12-2016

Actuation of an Inertia-Coupled Rimless Wheel Model across Level Ground

Seth Caleb Weeks

Follow this and additional works at: http://scholarworks.rit.edu/theses

Recommended Citation


This Thesis is brought to you for free and open access by the Thesis/Dissertation Collections at RIT Scholar Works. It has been accepted for inclusion in Theses by an authorized administrator of RIT Scholar Works. For more information, please contact ritscholarworks@rit.edu.
Actuation of an Inertia-Coupled Rimless Wheel Model across Level Ground

by

Seth Caleb Weeks

A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master of Science in Mechanical Engineering

Supervised by

Senior Lecturer Dr. Mario Gomes
Department of Mechanical Engineering
Kate Gleason College of Engineering
Rochester Institute of Technology
Rochester, New York
December 2016

Approved by:

Dr. Mario Gomes, Senior Lecturer
Thesis Advisor, Department of Mechanical Engineering

Dr. Kathleen Lamkin-Kenard, Associate Professor
Committee Member, Department of Mechanical Engineering

Dr. Jason Kolodziej, Associate Professor
Committee Member, Department of Mechanical Engineering

Dr. Agamemnon Crassidis, Graduate Director
Department Representative, Department of Mechanical Engineering
Thesis Release Permission Form

Rochester Institute of Technology
Kate Gleason College of Engineering

Title:

Actuation of an Inertia-Coupled Rimless Wheel Model across Level Ground

I, Seth Caleb Weeks, hereby grant permission to the Wallace Memorial Library to reproduce my thesis in whole or part.

Seth Caleb Weeks

Date
Acknowledgments

Special thanks to the people who I have worked with in the Energy and Motion Lab: Samket Dhami, Glenn Gavi, Ashley Kosak, Konrad Ahlin, Saloni Vardhan and Shree Shalia Thiperappa Swamy. Your support and encouragement has shaped and developed my interests. Thank you to my professors who have provided an excellent understanding of mechanical concepts. I am very grateful for the time and interest Dr. Gomes has invested in me since my freshman year at RIT. He has inspired my approach to both research and teaching. Thank you to my friends and family who have supported me along this long journey. May God be glorified in everything.
Abstract

Actuation of an Inertia-Coupled Rimless Wheel Model across Level Ground

Seth Caleb Weeks

Supervising Professor: Dr. Mario Gomes

The inertia-coupled rimless wheel model is a passive dynamic walking device which is theoretically capable of achieving highly efficient motion with no energy losses. Under non-ideal circumstances, energy losses due to air drag require the use of actuation to maintain stable motions. The Actuated Inertia-coupled Rimless Wheel Across Flat Terrain (AIRWAFT) model provides actuation to an inertia-coupled rimless wheel model across level ground to compensate for energy losses by applying hip-torque between the frame and inertia wheel via a motor. Two methods of defining the open-loop actuation are presented. Position control defines the relative position of the drum relative to the frame. Torque control specifies the amount of torque between the frame and the drum. The performance of the model was evaluated with respect to changes in various geometrical and control parameters and initial conditions. This parameter study led to the discovery of a stable, periodic motion with a cost of transport of 0.33.
# Contents

Acknowledgments .............................................. iv

Abstract .................................................. v

1 Background Information ................................. 2
   1.1 Problem Introduction .............................. 2
   1.2 Literature Review ................................ 3
       1.2.1 Motivation .................................. 3
       1.2.2 Learning from Nature ...................... 3
       1.2.3 Applications ................................ 6
       1.2.4 Areas of Study .............................. 8
       1.2.5 Passive Dynamic Walking ................... 9
       1.2.6 Energy Losses ............................... 11
       1.2.7 Methods of Actuation ...................... 12
       1.2.8 Gaps in Research ........................... 13
   1.3 Research Goals .................................. 13

2 AIRWAFT Model ........................................ 15
   2.1 Summary ......................................... 15
   2.2 Assumptions ..................................... 15
   2.3 Model Definition ................................ 16
   2.4 Single Stance ................................... 19
   2.5 Double Stance .................................. 22
   2.6 Event Detection ................................ 25
   2.7 Collision Transition Equations ................. 25
   2.8 Energy Calculations .............................. 26
   2.9 Simulation Methodology ......................... 29

3 Research Approach .................................. 32

4 Torque Control ........................................ 41
B.6 Event detection for foot lift off ........................................... 96
B.7 Root find ............................................................................. 96
B.8 Periodic conditions output .................................................. 99
B.9 Calculate performance metrics for periodic step ................... 101
B.10 Animation ......................................................................... 106
## List of Tables

