Bio-Inspired Worm Robot: Dynamics, Control and Digital Twin

by

Jay B Menon

A Thesis
Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

Approved April 2025 by the
Graduate Supervisory Committee

Sangram Redkar, Co-Chair
Mehran Rehmani, Co-Chair
Thomas Sugar

ARIZONA STATE UNIVERSITY
May 2025

ABSTRACT

This research introduces the design, modeling, and experimental validation of a worm-inspired robot that emulates the metachronal gait found in nature through a digital twin framework. Leveraging principles from bio-inspired robotics, the robot features five degrees of freedom, enabling adaptive locomotion across varied environments. Kinematic and dynamic models are developed utilizing Geometric (Clifford) Algebra, complemented by physics-based simulations carried out in Nvidia Isaacsim and MATLAB. The digital twin facilitates continuous data logging from embedded sensors, which enhances performance optimization and supports adaptive control strategies. Furthermore, the study explores the scope of a range sensor for real-time environmental feedback, improving the robot's adaptability. This work emphasizes the potential of bio-inspired robotic systems to tackle locomotion challenges in complex terrains, with significant implications for applications in search and rescue, industrial inspection, and beyond.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my parents and my grandmother, Dr. Sangram Redkar, Dr. Mehran Rahmani, Dr. Thomas Sugar, Apoorva Rahul Uplap, and TSMC Arizona Corporation for their unwavering motivation and guidance throughout my research. I would also like to extend my thanks to Aakash, Kevin, Saanvi, Srishti, Akshay, and all my friends who supported me in achieving this milestone in my thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

Motivation

Designs found in nature have consistently influenced robotics research, especially in emulating locomotion strategies inspired by creatures like snakes and their ability to adapt to dynamically changing environments. We refer to the multidisciplinary field that combines biology, engineering, robotics, and materials science to tackle challenges in complex and unpredictable environments as bio-inspired robotics. This approach promotes collaboration among interdisciplinary teams of biologists and roboticists who investigate and formulate hypotheses about how specific organisms address complex engineering challenges, especially in the realm of robot locomotion. Furthermore, a nuanced distinction exists between bio-inspired robotics and robotics-inspired biology as shown in Figure 1.1. Bio-inspired robotics pertains to the application of mechanisms derived from biological organisms to create robots that emulate their forms and functions. In contrast, robotics-inspired biology leverages robotics and mechanical models to examine biological principles, enabling scientists to test hypotheses related to animal locomotion, biomechanics, and control systems (Gravish and Lauder, 2018).

Robotic design increasingly draws from biological organisms, utilizing nature as a blueprint to enhance traditional engineering methodologies and improve control mechanisms. This biomimetic approach is influenced by a diverse array of biological systems, encompassing both flora and fauna, which inform the locomotion and operational strategies of robots. By observing and replicating the walking, swim-

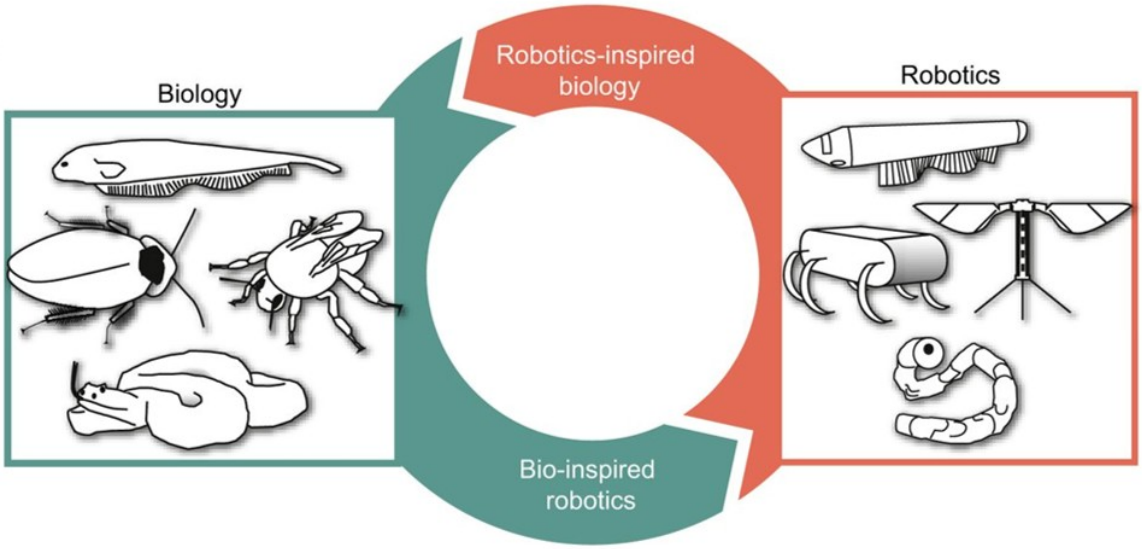


Figure 1.1: Bio-inspired Robotics and Robotics-inspired Biology (Gravish and Lauder, 2018)

ming, crawling, or flying behaviors of these organisms, engineers can develop robots that exhibit greater efficiency, adaptability, and functionality in various environments. There are various types of bio-inspired robots, such as legged mobile robots (bipeds, quadrupeds, or multi-legged robots), wheeled robots, and flying robots (Unmanned Aerial Vehicles (UAVs)). Some are also inspired by monkeys, fish, and snakes, or by feedback control mechanisms found in plants (Fukuda *et al.*, 2018; Pettersen, 2017). Some of the bio-inspired robots are shown in Figure 1.2.

For example, Rhex is a six-legged robot designed with inspiration drawn from the locomotion of cockroaches while in contrast, the use of robotic fish models to investigate the schooling behavior and maneuverability of real fish serves as a prime example of robotics-inspired biology. Additionally, researchers have investigated the capabilities of octopus arms using emulated models, exploring the functional morphology of internal tissues and focusing on the sinusoidal arrangement of the nerve cord and the specific insertion points of muscle fibers. (Gravish and Lauder, 2018;

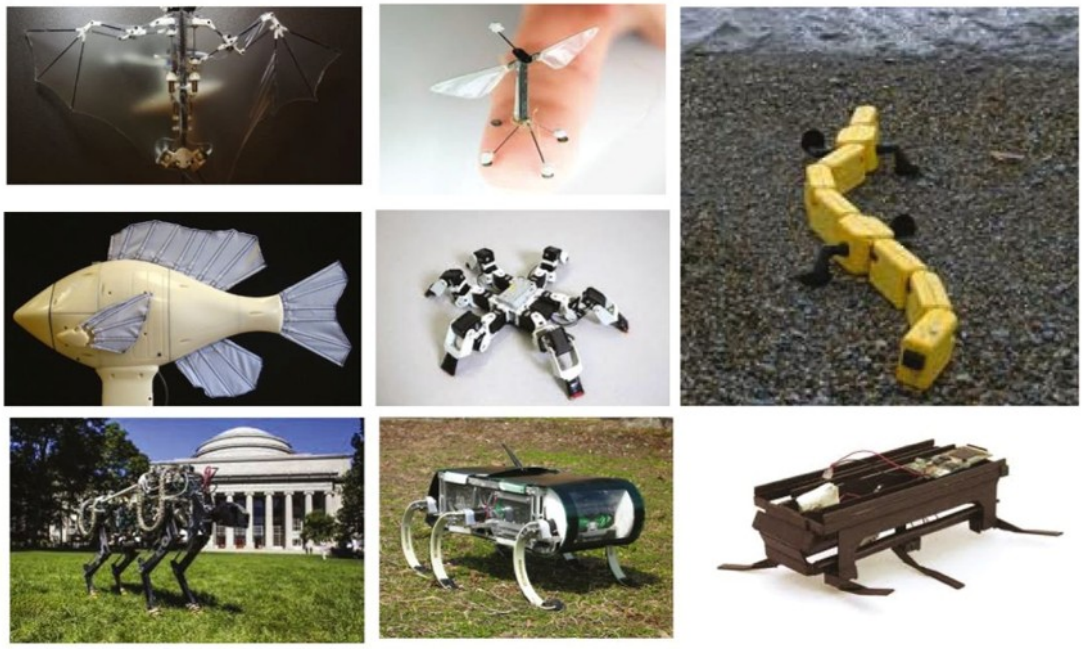


Figure 1.2: Types of Bio-inspired Robots That Fly, Run, Swim and Crawl (Gravish and Lauder, 2018)

Margheri *et al.*, 2012).

Multi-Modal Locomotion in Nature

The study of multi-modal locomotion in various animal species has historically served as a significant source of inspiration for the development of robotic systems, particularly in the context of navigating complex and unstructured environments, such as those found in disaster-affected areas and extraterrestrial terrains. Numerous biological organisms have evolved the ability to transition seamlessly between aerial, terrestrial, and aquatic modes of locomotion, thereby optimizing their movements for energy efficiency and adapting to diverse environmental challenges. Additionally, there are some animals that can thrive in two different environments simultaneously (Lock *et al.*, 2013).

Rather than relying solely on conventional design methodologies, the application of bio-inspired robotics leverages insights gained from these evolved species that have demonstrated proficiency in traversing challenging landscapes. Bio-inspired robots can also navigate such environments without relying on GPS or traditional navigation methods (Bogue, 2019). AntBot is a bio-inspired navigation robot application based on desert ants, which navigate extreme desert conditions using a celestial compass. This involves utilizing polarized light in the sky to determine direction, as well as optic flow, which counts the steps taken and measures ground movement through visual input to assess distance. This approach has the potential to reduce the reliance on GPS for outdoor robotic navigation (Bogue, 2019).

Other bio-inspired robots include the AquaMAV (Aquatic Micro Air Vehicle), inspired by plunge-diving birds, features reconfigurable wings and a CO2-powered water jet enabling it to escape from water. The DALER (Deployable Air-Land Exploration Robot), inspired by the vampire bat, possesses moveable tips on its wings, allowing it to crawl along the ground. Similarly, the Guillemot-Inspired VTOL (Vertical Takeoff-and-Landing) Robot emulates a seabird's ability to fly, dive, and swim, reflecting a life cycle from nesting to foraging. Additionally, numerous aquatic robots draw inspiration from marine life. MIT's RoboTuna focuses on fish population studies and vortex control, while the CIA's Robotics Catfish, affectionately named "Charlie," serves as an underwater intelligence-gathering device. Other notable examples include the iSplash-II robotic fish and SoFi, which employs a fluidic elastomer actuator to navigate coral reefs. Jollbot and Glumper are bio-inspired jumping robots that use a glide and landing mechanism to traverse rough terrain (Boyer *et al.*, 2012). Lastly, dragonfly-inspired micro air vehicles exhibit remarkable maneuverability, exemplified by the CIA's Insectothopter designed for espionage (Bogue, 2019). These capabilities position bio-inspired robots as transformative tools for disaster management, signifi-

cantly enhancing mobility, functionality, and safety in high-risk scenarios (Balla *et al.*, 2024) (Hunt, 2010).

Furthermore, the development of computer simulations for proposed prototypes, such as insect-inspired walking mechanisms, facilitates the visualization of all potential modes of action, ultimately reducing costs associated with physical prototyping (Taubes, 2000). We will discuss how real-time simulations for robot locomotion have proven to be beneficial for robotics ahead.

Snake and Worm-Inspired Robotics

Snakelike (Chirikjian, 2020) robots have been a pivotal area in robotics focusing on their diverse applications in search and rescue operations, firefighting, and maintenance tasks in challenging environments where conventional robots often encounter limitations. These biological organisms exhibit adaptive gaits that enable them to navigate challenging terrains, such as rough ground, narrow passages, low-friction surfaces, and water. Recent studies have focused on designing and developing robotic systems that emulate the unique locomotion and movement strategies found in nature. Researchers are particularly inspired by the physiology and kinematic mechanisms of snakes, inchworms, and caterpillars in creating snake robots. Snakes traverse using advanced movement techniques, such as lateral undulation, sidewinding, concertina, and rectilinear locomotion. This allows them to navigate a variety of environments with remarkable efficiency showing a remarkable degree of flexibility while maintaining stability, thereby presenting an ideal framework for modular robotic designs. Furthermore, the unique scalation pattern of snake skin confers a directional frictional advantage, facilitating forward locomotion with minimal energy expenditure (Transeth *et al.*, 2009). In contrast, inchworms and caterpillars exhibit peristaltic and crawling gaits that inform advancements in soft robotics and compli-

ant motion control, particularly in constrained and irregular environments. This is vital not only in disaster response scenarios but also in the inspection of pipelines and other infrastructures, underscoring the significant potential of biomimetic robotic systems in enhancing operational efficiency and safety (Ab Rashid *et al.*, 2020).

The study of snake locomotion has significantly influenced the development of snake robots, beginning with early qualitative research by Gray (1946). The first functional biologically inspired serpentine robot, the Active Cord Mechanism (ACM III) (Hirose and Yamada, 2009). This 2-meter-long robot had 20 revolute joints with 1 degree of freedom (DOF) each and utilized passive casters for movement, mimicking serpentine undulation through alternating lateral joint motions. Since then, various multi-link articulated robots, such as snake robots, hyper-redundant robots, and G-snakes, have been developed. These robots may incorporate passive wheels to enhance traction and reduce friction or function without wheels, relying on frictional anisotropy akin to real snake scales (Transeth *et al.*, 2009). Recent advancements in snake-inspired robotics over the past 10 to 15 years have been driven by the need for flexible, modular, and terrain-adaptive robotic systems. The complex kinematic and dynamic modeling of these robots, due to their high degrees of freedom (DOFs), necessitates the creation of mathematical models and control algorithms to optimize locomotion.

Snake robots' unique locomotion abilities have led to the development of various mathematical models to study their movement dynamics (Liljebäck *et al.*, 2010). These robots showcase a range of locomotion modalities, including hopping (Fukuoka and Kimura, 2009), wheeling (Shiotan *et al.*, 2006), brachiating (Davies *et al.*, 2018), slithering (Seeja *et al.*, 2022), walking (Playter *et al.*, 2006), swimming (Dudek *et al.*, 2005), and metachronal locomotion (Ghanbari *et al.*, 2008). Their slender bodies allow them to maneuver in confined spaces, making them particularly well-suited for

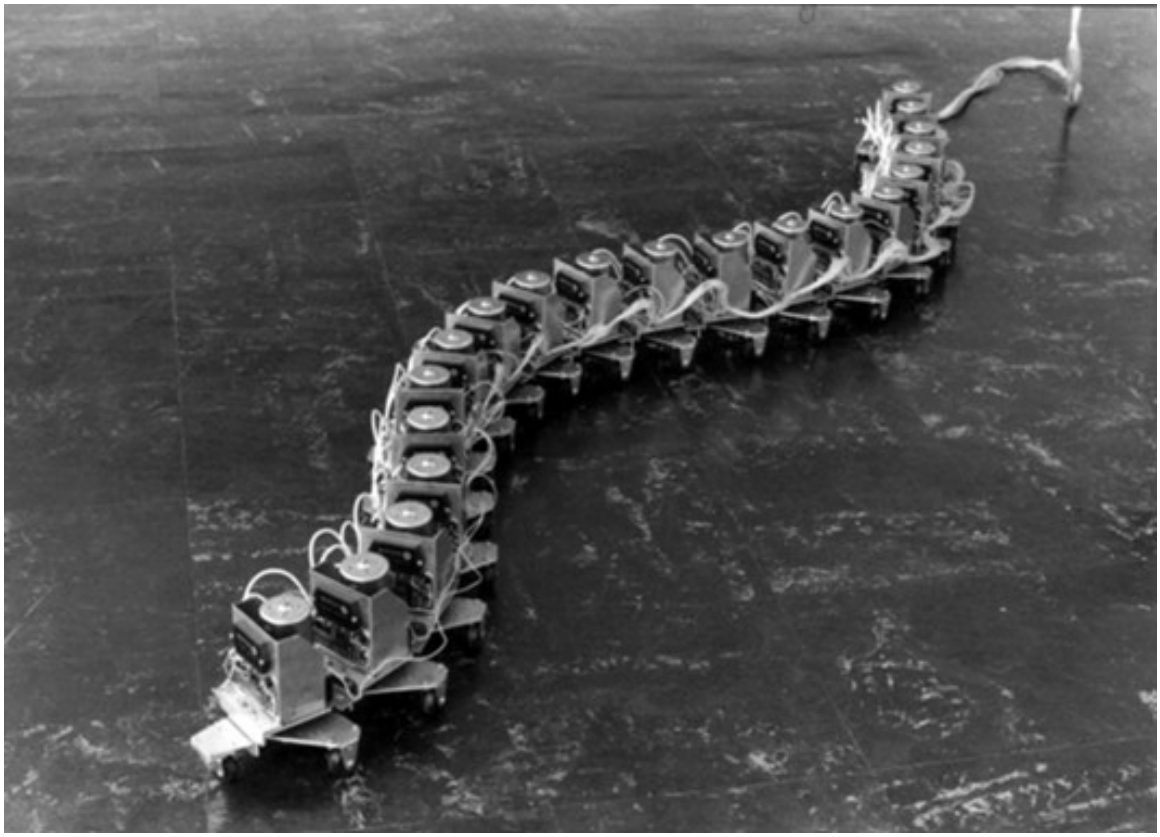


Figure 1.3: Active Cord Mechanism (ACM) III (Hirose and Yamada, 2009)

applications such as pipe inspection, search and rescue, and medical missions (Bi *et al.*, 2023; Mehringer *et al.*, 2017; Choset *et al.*, 2000). These robots can crawl, slither, and swim, allowing them to effectively navigate complex environments and overcome various obstacles. Common traversal techniques include lateral undulation, sidewinding, concertina, and rectilinear locomotion.

Traditional snake robots, such as Hirose's ACM-III and later models like the ACM-R4, have demonstrated effective locomotion using both passive and active wheel mechanisms (Yamada and Hirose, 2006). However, these designs are unsuitable for tightly confined spaces, like those encountered during pipe inspections. To address this challenge, Fjerdingen *et al.* (2009) introduced PIKo, an articulated, snake-inspired pipe inspection robot equipped with active wheels and two degrees of freedom in its joints.

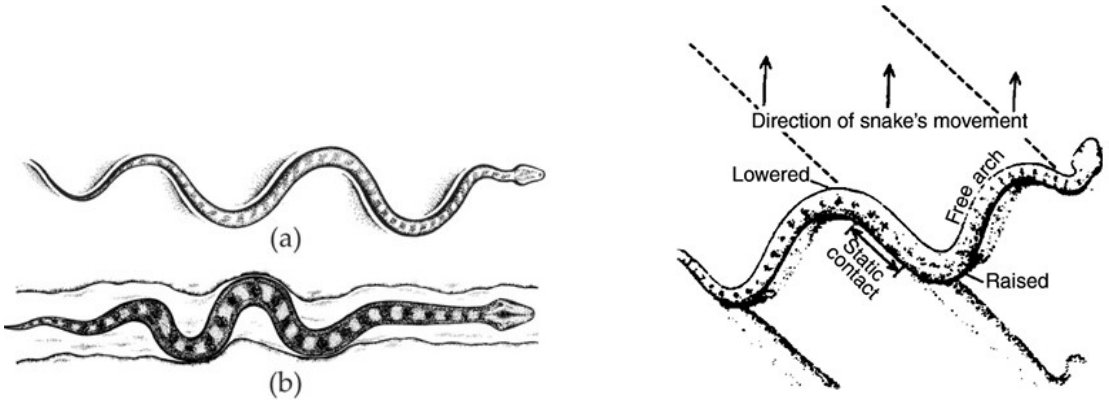
This modular robotic system enhances adaptability to different terrains, enabling it to navigate sharp turns, vertical climbs, and constricted pipe segments. The design allows for both horizontal and vertical locomotion; the robot climbs vertically by using an opposing-push mechanism that relies on friction between the wheels and the inner walls of the pipe. Additionally, Wakimoto *et al.* (2003) developed a micro snake-like robot specifically for inspecting small pipes. The snaking drive locomotion has proven to be an effective method for pipe inspection robots, especially in environments where pipe diameters vary significantly.

Wakimoto *et al.* (2003) developed robot for pipe inspections utilizes lateral undulation locomotion, which is the fastest and most common form of movement in snakes. In this mode of operation, the robot's joints are driven by continuous sine wave signals traveling from the head to the tail. Forward motion is achieved through the friction between the robot and the inner walls of the pipe. Another innovative work was the introduction of ACM-S1, a waterproof and dustproof snake robot specifically designed for applications such as pipe inspection, search and rescue, and underwater exploration. Unlike previous snake robots that utilized passive wheels, the ACM-S1 features a 3-DOF bending and expanding joint mechanism as shown in Figure 1.6. This allows it to perform inchworm-like crawling over uneven surfaces and angleworm-like retraction for smooth ground movement. By eliminating the need for ratcheted wheels, the system relies instead on elastic rods and slide screw actuation, proving to be highly adaptable to complex environments (Sugita *et al.*, 2008). Another application is the Small Pipe Inspector, an autonomous and tether-free robotic system, is engineered for the inspection of long and complex pipelines with small diameters, including applications in oilfield coiled tubing, refinery heat exchangers, and furnace tubes (Jamoussi, 2005).

Figure 1.4: ACM-S1 (Wakimoto *et al.*, 2003)

Locomotion Mechanisms in Snake Robots

Lateral undulation, also known as serpentine crawling, involves a continuous wave-like motion that propagates from the front to the rear of the snake's body, allowing it to push against the contours of the terrain as shown in Figure 1.5a. This method is particularly efficient in rough terrains where there are push points, such as rocks and bumps, that provide necessary traction. To maintain motion, at least three push points are required. Sidewinding locomotion occurs when the snake lifts parts of its body off the ground, creating a rolling movement at an angle as shown in Figure 1.5b. This technique is effective for low-friction surfaces like sand and mud. Concertina locomotion, or accordion motion, involves the snake stretching and folding



(a) Lateral Undulation and Concertina Locomotion

(b) Sidewinding

Figure 1.5: Different Locomotion Mechanisms in Snake (Transeth *et al.*, 2009)

its body alternately to propel itself forward. This method is particularly effective in tight spaces, such as pipes, tunnels, and branches, but requires high friction to push against the substrate. Other snake gaits include sinus-lifting, a modification of lateral undulation where sections of the trunk lift off the ground for optimized propulsion; skidding or slide-pushing, which is an energy-inefficient zigzag movement used on low-friction surfaces; swimming, where the snake undulates its body similar to eels for aquatic locomotion; and tree climbing, wherein thin-bodied snakes use vertical lateral undulation to ascend by wrapping around tree branches (Transeth *et al.*, 2009).

