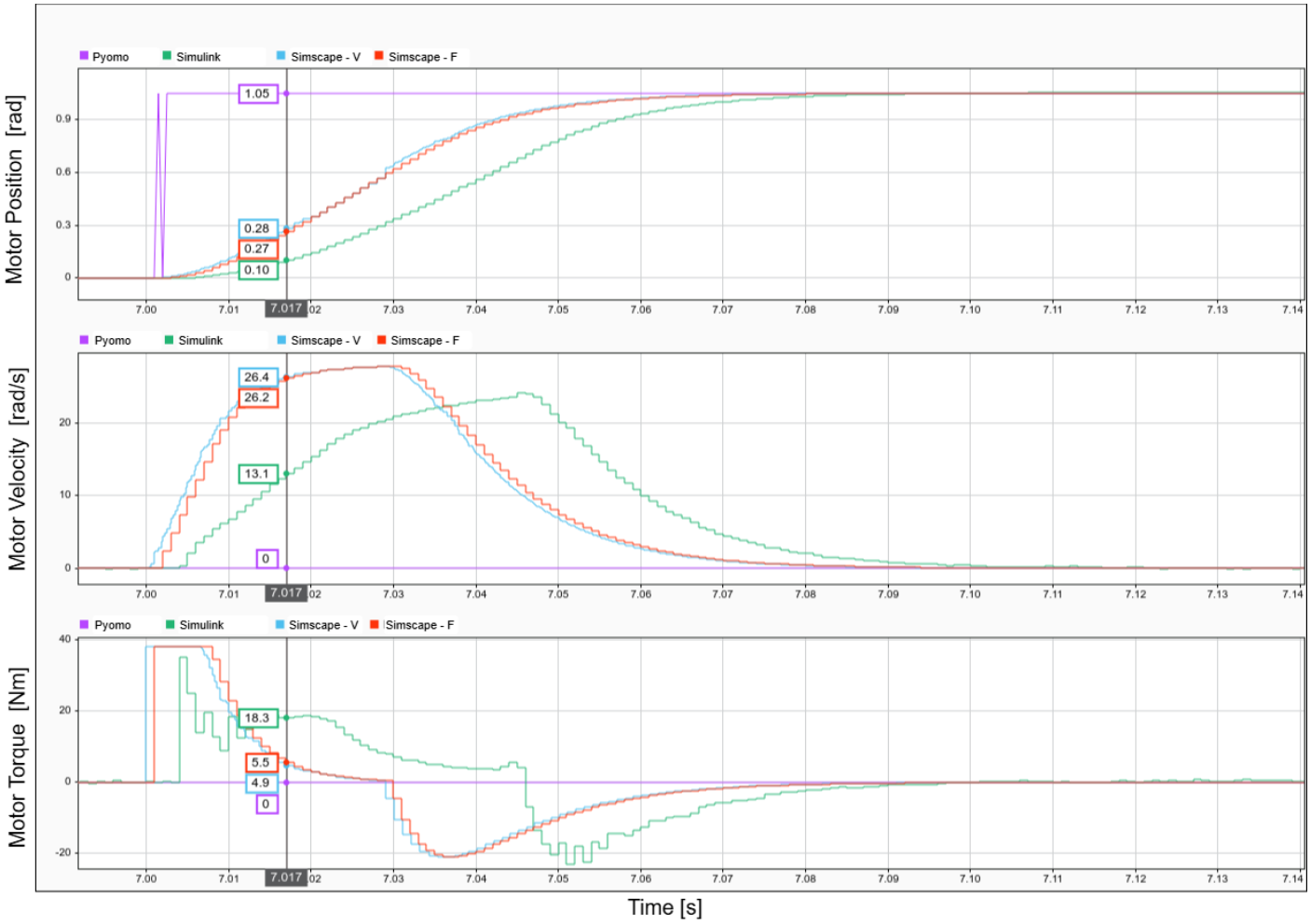

Figure 14: Step Test 2: Results of four step tests are shown for a single motor. Each plot consists of three subplots: position, velocity, and torque. Step 'Pyomo', 'Simulink', 'Simscape-F', and 'Simscape-V' represents the desired command, actual motor states, and fixed and variable time-step Simscape results, respectively.

Step Test 2:
The only difference between Step Test 1 and 2 was that $Kd = 4$. It is observed that both the Simscape and Simulink testing experience no overshoot of the position tracking nor undershoot of the velocity tracking. However, the Simscape motor has a faster response time and makes use of a high peak velocity. Nonetheless, the peak torques experienced by Test 2 is higher than Test 1.

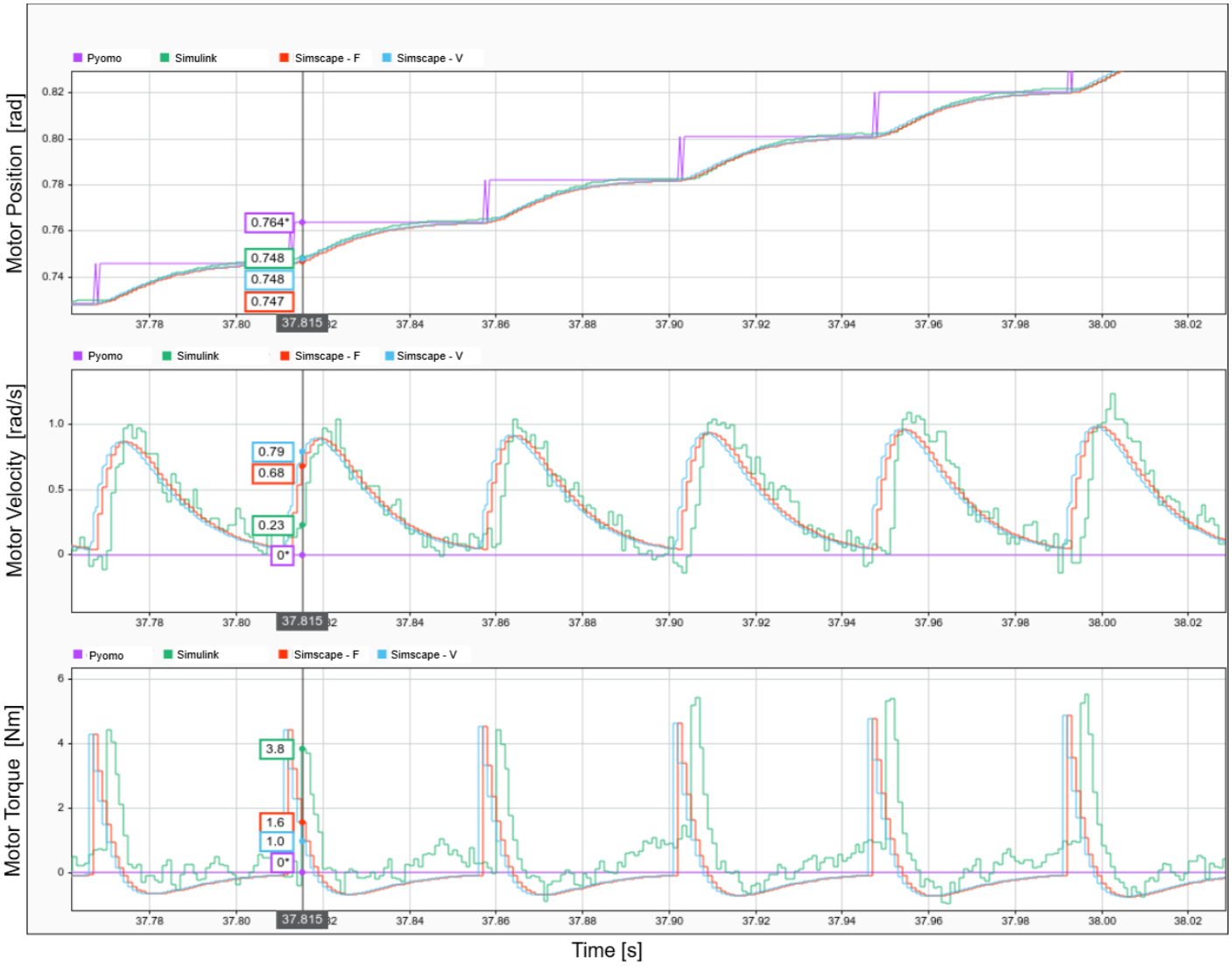

Figure 15: Step Test 3: Results of four step tests are shown for a single motor. Each plot consists of three subplots: position, velocity, and torque. Step 'Pyomo', 'Simulink', 'Simscape-F', and 'Simscape-V' represents the desired command, actual motor states, and fixed and variable time-step Simscape results, respectively.

Step Test 3:
This test focused on tracking a smooth input position curve (with the same setup as Test 1). It is observed that both the Simscape and Simulink motors has minor tracking errors. In addition, the response speed curves are closely related while both experience no overshoot. Furthermore, the velocity curves presents a close resemblance - however, there seems to be more variation in Simulink's response. Moreover, the torque curves also resembles a close similarity in values over time

(with more variation in the actual motors response) but the torque values of the actual motor has a slow response time.

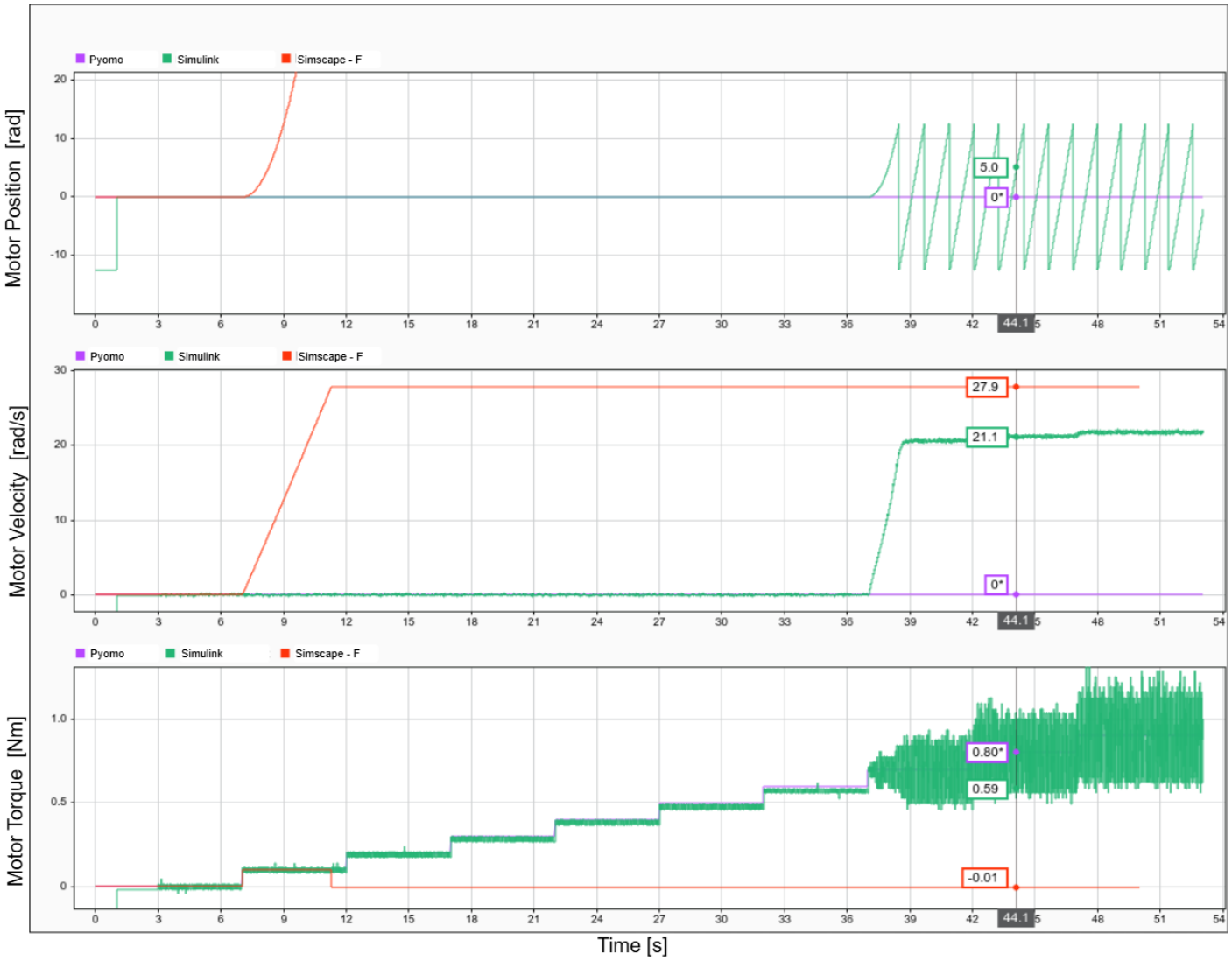

Figure 16: Step Test 4: Results of four step tests are shown for a single motor. Each plot consists of three subplots: position, velocity, and torque. Step 'Pyomo', 'Simulink', 'Simscape-F', and 'Simscape-V' represents the desired command, actual motor states, and fixed and variable time-step Simscape results, respectively.

Step Test 4:
This test investigated a pure torque input. At 0.8 Nm ($Kp = Kd = 0$), the motors overcome the start up torque. It is also observed that Sumulink's torque curve begins to oscillate once the motor begins to rotate (within 0.5 Nm variation from

the command). This is due to the vibration of the motor as it oscillates at high speeds. Furthermore, the Simscape model does not mimic the actual motor model as resetting the encoder (position and velocity) was not modelled - this caused the simulated model to increase in position indefinitely and the peak velocity to be reached. Nonetheless, high accuracy of this model was not required as the range of motion for the entire robot would never exceed a full revolution of any motor.

4.3 Summary

The parameter validation testing chapter confirmed the accuracy and realism of the foot and motor models used in the Baleka II robot.

For foot deflection, high-friction rubber feet combined with rubber mats yielded a friction coefficient of 0.8 to accommodate contact loads exceeding 150 N. Experimental data from force sensor testing enabled the estimation of foot stiffness as $K_{Foot} = 10^4$ N/m and foot damping as $D_{Foot} = 10^3$ Ns/m after applying multiple linear regression to deflection data. The maximum foot deflection experienced estimated to 0.01 m.

Motor modelling focused on the AK10-9 QDD motors, emphasizing their high torque-to-weight ratio, backdrivability, and efficient FOC. PD control loops governed motor behaviour, with parameters capped at $K_p = 500$ and $K_d = 5$. Simulink and Simscape models incorporated motor characteristics including torque-speed curves and rotor inertia. Four step tests validated motor response, confirming that simulated results closely matched physical motor behaviour, while also revealing differences due to unmodelled friction and encoder resets. Overall, modelling parameters were finalized based on empirical validation, with simplified stiffness, damping, and control parameters used to balance accuracy and computational efficiency.

5 Trajectory Optimization Simulations

This section provides a detailed discussion of the parameters referenced in the methodology for TO. Additionally, all relevant TO results will be presented and thoroughly analysed.

All TO code were written in Python within the Jupyter Notebook environment. Pyomo, an open-source library, was used for modelling and solving optimization tasks [76], while IPOPT was employed to solve NLP problems [77], using the MA86 linear solver for large sparse systems. The software ran on machines operating Ubuntu 20.04 LTS, with a solution feasibility tolerance set to 10^{-6} as per [67].

5.1 Setup and Design Decisions

5.1.1 Equations of Motion

Baleka II is a 12-DoF bipedal robot with four motor actuators, eight linkages, a body connected to a boom, and rubber feet soles that deflect upon impact. Each foot was modelled as a spring-damper system to account for deflection during ground contact. The EoM were derived using Euler-Lagrange dynamics, following the manipulator equation as shown in Equation 6. Moreover, absolute angles were used to reduce solver time with reference to Figure 17 [78]. The rest-state position of the upper links were used as a reference plane for the link angles, measuring 2.23 radians between the reference plane and the Y-plane. The matrices are defined as:

1. $\mathbf{q}$ is the generalized coordinates with $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ being the state velocity and acceleration, respectively;
2. $\mathbf{M}(\mathbf{q})$ is the Mass matrix;
3. $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ is the Coriolis matrix;
4. $\mathbf{G}$ is the gravitational potential matrix;
5. $\mathbf{B}$ mapped the applied/control forces and torques $\mathbf{u}$ to the generalised coordinates;
6. $\mathbf{A}$ is the mapping of external forces, $\mathbf{\Lambda}$ to the generalised coordinates;
7. $\mathbf{R}$ represents the effect of the damper.

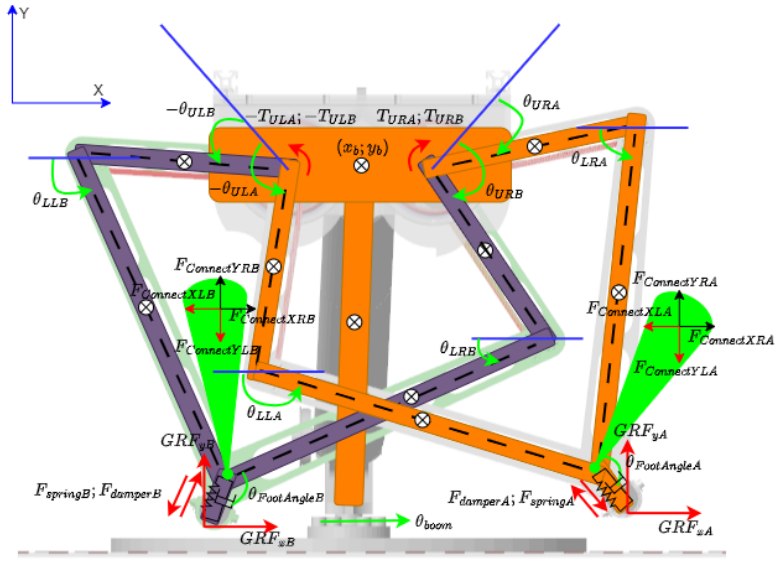

Figure 17: Representation of reference angles used to derive the kinematics (angles shown in green) and the forces used within the EoM (red). These consist of connection forces, spring-damper, GRFs, and torque actuators.

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G} = \mathbf{B}\mathbf{u} + \mathbf{A}\boldsymbol{\Lambda} + \mathbf{R} \tag{6}$$

The generalized coordinates $\mathbf{X_b}$ and $\mathbf{Y_b}$ represent the translational movement of the body in the x- and y-directions, respectively. In both equations 7 and 9, $\boldsymbol{\Theta}$ denotes an angle, while $\mathbf{T}$ represents the control input torque of the motors. The subscript notation is [U/L; L/R; A/B], where U/L refers to the upper or lower link, L/R to the left or right link, and A/B to the front and back leg, respectively. The term $\boldsymbol{\Theta}_{\mathbf{boom}}$ accounts for the boom dynamics, treating it as another linkage relative to the ground. Additionally, $\boldsymbol{\Delta}_{\mathbf{A}}$ and $\boldsymbol{\Delta}_{\mathbf{B}}$ represent the deflection of feet A and B, respectively. All mass property values used to build the model in Pyomo are described in Chapter 6.1 using Table 9. Furthermore, the in depth formulation of the EoM are provided in Appendix A and B.

$$\begin{aligned}\mathbf{q} = [&\mathbf{X_b}; \mathbf{Y_b}; \boldsymbol{\Theta}_{\mathbf{ULA}}; \boldsymbol{\Theta}_{\mathbf{URA}}; \boldsymbol{\Theta}_{\mathbf{LLA}}; \boldsymbol{\Theta}_{\mathbf{LRA}}; \\ &\boldsymbol{\Theta}_{\mathbf{ULB}}; \boldsymbol{\Theta}_{\mathbf{URB}}; \boldsymbol{\Theta}_{\mathbf{LLB}}; \boldsymbol{\Theta}_{\mathbf{LRB}}; \boldsymbol{\Delta}_{\mathbf{A}}; \boldsymbol{\Delta}_{\mathbf{B}}]^T\end{aligned} \tag{7}$$

Both potential and kinetic energy are necessary to determine the matrices $\mathbf{M}$, $\mathbf{C}$, and $\mathbf{G}$. The kinetic energy includes the contributions from the body, boom, and each leg link, accounting for both translational and rotational effects. Additionally,

the rotational kinetic energy of the motors (KE_{motors}) was modelled (see Equation 8), where N_{GR} represents the motor's gear reduction ratio, and I_{rotor} denotes the inertia of the motor's rotor [44]. The potential energy was calculated by considering the effects of the body, boom, leg links, and the springs at the feet.

$$KE_{motors} = 0.5 N_{GR} I_{rotor} (\dot{\mathbf{\Theta}}^{\mathbf{2}}_{\mathbf{ULA}} + \dot{\mathbf{\Theta}}^{\mathbf{2}}_{\mathbf{URA}} + \dot{\mathbf{\Theta}}^{\mathbf{2}}_{\mathbf{ULB}} + \dot{\mathbf{\Theta}}^{\mathbf{2}}_{\mathbf{URB}}) \tag{8}$$

There are two types of external forces: GRF during contact and connection forces between within angle linkages. The subscript notation for GRF include $\mathbf{X}$ and $\mathbf{Y}$ which represent the GRF in the x and y directions, while A and B denote the front and back leg, respectively. The connection force $\mathbf{Fc}$ applies between the lower left and right links, as the upper links have rotary connections. The foot contact damping force is given in Equation 11, where D_{Foot} is the damping coefficient, and $\dot{\mathbf{\Delta}}_{\mathbf{A}}$ and $\dot{\mathbf{\Delta}}_{\mathbf{B}}$ are the deflection rates of foot A and B, respectively.

$$\mathbf{u} = [\mathbf{T_{ULA}}; \mathbf{T_{URA}}; \mathbf{T_{ULB}}; \mathbf{T_{URB}}]^T \tag{9}$$

$$\begin{aligned} \mathbf{\Lambda} = \mathbf{GRF_{XA}} + \mathbf{GRF_{YA}} + \mathbf{Fc_{XRA}} + \mathbf{Fc_{YRA}} - \mathbf{Fc_{XLA}} - \mathbf{Fc_{YLA}} + \\ \mathbf{GRF_{XB}} + \mathbf{GRF_{YB}} + \mathbf{Fc_{XRB}} + \mathbf{Fc_{YRB}} - \mathbf{Fc_{XLB}} - \mathbf{Fc_{YLB}} \end{aligned} \tag{10}$$

$$\mathbf{R} = D_{Foot} \dot{\mathbf{\Delta}}_{\mathbf{A}} + D_{Foot} \dot{\mathbf{\Delta}}_{\mathbf{B}} \tag{11}$$

5.1.2 Collocation and Integration

Foot-ground collisions are unavoidable in legged locomotion and introduce discontinuous events that disrupt stability in the gait cycle. Accurately modelling these collisions is essential for maintaining balance. However, fixed time-steps in NLP solvers may fail to capture collisions precisely due to low resolution. By using variable-time intervals, the solver can more naturally detect collisions, improving the overall accuracy of the model [66].

Variable time-steps help prevent overshooting and ensure critical events are captured during transitions. To enhance the scalability of the non-linear problem, the variable time-step (h) is scaled by the total number of element intervals (hm), as defined in Equation 12. Here, N represents the total number of elements/nodes, while TT indicates the desired total simulation time. Equations 13 detail the actual simulation time at a specific node (n), where $t[N]$ is the total simulation time. In all tasks, h was varied between 0.5 and 1.2, while TT ranged from 0.3 s to 1.5 s, depending on the locomotive task.

$$\begin{aligned} hm &= \frac{TT}{N} \\ 0.5 <= h <= 1.2 \end{aligned} \tag{12}$$

$$t[n] = \sum_{n=1}^{N} t[n-1] + h_m h[n] \tag{13}$$

In this study, both generalized coordinates and control variables were discretized. Therefore, the ODE was integrated via equality constraints. The integration scheme influences the solving time, accuracy, and overall quality of the solution. Two implicit integration methods were incorporated in this study, namely: BWE and two stage Radau IIA (implicit Runge-Kutta method).

The BWE method was adopted given its fast solve rate while maintaining adequate stability, making it an ideal reference point for assessing the feasibility of the desired motions. Meanwhile, Radau was chosen to handle large step sizes suitable for stiff ODEs and for its improved accuracy compared to BWE. The global error for BWE and Radau are: $O(h)$ and $O(h^{2K-1})$ (where K is the number of collocation points), respectively. Thus, two stage Radau integration has a third order accuracy $O(h^3)$ - while BWE is only first order.

In this study, solutions were first solved using the BWE approach and thereafter Radau. The BWE approach determined the optimal constraints, bounds, contact orders, setup, initialization, and termination parameters for a desired motion. Equation 14 and 15 represents the BWE state position $\mathbf{q}$ and state velocity $\dot{\mathbf{q}}$ integration update schemes (at node n), respectively.

$$\mathbf{q}[\mathbf{n}] = \mathbf{q}[\mathbf{n-1}] + hm \times h[n]\dot{\mathbf{q}}[\mathbf{n}] \tag{14}$$

$$\dot{\mathbf{q}}[\mathbf{n}] = \dot{\mathbf{q}}[\mathbf{n-1}] + hm \times h[n]\ddot{\mathbf{q}}[\mathbf{n}] \tag{15}$$

Two stage Radau integrates the system equations at every second collocation point (p) to determine the optimal path between each node (n). This study adopts the following notation when specifying the state at a particular node and collocation point: $\mathbf{q}[n, p]$. Equations 17 and 18 represents the two stage Radau state position $\mathbf{q}$ and state velocity $\dot{\mathbf{q}}$ integration schemes, respectively. In both equations, the Butcher tableau coefficient matrix (**CM**) highlights are weights and contributions used to compute the current and future time-steps (defined in Equation 16). Furthermore, the last collocation point of the current node equates to the initial collocation point

of the next node to ensure continuity and smooth transitions between time intervals: $q[n, 2] = q[n + 1, 1]$. Figure 18 provides an overview of this approach.

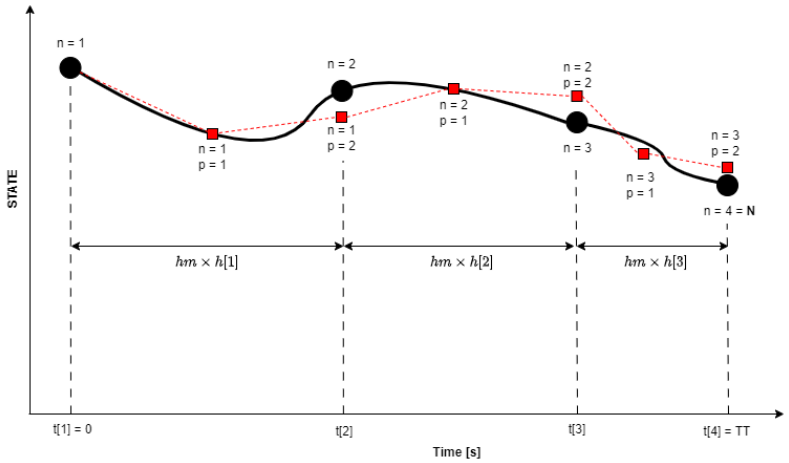

Figure 18: Representation of two-stage Radau progression of a state-variable. Where $hm \times h[n]$ denotes the time-step size. Between each node, two collocation points are specified with the state of the second p equating to the state of the initial p of the next node.

$$\mathbf{CM} = \begin{bmatrix} \frac{5}{12} & -\frac{1}{12} \\ \frac{3}{4} & \frac{1}{4} \end{bmatrix} \tag{16}$$

$$\mathbf{q}[n, p] = \begin{cases} \mathbf{q}[n-1, 2] & \text{for } p = 0 \\ \mathbf{q}[n, 0] + hm \times h[n] \sum_{pp=1}^{2} \mathbf{CM}[p-1][pp-1] \dot{\mathbf{q}}[n, pp] & \text{for } p > 0 \end{cases} \tag{17}$$

$$\dot{\mathbf{q}}[n, p] = \begin{cases} \dot{\mathbf{q}}[n-1, 2] & \text{for } p = 0 \\ \dot{\mathbf{q}}[n, 0] + hm \times h[n] \sum_{pp=1}^{2} \mathbf{CM}[p-1][pp-1] \ddot{\mathbf{q}}[n, pp] & \text{for } p > 0 \end{cases} \tag{18}$$

In this study, N was set to 100 for BWE simulations while N varied between 30 and 50 for Radau simulations. These values were selected to balance accuracy and solving speed. As a result, for TT = 1 s, BWE provides an accuracy of $O(h^k = O((1/100)) = O(10^{-2})$ while Radau provides an improved accuracy of $O(h^{2k-1}) = O((1/50)^3) = O(8^{-6})$.

5.1.3 Optimization Variable Bounds

Bounding the TO variables aids the solver by narrowing the search space for solutions. Specifically, the solver operates between an upper (UB) and lower bound (LB) of each variable (Var): $Var_{LB} <= Var <= Var_{UB}$. The adopted bounds includes knee angles, foot deflection, connection forces, and GRFs.

Due to the robot's under-actuated design, the knee angle ($\mathbf{\Theta_{KneeAngle}}$) could potentially reach or exceed 180° (see Figure 19), creating an unstable and irrecoverable configuration. Therefore, a 10° headroom was selected to compensate potential overshoot/undershoot of the knee angle. Hence, the knee angle bounds were set between 10 and 170° to prevent the robot's legs from reaching a singularity.

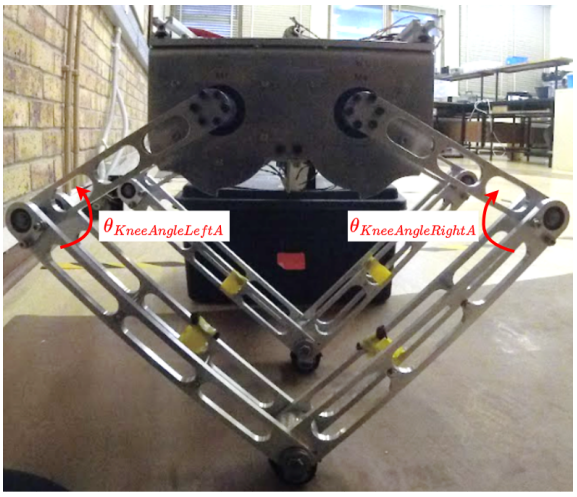

Figure 19: Baleka II on the preparation/setup stand, ($\mathbf{\Theta_{KneeAngleLeftA}}$) and ($\mathbf{\Theta_{KneeAngleRightA}}$) represents the left and right knee angles of leg A, respectively.

The foot deflection ($\mathbf{\Delta_A}$ and $\mathbf{\Delta_B}$) bounds were set in line with the maximum deflection experienced in Table 1 (upper bound = 0.01 m and lower bound = -0.01 m). In addition, the bounds of both **Fc** and **GRF** were set between ten times the total body weight - see Equation 19 (with TM being the total mass of the robot). This was set to ensure the connection could handle an impact 10 times its body weight at high speeds due to large GRFs.

$$-10 \times TM \times 9.81 <= \mathbf{Fc}, \mathbf{GRF} <= 10 \times TM \times 9.81 \tag{19}$$

5.1.4 Optimization Variable Constraints

This section will provide an overview of key constraints implemented for all locomotive tasks.

Motor Model

In an ideal world, the actuators has infinite torque. However, in practice - there are limitations on the motor actuation characteristics. Therefore, in this study a linear power model was approximated from the datasheet provided by the motor manufacturers (AK10-9 motors [79]).

The linear torque-speed curve constraint ensures the motor operates in the continuous region, to prevent overheating [44]. The curve shown in Figure 20 was used as torque-speed constraints and was derived using parameters from Table 3 and equations 20, 21, and 22. The following notation has been adopted: $\mathbf{T_U}$ represents actuator (see Equation 9) and $\dot{\mathbf{\Theta}}_\mathbf{U}$ represents the angular velocity of the upper link connected to the motor ($\dot{\mathbf{\Theta}}_\mathbf{ULA}$, $\dot{\mathbf{\Theta}}_\mathbf{URA}$, $\dot{\mathbf{\Theta}}_\mathbf{ULB}$ or $\dot{\mathbf{\Theta}}_\mathbf{URB}$). Moreover, $\mathbf{T_{rated}}$, $\mathbf{T_{peak}}$, $\dot{\mathbf{\Theta}}_\mathbf{NoLoad}$, and $\dot{\mathbf{\Theta}}_\mathbf{Trated}$ represents the rated torque, peak torque, no load angular velocity, and maximum angular velocity at the rated torque of the upper link, respectively. In addition, $\mathbf{T_{Des}}$ represents the bounds of the desired torque limit. This was done to allow headroom for the feedback controller - specifically, the torque command produced from the PD controllers.

Table 3: Table of parameters used for the motor model curve derivation.