2.1 Standard system parameters for used to simulate the model .......................... 18

3.1 Initial conditions that converge to periodic motion for no drag and no power .......................................................... 33
3.2 Initial conditions with drag and power that converge to periodic motion 34
3.3 Initial conditions after convergence with drag and power ............................ 36
3.4 Control parameters found by root find ................................................. 39
3.5 Initial conditions found by root find .................................................. 39

4.1 Parameters that define the motor torque function with respect to time. 44

5.1 Initial conditions as used for parameter study ....................................... 47
5.2 Parameters that define the motor torque function with respect to time as used for parameter study ................................................. 47
5.3 Performance metrics for periodic motion with lowest cost of transport found .......................................................... 56
5.4 Control function parameters, geometric parameters and initial conditions that produce a periodic motion with a cost of transport of 0.3267 58
List of Figures

1.2 Study of human response to visual cues while walking and standing.
   Image taken from [5] ................................................................. 4
1.3 Mobility of a monopodial amoeba. Image taken from [8] ................. 5
1.4 Brachiation motion as demonstrated in primates. Image taken from [10] 6
1.5 Cost of transport (COT) against speed for various robotic walkers.
   Image taken from [12] ............................................................... 7
1.6 An energy recycling foot. Image taken from [14] ............................... 7
1.7 Comparison of (a) rolling, (b) ZMP, and (c) passive dynamic walking.
   Image taken from [4] ................................................................. 8
1.8 Geometry of rimless wheel and bipedal models. Image taken from [16] 10
1.9 Percentage of energy lost per oscillation in relation to initial velocity.
   Image taken from [19] .............................................................. 11

2.1 Geometrical parameters and system variables for the AIRWAFT model 16
2.2 A motor supplies torque between the drum and the frame ................. 17
2.3 Free body diagram of single stance phase for the frame, inertia wheel
   and drum. (i,j,k is the fixed coordinate system; ˆuβ, ˆuξ moves with frame) 19
2.4 Free body diagram of double stance phase for the frame, inertia wheel
   and drum. (i,j,k is the fixed coordinate system) ............................ 22
2.5 Integrated values for θ fluctuate for different tolerances values until
   RelTol = AbsTol = 10^{-8} ....................................................... 29
2.6 Definition of angles before a step, after a step and after the coordinate
   reset. ...................................................................................... 31

3.1 Component velocities for five steps with no drag and no power .......... 33
3.2 System energy for five steps with no drag and no power .................. 34
3.3 Component velocities for five steps with drag and power ................. 35
3.4 System energy for five steps with drag and power ........................... 35
3.5 Phase plane trajectory for fifty steps after convergence. Part (a) shows the entire trajectory while part (b) shows the zoomed in section where the collision occurs.  
(a) Before root find  
(b) After root find

3.6 Total energy for fifty steps after convergence

3.7 Phase plane trajectory of motion found using root find

4.1 Torque profile from a single periodic step using position control along with sinusoidal curve of best fit

4.2 Projection of phase plane trajectory on the $\theta - \dot{\theta}$ plane for (a) the replicated step from position control and (b) the step found using a root find in torque control.  
(a) Before root find  
(b) After root find

5.1 Cost of transport has a minimum at eight legs and speed decreases as the number of legs increases.

5.2 Energy losses due to air drag and collisions increases as the number of legs increases.

5.3 Cost of transport increases and speed decreases as the damping coefficient increases.

5.4 Energy losses due to air drag and collisions both increase as the damping coefficient increases.

5.5 Speed increases as the phase shift increases. The lowest cost of transport occurs when $P = 3.6772$ radians.

5.6 Energy lost due to air drag increases as phase shift increases. Energy lost due to collisions increases until a value of 3.725 radians after which it begins to decrease again.

5.7 Cost of transport and speed increases as the initial velocity of the inertia wheel increases.

5.8 Energy losses due to air drag and collisions both increase as the initial velocity of the inertia wheel increases.

5.9 Cost of transport and speed both decrease as the mass of the frame increases.

5.10 Energy losses due to collisions and air drag both increase as the mass of the frame increases.
5.11 Cost of transport increases and speed decreases as the mass moment of inertia for the inertia wheel is increased ........................................... 54
5.12 Energy losses due to air drag and collisions both increase as the mass moment of inertia for the inertia wheel increases ......................... 54
5.13 Speed increases as the spring constant increases. the lowest cost of transport occurs when $k = 4 \text{[N-m/rad]}$. ........................................ 55
5.14 Energy losses due to air drag and collisions decrease as the spring constant is increased ................................................................. 55
5.15 Phase plane trajectory of motion with lowest COT found using torque control ................................................................. 57
5.16 Total energy for one step of motion with lowest COT found using torque control ................................................................. 57
5.17 Component angular positions of motion with lowest COT found using torque control ................................................................. 57
5.18 Component angular velocities of motion with lowest COT found using torque control ................................................................. 58