The flexible and compliant locomotion typified by inchworms involves techniques of anchoring, extending, and contracting their bodies. Such adaptive deformation facilitates precise directional adjustments in three-dimensional space and promotes efficient movement in confined environments. This bio-inspired methodology holds significant potential for soft robotics, allowing for smooth navigation across complex terrains (Zhang *et al.*, 2019). MagWorm is a biologically-inspired, magnetically actuated worm-like soft robot that employs permanent magnets for both actuation and

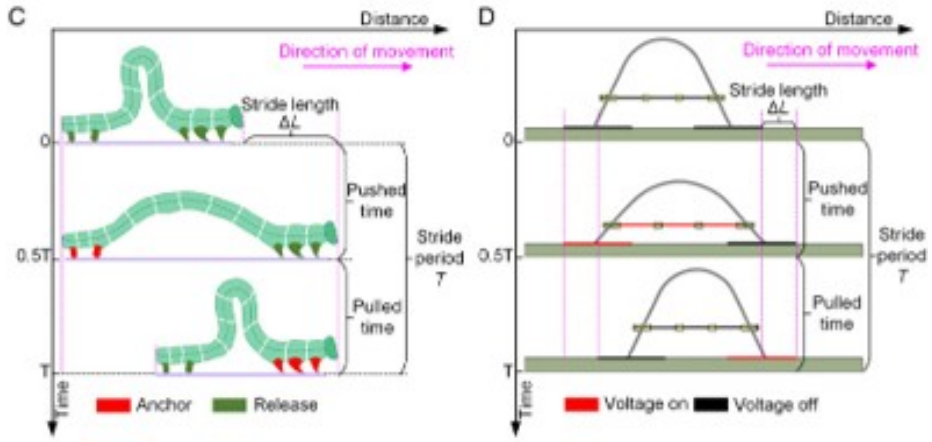


Figure 1.6: Inchworm Locomotion (Hu *et al.*, 2023)

driving mechanisms and is engineered for rapid crawling locomotion, mimicking the movements of real worms while also enabling streamlined fabrication and adaptive mobility (Niu *et al.*, 2021).

Inchworm locomotion is characterized by a methodical movement where the anterior limbs grasp the ground, followed by a contraction of the body as the posterior limbs advance, culminating in the release of the front legs. This controlled, stepwise locomotion contrasts with caterpillar motion, which employs a vertical traveling wave to facilitate movement from posterior to anterior, leveraging small legs for traction. The latter approach is recognized for its stability, slow pace, and energy efficiency (Transeth *et al.*, 2009). A significant feature of these robotic systems is their biomimetic inspiration; for example, worms enact a metachronal gait, where limb movements are synchronized, effectively creating a wave-like motion (Ghanbari *et al.*, 2008).

Mathematical modeling is vital in comprehending and enhancing snake robot locomotion. Prevailing models have concentrated on planar movement through Newton–Euler equations or Lagrangian approaches. Yet, the complexity of frictional interactions, stick-slip transitions, and contact forces with the environment presents

a daunting challenge for fully articulating the 3D dynamics of snake robots (Transeth *et al.*, 2006)(Lipták *et al.*, 2016). The study by Noorani *et al.* (2015) focuses on the development of a crawling robot inspired by caterpillars. Its locomotion mechanism relies on a vertical trapezoidal body wave in the vertical plane, which enables efficient forward movement. The robot's dynamics are represented by a planar five-link chain model, with motion equations derived for each segment with experimental validation. Some studies have also introduced non-smooth 3D modeling techniques to overcome frictional limitations (Transeth *et al.*, 2006).

Digital Twins in Robotics

A Digital Twin (DT) is a virtual model of a physical or conceptual entity that continuously updates to reflect real-world changes, enabling simulation, analysis, and predictive decision-making. This entity is primarily a physical object, although conceptual objects can also be digitally represented. The term "contextualized" implies that the software model can accurately replicate the behavior of the physical object, allowing for comprehensive analysis, study, and prediction of its actions within the specific environmental constraints it operates in (Crespi *et al.*, 2023).

The origins of Digital Twin (DT) technology can be traced back to NASA's Apollo 13 mission in 1970, when a physical twin system was utilized to simulate and address challenges faced by the spacecraft in real time. The formal concept of Digital Twins was first articulated by Dr. Michael Grieves in 2002 at a Society of Manufacturing Engineers (SME) conference at the University of Michigan. Initially referred to as the "Mirrored Spaces Model" (MSM), it was later renamed the Information Mirroring Model. In 2012, John Vickers from NASA officially coined the term "Digital Twins" (DTs). This innovative approach laid the groundwork for the modern understanding of digital twins (Rajamurugu and Karthik, 2022).

Initially the concept of DTs revolved significantly around Product Lifecycle Management (PLM) to encompass real time interconnected cyber physical systems and it has come a long way from there since. They leverage Internet of Things (IoT), Artificial Intelligence (AI) and communication technologies such as 5G and 6G to enable this seamless data exchange within (Ngadi *et al.*, 2023). It was introduced to enhance the manufacturing process simulations and from there DTs have eventually transitioned to cloud based distributed systems that enables cross domain integration including smart cities, healthcare and energy management.

Digital twins can be categorized into several types. The product twin represents a product's entire life cycle, providing real-time data from design to full functionality. Next comes the data twin, exemplified by Google Maps, which offers a digital representation of the Earth's surface and integrates real-time traffic data to enhance navigation. Systems twins model the interactions between physical and digital processes, encompassing areas like manufacturing, supply chain management, and customer experiences. Lastly, infrastructure twins depict physical structures such as highways, buildings, or stadiums, facilitating their monitoring and management. Similarly, Grieves also formalized a structured Digital Twin framework in his work for virtual factory models. His work introduced three fundamental DT components- Digital Twin Prototype (DTP), Digital Twin Instance (DTI) & Digital Twin Aggregate (DTA). DTP is a virtual model used during the design and development phase of a product. DTI is a real-time digital counterpart of a physical object, continuously updated with live data. Finally, a DTA is A system that integrates multiple DTIs, enabling large-scale simulations and advanced decision-making, commonly referred to as Digital Twin Computing (DTC) (Crespi *et al.*, 2023; TechnoHacks, 2024; McKinsey&Company, 2024).

Digital twin technology plays a crucial role in enhancing the performance and

reliability of manufactured products by allowing for the prediction of their future behavior and potential failures. This capability facilitates timely preventive measures to mitigate such issues. Furthermore, it contributes to the continuous improvement and development of the product. Tesla, Inc. exemplifies the application of digital twin technology, as every vehicle they sell is equipped with its own digital twin. Daily data collected from the car's sensors is analyzed to derive valuable insights, which are subsequently utilized to prevent errors and optimize performance. These insights inform the preparation of software updates for their vehicles, ensuring that users receive enhancements based on real-time data analysis (Rajamurugu and Karthik, 2022). Digital twins utilize simulation as a fundamental technology for forecasting future performance and determining optimal solutions (Biller *et al.*, 2022). Unlike traditional modeling and simulation techniques, digital twins are characterized by their real-time synchronization with their physical equivalents and a constant data stream from sensors (Taylor *et al.*, 2021; Kenett and Bortman, 2022). The application of digital twins is expanding across diverse sectors, including smart farming, manufacturing, and urban planning (Dineva and Atanasova, 2022).

There have been various applications of digital twins across the manufacturing sector such as Flex-Cell Digital Twin and Gunnerus Digital Twin. The Flex-Cell Digital Twin is a modular robotic system designed for precision assembly, featuring Kuka and UR5e robotic arms with OnRobot grippers, with synchronized execution and trajectory visualization. It incorporates discrete positioning commands, deviation checking for alignment, and virtual commissioning for pre-implementation simulation. While the Gunnerus Digital Twin showcases the use of maritime digital twins, focusing on ship maneuverability, anomaly detection, and predictive maintenance. It serves as a test platform for co-simulation frameworks that enhance data-driven decision-making and scenario analysis in dynamic marine environments. It operates as a "Digital

Shadow", continuously receiving live data from the physical vessel without controlling it. The system integrates various sensors, including GPS, motion reference units, wind sensors, and engine data, to improve tracking, navigation, and maintenance (Kulik *et al.*, 2024).

A wide array of software tools and technologies has emerged to facilitate the integration of Digital Twins (DTs) across various sectors, enhancing activities such as modeling, real-time monitoring, data management, and simulation. Modeling tools like ANSYS Twin Builder, Siemens NX, SolidWorks, AutoCAD, and CATIA are vital for creating geometric and structural representations. In addition, advanced simulation capabilities are offered by tools such as Nvidia Isaac Sim, MuJoCo, MATLAB, Simulink, and MARC. In industrial contexts, cloud-based service platforms such as Siemens MindSphere, ThingWorx, and Azure IoT Hub enhance IoT integration, predictive maintenance, and performance analysis. For behavioral modeling and control, CoDeSys and MWorks play a critical role in real-time motion management, while AI-driven platforms like ThingWorx and PTC's AI Edge Computing augment predictive analytics. Connectivity solutions like Predix, Cisco Jasper, and Microsoft Azure Digital Twin further bolster industrial IoT applications by merging sensor data with cloud services. Collectively, these technologies significantly enhance the lifecycle of Digital Twins, broadening their applications in sectors like manufacturing, aerospace, healthcare, and smart cities, thereby promoting the advancement of intelligent and interconnected systems (Qi *et al.*, 2021).

NVIDIA Omniverse Isaac Sim is a sophisticated robotics simulation toolkit that facilitates the creation of digital twins and virtual environments for robotic applications. It provides essential features for physically accurate simulations, synthetic dataset generation, sensor data simulation, and domain randomization. Built on the Omniverse Kit SDK and utilizing the USD file format for scene representation,

Nvidia Isaac Sim leverages Omniverse Nucleus for content accessibility. This toolkit is particularly beneficial for warehouse automation, enhancing the integration and management of autonomous robotic systems to improve operational efficiency. It supports both C++ and Python via compiled plugins and bindings, accommodating diverse workflows—ranging from standalone Omniverse applications to Visual Studio Code extensions and Jupyter Notebooks. Additionally, compatibility with ROS and ROS2 enables hardware-in-the-loop capabilities for seamless sim-to-real transfer (NVIDIA Corporation, 2025).

Relevance to Snake and Worm Robots

Digital Twin (DT) technology presents substantial opportunities for advancing the design, control, and deployment of snake and worm-inspired robots. These robots typically exhibit a high degree of freedom (DOFs) due to their segmented and modular architectures, resulting in complex kinematic and dynamic behaviors that are particularly sensitive to environmental interactions. Through real-time synchronization with physical snake and worm robots, Digital Twins facilitate continuous performance monitoring, predictive maintenance, and adaptive control adjustments. For instance, as a snake robot navigates a confined pipe or uneven terrain, its digital counterpart can simulate the forces acting on each segment, predict potential failure points, and propose optimized gait patterns or control parameters to enhance movement efficiency.

Moreover, Digital Twins enable pre-deployment testing of locomotion strategies under virtual conditions that closely approximate real-world environments. This capability is particularly advantageous for snake and worm robots intended for deployment in situations that are challenging, hazardous, or impossible to physically replicate in laboratory settings—such as deep-sea exploration, nuclear facility inspections,

or disaster response scenarios.

Additionally, Digital Twin platforms like Nvidia Isaac Sim facilitate the generation of synthetic datasets that are essential for training machine learning algorithms focused on perception, localization, and adaptive control within complex environments. For snake and worm robots, this advancement enhances their real-time decision-making capabilities when they encounter unforeseen obstacles or transition between locomotion modes (e.g., from slithering to climbing).

In conclusion, by effectively bridging the chasm between virtual modeling and physical reality, Digital Twins contribute to the development of safer, more efficient, and highly adaptable robotic systems capable of addressing complex challenges across diverse operational domains.

Scope and Overview

The aim of this thesis is presented as follows:

1. Development of a worm robot with five degrees of freedom that simulates the metachronal gait patterns observed in nature, offering potential applications across various fields.
2. Utilizing Geometric Algebra to model the kinematics and dynamics of this robot.
3. Implementing Nvidia Isaac Sim and MATLAB to experimentally validate the system with accurate physics modeling.
4. Creating a digital twin of a gait-based inchworm robot that includes a physical robot working in conjunction with its simulation running in real-time while logging data from the sensors.
5. Testing the physical worm robot on various surfaces and comparing its performance with results from Nvidia Isaac Sim.
6. Future planning to incorporate a sidewinding mechanism for the robot and stream-

ing range sensor data into Isaac Sim

Chapter 2 focuses on the mechanical and system-level design of the worm robot. The chapter details the design criteria, structural layout, actuation components, and sensor integration.

Chapter 3 presents the theoretical underpinnings of the worm robot's kinematics. It discusses various approaches—from rotor algebra to screw theory—for describing forward and inverse kinematics. These methods aim to replicate natural metachronal gaits forming the basis for the gait patterns tested in later chapters.

Chapter 4 delves into the worm robot's dynamic modeling, including nonlinear equations of motion and advanced formulations using Clifford (geometric) algebra. The chapter also addresses trajectory generation and the computations involved in both forward and inverse dynamics. These analyses are central to predicting system behavior and formulating robust control strategies.

Chapter 5 examines multiple control strategies for the worm robot, including PD, PID, fractional PID, and sliding-mode control. The chapter explains why these methods are suitable for highly nonlinear, high-DOF systems, and discusses how each approach can be tuned and evaluated for optimal performance.

Chapter 6 describes the experimental validation and presents results from both physical and simulated tests using Nvidia Isaac Sim and MATLAB.

Chapter 7 summarizes the primary contributions of the research while proposing avenues for improving the robot's performance, and discusses future directions such as integrating more advanced sensing or control strategies.

Chapter 2

DESIGN

Design Components

To develop the digital twin, we must first observe the components in the actual worm robot. The robot has five LX-16A servos attached from Hiwonder. The Hiwonder LX-16A servo motor is a compact, high-performance actuator designed for precision and durability in robotic systems. Weighing just 54 grams and measuring 45.22 × 24.72 × 36.3 mm, it is ideal for space-constrained applications. The servo features a rotation range of 0–240° in servo mode and offers continuous rotation in gear mode, achieving a speed of 0.19 seconds per 60° at 7.4 V with torque outputs of 17 kg·cm at 6 V and 19.5 kg·cm at 7.4 V. Operating within a voltage range of 6 to 8.4 V, it has a stall current of 2.4 to 3 A and a no-load current of about 100 mA. A UART serial communication interface enables half-duplex asynchronous transmission, allowing for daisy-chaining multiple servos, each with a unique ID from 0 to 253 (default ID = 1). The motor ensures an angular positioning accuracy of 0.3° and provides real-time feedback on temperature, input voltage, and operational status, supported by an integrated LED for diagnostics.

Complementing the servo is the Hiwonder TTL/USB Debugging Board, which converts USB signals to TTL-level serial communication, facilitating control and parameter adjustments of connected servos. It operates with a half-duplex UART interface at a specified baud rate, enabling multiple servos to be controlled individually through unique IDs. To control the LX-16A servos using Python, the PyLX-16A library is utilized, designed for compatibility with the Arduino framework.

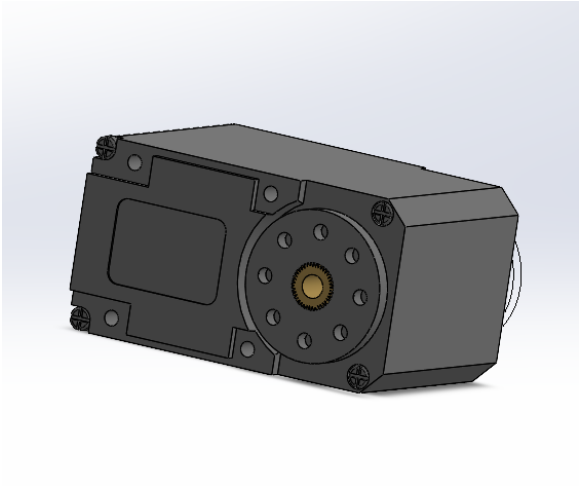


Figure 2.1: Hiwonder LX-16A Servo Motor

Finally for mounting, four brackets are provided: a side bracket, bottom bracket, inclined U-shaped bracket, and straight U-shaped bracket. The side and bottom brackets attach directly to the servo actuator, while the U-shaped brackets are mounted on top, connecting to the motor's rotary sections, offering versatile configuration options.

Each intermediate link is made up of identical components. We constructed these links using a three-part system: a bottom bracket, a servo actuator, and a straight U-shaped bracket. The head is slightly different, utilizing an inclined U-shaped bracket instead of a straight one. The tail is distinct from the intermediate links; it does not have a bottom bracket to reduce friction, which allows the tail to initiate the worm's motion more effectively. For the prototype of the worm robot, we used five links: the tail, three intermediate links, and the head link, as illustrated in Figure 2.2.

The design was replicated in the Nvidia Isaac Sim environment, which is used for digital twin development, as shown in the Figure 2.3. In this study, a six-link robot was designed using SolidWorks and then exported in URDF format to facili-

Figure 2.2: N-link Worm Robot

Figure 2.3: 5 DOF Worm Robot Design in Nvidia Isaac Sim

tate the development of a digital twin with Nvidia Isaac Sim. The URDF Importer tool (NVIDIA Corporation, 2024) was utilized to integrate the robot model into the simulation environment. Each joint of the robot was configured as a revolute joint, with adjustable stiffness and damping parameters. These parameters were specifically tailored to replicate the dynamic behavior of Hiwonder RC servos, which are also utilized in the robot's physical implementation. This approach ensured that the simulation accurately reflected the performance of the actual hardware, providing a reliable platform for design validation and optimization.

Nvidia Isaac Sim significantly enhances mechanical design and development by allowing for thorough refinement of a robot's structure before physical fabrication. Its simulation environment supports iterative testing of joint configurations, optimizing link lengths and weight distribution without the cost of physical prototypes. The URDF-based import process ensures that kinematic and dynamic properties from CAD software are maintained, facilitating accurate validation of joint performance and potential interferences. Furthermore, collision detection and physics-based con-

straints enable engineers to analyze contact forces and maintain structural integrity during motion. By leveraging these features, the digital twin approach in Isaac Sim effectively minimizes design flaws, reduces fabrication iterations, and ensures that the final physical prototype performs as expected.

Chapter 3

KINEMATICS

The kinematics of a robotic system refers to the motion of its links without considering the forces or torques involved. In this chapter, we will explore the forward and inverse kinematics of a caterpillar-inspired robot with 5 degrees of freedom (5-DOF) (Medrano-Hermosillo *et al.*, 2022). The robot is modeled as a serial chain manipulator that moves using sequential gaits. We begin by introducing a method for forward kinematics using rotor algebra, a branch of geometric algebra. Then we present derivations using motor algebra and screw theory and finally we derive a solution for inverse kinematics through a gait-based trigonometric analysis for a planar configuration (Ghanbari and Noorani, 2011; Ghanbari *et al.*, 2008).

Forward Kinematics using Rotor Algebra $G^+_{3,0,0}$

Rotors, as elements of Clifford (geometric) algebra are even grade subset of Euclidean algebra of 3D space and hence we denote it by $G^+_{3,0,0}$. They provide a succinct, coordinate-free representation of rotations. By encoding rotations through bivector exponentials, rotors enable numerically stable propagation of motion along a kinematic chain. This method is especially useful when the joints are purely rotational (Bayro-Corrochano, 2020).

In 3D Geometric Algebra for the Euclidean 3D Space $G^+_{3,0,0}$, we have $2^3 = 8$ elements namely, 1 (scalar or grade 0 element), e_1, e_2, e_3 (vectors or grade 1 elements), e_{12}, e_{23}, e_{31} (bivectors or grade 2 elements) and lastly e_{123} (psuedoscalar or grade 3 element). The even grade subset consists of only scalars and bivectors (Bayro-

Corrochano, 2020). We also observe that rotations are just two reflections and rotors are isomorphic to quaternions.

In the home configuration, all joint angles θ_i are zero. The joint positions along the e_1 axis are given by,

$$\begin{aligned} P_1(0) &= 0 \\ P_2(0) &= l_1 e_1 \\ P_3(0) &= (l_1 + l_2) e_1 \\ P_4(0) &= (l_1 + l_2 + l_3) e_1 \\ P_5(0) &= (l_1 + l_2 + l_3 + l_4) e_1 \\ X(0) &= (l_1 + l_2 + l_3 + l_4 + l_5) e_1 \end{aligned} \tag{3.1}$$

Rotor Definition

A rotor that rotates a vector by an angle θ_i in the e_{12} plane is defined as,

$$R_i \;=\; e^{-\frac{\theta_i}{2} e_{12}} \;=\; \cos\!\left(\frac{\theta_i}{2}\right) - \; e_{12} \, \sin\!\left(\frac{\theta_i}{2}\right) \tag{3.2}$$

with its reverse given by:

$$\widetilde{R_i} \;=\; \cos\!\left(\frac{\theta_i}{2}\right) + \; e_{12} \, \sin\!\left(\frac{\theta_i}{2}\right) \tag{3.3}$$

A vector v rotated by R_i becomes:

$$v' \;=\; R_i \, v \, \widetilde{R_i}. \tag{3.4}$$

For each link, we compute the translational displacement using

$$P_i \;=\; P_i(0) - R_i \, P_i(0) \, \widetilde{R_i} \tag{3.5}$$

The complete forward kinematics for the end-effector is then obtained by composing the rotations and the translations:

$$\begin{aligned} X = {} & R_1 R_2 R_3 R_4 R_5 \, X(0) \, \widetilde{R_5}\widetilde{R_4}\widetilde{R_3}\widetilde{R_2}\widetilde{R_1} \\ & + \; P_1 + R_1 P_2 \, \widetilde{R_1} + R_1 R_2 P_3 \, \widetilde{R_2}\widetilde{R_1} \\ & + \; R_1 R_2 R_3 P_4 \, \widetilde{R_3}\widetilde{R_2}\widetilde{R_1} + R_1 R_2 R_3 R_4 P_5 \, \widetilde{R_4}\widetilde{R_3}\widetilde{R_2}\widetilde{R_1} \end{aligned} \tag{3.6}$$

This expression encapsulates the entire transformation from the base frame to the end-effector by sequentially applying the appropriate rotors and translation vectors.