Parameter	Value
$\mathbf{T_{rated}}$	15.0 Nm
$\mathbf{T_{peak}}$	38.0 Nm
$\mathbf{T_{Des}}$	30.0 Nm

$$\begin{aligned} -\mathbf{T_{Des}} <= \mathbf{T_U} <= \mathbf{T_{Des}} \\ \dot{\mathbf{\Theta}}_\mathbf{NoLoad} = 24V(100/9)(2\pi/60) \\ \dot{\mathbf{\Theta}}_\mathbf{Trated} = 222(2\pi/60) \end{aligned} \tag{20}$$

$$\begin{aligned} \mathbf{m_{grad}} = (\mathbf{T_{rated}} - 0.0)/(\dot{\mathbf{\Theta}}_\mathbf{Trated} - \dot{\mathbf{\Theta}}_\mathbf{NoLoad}) \\ \mathbf{T_{stall}} = 0.0 - \mathbf{m_{grad}} \times \dot{\mathbf{\Theta}}_\mathbf{NoLoad} \end{aligned} \tag{21}$$

$$\mathbf{T_U} = \begin{cases} <= \dot{\mathbf{\Theta}}_\mathbf{U} \times \mathbf{m_{grad}} + \mathbf{T_{stall}} & \text{for } \dot{\mathbf{\Theta}}_\mathbf{U} >= 0 \\ >= \dot{\mathbf{\Theta}}_\mathbf{U} \times \mathbf{m_{grad}} - \mathbf{T_{stall}} & \text{for } \dot{\mathbf{\Theta}}_\mathbf{U} < 0 \end{cases} \tag{22}$$

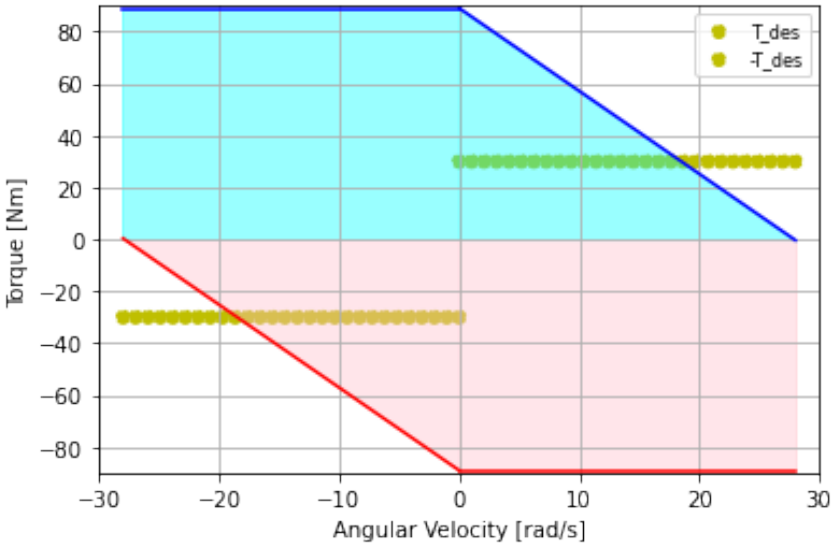

Figure 20: Torque Speed Curve used in Pyomo - highlighting the bounds in which the motors values are allowed to operate (with the maximum desired torque limit of 30 Nm).

Through-Contact Optimisation Method

In this study, contacts were modelled using through-contact methods. Specifically, the contact constraints were formulated with the use of complementarity conditions as it seamlessly integrates with direct formulation of TO [70]. This method focuses on treating the contact forces as parameters to be optimized. One downside of this approach is that it introduces additional parameters and constraints - increasing the problem's complexity. Nonetheless, this allows the solver to discover optimal results over a large search space by not constraining the contact order. Furthermore, this method frequently leads to solutions that are better conditioned and easily tractable [70].

Another tool used to model the contacts was the use of a friction cone. This cone displays the directions in which force can act without sliding - but once force exceed the cone area, sliding occurs. In this study, Coulomb Friction was adopted to present the sliding forces. Because Coulomb friction apply forces tangent to the contact surface (between the ground and foot), the GRF consist of an X (tangential) and Y (perpendicular) component as seen in Equation 23. The Y component is represented by $\mathbf{GRF_Y}$ while the X component consists of a positive component (GRF_X^+) and negative component (GRF_X^-). The separation in positive and negative magnitudes were done to handle contact interactions effectively, ensuring stability and accurate modelling of complex problems.

Furthermore, since the frictional force is direction-dependent, the tangential foot

velocity, $\mathbf{V_{FootX}}$, in the X-direction, was decomposed into its positive and negative components, as shown in Equation 24. These components were then utilized to define the friction cone, as specified in Equation 25. In this study, the value of $\mu = 0.8$. This value was estimated based on the material properties of the robot's feet, which are made of rubber, and the ground surface, composed of gum rubber—suggesting a high coefficient of friction.

$$\begin{gathered}\mathbf{GRF} = [GRF_X^+ - GRF_X^-, \mathbf{GRF_Y}]^T \\ GRF_X^+, GRF_X^- >= 0\end{gathered} \tag{23}$$

$$\begin{gathered}\mathbf{V_{FootX}} = V_{FootX}^+ - V_{FootX}^- \\ V_{FootX}^+, V_{FootX}^- >= 0\end{gathered} \tag{24}$$

$$\mu\mathbf{GRF_Y} - GRF_X^+ - GRF_X^- >= 0 \tag{25}$$

Equations 23, 24, and 25 were used to derive the complementarity constraints to model the inelastic collisions [80]. The below points provide a short description of each complementarity constraint in Equation 26 :

1. GP (ground penalty variable) is the resulting penalty when adding all penalties of the complementarity ground constraints (contact, friction, and slip).
2. $GP[contact]$ ensures the foot position in the Y direction ($\mathbf{PF_Y}$) can only experience a GRF when in contact with the ground, and vice versa.
3. $GP[friction]$ enforces the friction when there is no foot velocity in the X-direction.
4. $GP[slip^+]$ handles the slipping in the X-direction for a positive foot velocity value.
5. $GP[slip^-]$ handles the slipping in the X-direction for a negative foot velocity value.

$$\begin{gathered}GP = GP[contact] + GP[friction] + GP[slip^+] + GP[slip^-] \\ \mathbf{PF_Y GRF_Y} <= GP[contact] \\ (V_{FootX}^+ + V_{FootX}^-)(\mu\mathbf{GRF_Y} - (GRF_X^+ - GRF_X^-)) <= GP[friction] \\ V_{FootX}^+ GRF_X^+ <= GP[slip^+] \\ V_{FootX}^- GRF_X^- <= GP[slip^-]\end{gathered} \tag{26}$$

These constraints were applied to both of the robot's feet (leg A and leg B). The goal of the solver was to minimize GP to zero in the cost function, to achieve an optimal solution. Moreover, to aid the solver, relaxation techniques were adopted using inequalities to increase the feasibility region - at the cost of reducing the model accuracy [81]. To overcome this, a bound was set on the maximum allowable penalty region, where $0 < GP < 10^{-6}$.

Additional Constraints

Baleka II makes use of a closed kinematic chain whereby the angle points or the left and right link are connected. Specifically, for each leg, the left lower link connects to the right lower link. Equation 10 and 27 represents the connection constraints which ensures the net force between the connected points equates to zero and that the lower links of each respective foot position is connected (refer to Figure 17), respectively.

$$\begin{aligned} \mathbf{ConnectPoint_{RightX}} - \mathbf{ConnectPoint_{LeftX}} &= 0 \\ \mathbf{ConnectPoint_{RightY}} - \mathbf{ConnectPoint_{LeftY}} &= 0 \end{aligned} \tag{27}$$

In addition, to ensure no singularities arise and that the robot's feet are always below the body height, a foot-body constraint was applied as shown in Equation 28.

$$\mathbf{PF_Y} <= \mathbf{Y_b} \tag{28}$$

5.1.5 Cost Function

Previous research on legged locomotion using TO has emphasized energy efficiency [16], [82]. In this study, however, energy efficiency was not the central focus, though it still influenced the selection of optimal solutions. As a result, only two parameters were incorporated in the cost function, namely: ground penalty (GP) and torque squared. The GP term was evaluated in the above chapter (represented in Equation 29), while the torque squared term was included to minimize excessive torque demands (represented in Equation 30). This approach encouraged smoother torque profiles, promoting more efficient motion.

$$J_{GP} = \rho \sum_{n=1}^{N} \sum_{GP=contact}^{[friction, slip^{+}, slip^{-}]} GroundPenalty[GP, n] \tag{29}$$

$$J_T = \sum_{n=1}^{N} \frac{hm \times h[n] \mathbf{T_u}^2}{9.81 \times TM} \tag{30}$$

To improve solving time, a warm-start approach was adopted. This method involved initially solving the optimization using only the GP parameter in the cost function, followed by re-solving the simulation with both the GP and torque squared parameters included. The GP cost function was scaled by ρ (where $\rho = 10^3$) to ensure that the weighting of the penalty was prioritized over torque reduction. Furthermore, to reduce the complexity of the problem, both cost functions were only evaluated at node points rather than intermediate collocation points.

5.1.6 Summary

The TO simulations for the Baleka II bipedal robot were implemented in Python using Pyomo and solved with IPOPT, leveraging variable time-step integration to accurately model foot-ground collisions. The system's dynamics were derived using Euler-Lagrange equations, with a detailed representation of boom, motors, leg linkages, and rubber foot deflection via spring-damper models. Implicit integration methods—BWE for fast solutions and Radau IIA for higher accuracy—were employed, with BWE used for initial feasibility and Radau for refined optimization. Discretized states and controls enabled constraint-based ODE integration.

Key design choices included bounding optimization variables such as knee angles (10°–170°), foot deflection ($\pm$0.01 m), and variable time step (0.5 - 1.2 seconds) to ensure physical realism and solver stability. Motor torque-speed constraints were modelled linearly based on AK10-9 motor specs to avoid overheating (limiting the peak torque to 30 Nm). Contact interactions were handled via through-contact optimization using complementarity constraints, Coulomb friction ($\mu = 0.8$), and force decomposition to model complex interactions without preset contact order. Additional constraints ensured kinematic chain closure and maintained feasible foot-body positioning. The cost function minimized ground penalties and torque squared to promote smooth, realistic motion rather than strict energy efficiency.

5.2 Validation Tests

Validation tests were crucial in this study to assess whether the system's dynamics were implemented correctly. Three validation tests were conducted, namely: high drop, low drop, and leg rotation tests. The Pyomo solutions of these tests are represented as interpolated solutions - to allow a comparable baseline between the Simscape and Simulink solutions. Details surrounding the interpolation process are discussed in Chapter 6.3.

5.2.1 High Drop Test

The high drop test was conducted to assess the correctness of modelling the robot's dynamics. This was done by initializing: $\mathbf{Y_b} = 0.95$ m, velocity states set to zero ($\dot{\mathbf{q}}[1, 0] = 0$), and feet positions set to $(PF_X; PF_Y) = (0; 0.7)$. Furthermore, all torque control states were set to zero for the entire trajectory. This test was simulated for $TT = 0.3$ seconds using Radau at $N = 50$.

Figure 21, displays the resulting high drop test curves from both the Pyomo model and Simscape models. However, this discussion will only focus on the Pyomo results (as the Simscape results will be discussed in the upcoming sections). It's observed that $\mathbf{X_b}, \dot{\mathbf{X}}_\mathbf{b}, \ddot{\mathbf{X}}_\mathbf{b}, \mathbf{T_{URA}}, \boldsymbol{\Delta}$, and $\mathbf{GRF_Y}$ all remains zero, as expected since there was no motion in the X-direction, no actuation commands were specified, nor was there any ground contact as shown by $\mathbf{PF_Y}$. Furthermore, both the $\mathbf{Y_b}$ and PF_Y (of each foot) descends in the Y-direction as expected, with a constant body-foot distance. In addition, $\boldsymbol{\Theta}_\mathbf{URA}$ remained unchanged, as anticipated, given that the body orientation remained constant throughout the motion.

Furthermore, $\dot{\mathbf{Y}}_\mathbf{b}$ increases negatively when dropped while $\ddot{\mathbf{Y}}_\mathbf{b} > 9.81$ m/s^2 as anticipated. This is highlighted in Equation 31, indicating that $\ddot{\mathbf{Y}}_\mathbf{b}$ equates to $9.81 \times \dfrac{r_b}{r_{wb}}$ as the robot was mounted on the boom tip with a fixed base point (similar to a pendulum). Here, r_{wb} and r_b represents the distance between the boom-base to the CoM of the combined boom-robot, and the distance between the boom-base to body (where $r_b > r_{wb}$), respectively. In addition, $\mathbf{a_t}$ and $\mathbf{A}$ represents the tangential acceleration at the CoM (of the combined boom-body system) and angular acceleration of the boom in the pitch axis, respectively. Therefore, given that the results present stable and feasible solutions - the model was deemed acceptable.

$$\begin{aligned} \mathbf{a_t} &= \mathbf{A} \times r \\ 9.81 &= \mathbf{A} \times r_{wb} \\ \ddot{\mathbf{Y}}_\mathbf{b} &= \mathbf{A} \times r_b \\ \ddot{\mathbf{Y}}_\mathbf{b} &= 9.81 \times \frac{r_b}{r_{wb}} \end{aligned} \tag{31}$$

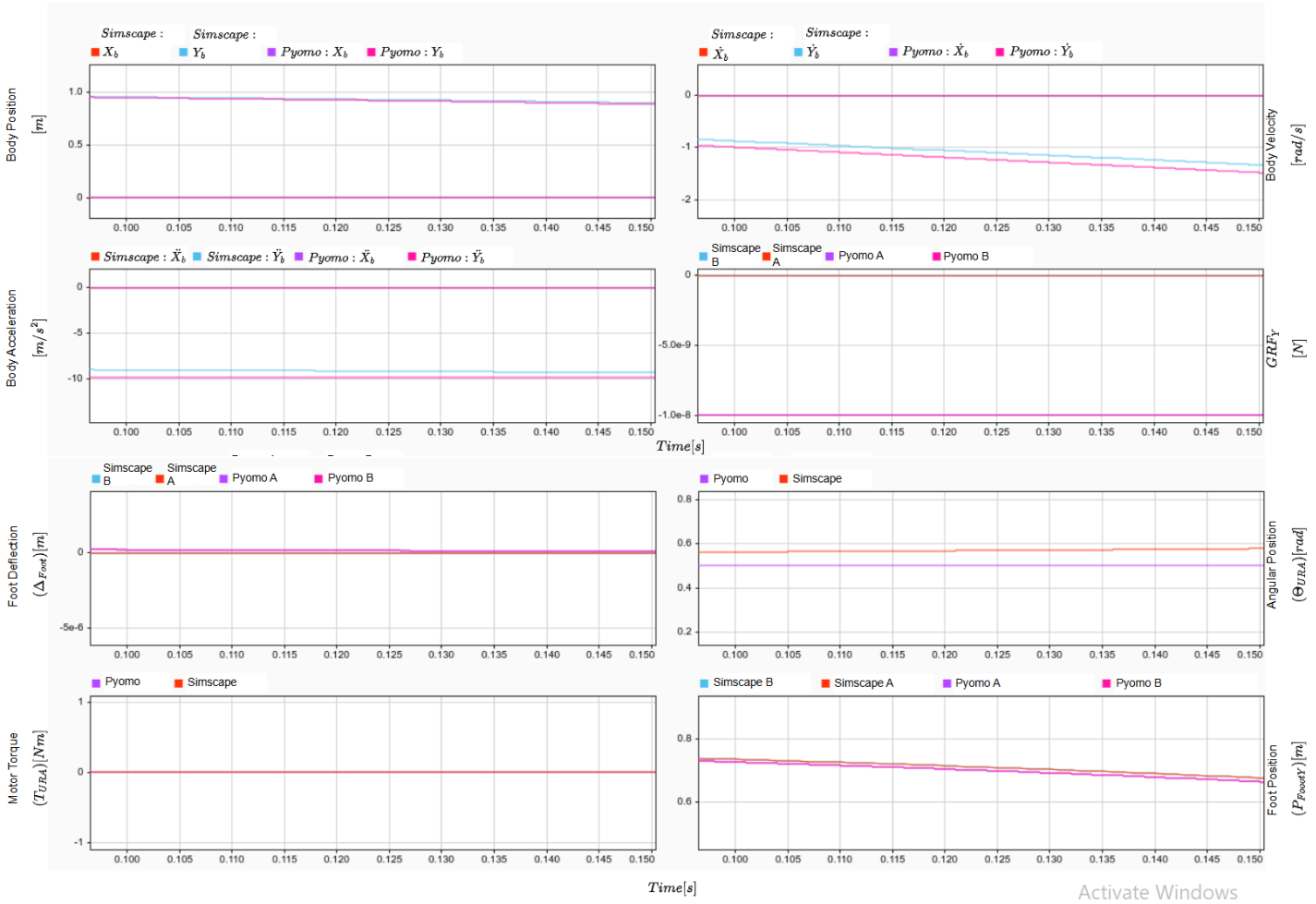

Figure 21: High Drop Test curves resulting from both Pyomo and Simscape simulations. All symbols which ends in "A" and "B" refers to the front and back leg, respectively. The characteristics of the upper right link of leg A (URA) are only displayed for ease of reference.

5.2.2 Low Drop Test

The low drop test was conducted to assess the correctness of modelling the contact between the feet and ground. This was done by initializing: $\mathbf{Y_b} = 0.5$ m, velocity states set to zero ($\dot{\mathbf{q}}[1,0] = 0$), and feet positions set to $(PF_X; PF_Y) = (0; 0.3)$. This test was simulated for $TT = 0.3$ seconds (such that each foot contacts the ground) using Radau at $N = 20$.

Figure 22, displays the resulting low drop test curves from the Pyomo and Simscape models. It's observed that $\mathbf{X_b}, \dot{\mathbf{X}}_\mathbf{b}, \ddot{\mathbf{X}}_\mathbf{b}$, and $\mathbf{T_{URA}}$ remains zero as expected since there's no motion in the X-direction, nor actuation commands. However, each foot

makes contact with the ground ($\mathbf{PF_Y} = 0.0$) at $t = 0.22$ seconds, resulting in a direction change for $\dot{\mathbf{Y}}_\mathbf{b}$ and $\ddot{\mathbf{Y}}_\mathbf{b}$ due to the disturbance experienced during the robot's free-fall. At the same time, the $\mathbf{GRF_Y}$ peaks at ≈ 600 N and the $\mathbf{\Delta}$ changes within the specified bounds. The difference in contact time between $\mathbf{PF_Y}$ and $\mathbf{GRF_Y}$ was one time step due to solving complexity of occurring at the same node (should the GRF only start when $\mathbf{PF_Y} = 0$, it would have to immediately stop the foot's downward movement in a single moment, which is difficult to achieve accurately).The resulting characteristics occurred as expected, therefore, the model was deemed acceptable.

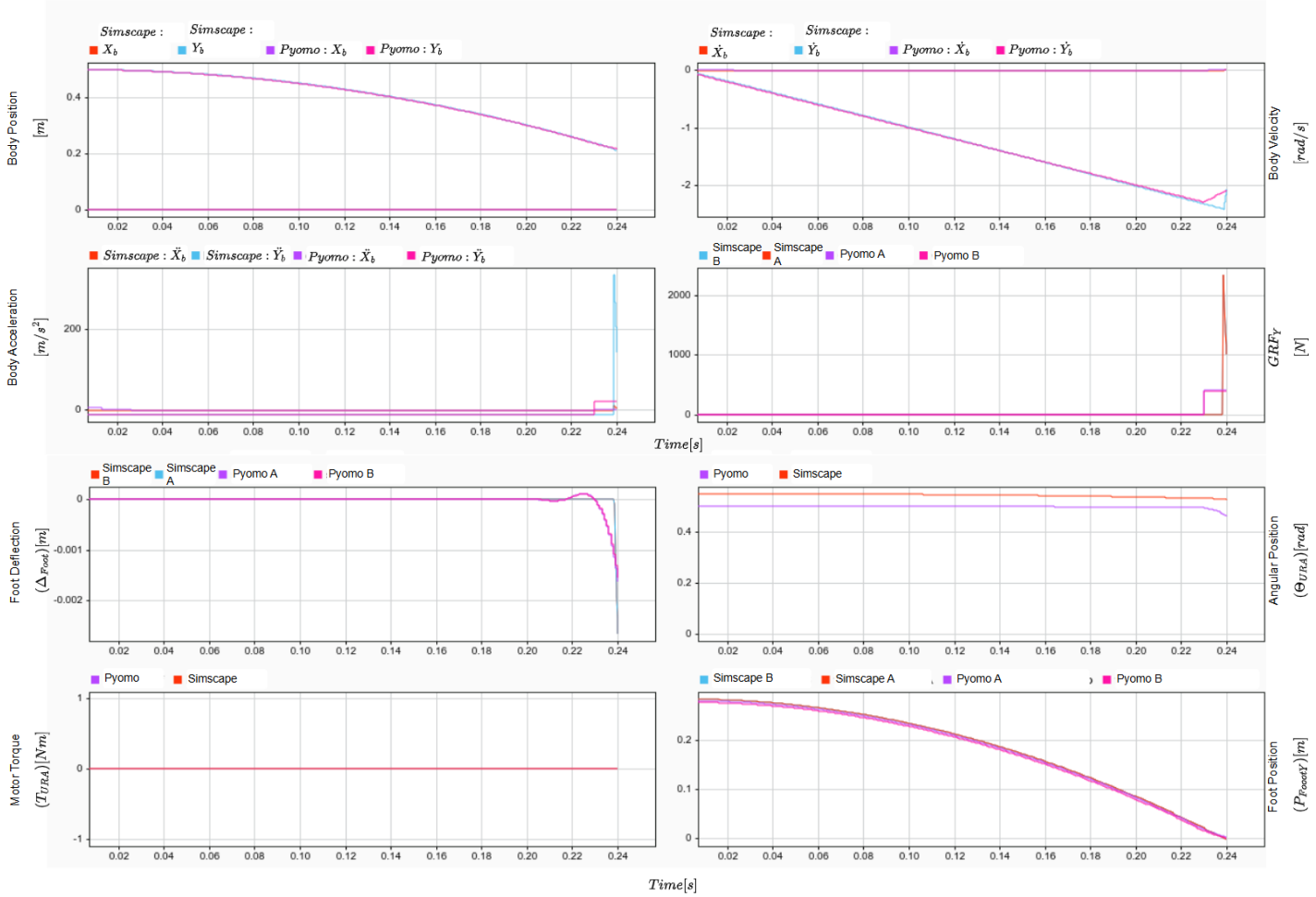

Figure 22: Low Drop Test resulting curves obtained from both Pyomo and Simscape simulations. All symbols which ends in "A" and "B" refers to the front and back leg, respectively. The characteristics of the upper right link of leg A (URA) are only displayed for ease of reference.

5.2.3 Leg Rotation

The leg rotation test was conducted to assess the correctness of modelling the actuators. This was done by initializing: $\mathbf{Y_b} = 0.95$ m, velocity states set to zero ($\dot{\mathbf{q}}[1,0] = 0$), and feet positions set to $(PF_X; PF_Y) = (0.1; 0.7)$. This test was simulated for $TT = 0.3$ seconds using Radau at $N = 30$. Furthermore, the termination requirements were that PF_{XA} < -0.05 m and $PF_{XB} > 0.05$ m, highlighting the need for leg rotation and extension/retraction.

Figure 24, displays the resulting leg rotation test curves from the Pyomo model and Simscape models. It is observed that $\mathbf{\Theta_{ULA}}$ and $\mathbf{\Theta_{URA}}$ exhibit the same curve shape and $PF_{XB} > 0.05$ m, which was expected, as both upper links must rotate counter-clockwise to ensure PF_{XA} < -0.05 m. Furthermore, it's observed that the optimal strategy for using $\mathbf{T_{URA}}$ was to apply a high actuation effort at the beginning of the simulation and then reduce it once the termination conditions were satisfied. In addition, Figure 23 shows that the torque-speed relationship was compliant - falling within the boundary limits. Thus, the actuation model was deemed acceptable.

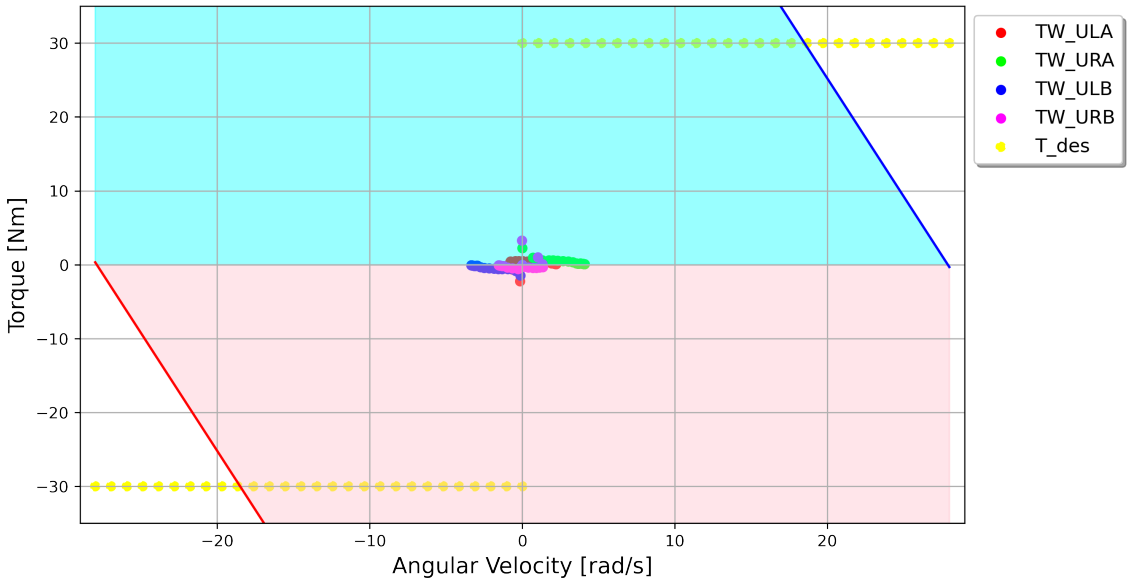

Figure 23: Torque Speed curve highlighting that all four actuation commands are well within the bounds.

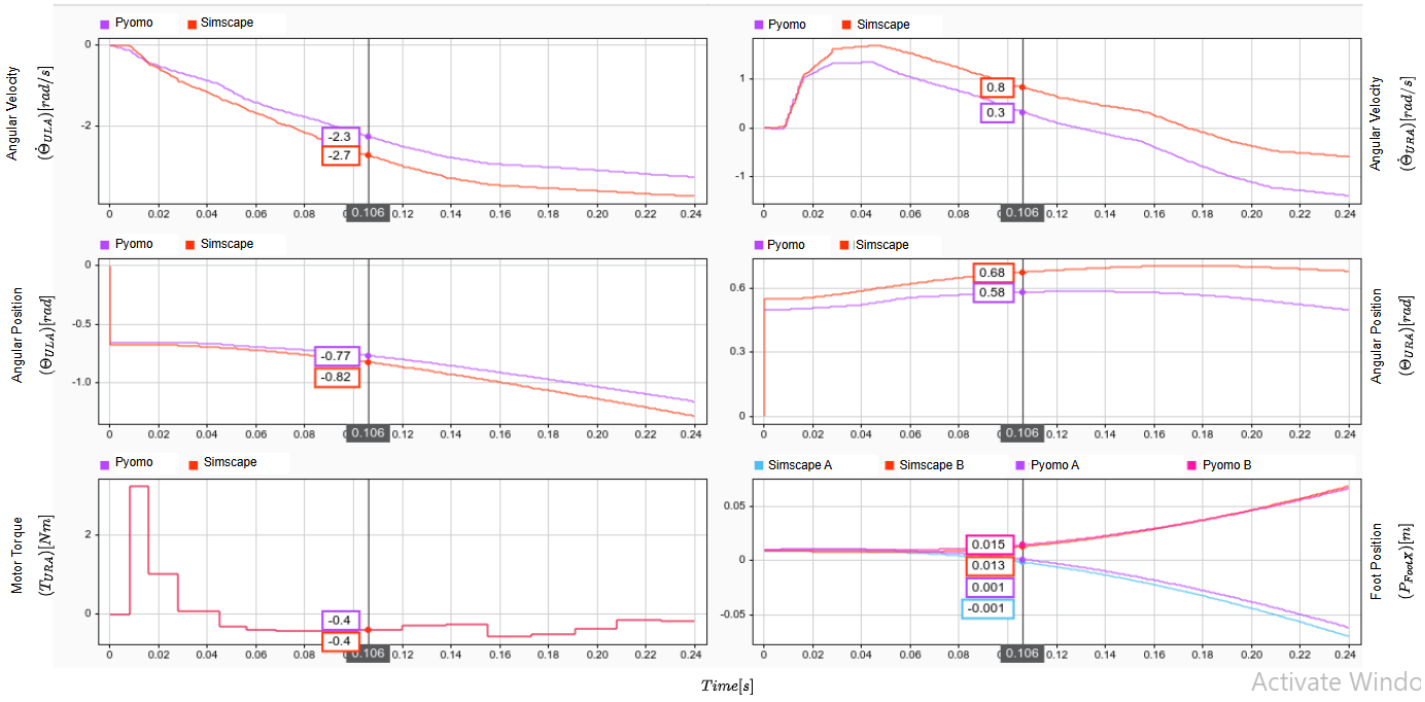

Figure 24: Leg Rotation Test resulting curves obtained from both Pyomo and Simscape simulations. All symbols which ends in "A" refers to the front leg and "B" refers to the back leg. Note, the bottom right curve only displays the foot positions in the X-direction.

5.2.4 Summary

Three validation tests—high drop, low drop, and leg rotation—were performed to confirm the correctness of the system's dynamic modelling. The high drop test validated free-fall dynamics with no actuation or ground contact, showing expected motion in the vertical direction only, and confirmed the model's pendulum-based acceleration behaviour. The low drop test assessed contact dynamics, with each foot correctly impacting the ground at 0.22 seconds, leading to expected changes in velocity, acceleration, and ground reaction force, confirming the model's ability to capture foot-ground interaction. The leg rotation test evaluated actuator modelling, where the legs rotated as expected to meet termination conditions, torque inputs followed an optimal pattern, and all torque-speed outputs remained within bounds. Interpolated Pyomo results aligned well with expected physical behaviour, validating the implementation of dynamics, contact, and actuation across all tests.

5.3 Locomotion Simulation Tests

This study investigates different locomotive tasks including acceleration, steady-state gaits at different velocities, and deceleration. Figure 25 highlights the complete

stitched gait cycle: the robot starts at rest, accelerates from $t = t[1] = 0$ seconds to $t = t_a$, enters a steady-state walk or run from $t = t_a$ to $t = t_d$, and decelerates back to rest from $t = t_d$ to $t = TT$. To implement this *stitching* approach, the system's state positions and state velocities for each locomotive task, were overlapped as per Equation 32. Specifically, the final states of the acceleration task equates to the initial states of the steady-state locomotive task. Similarly, the final states of the steady-state task equates to the initial states of the deceleration task. In both instances: $\mathbf{X_b}$, $\mathbf{\Delta}$, and $\dot{\mathbf{\Delta}}$ were excluded from this constraint.

$$\begin{gathered}\text{Acceleration:} \\ \mathbf{q_{SteadyState}}[1,0] = \mathbf{q_{Acceleration}}[N,0] \\ \dot{\mathbf{q}}_{\mathbf{SteadyState}}[1,0] = \dot{\mathbf{q}}_{\mathbf{Acceleration}}[N,0] \\ \text{Deceleration:} \\ \mathbf{q_{SteadyState}}[N,0] = \mathbf{q_{Deceleration}}[1,0] \\ \dot{\mathbf{q}}_{\mathbf{SteadyState}}[N,0] = \dot{\mathbf{q}}_{\mathbf{Deceleration}}[1,0]\end{gathered} \tag{32}$$

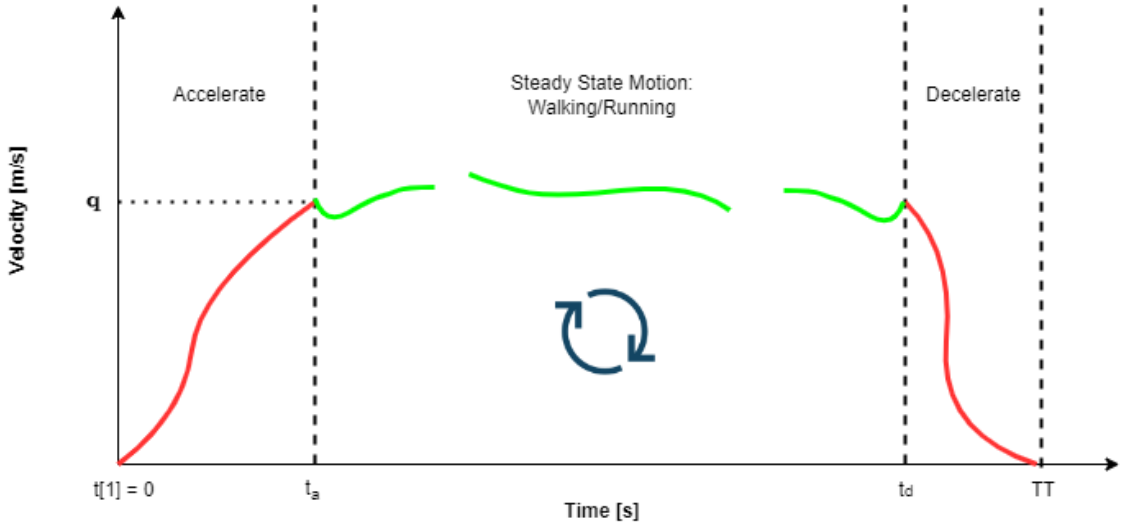

Figure 25: Representation of the complete gait cycle: The robot starts at rest, accelerates into a walking or running gait, and then decelerates back to rest.

Throughout the steady-state gait cycle, the robot's feet makes and breaks contact with the ground. In legged robots, locomotion is classified as walking when at least one foot maintains contact with the ground at all times. Conversely, it is classified as running when all feet momentarily leave the ground (flight phase). To determine the walk and walk-to-run transition velocities - an FN of 0.05 and 0.5 was used

(Equation 1), respectively. With the maximum leg length of 0.45 m, the walking velocity was calculated at 0.5 m/s. Moreover, the resulting walk-to-run transition velocity is 1.5 m/s as the FN exceeds 0.5 (Equation 33). Therefore, in this study - when $\dot{\mathbf{X}}_\mathbf{b} < 1.5$ m/s it's considered walking and when $\dot{\mathbf{X}}_\mathbf{b} > 1.5$ m/s it's considered running. Furthermore, to obtain the achievable maximum velocity - simulations were done with increasing the desired velocity, up to the point where simulations resulted infeasible.

$$FN = \frac{1.5^2}{9.81 \times 0.45} \approx 0.51 \tag{33}$$

The upcoming three sub-chapters dive into the TO results for the steady-state gait motions, acceleration, and deceleration tasks. For each task, the model's parameter values were setup once the EoM were determined. The setup parameters varied in: total nodes, $\dot{\mathbf{X}}_{\mathbf{bDes}}$, $\mathbf{\Theta}_{\mathbf{KneeAngle}}$ bounds, and h bounds. Other parameters were fixed such as: mass, inertia, link length, N_{GR}, D_{Foot}, K_{Foot}, collocation points, μ, $\mathbf{g}$, maximum and minimum foot deflection ($\mathbf{\Delta}$), motor model, and the limitations on the connection forces and GRFs. All solutions presented in this chapter can be found here: **Trajectory Optimization Solutions**.