6.1 Power required by the motor for a single periodic step with the lowest COT found ................................................................. 60
6.2 Torque-velocity curve for selected motor along with motion of the device 60
Nomenclature

\( n \)  Number of legs \\
\( l \)  Length of legs \\
\( r \)  Drum radius \\
\( m_f \)  Mass of the frame \\
\( m_i \)  Mass of the inertia wheel \\
\( m_d \)  Mass of the drum \\
\( I_f \)  Moment of inertia of the frame with respect to center of mass \\
\( I_i \)  Moment of inertia of the inertia wheel with respect to center of mass \\
\( I_d \)  Moment of inertia of the drum with respect to center of mass \\
\( g \)  Acceleration due to gravity \\
\( k \)  Angular spring constant \\
\( \beta \)  Angular position of the frame \\
\( \dot{\beta} \)  Angular velocity of the frame \\
\( \ddot{\beta} \)  Angular acceleration of the frame \\
\( \theta \)  Angular position of the inertia wheel \\
\( \dot{\theta} \)  Angular velocity of the inertia wheel \\
\( \ddot{\theta} \)  Angular acceleration of the inertia wheel \\
\( \phi \)  Angular position of the drum \\
\( \dot{\phi} \)  Angular velocity of the drum \\
\( \ddot{\phi} \)  Angular acceleration of the drum \\
\( \tau_m \)  Motor torque \\
\( \tau_s \)  Spring torque \\
\( \tau_d \)  Air drag torque \\
\( R_x \)  Horizontal reaction force on foot during single stance \\
\( R_y \)  Vertical reaction force on foot during single stance \\
\( R_{1x} \)  Horizontal reaction force on trailing foot during double stance \\
\( R_{1y} \)  Vertical reaction force on trailing foot during double stance \\
\( R_{2y} \)  Vertical reaction force on leading foot during double stance \\
\( F_x \)  Horizontal force between frame and drum \\
\( F_y \)  Vertical force between frame and drum \\
\( S_x \)  Horizontal force between drum and inertia wheel \\
\( S_y \)  Vertical force between drum and inertia wheel \\
\( COT \)  Cost of transport \\
\( A \)  Amplitude of sinusoidal control function \\
\( T \)  Period of sinusoidal control function \\
\( F \)  Frequency of sinusoidal control function \\
\( P \)  Phase shift of sinusoidal control function
Chapter 1

Background Information

1.1 Problem Introduction

The study of walking robotics involves observation and implementation of archetypes found in nature, the pursuit of advanced functionality and performance, and application to real world problems. Unfortunately, these systems often have a large energetic cost relative to living organisms. Energy losses prevent perfect efficiency, but understanding how to reduce these losses can contribute to more efficient walking motions. The inertia-coupled rimless wheel model utilizes the concepts of passive dynamic walking in conjunction with natural spring oscillations to minimize energy losses due to collisions. Physically implementing this model has proven difficult for motions across level ground. This research investigates a method for providing actuation to the inertia-coupled rimless wheel model that is feasible to physically construct with the objective of maintaining a high level of energy efficiency.
1.2 Literature Review

1.2.1 Motivation

Although rolling motion has dominated as the preferred method of transportation, there are apparent advantages to walking devices. Wheels permit highly efficient transfer of kinetic energy with minimal losses to contact friction. However, without a continuously smooth surface, wheels cannot traverse as proficiently, eliciting losses in energy and stability. Additionally, vehicles with wheels are not naturally capable of complex motions such as jumping or climbing.

Despite some of the apparent advantages of walking, this topic has been considered as little more than a curiosity for centuries. Simple mechanical toys demonstrated that walking motion could be sustained naturally, one of which is shown in Fig. 1.1. In response to the increasing need for expanded versatility and capabilities of moving vehicles over more complex terrain, the investigation of modern walking machines began in the late 20th century with the work of Alexander, McGeer and others [1][2].

![Fig. 1.1: Passive dynamic walking toy. Image taken from [3]](image)

1.2.2 Learning from Nature

Since walking is a natural mode of locomotion for humans, emulating human-like gait is large motivator for many who work in the robotics community. Humans exhibit
gait patterns that are both energy efficient and highly versatile [4]. Therefore, the human anatomy serves as an excellent example for energetic walking devices. Gait patterns, stability control and actuation methods are a few categories investigated through the study of human walking.