Forward Kinematics Using Motor Algebra

Motor algebra extends rotor algebra by unifying rotation and translation into a single entity. A motor M can be written as:

$$M = R\left(1 + \tfrac{1}{2}\, e_4\, T\right) \tag{3.7}$$

where:

- R is the rotor part representing rotation,
- T is a translation vector,
- e_4 is the unit bivector that commutes with scalars and squares to zero.

Motor Derivation for a Single Link

For a given joint, assume the rotation is $R_i = e^{-\frac{\theta_i}{2} e_{12}}$ and the translation (in the link's local frame) is $T_i = l_i e_1$. The corresponding motor is then:

$$M_i = e^{-\frac{\theta_i}{2} e_{12}} \left(1 + \frac{1}{2}\, e_4\, l_i e_1\right) \tag{3.8}$$

The reverse motor is given by:

$$\widetilde{M_i} = \left(1 - \frac{1}{2}\, e_4\, l_i e_1\right) e^{\frac{\theta_i}{2} e_{12}} \tag{3.9}$$

Sequential Composition

The end-effector pose is obtained by composing the motors for each joint:

$$X = M_1 M_2 M_3 M_4 M_5 \, X(0) \, \widetilde{M_5} \widetilde{M_4} \widetilde{M_3} \widetilde{M_2} \widetilde{M_1} \tag{3.10}$$

This product captures both the rotation and the translation at each stage. Note that when the translation parts are small compared to the link lengths, the exponential form naturally approximates the product of separate rotations and translations. This derivation is consistent with the rotor-only formulation, where translations were handled separately. In the motor algebra formulation, both effects are integrated, showing that both approaches yield the same overall transformation when interpreted properly.

Forward Kinematics Using Screw Theory

In this context of a planar serial robot with revolute joints, each joint rotates about an axis perpendicular to the plane of motion (the z–axis), and each link is constrained to move in the xy–plane. Screw theory provides a unifying way to represent rigid–body motions as a combination of rotation about and translation along a common axis. The Product of Exponentials (PoE) formula is particularly elegant for analyzing the forward kinematics of such robotic mechanisms.

According to Medrano-Hermosillo *et al.* (2022), for an n–joint serial manipulator, the forward kinematics can be written as the product of exponentials acting on the end–effector's home position $X(0)$:

$$\begin{aligned} X(\theta_1, \ldots, \theta_n) &= e^{\xi_1 \theta_1} \, e^{\xi_2 \theta_2} \cdots e^{\xi_n \theta_n} \, X(0) \\ &= R_1 R_2 \cdots R_n \, X(0) + \Big[P_1 + R_1 P_2 + R_1 R_2 P_3 + \cdots + R_1 R_2 \cdots R_{n-1} P_n \Big] \end{aligned} \tag{3.11}$$

where each R_i is a rotation about the z–axis by θ_i and each P_i is the translation given by

$$P_i = \big(I - R_i\big) P_i(0) \tag{3.12}$$

Consider a 5–link planar manipulator with five revolute joints, each rotating about the z–axis. Let the link lengths be l_1, l_2, l_3, l_4, l_5. In the home (zero–angle) configuration the links are aligned along the x–axis. Thus, the end–effector's home position is

$$X(0) = \begin{bmatrix} l_1 + l_2 + l_3 + l_4 + l_5 \\ 0 \\ 0 \end{bmatrix} \tag{3.13}$$

Each joint i has the rotation matrix

$$R_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.14}$$

For the base joint, $P_1(0) = \mathbf{0}$, hence

$$P_1 = \big(I_{3\times 3} - R_1\big) P_1(0) = \mathbf{0} \tag{3.15}$$

where $I_{3\times r}$ stands for 3x3 identity matrix.

Similarly we find the home positions for joints 2 to 5. Thus, the complete forward kinematics expression for the 5–link robot is

$$\begin{aligned} X(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) &= R_1 R_2 R_3 R_4 R_5 \, X(0) \\ &\quad + \Big[P_1 + R_1 P_2 + R_1 R_2 P_3 + R_1 R_2 R_3 P_4 + R_1 R_2 R_3 R_4 P_5 \Big] \end{aligned} \tag{3.16}$$

Since the product $R_1R_2R_3R_4R_5$ is a rotation by $\Theta = \theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5$, we have

$$R_1R_2R_3R_4R_5\, X(0) = \begin{bmatrix} (l_1 + l_2 + l_3 + l_4 + l_5)\cos\Theta \\ (l_1 + l_2 + l_3 + l_4 + l_5)\sin\Theta \\ 0 \end{bmatrix}. \tag{3.17}$$

For many planar serial arms with links attached end–to–end (without additional offsets), the forward kinematics can be simplified to the familiar chain form. In the case of a 5–link planar manipulator, the position of the end–effector is given by

$$X(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) = \begin{bmatrix} x(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) \\ y(\theta_1, \theta_2, \theta_3, \theta_4, \theta_5) \\ 0 \end{bmatrix},$$

$$\begin{aligned} \text{where} \quad x(\theta_1, \ldots, \theta_5) &= l_1\cos\theta_1 + l_2\cos(\theta_1 + \theta_2) + l_3\cos(\theta_1 + \theta_2 + \theta_3) \\ &\quad + l_4\cos(\theta_1 + \theta_2 + \theta_3 + \theta_4) + l_5\cos(\theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5), \\ y(\theta_1, \ldots, \theta_5) &= l_1\sin\theta_1 + l_2\sin(\theta_1 + \theta_2) + l_3\sin(\theta_1 + \theta_2 + \theta_3) \\ &\quad + l_4\sin(\theta_1 + \theta_2 + \theta_3 + \theta_4) + l_5\sin(\theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5). \end{aligned} \tag{3.18}$$

Inverse Kinematics Using a Trigonometric Approach for Mechatronal Gaits

In certain cases, such as planar robots with negligible link mass and caterpillar-like locomotion, an inverse kinematics solution can be derived using trigonometry. This section focuses on a gait-based approach as shown below in Figure 3.1.

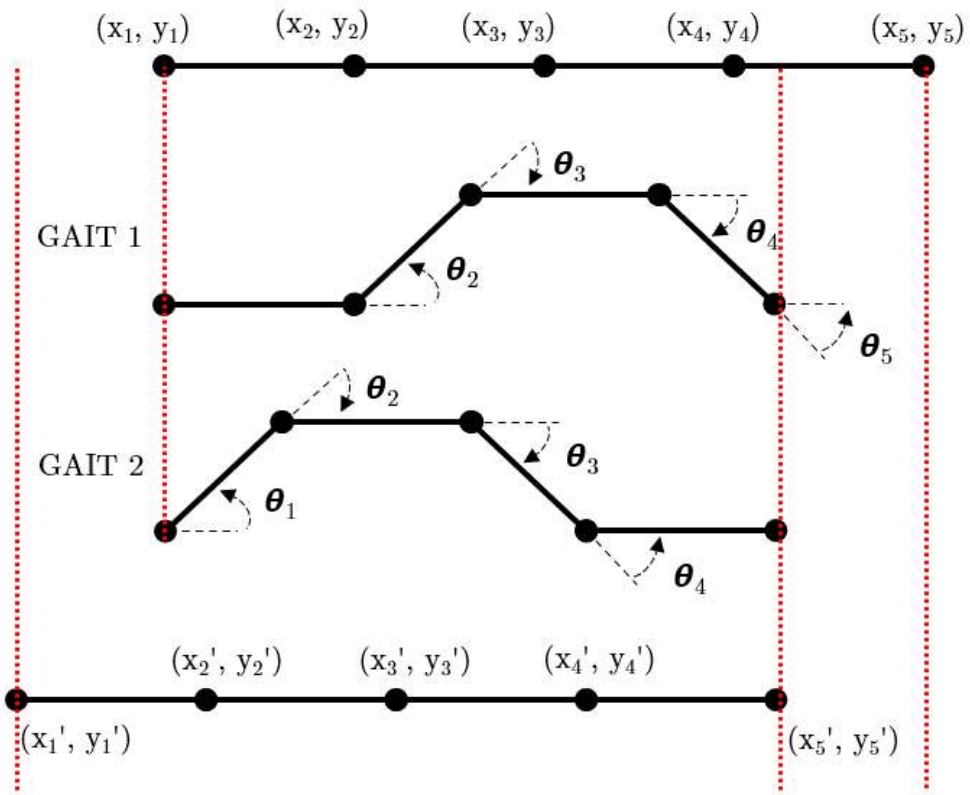


Figure 3.1: Traversal Cycle of the Caterpillar Robot

Gait 1 Derivation

Assume a planar robot with uniform link length l and six links. For Gait 1 as shown in Figure 6.2, the joint angles are set as:

$$\theta_1 = 0, \quad \theta_2 = +\theta, \quad \theta_3 = -\theta, \quad \theta_4 = -\theta, \quad \theta_5 = +\theta.$$

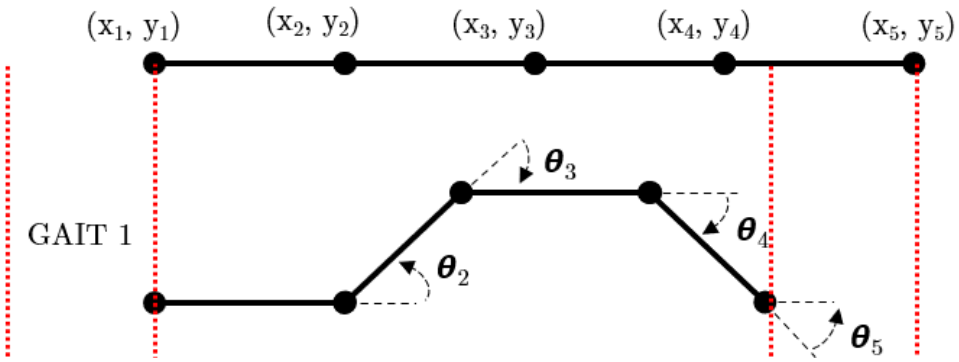


Figure 3.2: Gait 1

Define the cumulative angles:

$$\phi_1 = \theta_1 = 0,$$

$$\phi_2 = \theta_1 + \theta_2 = \theta,$$

$$\phi_3 = \theta_1 + \theta_2 + \theta_3 = 0,$$

$$\phi_4 = \theta_1 + \theta_2 + \theta_3 + \theta_4 = -\theta,$$

$$\phi_5 = \theta_1 + \theta_2 + \theta_3 + \theta_4 + \theta_5 = 0.$$

Starting with the head at $(x_0, y_0) = (0, 0)$, the coordinates of subsequent joints are computed as:

$$(x_i, y_i) = \big(x_{i-1} + l\cos(\phi_{i-1}),\ y_{i-1} + l\sin(\phi_{i-1})\big).$$

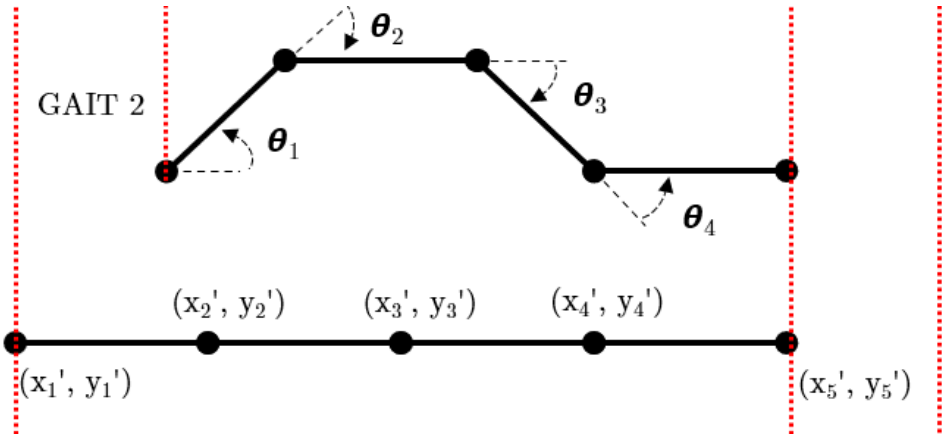


Figure 3.3: Gait 2

For example:

$$(x_1, y_1) = (l, 0),$$

$$(x_2, y_2) = (2l, 0),$$

$$(x_3, y_3) = \Big(2l + l\cos\theta,\ l\sin\theta\Big),$$

$$(x_4, y_4) = \Big(3l + l\cos\theta,\ l\sin\theta\Big),$$

$$(x_5, y_5) = \Big(3l + 2l\cos\theta,\ 0\Big),$$

$$(x_6, y_6) = \Big(4l + 2l\cos\theta,\ 0\Big).$$

The initial total length is $L_{\text{initial}} = 6l$. After Gait 1, the effective length along the x-axis becomes:

$$L_{\text{gait}} = 4l + 2l\cos\theta.$$

Thus, the net displacement (tail pull) is:

$$\eta = L_{\text{initial}} - L_{\text{gait}} = 6l - (4l + 2l\cos\theta) = 2l\,(1 - \cos\theta).$$

Gait 2 Derivation

For Gait 2 as shown in Figure 3.3, the joint angles are configured as:

$$\theta_1 = +\theta, \quad \theta_2 = -\theta, \quad \theta_3 = -\theta, \quad \theta_4 = +\theta, \quad \theta_5 = 0,$$

yielding cumulative angles:

$$\begin{aligned}
\phi_1 &= \theta, \\
\phi_2 &= \theta + (-\theta) = 0, \\
\phi_3 &= \theta + (-\theta) + (-\theta) = -\theta, \\
\phi_4 &= \theta + (-\theta) + (-\theta) + (+\theta) = 0, \\
\phi_5 &= \theta + (-\theta) + (-\theta) + (+\theta) + 0 = 0.
\end{aligned}$$

Again, using the forward coordinate relations,

$$(x_i, y_i) = \big(x_{i-1} + l\cos(\phi_{i-1}),\ y_{i-1} + l\sin(\phi_{i-1})\big),$$

we calculate the new positions. Once the gait is complete, the joints relax back to their home positions by uniformly shifting each joint coordinate by η in the $-x$ direction:

$$x_i' = x_i - \eta \quad \text{for } i = 1, \ldots, 6.$$

Thus, a cyclic gait motion results in a net translation of η per cycle.

Chapter 4

DYNAMICS

The dynamics of a worm robot are characterized by non-linear interactions, segmental motions, inertial properties, coriolis forces, gravitational effects, and external reaction forces from the environment. Its movement mimics the gait of an inchworm, where coordinated contractions and extensions create a continuous wave that travels from the tail to the head, resulting in forward motion through small, incremental steps. In contrast, the caterpillar robot, with its longer joint modules, demonstrates a more distributed application of force. The inchworm model operates within an open-chain kinematic framework, achieving movement through periodic anchoring and stretching. Meanwhile, the caterpillar robot transitions between open-chain and closed-chain states, facilitating more continuous motion. A crawling robot inspired by the inchworm's locomotion employs Euler-Lagrange dynamics to model this crawling motion (Rahmani and Redkar, 2024b; Ghanbari and Noorani, 2011; Ab Rashid *et al.*, 2020; Rahmani and Redkar, 2024a).

Non-Linear Dynamics using Euler-Lagrange Equation

We developed a dynamic model of a planner manipulator with revolute joints, incorporating a closed-loop chain and reaction forces. The non linear gait dynamics as per Ghanbari and Noorani (2011) (as shown in Figure 4.1) is calculated as,

$$M(\theta)\ddot{\theta} + C(\theta)\dot{\theta}^2 + G(\theta) = D\tau + \lambda(\theta)F_R \tag{4.1}$$

where for a 4 DOF robot, $M(\theta) \in R^{4\times4}$, $C(\theta) \in R^{4\times4}$, $G(\theta) \in R^{4\times1}$, $D \in R^{4\times4}$, F_R stands for mass matrix, coriolis matrix, gravity matrix, subtraction matrix

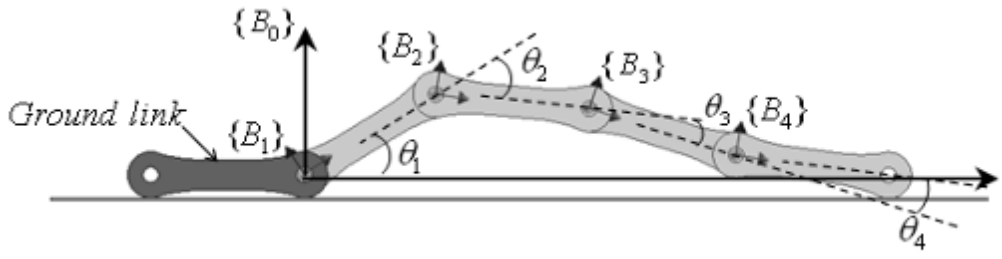


Figure 4.1: 4 DOF Manipulator Model (Ghanbari and Noorani, 2011)

and reaction forces with respect to surface. But for a 5 DOF the dimensions change as $M(\theta) \in R^{5\times5}$, $C(\theta) \in R^{5\times5}$, $G(\theta) \in R^{5\times1}$, $D \in R^{5\times5}$. Let us begin by calculating the square of velocity of the i^th link centroid,

$$v_p^2 = \frac{l_p^2}{4}\dot{\varphi_p^2} + \sum_{i=1}^{p}\sum_{j=1}^{p-1} l_i l_j \dot{\varphi_i}\dot{\varphi_j} cos(\dot{\varphi_i} - \dot{\varphi_j}) \tag{4.2}$$

The height of the centroid for each link is given by,

$$h_p = \sum_{j=1}^{p} l_j sin(\varphi_j) - \frac{l_p}{2} sin(\varphi_p) \tag{4.3}$$

Next, we calculate the total kinetic energy and potential energy,

$$T = \frac{1}{2}\sum_{p=1}^{5}\left[m_p v_p^2 + I_p \dot{\varphi_p^2}\right] \tag{4.4}$$

$$V = \sum_{p=1}^{5} m_p g h_p \tag{4.5}$$

Assuming the links are identical, the masses stay the same and I equals $\frac{ml^2}{12}$. Equation 4.4 can be expanded as,

$$T = \frac{m}{2}\left[v_1^2 + v_2^2 + v_3^2 + v_4^2 + v_5^2\right] + \frac{ml^2}{24}\left[\dot{\varphi_1^2} + \dot{\varphi_2^2} + \dot{\varphi_3^2} + \dot{\varphi_4^2} + \dot{\varphi_5^2}\right] \tag{4.6}$$

Similarly, equation 4.5 can also be expanded as,

$$V = g\,l\left[\frac{9m}{2}\sin(\varphi_1) + \frac{7m}{2}\sin(\varphi_2) + \frac{5m}{2}\sin(\varphi_3) + \frac{3m}{2}\sin(\varphi_4) + \frac{m}{2}\sin(\varphi_5)\right]$$

$$V = \frac{m\,g\,l}{2}\Big[9\sin(\varphi_1) + 7\sin(\varphi_2) + 5\sin(\varphi_3) + 3\sin(\varphi_4) + \sin(\varphi_5)\Big] \tag{4.7}$$

Considering the non-conservative forces acting on the system represented by Q_k, the virtual work done by all such forces are given by,

$$\delta W = \tau \cdot \delta\Theta + \left(F_j\hat{i} + F_R\hat{j}\right) \cdot \delta\mathbf{P}_{p+1} = \sum_{p=1}^{5} Q_p \delta\varphi_p \tag{4.8}$$

where $Q_p = \tau_p - \tau_{p-1} + \lambda_p F_R$ and $\lambda_p = l_p(cos\phi_p + \bar{\mu}sin\phi_p)$ Using Euler Lagrange equations, motion equations of dynamic systems are written as,

$$\frac{d}{dt}\left[\frac{\partial T}{\partial\dot{\varphi}_k}\right] - \frac{\partial T}{\partial\varphi_k} + \frac{\partial V}{\partial\varphi_k} = Q_k \tag{4.9}$$

Substituting 4.4, 4.5, 4.8 and transforming it in the form 4.1, we get,

$$M(\theta) = \frac{m\,l^2}{6}\begin{pmatrix} 26 & 21\,C_{12} & 15\,C_{13} & 12\,C_{14} & 6\,C_{15} \\ 21\,C_{12} & 20 & 18\,C_{23} & 12\,C_{24} & 6\,C_{25} \\ 15\,C_{13} & 18\,C_{23} & 14 & 12\,C_{34} & 6\,C_{35} \\ 12\,C_{14} & 12\,C_{24} & 12\,C_{34} & 8 & 6\,C_{45} \\ 6\,C_{15} & 6\,C_{25} & 6\,C_{35} & 6\,C_{45} & 2 \end{pmatrix} \tag{4.10}$$

$$C(\theta) = \frac{m\,l^2}{6}\begin{pmatrix} 0 & 21\,S_{12} & 15\,S_{13} & 12\,S_{14} & 6\,S_{15} \\ -21\,S_{12} & 0 & 18\,S_{23} & 12\,S_{24} & 6\,S_{25} \\ -15\,S_{13} & -18\,S_{23} & 0 & 12\,S_{34} & 6\,S_{35} \\ -12\,S_{14} & -12\,S_{24} & -12\,S_{34} & 0 & 6\,S_{45} \\ -6\,S_{15} & -6\,S_{25} & -6\,S_{35} & -6\,S_{45} & 0 \end{pmatrix} \tag{4.11}$$

$$G(\theta) = \frac{m\,g\,l}{2} \begin{pmatrix} 9\,\cos(\varphi_1) \\ 7\,\cos(\varphi_2) \\ 5\,\cos(\varphi_3) \\ 3\,\cos(\varphi_4) \\ \cos(\varphi_5) \end{pmatrix} \tag{4.12}$$

$$D = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.13}$$

Dynamics using Motor Algebra

In motor algebra, line axes are represented as Plücker lines, which simultaneously encode both the direction and the moment of the line in 3D space in bivector format. A line in 3D space is defined by:

- A direction vector $\mathbf{n}$, indicating the orientation of the line.
- A moment vector $\mathbf{m} = \mathbf{p} \times \mathbf{n}$, capturing the offset of the line relative to the origin, where $\mathbf{p}$ is any point on the line.