In this study, contact orders were specified for a selected few gait patterns - to aid the solver. Therefore, many BWE simulations were solved with varying the setup parameters without specifying a contact order. Thereafter, of the many simulated solutions, the *best solution*'s contact order was selected and implemented to solve trajectories using Radau. The *best optimal solution* consist of the following categories:

1. **Category A**: motion which completes the task in least amount of time (agility).
2. **Category B**: lowest J_T value - referencing the most efficient solution with least amount of actuation effort.
3. **Category C**: lowest J_{GP} value - indicating the solution that best satisfies the complementarity contact conditions.
4. **Category D**: lowest $\mathbf{T_P}$ (peak torque) command value of all the motors - facilitating additional headroom for control torque of PD controller (also reduces overheating of motors).
5. **Category E**: the smallest *maximum* $\mathbf{\Theta}_{\mathbf{KneeAngle}}$ experienced ($\mathbf{\Theta}_{\mathbf{MKA}}$). This reduces the probability of a singularity - due to over/undershooting while tracking desired inputs.

6. **Category F**: lowest $\mathbf{GRF_Y}$ value - indicating the least stress/damage on the foot and ankle joint during contact.
7. **Category G**: for steady-state periodic motions, $\dot{\mathbf{X}}_\mathbf{b}$ with the least amount of variation from the desired velocity ($\dot{\mathbf{X}}_\mathbf{bDes}$) - reducing acceleration/deceleration and promoting stable steady-state motions.
8. **Category H**: for acceleration and deceleration tasks, solutions with the highest node (N) value - highlighting accuracy and stability in implementation.

5.3.1 Steady-State Walking and Running

To enforce that a constant $\dot{\mathbf{X}}_\mathbf{b}$, an average velocity constraint was implemented as shown in Equation 34. This equation was defined such that the average velocity of the robot's body for the total time is equal to the desired velocity (($\dot{\mathbf{X}}_\mathbf{bDes}$).

$$\dot{\mathbf{X}}_\mathbf{bDes} = \frac{\mathbf{X_b}[N] - \mathbf{X_b}[n=1]}{\sum_{n=1}^{N} hm \times h[n]} \tag{34}$$

Due to the symmetry of Baleka II, only half of the gait cycle needed solving, as the second half mirrors the first with leg positions and velocities, swapped. This significantly reduced the solving load. To ensure periodic conditions, the position and velocity states of leg A at the first node had to match those of leg B, and vice versa [82]. This is represented in Equation 35 (where $\mathbf{q_A}/\dot{\mathbf{q}}_\mathbf{A}$ and $\mathbf{q_B}/\dot{\mathbf{q}}_\mathbf{B}$ refers to the state positions/velocities of leg A and leg B, respectively) and in Figure 26, representing the same pattern but for different legs. Figure 27 provides a visual representation of a 4.0 m/s run half stride - highlighting that at n = 1 and n = N, the orientations of the legs are mirrored.

$$\begin{aligned} \mathbf{q_A}[n=1] &= \mathbf{q_B}[N] \\ \mathbf{q_B}[n=1] &= \mathbf{q_A}[N] \\ \dot{\mathbf{q}}_\mathbf{A}[n=1] &= \dot{\mathbf{q}}_\mathbf{B}[N] \\ \dot{\mathbf{q}}_\mathbf{B}[n=1] &= \dot{\mathbf{q}}_\mathbf{A}[N] \end{aligned} \tag{35}$$

After the half gait was solved, the position and velocity trajectories for each leg was then switched for the second-half of the gait cycle (except for $\mathbf{X_b}$ and $\boldsymbol{\Delta}$ as the body moves forward and due to the deformation nature of contacts, respectively). Furthermore, an approach used to ease the process of switching and combining the half strides to generate a long time horizon, a torque constraint was applied (Equation 36). Where $\mathbf{T_A}$ and $\mathbf{T_B}$ refers to torques of leg A and leg B, respectively.

Furthermore, $i = 1, 2, 3, 4, 5$, such that the torque transition was smooth (reducing the torque difference when doing the state swap).

$$\begin{aligned} \mathbf{T_A}[n = i] &= \mathbf{T_B}[N - i - 1] \\ \mathbf{T_B}[n = i] &= \mathbf{T_A}[N - i - 1] \end{aligned} \tag{36}$$

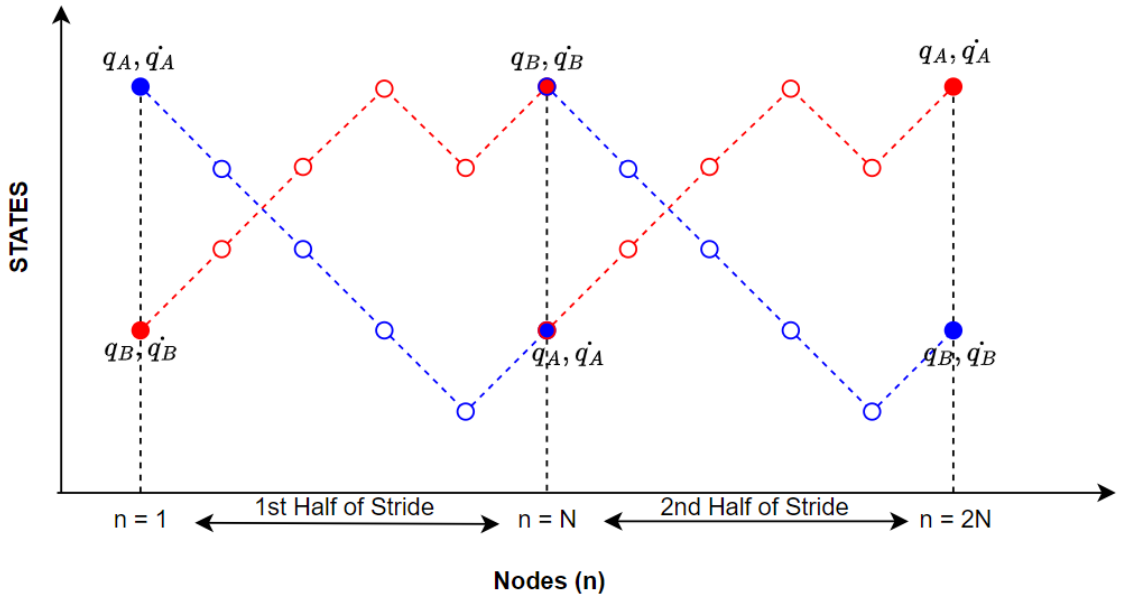

Figure 26: Graphical depiction of the state-like behaviour of Legs A and B under periodic conditions throughout a complete gait cycle.

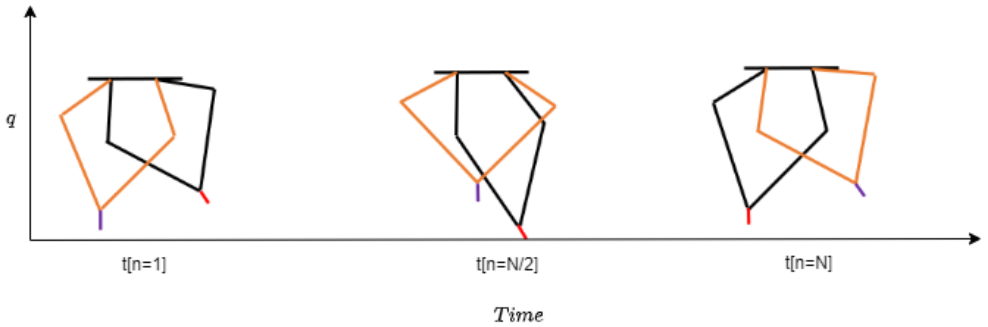

Figure 27: Animation of a half stride of a 4.0 m/s run (within Pyomo) - the states of leg A (orange) at $n = 1$ equates to the states of leg B (black) at $n = N$.

Table 4 displays the parameters used to setup *Tests* for $\dot{\mathbf{X}}_{\mathbf{bDes}} = 0.5$ m/s, 1.5 m/s, and 4.0 m/s. The resulting optimal solutions which produced the *best results* using

BWE are displayed in Table 5. Here, 4.0 m/s was deemed the fastest achievable body velocity - any velocity above 4.0 m/s were infeasible. This is further supported using Table 5, which highlights that the torque and knee angle values were close to reaching their respective bounds when increasing $\dot{\mathbf{X}}_{\mathbf{bDes}}$. In addition, the variation in $\dot{\mathbf{X}}_{\mathbf{b}}$ increases as $\dot{\mathbf{X}}_{\mathbf{bDes}}$ increases - indicating higher oscillations and reduced stability of steady-state locomotion. At 4.0 m/s the variation observed is 0.38 m/s.

For all simulations, the total number of nodes for each BWE simulation was $N = 100$ and the time-step bounds were between $0.5 < h < 1.0$. Furthermore, no contact order was specified, only initialization requirements were setup at n=1, these were: $\mathbf{PF_{XA}}, \mathbf{PF_{XB}} <= 0.05$ m and $\mathbf{X_b}, \boldsymbol{\Delta_A}, \boldsymbol{\Delta_B}, \dot{\boldsymbol{\Delta}}_{\mathbf{A}}$, and $\dot{\boldsymbol{\Delta}}_{\mathbf{B}}$ were all set to zero.

Table 4: Table representing setup parameters of five optimal solutions for $\dot{\mathbf{X}}_{\mathbf{bDes}} = 0.5$ m/s and 1.5 m/s. However, only two optimal solutions were obtained for the 4.0 m/s run.

Test	$\dot{\mathbf{X}}_{\mathbf{bDes}}$(m/s)	**TT** (s)	**Maximum** $\boldsymbol{\Theta}_{\mathbf{KneeAngle}}$ (°)
1	0.50	0.40	150.00
2	0.50	0.60	160.00
3	0.50	0.70	145.00
4	0.50	1.00	145.00
5	0.50	1.40	150.00
1	1.50	0.30	145.00
2	1.50	0.60	150.00
3	1.50	1.00	160.00
4	1.50	1.20	150.00
5	1.50	1.50	160.00
1	4.00	0.40	160.00
2	4.00	0.60	160.00

Table 5: Representation of simulation parameters obtained from optimal half gait simulations for 0.5 m/s, 1.5 m/s and 4.0 m/s using BWE integration. Here, $\dot{\mathbf{X}}_\mathbf{b}$ highlights the minimum and maximum body velocity experienced.

Test	$\dot{\mathbf{X}}_{\mathbf{bDes}}$ (m/s)	$t[N]$ (s)	J_T	J_{GP}	$\mathbf{T_P}$ (Nm)	$\mathbf{\Theta_{MKA}}$ (°)	$\mathbf{GRF_Y}$ (N)	$\dot{\mathbf{X}}_\mathbf{b}$ (m/s)
1	0.50	0.30	0.20	-5.16×10^{-6}	-11.29	150.00	877.60	0.40-0.57
2	0.50	0.36	0.67	-4.83×10^{-6}	-20.98	75.27	473.08	0.43-0.60
3	0.50	0.40	0.39	-4.84×10^{-6}	-21.01	145.00	711.61	0.40-0.71
4	0.50	0.50	0.32	-5.19×10^{-6}	11.13	145.00	437.37	0.37-0.64
5	0.50	0.69	1.08	-5.16×10^{-6}	-14.52	140.39	378.96	0.37-0.65
1	1.50	0.16	0.34	-5.10×10^{-6}	15.55	145.00	915.26	1.40-1.61
2	1.50	0.36	2.81	-5.06×10^{-6}	-29.80	142.20	165.56	0.98-2.08
3	1.50	0.55	2.84	-5.23×10^{-6}	29.80	159.87	399.42	1.21-1.80
4	1.50	0.60	2.53	-5.26×10^{-6}	30.00	150.00	915.27	1.01-2.06
5	1.50	0.74	3.36	-5.26×10^{-6}	30.00	160.00	915.24	0.97-1.74
1	4.00	0.21	0.96	-5.26×10^{-6}	27.57	160.00	915.23	3.62-4.30
2	4.00	0.36	2.68	0.00	30.00	141.95	915.26	3.49-4.83

When investigating the *best solution* at 0.5 m/s, maximum knee angles and actuation effort were prioritized. From Table 5 (Tests 1-5 for $\dot{\mathbf{X}}_{\mathbf{bDes}} = 0.5$ m/s), there's no clear *best solution* as there's a trade-off between time and force (actuation effort and GRF). Test 4 was selected as the overall *best solution* - it performs best in Category D but ranks second in Category B, F, and G when compared to the other four tests. Here, the low $\mathbf{T_P}$, $\mathbf{GRF_Y}$, and $\mathbf{\Theta_{MKA}}$ was favoured as it provides the most stable/safest solution. Thus, Figure 28 displays the contact order of Test 4 (left). Furthermore, confirms that there's at least always one foot in contact with the ground, indicating a walking motion.

Test 1 was selected as the *best solution* from Table 5 (Tests 1-5 for $\dot{\mathbf{X}}_{\mathbf{bDes}} = 1.5$ m/s), as it performs best in Category A, B, D, and G (and second best in both Category C and E). Although this solution was the fastest, most efficient, and safest (knee angle) - it experienced the highest $\mathbf{GRF_Y}$ (stress on the mechanical mechanisms). Thus, Figure 28 displays the contact order of Test 1 (middle). The contact order confirms that there's a flight phase within the gait cycle (between node 70-100), indicating a running/jogging motion.

Lastly, the solver could only produce two optimal solutions (Tests 1-2) as shown in Table 5 with $\dot{\mathbf{X}}_{\mathbf{bDes}} = 4.0$ m/s. Test 1 was selected as the *best solution*, as it performed best in Category A, B, D, F, and G. Although Test 2 does display a lower $\mathbf{\Theta_{MKA}}$ - Test 1 was well within the bounds. Therefore, Figure 28 displays the contact order of Test 1 (right). Furthermore, it is observed that the 4.0 m/s solution

exhibits a longer flight phase for the running motion (compared to 1.5 m/s): as seen between node 0-50. This highlights an increase in stride length as the velocity increases.

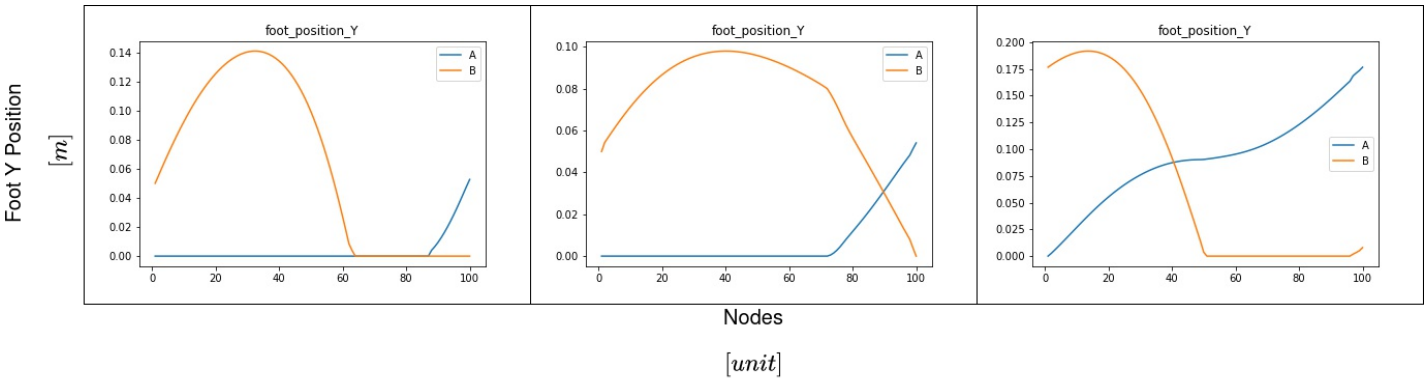

Figure 28: Represents the foot contact order of foot A and B of the *best solutions*. These are: Test 4 where $\dot{\mathbf{X}}_{\mathbf{bDes}} = 0.5$ m/s (left), Test 1 where $\dot{\mathbf{X}}_{\mathbf{bDes}} = 1.5$ m/s (middle), and Test 1 where $\dot{\mathbf{X}}_{\mathbf{bDes}} = 4.0$ m/s (right) over 100 nodes.

Although BWE integration produced optimal results - the Radau approach was adopted as it has an improved solution accuracy. The contact orders for the *best solutions* described in Figure 28 were used to solve the Radau integration solutions. Here, the setup parameters were tuned to obtain optimal results (including the contact order). Furthermore, a seeding approach was used to ensure alternative areas in search space were investigated. In addition, N was set to 50 nodes and the time step bounds were between $0.5 < h < 1.2$. The below list provides an overview of the setup parameters and contact orders for each simulation.

Algorithm 1 Setup Conditions for 0.5 m/s

TT = 0.7,
Maximum $\mathbf{\Theta_{KneeAngle}} = 160$
$\dot{\mathbf{X}}_{\mathbf{bDes}} = 0.5$
At n = 1
    $\mathbf{X_b} = \mathbf{\Delta_B} = \dot{\mathbf{\Delta}}_{\mathbf{B}} = 0.0$
    $\mathbf{PF_{YB}} > 0.05$
for n = 1 **to** N **do**
  if $n < 0.9 \times N$ **then**
    $\mathbf{PF_{YA}} = \dot{\mathbf{PF}}_{\mathbf{XA}} = 0.0$
  else
    $\mathbf{GRF_{YA}} = 0$
  end if
  if $n < 0.65 \times N$ **then**
    $\mathbf{GRF_{YB}} = 0$
  else
    $\mathbf{PF_{YB}} = \dot{\mathbf{PF}}_{\mathbf{YB}} = 0.0$
  end if
end for

Algorithm 2 Setup Conditions for 1.5 m/s

TT = 0.7
Maximum $\mathbf{\Theta_{KneeAngle}} = 160$
$\mathbf{\dot{X}_{bDes}} = 1.5$
At n = 1
    $\mathbf{X_b} = \mathbf{\Delta_B} = \mathbf{\dot{\Delta}_B} = 0.0$
    $\mathbf{PF_{YB}} > 0.03$
for n = 1 **to** N **do**
    $\mathbf{Y_b} > 0.27$
    if $n < 0.75 \times N$ **then**
        $\mathbf{PF_{YA}} = \mathbf{\dot{PF}_{XA}} = 0.0$
    else
        $\mathbf{GRF_{YA}} = 0$
    end if
    if $n < 0.90 \times N$ **then**
        $\mathbf{GRF_{YB}} = 0$
    else
        $\mathbf{PF_{YB}} = \mathbf{\dot{PF}_{YB}} = 0.0$
    end if
end for

Algorithm 3 Setup Conditions for 4.0 m/s

TT = 0.4
Maximum $\mathbf{\Theta_{KneeAngle}} = 160$
$\mathbf{\dot{X}_{bDes}} = 4.0$
At n = 1
    $\mathbf{X_b} = \mathbf{\Delta_A} = \mathbf{\dot{\Delta}_A} = 0.0$
for n = 1 **to** N **do**
    $\mathbf{Y_b} > 0.25$
    $\mathbf{GRF_{YB}} = 0$
    if $n < 0.65 \times N$ **and** $n > 0.45 \times N$ **then**
        $\mathbf{PF_{YB}} > 0.03$
        $\mathbf{PF_{YA}} = \mathbf{\dot{PF}_{XA}} = 0.0$
    else
        $\mathbf{GRF_{YA}} = 0$
    end if
end for

To broaden the search space, multiple simulations were conducted using random

uniformly distributed seed values. These simulations leveraged the previously solved solution as a reference point, serving as the initial starting point for the optimizer [16]. The following states were randomly initialized:

1. $0 < \mathbf{X_b} < TT \times hm \times \dot{\mathbf{X}}_{\mathbf{bDes}}$ (distance using desired velocity)
2. $0.25 < \mathbf{Y_b} < 0.4$ (limitation on body height)
3. $-1.5\pi < \mathbf{\Theta_{ULA}}; \mathbf{\Theta_{URA}}; \mathbf{\Theta_{ULB}}; \mathbf{\Theta_{URB}} < 1.5\pi$
4. Minimum $\mathbf{\Theta_{KneeAngle}} < \mathbf{\Theta_{KneeAngle}} <$ Maximum $\mathbf{\Theta_{KneeAngle}}$
5. Minimum $\mathbf{\Delta} < \mathbf{\Delta_A}; \mathbf{\Delta_B} <$ Maximum $\mathbf{\Delta}$

Unfortunately, providing random seed values for $\dot{\mathbf{q}}$ produced infeasible results - thus, was not used. Table 6 displays the resulting solutions with different seed values. From the table, it is observed that for the 0.5 m/s walk, loop 4 provides the *best solution* in terms of stabilization (loops 2 and 5 produced results with slipping). Moreover, loop 1 produced the *best solution* for the 1.5 m/s simulation in terms of Category A, B, D, and second best in terms of stability: Category C, E, and F. Although the solution experiences a high variation in $\dot{\mathbf{X}}_{\mathbf{b}}$, it is well within an acceptable range. Lastly, loop 1 was selected as the *best solution* for the 4.0 m/s simulation as it prioritized top speed (Category A).

Table 6: Resulting optimal solution parameters using the seeding approach and Radau integration for 0.5 m/s, 1.5 m/s, and 4 m/s. All variables remain the same as in Table 5 with the inclusion of the seed loop.

Loop	$\dot{\mathbf{X}}_{\mathbf{bDes}}$ (m/s)	$t[N]$ (s)	J_T	J_{GP}	$\mathbf{T_P}$ (Nm)	$\mathbf{\Theta_{MKA}}$ (°)	$\mathbf{GRF_Y}$ (N)	$\dot{\mathbf{X}}_{\mathbf{b}}$ (m/s)
1	0.50	0.41	0.23	0.00	12.34	159.95	199.27	0.41-0.70
2	0.50	0.41	0.23	7.36×10^{-6}	13.40	159.86	29.23	0.41-0.70
3	0.50	0.40	0.23	0.00	13.09	159.89	159.05	0.41-0.70
4	0.50	0.41	0.23	0.00	12.59	159.93	166.44	0.41-0.70
5	0.50	0.36	0.27	0.00	12.17	159.63	131.44	0.42-0.69
1	1.50	0.36	0.77	-3.42×10^{-6}	29.48	160.00	188.05	1.31-1.93
2	1.50	0.37	0.87	-3.41×10^{-6}	30.00	160.00	197.22	1.29-1.92
3	1.50	0.37	0.88	-3.43×10^{-6}	30.00	159.96	239.22	1.30-1.90
4	1.50	0.37	0.88	-3.43×10^{-6}	30.00	160.00	218.16	1.28-1.93
5	1.50	0.37	0.83	-3.51×10^{-6}	30.00	160.00	149.10	1.31-1.92
1	4.00	0.21	1.09	-3.53×10^{-6}	30.00	158.44	271.55	3.93-4.24
2	4.00	0.24	1.51	-3.35×10^{-6}	30.00	154.70	240.51	3.80-4.27
3	4.00	0.22	1.43	-3.49×10^{-6}	30.00	153.25	319.25	3.89-4.23
4	4.00	0.21	1.30	-3.51×10^{-6}	30.00	159.94	271.64	3.99-4.20
5	4.00	0.22	1.31	-3.47×10^{-6}	30.00	150.02	460.71	3.94-4.23

From Table 6, as $\dot{\mathbf{X}}_{\mathbf{bDes}}$ increases, J_T, $\mathbf{T_P}$, and $\mathbf{GRF_Y}$ also rise, driven by increased stride length, swing speed, and higher body velocity and height, which in turn require the feet to manage higher momentum upon contact. However, deviations between $\dot{\mathbf{X}}_{\mathbf{b}}$ and $\dot{\mathbf{X}}_{\mathbf{bDes}}$ worsen due to actuation limits. Additionally, as $\dot{\mathbf{X}}_{\mathbf{bDes}}$ increases, the system approaches the torque-speed boundary, as shown in Figure 29, where all *best solutions* remain within the torque-speed limits.

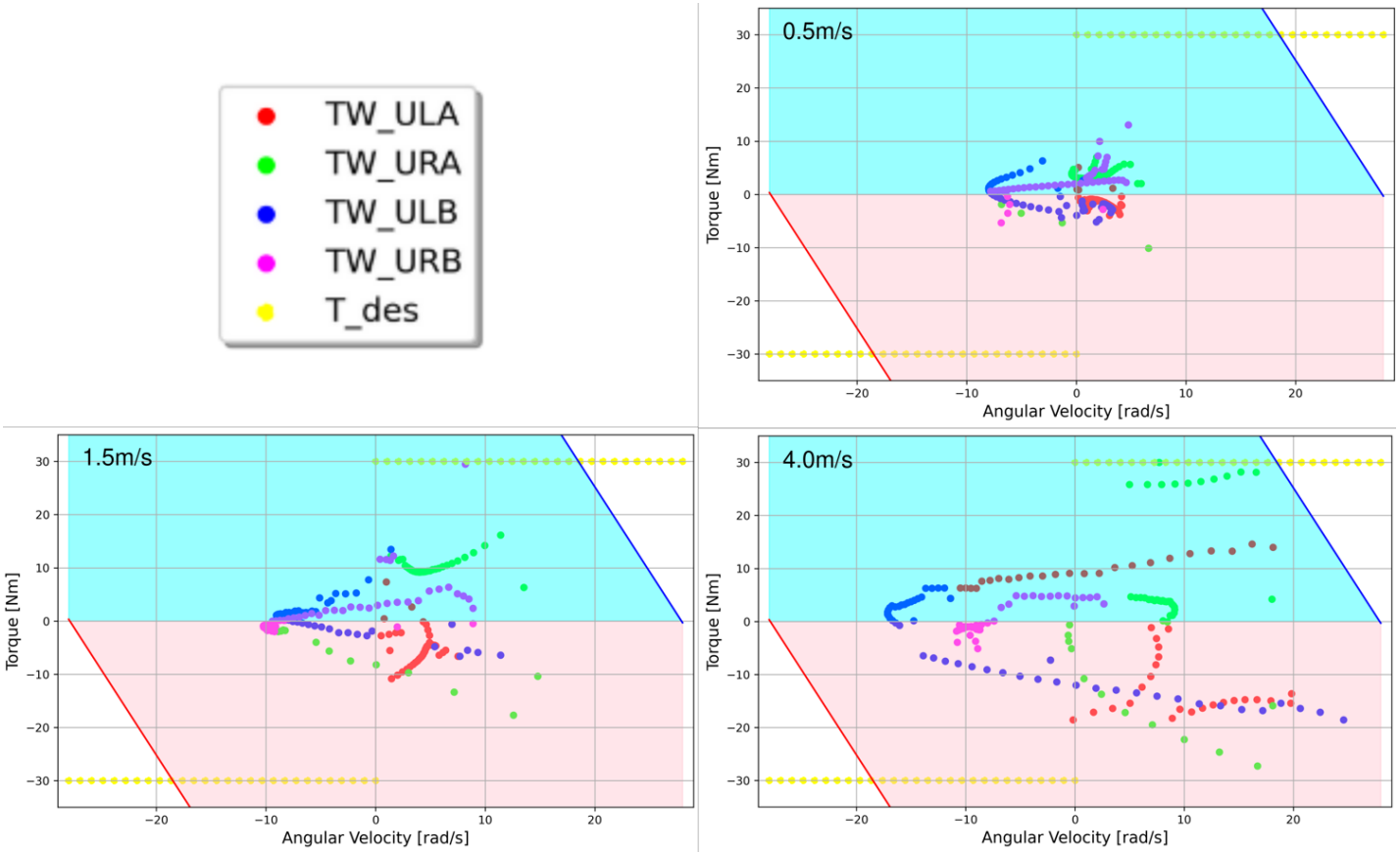

Figure 29: The torque-speed curves for the optimal solutions at 0.5 m/s (top right), 1.5 m/s (bottom left), and 4 m/s (bottom right) are shown for all four motors during half-gait steady-state motion.

5.3.2 Acceleration

The approach used for achieving rapid transition was to *stitch* the acceleration trajectories to the pre-solved steady-state trajectories. Therefore, this section solves solutions based on the *best steady-state solutions* selected previously. While the ideal approach is to equate the final states of the acceleration task with the initial states of the steady-state gait, this poses challenges for the solver. Consequently, a relaxation technique was employed, permitting a 10% margin for state equality. In addition, contact orders were applied for all acceleration tasks - centred around launching the robot into a flight phase.

When solving for the *best acceleration solution* for each $\dot{\mathbf{X}}_{\mathbf{bDes}}$ - a similar approach was taken as above. However, the BWE integration was not implemented - only Radau, with varying total node N values. Here, the time-step bounds were limited between $0.5 < h < 1.2$ to allow for a wider search space. Although rapid transition

was a key aspect to this study, selecting acceleration values with the higher factors of stability was desired (Category G) . Thus, the quickest simulation time wasn't always ideal as high GRF and torque values could cause damage to the robot over many experiments.

Table 7 highlights the resulting optimal solutions which produced the *best acceleration solutions*. The results include five solutions at 0.5 m/s, five at 1.5 m/s, and two at 4.0 m/s.

In the 0.5 m/s simulations, loops 1 and 3, which used the highest number of nodes (N), were compared. Loop 3 excelled in Categories A and E but experienced large GRFs. Conversely, Loop 1 performed best in Categories D and F and was competitive with Loop 3 in the other categories. To avoid the large GRF, loop 1 was selected as the acceleration trajectory for the 0.5 m/s implementation.

On the other hand, loop 1 was selected as the *best acceleration solution* for $\dot{\mathbf{X}}_{\mathbf{bDes}} =$ 1.5 m/s as it use the most nodes N and performed best in Category B and second best in Category D. All remaining values fall within a close range of those from the other loops (2-4).

Lastly, only two optimal results were solved for the 4.0 m/s simulation. Here, top speed was prioritized. Although loop 1 presented the fastest solution - it experienced high levels of slipping. As a result, loop 2 was selected as it's best in Categories B and E and provided a more stable solution (less slipping).

Table 7: Optimal acceleration solutions for 0.5 m/s, 1.5 m/s, and 4.0 m/s. These solutions were obtained by setting the termination conditions as the initial state conditions of the previously solved *best steady-state solutions*. N and TT, the total number of nodes used and allowed total simulation time, respectively.

Loop	$\dot{\mathbf{X}}_{\mathbf{bDes}}$ (m/s)	N	TT (s)	$t[N]$ (s)	J_T	J_{GP}	$\mathbf{T_P}$ (Nm)	$\mathbf{\Theta_{MKA}}$ (°)	$\mathbf{GRF_Y}$ (N)
1	0.50	40	0.60	0.52	1.53	1.46×10^{-5}	-20.80	157.09	165.53
2	0.50	20	0.50	0.36	0.88	-1.35×10^{-6}	18.81	156.29	423.11
3	0.50	50	0.50	0.34	0.97	-3.53×10^{-6}	29.34	155.28	501.20
4	0.50	20	0.50	0.38	0.62	-1.39×10^{-6}	12.72	159.99	249.73
5	0.50	20	0.50	0.37	0.61	-1.38×10^{-6}	12.79	160.00	324.52
1	1.50	50	0.60	0.66	1.09	-3.50×10^{-6}	-19.99	159.99	149.33
2	1.50	40	0.60	0.66	1.17	-2.79×10^{-6}	-19.80	156.07	113.90
3	1.50	40	0.30	0.31	2.59	-2.70×10^{-6}	-30.00	152.11	127.31
4	1.50	40	0.40	0.37	1.87	-2.60×10^{-6}	-30.00	152.47	192.57
5	1.50	40	0.50	0.58	1.52	-2.80×10^{-6}	-30.00	151.37	206.35
1	4.00	30	0.40	0.47	4.49	0.0	30.00	170.00	32.73
2	4.00	30	0.50	0.55	3.36	0.0	30.00	169.73	101.27

From Table 7 it's observed that as the acceleration increases (to reach $\dot{\mathbf{X}}_{\mathbf{bDes}}$), there's an increase in J_T, $\mathbf{T_P}$, and $\mathbf{\Theta_{MKA}}$. This was due to the requirement of launching the robot further and higher to ensure the leg swing aligns with the steady-state conditions. In addition, as $\dot{\mathbf{X}}_{\mathbf{bDes}}$ increases, the closer the torque and speed curve is to the boundary as seen in Figure 30. This curve highlights the torque-speed curve indicates that all solutions are within torque-speed boundary.