A study was conducted in which human subjects were exposed to visual feedback mimicking alterations in their orientation as shown in Fig. 1.2. The results showed that during walking, there is a low sensitivity to visual cues for forward motion but a high sensitivity for lateral movement. When standing, there is a high sensitivity to visual cues in both directions [5]. This implies that less control is required to maintain forward motion than lateral stability while walking and validates that humans exhibit passive dynamic motion.

![Fig. 1.2: Study of human response to visual cues while walking and standing. Image taken from [5]](image)

Advanced technology has made learning from nature even easier. Using the positional data from nonlinear dynamical systems at strategic locations, reverse engineering of symbolic representations can be automated [6]. These equations can be used to accurately simulate complex motions including walking. The gait pattern of the simplest passive dynamic walker (PDW) model was shown to be comparable to that of humans using the root mean square error comparison of hip angles and
angular velocities, to within 6.9% and 12.2% respectively. Because of the apparent similarities, it was concluded that emulating human muscle movement has potential improvements for PDWs, particularly in maintaining stability and energetic efficiency [7].

![Fig. 1.3: Mobility of a monopodial amoeba. Image taken from [8]](image)

The study of natural motion exhibited in other organisms has also contributed to the progression of legged locomotion. A series of robots were developed at the Robotics and Mechanisms Laboratory at Virginia Tech which incorporate dynamics based on amoebas and geckos. Whole Skin Locomotion is inspired by the cytoplasmic streaming and amorphous membrane of the amoeba proteus as seen in Fig. 1.3. This demonstrates a form of single-foot walking. The concept of dry-adhesive feet for walking in zero gravity environments is adapted from gecko feet [8]. The Whegs II imitates the jointed body segments of the cockroach in order to optimize energy efficiency while traversing rough terrain [9]. A control scheme for brachiation motion, shown in Fig. 1.4, has been developed based on swinging primates [10]. Brachiation motion is similar to the simplest form of bipedal gait differing mainly in that the limbs are located below the surface of contact instead of above. These four examples provide references of improvement in robotic agility and efficiency.

Certain concepts involved in the functionality of the inertia-coupled rimless wheel model are correlated with nature. The passive dynamic walking approach is analogous to a simplified human gait. Humans tend to accentuate each step while attempting to walk quietly, minimizing impact with the ground. The inertia wheel implements a similar concept by slowly decelerating the frame such that the velocity of the frame
approaches zero as the leading foot nears the ground.

1.2.3 Applications

The concepts and technologies developed in the field of robotic walking have applications both within and beyond the scope of robotics. The capabilities of walking machines are expanding while energetic costs are diminishing. Additionally, expanded knowledge advances research regarding artificial limb support and increased balance control. Walking devices are being designed for use in space exploration, human interface, assisted home living and other sectors.

Controlled with high level artificial intelligence, the Honda ASIMO has been leased to companies to perform receptionist work. This humanoid is capable of detecting obstructions and traffic, recognizing speech and sound patterns, responding to human gestures and processing information. The function of navigation utilizes highly controlled zero moment point walking to ensure smooth and stable motion [11].

Cotton et al. simulated a fast running robot based on the biological makeup of ostriches [12]. The model demonstrated improved energetic efficiency compared to other walking robots at velocities exceeding most human running speeds as shown in Fig. 1.5. While designing a physical prototype of a related system, developments resulted in a commercial robot, called the Outrunner, that is similar in nature to the rimless wheel model [13]. This machine demonstrates the energetic efficiency that
can be achieved in experimental devices while maintaining exceptional agility.

![Cost of Transport vs Speed](image)

Fig. 1.5: Cost of transport (COT) against speed for various robotic walkers. Image taken from [12]

An energy recycling artificial foot reduces ankle push-off by recovering energy from heel impact as seen in Fig. 1.6. The device was constructed based on the simple bipedal gait pattern demonstrated by humans. It stores energy while the subject has both feet on the ground and is shifting weight from the rear foot to the front foot. As the rear foot is about to leave the ground, energy is released. This study of energetic recycling is useful in developing assistive technologies [14].

![Energy Recycling Foot](image)

Fig. 1.6: An energy recycling foot. Image taken from [14]

The Honda ASIMO incorporates robotic walking into a full humanoid system while the energy recycling foot applies knowledge gained about push-off actuation techniques to supportive devices. Similarly to how the Outrunner maximizes speed while maintaining a low COT, the inertia-coupled rimless wheel model attempts to
maximize efficiency. Decreasing energy usage has the potential of significantly reducing operational costs, subsequently increasing the practicality of walking robots.

1.2.4 Areas of Study

In the field of walking devices, there are three main categories of study: versatility, efficiency and stability. In a consumer driven society where technology is influenced by needs and demands, most research in this area is invested into improved versatility and agility. Once this initial challenge has been accomplished, the other issues of stability and energy efficiency become important in making walking devices practical [4].