In motor algebra, these components are combined into a single object known as the *screw axis*:

$$L = n + Im \tag{4.14}$$

where:

- n is the bivector representing the line's direction.

- m is the bivector representing the line's moment.
- I is the pseudoscalar.

Consider a serial chain of revolute joints, each rotating about the y-axis. The base joint (Joint 1) is located at the origin with:

$$n = -e_{13}, \quad m = 0 \tag{4.15}$$

Thus, the screw axis of the first joint is:

$$L_1 = -e_{13} \tag{4.16}$$

For subsequent joints, which are offset along the x-axis by distances $l_1, l_2, \ldots, l_5$, the position of the i-th joint is:

$$\mathbf{p}_i = \begin{bmatrix} \sum_{j=1}^{i-1} l_j \\ 0 \\ 0 \end{bmatrix} \tag{4.17}$$

The corresponding moment is:

$$m_i = \mathbf{p}_i \times \mathbf{n} = \left(\sum_{j=1}^{i-1} l_j \right) e_{12} \tag{4.18}$$

The screw axis of the i-th joint becomes:

$$L_i = -e_{13} + I \left(\sum_{j=1}^{i-1} l_j \right) e_{12} \tag{4.19}$$

After translating the joint by l_1 in the $+x$ direction, the new screw axis is:

$$L_2 = -e_{13} + I \, (l_1 \, e_{12}) \tag{4.20}$$

This result shows that the direction of the joint (along the y-axis) remains the same, while the moment now reflects an offset of l_1 in the x-direction.

Specifically, we have:

- **Joint 1:** At the origin,

$$L_1 = -e_{13}. \tag{4.21}$$

- **Joint 2:** Translated by l_1 along $+x$,

$$L_2 = -e_{13} + I\left(l_1\, e_{12}\right). \tag{4.22}$$

- **Joint 3:** Translated by $l_1 + l_2$ along $+x$,

$$L_3 = -e_{13} + I\left((l_1 + l_2)\, e_{12}\right). \tag{4.23}$$

- **Joint 4:** Translated by $l_1 + l_2 + l_3$ along $+x$,

$$L_4 = -e_{13} + I\left((l_1 + l_2 + l_3)\, e_{12}\right). \tag{4.24}$$

- **Joint 5:** Translated by $l_1 + l_2 + l_3 + l_4$ along $+x$,

$$L_5 = -e_{13} + I\left((l_1 + l_2 + l_3 + l_4)\, e_{12}\right). \tag{4.25}$$

- **Joint 6:** Translated by $l_1 + l_2 + l_3 + l_4 + l_5$ along $+x$,

$$L_6 = -e_{13} + I\left((l_1 + l_2 + l_3 + l_4 + l_5)\, e_{12}\right). \tag{4.26}$$

The velocity screw (twist) s_j of the j-th link is computed recursively as:

$$s_j = \sum_{k=1}^{j} L_k \dot{\theta}_k \tag{4.27}$$

where $\dot{\theta}_k$ is the joint velocity of the k-th joint. The acceleration $\dot{s}_j$ of the j-th link is given by:

$$\dot{s}_j = \sum_{k=1}^{j} \ddot{\theta}_k L_k + \sum_{k=1}^{j} \dot{\theta}_k \dot{L}_k \tag{4.28}$$

where the time derivative of each screw axis L_k involves Lie brackets:

$$\dot{L}_k = [s_{k-1}, L_k] \tag{4.29}$$

Equations of motion for each link can be given as,

$$
\begin{aligned}
&w_5 + R_5 + G_5 = N_5(\dot{s_5}) + \{s_5, N_5(s_5)\} \\
&w_4 + R_4 + G_4 - w_5 - R_5 = N_4(\dot{s_4}) + \{s_4, N_4(s_4)\} \\
&w_3 + R_3 + G_3 - w_4 - R_4 = N_3(\dot{s_3}) + \{s_3, N_3(s_3)\} \\
&w_2 + R_2 + G_2 - w_3 - R_3 = N_2(\dot{s_2}) + \{s_2, N_2(s_2)\} \\
&w_1 + R_1 + G_1 - w_2 - R_2 = N_1(\dot{s_1}) + \{s_1, N_1(s_1)\}
\end{aligned}
$$

After reordering these equations,

$$
\begin{aligned}
w_5 + R_5 &= N_5(\dot{s_5}) + \{s_5, N_5(s_5)\} - G_5 \\
w_4 + R_4 &= N_4(\dot{s_4}) + \{s_4, N_4(s_4)\} + N_5(\dot{s_5}) + \{s_5, N_5(s_5)\} \\
&\quad - G_5 - G_4 \\
w_3 + R_3 &= N_3(\dot{s_3}) + \{s_3, N_3(s_3)\} + N_4(\dot{s_4}) + \{s_4, N_4(s_4)\} \\
&\quad + N_5(\dot{s_5}) + \{s_5, N_5(s_5)\} - G_5 - G_4 - G_3 \\
w_2 + R_2 &= N_2(\dot{s_2}) + \{s_2, N_2(s_2)\} + N_3(\dot{s_3}) + \{s_3, N_3(s_3)\} \\
&\quad + N_4(\dot{s_4}) + \{s_4, N_4(s_4)\} + N_5(\dot{s_5}) + \{s_5, N_5(s_5)\} \\
&\quad - G_5 - G_4 - G_3 - G_2 \\
w_1 + R_1 &= N_1(\dot{s_1}) + \{s_1, N_1(s_1)\} + N_2(\dot{s_2}) + \{s_2, N_2(s_2)\} \\
&\quad + N_3(\dot{s_3}) + \{s_3, N_3(s_3)\} + N_4(\dot{s_4}) + \{s_4, N_4(s_4)\} \\
&\quad + N_5(\dot{s_5}) + \{s_5, N_5(s_5)\} - G_5 - G_4 - G_3 - G_2 \\
&\quad - G_1
\end{aligned}
\tag{4.30}
$$

We will pair each equation with L_i (or $[L_1 \ L_2 \ L_3 \ L_4 \ L_5]$) to eliminate the reaction

forces,

$$
\begin{aligned}
\tau_5 &= N_5(\dot{s_5}) \cdot L_5 + \{s_5, N_5(s_5)\} \cdot L_5 - G_5 \cdot L_5 \\
\tau_4 &= N_4(\dot{s_4}) \cdot L_4 + \{s_4, N_4(s_4)\} \cdot L_4 + N_5(\dot{s_5}) \cdot L_4 \\
&\quad + \{s_5, N_5(s_5)\} \cdot L_4 - (G_4 + G_5) \cdot L_4 \\
\tau_3 &= N_3(\dot{s_3}) \cdot L_3 + \{s_3, N_3(s_3)\} \cdot L_3 + N_4(\dot{s_4}) \cdot L_3 \\
&\quad + \{s_4, N_4(s_4)\} \cdot L_3 + N_5(\dot{s_5}) \cdot L_3 + \{s_5, N_5(s_5)\} \cdot L_3 \\
&\quad - (G_3 + G_4 + G_5) \cdot L_3 \\
\tau_2 &= N_2(\dot{s_2}) \cdot L_2 + \{s_2, N_2(s_2)\} \cdot L_2 + N_3(\dot{s_3}) \cdot L_2 \\
&\quad + \{s_3, N_3(s_3)\} \cdot L_2 + N_4(\dot{s_4}) \cdot L_2 \\
&\quad + \{s_4, N_4(s_4)\} \cdot L_2 + N_5(\dot{s_5}) \cdot L_2 + \{s_5, N_5(s_5)\} \cdot L_2 \\
&\quad - (G_2 + G_3 + G_4 + G_5) \cdot L_2 \\
\tau_1 &= N_1(\dot{s_1}) \cdot L_1 + \{s_1, N_1(s_1)\} \cdot L_1 + N_2(\dot{s_2}) \cdot L_1 \\
&\quad + \{s_2, N_2(s_2)\} \cdot L_1 + N_3(\dot{s_3}) \cdot L_1 + \{s_3, N_3(s_3)\} \cdot L_1 \\
&\quad + N_4(\dot{s_4}) \cdot L_1 + \{s_4, N_4(s_4)\} \cdot L_1 + N_5(\dot{s_5}) \cdot L_1 \\
&\quad + \{s_5, N_5(s_5)\} \cdot L_1 - (G_1 + G_2 + G_3 + G_4 + G_5) \cdot L_1
\end{aligned}
\tag{4.31}
$$

The velocity screw or twist for link j is given using Jacobian J with columns as joint

line screws L_i,

$$s_j = J\dot{\theta} = [L_1\ L_2\ L_3\ L_4\ L_5][\dot{\theta_1}\ \dot{\theta_2}\ \dot{\theta_3}\ \dot{\theta_4}\ \dot{\theta_5}\ ]^T$$

The link velocities are,

$$s_1 = L_1\dot{\theta_1}$$

$$s_2 = L_1\dot{\theta_1} + L_2\dot{\theta_2}$$

$$s_3 = L_1\dot{\theta_1} + L_2\dot{\theta_2} + L_3\dot{\theta_3}$$

$$s_4 = L_1\dot{\theta_1} + L_2\dot{\theta_2} + L_3\dot{\theta_3} + L_4\dot{\theta_4}$$

$$s_5 = L_1\dot{\theta_1} + L_2\dot{\theta_2} + L_3\dot{\theta_3} + L_4\dot{\theta_4} + L_5\dot{\theta_5}$$

$$s_j = \sum_{k=1}^{n} L_j\dot{\theta_j} \tag{4.32}$$

Subsequently the link accelerations are given by,

$$\dot{s_1} = \ddot{\theta_1}L_1 + \dot{\theta_1}\dot{L_1}$$

$$\dot{s_2} = \ddot{\theta_2}L_2 + \dot{\theta_2}\dot{L_2} + \ddot{\theta_1}L_1 + \dot{\theta_1}\dot{L_1}$$

$$\dot{s_3} = \ddot{\theta_3}L_3 + \dot{\theta_3}\dot{L_3} + \ddot{\theta_2}L_2 + \dot{\theta_2}\dot{L_2} + \ddot{\theta_1}L_1 + \dot{\theta_1}\dot{L_1}$$

$$\dot{s_4} = \ddot{\theta_4}L_4 + \dot{\theta_4}\dot{L_4} + \ddot{\theta_3}L_3 + \dot{\theta_3}\dot{L_3} + \ddot{\theta_2}L_2 + \dot{\theta_2}\dot{L_2} + \ddot{\theta_1}L_1 + \dot{\theta_1}\dot{L_1}$$

$$\dot{s_5} = \ddot{\theta_5}L_5 + \dot{\theta_5}\dot{L_5} + \ddot{\theta_4}L_4 + \dot{\theta_4}\dot{L_4} + \ddot{\theta_3}L_3 + \dot{\theta_3}\dot{L_3} + \ddot{\theta_2}L_2 + \dot{\theta_2}\dot{L_2} + \ddot{\theta_1}L_1 + \dot{\theta_1}\dot{L_1}$$

We know that,

$$\begin{aligned} L_2(t) &= M_1(t)L_2(0)\widetilde{M_1}(t) \\ L_3(t) &= M_2(t)L_3(0)\widetilde{M_2}(t) \\ L_4(t) &= M_3(t)L_4(0)\widetilde{M_3}(t) \\ L_5(t) &= M_4(t)L_5(0)\widetilde{M_4}(t) \end{aligned} \tag{4.33}$$

We know that derivative of screw line is, $\frac{d}{dt}M_i[n_j\ m_j]^T = [s_i\ L_j]$

Let us replace the derivatives of screw axes ($\dot{L}_i$) with Lie brackets,

$$[s_1, L_1] \equiv ad(s_1)L_1 \tag{4.34}$$

$$\begin{aligned} \dot{L_1} &= \frac{d}{dt}L_1(t) = [s_1, L_1] = \dot{\theta_1}[L_1, L_1] = 0 \\ \dot{L_2} &= \frac{d}{dt}L_2(t) = [s_1, L_2] = \dot{\theta_1}[L_1, L_2] \\ \dot{L_3} &= \frac{d}{dt}L_3(t) = [s_2, L_3] = [L_1\dot{\theta_1} + L_2\dot{\theta_2}, L_3] \\ \dot{L_4} &= \frac{d}{dt}L_4(t) = [s_3, L_4] = [L_1\dot{\theta_1} + L_2\dot{\theta_2} + L_3\dot{\theta_3}, L_4] \\ \dot{L_5} &= \frac{d}{dt}L_5(t) = [s_4, L_5] = [L_1\dot{\theta_1} + L_2\dot{\theta_2} + L_3\dot{\theta_3} + L_4\dot{\theta_4}, L_5] \end{aligned} \tag{4.35}$$

Further simplifying the link accelerations,

$$
\begin{aligned}
\dot{s_1} &= \ddot{\theta_1} L_1 \\
\dot{s_2} &= \ddot{\theta_1} L_1 + \ddot{\theta_2} L_2 + \dot{\theta_1}\dot{\theta_2}[L_1, L_2] \\
\dot{s_3} &= \ddot{\theta_1} L_1 + \ddot{\theta_2} L_2 + \ddot{\theta_3} L_3 + \dot{\theta_3}[L_1\dot{\theta_1} + L_2\dot{\theta_2}, L_3] \\
&\quad + \dot{\theta_2}\dot{\theta_1}[L_1, L_2] \\
\dot{s_4} &= \ddot{\theta_1} L_1 + \ddot{\theta_2} L_2 + \ddot{\theta_3} L_3 + \ddot{\theta_4} L_4 \\
&\quad + \dot{\theta_4}[L_1\dot{\theta_1} + L_2\dot{\theta_2} + L_3\dot{\theta_3}, L_4] + \dot{\theta_3}[L_1\dot{\theta_1} + L_2\dot{\theta_2}, L_3] \\
&\quad + \dot{\theta_2}\dot{\theta_1}[L_1, L_2] \\
\dot{s_5} &= \ddot{\theta_1} L_1 + \ddot{\theta_2} L_2 + \ddot{\theta_3} L_3 + \ddot{\theta_4} L_4 + \ddot{\theta_5} L_5 \\
&\quad + \dot{\theta_5}[L_1\dot{\theta_1} + L_2\dot{\theta_2} + L_3\dot{\theta_3} + L_4\dot{\theta_4}, L_5] \\
&\quad + \dot{\theta_4}[L_1\dot{\theta_1} + L_2\dot{\theta_2} + L_3\dot{\theta_3}, L_4] + \dot{\theta_3}[L_1\dot{\theta_1} + L_2\dot{\theta_2}, L_3] \\
&\quad + \dot{\theta_2}\dot{\theta_1}[L_1, L_2]
\end{aligned}
\tag{4.36}
$$

Using Linearity and Leibniz Rule of Lie Brackets, we can expand each Lie bracket more for Coriolis terms,

For example,

$$
\dot{\theta_3}[L_1\dot{\theta_1} + L_2\dot{\theta_2}, L_3] = \dot{\theta_3}\dot{\theta_1}[L_1, L_3] + \dot{\theta_3}\dot{\theta_2}[L_2, L_3] \tag{4.37}
$$

$$
\begin{aligned}
\tau_i &= \sum_{j=i}^{5} [N_j(\dot{s_j}) \cdot L_i + \{s_j, N_j(s_j)\} \cdot L_i - G_j \cdot L_i] \\
\tau_i &= \sum_{j=i}^{5} [N_j(\dot{s_j} - G_j) \cdot L_i + \{s_j, N_j(s_j)\} \cdot L_i]
\end{aligned}
\tag{4.38}
$$

as $G_j = N_j(G)$

Replacing s_j with S_i, as a 6×1 vector $S = [\omega, v]^T$,

$$
\begin{aligned}
\tau_5 &= \dot{S_5}^T N_5 L_5 + {S_5}^T N_5 [L_5, S_5] - G_5 \cdot L_5 \\
\tau_4 &= \dot{S_4}^T N_4 L_4 + {S_4}^T N_4 [L_4, S_4] - G_4 \cdot L_4 \\
&\quad + \dot{S_5}^T N_5 L_4 + {S_5}^T N_5 [L_4, S_5] - G_5 \cdot L_4 \\
\tau_3 &= \dot{S_3}^T N_3 L_3 + {S_3}^T N_3 [L_3, S_3] - G_3 \cdot L_3 + \dot{S_4}^T N_4 L_3 \\
&\quad + {S_4}^T N_4 [L_3, S_3] - G_4 \cdot L_3 + \dot{S_5}^T N_5 L_3 + {S_5}^T N_5 [L_3, S_5] \\
&\quad - G_5 \cdot L_3 \\
\tau_2 &= \dot{S_2}^T N_2 L_2 + {S_2}^T N_2 [L_2, S_2] - G_2 \cdot L_2 + \dot{S_3}^T N_3 L_2 \\
&\quad + {S_3}^T N_3 [L_2, S_3] - G_3 \cdot L_2 + \dot{S_4}^T N_4 L_2 + {S_4}^T N_4 [L_2, S_4] \\
&\quad - G_4 \cdot L_2 + \dot{S_5}^T N_5 L_2 + {S_5}^T N_5 [L_2, S_5] - G_5 \cdot L_2 \\
\tau_1 &= \dot{S_1}^T N_1 L_1 + {S_1}^T N_1 [L_1, S_1] - G_1 \cdot L_1 + \dot{S_2}^T N_2 L_1 \\
&\quad + {S_2}^T N_2 [L_1, S_2] - G_2 \cdot L_1 + \dot{S_3}^T N_3 L_1 + {S_3}^T N_3 [L_1, S_3] \\
&\quad - G_3 \cdot L_1 + \dot{S_4}^T N_4 L_1 + {S_4}^T N_4 [L_1, S_4] - G_4 \cdot L_1 \\
&\quad + \dot{S_5}^T N_5 L_1 + {S_5}^T N_5 [L_1, S_5] - G_5 \cdot L_1
\end{aligned}
\tag{4.39}
$$

Let us analyze τ_5 part by part,

$$
\tau_5 = \underbrace{\dot{S_5}^T N_5 L_5}_{(i)} + \underbrace{{S_5}^T N_5 [L_5, S_5]}_{(ii)} - G_5 \cdot L_5
$$

Taking (i) we know,

$$
\begin{aligned}
S_5 &= L_1\dot{\theta}_1 + L_2\dot{\theta}_2 + L_3\dot{\theta}_3 + L_4\dot{\theta}_4 + L_5\dot{\theta}_5 \\
\dot{S}_5 &= \dot{\theta}_1\dot{L}_1 + \ddot{\theta}_1 L_1 + \dot{\theta}_2\dot{L}_2 + \ddot{\theta}_2 L_2 + \dot{\theta}_3\dot{L}_3 \\
&\quad + \ddot{\theta}_3 L_3 + \dot{\theta}_4\dot{L}_4 + \ddot{\theta}_4 L_4 + \dot{\theta}_5\dot{L}_5 + \ddot{\theta}_5 L_5
\end{aligned}
\tag{4.40}
$$

We also know that $\dot{L_k} = [S_{k-1}, L_k]$,

$$\begin{aligned}
\dot{L_1} &= [S_1, L_1] \\
\dot{L_2} &= [S_1, L_2] \\
\dot{L_3} &= [S_2, L_3] \\
\dot{L_4} &= [S_3, L_4] \\
\dot{L_5} &= [S_4, L_5]
\end{aligned} \tag{4.41}$$

$$\begin{aligned}
\dot{S_5} = &\ddot{\theta_1} L_1 + \ddot{\theta_2} L_2 + \ddot{\theta_3} L_3 + \ddot{\theta_4} L_4 + \ddot{\theta_5} L_5 \\
&+ \dot{\theta_2}(\dot{\theta_1}[L_1, L_2]) + \dot{\theta_3}(\dot{\theta_1}[L_1, L_3] + \dot{\theta_2}[L_2, L_3]) \\
&+ \dot{\theta_4}(\dot{\theta_1}[L_1, L_4] + \dot{\theta_2}[L_2, L_4] + \dot{\theta_3}[L_3, L_4]) \\
&+ \dot{\theta_5}(\dot{\theta_1}[L_1, L_5] + \dot{\theta_2}[L_2, L_5] + \dot{\theta_3}[L_3, L_5] \\
&+ \dot{\theta_4}[L_4, L_5])
\end{aligned} \tag{4.42}$$

Therefore,

$$\begin{aligned}
\dot{S_5}^T N_5 L_5 &= \left[\sum_{k=1}^{5} \ddot{\theta_k} L_k + \sum_{p=1}^{5} \sum_{q=1}^{5} \dot{\theta_p} \dot{\theta_q} [L_p, L_q] \right]^T N_5 L_5 \\
&= \underbrace{\sum_{k=1}^{5} \ddot{\theta_k} {L_k}^T N_5 L_5}_{\text{contributes to } M(\theta)} + \underbrace{\sum_{p=1}^{5} \sum_{q=1}^{5} \dot{\theta_p} \dot{\theta_q} [L_p, L_q]^T N_5 L_5}_{\text{contributes to } C(\theta, \dot{\theta})}
\end{aligned} \tag{4.43}$$

Next,

$$\begin{aligned}
S_5 &= \sum_{m=1}^{5} L_m \dot{\theta_m} \\
[L_5, S_5] &= \sum_{m=1}^{5} \dot{\theta_m} [L_5, L_m] \\
S_5^T N_5 [L_5, S_5] &= \left[\sum_{q=1}^{5} L_q \dot{\theta_q} \right]^T N_5 \left[\sum_{m=1}^{5} \dot{\theta_m} [L_5, L_m] \right] \\
S_5^T N_5 [L_5, S_5] &= \sum_{q=1}^{5} \sum_{m=1}^{5} \dot{\theta_q} \dot{\theta_m} L_q^T N_5 [L_5, L_m]
\end{aligned} \tag{4.44}$$