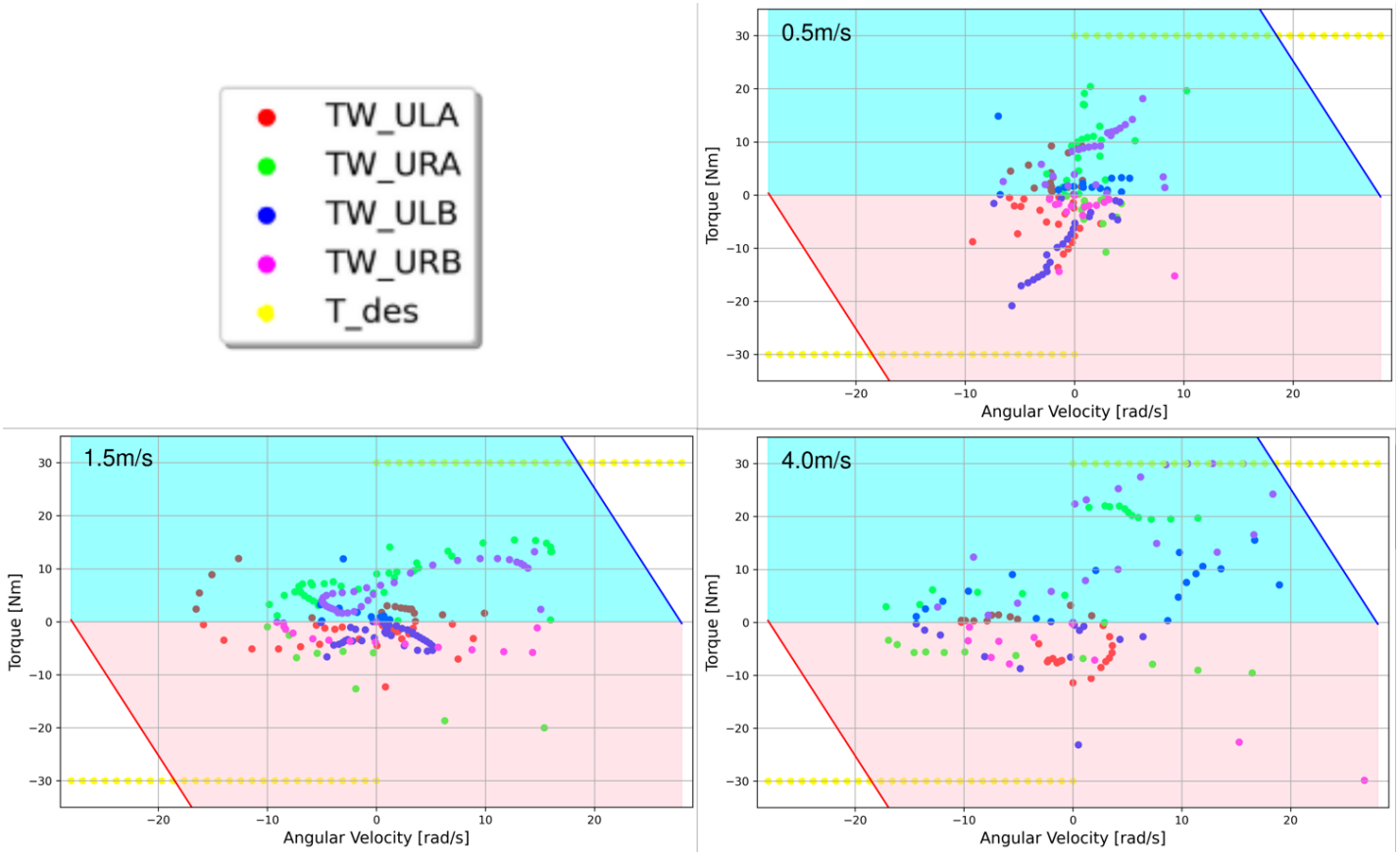

Figure 30: Displays the torque-speed curve of the optimal solutions for the 0.5 m/s (top right), 1.5 m/s (bottom left) and 4.0 m/s (bottom right) of all four motors for acceleration motions.

5.3.3 Deceleration

The deceleration task ensured that the robot's motion was terminated in a stable manner. These trajectories were *stitched* at the end of either gait (in a similar manner to the acceleration task). In addition, a stabilizing condition was set such that the last 20% of the simulation time ($n > 0.8 \times N$), $\dot{\mathbf{q}}$ was set to zero - with the exception of $\dot{\mathbf{X}}_{\mathbf{b}} < 0$.

Although rapid deceleration is important, accuracy, measured by the number of nodes, was prioritized to minimize the error between the desired and actual states. Table 8 highlights the optimal deceleration solutions for each $\dot{\mathbf{X}}_{\mathbf{bDes}}$.

In the 0.5 m/s simulation, loop 3 has the highest number of nodes; however, the relatively high value of J_{GP} suggests a failure to fully satisfy the complementarity conditions. Thus, among the solutions with the highest number of nodes (40), loop

1 performs best across Categories A, B, C, D, E, and F.

Conversely, loop 5 was selected as the *best deceleration solution* for $\dot{\mathbf{X}}_{\mathbf{bDes}} = 1.5$ m/s. Both loops 2 and 5 have the highest number of nodes, but loop 5 outperforms loop 2 in Categories B, D, and F. Although loop 2 performs best in Category A, it experiences significantly high $\mathbf{T_P}$.

Lastly, as the $\dot{\mathbf{X}}_{\mathbf{bDes}} = 4.0$ m/s solution focused on top speeds for the fastest gait termination, loop 3 was selected as the *best deceleration solution*, offering the quickest solution in Category A while utilizing the highest number of nodes (N).

Table 8: Representation of optimal deceleration solutions for 0.5 m/s, 1.5 m/s and 4.0 m/s. These solutions were obtained by setting the initial conditions as the final state conditions of the previously solved *best steady-state solutions*.

Loop	$\dot{\mathbf{X}}_{\mathbf{bDes}}$ (m/s)	N	TT (s)	$t[N]$ (s)	J_T	J_{GP}	$\mathbf{T_P}$ (Nm)	$\mathbf{\Theta_{MKA}}$ (°)	$\mathbf{GRF_Y}$ (N)
1	0.50	40	0.40	0.34	0.05	0.00	-5.00	159.96	116.93
2	0.50	30	0.30	0.23	0.05	0.00	-7.43	159.82	191.54
3	0.50	50	0.5	0.29	0.77	8.71×10^{-4}	-28.12	159.63	421.25
4	0.50	40	0.50	0.31	0.05	0.00	-5.24	160.00	264.36
5	0.50	40	0.70	0.41	0.08	0.00	-6.12	159.99	158.12
1	1.50	40	0.70	0.46	0.11	-6.20×10^{-6}	-5.62	159.96	124.02
2	1.50	50	0.30	0.19	0.55	-1.86×10^{-3}	-25.21	149.46	167.54
3	1.50	40	0.50	0.37	0.16	0.00	-12.39	160.00	186.56
4	1.50	40	0.60	0.53	0.16	-2.80×10^{-6}	-10.14	160.00	88.76
5	1.50	50	0.50	0.47	0.16	-3.56×10^{-6}	-14.86	160.00	80.00
1	4.00	40	0.30	0.33	0.44	-2.78×10^{-6}	-20.83	160.00	182.71
2	4.00	40	0.40	0.44	0.28	-2.82×10^{-6}	-11.75	160.00	176.23
3	4.00	50	0.30	0.33	0.54	-3.52×10^{-6}	-29.96	160.00	179.55
4	4.00	50	0.50	0.55	0.25	-3.55×10^{-6}	-13.82	160.00	122.57

From Table 8 it's observed that as the deceleration to terminate $\dot{\mathbf{X}}_{\mathbf{bDes}}$ increases, there's an increases in J_T, $\mathbf{T_P}$, and TT.

A higher $\dot{\mathbf{X}}_{\mathbf{b}}$ results in greater momentum, requiring more force to bring the body to a stop. The increased total time is due to friction bounds, demanding more actuation effort. The optimal solutions remain within the torque-speed limits, as shown in Figure 31. In addition, as $\dot{\mathbf{X}}_{\mathbf{bDes}}$ increases, the closer the torque and speed curve is to the boundary. Furthermore, the deceleration motion uses the least torque compared to both the acceleration and steady-state motions.

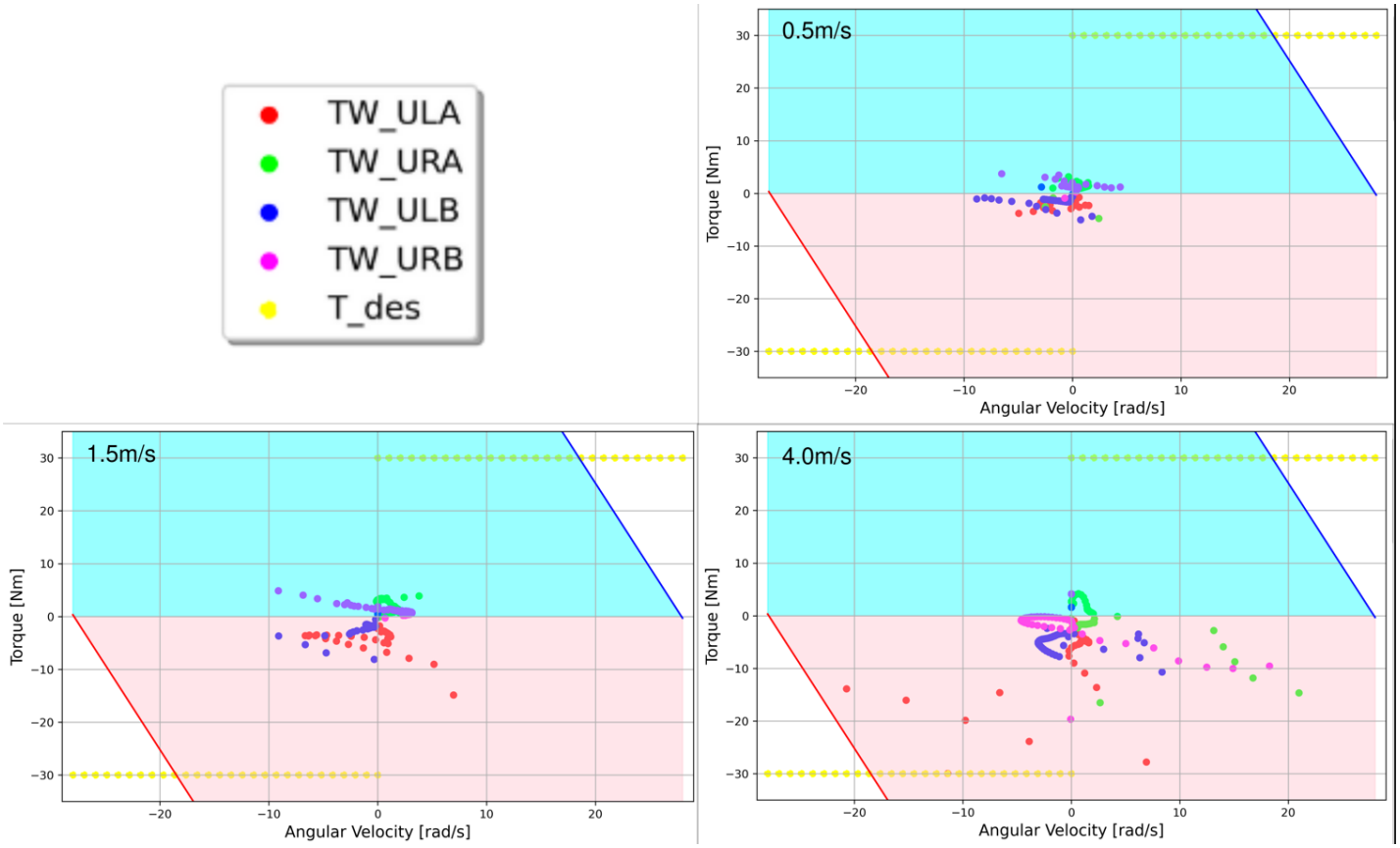

Figure 31: Displays the torque-speed curve of the optimal solutions for the 0.5 m/s (top right), 1.5 m/s (bottom left) and 4 m/s (bottom right) of all four motors for deceleration motions.

5.3.4 Summary

This study examines Baleka II's locomotion through TO across acceleration, steady-state walking/running, and deceleration phases. Gait cycles were *stitched* together, transitioning smoothly between phases by matching state variables, excluding body position and foot deformation. Walking and running were differentiated using FN = 0.05 for walking (0.5 m/s) and FN = 0.5 for transitioning to running (1.5 m/s), with a maximum tested velocity of 4.0 m/s — higher speeds proved infeasible. For steady-state gait cycles, symmetry allowed solving only half the gait, reducing computational cost, and switching leg states ensured full cycles. Best solutions were chosen based on multiple categories (A–H), including efficiency, torque usage, peak knee angles, GRF loads, and stability. At 0.5 m/s, Test 4 offered a stable walking gait; at 1.5 m/s and 4.0 m/s, Test 1 of each set showed the best performance, including successful flight phases for running. Radau integration provided improved

accuracy over BWE and incorporated seeding to explore diverse solutions. Acceleration phases were stitched onto steady-state gaits, using Radau and contact orders to ensure feasible launches into flight. Key setup parameters like node count, torque limits, and time steps were varied per test, with results showing increased strain and variability as speed increased, all remaining within motor torque-speed constraints.

6 Simscape Multibody and Simulink Software, Hardware, and Experiments

Using the TO solutions derived above, the resulting optimal *stitched* trajectories were implemented in Simscape to assess their feasibility. Once the solution proved feasible, it was deployed on the actual robot using the Simulink Real-Time platform. This section will provide an overview of the software, communication protocols, hardware, and experiments conducted on both the Simscape and Simulink Real-Time platforms. All resulting code, robot parameters, TO simulations, and videos are found in Appendix D.

6.1 Software and Communication

In this study, data management was divided between Windows and Ubuntu operating systems, requiring compatibility between the two. After computing optimal trajectories, the data points for each locomotive task were saved as CSV files, compatible with both LibreOffice Calc (Ubuntu) and Microsoft Excel (Windows). These files were then transferred to Windows for interpolation in Matlab R2022a (9.12).

Simulink (version 10.5 R2022a) and Simscape Multibody (version R2022a) were used to assess the feasibility of the interpolated trajectory points. Simulink is an integrated graphical modelling tool in Matlab, while Simscape, part of the Simulink environment, is designed for modelling physical 3D systems such as bodies, joints, links, and force elements, and solving EoM to run the simulation tests.

After successful simulations in Simscape, a Simulink Real-Time model (R2022a) was created. This model allowed the Simscape setup to interface with electronic control units and the robotic platform via Matlab and Simulink. The initialized code was then compiled and deployed to the Speedgoat Real-Time target computer for real-time execution on the physical robot.

To further enhance the mechanical modelling accuracy of the robot, SolidWorks 3D was employed. The platform provided insight on detailed specifications including: material properties, mass properties, dimensions, CoM, locations, and the range of motion boundaries. Thus, SolidWorks 3D models were developed for the robot, boom, and motors (highlighting individual elements such as links, bolts, nuts, clamps, and the gearbox). Specifically, Mr. Bright and Mr. Mailer concentrated on modelling the robot and boom in SolidWorks, while my focus on accurately modelling the motors in Solidworks. Table 9 details the robot's properties, adopted for all modelling platforms. Moreover, these models were stored as STL files to facilitate the exchange of 3D models and integration between Solidworks and Simscape. Figure 32 displays the pipeline of simulation used to acquire, assess and execute the

trajectories.

To facilitate the motor mode setup, encoder calibration, zeroing, Controller Area Network (CAN) identification and control calibration (position, velocity and torque), the CubeMars software tool was installed on the upper computer (found **here**). This interface was developed by the manufacturer to verify the motor's operations.

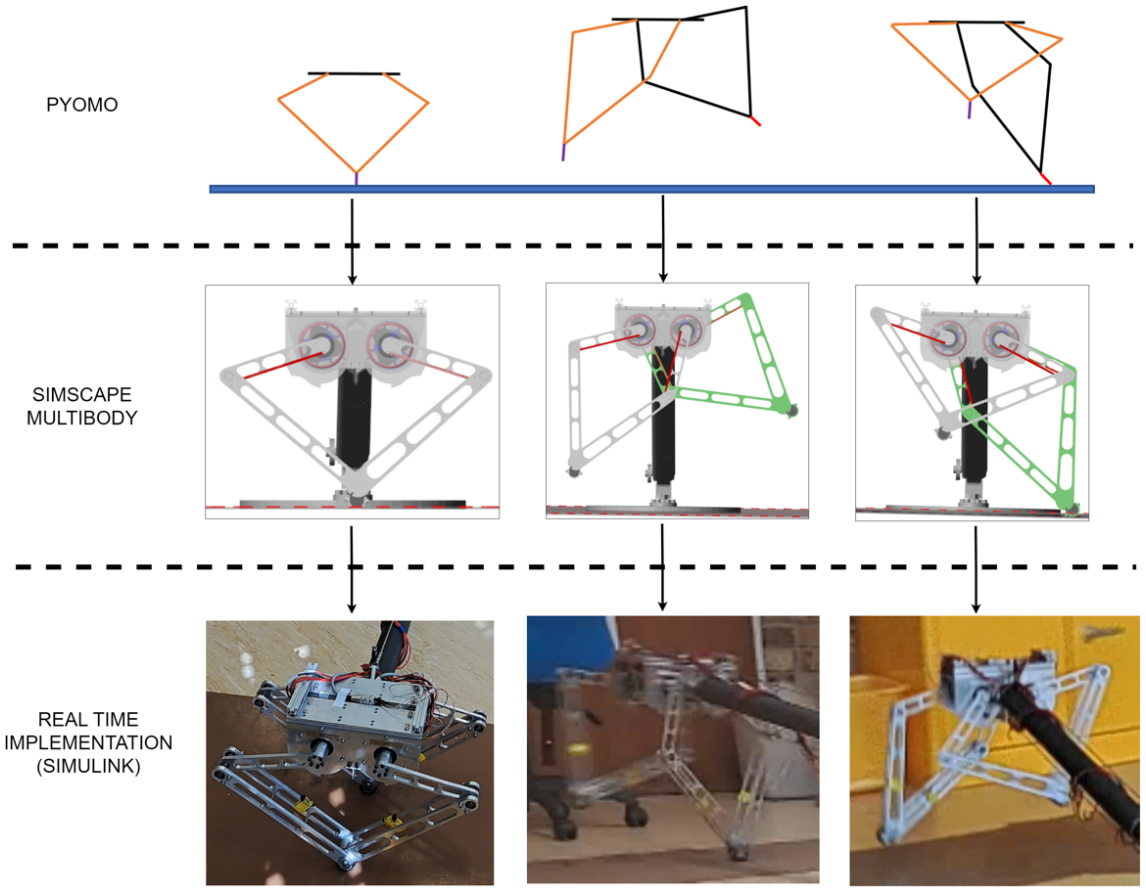

Figure 32: Pipeline highlighting using Pyomo to simulate TO (first row), followed by feasibility testing in Simscape (second row), and concluding with the implementation of the trajectories on the physical robot using Simulink Real-Time (third row).

Table 9: Baleka II values from Solidworks, including mass, total length, length to CoM (CoM Length), and MoI for each link.

Link	**Mass**(kg)	**Total Length**(m)	**CoM Length**(m)	**MoI**(kg·m^2)
Body (with motors)	5.76	0.25	0.0	0.03
Boom	2.47	2.50	N/A	1.95
ULA	0.15	0.17	0.04	0.0006
LLA	0.35	0.30	0.13	0.004
URA	0.15	0.17	0.04	0.0006
LRA	0.50	0.30	0.19	0.007
sfootA	0.01	0.0004	0.0002	0.0
ULB	0.15	0.17	0.04	0.0006
LLB	0.35	0.30	0.13	0.004
URB	0.15	0.17	0.04	0.0006
LRB	0.50	0.30	0.19	0.007
sfootB	0.01	0.0004	0.0002	0.0
Motors	N/A	N/A	N/A	0.0002

The execution of robot control in this research relied on three primary lines of communication. First, the EtherCAT protocol facilitated communication between the Simulink Real-Time application and the Speedgoat Real-Time Target Machine. EtherCAT efficiently converts modelled Simulink blocks into C-code, enabling real-time code compilation and execution at a designated sample rate.

Second, a combination of serial communication protocols, specifically UART and CAN, was established between the upper computer (utilizing the CubeMars interface) and each motor driver board.

Finally, the CAN protocol was employed for managing data exchange between the motors and the Speedgoat Real-Time Target Machine during trajectory execution. CAN is well-suited for environments requiring communication between an upper computer and multiple devices, efficiently coordinating status updates on a shared network. In this system, two CAN lines were configured, each supporting communication with two motors.

6.1.1 Summary

To conclude, this chapter detailed the complete pipeline from trajectory optimization to real-time execution on the physical robot. Starting with the generation of stitched optimal trajectories via Pyomo, feasibility testing was conducted in Simscape Multibody using Matlab R2022a. These simulations incorporated accurate physical models developed in SolidWorks, capturing critical properties such as mass

distribution, CoM, and MoI. Following successful validation, the trajectories were deployed through a Simulink Real-Time model onto the Speedgoat Real-Time Target Machine. Communication between system components was managed using EtherCAT for real-time execution, serial protocols (UART and CAN) for motor calibration via the CubeMars interface, and CAN for real-time control. This streamlined software-hardware pipeline ensured reliable and accurate replication of optimized trajectories on the physical platform, thereby bridging the gap between simulation and implementation.

6.2 Hardware

Three computers were used throughout this research. Two computers were used to execute TO using Pyomo with IPOPT solvers (Intel Core i3-6100U CPU @2.3 GHz and an Intel Core i5-3570 CPU@3.4 GHz). The IPOPT solver was not setup to run on GPUs and therefore, the best accessible CPUs were used. Furthermore, for all Matlab processes/compilation, an Intel Core i7-4790 CPU@ 3.6 GHz was used.

6.2.1 The Robot

Baleka II consists of many aluminium parts, some parts were laser cut and others were computer numerical controlled, along with multiple bolts and nuts holding it together. The robot's configuration consists of two legs, two feet, and a body. This body serves two functions, namely: housing the motors and to facilitate a support attachment when connecting to the boom. Figure 33 provides and overview of the robotic platform.

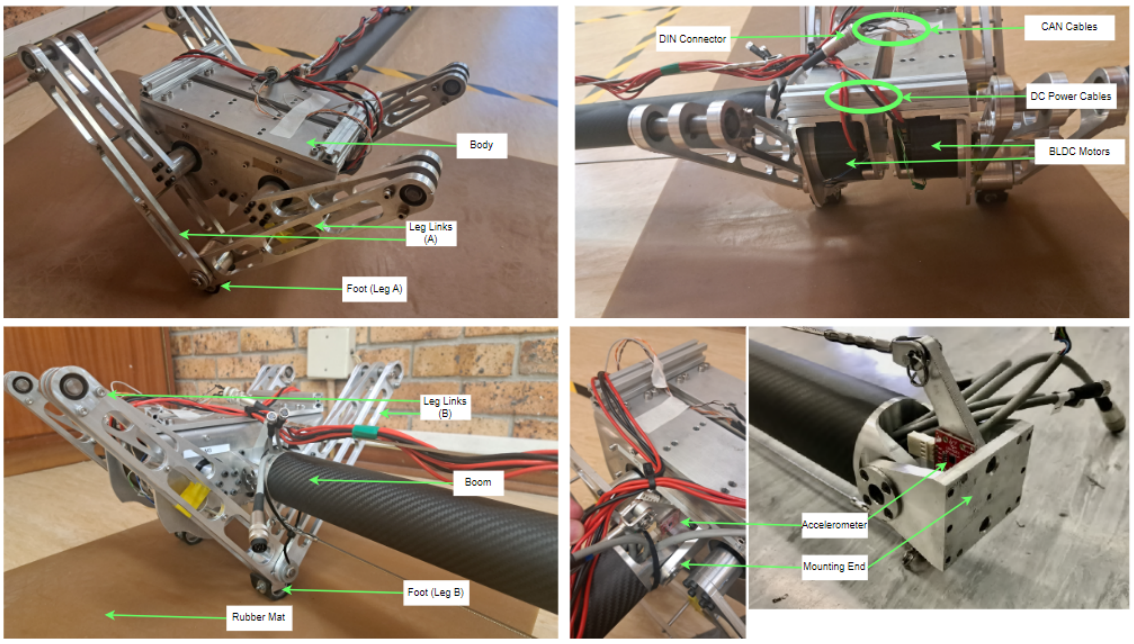

Figure 33: Baleka II robotic platform from multiple views, highlighting the legs, body, feet, motors, and boom.

The robot features four QDD motors housed between front, back, and top support plates for each motor pair. Each motor includes two ring supports for axial mounting of the front and back plates, which in turn support the top plate. Additionally, a bottom T-section support rod connects the two motor pairs, ensuring the body remains vertical relative to the boom.

Legs A and B consist of four linkages: upper left (UL), upper right (UR), lower left (LL), and lower right (LR). Each leg's UL and UR links attach axially to their respective motor output shafts using 25 mm bolts, enabling rotational movement. The UL and UR links connect at their lower ends to the LL and LR links, respectively, at a joint featuring a ball bearing mount to facilitate smooth rotation. The LR link's bottom connects to the foot, forming a fixed ankle joint with no rotation. Both legs utilize the same feet as the first iteration of Baleka, equipped with support discs and bolts. Lastly, the LL and LR links connect at the ankle via a hinge point, allowing for rotation, while inner leg pins provide additional support and rigidity.

The boom, initially outside the project scope, was included to enhance the robot's planar motion. Its mounting mechanism provided a fixed connection, allowing circular movement without rolling. An encoder tracked the boom's angular position, while three accelerometers and an Inertial Measurement Unit collected pitch, yaw, and roll data, enabling velocity estimation. This setup offered comprehensive data

on the robot's position, velocity, and acceleration. The boom was secured to the robot with four bolts, but because it was shared among multiple robots, the base plate was anchored to the ground with tape strips instead of permanent bolts.

6.2.2 Speedgoat Real-Time Target Machine and Communication Rate

The University of Cape Town (UCT) has a license agreement with MathWorks that enables students to use MATLAB, Simscape Multibody, and Simulink Real-Time. At the time of this research, several master's students were conducting similar studies on dynamic legged locomotion with these tools. Consequently, a communal plug-and-play solution was needed to integrate seamlessly with Simulink Real-Time for validating simulations through real-time testing. In this study, the Baseline Real-Time Target Machine was acquired as the real-time computer, serving as an entry-level solution for real-time simulation and testing of Simulink models. A summary of the machine's specifications is provided in Table 10.

Table 10: Speedgoat Real-Time target machine system specifications.

Specification	**Details**
CPU	Intel Celeron® J1900, Quad Core, 2 GHz
Memory	4 GiB DDR3
Graphics	Intel HD Graphics, 688 – 854 MHz
Storage	1x mSATA 2.0
Expansion slots	4x mPCIe
PCI clock	33/66 MHz

A key advantage of this system is its support for multiple interconnected I/O modules. Notably, the CAN module (IO691) features two independent CAN bus channels, each capable of speeds up to 1 Mbps and an 8-byte data payload. Shielded three-core cables were used for each CAN bus (CAN-HIGH, CAN-LOW, and ground) to enhance noise immunity, as recommended by the manufacturer. Additionally, two 120 Ohm termination resistors were placed between the CAN-HIGH and CAN-LOW lines for each bus. The motors were assigned CAN IDs 1, 2, 3, and 4, with CAN Bus A (leg A) utilizing CAN IDs 1 and 4, while CAN Bus B (leg B) used IDs 2 and 3. Figure 34 illustrates the equipment setup for the Speedgoat machine.

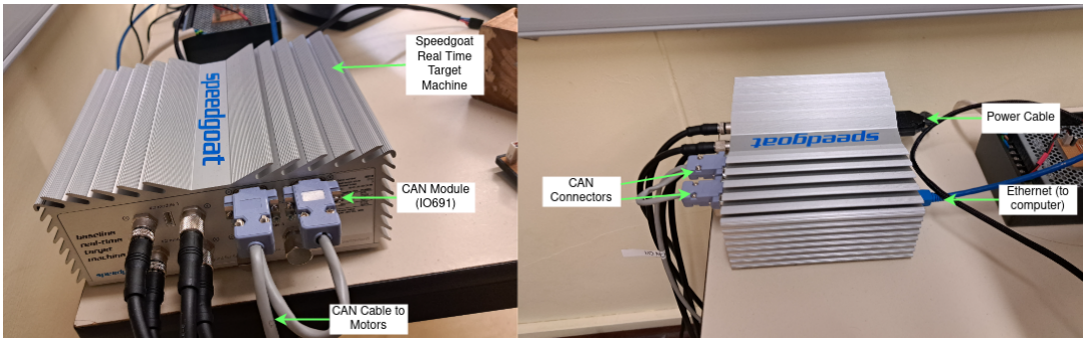

Figure 34: Speedgoat Real-Time Target Machine setup.

For successful communication between the target machine and the motors, the target machine sent a message, to which the motors responded. The target machine transmitted an 8-byte message containing data such as motor ID, position, velocity, torque, and gain commands. After executing the command, the motor replied with a 6-byte message that included the CAN ID and the motor's position, velocity, and torque measurements. In total, 14 bytes (or 112 bits) were required for each exchange. Given that there were two motors on each bus and the CAN module communicated at a rate of 1 Mbps, the required send and receive rate for each message was calculated to be approximately 2,232.14 Hz (as per Equation 37). To ensure adequate overhead for arbitration, control, error checking, and reliable communication, while allowing the motors time to process the information, a communication rate of 1 kHz was selected.

$$\begin{aligned} \text{Communication Rate} &= \frac{\text{Maximum Speed}}{\text{motors} \times \text{bytes} \times \text{bit conversion}} \\ \text{Communication Rate} &= \frac{1\text{Mbps}}{4 \times 14 \times 8} = 2,232.14 \text{ Hz} \end{aligned} \tag{37}$$

This communication rate was also used as the sample rate for the Simscape simulations. To evaluate the feasibility of modelling each foot contact as a spring-damper system in Simscape, the sample rate needed to be high enough to capture high-frequency variations in the system. To validate this, a simple mass-spring-damper system was modelled using the derived stiffness and damping coefficient values. The spring was displaced to assess whether the sample rate could adequately detect the resulting oscillations.

Figure 35 shows the step response at 100 Hz, demonstrating that a sample time of 1 kHz was sufficient to capture position and velocity oscillations without losing critical information between samples. The rise time of the system's velocity response was 0.015 seconds (67 Hz), indicating that the sample frequency should generally be set

to ten times the rise time, resulting in a minimum sample rate of 670 Hz. Thus, a sample time of 1 kHz proved adequate for this system.

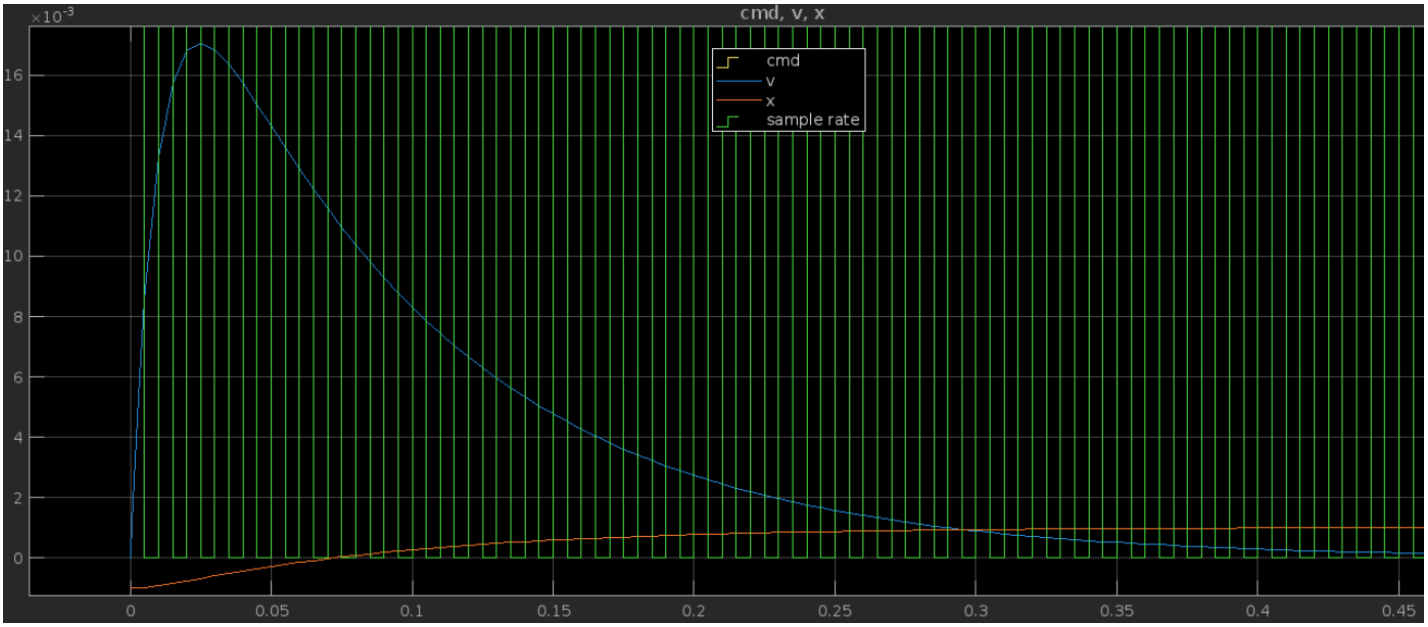

Figure 35: Spring-Damper simulation with the use of spring and damper coefficients of $K_{Foot} = 10^4$ N/m and $D_{Foot} = 10^3$ Ns/m, respectively. Here the blue curve represents velocity response, orange the position response and the green is the sample rate (100 Hz).

To execute the real-time control trajectories from the Simulink Real-Time Model, the code was compiled onto the target machine, which implemented the commands to the actuators by converting the controller commands into CAN data frame format. Once the messages were received by each motor, the target machine seamlessly converted the CAN data back into usable information, including measured torque, position, and velocity.

6.2.3 Power Supply and Cable Selection

The QPX1200SP DC power supply unit (PSU), rated at 1.2 kW with a maximum output of 60 V, was used to power the robot. Each motor operates at 24 V with a rated current of 18.5 A, resulting in a theoretical total current requirement of 74 A. However, analysis of the torque-speed curves (Figures 29, 30, and 31) indicates that the motors operate below their rated torque for 85% of the simulation, exceeding rated torque only 15% of the time. Applying the diversity factor equation (38), the adjusted current requirement was calculated to be approximately 46 A, reflecting a diversity factor of 38%.

$$\begin{aligned} \text{Avg Current} &= \text{Maximum Current} \times \text{Maximum Load \%} \\ &+ \text{Maximum Current} \times \text{Avg Load \%} \times (1 + \text{DF}) \\ 50A &= 74A \times 0.15 + 74A \times 0.85 \times (1 + \text{DF}) \\ \text{DF} &= \text{Diversity Factor} = 38.20\% \end{aligned} \tag{38}$$

Due to lab resource constraints and budget limitations, purchasing higher-rated power supplies was infeasible. An alternative PSU with double the current capacity was tested but caused significant disturbances in the CAN lines, as illustrated in Figure 36. In this figure, PSU 2 represents the higher-rated unit, while PSU 1 is the QPX1200SP. A continuous test signal sent to a motor revealed that the CAN Bus signals from PSU 2 were noisier than those from PSU 1. According to the IO691 manual, the CAN HIGH signal should range from 2.75 V to 4.5 V, and CAN LOW from 0.5 V to 2.25 V. Figure 36 shows that PSU 2 caused peak voltages exceeding 6 V for both signals. Consequently, PSU 1 was chosen for all experiments.