![Comparison of (a) rolling, (b) ZMP, and (c) passive dynamic walking. Image taken from [4]](image)

A majority of modern walking robots employ some level of zero moment point
(ZMP) dynamics as shown in Fig. 1.7(b). This method simplifies the stability of dynamics to a set of steady-state equations where all forces are in balance [15]. Essentially, the issue of stability is marginalized by maintaining static and dynamic equilibrium at all times, and efforts can be concentrated towards improving versatility and agility.

Passive dynamic walking generally attempts to optimize energy efficiency. It has been demonstrated that walking motion can be sustained with very minimal control. There are various methods of solving for stable solutions including observing period bifurcation, perturbations from known solutions and convergence analysis. Since most passive dynamic walkers are simplified to two dimensional systems, lateral stability in three dimensions is also an extended area of interest. While these systems are generally limited in performance, attempts have been made to expand versatility by improving speed and response to rough terrain, among other pursuits.

In an attempt to find the most energetically efficient walking model, the inertia-coupled rimless wheel model uses the passive dynamic walking approach instead of ZMP locomotion. While the device may not be as versatile or agile, the efficiency is far superior to that of other models.

1.2.5 Passive Dynamic Walking

The first mathematical bipedal model, described by Alexander, consists of an upper-body with massless legs that are able to perform work on the body while in contact with the ground [1]. McGeer demonstrated that this model was capable of maintaining stable periodic motion down an incline [2]. Extensions to this model have led to more complicated systems with knees, ankles and extended upper-bodies. Bipedal devices are of particular interest for research because of their observable similarities to human walking.

While introducing his experimental model, McGeer describes the rimless wheel
model, which derives its name from a wagon wheel without a rim such that only the spokes and hub are remaining. This resulting perimeter outlines the shape of a regular polygon with a finite number of sides [2]. Although it is debated as to whether the motion of a rimless wheel constitutes walking or rolling, the model provides a simplistic approach to maximizing energy efficiency without the added difficulties of leg extension and lateral stability. These two models have clear similarities in geometry (see Fig. 1.8) and dynamics which allows for comparison of performance and stability as studied by Byl [16].

Gomes presents an extension to the rimless wheel model which includes an inertia wheel. This added component utilizes naturally occurring periodic spring oscillations to control the stride velocity. Physical experiments have demonstrated that nearly collisionless gait can be achieved down a slight decline. The ramp angle is directly related to the cost of transport (COT), which is a dimensionless performance factor characterizing the amount of energy required to move a certain weight a unit distance. A prototype was constructed with a COT of approximately 0.052 [17].

\[
COT = \frac{(\text{Energy required})}{(\text{Weight})(\text{Distance travelled})} \tag{1.1}
\]
1.2.6 Energy Losses

The inertia-coupled rimless wheel requires an inclined plane because there are energy losses in the system which gravity compensates for. Energy is lost due to air friction, spring hysteresis, structural flexure and collisional dissipation. It is difficult to determine precisely where energy is being lost throughout the entire step cycle.

A dynamic tensile test of the springs and various strings demonstrated that roughly 1.23% of total energy is lost due to hysteresis per cycle (for a given string length and material). The study showed that there is not a significant increase in energy losses at higher velocities [18].

Experimental studies coupled with simulations have shown that air friction might be the primary cause of energy losses, accounting for about 30% of total energy loss per cycle as seen in Fig. 1.9. Aerodynamics is not often considered in robotics, but becomes significantly important when attempting to design the most efficient walking device. This is especially true for fast moving robots [19].

Fig. 1.9: Percentage of energy lost per oscillation in relation to initial velocity. Image taken from [19]
Although it has not been fully determined exactly where and how energy is lost for the inertia coupled rimless wheel, a working understanding of how to model such losses in simulation has been formulated. For oscillations, energy is not lost as a linear function of time, but as a function of the velocities of the frame and inertia wheel.

1.2.7 Methods of Actuation

In order to compensate for energy losses, actuation must be provided. The two primary methods of actuating both the rimless wheel and the bipedal model are toe-off and hip-torque. These approaches add energy as needed to maintain natural dynamics rather than acting as a driving force.