Similarly, we substitute and find all the torques,

$$\tau_5 = \sum_{k=1}^{5} {L_k}^T N_5 L_5 \ddot{\theta_k} + \sum_{p=1}^{5}\sum_{q=1}^{5} [L_p, L_q]^T N_5 L_5 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{5}\sum_{q=1}^{5} L_p^T N_5 [L_5, L_q]\dot{\theta_p}\dot{\theta_q} - G_5^T L_5$$

$$\begin{aligned}\tau_4 = &\sum_{k=1}^{4} {L_k}^T N_4 L_4 \ddot{\theta_k} + \sum_{p=1}^{4}\sum_{q=1}^{4} [L_p, L_q]^T N_4 L_4 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{4}\sum_{q=1}^{4} L_p^T N_4 [L_4, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{5} {L_k}^T N_5 L_4 \ddot{\theta_k} + \sum_{p=1}^{5}\sum_{q=1}^{5} [L_p, L_q]^T N_5 L_4 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{5}\sum_{q=1}^{5} L_p^T N_5 [L_4, L_q]\dot{\theta_p}\dot{\theta_q} - \left[\sum_{i=4}^{5} G_i^T\right] L_4\end{aligned}$$

$$\begin{aligned}\tau_3 = &\sum_{k=1}^{3} {L_k}^T N_3 L_3 \ddot{\theta_k} + \sum_{p=1}^{3}\sum_{q=1}^{3} [L_p, L_q]^T N_3 L_3 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{3}\sum_{q=1}^{3} L_p^T N_3 [L_3, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{4} {L_k}^T N_4 L_3 \ddot{\theta_k} + \sum_{p=1}^{4}\sum_{q=1}^{4} [L_p, L_q]^T N_4 L_3 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{4}\sum_{q=1}^{4} L_p^T N_4 [L_3, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{5} {L_k}^T N_5 L_3 \ddot{\theta_k} + \sum_{p=1}^{5}\sum_{q=1}^{5} [L_p, L_q]^T N_5 L_3 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{5}\sum_{q=1}^{5} L_p^T N_5 [L_3, L_q]\dot{\theta_p}\dot{\theta_q} - \left[\sum_{i=3}^{5} G_i^T\right] L_3\end{aligned}$$

$$\begin{aligned}\tau_2 = &\sum_{k=1}^{2} {L_k}^T N_2 L_2 \ddot{\theta_k} + \sum_{p=1}^{2}\sum_{q=1}^{2} [L_p, L_q]^T N_2 L_2 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{2}\sum_{q=1}^{2} L_p^T N_2 [L_2, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{3} {L_k}^T N_3 L_2 \ddot{\theta_k} + \sum_{p=1}^{3}\sum_{q=1}^{3} [L_p, L_q]^T N_3 L_2 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{3}\sum_{q=1}^{3} L_p^T N_3 [L_2, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{4} {L_k}^T N_4 L_2 \ddot{\theta_k} + \sum_{p=1}^{4}\sum_{q=1}^{4} [L_p, L_q]^T N_4 L_2 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{4}\sum_{q=1}^{4} L_p^T N_4 [L_2, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{5} {L_k}^T N_5 L_2 \ddot{\theta_k} + \sum_{p=1}^{5}\sum_{q=1}^{5} [L_p, L_q]^T N_5 L_2 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{5}\sum_{q=1}^{5} L_p^T N_5 [L_2, L_q]\dot{\theta_p}\dot{\theta_q} - \left[\sum_{i=2}^{5} G_i^T\right] L_2\end{aligned}$$

$$\begin{aligned}\tau_1 = &\sum_{k=1}^{1} {L_k}^T N_1 L_1 \ddot{\theta_k} + \sum_{p=1}^{1}\sum_{q=1}^{1} [L_p, L_q]^T N_1 L_1 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{1}\sum_{q=1}^{1} L_p^T N_1 [L_1, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{2} {L_k}^T N_2 L_1 \ddot{\theta_k} + \sum_{p=1}^{2}\sum_{q=1}^{2} [L_p, L_q]^T N_2 L_1 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{2}\sum_{q=1}^{2} L_p^T N_2 [L_1, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{3} {L_k}^T N_3 L_1 \ddot{\theta_k} + \sum_{p=1}^{3}\sum_{q=1}^{3} [L_p, L_q]^T N_3 L_1 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{3}\sum_{q=1}^{3} L_p^T N_3 [L_1, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{4} {L_k}^T N_4 L_1 \ddot{\theta_k} + \sum_{p=1}^{4}\sum_{q=1}^{4} [L_p, L_q]^T N_4 L_1 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{4}\sum_{q=1}^{4} L_p^T N_4 [L_1, L_q]\dot{\theta_p}\dot{\theta_q} \\ &+ \sum_{k=1}^{5} {L_k}^T N_5 L_1 \ddot{\theta_k} + \sum_{p=1}^{5}\sum_{q=1}^{5} [L_p, L_q]^T N_5 L_1 \dot{\theta_p}\dot{\theta_q} + \sum_{p=1}^{5}\sum_{q=1}^{5} L_p^T N_5 [L_1, L_q]\dot{\theta_p}\dot{\theta_q} - \left[\sum_{i=1}^{5} G_i^T\right] L_1\end{aligned} \tag{4.45}$$

The final closed loop system for PD control is illustrated by,

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -M^{-1}\Big(K_p\, x_1 + K_v\, x_2 - B(\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5)\Big) \end{bmatrix}. \tag{4.46}$$

where,

$$M = \begin{bmatrix} L_1^T(N_1+N_2+N_3+N_4+N_5)L_1 & L_2^T(N_2+N_3+N_4+N_5)L_1 & L_3^T(N_3+N_4+N_5)L_1 & L_4^T(N_4+N_5)L_1 & L_5^T N_5 L_1 \\ L_1^T(N_2+N_3+N_4+N_5)L_2 & L_2^T(N_2+N_3+N_4+N_5)L_2 & L_3^T(N_3+N_4+N_5)L_2 & L_4^T(N_4+N_5)L_2 & L_5^T N_5 L_2 \\ L_1^T(N_3+N_4+N_5)L_3 & L_2^T(N_3+N_4+N_5)L_3 & L_3^T(N_3+N_4+N_5)L_3 & L_4^T(N_4+N_5)L_3 & L_5^T N_5 L_3 \\ L_1^T(N_4+N_5)L_4 & L_2^T(N_4+N_5)L_4 & L_3^T(N_4+N_5)L_4 & L_4^T(N_4+N_5)L_4 & L_5^T N_5 L_4 \\ L_1^T N_5 L_5 & L_2^T N_5 L_5 & L_3^T N_5 L_5 & L_4^T N_5 L_5 & L_5^T N_5 L_5 \end{bmatrix} \tag{4.47}$$

$$K_p = \begin{bmatrix} K_{p1} & 0 & 0 & 0 & 0 \\ 0 & K_{p2} & 0 & 0 & 0 \\ 0 & 0 & K_{p3} & 0 & 0 \\ 0 & 0 & 0 & K_{p4} & 0 \\ 0 & 0 & 0 & 0 & K_{p5} \end{bmatrix}, \qquad K_v = \begin{bmatrix} K_{v1} & 0 & 0 & 0 & 0 \\ 0 & K_{v2} & 0 & 0 & 0 \\ 0 & 0 & K_{v3} & 0 & 0 \\ 0 & 0 & 0 & K_{v4} & 0 \\ 0 & 0 & 0 & 0 & K_{v5} \end{bmatrix}. \tag{4.48}$$

Next we use the polynomial trajectory generation for modeling the actuator trajectory as a continuous cubic polynomial function. This allows for smooth differentiation of angular velocities, simplifying the calculation of angular positions. By employing cubic polynomial interpolation, we can derive both position and velocity profiles analytically and efficiently, ensuring precise control over actuator motion. For an actuator transitioning from an initial position θ_i to a final position θ_f over the time interval $[0, t_f]$, with boundary conditions $\dot{\theta}(0) = \dot{\theta}(t_f) = 0$, the cubic polynomial trajectory can be expressed as follows (Craig, 2009):

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{4.49}$$

Chapter 5

CONTROL

A control system is integral to managing the functions of a robot, facilitating the achievement of desired angles and motion trajectories. The controller interprets input commands and, utilizing feedback from various sensors or servos mitigates any discrepancies that may arise during practical applications. Numerous environmental factors can interfere with the robot's intended motion. An effectively designed robotic controller ensures precise tracking of specified joint angles, trajectories, or end-effector positions by minimizing the deviation between commanded and actual states, commonly referred to as tracking error. This objective is typically realized through feedback loops, which employ sensor measurements—such as joint encoders, inertial sensors, or visual systems—to calculate these deviations and generate corrective signals for the actuators to implement. Frequently employed control methodologies in robotics encompass classical techniques like proportional-integral-derivative (PID) control, as well as more advanced strategies including model predictive control (MPC), sliding mode control (SMC), and adaptive or robust controllers specifically designed to address uncertainties and external disturbances.

The foundation of robotic control lies in the mathematical representation of a robot's motion dynamics. A general form of the equation of motion for an n-DOF robotic system is given by,

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) = \tau \tag{5.1}$$

where $M(\theta)$ is mass matrix, $C(\theta, \dot{\theta})\dot{\theta}$ is coriolis and centrifugal matrix, $G(\theta)$ is gravitation force vector and τ is the input control torque. The following sections will

explore different control techniques in detail.

PD Control

The Proportional-Derivative (PD) controller is a classical and widely adopted control strategy in robotic systems due to its simplicity and effectiveness in stabilizing joint trajectories. The PD controller applies corrective torques based on the current position and velocity errors of each joint, without considering accumulated errors over time (as in integral action).

The PD control states that,

$$\tau = K_p(\theta_d - \theta) + K_v(\dot{\theta}_d - \dot{\theta}) \tag{5.2}$$

Substituting this in 5.1 gives,

$$\begin{aligned} M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + G(\theta) &= K_p(\theta_d - \theta) + K_v(\dot{\theta}_d - \dot{\theta}) \\ \implies \ddot{\theta} &= M^{-1}[K_p(\theta_d - \theta) + K_v(\dot{\theta}_d - \dot{\theta}) - C(\theta,\dot{\theta})\dot{\theta} - G(\theta)] \end{aligned} \tag{5.3}$$

Now similarly for each joint i, the PD control torque is defined as:

$$\tau_i = K_{p_i}\widetilde{\theta}_i + K_{v_i}\dot{\widetilde{\theta}}_i, \tag{5.4}$$

where

- $\widetilde{\theta}_i$ is the position error at joint i,
- $\dot{\widetilde{\theta}}_i$ is the velocity error at joint i,
- K_{p_i}, K_{v_i} are the proportional and velocity gains, respectively.

For the full system, the PD control torque vector is expressed as:

$$\tau = K_p x_1 + K_v x_2, \tag{5.5}$$

where

$$\begin{aligned} K_p &= \text{diag}(K_{p1}, K_{p2}, K_{p3}, K_{p4}, K_{p5}), \\ K_v &= \text{diag}(K_{v1}, K_{v2}, K_{v3}, K_{v4}, K_{v5}). \end{aligned} \tag{5.6}$$

Substituting the PD control law into the robot dynamics derived via motor algebra, the state-space representation becomes:

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} x_2 \\ -M(\theta)^{-1}\Big(K_p x_1 + K_v x_2 - B(\theta, \dot{\theta})\Big) \end{bmatrix} \tag{5.7}$$

where

- $M(\theta)$ is the inertia matrix derived from Clifford algebra,
- $B(\theta, \dot{\theta})$ represents the combined Coriolis, centrifugal, and gravitational effects,
- x_1 is the position error vector,
- x_2 is the velocity error vector.

PID Control

Proportional–Integral–Derivative (PID) control is a classical feedback strategy widely used in robotic systems due to its simplicity and effectiveness in minimizing trajectory-tracking errors. The PID control states that,

$$\tau = K_p(\theta_d - \theta) + K_v(\dot{\theta}_d - \dot{\theta}) + K_i \int_0^t (\theta_d - \theta) d\tau \tag{5.8}$$

Substituting this in 5.1 gives,

$$\begin{aligned} M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) &= K_p(\theta_d - \theta) + K_v(\dot{\theta}_d - \dot{\theta}) + K_i \int_0^t (\theta_d - \theta) d\tau \\ \implies \ddot{\theta} &= M^{-1}[K_p(\theta_d - \theta) + K_v(\dot{\theta}_d - \dot{\theta}) + K_i \int_0^t (\theta_d - \theta) d\tau - C(\theta, \dot{\theta})\dot{\theta} - G(\theta)] \end{aligned} \tag{5.9}$$

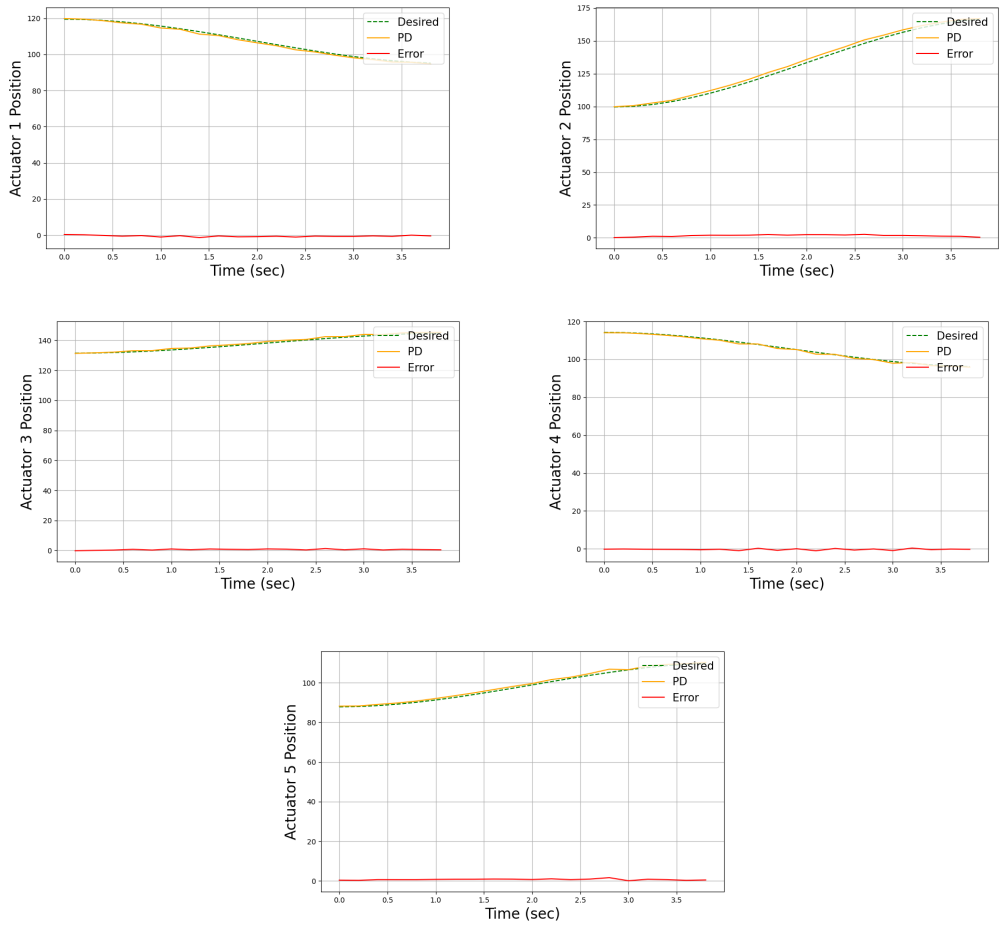


Figure 5.1: PD Implementation for $K_p = 0.7$ and $K_d = 0.009$

Similarly, integrating this control into the motor algebra-based dynamics of the robot, the PID control input u_{PID} is defined as:

$$u_{PID} = K_p x_1 + K_d x_2 + K_i x_3 \tag{5.10}$$

where

$$\begin{aligned} x_1 &= \left(\widetilde{\theta}_1, \widetilde{\theta}_2, \widetilde{\theta}_3, \widetilde{\theta}_4, \widetilde{\theta}_5\right)^T, \\ x_2 &= \left(\dot{\widetilde{\theta}}_1, \dot{\widetilde{\theta}}_2, \dot{\widetilde{\theta}}_3, \dot{\widetilde{\theta}}_4, \dot{\widetilde{\theta}}_5\right)^T, \\ x_3 &= \int_0^t x_1(\tau)\, d\tau. \end{aligned} \tag{5.11}$$

The proportional, derivative, and integral gain matrices are given by

$$K_p = \text{diag}(K_{p1}, K_{p2}, K_{p3}, K_{p4}, K_{p5}), \tag{5.12}$$

$$K_d = \text{diag}(K_{d1}, K_{d2}, K_{d3}, K_{d4}, K_{d5}), \tag{5.13}$$

$$K_i = \text{diag}(K_{i1}, K_{i2}, K_{i3}, K_{i4}, K_{i5}). \tag{5.14}$$

The closed-loop state-space dynamics of the system under PID control are

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= -M^{-1}\big(K_p x_1 + K_d x_2 + K_i x_3 \; - \; B(\theta, \dot{\theta})\big), \\ \dot{x}_3 &= x_1, \end{aligned} \tag{5.15}$$

where the inverse inertia matrix M^{-1} is defined as

$$M^{-1} \;=\; \begin{bmatrix} L_1^T(N_1 + N_2 + N_3 + N_4 + N_5)L_1 & \cdots \\ \vdots & \ddots \end{bmatrix}^{-1} \tag{5.16}$$

Here, $B(\theta, \dot{\theta})$ incorporates the Coriolis, centrifugal, and gravitational terms derived using the recursive Lie bracket and line screw structures.

The total torque vector τ applied to the system is given by

$$\tau \;=\; M(\theta)\,\ddot{\theta} \;+\; C(\theta, \dot{\theta})\,\dot{\theta} \;+\; G(\theta), \tag{5.17}$$

and the joint accelerations satisfy

$$\ddot{\theta} \;=\; -M^{-1}\Big(K_p\, x_1 \;+\; K_d\, x_2 \;+\; K_i\, x_3 \;-\; B(\theta, \dot{\theta})\Big). \tag{5.18}$$

This formulation embeds PID control directly into the motor algebra-based dynamics without altering the recursive structure of the inertial, Coriolis, or gravitational components. The inclusion of the integral term x_3 compensates for steady-state errors, enhancing trajectory tracking performance and robustness against persistent disturbances.

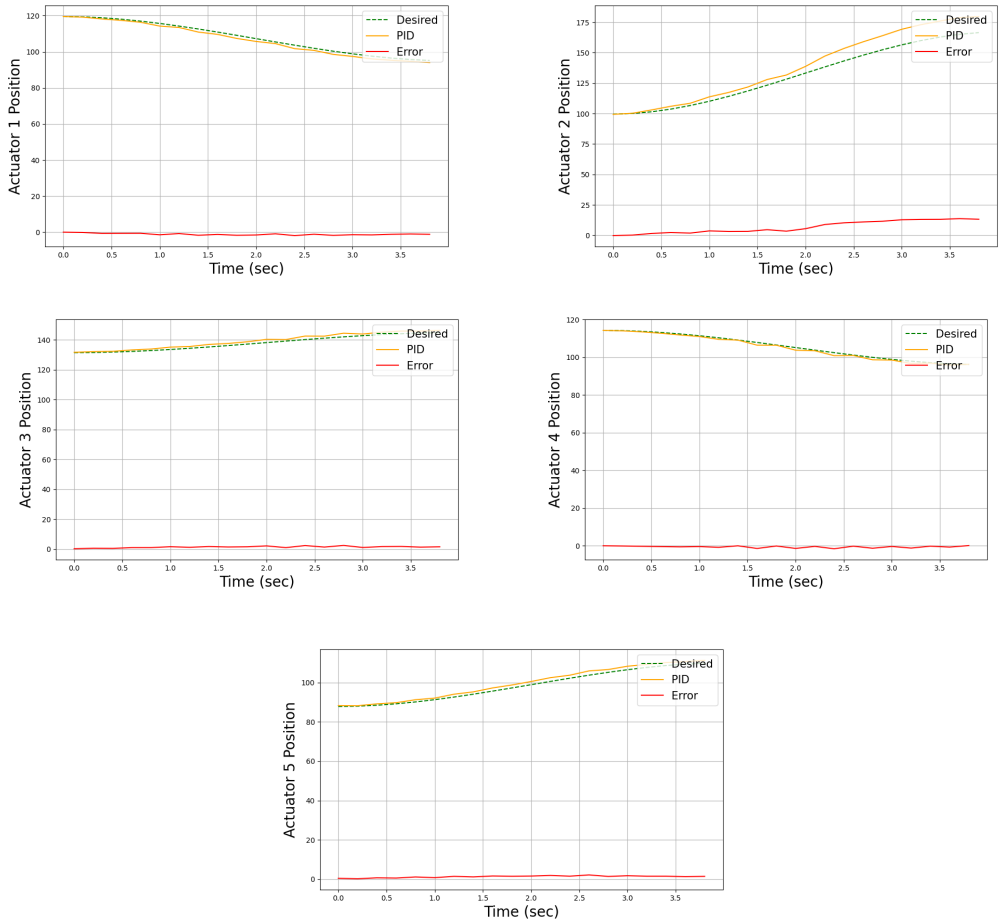


Figure 5.2: PID Implementation for $K_p = 0.7$, $K_i = 1.0$ and $K_d = 0.009$

Fractional PID Control

The Fractional Proportional-Integral-Derivative (FPID) controller extends the classical PID approach by incorporating fractional-order integral and derivative actions (Shah and Agashe, 2016). These additional degrees of freedom enhance control flexibility, robustness to disturbances, and performance in systems exhibiting long memory or non-local dynamics.

The FPID control states that,

$$\tau = K_p(\theta_d - \theta) + K_v D^{\mu}(\theta_d - \theta) + K_i D^{-\lambda}(\theta_d - \theta) \tag{5.19}$$

Substituting this in 5.1 gives,

$$M(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + G(\theta) = K_p(\theta_d - \theta) + K_v D^{\mu}(\theta_d - \theta) + K_i D^{-\lambda}(\theta_d - \theta)$$
$$\implies \ddot{\theta} = M^{-1}[K_p(\theta_d - \theta) + K_v D^{\mu}(\theta_d - \theta) + K_i D^{-\lambda}(\theta_d - \theta) - C(\theta,\dot{\theta})\dot{\theta} - G(\theta)] \tag{5.20}$$

Similarly for each joint i, the FPID control torque is defined as:

$$\tau_i = K_{p_i}\widetilde{\theta}_i + K_{v_i} D^{\mu_i}\widetilde{\theta}_i + K_{I_i} D^{-\lambda_i}\widetilde{\theta}_i \tag{5.21}$$

where

- $\widetilde{\theta}_i$ is the position error at joint i,
- $K_{p_i}, K_{v_i}, K_{I_i}$ are the proportional, derivative and integral gains, respectively,
- $D^{-\lambda_i}$ is the fractional integral of order λ_i,
- D^{μ_i} is the fractional derivative of order μ_i.