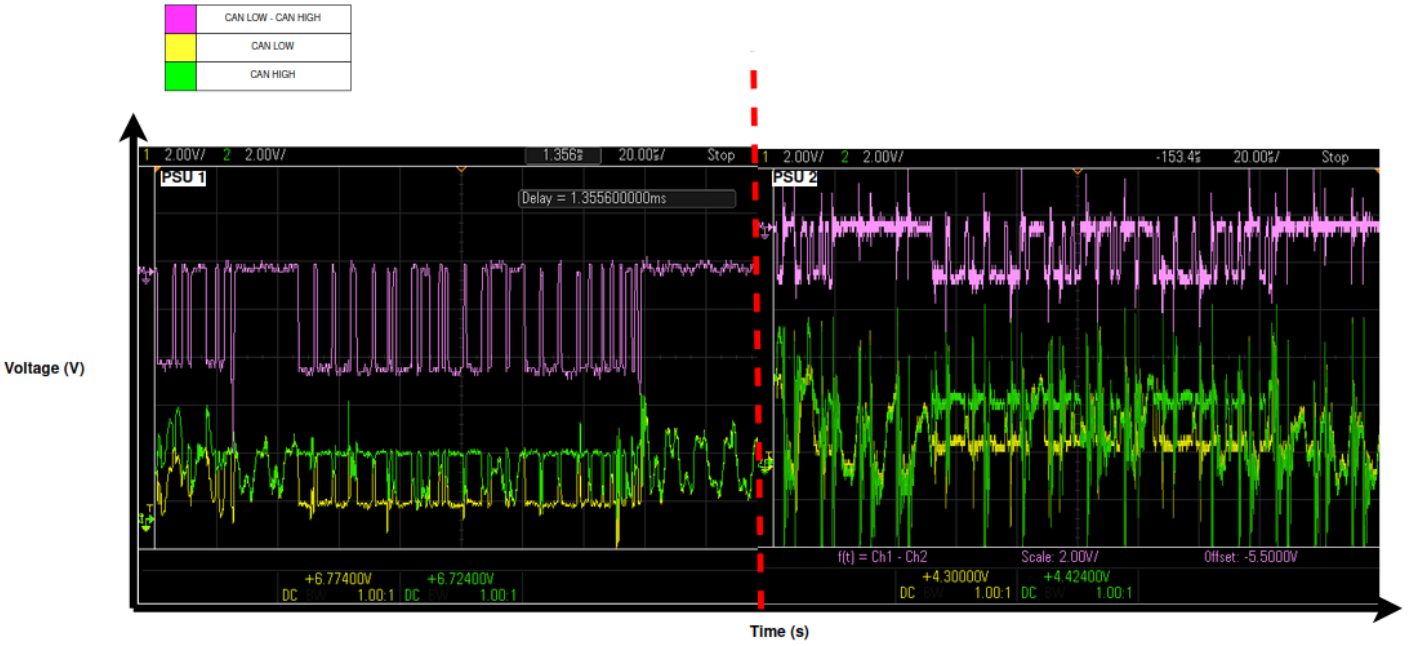

Figure 36: CAN Buses of two PSUs sending a test command. It can be seen that PSU 2 presents a nosier signal compared to PSU 1.

To connect the PSU to the motors, 10 AWG and 16 AWG copper cables with silicon insulation were used. The 16 AWG cables were specified by the manufacturer, while 10 AWG cables facilitated multiple connections between the PSU's single 10 AWG cable and the motors' 16 AWG cables.

The total voltage drop, calculated using Equation 39, considers the current and cable length for both cables: 74 A and 18.5 A for 10 AWG and 16 AWG cables, respectively, with lengths of 10 m and 2 m. The conductor DC resistance for the 16 AWG and 10 AWG cables was 4.4 Ω/1000 ft @ 20 °C and 1.08 Ω/1000 ft @ 20 °C, respectively [83], [84]. The voltage drop was 0.53 V for the 16 AWG cable and 2.62 V for the 10 AWG cable, yielding a total of 3.15 V. Due to the motor's operating manual not specifying an acceptable operating voltage range, the PSU was set to 24 V to prevent overheating and potential damage from excessive power. Additionally, the motor does not consistently operate at its rated current, as the commanded current varies over a wide range.

$$\begin{gathered}\text{Voltage Drop} = \text{Current} \times \text{Cable Length} \times \text{Cable Resistance per unit Length} \\ \text{Voltage Drop 16 AWG} = 18.5 \times 2 \times (4.4 \times \frac{1}{304.8}) = 0.53\text{V} \\ \text{Voltage Drop 10 AWG} = 74 \times 10 \times (1.08 \times \frac{1}{304.8}) = 2.62\text{V}\end{gathered} \tag{39}$$

6.2.4 Summary

This chapter outlined the key hardware used in developing and testing the Baleka II robot. TO ran on Intel Core i3 and i5 CPUs using Pyomo and IPOPT, while MATLAB processes used an i7 system. The robot features two legs with four-bar linkages, four QDD motors, and a body mounted to a boom for planar motion. Real-time control was achieved using a Speedgoat Baseline Real-Time Target Machine and Simulink Real-Time, with reliable 1 kHz CAN communication via IO691 modules. A 1.2 kW QPX1200SP power supply was selected over a higher-rated unit due to reduced electrical noise, and current demand was optimized to 46 A using diversity factor analysis. Cable sizing ensured minimal voltage drop, supporting safe and stable operation within lab constraints.

6.3 Interpolation, Simscape Modelling, and Validation Tests

6.3.1 Interpolation

Upon completing the trajectory tasks, the data was utilized in the Simscape model. To accurately implement the trajectories in Simscape, the motor commands for position, velocity, and torque required interpolation for smooth integration into the model (rate of control). This step ensured that the specified nodes (with their respective scaling factors) were converted to the correct time interval as per Equation 13. Simscape ran on a 1 kHz time step, while the trajectories could only provide a

0.5 seconds/100 nodes = 200 Hz (best case). In order to construct an interpolation function, a query point array was required. This query point array specifies the time intervals over which the functions (position, velocity and control input) should be interpolated. Specifically, the array is setup such that:

1. the first index begins at 0 seconds,
2. the final index equals the total time to complete a trajectory task.
3. the total length of the array equals the desired number of time points (communication rate of the simulation, i.e. 1000 Hz). Thus, the time intervals are separated equally by 0.001 seconds from 0 seconds to the total time it takes to complete trajectory task.

In this study, MATLAB's *spline* and *interp1* functions were utilized for interpolation. Position trajectories were interpolated using cubic splines to ensure continuous motion, avoiding abrupt changes in velocity and acceleration. Velocity states and control inputs were linearly interpolated with *interp1*, with a zero-order hold method applied to control inputs, maintaining each value constant until a change was detected. This approach was chosen for its computational efficiency while ensuring smooth transitions between trajectory points [63]. The resulting interpolated arrays were then converted into structures as inputs to the Simscape model.

Once all trajectories were interpolated, their final value sets were used in the Simscape physical system. Simscape was used as a verification platform to assess the feasibility of the interpolated trajectory values (position, velocity and control input). This software was used in the Simulink environment where all physical components were represented as block models. The model developed consisted of both a *controller* aspect and a *physical model* aspect. The controller aspect was developed to contain all controller related aspects (commands), while the physical model consisted of all the physical aspects of the robot (leg links, body, boom, motors and feet). Figure 37 displays an overview of the Simscape model which consist of both the physical robot models and the controller blocks with connection pathways.

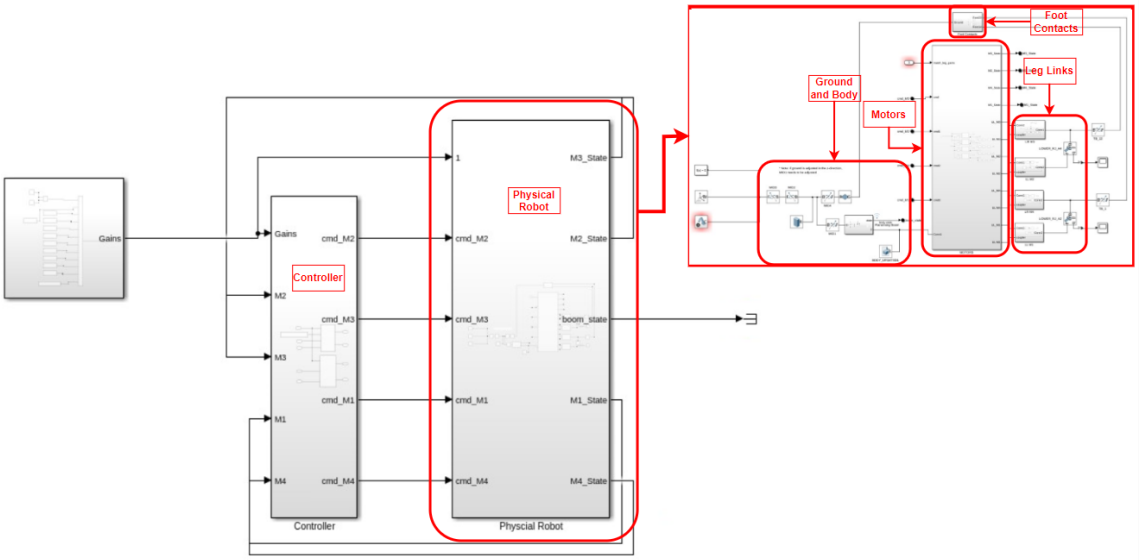

Figure 37: Simscape layout representing both controller and physical robot model blocks, with its exploded view.

A key limitation of this study was the absence of body state feedback, with only motor FOC feedback based on desired position, velocity, and torque commands. The system's physical model was simulated using a numerical solver to solve its ODEs and DAEs over time. Although the model operated in real-time with finite time steps, an implicit variable-step solver was chosen due to the system's physical nature and its ability to handle complex events, such as ground contact and friction, which introduce stiff solving challenges [85], [86], [87]. The DAESSC solver was selected for its robustness in physical modelling, and solver parameters (min/maximum step sizes and tolerances) were tuned to balance speed and accuracy as shown below:

1. Relative tolerance = 10^{-4}
2. Absolute tolerance = 10^{-7}
3. Maximum step size = 10^{-3}
4. Minimum step size = auto
5. Activated an adaptive zero crossing algorithm to improve the accuracy allowing the solver to adjust the zero crossing (discontinuity) threshold.

Figure 38 illustrates the solver profile of a 4.0 m/s simulation in Simscape over a short duration. The solver's performance highlighted that smaller time-steps were required at critical points, such as foot-ground contact and torque variations, particularly to adhere to the torque-speed curve. This suggests that using a fixed time-step solver

would be challenging, as both torque-speed requirements and contact events demand fine step sizes to ensure stable and feasible results.

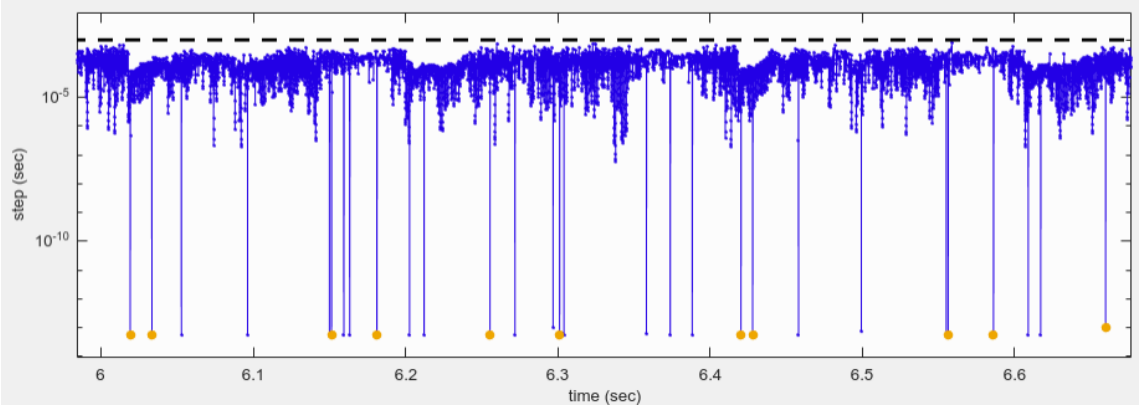

Figure 38: Illustration of the Simscape variable time-step solver profile for a 4.0 m/s run over a short duration - highlighting the time-step sizes. All orange dots displays the zero crossings of the foot contact.

The model parameters included link inertia and length values as listed in Table 9. Additionally, the motor's inertia, torque-speed model (Figure 20), and a PD controller were implemented. The PD controller tracked desired position, velocity, and torque commands, with proportional gain $Kp = 250$ and differential gain $Kd = 4$. These values were set below the manufacturer's recommended maximum values to balance the torque input and the torque generated from position and velocity tracking. The Kp was halved to prevent excessive input, while the Kd was set near the maximum values to improve stability and reduce overshoot for rapidly varying command inputs.

6.3.2 Simscape Multibody Modelling

Spatial Contact Force blocks in Simscape were used to model foot-ground contact in the robot. This block represents contact through a frame normal to the GRFs and models both normal and frictional forces [88]. The normal force, preventing body penetration, is modelled by a spring-damper function with spring stiffness $K_{Foot} = 10^4$ and damping coefficient $D_{Foot} = 10^3$. Friction, opposing tangential velocities, is modelled using a smooth stick-slip curve, with both static and dynamic friction coefficients $\mu_s = \mu_d = 0.8$ set based on the rubber-rubber contact surfaces.

After completing the Simscape simulations, the final step was to integrate the controller into Simulink Real-Time for testing on the robot. This enabled a straightforward transfer of controller blocks from Simscape to Simulink Real-Time. Additional

blocks were added to develop CAN signals for motor communication. A simulation speed of 2 kHz was used, while the communication speed for message exchange was set to 1 kHz.

6.3.3 Validation Tests

The Simscape model was validated using three tests: high drop, low drop, and leg rotation. Each test employed the same setup as the Pyomo model verification tests, including identical run times and variable time-step solvers. In the high and low drop tests, the boom's pitch angle was initialized at 21.2° and 9.3°, respectively, with knee angles set to 120°. The leg rotation test followed Pyomo command trajectories to evaluate tracking performance.

Figures 21, 22, 24 present the validation test results for both Pyomo and Simscape models. In the high drop test (Figure 21), the Simscape dynamics closely align with Pyomo dynamics, indicating feasibility with low tracking errors. However, differences in downward velocity and acceleration suggest inaccuracies in mechanical mass properties. While Pyomo relies on assumptions like rigid links in the manipulator equation, Simscape integrates SolidWorks mass properties, offering a more realistic model.

In the low drop test (Figure 22), the robot makes contact at 0.23 seconds. While the GRF, downward body acceleration, and foot deflection differ, they change as expected. Body position, motor torque, and foot positions are consistent across models. The differences in GRF and foot deflection are likely due to variations in mass properties and contact modelling. Simscape uses the actual geometric foot shapes and spring-damper normal forces with penetration depth smoothing, whereas Pyomo assumes point contacts and simple spring-damper deflection.

In the leg rotation test (Figure 24), foot positions align, and torque-speed curves remain within limits for both models. However, motor position and velocity curves (motors 1 and 4) show discrepancies, likely due to combined inaccuracies in the motor PD control and mass properties.

6.3.4 Summary

This chapter outlines the critical process of interpolating trajectory data, implementing it in Simscape, and conducting validation tests. Motor commands for position, velocity, and torque were interpolated from 200 Hz trajectory data to 1 kHz for Simscape using MATLAB's `spline` and `interp1` functions. Cubic splines ensured smooth position trajectories, while velocity and control inputs used linear and zero-order hold interpolation, respectively. These inputs were structured and fed into a

detailed Simscape model that represented both the robot's physical system and controller components. A DAESSC implicit variable-step solver was chosen for its ability to handle stiff dynamics, with parameters tuned for accuracy and stability (relative tolerance: 10^{-4}, absolute tolerance: 10^{-7}, maximum step size: 10^{-3}). The physical model included realistic contact modelling using spring-damper elements (stiffness $K_{Foot} = 10^4$, damping $D_{Foot} = 10^3$) and friction coefficients ($\mu_s = \mu_d = 0.8$). A PD controller with gains $Kp = 250$ and $Kd = 4$ was applied. Final controller designs were ported to Simulink Real-Time at a 2 kHz simulation speed and 1 kHz communication rate. Validation was performed using high drop, low drop, and leg rotation tests. Simscape and Pyomo models showed strong agreement in dynamics and motor torque tracking, though some discrepancies were noted due to differences in mass property modelling, contact assumptions, and controller fidelity. Overall, the Simscape model proved to be a feasible and accurate representation for real-time control deployment.

6.4 Rest to Standby State Transition

This research examined a bipedal robot's capability to accelerate from rest into a steady gait. The process began with the robot entering a standby state to define its initial configuration. In this state, parameters were set as follows: $(\mathbf{X_b}; \mathbf{Y_b}) = (0.0; 0.35)$ and $\dot{\mathbf{q}}, \mathbf{PF_Y}, \mathbf{PF_X}$, and $\mathbf{T_u}$ were all set to zero.

The standby state positioned the knee angles at 77°, within a 20% tolerance of prior studies on sprinters using start blocks (94.5 ± 11.2°) and bipedal robots' running gaits (50° initial knee flexion) [89], [90]. These knee angles are shown in Figure 19. Transitioning to standby required the robot to move from its initial resting state, defined by un-actuated conditions with the knee angles set to 0°, as depicted in Figure 39. This was achieved using impedance control.

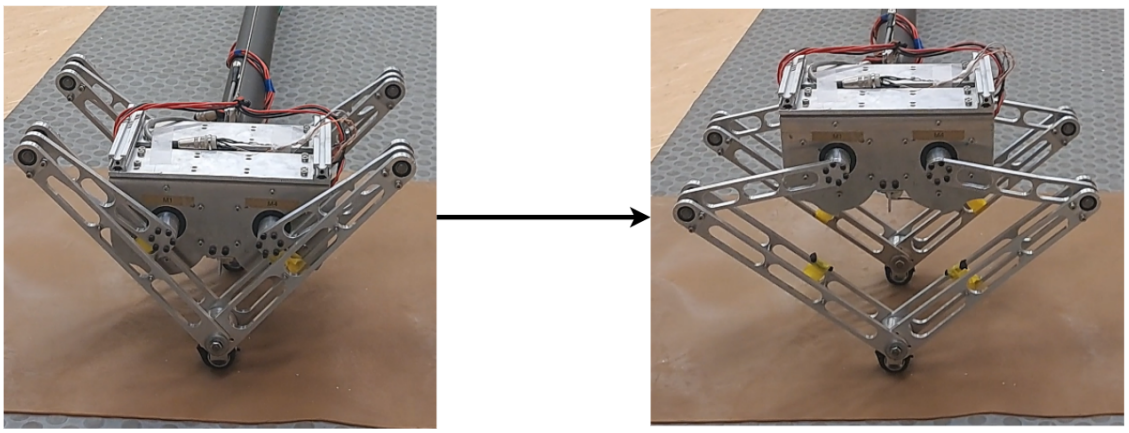

Figure 39: Transitioning from initial resting state to the standby state using impedance control.

6.4.1 Impedance Control Setup

To enable the transition, an impedance controller was implemented, utilizing the relationship between motor states (position and velocity) and the end effectors (feet) through inverse kinematics. The inverse kinematics function converted motor position and velocity into polar coordinates of a virtual leg link, as illustrated in Figure 40. Detailed equations for this function are provided in Appendix C.

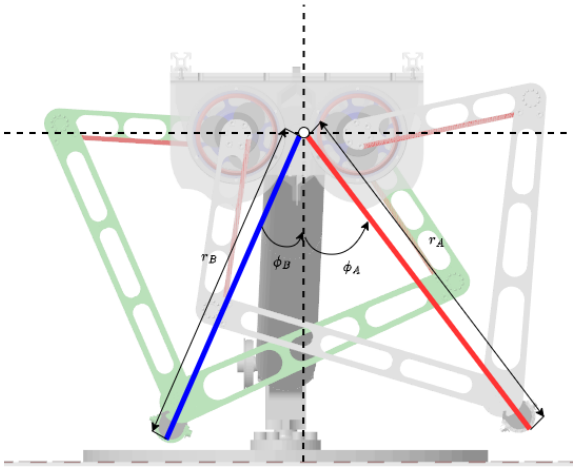

Figure 40: The diagram used to model the inverse kinematics of Baleka II.

The impedance controller separates radial force ($\mathbf{F_r}$) and angular force ($\mathbf{F_\Phi}$) for each leg, as defined by equations 40 and 41, using Equation 4 as a reference. Here, Kp_r and Kp_Φ represent proportional gains in the radial and angular directions, respectively, while Kd_r and Kd_Φ denote the corresponding differential gains. Variables **r**

and $\mathbf{\Phi}$ refer to the radial distance (from body centre to foot) and the angle (between the virtual leg's radial plane [r] and vertical plane [Y]), respectively. States labelled **Mes** and **Des** indicate measured and desired values, respectively.

$$\mathbf{F_r} = -Kp_r(\mathbf{r_{Mes}} - \mathbf{r_{Des}}) - Kd_r(\dot{\mathbf{r}}_{\mathbf{Mes}} - \dot{\mathbf{r}}_{\mathbf{Des}}) \tag{40}$$

$$\mathbf{F_\Phi} = -Kp_\Phi(\mathbf{\Phi_{Mes}} - \mathbf{\Phi_{Des}}) - Kd_\Phi(\dot{\mathbf{\Phi}}_{\mathbf{Mes}} - \dot{\mathbf{\Phi}}_{\mathbf{Des}}) \tag{41}$$

To map the control forces to the corresponding motor torques - $\mathbf{F_r}$ and $\mathbf{F_\Phi}$ were multiplied by a mapping Jacobian (as per Equation 42).

$$\mathbf{B} = \mathbf{J^T} \begin{bmatrix} \mathbf{F_r} \\ \mathbf{F_\Phi} \end{bmatrix} \tag{42}$$

6.4.2 Impedance Control Limitations

To determine the maximum gain values for the impedance controller, PD gains were incrementally adjusted until motor vibrations occurred. The boundary values were:

1. $Kp_r = 3000$
2. $Kd_r = 20$
3. $Kp_\Phi = 3000$
4. $Kd_\Phi = 20$

A trade-off was observed between proportional and derivative gains; higher proportional gains required lower derivative gains. Final boundary limits were determined as $Kp_r = 2500, Kd_r = 15, Kp_\Phi = 800$ and $Kd_\Phi = 7$. These limits were identified based on auditory feedback (motor vibrations), beyond which the risk of motor damage increased.

The focus was on steady-state error (body height) and stability rather than response characteristics. Therefore, the gains were balanced to ensure stability during standby state transitions, with minimal steady-state error, and avoidance of actuator limits. The gain and parameter values are summarized in Table 11.

Table 11: Table of parameters for the impedance controller gains.

Parameter	Value
$\mathbf{r_{Des}}$	0.35 m
$\mathbf{\Phi_{Des}}$	0.0 radians
$\dot{\mathbf{r}}_{\mathbf{Des}}$	0.0 m/s
$\dot{\mathbf{\Phi}}_{\mathbf{Des}}$	0.0 rad/s
Kp_r	600
Kd_r	60
Kp_Φ	60
Kd_Φ	6

6.4.3 Summary

This chapter detailed the transition of the Baleka II bipedal robot from a passive rest state into an active standby configuration using impedance control. The standby posture was defined with a body height of 0.35 m and zero initial velocities and torques, while the knees were flexed at 77°, aligning with biomechanical literature. The impedance controller, implemented via inverse kinematics, mapped desired radial and angular leg forces to motor torques using a Jacobian matrix. Through iterative tuning, the final gain values were selected to minimize steady-state error and ensure stable transitions without inducing harmful motor vibrations. These gains balanced responsiveness with actuator safety, setting the foundation for initiating locomotion from a well-defined initial state.

6.5 Locomotion Experiments

During trajectory simulation and testing, the robot was initialized in its rest state at t = 0 seconds. At t = 3 seconds, the robot transitioned to the standby state using the impedance controller. This delay allowed adequate time for setting up cameras and positioning the experimenter to follow and catch the robot if necessary. At t = 5 seconds, the *stitched* trajectory controller was activated, initiating acceleration, walking/running, and deceleration motions.

Simulation durations were set to complete a full rotation, with run-times of 50 seconds, 20 seconds, and 15 seconds for speeds of 0.5 m/s, 1.5 m/s, and 4.0 m/s, respectively. The following sections compare interpolated Pyomo results, Simscape simulations, and Simulink Real-Time outcomes (actual robot).

Although the robot employs four motors, only results for motors 1 and 4 are presented (leg A), as the robot's symmetrical mechanical structure, actuators, and commands make displaying all four redundant.

6.5.1 Impedance Control

Figure 41 illustrates the implementation of the impedance controller on the robotic platform using Simulink Real-Time. The low gain settings resulted in a slow response and a steady-state error, evident in the body height curve. Despite these limitations, the robot successfully transitioned to the standby state. Notably, the robot remained stable between t = 3 seconds to t = 5 seconds, with no changes observed in position or velocity. However, the low gains resulted in a 0.03 m body height error.

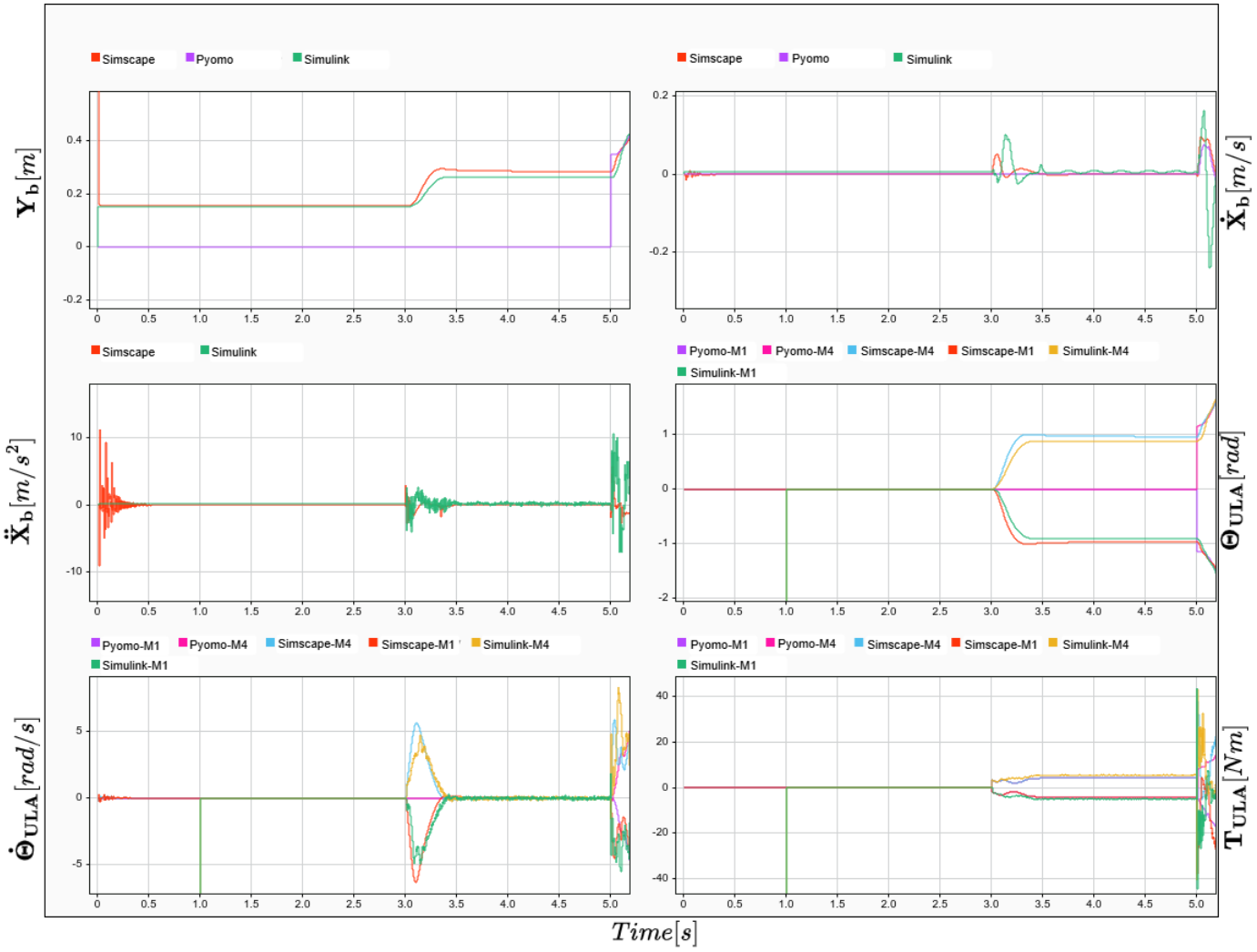

Figure 41: Impedance controller implementation, representing the interpolated Pyomo trajectories (purple and pink curves), Simscape curves (red and cyan) and the Simulink Real-Time implementation curves (green and yellow).

6.5.2 Walking Experiments (0.5 m/s)

The walking simulation (0.5 m/s) was implemented on the robot using *stitched* trajectories, starting at t = 5 seconds. Between t = 5 and t = 5.6 seconds, the robot accelerated, followed by a steady-state walk from t = 5.6 to t = 38.1 seconds, and decelerated thereafter. Figures 42, 43, and 44 illustrate snapshots of the acceleration, steady-state, and deceleration phases, respectively.

Figure 42: Snapshot of the 0.5 m/s walk, displaying Pyomo, Simscape, and Simulink curves. Between time-stamp t = 5 seconds and t = 5.6 seconds highlights the acceleration trajectories.

In the acceleration phase (Figure 42), both Simscape and Simulink curves for $\mathbf{Y_b}$

and motor positions followed similar patterns across simulations in real-time. The Simscape curve undershot the Pyomo reference, while the Simulink curve initially overshot before undershooting by less than 0.02 m. Both Simscape and Simulink demonstrated slower rise times, likely due to differences in modelled mass properties or Pyomo's simplified motor model. In addition, the Simscape and Simulink curves exhibit an initial $\mathbf{Y_b}$ error relative to the Pyomo reference, attributed to insufficient impedance controller gains to achieve the desired body height.

For $\dot{\mathbf{X}}_\mathbf{b}$, Pyomo and Simscape results were consistent, while Simulink displayed oscillations up to 0.8 m/s and a 0.7-second delay in reaching the desired velocity. This discrepancy may result from sensitive equipment, unmodelled foot friction, and inertia. Additionally, Simulink and Simscape struggled with the discontinuity at t = 5.5 seconds, revealing limitations in Pyomo's approximations.

Motor velocity curves in Simscape and Simulink oscillated before stabilizing at Pyomo trajectory points, with Simulink experiencing larger overshoots. Initial $\mathbf{Y_b}$ errors (0.26 m and 0.28 m for Simulink and Simscape, respectively, compared to 0.35 m in Pyomo) likely contributed to these discrepancies - specifically, influencing the motors' velocities to reduce the error.

Regarding $\ddot{\mathbf{X}}_\mathbf{b}$, Simulink oscillated and accelerated slower than Simscape, decelerating upon entering the periodic gait, contrary to Simscape's continued acceleration. This mismatch is attributed to differences in contact model assumptions, as the robot naturally decelerates during landing.

Lastly, torque curves showed significant differences between Pyomo, Simscape, and Simulink. Pyomo trajectories excluded motor PD controller effects, with torque variations arising from position and velocity error corrections. The high differential gain in the motors amplified torque requirements even for small errors.

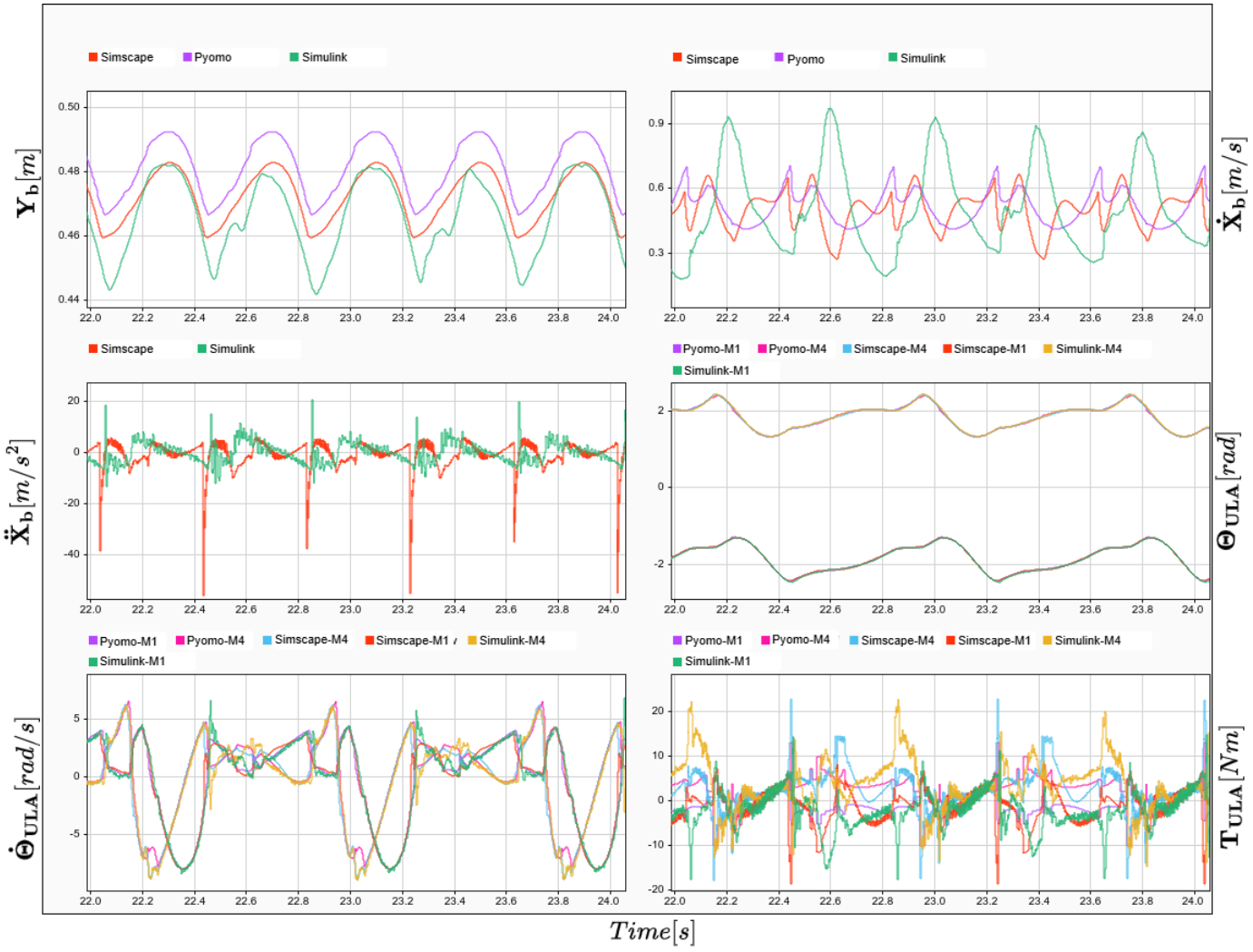

Figure 43: Snapshot of the 0.5 m/s walk, displaying Pyomo, Simscape, and Simulink curves. This snapshot displays the steady-state gait.