![Fig. 1.10: Toe-off method of actuation. Image taken from [4]](image)

It was previously thought that passive dynamic walkers depended specifically on gravitational energy for ambulation. However, experimental models have proven that toe-off and hip-torque actuation are just as effective energy sources [20]. Toe-off describes the method of adding energy between the contact foot and the ground as shown in Fig. 1.10. This is accomplished in various ways including extendable legs and asymmetric feet. Applying hip-torque provides a force between the legs and a torso. In the rimless wheel model, the hip-torque method is achieved by applying a force between the frame and the inertia wheel.
1.2.8 Gaps in Research

Gomes and Ahlin have attempted to find the most energetically efficient model for walking robots [19]. They developed an inertia-coupled rimless wheel that is capable of sustaining periodic and collisionless motion down a ramp with frictional energy losses. However, a walking device is only practical if it is able to traverse across various levels of inclination, including level ground.

Attempts have been made to actuate different physical and mathematical models across level ground. Gomes and Ahlin have investigated various methods of actuating the inertia-coupled rimless wheel model. The Central Motor System and the Offset Motor System use a motor to add a torque between the frame and the inertia wheel. The Frame Link System adds torque between the inertia wheel and the ground via a chain of rigid links. Each of these three methods only provide power at intermittent intervals. However, the Reaction Device System uses a secondary inertial device to continuously apply torque to the frame. This approach resulted in periodic, collisionless motion with a system COT of 0.0233 [17].

Although the concept of the Central Motor System seems promising, attempts to construct a physical system exposed difficulties as the motor needed to be coupled directly to the shaft that connects to the inertia wheel. Additionally, the motor is only engaged with the inertia wheel during double stance phase which results in greater energy losses during the single stance phase. While Ahlin attempted to solve these issues using the Offset Motor System, the motor was still only activated during certain periods of the step cycle.

1.3 Research Goals

The goal of this research is to characterize energetic performance of a stable, low-energy cost system on level ground actuated via hip-torque by answering the following
three key questions:

1. How do variations in the controller and/or system parameters affect energy efficiency?

2. What is a local minimum for the cost of transport for a powered system with energy losses given the variation of parameters as previously determined?

3. Are there motor parameters that produce reasonably achievable results using readily available physical components?
Chapter 2

AIRWAFT Model

2.1 Summary

The Actuated Inertia-coupled Rimless Wheel Across Flat Terrain (AIRWAFT) model is an extension to the model developed by Ahlin [17]. The model attempts to address the issues with physically constructing the Central Motor System used to power the device by adding a secondary central axis and continuously adjusting the position of the spring during both single and double stance phases. The system is modeled as a two dimensional system for the sake of simplicity. Using this approach simplifies the geometry of components and eliminates lateral stability concerns. It is also feasible to construct physical models that closely align with this two dimensional approach. One such model that was constructed Ahlin [19].

2.2 Assumptions

For the purposes of this thesis, certain assumptions were made regarding the model to simplify simulation. Motor dynamics were not modeled with the assumption that drum position and motor torque is defined as a known function of time. Stiction is neglected in the motor and surfaces of contact such as the location where the torsional spring interfaces with the inertia wheel. The model assumes that bodies are perfectly rigid. It is also assumed that the values for parameters are known precisely, and the
simulation does not account for tolerances or deviations from these values.

2.3 Model Definition

The AIRWAFT system consists of three main components: the frame, inertia wheel, and drum as shown in Fig. 2.1. The frame is defined as a regular polygon, modeling the geometry of a simple rimless wheel in the physical prototypes. The purpose of the inertia wheel is to regulate the motion of the step, introducing periodic motion and minimizing collisions. The inertia wheel is coupled to the drum by a torsional spring. The drum adds energy to the system and is coupled to the frame via an electric motor. The center of mass for each component is at the same location in two dimensional space for all three components.

![Diagram of AIRWAFT model components](image)

Fig. 2.1: Geometrical parameters and system variables for the AIRWAFT model

In a real physical system, energy is lost throughout the cycle through various forms of friction, at collisions between the feet and the ground and negative work absorbed by the motor. In order to compensate for these energy losses, a motor torque is supplied to rotate the drum so that the motion can be sustained on level ground as seen in Fig. 2.2. In this model, air drag is modeled as an external torque on the inertia wheel which is linearly proportional to the absolute angular velocity of
the inertia wheel ($\tau_d = c\dot{\theta}$).

System parameters shown in Table 2.1 were chosen based on existing components for physical prototypes [19]. The length of legs, mass of frame and inertia wheel, and mass moment of inertia about the center of mass for the frame and inertia wheel are taken from a physical model of an inertia-coupled rimless wheel that Ahlin constructed. The torsional spring constant ($c$) is a simplified version of the air drag model used by Ahlin in simulated models. Specifications for the drum ($m_d, r, I_d$) were measured from a proposed physical drum for use in constructing the AIRWAFT model.

In the context of the rimless wheel model, walking is defined as a motion with intermittent contact between the feet and the ground with single and double stance phases. During single stance, only one foot is in contact with the ground while two feet are touching the ground in double stance. In contrast, running is defined by a single stance phase and a flight phase, during which no feet are in contact with the ground.