For the complete system, the FPID control torque vector is expressed as:

$$\tau = K_p\, x_1 + K_d\, D^{\mu}\, x_1 + K_I\, D^{-\lambda}\, x_1, \tag{5.22}$$

where

$$\begin{aligned} K_p &= \mathrm{diag}(K_{p1}, K_{p2}, K_{p3}, K_{p4}, K_{p5}), \\ K_d &= \mathrm{diag}(K_{d1}, K_{d2}, K_{d3}, K_{d4}, K_{d5}), \\ K_I &= \mathrm{diag}(K_{I1}, K_{I2}, K_{I3}, K_{I4}, K_{I5}), \\ D^{-\lambda} &= \mathrm{diag}(D^{-\lambda_1}, D^{-\lambda_2}, D^{-\lambda_3}, D^{-\lambda_4}, D^{-\lambda_5}), \\ D^{\mu} &= \mathrm{diag}(D^{\mu_1}, D^{\mu_2}, D^{\mu_3}, D^{\mu_4}, D^{\mu_5}). \end{aligned} \tag{5.23}$$

By substituting the FPID control law into the robot dynamics derived via motor algebra, the state-space representation becomes:

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} x_2 \\ -M(\theta)^{-1}\Big(K_p\, x_1 + K_d\, D^{\mu}\, x_1 + K_I\, D^{-\lambda}\, x_1 - B(\theta, \dot{\theta})\Big) \end{bmatrix}, \tag{5.24}$$

where:

- $M(\theta)$ is the inertia matrix derived from Clifford algebra,
- $B(\theta, \dot{\theta})$ represents the combined Coriolis, centrifugal, and gravitational effects,
- x_1 is the position error vector,
- x_2 is the velocity error vector.

Sliding Mode Control (SMC)

Sliding Mode Control (SMC) is applied to the motor algebra-based dynamics of the 5-DOF robot to achieve robust trajectory tracking under model uncertainties and external disturbances. Unlike the conventional PD control, SMC introduces a discontinuous control action designed to force the system states onto a predefined sliding surface and maintain them on this surface despite disturbances (Rahmani *et al.*, 2016).

The state variables are defined as

$$\begin{aligned} x_1 &= (\widetilde{\theta_1}, \widetilde{\theta_2}, \widetilde{\theta_3}, \widetilde{\theta_4}, \widetilde{\theta_5})^T \\ x_2 &= (\dot{\widetilde{\theta_1}}, \dot{\widetilde{\theta_2}}, \dot{\widetilde{\theta_3}}, \dot{\widetilde{\theta_4}}, \dot{\widetilde{\theta_5}})^T \end{aligned} \tag{5.25}$$

where x_1 represents the position tracking error and x_2 the velocity tracking error.

The sliding surface s is defined as

$$s = x_2 + \beta x_1 \tag{5.26}$$

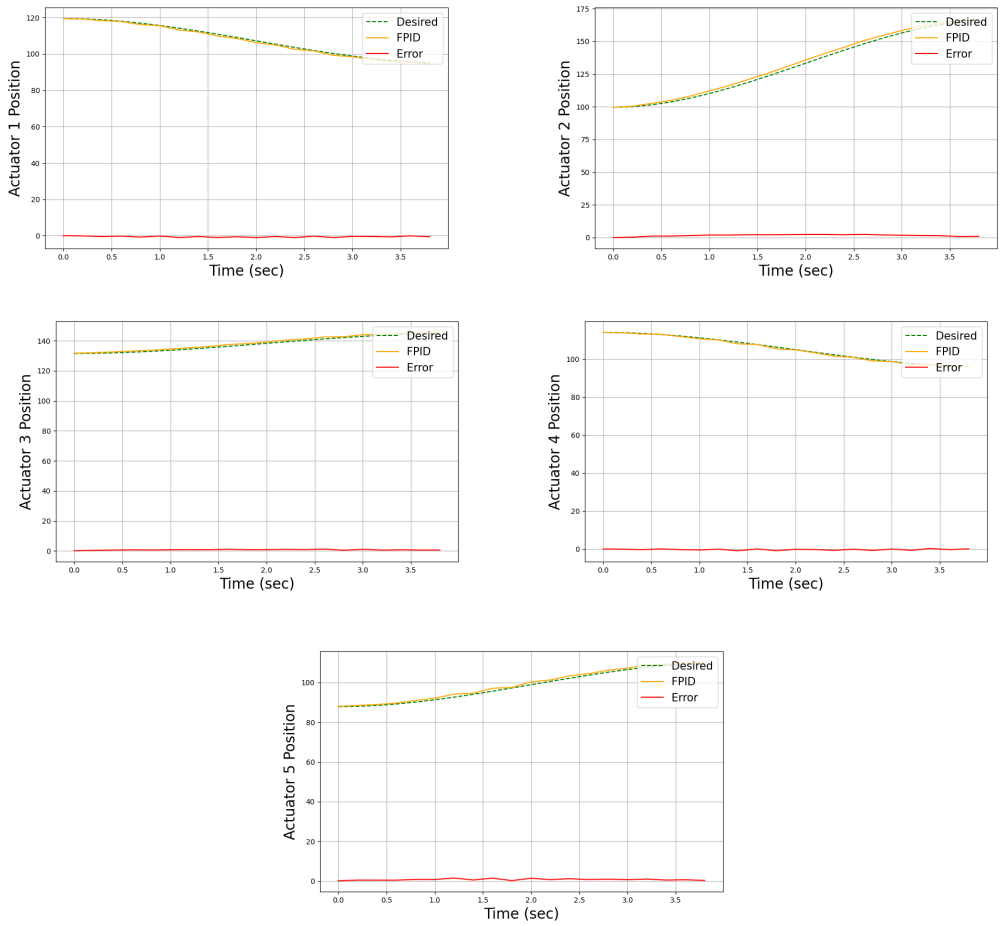


Figure 5.3: FPID Implementation for $\lambda = 0.8$, $\mu = 0.4$, $K_p = 0.7$, $K_i = 1.0$ and $K_d = 0.009$

where

$$\beta = \text{diag}(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5) \tag{5.27}$$

is a positive diagonal matrix determining the convergence speed of the errors toward the surface.

The control input τ replaces the PD control torque in the dynamics and is designed as the sum of an equivalent control τ_{eq} and a switching control τ_s,

$$\tau = \tau_{eq} + \tau_s \tag{5.28}$$

where:

$$\tau_{eq} = B(\dot{\theta}_1, \dot{\theta}_2, \dot{\theta}_3, \dot{\theta}_4, \dot{\theta}_5) \tag{5.29}$$

with B capturing the Coriolis, centrifugal, and gravitational effects derived from the motor algebra dynamics, and

$$\tau_s = K_s \cdot \text{sign}(s) \tag{5.30}$$

where

$$K_s = \text{diag}(K_{s1}, K_{s2}, K_{s3}, K_{s4}, K_{s5}) \tag{5.31}$$

is a diagonal matrix of positive switching gains selected to upper-bound the system uncertainties.

The closed-loop dynamics under SMC are expressed as

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} x_2 \\ -M^{-1}\left(B(\dot{\theta}) + K_s \cdot \text{sign}(x_2 + \beta x_1)\right) \end{bmatrix} \tag{5.32}$$

where the inertia matrix M is derived via motor algebra as

$$M = \begin{bmatrix} L_1^T(\sum N)L_1 & \dots & L_5^T N_5 L_1 \\ \vdots & \ddots & \vdots \\ L_1^T N_5 L_5 & \dots & L_5^T N_5 L_5 \end{bmatrix} \tag{5.33}$$

and the vector B encapsulates the dynamic effects.

To reduce the chattering effect caused by the discontinuity of the sign function, the switching term can be softened using a saturation function $\text{sat}(s/\xi)$,

$$\tau_s = K_s \cdot \text{sat}\left(\frac{s}{\xi}\right) \tag{5.34}$$

where ξ is a small positive boundary layer parameter (Rahmani *et al.*, 2016).

Chapter 6

EXPERIMENTATION & RESULTS

Building on the theoretical foundation and kinematic principles established in earlier chapters, this section explores the practical implementation of a 5-DOF worm robot and evaluates its performance in both virtual (Isaac Sim) and physical environments. The experiments aim to validate the inverse kinematics approach introduced earlier, demonstrating the effectiveness of two different inchworm-inspired gaits. Additionally, this chapter examines how environmental conditions, particularly variations in friction, affect the robot's locomotion. To enhance performance, various non-linear control strategies are evaluated for their effectiveness in achieving precise and stable joint positioning. Following the successful import of the URDF model into Isaac Sim as shown in Figure 6.1, we conducted a comprehensive validation process. We visually and numerically verified the robot's physical dimensions, joint positions, and collision meshes within the virtual environment against the original Solidworks model. To test the simulation's accuracy, we meticulously examined each joint's range of motion,

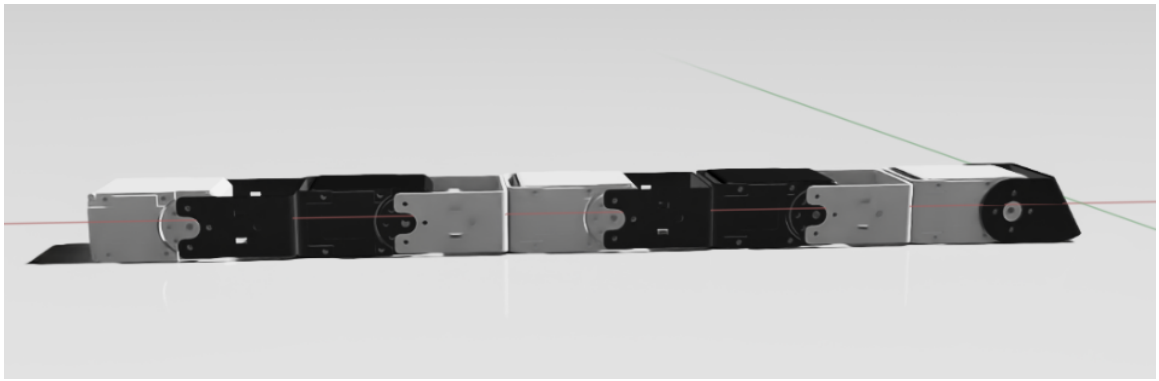


Figure 6.1: 5 DOF Worm Robot in Isaac Sim

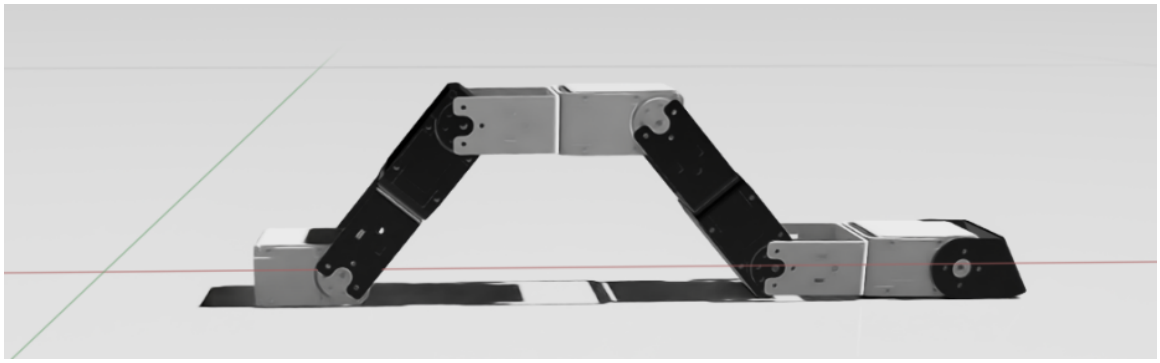


Figure 6.2: Gait 1

ensuring that Isaac Sim faithfully replicated the intended constraints and kinematics. Any discrepancies, including misalignments of pivot axes or incorrect joint limits, were promptly identified and rectified. This crucial step ensured that the simulations realistically mirrored real-world behavior.

Inverse Kinematics Validation in Isaac Sim

Building upon the principles of inverse kinematics (IK) outlined in previous chapters, two main inchworm gaits were developed in Isaac Sim to facilitate forward movement. These gaits consist of three sequential phases: anchoring, extending, and contracting. In the first phase (Figure 6.2), the head of the robot firmly anchors to the ground while the remaining segments are pulled forward. Next, this trapezoidal gait propagates toward the head in a wave-like motion as shown in Figure 6.3. Finally, the tail anchors itself, allowing the entire robot to be propelled forward by a specified distance. These gaits employ symmetrical joint angle distributions, which are computed automatically by the IK algorithm. This method simplifies the robot's control, requiring only high-level input commands to indicate the desired displacement. Table 6.2 presents various joint values for a precise simulation environment that accurately reflects the robot's mass, joint, and surface characteristics.

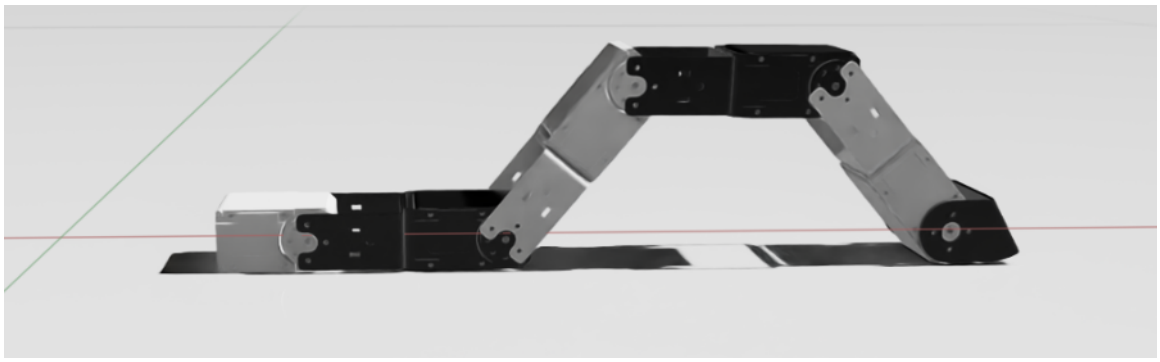


Figure 6.3: Gait 2

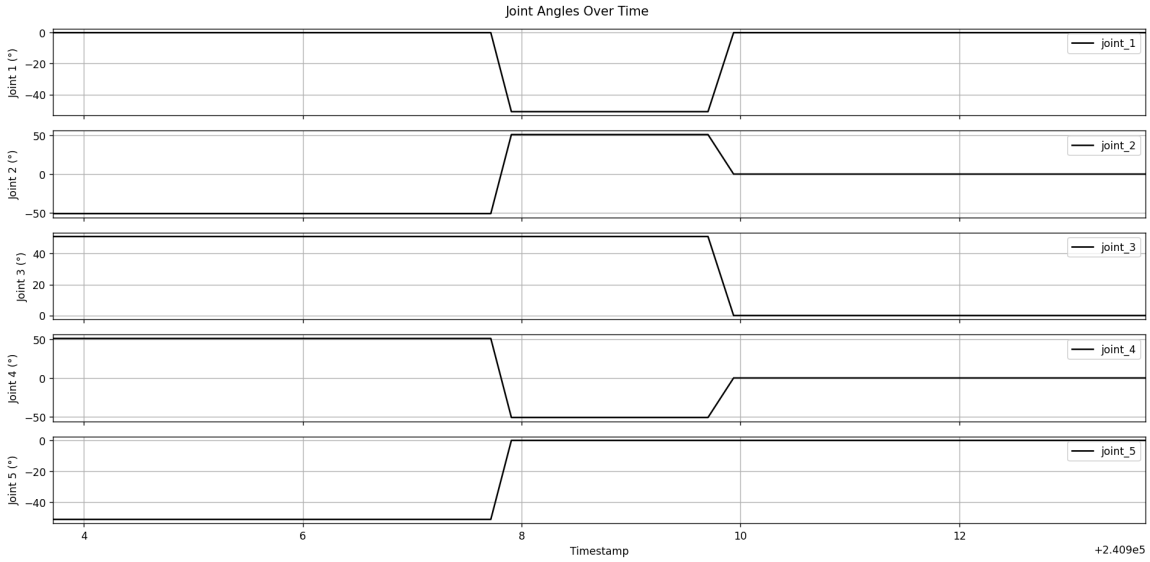


Figure 6.4: Joint Angles for $\eta = 2$ cm

Range Sensor Integration in Isaac Sim

To enhance environmental awareness and enable autonomous navigation, an ultrasonic range sensor was mounted on the front section of the worm robot. This sensor streams azimuth and zenith values based on the location of obstacles within the Isaac Sim environment. It was secured in a way that minimizes additional inertia and interference with the robot's movement on its base link. This integration improves the digital twin's ability to realistically simulate robotic tasks, extending beyond basic

Table 6.1: Comparison of Desired and Actual Displacement Values in Isaac Sim

$\eta_{desired}$ [cm]	θ [°]	η_{actual} [cm]	Error e [cm]
0.2	9.29	0.219	0.019
0.8	18.66	0.786	−0.014
1.0	20.87	1.008	0.008
2.0	29.68	1.977	−0.023
3.0	36.57	3.087	0.087
4.0	42.48	4.000	0.000
5.0	47.78	5.005	0.005
10.0	69.89	10.142	0.142
12.0	77.73	11.989	−0.011
15.0	89.11	15.016	0.016
20.0	108.22	20.314	0.314

locomotion to more complex interaction scenarios. The goal is to continuously stream data from a real ultrasonic or other suitable range sensor into this environment for effective mapping. As illustrated in Figure 6.18, the robot approaches the obstacle, and the initial white envelope of sensor input continuously changes color in the RGB spectrum as the obstacle gets nearer.

MATLAB implementation

To quantitatively evaluate the performance of the motor-algebra controller against three common baseline algorithms (PD, classical PID, and fractional-order PID) as well as a sliding-mode controller (SMC) as shown in Chapter 5, we developed a MATLAB script that simulates a simplified five-degree-of-freedom serial manipula-

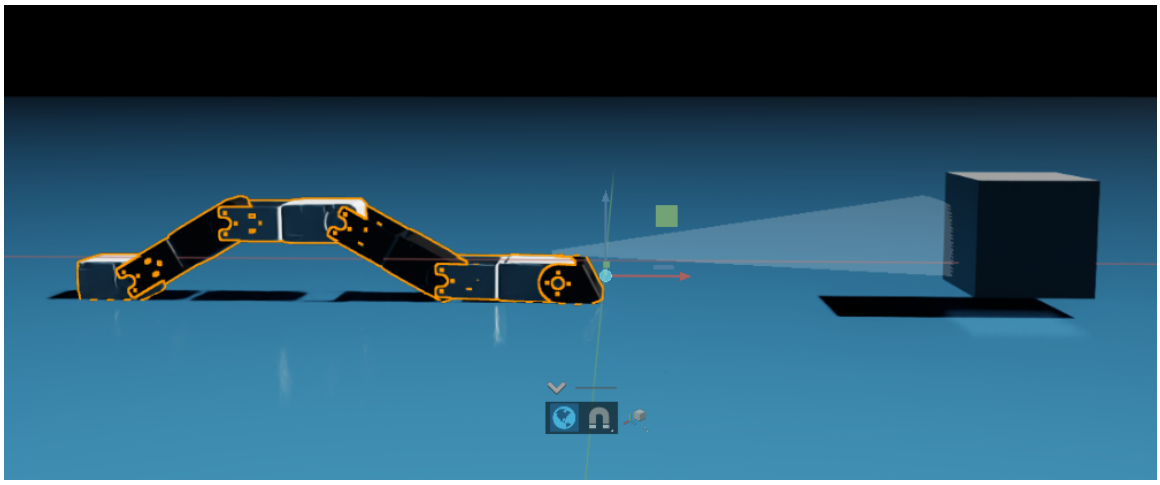


Figure 6.5: Range Sensor Mounted on Robot

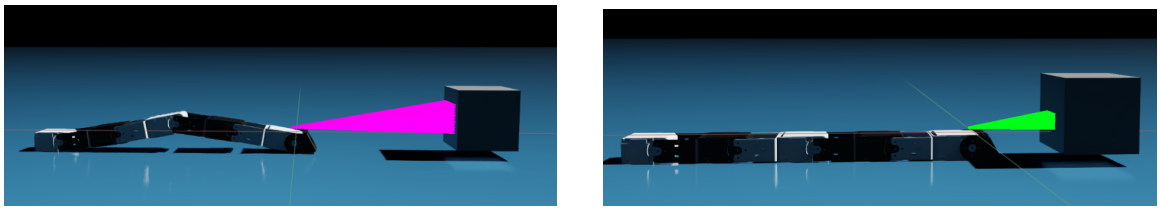


Figure 6.6: Locomotion Timeline with the Sensor

tor model (Bayro-Corrochano, 2020). For validation and demonstration purposes, we implemented a locomotion gait characterized by a joint angle of $\theta = 45°$. The physical parameters of the system, including the masses and lengths of the individual links, were assigned based on the specifications outlined in Chapter 2. Desired joint positions were set to a constant vector of $\theta_d = [0,\ 45,\ -45,\ -45,\ 45]^\mathsf{T}$ converted internally to radians. Each controller was tuned to comparable bandwidth: PD gains of $K_p = 12$, $K_v = 4$; PID gains of $K_p = 0.4$, $K_d = 0.002$, $K_i = 0.05$; FPID $K_p = 1.7$, $K_d = 0.002$, $K_i = 0.05$, $\lambda = 0.9$ and $\mu = 0.2$; and SMC parameters $\beta = 5$, $K_s = 0.1$, $\varepsilon = 10^{-2}$. Joint friction was modeled as viscous damping (0.5 Nms/rad per joint), and Coriolis forces were neglected. The results are shown in Figures 6.7 and 6.8.