In the steady-state walking phase (Figure 43), the Simscape and Simulink curves for $\mathbf{Y_b}$ and motor positions exhibit similar patterns to Pyomo in real-time, with a relatively constant error. Both Simscape and Simulink curves align at their peak values when the airborne foot lifts. However, during ground contact, the Simscape curve shows an error of approximately 0.025 m, likely due to the unmodelled backdrivability of the motors. Specifically, the landing torque was insufficient to counteract the back-EMF, resulting in an inability to maintain the desired body height. Additionally, the Simulink curve displays disturbances at t = 22.55 and t = 23.35 seconds, coinciding with the switching of the half-gait trajectory to the opposite leg. This

highlights a limitation in not simulating a complete gait cycle.

For $\dot{\mathbf{X}}_\mathbf{b}$, all three models average 0.5 m/s. However, the Simulink curve shows significant errors (estimate of 0.5 m/s) and a 0.2 s delay compared to Pyomo and Simscape, with peak errors occurring when the airborne foot overshoots its target velocity. This highlights the system's limited ability to adjust velocity rapidly.

Regarding $\ddot{\mathbf{X}}_\mathbf{b}$, the Simscape curve shows significant acceleration changes during foot-ground contact. The modelled foot contact injects an upward force, causing knee flexion and unplanned motor velocity changes. Similar disturbances occur in the Simulink results but are delayed and smaller in magnitude. This indicates that the Simscape foot-contact model produces greater deformation between the foot and the ground.

Finally, motor velocity values are consistent across all models except during foot-ground collisions, where variations reflect inaccuracies in the contact models of both Pyomo and Simscape.

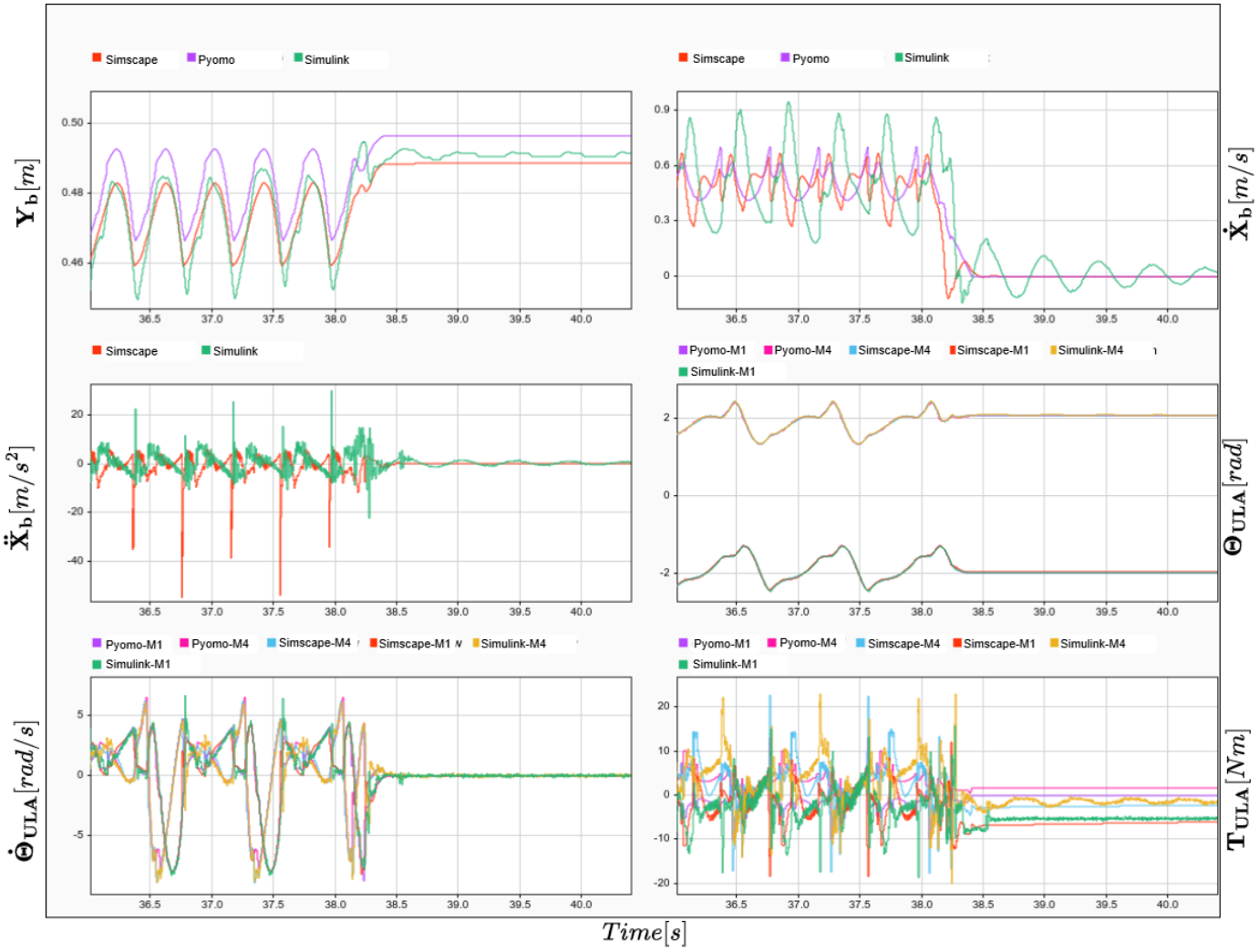

Figure 44: Snapshot of the 0.5 m/s walk, displaying Pyomo, Simscape, and Simulink curves when the robot decelerates.

During the deceleration phase (Figure 44), both the Simulink and Simscape $\mathbf{Y_b}$ curves observe overshoots of 0.007 m and 0.005 m, and steady-state errors of 0.007 m and 0.01 m, respectively. While the Simulink and Simscape $\dot{\mathbf{X}}_\mathbf{b}$ curves experience 0.25 m/s and 0.3 m/s undershoot, receptively.

The Simulink curve exhibits two oscillation phases in $\mathbf{Y_b}$ and $\dot{\mathbf{X}}_\mathbf{b}$. The first phase, from t = 38.3 to t = 38.4 seconds, arises from foot-rubber mat interaction during rapid braking. High friction and forward velocity cause the robot to vibrate, intermittently losing ground contact while maintaining foot orientation. Despite this, the

robot stabilizes post-deceleration, as indicated by consistent $\mathbf{Y_b}$, $\dot{\mathbf{X}}_\mathbf{b}$, $\ddot{\mathbf{X}}_\mathbf{b}$, and motor profiles. The robot's feet comes to rest after 0.8 seconds after the the deceleration phase activation.

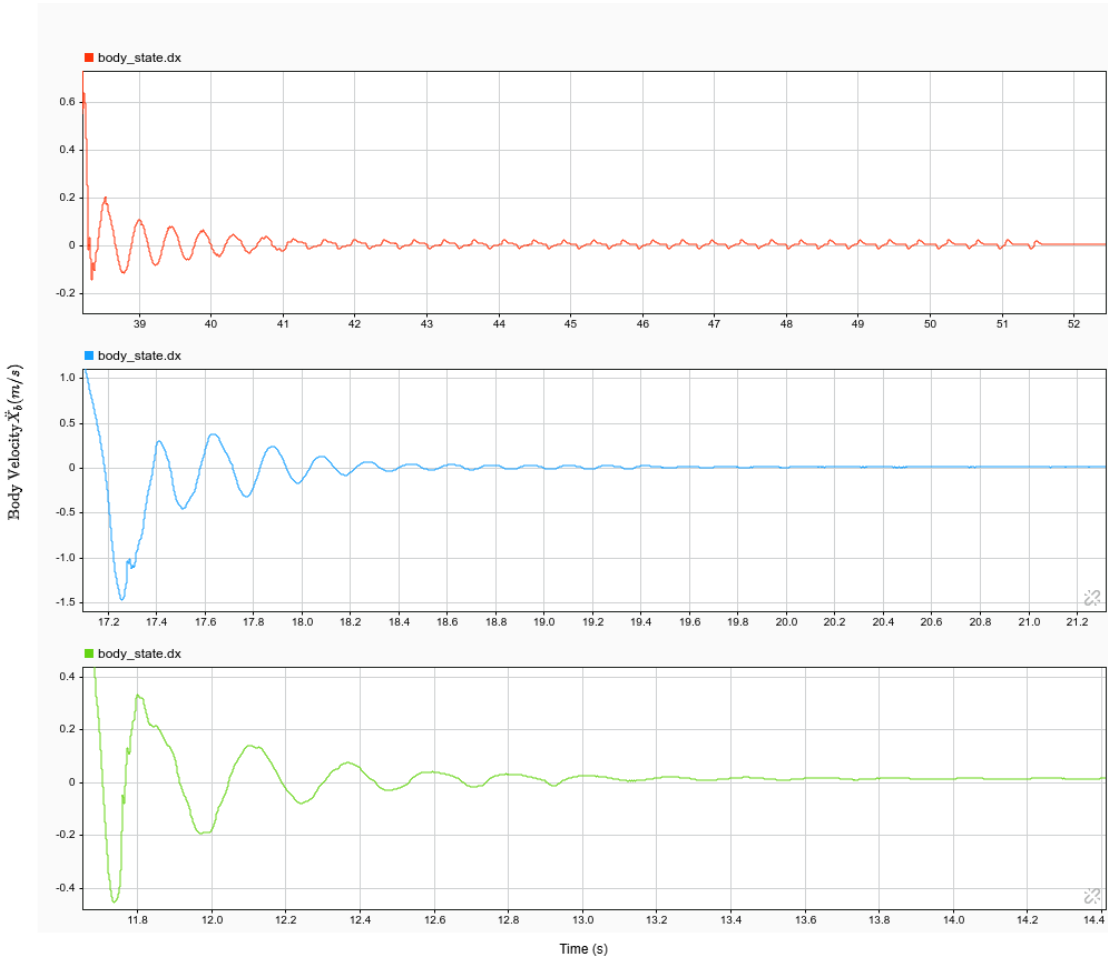

Figure 45: Deceleration curves of the 0.5 m/s (top), 1.5 m/s (centre), and 4.0 m/s (bottom) Simulink body velocity results. Highlighting the second phase of oscillation.

Figure 46: Images of the 0.5 m/s (left), 1.5 m/s (centre), and 4.0 m/s (right) final deceleration stance.

The second phase of oscillations occurs as the body adjusts to maintain stability after the feet abruptly stop. This adjustment results in a stabilization period of approximately 13 seconds before the system comes to rest, as illustrated in Figure 45. The primary cause of these oscillations is the displacement of the CoM outside the support polygon formed by the contact area of the two foot soles with the ground. Notably, the foot positions are located directly beneath the body, creating a small support polygon area that resembles the dynamics of an inverted pendulum (Figure 46). To mitigate these oscillations, increasing the lateral separation of the legs during deceleration would enhance the support polygon area, thereby improving stability and reducing reliance on pendulum-like dynamics.

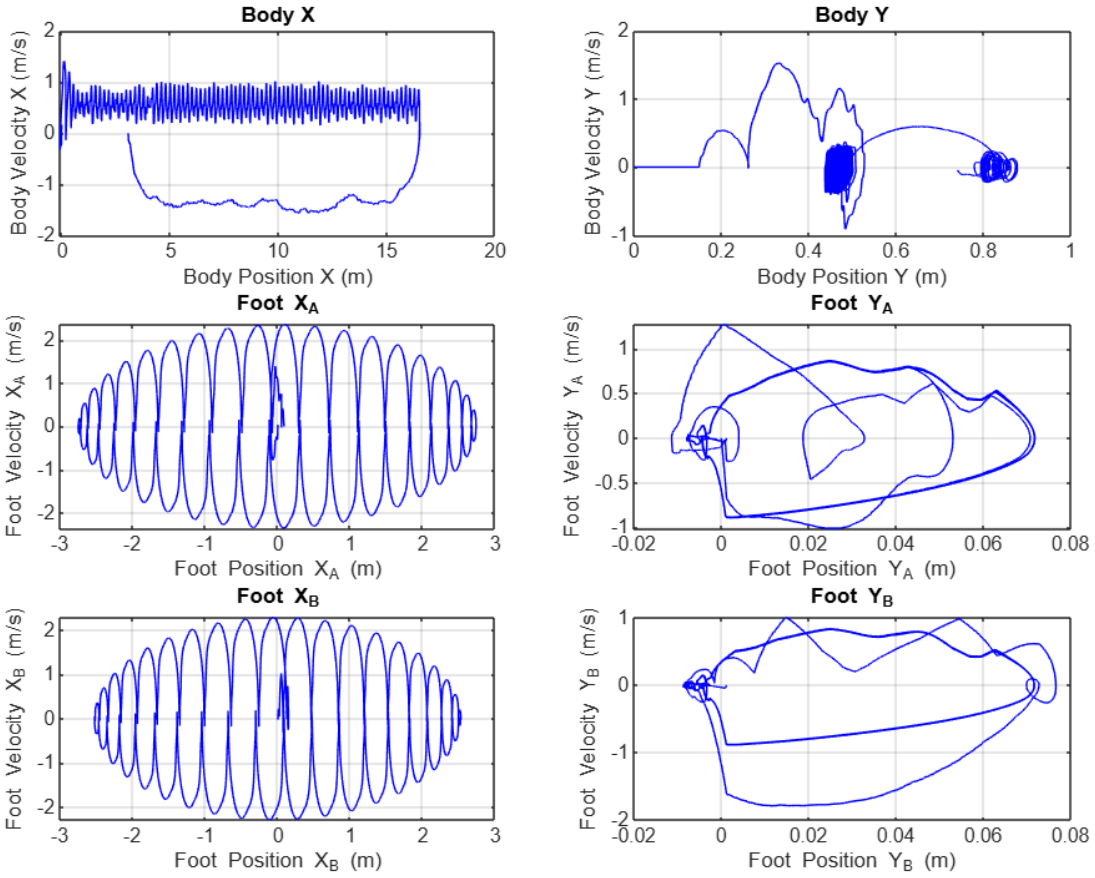

Figure 47: Phase plots illustrating the body dynamics (top) derived from Simulink data, along with the foot positions of leg A (centre) and leg B (bottom) obtained from the Simscape simulation at 0.5 m/s. The foot positions in the X-direction are constrained to oscillate between -2.8 and 2.8 to emphasize the limit cycle characteristics of the gait.

The phase plots in Figure 47 demonstrate a stable limit cycle for both body and foot positions. The $\mathbf{X_b}$ data indicate an average speed of 0.5 m/s over the total distance, while $\mathbf{Y_b}$ exhibits periodic, elliptical behaviour ranging between 0.4 m and 0.6 m. The foot positions in the X-direction show consistent oscillatory behaviour at a steady velocity, reflected in the elliptical curves, confirming the robot's steady-state gait.

In contrast, the foot positions in the Y-direction show less periodic motion due to unmodelled deformations between the feet and ground, with values oscillating between -0.02 m and 0.01 m. However, oscillatory behaviour is evident during the foot flight phases. Non-periodic curves in all instances can be attributed to the acceleration and deceleration phases of the trajectories.

6.5.3 Walk-To-Run Experiments (1.5 m/s)

The walk-to-run transition simulation (1.5 m/s) began acceleration at t = 5 seconds, reached steady-state gait at t = 5.65 seconds, and continued until t = 17 seconds. Figures 48, 49, and 50 depict snapshots of the acceleration phase, steady-state run, and deceleration phase, respectively.

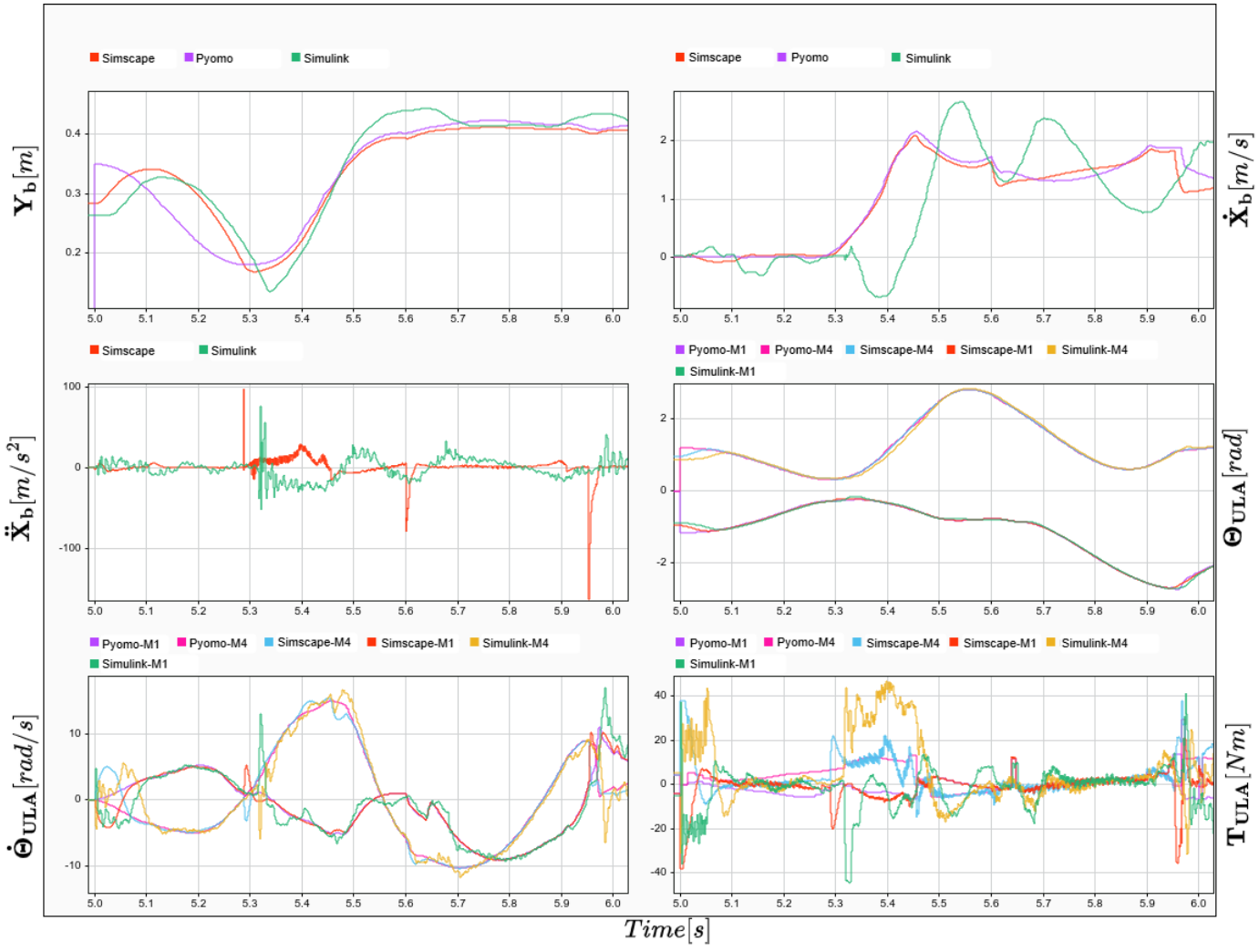

Figure 48: Snapshot of the 1.5 m/s results, displaying Pyomo, Simscape and Simulink curves. Between time-stamp t = 5 seconds and t = 5.65 seconds highlights the acceleration trajectories.

During the acceleration phase (Figure 48), the states exhibit behaviour similar to the 0.5 m/s simulation. However, the Simulink $\mathbf{Y_b}$ shows a greater overshoot of

0.05 m, exceeding that of the 0.5 m/s case. Additionally, the Simulink $\dot{\mathbf{X}}_\mathbf{b}$ displays larger tracking errors, reaching up to 2 m/s, but achieves the desired velocity in 0.47 seconds—an improvement over the 0.5 m/s case. This results in anticipated higher $\ddot{\mathbf{X}}_\mathbf{b}$ and motor torque peaks.These state response characteristics arise from increased velocity and torque demands, with tracking errors struggling to adapt to rapid velocity changes, highlighting the limitations discussed in the Walking Experiments Chapter.

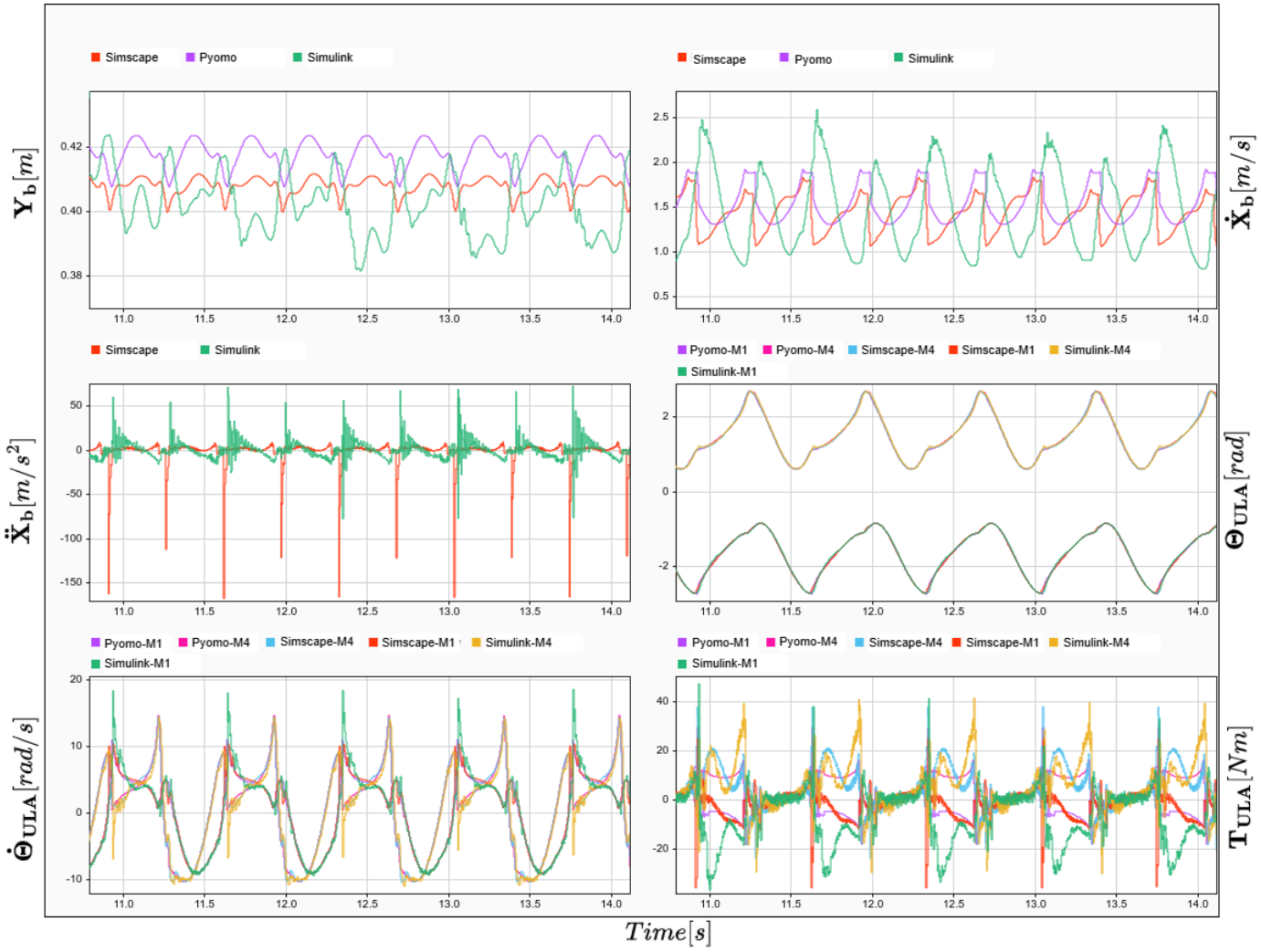

Figure 49: Snapshot of the 1.5 m/s walk-to-run transition velocity, displaying Pyomo, Simscape, and Simulink curves. This snapshot displays the steady-state gait.

In the steady-state phase (Figure 49), the $\mathbf{Y_b}$ state from the Simscape curve closely

follows the Pyomo reference, though it exhibits tracking errors of up to 0.1 m. In contrast, the Simulink curve deviates significantly, with errors reaching up to 0.25 m.

For $\dot{\mathbf{X}}_\mathbf{b}$, all three models oscillate around 1.5 m/s. However, the Simulink curve shows substantial errors (approximately 1.1 m/s) and a 0.2 seconds delay relative to Pyomo and Simscape. The $\ddot{\mathbf{X}}_\mathbf{b}$ curves exhibit larger acceleration magnitudes compared to the 0.5 m/s case, driven by increased ground contact forces and ground/foot deformations due to higher torque demands, as evidenced by the motor torque curves.

The motor torque reaches its peak limit during both the acceleration and steady-state phases, while the motor velocity curves show greater magnitude variations in both the Simscape and Simulink models. These delays and errors stem from rapid velocity changes, increased torque demands, and the transition between half-gait trajectories to the opposite leg.

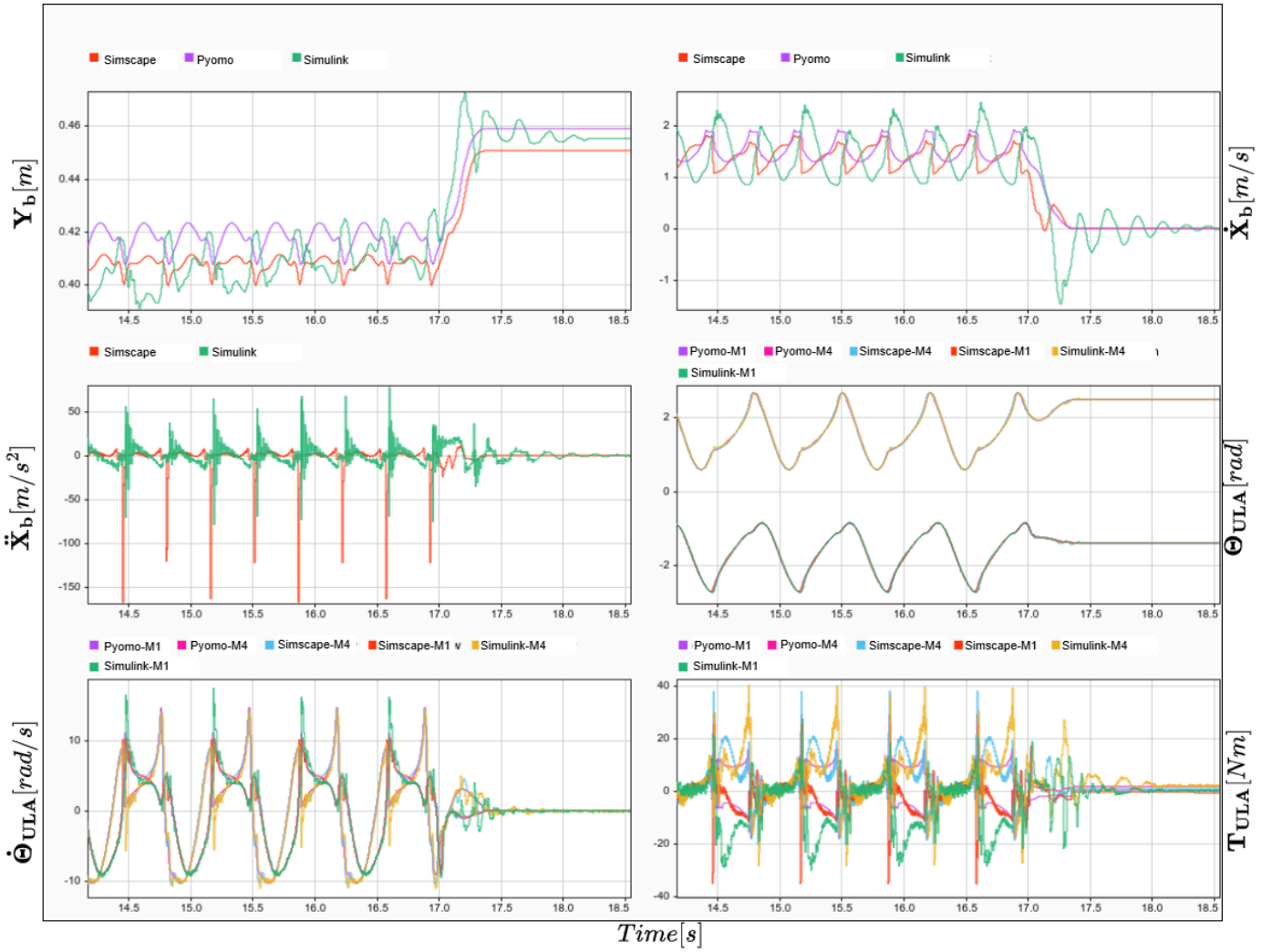

Figure 50: Snapshot of the 1.5 m/s walk-to-run, displaying Pyomo, Simscape, and Simulink curves when the robot decelerates.

During the deceleration phase (Figure 50), the Simulink $\mathbf{Y_b}$ curve exhibits a larger overshoot error (0.02 m) but a smaller steady-state error (0.005 m) compared to the 0.5 m/s deceleration case. Conversely, the Simulink $\dot{\mathbf{X}}_\mathbf{b}$ curve shows a larger undershoot of 1.8 m/s relative to the 0.5 m/s simulation.

As shown in Figure 50, the robot undergoes two oscillation phases. During the first phase, the feet come to rest within 0.4 seconds, faster than in the 0.5 m/s simulation. Figure 45 illustrates the second phase, where the robot halts completely and stabilize after approximately 2.6 seconds, also an improvement over the 0.5 m/s

deceleration. This enhanced gait termination rate is attributed to the robot's stance. Specifically, Figure 46 reveals an increased foot separation distance, which provides a larger stability support polygon area, effectively reducing body oscillations.

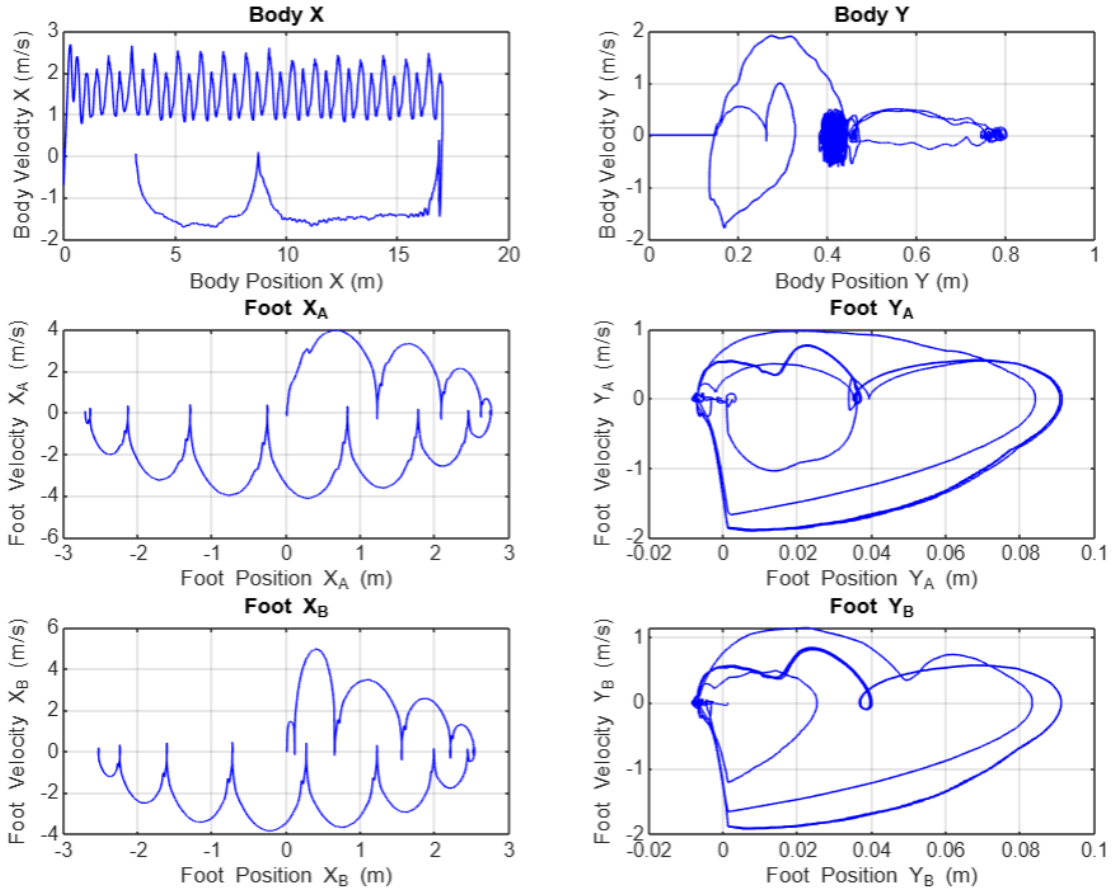

Figure 51: Phase plots illustrating the body dynamics (top) derived from Simulink data, along with the foot positions of leg A (centre) and leg B (bottom) obtained from the Simscape simulation at 1.5 m/s. The foot positions in the X-direction are constrained to oscillate between -2.8 and 2.8 to emphasize the limit cycle characteristics of the gait.