A step in the AIRWAFT model is defined to start at the beginning of single stance. When the leading foot hits the ground, the model switches to double stance. The step is complete when the trailing leg is about to lift off the ground again which
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>8</td>
<td>-</td>
<td>Number of legs</td>
</tr>
<tr>
<td>l</td>
<td>0.3935</td>
<td>m</td>
<td>Length of legs</td>
</tr>
<tr>
<td>r</td>
<td>0.0646</td>
<td>m</td>
<td>Drum radius</td>
</tr>
<tr>
<td>m_f</td>
<td>1.438</td>
<td>kg</td>
<td>Frame mass</td>
</tr>
<tr>
<td>m_i</td>
<td>1.803</td>
<td>kg</td>
<td>Inertia wheel mass</td>
</tr>
<tr>
<td>m_d</td>
<td>0.303</td>
<td>kg</td>
<td>Drum mass</td>
</tr>
<tr>
<td>I_f</td>
<td>0.0851</td>
<td>kg·m²</td>
<td>Frame mass moment of inertia</td>
</tr>
<tr>
<td>I_i</td>
<td>0.0984</td>
<td>kg·m²</td>
<td>Inertia wheel mass moment of inertia</td>
</tr>
<tr>
<td>I_d</td>
<td>0.0018</td>
<td>kg·m²</td>
<td>Drum mass moment of inertia</td>
</tr>
<tr>
<td>g</td>
<td>9.81</td>
<td>m/s²</td>
<td>Gravity constant</td>
</tr>
<tr>
<td>k</td>
<td>4</td>
<td>N·m/rad</td>
<td>Torsional spring constant</td>
</tr>
<tr>
<td>c</td>
<td>0.004</td>
<td>N·m·s/rad</td>
<td>Torsional air drag coefficient</td>
</tr>
</tbody>
</table>

Table 2.1: Standard system parameters for used to simulate the model

transitions to the beginning of single stance.
2.4 Single Stance

During single stance, the center of mass of each component rotates around the point where the foot is contact with the ground. Gravity acts on the center of mass of each component. It is assumed that the force of friction between the foot and the ground is large enough to ensure that the foot does not slip. This assumption is validated during simulation by calculating and monitoring the horizontal reaction force on the foot. It is also assumed that the three main bodies are perfectly rigid. In order to assist with mathematical calculations, a secondary coordinate system was specified ($\hat{u}_\beta$ and $\hat{u}_l$) which rotates with the frame. The drum exerts a torque on the frame through the motor and the inertia wheel exerts a torque on the drum through the springs. Free body diagrams for the system during single stance are shown in Fig. 2.3.

Using Newton’s second law of motion, a set of linearly independent equations with nine unknowns ($\ddot{\beta}, \dot{\theta}, F_x, F_y, R_x, R_y, S_x, S_y, \tau_m$) was derived. The acceleration of the center of mass of all three components are the same since the center of mass is at the same geometrical location.

$$\ddot{a}_{CM} = l\ddot{\beta}\hat{u}_\beta - l\dot{\beta}^2\hat{u}_l$$

(2.1)
\[
\hat{u}_\beta = \sin(\beta) \hat{i} + \cos(\beta) \hat{j} 
\] (2.2)

\[
\hat{u}_l = -\cos(\beta) \hat{i} + \sin(\beta) \hat{j} 
\] (2.3)

For the frame:

\[
\sum \vec{F} = m_f \vec{a}_{CM} = F_x \hat{i} + F_y \hat{j} - m_f g \hat{j} + R_x \hat{i} + R_y \hat{j} 
\] (2.4)

\[
\sum \vec{M}_{f/CM} = I_{f/CM} \vec{\alpha}_f = \tau_m \hat{k} - l \hat{u}_l \times (R_x \hat{i} + R_y \hat{j}) 
\] (2.5)

For the inertia wheel:

\[
\sum \vec{F} = m_i \vec{a}_{CM} = -S_x \hat{i} - S_y \hat{j} - m_i g \hat{j} 
\] (2.6)

\[
\sum \vec{M}_{i/CM} = I_{i/CM} \vec{\alpha}_i = \tau_s \hat{k} + \tau_d \hat{k} 
\] (2.7)

For the drum:

\[
\sum \vec{F} = m_d \vec{a}_{CM} = S_x \hat{i} + S_y \hat{j} - F_x \hat{i} - F_y \hat{j} - m_d g \hat{j} 
\] (2.8)

\[
\sum \vec{M}_{d/CM} = I_{d/CM} \vec{\alpha}_d = -\tau_m \hat{k} - \tau_s \hat{k} 
\] (2.9)