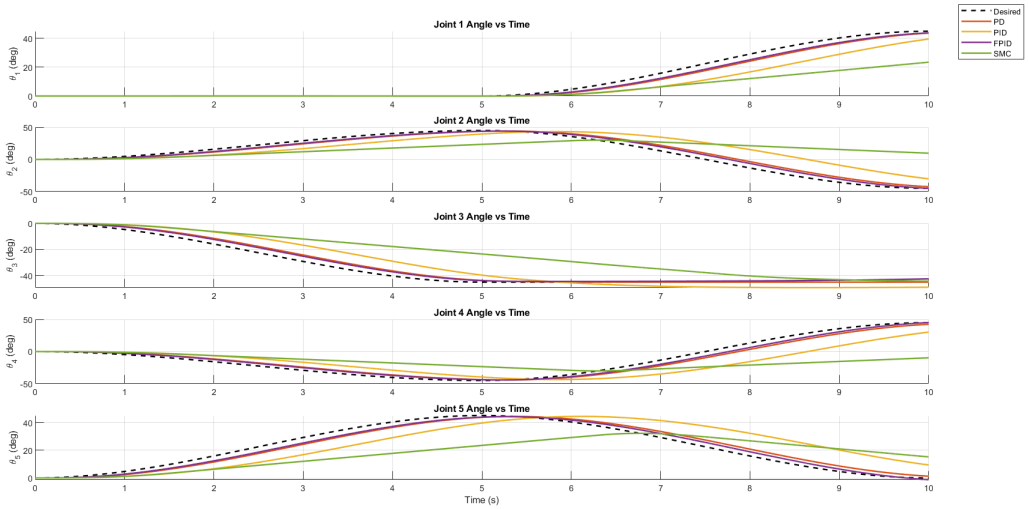


Figure 6.7: Joint Angles Based Comparison of Controller Performance

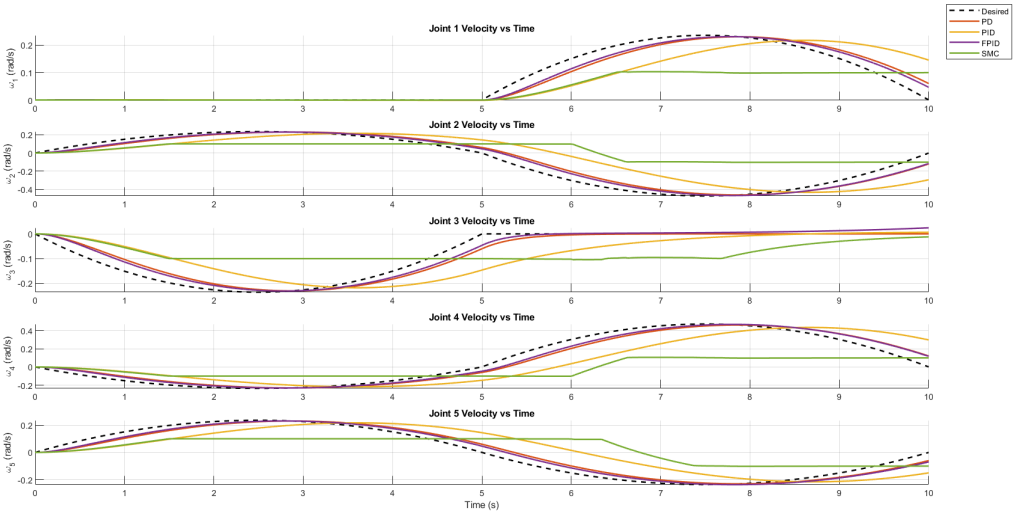


Figure 6.8: Joint Velocities Based Comparison of Controller Performance

Digital Twin Implementation

To evaluate robot performance comprehensively, extensive testing was conducted using a digital twin on various surfaces, each with unique frictional properties. Four distinct environments were chosen: a wooden surface with moderate friction, a rugged fabric that increased frictional resistance, a smooth fabric that reduced friction and irregularities, and paper, which demonstrated moderate friction conditions. The control systems discussed in previous chapters were applied to the 5-DOF robot to evaluate and validate controller performance. In the initial phase, the robot was kept stationary while the joint angles were contracted, serving as a baseline check for the controllers. For instance, with PD control parameters set at Kp = 0.7 and Kd = 0.009, minimal errors were observed, as evidenced by the corresponding error graph (refer to Figure 5.1). Similar evaluations were conducted for the PID and FPID controllers as shown in Figures 5.2 and 5.3 respectively, and their error graphs further confirmed stable performance. Experimentally, the home or zero positions were recorded as 119.52° for servo 1, 99.6° for servo 2, 131.52° for servo 3, 114.24° for servo 4, and 87.84° for servo 5. Following this initial validation, the inverse kinematic gait-based approach described earlier was integrated with these controllers. To smooth the gait movement, servo easing (Arduino, 2025) was initially implemented; however, the above-mentioned control scheme was subsequently refined by directly applying the PD, PID, and FPID controllers. Their performance was benchmarked using the home position as the reference point (Refer Figures 6.9, 6.10, 6.11, 6.12, 6.13 and 6.14). The results indicated that on the moderately frictional wooden surface, the robot performed tasks predictably and efficiently, closely aligning with theoretical predictions and exhibiting minimal slippage. In contrast, the higher friction fabric improved the robot's anchoring stability, while the smooth fabric mitigated some

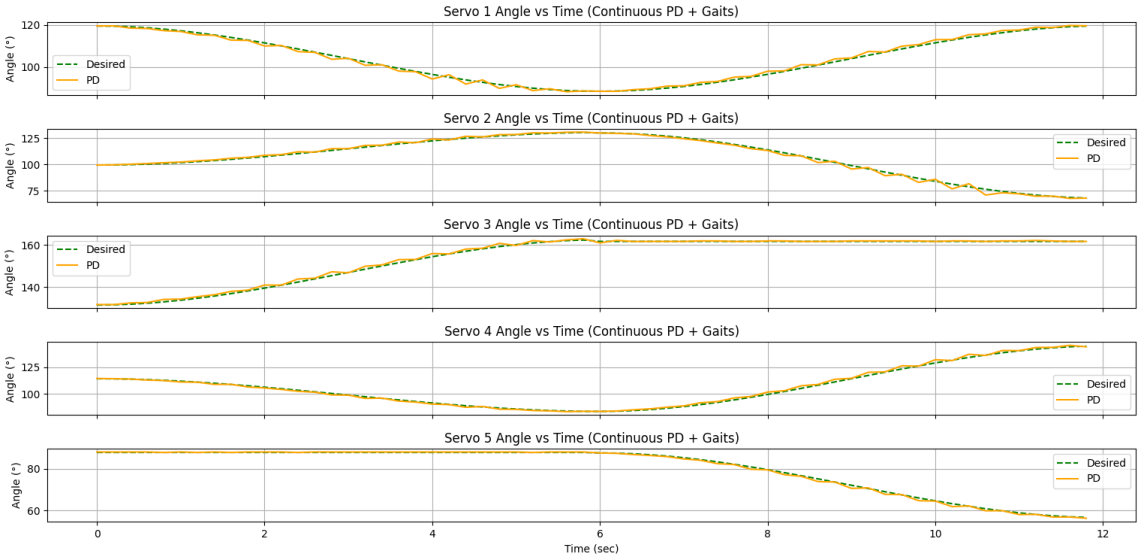


Figure 6.9: PD Controller With Inverse Kinematic Gaits (Trajectory)

challenges. The paper surface provided moderate friction and limited slippage.

To develop the final digital twin that implements these approaches on different surfaces, we modeled the surface properties in Isaac Sim for roughly estimated combinations such as hardwood-steel, paper-steel, rugged cloth/canvas-steel, and smooth fabric-steel. Both static and dynamic friction coefficients were closely estimated and average combination mode was applied in the environment. These simulations were carefully calibrated to reflect the frictional interactions expected in real-world scenarios. Experimental data, presented both graphically and in tabular form, substantiated the simulation findings by clearly demonstrating the direct relationship between surface friction and robot performance.

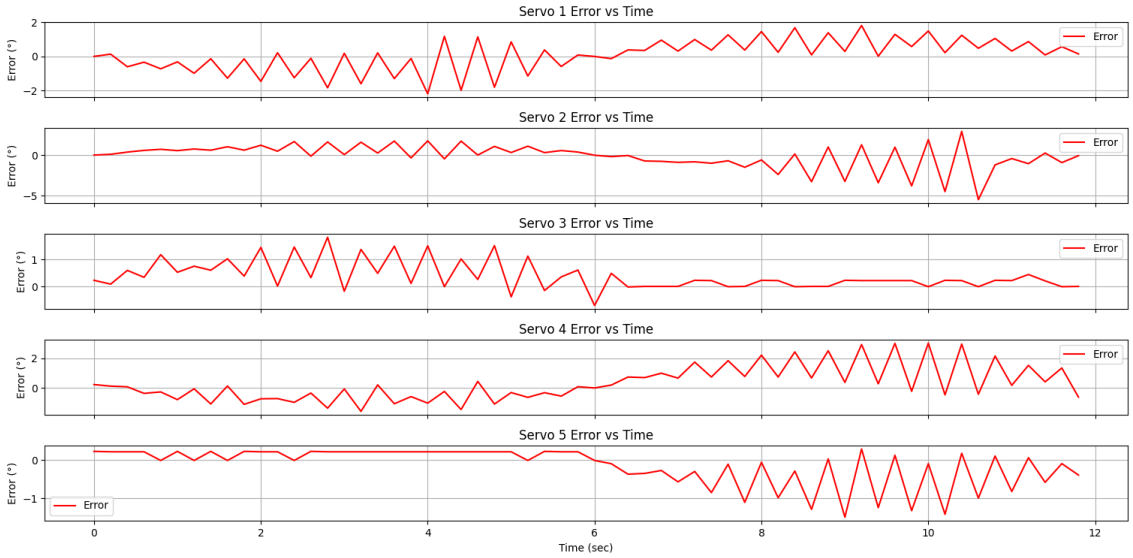


Figure 6.10: PD Controller With Inverse Kinematic Gaits (Error)

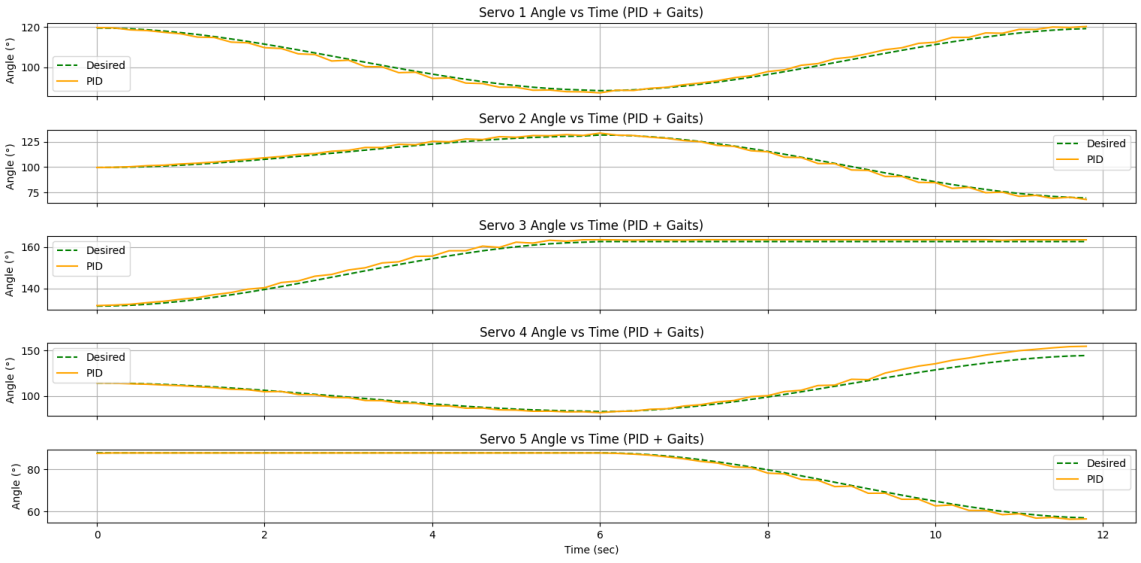


Figure 6.11: PID Controller With Inverse Kinematic Gaits (Trajectory)

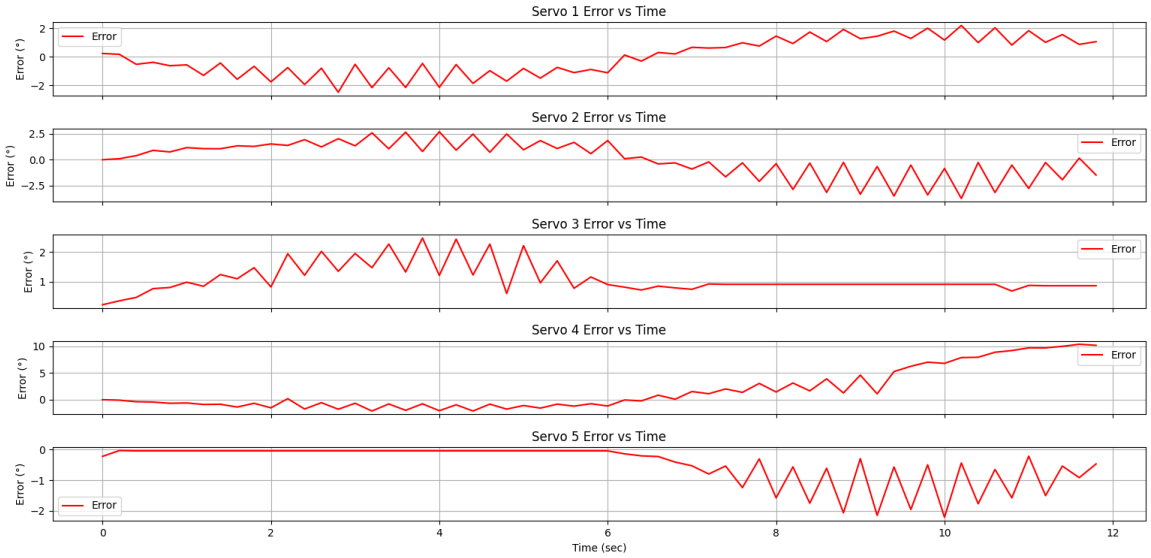


Figure 6.12: PID Controller With Inverse Kinematic Gaits (Error)

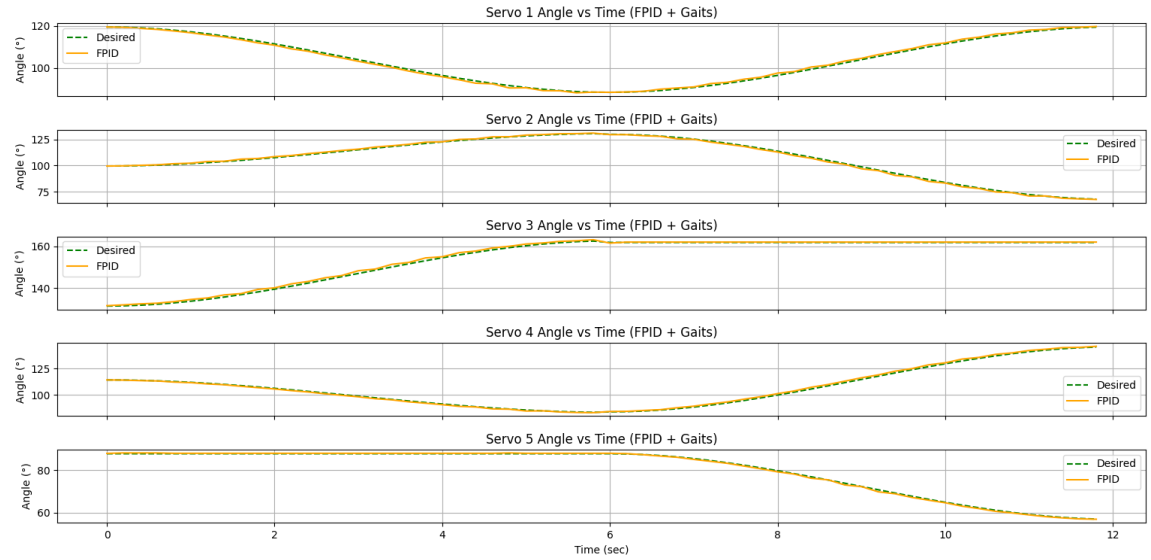


Figure 6.13: FPID Controller With Inverse Kinematic Gaits (Trajectory)

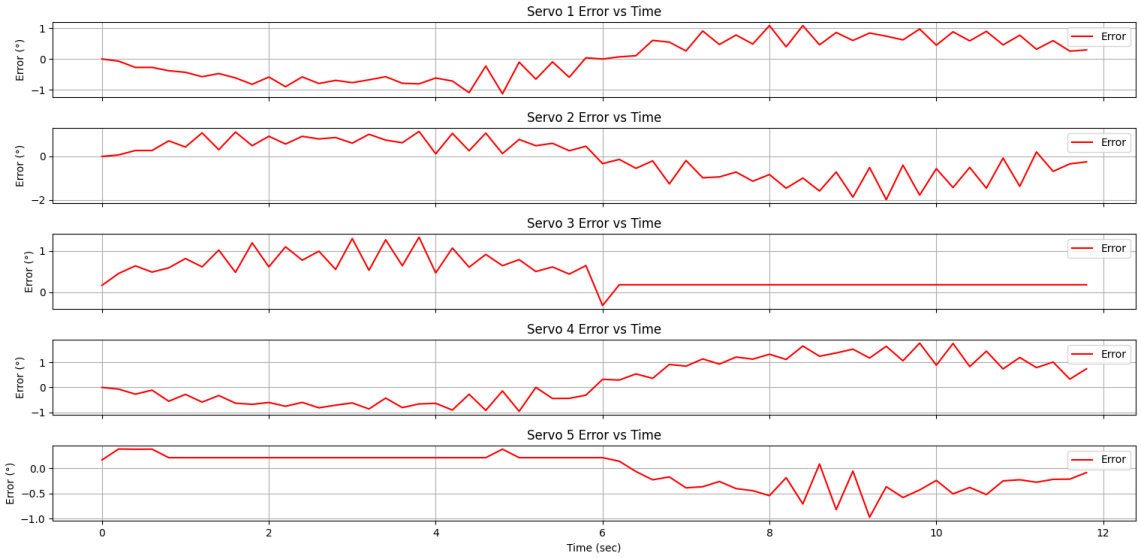
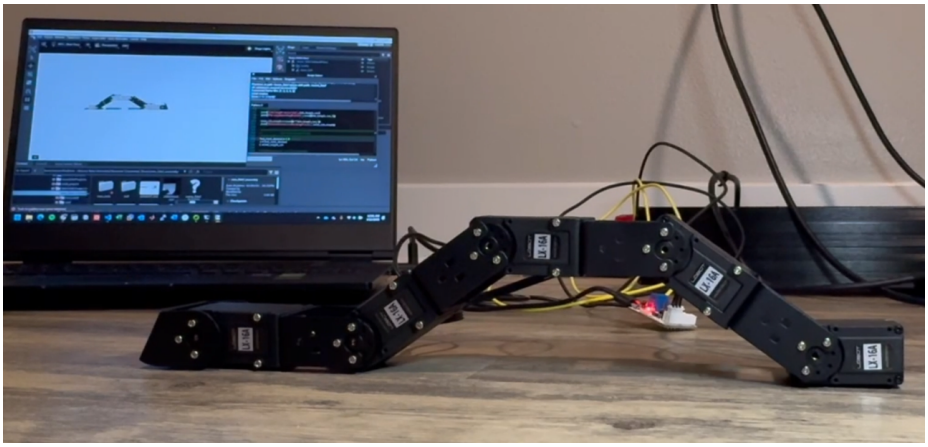


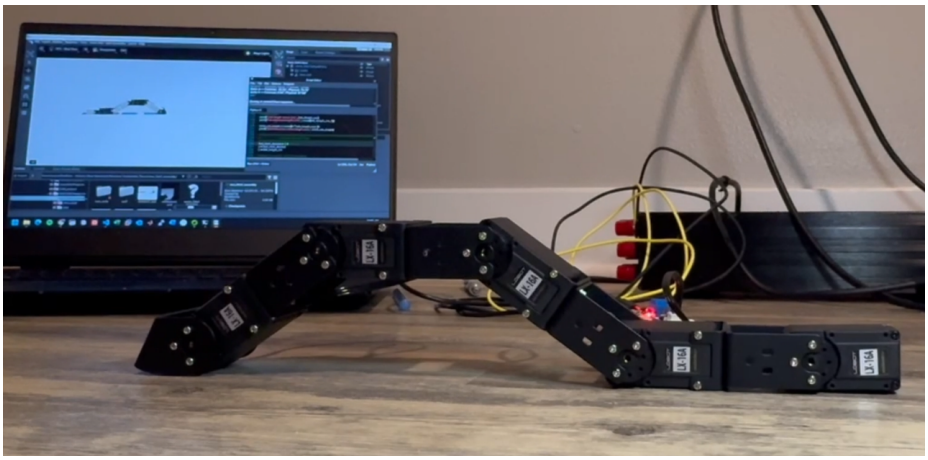
Figure 6.14: FPID Controller With Inverse Kinematic Gaits (Error)

Table 6.2: Comparison of Desired and Actual Displacement Values in Isaac Sim and Real World (cms)

Material combination	μ_s	μ_k	actual$_{IsaacSim}$	$e_{\text{Isaac Sim}}$	actual$_{real}$	e_{real}
Steel-Wood	0.6	0.5	1.6	-0.4	4	2
Steel-Paper	0.4	0.3	2.4	0.4	9	5
Steel-Rugged Fabric	0.5	0.4	2.33	0.33	5	3
Steel-Smooth Fabric	0.3	0.15	2.35	0.35	3.7	1.7