The phase plots in Figure 51 resemble those from the 0.5 m/s case (stable limit cycle). However, $\mathbf{X_b}$ shows an average speed of 1.5 m/s, while $\mathbf{Y_b}$ exhibits periodic, elliptical behaviour near 0.4 m, indicating a reduced body height to achieve higher $\dot{\mathbf{X}}_\mathbf{b}$.

Foot positions in the X-direction display a larger wavelength, reflecting increased stride length, supported by greater Y-direction motion, allowing more swing space and time for the extended stride (compared to the 0.5 m/s case).

6.5.4 Top Speed Experiments (4.0 m/s)

The top speed simulation resulted in an average $\dot{\mathbf{X}}_\mathbf{b}$ of 4.0 m/s. The simulation began acceleration at t = 5 seconds, reached steady-state gait at t = 5.5 seconds, and continued until t = 11.2 seconds. Figures 52, 53, and 54 depict snapshots of the acceleration phase, steady-state run, and deceleration phase, respectively.

Figure 52: Snapshot of the 4.0 m/s results, displaying Pyomo, Simscape and Simulink curves. Between time-stamp t = 5 seconds and t = 5.5 seconds highlights the acceleration trajectories.

When comparing the 4.0 m/s and 1.5 m/s acceleration curves (Figure 52), the $\mathbf{Y_b}$ Pyomo curve exhibits less body height reduction during launch at 4.0 m/s than at 1.5

m/s. Consequently, the Simulink and Simscape curves show smoother transitions, although the Simulink curve overshoots and undershoots the Pyomo curve by 0.04 m and 0.08 m, respectively. The Simulink motor position and velocity curves also exhibit tracking errors and delays, failing to reach desired values.

The Simulink $\dot{\mathbf{X}}_\mathbf{b}$ curve shows larger tracking errors (compared to the 1.5 m/s case), exceeding approximately 2.5 m/s, and fails to reach the target velocity within the allocated acceleration time. Additionally, the Simulink and Simscape $\ddot{\mathbf{X}}_\mathbf{b}$ and motor torque curves display higher peaks compared to the 1.5 m/s case, due to increased ground contact forces at higher speeds.

These tracking errors and delays in Simulink and Simscape are caused by motor torque limitations. While Pyomo torques remain within limits, the PD controller's error correction increases torque demands, pushing the motors to their limits. As a result, the PSU is unable to meet the increased current consumption.

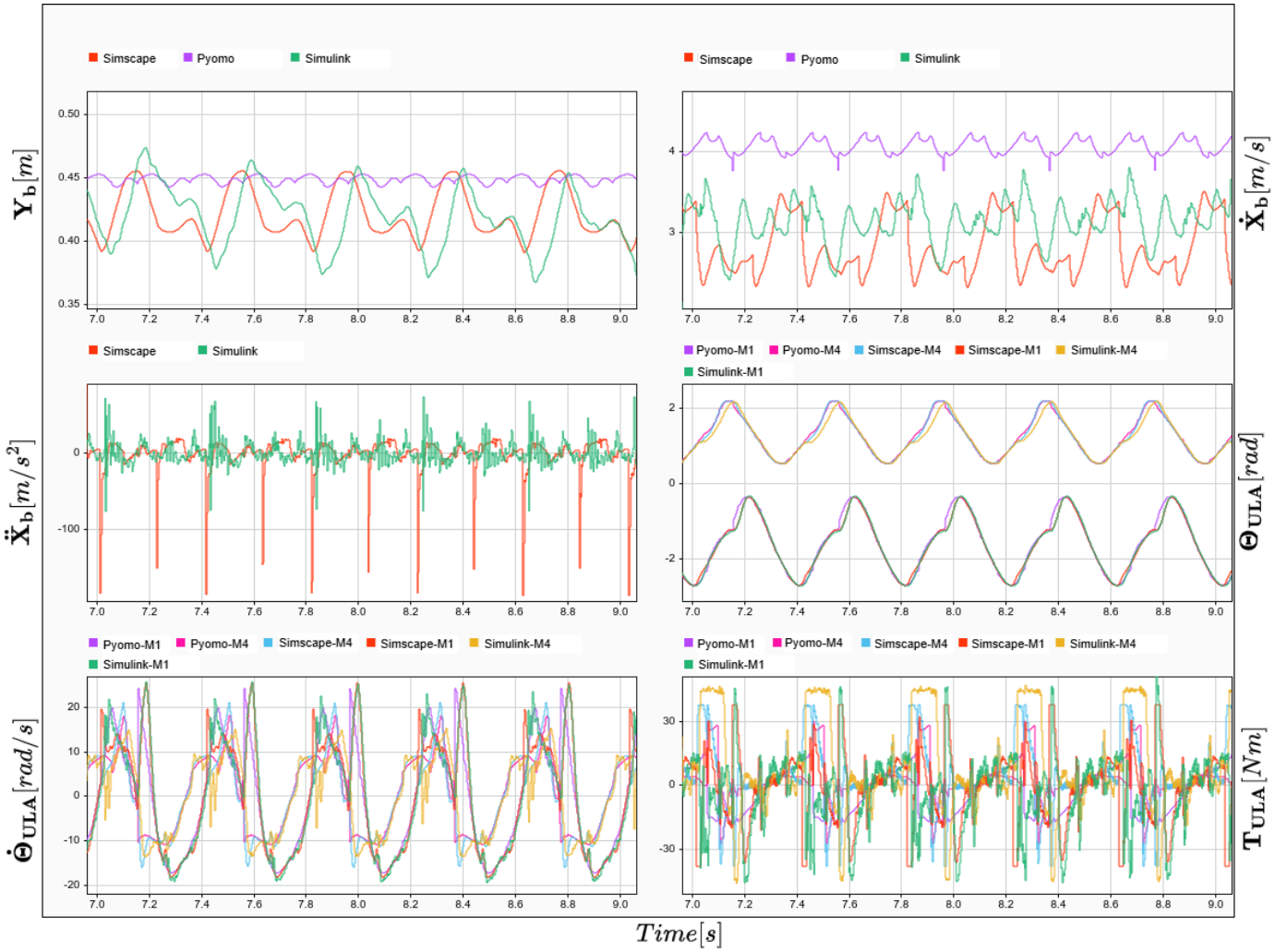

Figure 53: Snapshot of the 4.0 m/s top speed, displaying Pyomo, Simscape, and Simulink curves. This snapshot displays the steady-state gait.

In the steady-state phase (Figure 53), the $\mathbf{Y_b}$ state in the Simulink and Simscape models fails to track the Pyomo curve, with delays and tracking errors up to 0.07 m. The motor position curves also reveal response delays between the Simulink and Pyomo models. For the $\dot{\mathbf{X}}_\mathbf{b}$ state, the Simulink and Simscape models fail to achieve the desired average velocity, peaking at 3.7 m/s but averaging only 3.2 m/s, while exhibiting significant oscillations.

These issues arise from increased ground contact forces, causing vibrations during impact and deformation between the foot and ground, which reduce propulsion efficiency (as seen in the $\ddot{\mathbf{X}}_\mathbf{b}$ curves). Additionally, the robot's amplified inertia at

higher speeds exacerbates disturbances, which the PD controller struggles to manage due to high torque demands. This is evident in the motor torque curves, where both Simulink and Simscape consistently reach their maximum torque limits.

Figure 54: Snapshot of the 4.0 m/s top speed simulation, displaying Pyomo, Simscape, and Simulink curves when the robot decelerates.

During the deceleration phase (Figure 54), the Simulink $\mathbf{Y_b}$ curve exhibits a larger overshoot (0.05 m) but a smaller steady-state error compared to the 1.5 m/s case. Similarly, the Simulink $\dot{\mathbf{X}}_\mathbf{b}$ curve shows slower response times (0.5 seconds) but with reduced overshoot (0.2 m/s).

The robot experiences two oscillation phases during deceleration. In the first phase,

the feet come to rest within 0.3 seconds, faster than in the 1.5 m/s simulation. In the second phase, shown in Figure 45, the robot fully stabilizes after 2.2 seconds, also an improvement over the 1.5 m/s case. These improvements are attributed to the increased foot separation distance (Figure 46).

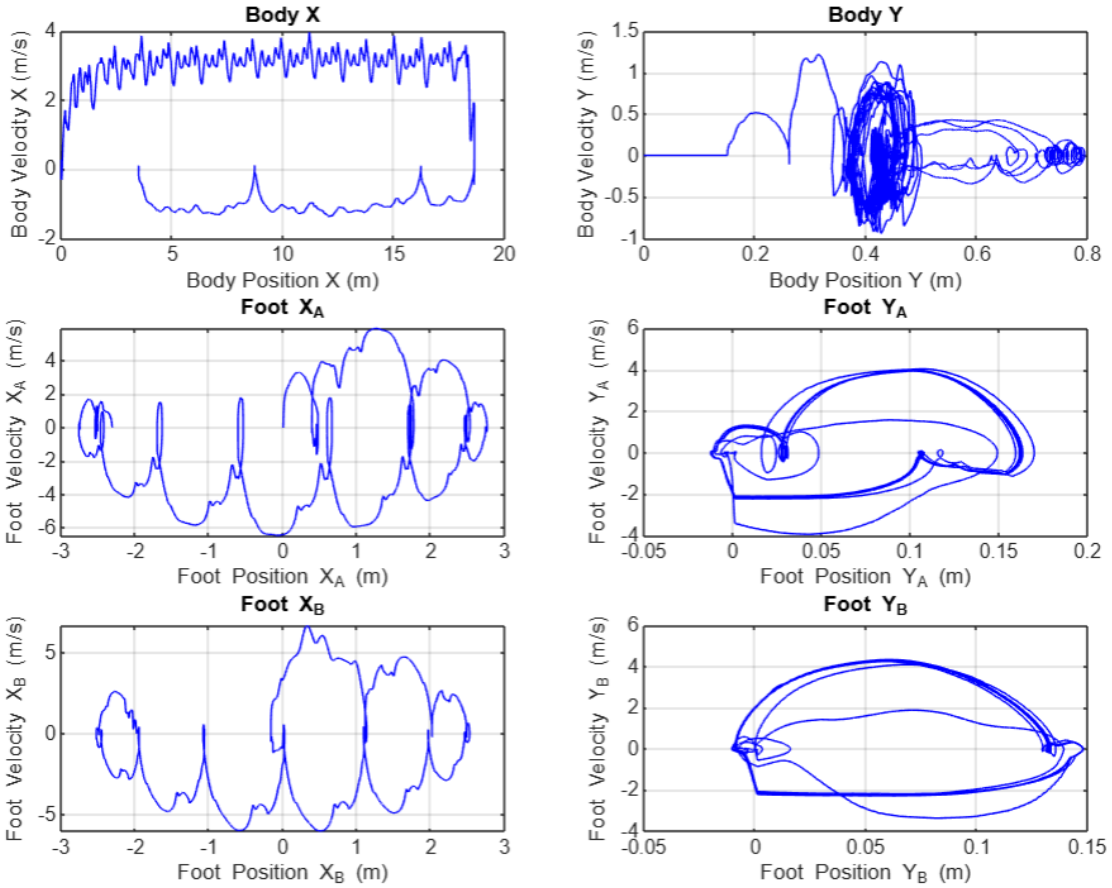

Figure 55: Phase plots illustrating the body dynamics (top) derived from Simulink data, along with the foot positions of leg A (centre) and leg B (bottom) obtained from the Simscape simulation at 4.0 m/s. The foot positions in the X-direction are constrained to oscillate between -2.8 and 2.8 to emphasize the limit cycle characteristics of the gait.

The phase plots in Figure 55 resemble those from the 1.5 m/s case (stable limit cycle). However, $\mathbf{X_b}$ shows an average speed of 3.2 m/s, while $\mathbf{Y_b}$ exhibits periodic, elliptical behaviour near 0.4 m at high $\dot{\mathbf{Y}}_\mathbf{b}$, indicating a reduced body height to achieve a higher $\dot{\mathbf{X}}_\mathbf{b}$.

Foot positions in the X-direction display a larger wavelength, reflecting increased stride length, supported by greater Y-direction motion, allowing more swing space

and time for the extended stride (compared to the 1.5 m/s case). Furthermore, the foot positions in the X-direction display phases of constant velocity - highlighting the legs maintain a constant foot velocity while in the flight phase.

6.5.5 Summary

The walking experiments (0.5 m/s) demonstrated the effectiveness of stitched trajectories, with acceleration, steady-state, and deceleration phases analysed. Simulink and Simscape closely followed Pyomo references but exhibited tracking errors and oscillations due to unmodelled dynamics and controller limitations. At higher speeds (1.5 m/s and 4.0 m/s), larger deviations and delays were observed, driven by increased torque demands, elevated ground contact forces, and deceleration spikes, reflecting propulsion efficiency challenges. The target top speed of 4.0 m/s could not be achieved, with a maximum speed of 3.2 m/s reached due to motor torque limitations. Stability improved with wider foot separation during deceleration. Phase plots confirmed steady-state limit cycles across experiments, with increased stride length and body dynamics adaptations at higher speeds. Motor torque and ground contact modelling limitations were persistent challenges across simulations.

To conclude on the outlined improvements to the robot's model and performance, a more accurate motor model is required by increasing rotor inertia and adding a friction model, resulting in better simulation results. To reduce power losses and improve actuator performance, it suggests using shorter, thicker cables and a higher-capacity power supply. Discrepancies between Simscape and Pyomo simulations in contact dynamics were noted, particularly in body deceleration and foot deflection.

Mechanical issues like loose bolts and rotating pins were identified and linked to high impact forces, requiring stronger joints. Stability improved with wider leg spacing during deceleration. The chapter recommends exploring whole-body control to address body height drift and enhancing computational speed for trajectory solving.

7 Improvements and Conclusion

7.1 Improvements and Future Work

To improve the capabilities of the robot, additional care should be given to the model accuracy. Step test simulations were conducted in Simscape, by adjusting the motor model. Specifically increasing the inertia value of the motor (to double its current value) and implementing a friction block (offered in Simscape). The friction model consisted of a Coulomb value 1 Nm (tested on motors) and a Viscous friction value where the motor ran a constant velocity (and took worst case), plotted a torque-speed curve and assumed the friction model was a straight line. Two points were selected (0.5;0.1) and (1;0.7) to obtain a gradient of B = 1.2. The resulting step test is shown in Figure 56 - which displays an improved model compared to the existing model (operating closer to the actual motor).

To address performance constraints, an effective improvement would be to reduce cable lengths and utilize cables with a higher cross-sectional area, thereby increasing current-carrying capacity. This adjustment would minimize power losses experienced at the motors. In addition, adopt a PSU with greater power capacity to accommodate the increased current demand, which is directly linked to the actuator limitations. This is evident in the 4.0 m/s torque curve, where torque limits were reached due to elevated PD controller errors, preventing the robot from achieving the desired body velocity. Enhancing the power supply would also expand the solver's search space by relaxing the bounds of the torque-speed curve. Furthermore, a critical aspect for refinement is the contact model. The drop test results reveal discrepancies between the Simscape and Pyomo models, particularly in foot deflection and GRFs. Additionally, the Simscape data exaggerates the deceleration of the body during touchdown compared to the Simulink data, indicating that the robot deforms less than anticipated upon impact.

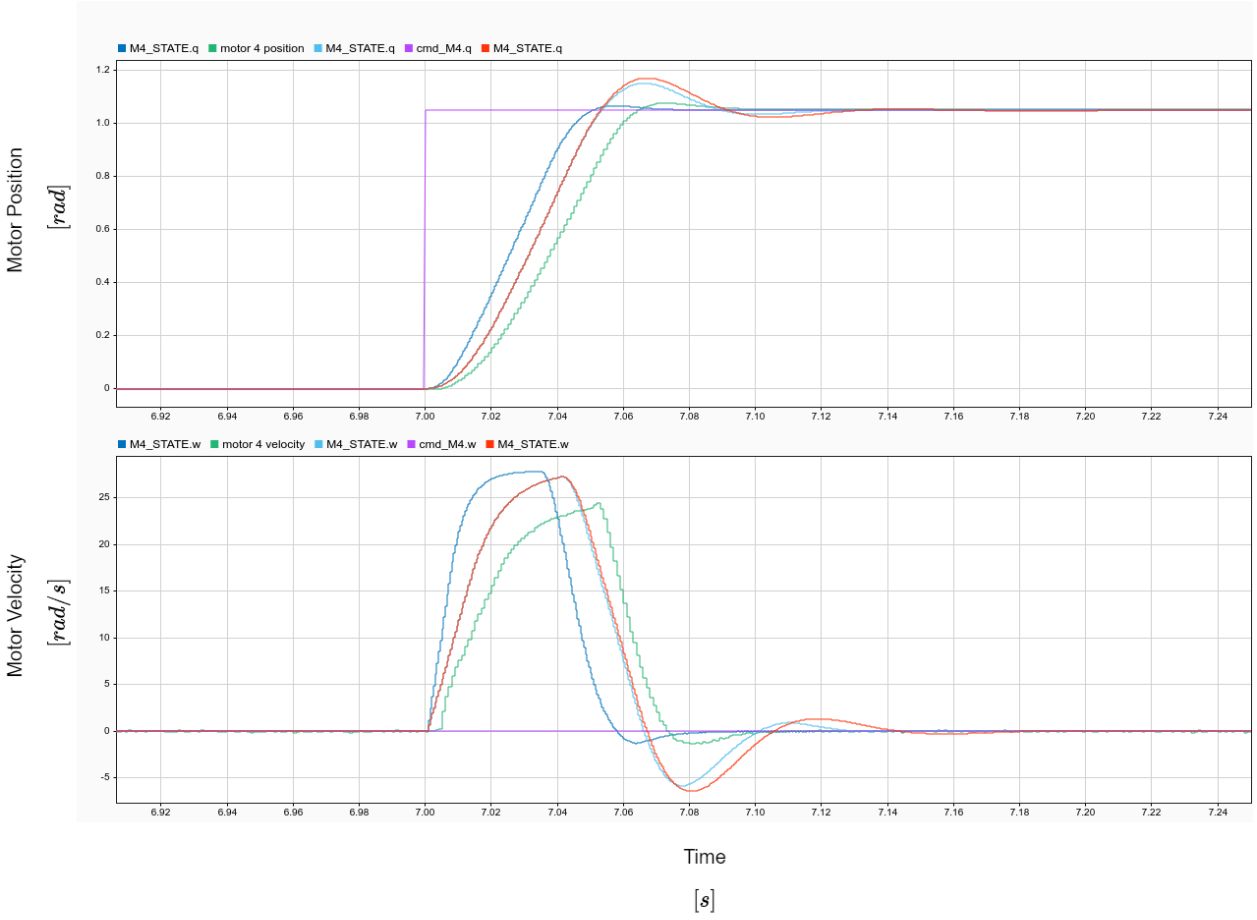

Figure 56: Step test of the updated Simscape motor model, incorporating both inertia adjustments and the friction model. The command signal, Simscape result using the current model implementation, actual motor response, Simscape result with increased rotor inertia but no friction, and the Simscape result with both friction and increased rotor inertia is shown in purple, dark blue, green, red, and cyan, respectively.

A key outcome of these experiments was the oscillations observed during deceleration. Specifically, it was observed that increasing the distance between the legs improves the robot's stability by expanding the support polygon area, which in turn reduces body oscillations. An improvement to this study would be to ensure that all deceleration trajectory curves are optimized to solve for the maximum final leg separation, further enhancing stability and minimizing oscillatory behaviour.

While TO was the primary approach used in this study, investigating the control of the entire system, particularly through whole-body dynamic control, is an important area for improvement. This is especially relevant since, even at low velocities

(0.5 m/s), the robot's body height tends to drift from the anticipated value predicted by Pyomo. This drift becomes critical at higher speeds, where any deviation between the robot's actual position and the expected trajectory could lead to potential damage, particularly in the absence of feedback mechanisms to correct for such discrepancies.

On the mechanical side, executed runs revealed issues with the bolt securing the foot and ankle joint, as it loosened due to high impact forces. Additionally, the links connected via a pin showed signs of rotation, potentially bending the bolt inside. To address these issues, a thorough analysis of the expected GRFs is needed, along with improvements to enhance the robot's durability. In addition, even though sensor noise and robustness was demonstrated through the real-world robotic experiments, future work should explore more analytical techniques for quantifying robustness such as QFT or H-infinity control.

Whilst the boom remained stable at low velocities, it required additional support when running at 4.0 m/s, as the base of the boom tended to lift. Therefore, further investigation into adding support at the boom's base is necessary. On the simulation front, the runtime for most trajectory-solving tasks was between 30 minutes to an hour. Upgrading the hardware for trajectory computation would significantly improve the speed of feasibility tests and allow for the use of higher-order Radau methods to enhance solution accuracy.

7.2 Conclusion

The development of Baleka II was a direct continuation of the first iteration, with the goal of achieving rapid legged locomotion—a key area of focus in robotic engineering. By integrating TO algorithms and conducting real-world experiments, this research assessed the feasibility of implementing an open-loop controller on a bipedal robot, specifically aiming for rapid agility.

This study had four primary objectives: (1) developing an improved embedded system configuration, (2) generating control inputs using TO, (3) validating these solutions through physical model simulations, and (4) implementing them on the Baleka II robot. The first objective was accomplished using the Speedgoat Real-Time Target Machine, which seamlessly integrated with Simulink Real-Time and Simscape to execute control commands. The second objective was achieved through TO simulations using Pyomo and IPOPT, generating control solutions for walking (0.5 m/s), walking-to-running (1.5 m/s), and maximum forward speed (4.0 m/s), including acceleration, deceleration, and steady-state gait trajectories. physical parameter constraint, bounds, and cost functions aided the solver to achieve the optimal solutions. Multiple simulations were run with a variety of seeding values, these solutions

were compared, where the optimal solution for implementation was selected on the categories of interest. For the third objective, Simscape was employed to validate the feasibility of the optimized trajectories. This ensured that the trajectories under test were practical with a more realistic model. The final objective involved the implementation on the physical robot, which was equipped with four motors using a CAN bus protocol compatible with the Simulink and Speedgoat system. Despite being open-loop, each motor was controlled by a PD controller, which provided a degree of stability in the open-loop system.

Key findings from this research showed that, the walking experiments at 0.5 m/s demonstrated effective stitched trajectories, with minimal delays and tracking errors. However, at higher speeds (1.5 m/s and 4.0 m/s) showed larger deviations and delays due to increased torque demands, actuation effort, maximum knee angles and ground contact forces. Furthermore, it was found that improving the leg separation distance during deceleration, greatly improved the settling time for the body to come to a complete stop.

In this study, Baleka II successfully accelerated from rest to a 3.2 m/s run. While its top speed lags behind robots such as Cassie and Raptor, Baleka II demonstrated the ability to transition quickly into a steady-state running gait and terminate it rapidly yet stably. Furthermore, Baleka II's acceleration and deceleration rates placed it within a competitive range compared to other robots.

These findings contribute valuable insights into the potential for bipedal robots to perform rapid transitions using open-loop optimal control. This technology holds promise for search-and-rescue operations where legged robots must rapidly accelerate and decelerate in a stable manner to avoid obstacles and reach their destination. Additionally, it could be beneficial in humanitarian aid, industrial fields, and environmental monitoring for surveillance and ground assistance, offering improved response times. Lastly, it could aid in the transportation and logistics sector by moving goods or performing tasks in warehouses and distribution centres - improving efficiency and reducing manual labour.

Baleka II represents the second iteration in a series of many future versions, focusing on rapid transitions and steady-state running through TO. Future work will aim to address the identified improvements, and future scope such as incorporating appendages to enhance deceleration, implementing foot sensors, exploring new control techniques, and testing the robot on uneven terrain without the support of a boom.

7.3 Final Remarks

The journey of developing Baleka II has been both challenging and rewarding. This research has demonstrated the capability of bipedal robots to achieve rapid agility using open-loop control, contributing to a broader understanding of legged locomotion in robotics. The potential applications of this technology, from search and rescue to industrial and entertainment uses, reflect the significance of these advancements.

While Baleka II has made strides in the realm of rapid locomotion, there is still much room for further development. Future iterations will continue to push the boundaries of speed, control, and adaptability, bringing us closer to fully autonomous, agile bipedal robots capable of navigating complex environments. This project represents just one step in the ongoing journey of robotic innovation, and I am excited for the advancements yet to come.

APPENDICES

A Boom Configuration

This appendix provides a detailed derivation of the CoM position of the boom, which is utilized in the formulation of the EoM.

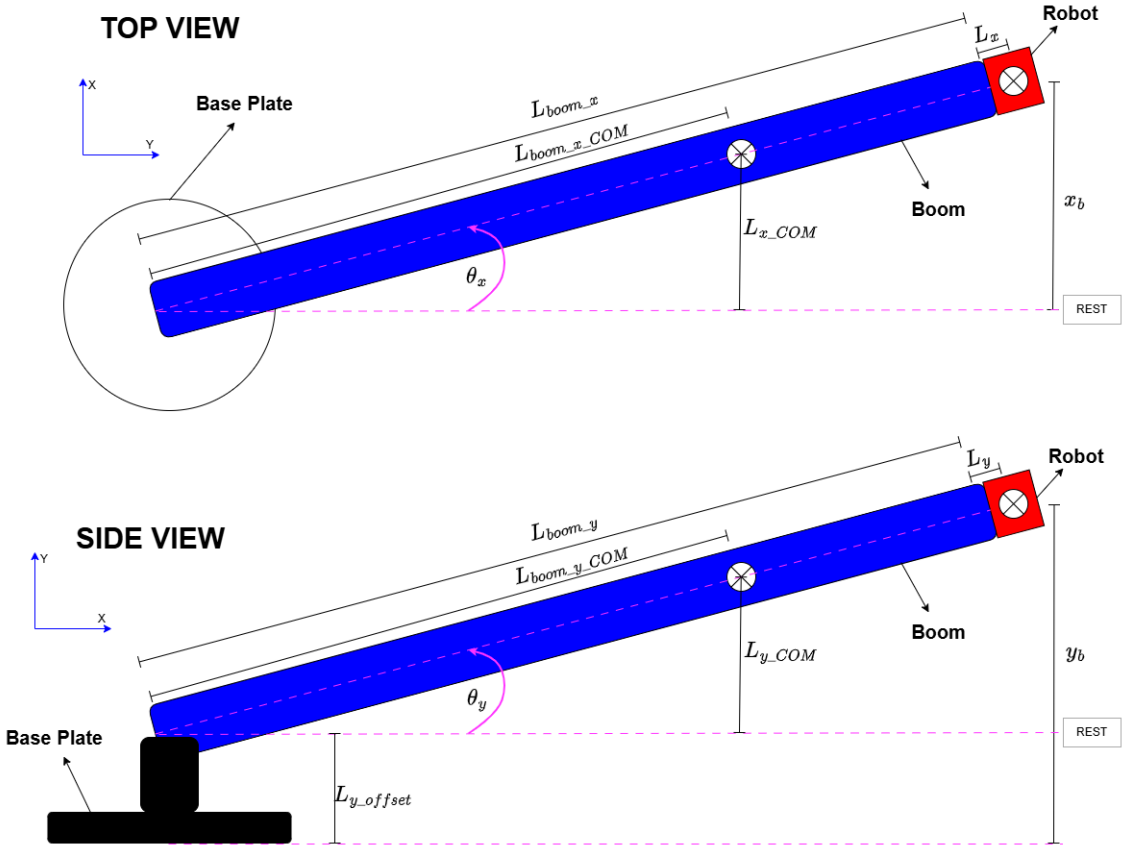

Figure A: Representation of boom diagram used to determine the boom's CoM with respect to the robot's body CoM.

Given from boom diagram:

$$\sin(\boldsymbol{\Theta}_{\mathbf{x}}) = \frac{\mathbf{x_b}}{L_{\text{boom\_x}} + L_x} \tag{A.1}$$

$$\sin(\boldsymbol{\Theta}_{\mathbf{x}}) = \frac{L_{x\_\text{CoM}}}{L_{\text{boom\_x\_CoM}}} \tag{A.2}$$

$$\sin(\mathbf{\Theta_y}) = \frac{\mathbf{y_b} - L_{y\_\text{offset}}}{L_{\text{boom\_y}} + L_y} \tag{A.3}$$

$$\sin(\mathbf{\Theta_y}) = \frac{L_{y\_\text{CoM}}}{L_{\text{boom\_y\_CoM}}} \tag{A.4}$$

Reorganize equations and solve for boom CoM with respect to the robot's body CoM:

$$\mathbf{r}_{\text{boom\_x}} = \mathbf{x_b} - \left(\frac{\mathbf{x_b}}{L_{\text{boom\_x}} + L_x}\right) L_{\text{boom\_x\_CoM}} \tag{A.5}$$

$$\mathbf{r}_{\text{boom\_y}} = \mathbf{y_b} - L_{y\_\text{offset}} - \left(\frac{\mathbf{y_b} - L_{y\_\text{offset}}}{L_{\text{boom\_y}} + L_y}\right) L_{\text{boom\_y\_CoM}} \tag{A.6}$$

B Equations of Motion

This section outlines the formulation of the EoM, employing the Manipulator Equation methodology. The derivation explicitly incorporates the dynamic effects of both the boom and the motor.

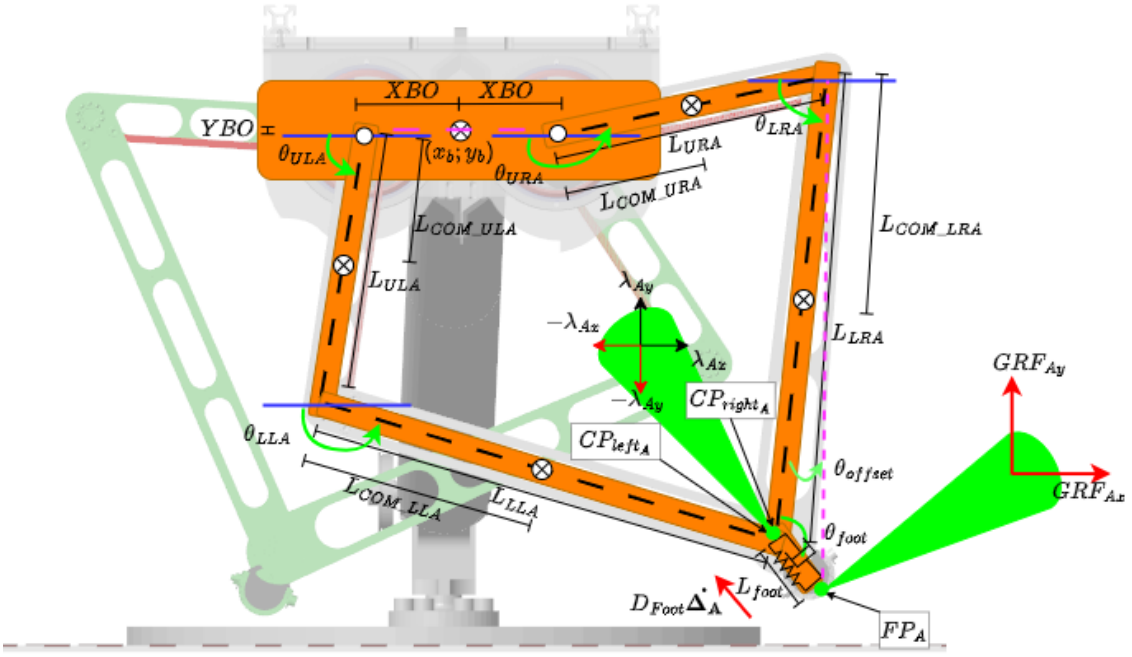

Figure B: Representation of the robot's sketch used to determine the manipulator equation.