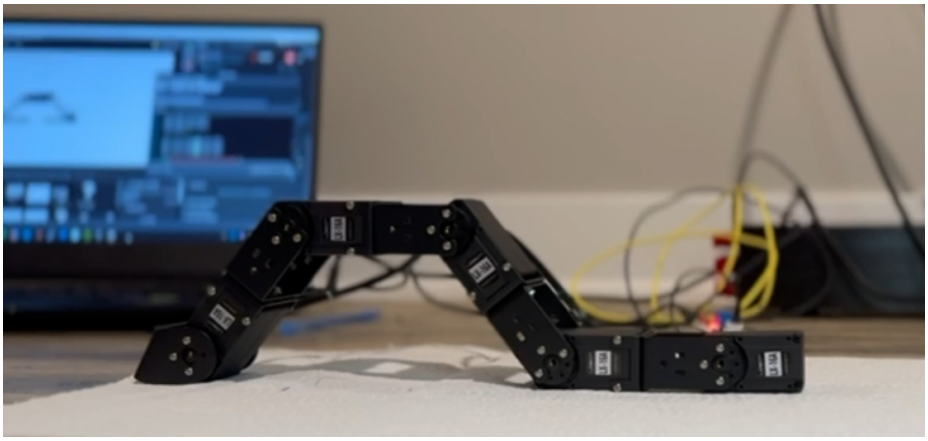
(a) Gait 1



(b) Gait 2

Figure 6.15: Steel on Wood Implementation ($\mu_s = 0.6, \mu_k = 0.5$)

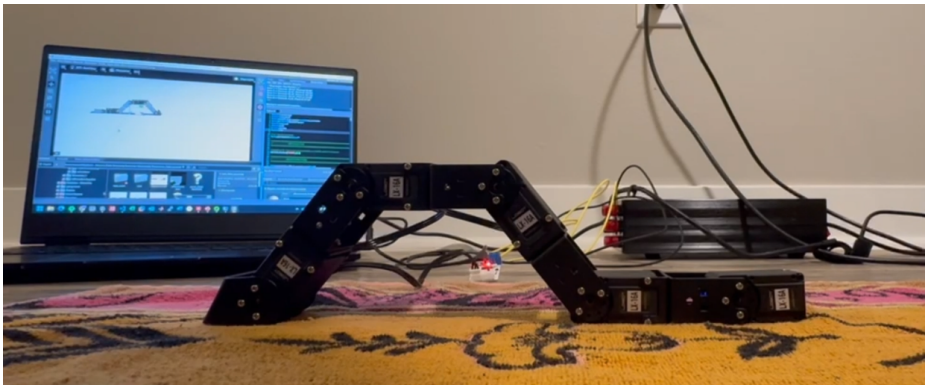
(a) Gait 1



(b) Gait 2

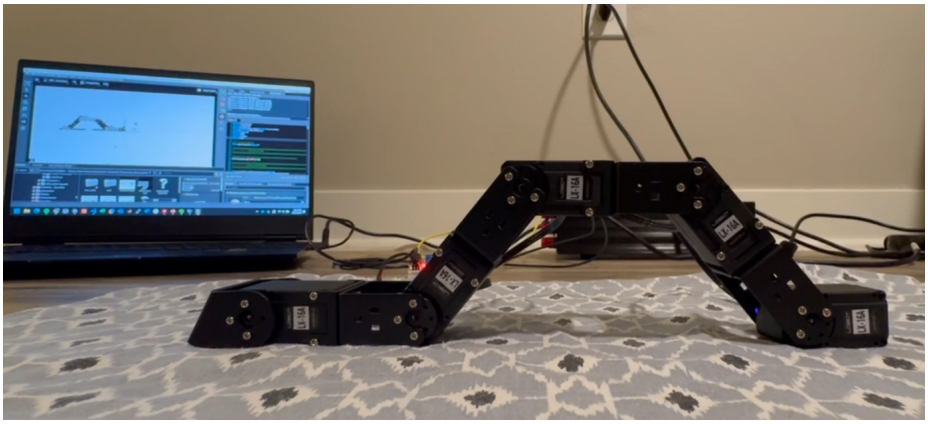
Figure 6.16: Steel on Paper Implementation ($\mu_s = 0.4, \mu_k = 0.3$)

(a) Gait 1

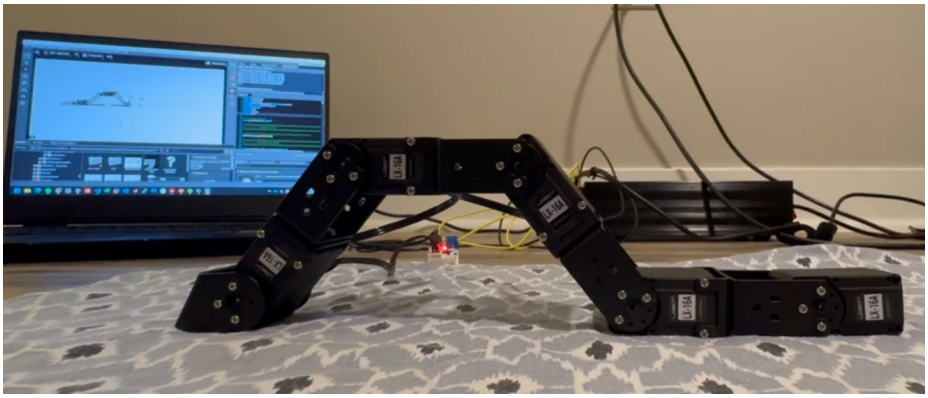


(b) Gait 2

Figure 6.17: Steel on Rugged Fabric Implementation ($\mu_s = 0.5, \mu_k = 0.4$)



(a) Gait 1



(b) Gait 2

Figure 6.18: Steel on Smooth Fabric Implementation ($\mu_s = 0.3, \mu_k = 0.2$)

Chapter 7

CONCLUSION & FUTURE WORK

This thesis has presented the design, modeling, control, and validation of a bio-inspired inchworm robot capable of executing metachronal gait patterns using a five-degree-of-freedom architecture. Drawing from the principles of bio-inspired robotics and the locomotion strategies observed in nature, the proposed system demonstrates how compliant and adaptive motion can be replicated in robotic platforms to navigate complex environments. To achieve this, geometric algebra-based formulations were applied to develop the kinematic and dynamic models of the robot, providing an efficient and unified mathematical framework for analyzing its motion. These models laid the foundation for implementing a variety of control strategies, including PD, PID, fractional PID, and sliding mode control, which were systematically evaluated for their performance in managing the robot's nonlinear dynamics.

The future work aims to develop a system in which robots are controlled using Large Language Models (LLMs). Current control strategies for worm robots rely heavily on precise system modeling and predefined trajectory information, which significantly limits their effectiveness in dynamic environments. While traditional controllers, such as PID (Proportional-Integral-Derivative) and MPC (Model Predictive Control), perform well in structured settings, they often struggle to adapt in real-time to complex and unknown terrains. To overcome these challenges, we propose an LLM-based framework that allows worm robots to dynamically adjust their locomotion based on task demands and environmental changes. Furthermore, to improve the robot's maneuverability and efficiency, we plan to incorporate an additional degree of freedom (DoF) to facilitate sidewinding locomotion. Sidewinding

is a biomechanically inspired movement strategy commonly seen in snakes, enabling efficient traversal across challenging terrains such as loose sand, rubble, and uneven surfaces.

Finally, this thesis shows how a digital twin of the inchworm robot was developed using Nvidia Isaacsim, which allows real-time simulation, control validation, and data streaming between the physical and virtual robots. This integration of digital twin technology not only improved the design and testing phases but also enabled predictive insights into the robot's behavior across diverse terrains. Through experimental validation on various surfaces and comparison with simulation data, the effectiveness of the metachronal gait locomotion strategies was successfully demonstrated.

REFERENCES

Ab Rashid, M. Z., M. F. Mohd Yakub, S. A. Zaki bin Shaikh Salim, N. Mamat, S. M. Syed Mohd Putra and S. A. Roslan, "Modeling of the in-pipe inspection robot: A comprehensive review", Ocean Engineering **203**, 107206, URL https://www.sciencedirect.com/science/article/pii/S0029801820302638 (2020).

Arduino, "Servoeasing", URL https://docs.arduino.cc/libraries/servoeasing/, accessed: 2024-03-05 (2025).

Balla, Wahyu, M. Budiarto, B. Syahputra, Satria, D. Nurpriyanto, U. Dwiyanto, Yusmaniar and Oktiawati, "Design and analysis of bio-inspired robotic systems for search and rescue operations", The Journal of Academic Science **1**, 4, URL http://thejoas.com/index.php/thejoas/article/view/47 (2024).

Bayro-Corrochano, E., *Geometric Algebra Applications Vol. II: Robot Modelling and Control* (2020).

Bi, Z., Q. Zhou and H. Fang, "A worm-snake-inspired metameric robot for multimodal locomotion: Design, modeling, and unified gait control", International Journal of Mechanical Sciences **254**, 108436, URL https://www.sciencedirect.com/science/article/pii/S0020740323003387 (2023).

Biller, B., X. Jiang, J. Yi, P. Venditti and S. Biller, "Simulation: the critical technology in digital twin development", in "2022 Winter Simulation Conference (WSC)", pp. 1340–1355 (2022).

Bogue, R., "Bioinspired designs impart robots with unique capabilities", Industrial Robot: the international journal of robotics research and application **46**, 5, 561–567, URL https://doi.org/10.1108/IR-05-2019-0100, publisher: Emerald Publishing Limited (2019).

Boyer, F., C. Stefanini, F. Ruffier and S. Viollet, "Special issue featuring selected papers from the international workshop on bio-inspired robots (nantes, france, 6–8 april 2011)", Bioinspiration Biomimetics **7**, 2, 020201, URL https://dx.doi.org/10.1088/1748-3182/7/2/020201 (2012).

Chirikjian, G. S., *Snakelike and Continuum Robots: A Review of Reviews*, pp. 1–14 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2020), URL https://doi.org/10.1007/978-3-642-41610-1_147-1.

Choset, H. M., D. Hull, J. E. Luntz, E. Shammas, T. Rached and C. C. Dent, "Search and rescue with serpentine robots", in "Unmanned Ground Vehicle Technology II", edited by G. R. Gerhart, R. W. Gunderson and C. M. Shoemaker, vol. 4024, pp. 283 – 291, International Society for Optics and Photonics (SPIE, 2000), URL https://doi.org/10.1117/12.391639.

Craig, J. J., *Introduction to robotics: mechanics and control, 3/E* (Pearson Education India, 2009).

Crespi, N., A. T. Drobot and R. Minerva, *The Digital Twin: What and Why?*, pp. 3–20 (Springer International Publishing, Cham, 2023), URL https://doi.org/10.1007/978-3-031-21343-4_1.

Davies, E., A. Garlow, S. Farzan, J. Rogers and A.-P. Hu, "Tarzan: Design, prototyping, and testing of a wire-borne brachiating robot", in "2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)", pp. 7609–7614 (2018).

Dineva, K. and T. Atanasova, "Modelling and simulation of cloud-based digital twins in smart farming", Proceedings of the International Multidisciplinary Scientific GeoConference SGEM **22**, 6.2, 243–250, URL https://epslibrary.at/sgem_jresearch_publication_view.php?page=view&editid1=8926 (2022).

Dudek, G., M. Jenkin, C. Prahacs, A. Hogue, J. Sattar, P. Giguere, A. German, H. Liu, S. Saunderson, A. Ripsman, S. Simhon, L.-A. Torres, E. Milios, P. Zhang and I. Rekletis, "A visually guided swimming robot", in "2005 IEEE/RSJ International Conference on Intelligent Robots and Systems", pp. 3604–3609 (2005).

Fjerdingen, S. A., P. Liljebäck and A. A. Transeth, "A snake-like robot for internal inspection of complex pipe structures (piko)", in "2009 IEEE/RSJ International Conference on Intelligent Robots and Systems", pp. 5665–5671 (2009).

Fukuda, T., F. Chen and Q. Shi, "Special feature on bio-inspired robotics", Applied Sciences **8**, 5, URL https://www.mdpi.com/2076-3417/8/5/817 (2018).

Fukuoka, Y. and H. Kimura, "Dynamic locomotion of a biomorphic quadruped 'tekken' robot using various gaits: Walk, trot, free-gait and bound", Applied Bionics and Biomechanics **6**, 1, 743713, URL https://onlinelibrary.wiley.com/doi/abs/10.1080/11762320902734208 (2009).

Ghanbari, A. and S. Noorani, "Optimal trajectory planning for design of a crawling gait in a robot using genetic algorithm", International Journal of Advanced Robotic Systems **8**, 1, 6, URL https://doi.org/10.5772/10526 (2011).

Ghanbari, A., A. Rostami, S. M. R. S. Noorani and M. M. S. Fakhrabadi, "Modeling and simulation of inchworm mode locomotion", in "Intelligent Robotics and Applications", edited by C. Xiong, Y. Huang, Y. Xiong and H. Liu, pp. 617–624 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2008).

Gravish, N. and G. V. Lauder, "Robotics-inspired biology", Journal of Experimental Biology **221**, 7, jeb138438, URL https://doi.org/10.1242/jeb.138438 (2018).

Gray, J., "The mechanism of locomotion in snakes", Journal of Experimental Biology **23**, 2, 101–120, URL https://doi.org/10.1242/jeb.23.2.101 (1946).

Hirose, S. and H. Yamada, "Snake-like robots [tutorial]", IEEE Robotics Automation Magazine **16**, 1, 88–98 (2009).

Hu, T., X. Lu and J. Liu, "Inchworm-like soft robot with multimodal locomotion using an acrylic stick-constrained dielectric elastomer actuator", Advanced Intelligent Systems **5**, 2, 2200209, URL https://advanced.onlinelibrary.wiley.com/doi/abs/10.1002/aisy.202200209 (2023).

Hunt, A., *A Biologically Inspired Robot for Assistance in Urban Search and Rescue*, Ph.D. thesis, Case Western Reserve University, URL https://etd.ohiolink.edu/acprod/odb_etd/etd/r/1501/10?clear=10&p10_accession_num=case1270137669 (2010).

Jamoussi, A., "Robotic nde: A new solution for in-line pipe inspection", Tech. rep. (2005).

Kenett, R. S. and J. Bortman, "The digital twin in industry 4.0: A wide-angle perspective", Quality and Reliability Engineering International **38**, 3, 1357–1366, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/qre.2948 (2022).

Kulik, T., Z. Kazemi and P. G. Larsen, *Security and Privacy-related Issues in a Digital Twin Context*, pp. 313–344 (Springer International Publishing, Cham, 2024), URL https://doi.org/10.1007/978-3-031-66719-0_13.

Liljebäck, P., K. Y. Pettersen, Stavdahl and J. T. Gravdahl, "A simplified model of planar snake robot locomotion", in "2010 IEEE/RSJ International Conference on Intelligent Robots and Systems", pp. 2868–2875 (2010).

Lipták, T., I. Virgala, P. Frankovský, P. Šarga, A. Gmiterko and L. Baločková, "A geometric approach to modeling of four- and five-link planar snake-like robot", International Journal of Advanced Robotic Systems **13**, 5, 1729881416663714, URL https://doi.org/10.1177/1729881416663714 (2016).

Lock, R. J., S. C. Burgess and R. Vaidyanathan, "Multi-modal locomotion: from animal to application", Bioinspiration Biomimetics **9**, 1, 011001, URL https://dx.doi.org/10.1088/1748-3182/9/1/011001 (2013).

Margheri, L., C. Laschi and B. Mazzolai, "Soft robotic arm inspired by the octopus: I. from biological functions to artificial requirements", Bioinspiration Biomimetics **7**, 2, 025004, URL https://dx.doi.org/10.1088/1748-3182/7/2/025004 (2012).

McKinsey&Company, "What is digital-twin technology?", URL https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-digital-twin-technology (2024).

Medrano-Hermosillo, J. A., R. Lozoya-Ponce, J. Ramírez-Quintana and R. Baray-Arana, "Forward kinematics analysis of 6-dof articulated robot using screw theory and geometric algebra", in "2022 XXIV Robotics Mexican Congress (COMRob)", pp. 1–6 (2022).

Mehringer, A., A. Kandhari, H. Chiel, R. Quinn and K. Daltorio, "An integrated compliant fabric skin softens, lightens, and simplifies a mesh robot", in "Biomimetic

and Biohybrid Systems", edited by M. Mangan, M. Cutkosky, A. Mura, P. F. Verschure, T. Prescott and N. Lepora, pp. 315–327 (Springer International Publishing, Cham, 2017).

Ngadi, H., A. Bounceur, M. Hammoudeh, S. Ouchani, M. Bezoui, R. Euler and A. Laouid, "From simulation to digital twins, the case of internet of things research and tools", in "Proceedings of the 6th International Conference on Future Networks & Distributed Systems", ICFNDS '22, p. 657–666 (Association for Computing Machinery, New York, NY, USA, 2023), URL https://doi.org/10.1145/3584202.3584302.

Niu, H., R. Feng, Y. Xie, B. Jiang, Y. Sheng, Y. Yu, H. Baoyin and X. Zeng, "Magworm: A biomimetic magnet embedded worm-like soft robot", Soft Robotics **8**, 5, 507–518, URL https://doi.org/10.1089/soro.2019.0167, pMID: 32822273 (2021).

Noorani, M.-R. S., A. Ghanbari and S. Aghli, "Design and fabrication of a worm robot prototype", in "2015 3rd RSI International Conference on Robotics and Mechatronics (ICROM)", pp. 073–078 (2015).

NVIDIA Corporation, "Urdf importer", URL https://docs.omniverse.nvidia.com/isaacsim/latest/features/environment_setup/ext_omni_isaac_urdf.html, accessed: 2024-03-10 (2024).

NVIDIA Corporation, *Bringing in Autonomous Systems*, URL https://docs.omniverse.nvidia.com/digital-twins/latest/auto-sys.html, accessed: 2025-03-03 (2025).

Pettersen, K. Y., "Snake robots", Annual Reviews in Control **44**, 19–44, URL https://www.sciencedirect.com/science/article/pii/S1367578817301050 (2017).

Playter, R., M. Buehler and M. Raibert, "BigDog", in "Unmanned Systems Technology VIII", edited by G. R. Gerhart, C. M. Shoemaker and D. W. Gage, vol. 6230, p. 62302O, International Society for Optics and Photonics (SPIE, 2006), URL https://doi.org/10.1117/12.684087.

Qi, Q., F. Tao, T. Hu, N. Anwer, A. Liu, Y. Wei, L. Wang and A. Nee, "Enabling technologies and tools for digital twin", Journal of Manufacturing Systems **58**, 3–21, URL https://www.sciencedirect.com/science/article/pii/S0278612519300860X, digital Twin towards Smart Manufacturing and Industry 4.0 (2021).

Rahmani, M., A. Ghanbari and M. M. Ettefagh, "Hybrid neural network fraction integral terminal sliding mode control of an inchworm robot manipulator", Mechanical Systems and Signal Processing **80**, 117–136, URL https://www.sciencedirect.com/science/article/pii/S0888327016300449 (2016).

Rahmani, M. and S. Redkar, "Deep neural data-driven koopman fractional control of a worm robot", Expert Systems with Applications **256**, 124916, URL https://www.sciencedirect.com/science/article/pii/S0957417424017834 (2024a).

Rahmani, M. and S. Redkar, "Optimal dmd koopman data-driven control of a worm robot", Biomimetics **9**, 11, URL https://www.mdpi.com/2313-7673/9/11/666 (2024b).

Rajamurugu, N. and M. K. Karthik, *Introduction, History, and Concept of Digital Twin*, chap. 2, pp. 19–32 (John Wiley Sons, Ltd, 2022), URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119842316.ch2.

Seeja, G., A. Selvakumar Arockia Doss and V. B. Hency, "A survey on snake robot locomotion", IEEE Access **10**, 112100–112116 (2022).

Shah, P. and S. Agashe, "Review of fractional pid controller", Mechatronics **38**, 29–41, URL https://www.sciencedirect.com/science/article/pii/S095741581630068X (2016).

Shiotan, S., K. KEMMOTSU, T. TOMONAKA, S. ASANO, K. OONISHI and R. HIURA, "World's first full-fledged communication robot "wakamaru" capable of living with family and supporting persons", Mitsubishi Heavy Industries, Ltd. Technical Review **43**, 1, URL https://www.mhi.co.jp/technology/review/pdf/e431/e431044.pdf (2006).

Sugita, S., K. Ogami, G. Michele, S. Hirose and K. Takita, "A study on the mechanism and locomotion strategy for new snake-like robot active cord mechanism – slime model 1 acm-s1", Journal of Robotics and Mechatronics **20**, 2, 302–310 (2008).

Taubes, G., "Biologists and engineers create a new generation of robots that imitate life", Science **288**, 5463, 80–83, URL https://www.science.org/doi/abs/10.1126/science.288.5463.80 (2000).

Taylor, J. E., G. Bennett and N. Mohammadi, "Engineering smarter cities with smart city digital twins", Journal of Management in Engineering **37**, 6, 02021001, URL https://ascelibrary.org/doi/abs/10.1061/%28ASCE%29ME.1943-5479.0000974 (2021).

TechnoHacks, "Digital twins: The virtual replicas revolutionizing industries", URL https://www.technohacks.net/digital-twins-the-virtual-replicas-revolutionizing-industries/, accessed: 2024-03-05 (2024).

Transeth, A. A., R. I. Leine, C. Glocker and K. Y. Pettersen, "Non-smooth 3d modeling of a snake robot with frictional unilateral constraints", in "2006 IEEE International Conference on Robotics and Biomimetics", pp. 1181–1188 (2006).

Transeth, A. A., K. Y. Pettersen and P. Liljebäck, "A survey on snake robot modeling and locomotion", Robotica **27**, 7, 999–1015 (2009).

Wakimoto, S., J. Nakajima, M. Takata, T. Kanda and K. Suzumori, "A micro snake-like robot for small pipe inspection", in "MHS2003. Proceedings of 2003 International Symposium on Micromechatronics and Human Science (IEEE Cat. No.03TH8717)", pp. 303–308 (2003).

Yamada, H. and S. Hirose, "Development of practical 3-dimensional active cord mechanism acm-r4", Journal of Robotics and Mechatronics **18**, 3, 305–311 (2006).

Zhang, B., Y. Fan, P. Yang, T. Cao and H. Liao, "Worm-like soft robot for complicated tubular environments", Soft Robotics **6**, 3, 399–413, URL `https://doi.org/10.1089/soro.2018.0088`, pMID: 31180823 (2019).

APPENDIX A

RAW DATA AND CODE

Consult the attached files. The MATLAB and Python codes used in this thesis are hosted on GitHub. Please visit Github to access these files. Note: MATLAB or Python 3.0 is required to run the code.