Define the state variables:

$$\mathbf{q} = \left[\mathbf{x_b}; \mathbf{y_b}; \mathbf{\Theta_{ULA}}; \mathbf{\Theta_{LLA}}; \mathbf{\Theta_{URA}}; \mathbf{\Theta_{LRA}}; \mathbf{\Delta_A}; \mathbf{\Theta_{ULB}}; \mathbf{\Theta_{LLB}}; \mathbf{\Theta_{URB}}; \mathbf{\Theta_{LRB}}; \mathbf{\Delta_B}\right]^T \tag{B.1}$$

Define the state variable derivatives (velocity and acceleration):

$$\dot{\mathbf{q}} = \frac{d}{dt}\mathbf{q} \tag{B.2}$$

$$\ddot{\mathbf{q}} = \frac{d}{dt}\dot{\mathbf{q}} \tag{B.3}$$

Define the position vectors:

$$\mathbf{r_{body}} = \begin{bmatrix} \mathbf{x_b} \\ \mathbf{y_b} \end{bmatrix} \tag{B.4}$$

The following is obtained from Appendix A:

$$\mathbf{r_{boom}} = \begin{bmatrix} \mathbf{r}_{\text{boom\_x}} \\ \mathbf{r}_{\text{boom\_y}} \end{bmatrix} \tag{B.5}$$

The below matrices define the position vectors for the links of leg A; however, the same applies for leg B:

$$\mathbf{r_{ULA}} = \begin{bmatrix} \mathbf{x_b} - XB0 - L_{CoM\_ULA}\cos(\mathbf{\Theta_{ULA}}) \\ \mathbf{y_b} - YB0 - L_{CoM\_ULA}\sin(\mathbf{\Theta_{ULA}}) \end{bmatrix} \tag{B.6}$$

$$\mathbf{r_{URA}} = \begin{bmatrix} \mathbf{x_b} + XB0 + L_{CoM\_URA}\cos(\pi - \mathbf{\Theta_{URA}}) \\ \mathbf{y_b} - YB0 - L_{CoM\_URA}\sin(\pi - \mathbf{\Theta_{URA}}) \end{bmatrix} \tag{B.7}$$

$$\mathbf{r_{LLA}} = \begin{bmatrix} \mathbf{x_b} - XB0 - L_{ULA}\cos(\mathbf{\Theta_{ULA}}) + L_{CoM\_LLA}\cos(\pi - \mathbf{\Theta_{LLA}}) \\ \mathbf{y_b} - YB0 - L_{ULA}\sin(\mathbf{\Theta_{ULA}}) - L_{CoM\_LLA}\sin(\pi - \mathbf{\Theta_{LLA}}) \end{bmatrix} \tag{B.8}$$

$$\mathbf{r_{LRA}} = \begin{bmatrix} \mathbf{x_b} + XB0 + L_{URA}\cos(\pi - \mathbf{\Theta_{URA}}) - L_{CoM\_LRA}\cos(\mathbf{\Theta_{LRA}}) \\ \mathbf{y_b} - YB0 - L_{URA}\sin(\pi - \mathbf{\Theta_{URA}}) - L_{CoM\_LRA}\sin(\mathbf{\Theta_{LRA}}) \end{bmatrix} \tag{B.9}$$

$$\mathbf{r_{\Delta_A}} = \begin{bmatrix} \mathbf{x_b} + XB0 + L_{URA}\cos(\pi - \mathbf{\Theta_{URA}}) - L_{LRA}\cos(\mathbf{\Theta_{LRA}} - \theta_{offset}) \\ +(L_{foot} + \mathbf{\Delta_A})\cos(\theta_{foot} - \mathbf{\Theta_{LRA}} - \theta_{offset}) \\ \mathbf{y_b} - YB0 - L_{URA}\sin(\pi - \mathbf{\Theta_{URA}}) - L_{LRA}\sin(\mathbf{\Theta_{LRA}} - \theta_{offset}) \\ -(L_{foot} + \mathbf{\Delta_A})\sin(\theta_{foot} - \mathbf{\Theta_{LRA}} - \theta_{offset}) \end{bmatrix} \tag{B.10}$$

Calculate the linear velocity vectors:

$$\dot{\mathbf{r}} = \frac{\partial \mathbf{r}}{\partial \mathbf{q}}\dot{\mathbf{q}} \tag{B.11}$$

Calculate the angular velocity vectors, including the boom ($\mathbf{\Theta_x}$ and $\mathbf{\Theta_y}$) but excluding the body:

$$\omega = \dot{\mathbf{q}} \tag{B.12}$$

Calculate the kinetic energies of each body, including the kinetic energy generated from each motor (both leg A and leg B):

$$\mathbf{T_{motors}} = \frac{1}{2}N_{GR}I_{rotor}(\dot{\mathbf{\Theta}}^2_{\mathbf{ULA}} + \dot{\mathbf{\Theta}}^2_{\mathbf{URA}} + \dot{\mathbf{\Theta}}^2_{\mathbf{ULB}} + \dot{\mathbf{\Theta}}^2_{\mathbf{URB}}) \tag{B.13}$$

$$\mathbf{T} = \sum_{i=1}^{r} \left(\frac{1}{2} M_i \dot{\mathbf{r}}_i^2 + \frac{1}{2} I_i \omega_i^2 \right) + \mathbf{T_{motors}} \tag{B.14}$$

Calculate the potential energies, including the potential energy generated from the virtual spring model (both leg A and leg B):

$$\mathbf{V_{foot}} = \frac{1}{2} K_{spring} (\Delta_A^2 + \Delta_B^2) \tag{B.15}$$

$$\mathbf{V} = \sum_{i=1}^{r} M_i \mathbf{g} \mathbf{r}_{\mathbf{y_i}} + \mathbf{V_{foot}} \tag{B.16}$$

Define the Manipulator Equation:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G} = \mathbf{B}\mathbf{u} + \mathbf{R} + \mathbf{A}\boldsymbol{\Lambda} \tag{B.17}$$

Solve for the $\mathbf{M}$ (mass/inertia), $\mathbf{C}$ (Coriolis forces), and $\mathbf{G}$ (gravitational forces) matrices. Note that the i and j notations indicate two loop iterations over the state variables.

$$\mathbf{M_{ij}} = \frac{\partial^2 T}{\partial \dot{\mathbf{q}}_\mathbf{i} \partial \dot{\mathbf{q}}_\mathbf{j}} \tag{B.18}$$

$$\mathbf{dM}_{ij} = \frac{\partial \mathbf{M_{ij}}}{\partial \mathbf{q}} \dot{\mathbf{q}} \tag{B.19}$$

$$\mathbf{C} = \mathbf{dM}\dot{\mathbf{q}} - \left(\frac{\partial T}{\partial \mathbf{q}} \right)^T \tag{B.20}$$

$$\mathbf{G} = \left(\frac{\partial V}{\partial \mathbf{q}} \right)^T \tag{B.21}$$

Define the input control matrix (with correct mapping):

$$\mathbf{Bu} = \left[0; 0; \mathbf{T_{ULA}}; 0; \mathbf{T_{URA}}; 0; 0; \mathbf{T_{ULB}}; 0; \mathbf{T_{URB}}; 0; 0 \right]^T \tag{B.22}$$

Define the damping force matrix (with correct mapping):

$$\mathbf{R} = \left[0; 0; 0; 0; 0; 0; D_{Foot}\dot{\mathbf{\Delta}}_{\mathbf{A}}; 0; 0; 0; 0; D_{Foot}\dot{\mathbf{\Delta}}_{\mathbf{B}}\right]^T \tag{B.23}$$

Define the external force matrix (with correct mapping):

The following matrices highlight the left and right connection points of the two lower links (ankle) of leg A, defined as $\mathbf{CP}_{left_A}$ and $\mathbf{CP}_{right_A}$, respectively. Note that the same equations apply to leg B.

$$\mathbf{CP}_{left_A} = \begin{bmatrix} \mathbf{x_b} - XB0 - L_{ULA}\cos(\mathbf{\Theta_{ULA}}) + L_{LLA}\cos(\pi - \mathbf{\Theta_{LLA}}) \\ \mathbf{y_b} - YB0 - L_{ULA}\sin(\mathbf{\Theta_{ULA}}) - L_{LLA}\sin(\pi - \mathbf{\Theta_{LLA}}) \end{bmatrix} \tag{B.24}$$

$$\mathbf{CP}_{right_A} = \begin{bmatrix} \mathbf{x_b} + XB0 + L_{URA}\cos(\pi - \mathbf{\Theta_{URA}}) - L_{LRA}\cos(\mathbf{\Theta_{LRA}} - \theta_{offset}) \\ \mathbf{y_b} - YB0 - L_{URA}\sin(\pi - \mathbf{\Theta_{URA}}) - L_{LRA}\sin(\mathbf{\Theta_{LRA}} - \theta_{offset}) \end{bmatrix} \tag{B.25}$$

$$\mathbf{FP_A} = \begin{bmatrix} \mathbf{CP}_{right_x} + (L_{foot} + \mathbf{\Delta_A})\cos(\theta_{foot} - \mathbf{\Theta_{LRA}} - \theta_{offset}) \\ \mathbf{CP}_{right_y} - (L_{foot} - \mathbf{\Delta_A})\sin(\theta_{foot} - \mathbf{\Theta_{LRA}} - \theta_{offset}) \end{bmatrix} \tag{B.26}$$

$$\mathbf{J}_{left_A} = \frac{\partial \mathbf{CP}_{left_A}}{\partial \mathbf{q}} \tag{B.27}$$

$$\mathbf{J}_{right_A} = \frac{\partial \mathbf{CP}_{right_A}}{\partial \mathbf{q}} \tag{B.28}$$

$$\mathbf{J}_{FP_A} = \frac{\partial \mathbf{FP_A}}{\partial \mathbf{q}} \tag{B.29}$$

The following matrices define the external forces and the Jacobians used to map these forces to the state coordinates. Although they only represent leg A, the same equations apply for leg B.

$$\mathbf{\Lambda_{GRF_A}} = \begin{bmatrix} \mathbf{GRF_{Ax}} \\ \mathbf{GRF_{Ay}} \end{bmatrix} \tag{B.30}$$

$$\mathbf{\Lambda_{CP_{left_A}}} = \begin{bmatrix} -\lambda_{\mathbf{A_x}} \\ -\lambda_{\mathbf{A_y}} \end{bmatrix} \tag{B.31}$$

$$\mathbf{\Lambda_{CP_{right_A}}} = \begin{bmatrix} \lambda_{\mathbf{A_x}} \\ \lambda_{\mathbf{A_y}} \end{bmatrix} \tag{B.32}$$

$$\mathbf{A\Lambda_A} = \mathbf{J}_{FP_A}^{\top}\mathbf{\Lambda_{GRF_A}} + \mathbf{J}_{left_A}^{\top}\mathbf{\Lambda_{CP_{left_A}}} + \mathbf{J}_{right_A}^{\top}\mathbf{\Lambda_{CP_{right_A}}} \tag{B.33}$$

Therefore, to incorporate leg B, the final external force matrix is defined as:

$$\mathbf{A\Lambda} = \mathbf{A\Lambda_A} + \mathbf{A\Lambda_B} \tag{B.34}$$

C Forward Kinematics

This appendix outlines the derivation of the kinematic equations used in the development of the Impedance Controller. Specifically, it focuses on the definition of polar coordinates relevant to the system. The following key points should be noted:

1. The below kinematics applies to both legs A and B.
2. $\mathbf{Z_{offset}} = 2.23$ radians, this defines the angle between the REST axis and the Y axis.
3. All link lengths are defined in the above Chapters.
4. $\mathbf{\Theta_{ML}}$ and $\mathbf{\Theta_{MR}}$ are the angular positions obtained from the left and right motors, respectively.

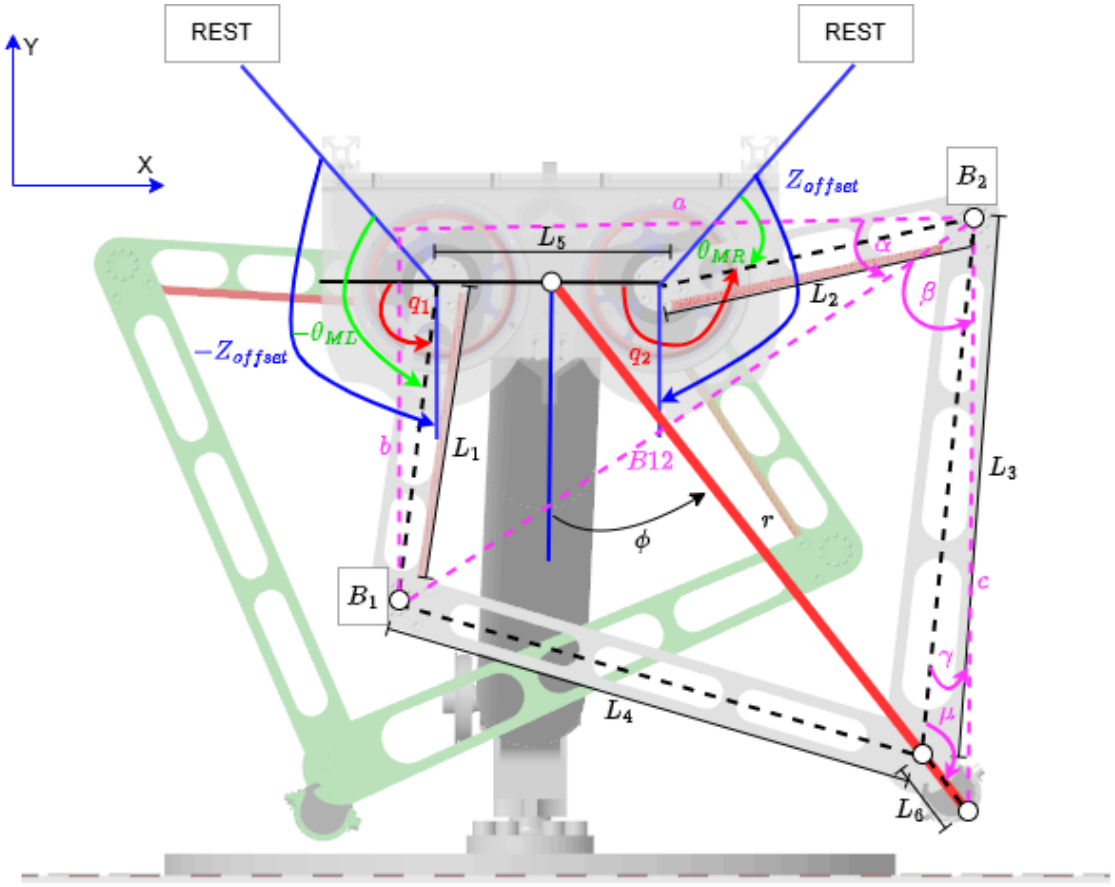

Figure C: Kinematics used for impedance control derivation.

Define the first joint angle $\mathbf{q_1}$ and $\mathbf{q_2}$:

$$q_1 = -\left(\mathbf{Z}_{\text{offset}} - \frac{\pi}{2}\right) - \theta_{ML} \tag{C.1}$$

$$q_2 = \left(\mathbf{Z}_{\text{offset}} - \frac{\pi}{2}\right) + \pi - \theta_{MR} \tag{C.2}$$

Define the position of the left and right knee ($\mathbf{B}_1$ and $\mathbf{B}_2$):

$$\mathbf{B}_1 = \begin{bmatrix} -0.5L_5 - L_1\cos(q_1) \\ L_1\sin(q_1) \end{bmatrix} \tag{C.3}$$

$$\mathbf{B}_2 = \begin{bmatrix} 0.5L_5 + L_2\cos(\pi - q_2) \\ L_2\sin(\pi - q_2) \end{bmatrix} \tag{C.4}$$

Calculate the horizontal (a) and vertical (b) distance components between knees:

$$a = |B_{1x} - B_{2x}|, \quad b = B_{1y} - B_{2y} \tag{C.5}$$

Calculate the total distance between the knees (B_{12}):

$$B_{12} = \sqrt{a^2 + b^2} \tag{C.6}$$

Calculate the knee angle q_3:

$$\alpha = \text{atan2}(b, a), \quad \beta = \arccos\left(\frac{L_4^2 - B_{12}^2 - L_3^2}{-2B_{12}L_3}\right) \tag{C.7}$$

$$q_3 = \alpha + \beta \tag{C.8}$$

Calculate the length of the virtual leg (c) and its orientation (λ):

$$c = \sqrt{L_3^2 + L_6^2 - 2L_3L_6\cos(\mu)} \tag{C.9}$$

$$\lambda = \arcsin\left(\frac{L_6\sin(\mu)}{c}\right) \tag{C.10}$$

Determine the foot coordinates in the Cartesian coordinate system:

$$X_{\text{foot}} = 0.5L_5 + L_2\cos(\pi - q_2) - c\cos(q_3 + \lambda) \tag{C.11}$$

$$Y_{\text{foot}} = L_2\sin(\pi - q_2) + c\sin(q_3 + \lambda) \tag{C.12}$$

Determine the foot coordinates in the Polar coordinate system (from the origin):

$$r = \sqrt{X_{\text{foot}}^2 + Y_{\text{foot}}^2} \tag{C.13}$$

$$\phi = \frac{\pi}{2} - \text{atan2}(Y_{\text{foot}}, X_{\text{foot}}) \tag{C.14}$$

Where atan2 restricts the angle as follows:

$$\theta = \begin{cases} \arctan\left(\frac{y}{x}\right) & \text{if } x > 0, \\ \arctan\left(\frac{y}{x}\right) + \pi & \text{if } x < 0 \text{ and } y \geq 0, \\ \arctan\left(\frac{y}{x}\right) - \pi & \text{if } x < 0 \text{ and } y < 0, \\ \frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0, \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0, \\ \text{undefined} & \text{if } x = 0 \text{ and } y = 0. \end{cases} \tag{C.15}$$

D Additional Resources

The appendix provides a comprehensive list of all supporting resources related to the Baleka II project. These resources include code, videos, simulation data, and other relevant files, organized for ease of reference. The main folder containing all resources is accessible via the following link:

Baleka II Resources

Folder Structure and Resource Descriptions:

1. Code.zip: Contains the code files developed for the project in Pyomo, Simscape, and Simulink Real-Time (Speedgoat). It also includes the CSV files used in Simscape and Simulink.
2. Robot Parameters.zip: Includes SolidWorks and STL files, along with details on the robot's link masses and lengths.
3. Simulink Real Time (Speedgoat) Results.zip: Provides experimental results from Simulink Real-Time, stored as .mat files.
4. Trajectory Optimization Simulations.zip: Contains simulation results from trajectory optimization, including the top five optimal outcomes for each locomotive task.
5. Videos.zip: Includes all recorded videos showcasing Pyomo simulations, Simscape models, and real-time experimental results.

These resources provide detailed insights and support for understanding the development and performance of the Baleka II project.

References

[1] Z. He, J. Hua, B. Adu Gyamfi, and R. Shaw, "Robotics and its advancement in modern china," *Considerations for a Post-COVID-19 Technology and Innovation Ecosystem in China*, pp. 101–114, 2022.

[2] S.-C. Chen, K.-J. Huang, W.-H. Chen, S.-Y. Shen, C.-H. Li, and P.-C. Lin, "Quattroped: a leg–wheel transformable robot," *IEEE/ASME Transactions On Mechatronics*, vol. 19, no. 2, pp. 730–742, 2013.

[3] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "Mit cheetah 3: Design and control of a robust, dynamic quadruped robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2245–2252.

[4] K. Sreenath, H.-W. Park, I. Poulakakis, and J. W. Grizzle, "Embedding active force control within the compliant hybrid zero dynamics to achieve stable, fast running on mabel," *The International Journal of Robotics Research*, vol. 32, no. 3, pp. 324–345, 2013.

[5] S. Kuindersma, R. Deits, M. Fallon, A. Valenzuela, H. Dai, F. Permenter, T. Koolen, P. Marion, and R. Tedrake, "Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot," *Autonomous robots*, vol. 40, pp. 429–455, 2016.

[6] J. Reher, W.-L. Ma, and A. D. Ames, "Dynamic walking with compliance on a cassie bipedal robot," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 2589–2595.

[7] T. Mikolajczyk, E. Mikołajewska, H. F. Al-Shuka, T. Malinowski, A. Kłodowski, D. Y. Pimenov, T. Paczkowski, F. Hu, K. Giasin, D. Mikołajewski *et al.*, "Recent advances in bipedal walking robots: Review of gait, drive, sensors and control systems," *Sensors*, vol. 22, no. 12, p. 4440, 2022.

[8] F. Delcomyn, "Biologically inspired robots," in *Bioinspiration and Robotics Walking and Climbing Robots*. IntechOpen, 2007.

[9] M. K. Habib, K. Watanabe, and K. Izumi, "Biomimetics robots from bio-inspiration to implementation," in *IECON 2007-33rd annual conference of the IEEE Industrial Electronics Society*. IEEE, 2007, pp. 143–148.

[10] D. Crowley, J. Dao, H. Duan, K. Green, J. Hurst, and A. Fern, "Optimizing bipedal locomotion for the 100m dash with comparison to human running," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 12 205–12 211.

[11] M. H. Raibert, *Legged robots that balance*. MIT press, 1986.

[12] Z. Gan, Z. Jiao, and C. D. Remy, "On the dynamic similarity between bipeds and quadrupeds: a case study on bounding," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3614–3621, 2018.

[13] Y. Yang, T. Zhang, E. Coumans, J. Tan, and B. Boots, "Fast and efficient locomotion via learned gait transitions," in *Conference on robot learning*. PMLR, 2022, pp. 773–783.

[14] M. Shafiee, G. Bellegarda, and A. Ijspeert, "Viability leads to the emergence of gait transitions in learning agile quadrupedal locomotion on challenging terrains," *Nature Communications*, vol. 15, no. 1, p. 3073, 2024.

[15] S. Shield, R. Jericevich, A. Patel, and A. Jusufi, "Tails, flails, and sails: How appendages improve terrestrial maneuverability by improving stability," *Integrative and comparative biology*, vol. 61, no. 2, pp. 506–520, 2021.

[16] C. Fisher, C. Hubicki, and A. Patel, "Do intermediate gaits matter when rapidly accelerating?" *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3418–3424, 2019.

[17] A. F. Blom, "Design of a bipedal robot for rapid acceleration and braking manoeuvres," Master's thesis, University of Cape Town, 2019.

[18] M. Thompson, K. Hoffman, L. Blythe, R. Hasler, and M. Longtain, "The coupling of stride length and foot strike in running," *Frontiers in Sports and Active Living*, vol. 4, p. 768801, 2022.

[19] G. Kenneally, A. De, and D. E. Koditschek, "Design principles for a family of direct-drive legged robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 900–907, 2016.

[20] N. Kau, A. Schultz, N. Ferrante, and P. Slade, "Stanford doggo: An open-source, quasi-direct-drive quadruped," in *2019 International conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6309–6315.

[21] A. Singh, N. Kashiri, and N. Tsagarakis, "Design of a quasi-direct-drive actuator for dynamic motions," in *Proceedings*, vol. 64. MDPI, 2020, p. 11.

[22] J. Carpentier and P.-B. Wieber, "Recent progress in legged robots locomotion control," *Current Robotics Reports*, vol. 2, no. 3, pp. 231–238, 2021.

[23] A. L. Ciancio, R. Barone, L. Zollo, G. Carpino, A. Davalli, R. Sacchetti, and E. Guglielmelli, "A bio-inspired force control for cyclic manipulation of pros-

thetic hands," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2015, pp. 4824–4827.

[24] S. Hirose, "Three basic types of locomotion in mobile robots," in *Fifth International Conference on Advanced Robotics' Robots in Unstructured Environments*. IEEE, 1991, pp. 12–17.

[25] H.-J. Kwak and G.-T. Park, "Study on the mobility of service robots," *International Journal of Engineering and Technology Innovation*, vol. 2, no. 2, p. 97, 2012.

[26] E. Yoshida, I. Belousov, C. Esteves, and J.-P. Laumond, "Humanoid motion planning for dynamic tasks," in *5th IEEE-RAS International Conference on Humanoid Robots, 2005.* IEEE, 2005, pp. 1–6.

[27] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to autonomous mobile robots*. MIT press, 2011.

[28] T. Takubo, T. Tanaka, K. Inoue, and T. Arai, "Emergent walking stop using 3-d zmp modification criteria map for humanoid robot," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, pp. 2676–2681.

[29] M. Morisawa, S. Kajita, K. Harada, K. Fujiwara, F. Kanehiro, K. Kaneko, and H. Hirukawa, "Emergency stop algorithm for walking humanoid robots," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2005, pp. 2109–2115.

[30] A. D. Perkins, *Control of dynamic maneuvers for bipedal robots*. Stanford University, 2010.

[31] R. Kram, A. Domingo, and D. P. Ferriss, "Effect of reduced gravity on the preferred walk–run transition speed," *Journal of Experimental Biology*, vol. 200, no. 4, pp. 821–826, 1997.

[32] D. L. Jindrich and M. Qiao, "Maneuvers during legged locomotion," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 19, no. 2, p. 026105, 2009.

[33] R. J. Full, T. Kubow, J. Schmitt, P. Holmes, and D. Koditschek, "Quantifying dynamic stability and maneuverability in legged locomotion," *Integrative and comparative biology*, vol. 42, no. 1, pp. 149–157, 2002.

[34] P. W. Webb, "Maneuverability-general issues," *IEEE Journal of Oceanic Engineering*, vol. 29, no. 3, pp. 547–555, 2004.

[35] F. E. Fish, "Balancing requirements for stability and maneuverability in cetaceans," *Integrative and comparative biology*, vol. 42, no. 1, pp. 85–93, 2002.

[36] A. M. Wilson, J. Lowe, K. Roskilly, P. E. Hudson, K. Golabek, and J. McNutt, "Locomotion dynamics of hunting in wild cheetahs," *Nature*, vol. 498, no. 7453, pp. 185–189, 2013.

[37] K. Hase and R. Stein, "Analysis of rapid stopping during human walking," *Journal of neurophysiology*, vol. 80, no. 1, pp. 255–261, 1998.

[38] M. Bishop, D. Brunt, N. Pathare, and B. Patel, "The effect of velocity on the strategies used during gait termination," *Gait & posture*, vol. 20, no. 2, pp. 134–139, 2004.

[39] M. Qiao and D. L. Jindrich, "Task-level strategies for human sagittal-plane running maneuvers are consistent with robotic control policies," *PLoS One*, vol. 7, no. 12, p. e51888, 2012.

[40] J. Hewit, J. Cronin, C. Button, and P. Hume, "Understanding deceleration in sport," *Strength & Conditioning Journal*, vol. 33, no. 1, pp. 47–52, 2011.

[41] S. Böttcher, "Principles of robot locomotion," in *Proceedings of human robot interaction seminar*, 2006.

[42] S. Kim and P. Wensing, "Design of dynamic legged robots," *Foundations and Trends in Robotics*, vol. 5, pp. 117–190, 01 2017.

[43] M. F. Silva and J. Tenreiro Machado, "A historical perspective of legged robots," *Journal of Vibration and Control*, vol. 13, no. 9-10, pp. 1447–1486, 2007.

[44] A. Mueller, "Modern robotics: Mechanics, planning, and control [bookshelf]," *IEEE Control Systems Magazine*, vol. 39, no. 6, pp. 100–102, 2019.

[45] J. W. Grizzle, J. Hurst, B. Morris, H.-W. Park, and K. Sreenath, "Mabel, a new robotic bipedal walker and runner," in *2009 American Control Conference*. IEEE, 2009, pp. 2030–2036.

[46] B. Katz, J. Di Carlo, and S. Kim, "Mini cheetah: A platform for pushing the limits of dynamic quadruped control," in *2019 international conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 6295–6301.

[47] S. Seok, A. Wang, D. Otten, and S. Kim, "Actuator design for high force proprioceptive control in fast legged locomotion," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 1970–1975.

[48] Y. Ding, L. Wang, Y. Li, and D. Li, "Model predictive control and its application in agriculture: A review," *Computers and Electronics in Agriculture*, vol. 151, pp. 104–117, 2018.

[49] M. Vukov, S. Gros, G. Horn, G. Frison, K. Geebelen, J. B. Jørgensen, J. Swevers, and M. Diehl, "Real-time nonlinear mpc and mhe for a large-scale mechatronic application," *Control Engineering Practice*, vol. 45, pp. 64–78, 2015.

[50] N. Rathod, A. Bratta, M. Focchi, M. Zanon, O. Villarreal, C. Semini, and A. Bemporad, "Terrain aware model predictive control for legged locomotion," 2021.

[51] M. Gaertner, M. Bjelonic, F. Farshidian, and M. Hutter, "Collision-free mpc for legged robots in static and dynamic scenes," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8266–8272.

[52] Boston Dynamics, "Picking up momentum," https://bostondynamics.com/blog/picking-up-momentum/, 2024, accessed: 2024-08-10.

[53] K. Ishihara, T. D. Itoh, and J. Morimoto, "Full-body optimal control toward versatile and agile behaviors in a humanoid robot," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 119–126, 2019.

[54] J. Eßer, "Introduction to reinforcement learning – a robotics perspective," 2023, accessed: 2024-08-10. [Online]. Available: https://lamarr-institute.org/blog/reinforcement-learning-and-robotics/

[55] Z. Xie, "Reinforcement learning for legged robot locomotion," Ph.D. dissertation, University of British Columbia, 2021.

[56] E. Guizzo, "Researchers build fast running robot inspired by velociraptor run, raptor, run," 2014, [Accessed 01-07-2024]. [Online]. Available: https://spectrum.ieee.org/fast-running-biped-robot-based-on-velociraptor

[57] D. Orf, "A Chinese Humanoid Just Broke the Robot Speed Record—and Now It's a Race Against Time — popularmechanics.com," https://www.popularmechanics.com/technology/robots/a60192610/h1-robot-speed-record/, 2024, [Accessed 01-07-2024].

[58] D. E. Kirk, *Optimal control theory: an introduction*. Courier Corporation, 2004.

[59] K. Hauser, "Optimal control for robotic systems," n.d., [Accessed 14-01-2024]. [Online]. Available: https://motion.cs.illinois.edu/RoboticSystems/OptimalControl.html

[60] W. Xi and C. D. Remy, "Optimal gaits and motions for legged robots," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 3259–3265.

[61] D. Pardo, L. Möller, M. Neunert, A. W. Winkler, and J. Buchli, "Evaluating direct transcription and nonlinear optimization methods for robot motion planning," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 946–953, 2016.

[62] R. Tedrake, "Underactuated robotics: Algorithms for walking, running, swimming, flying, and manipulation," 2022, course Notes for MIT 6.832. [Online]. Available: http://underactuated.mit.edu

[63] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, no. 4, pp. 849–904, 2017.

[64] S. Dafarra, S. Bertrand, R. J. Griffin, G. Metta, D. Pucci, and J. Pratt, "Nonlinear trajectory optimization for large step-ups: Application to the humanoid robot atlas," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 3884–3891.

[65] J. Lee, M. Bjelonic, A. Reske, L. Wellhausen, T. Miki, and M. Hutter, "Learning robust autonomous navigation and locomotion for wheeled-legged robots," *Science Robotics*, vol. 9, no. 89, p. eadi9641, 2024.

[66] S. Shield, "A contact-implicit direct trajectory optimization scheme for the study of legged maneuverability," Ph.D. dissertation, University of Cape Town, 2022, phD Thesis.

[67] C. Fisher, "Trajectory optimisation inspired design for legged robotics," Ph.D. dissertation, University of Cape Town, 2021, phD Thesis.

[68] J. T. Betts, "Survey of numerical methods for trajectory optimization," *Journal of guidance, control, and dynamics*, vol. 21, no. 2, pp. 193–207, 1998.

[69] C. Mastalli, W. Merkt, J. Marti-Saumell, H. Ferrolho, J. Solà, N. Mansard, and S. Vijayakumar, "A feasibility-driven approach to control-limited ddp," *Autonomous Robots*, vol. 46, no. 8, pp. 985–1005, 2022.

[70] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 69–81, 2014.

[71] W. Xi, Y. Yesilevskiy, and C. D. Remy, "Selecting gaits for economical locomotion of legged robots," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1140–1154, 2016.

[72] M. Focchi, V. Barasuol, M. Frigerio, D. G. Caldwell, and C. Semini, "Slip detection and recovery for quadruped robots," *Robotics Research: Volume 2*, pp. 185–199, 2018.

[73] P. Zhao, J. Liu, Y. Li, and C. Wu, "A spring-damping contact force model considering normal friction for impact analysis," *Nonlinear Dynamics*, vol. 105, pp. 1437–1457, 2021.

[74] H. Lee and V. I. Utkin, "Chattering suppression methods in sliding mode control systems," *Annual reviews in control*, vol. 31, no. 2, pp. 179–188, 2007.

[75] V. Bist, "Field oriented control (foc) made easy for brushless dc (bldc) motors using ti smart gate drivers," *Application Brief*, 2018.

[76] M. L. Bynum, G. A. Hackebeil, W. E. Hart, C. D. Laird, B. L. Nicholson, J. D. Siirola, J.-P. Watson, D. L. Woodruff *et al.*, *Pyomo-optimization modeling in python*. Springer, 2021, vol. 67.

[77] Y. Kawajir, "Ipopt: Documentation — coin-or.github.io," https://coin-or.github.io/Ipopt/index.html#Overview, [Accessed 21-07-2024].

[78] A. Knemeyer, S. Shield, and A. Patel, "Minor change, major gains: the effect of orientation formulation on solving time for multi-body trajectory optimization," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5331–5338, 2020.

[79] Tmotor, "AK10-9_AK Series Dynamical Modular_Robot Dynamics_T-MOTOR Official Store - UAV Power System, Robot Power System, Model Power System — store.tmotor.com," https://store.tmotor.com/goods-1030-AK10-9.html, [Accessed 06-07-2024].

[80] Z. Manchester, N. Doshi, R. J. Wood, and S. Kuindersma, "Contact-implicit trajectory optimization using variational integrators," *The International Journal of Robotics Research*, vol. 38, no. 12-13, pp. 1463–1476, 2019.

[81] S. Dafarra, G. Romualdi, and D. Pucci, "Dynamic complementarity conditions and whole-body trajectory optimization for humanoid robot locomotion," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3414–3433, 2022.

[82] W. Xi, Y. Yesilevskiy, and C. D. Remy, "Selecting gaits for economical locomotion of legged robots," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1140–1154, 2016.

[83] R. Components, "Alphawire customer specification part no. 391640," https://za.rs-online.com/web/p/hook-up-wire/2850252?gb=s; https://docs.rs-online.com/0584/A700000011148306.pdf, 2024.

[84] ——, "Alphawire customer specification part no. 391040," https://za.rs-online.com/web/p/hook-up-wire/2850142?gb=s; https://docs.rs-online.com/931f/A700000011148280.pdf, 2024.

[85] Mathworks, "Setting Up Solvers for Physical Models - MATLAB & Simulink — mathworks.com," https://www.mathworks.com/help/simscape/ug/setting-up-solvers-for-physical-models.html, [Accessed 13-07-2024].

[86] ——, "Important Concepts and Choices in Physical Simulation - MATLAB & Simulink — mathworks.com," https://www.mathworks.com/help/simscape/ug/important-concepts-and-choices-in-physical-simulation.html, [Accessed 13-07-2024].

[87] ——, "Making Optimal Solver Choices for Physical Simulation - MATLAB & Simulink — mathworks.com," https://www.mathworks.com/help/simscape/ug/making-optimal-solver-choices-for-physical-simulation.html, [Accessed 13-07-2024].

[88] ——, "Model spatial contact between two geometries - MATLAB — mathworks.com," https://www.mathworks.com/help/sm/ref/spatialcontactforce.html, [Accessed 14-07-2024].

[89] S. Debaere, C. Delecluse, D. Aerenhouts, F. Hagman, and I. Jonkers, "From block clearance to sprint running: Characteristics underlying an effective transition," *Journal of sports sciences*, vol. 31, no. 2, pp. 137–149, 2013.

[90] A. Maiorino and G. G. Muscolo, "Biped robots with compliant joints for walking and running performance growing," *Frontiers in Mechanical Engineering*, vol. 6, p. 11, 2020.