Separating the equations into \( \hat{i} \), \( \hat{j} \) and \( \hat{k} \) components yields the following nine scalar equations:

\( \hat{i} \):

\[
m_f l \dot{\beta} \sin(\beta) + m_f l \dot{\beta}^2 \cos(\beta) = F_x + R_x 
\] (2.10)

\[
m_i l \dot{\beta} \sin(\beta) + m_i l \dot{\beta}^2 \cos(\beta) = -S_x 
\] (2.11)

\[
m_d l \dot{\beta} \sin(\beta) + m_d l \dot{\beta}^2 \cos(\beta) = S_x - F_x 
\] (2.12)

\( \hat{j} \):

\[
m_f l \dot{\beta} \cos(\beta) - m_f l \dot{\beta}^2 \sin(\beta) = F_y + R_y - m_f g 
\] (2.13)
\[ m_i l \ddot{\beta} \cos(\beta) - m_i l \dot{\beta}^2 \sin(\beta) = -S_y - m_i g \]  
\[ (2.14) \]

\[ m_d l \ddot{\beta} \cos(\beta) - m_d l \dot{\beta}^2 \sin(\beta) = S_y - F_y - m_d g \]  
\[ (2.15) \]

\[ \hat{k} : \]

\[ I_{f/CM} \ddot{\beta} = \tau_m + l \sin(\beta) R_x + l \cos(\beta) R_y \]  
\[ (2.16) \]

\[ I_{i/CM} (\ddot{\beta} + \ddot{\phi} + \ddot{\theta}) = -\tau_s - \tau_d \]  
\[ (2.17) \]

\[ I_{d/CM} (\ddot{\beta} + \ddot{\phi}) = -\tau_m - \tau_s \]  
\[ (2.18) \]

In order to solve for the unknown variables, the equations are written in matrix form \([A] \vec{x} = \vec{b}\), where \([A]\) contains the coefficients of the unknown variables, \(\vec{x}\) is a column vector of those same variables, and \(\vec{b}\) specifies known expressions as shown in equation 2.19.

\[
\begin{bmatrix}
m_f l \sin(\beta) & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\
m_f l \cos(\beta) & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 \\
-I_{f/CM} & 0 & 0 & 0 & -l \sin(\beta) & -l \cos(\beta) & 0 & 0 & -1 \\
m_d l \sin(\beta) & 0 & 1 & 0 & 0 & 0 & -1 & 0 & 0 \\
m_d l \cos(\beta) & 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\
-I_{d/CM} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
m_i l \sin(\beta) & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
m_i l \cos(\beta) & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
-I_{i/CM} & -I_{i/CM} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
\ddot{\beta} \\
\ddot{\theta} \\
F_x \\
F_y \\
R_x \\
R_y \\
S_x \\
S_y \\
\tau_m \\
\end{bmatrix}
= \begin{bmatrix}
-m_f l \dot{\beta}^2 \cos \beta \\
m_f l \dot{\beta}^2 \sin \beta - m_i g \\
0 \\
-m_d l \dot{\beta}^2 \cos \beta \\
m_d l \dot{\beta}^2 \sin \beta - m_d g \\
I_{d/CM} \ddot{\phi} - \tau_s \\
-m_i l \dot{\beta}^2 \cos \beta \\
-m_i l \dot{\beta}^2 \sin \beta - m_i g \\
I_{i/CM} \ddot{\phi} + \tau_s + \tau_d \\
\end{bmatrix}
\]  
\[ (2.19) \]

The unknown variables can be determined by solving the linear system for \(\vec{x}\).

\[ \vec{x} = [A]^{-1} \vec{b} \]  
\[ (2.20) \]
2.5 Double Stance

During the double stance phase of motion, two feet are in contact with the ground and the linear acceleration of the center of mass of all three bodies is zero. It is assumed that there is friction between the feet and the ground such that they do not slide. This is verified through simulation by calculating the horizontal reaction force on the trailing foot. The horizontal reaction forces are assumed to act only on one foot because otherwise the horizontal forces acting on the frame would be indeterminate. The free body diagrams during double stance are very similar to those in single stance with the key difference being an additional set of reaction forces on the second foot. Since the frame is not moving, the $\hat{u}_g$ and $\hat{u}_l$ coordinate system shown in Fig. 2.3 is unnecessary. Free body diagrams for the system during double stance are shown in Fig. 2.4.

![Free body diagrams](image)

Frame (f)  Inertia Wheel (i)  Drum (d)

Fig. 2.4: Free body diagram of double stance phase for the frame, inertia wheel and drum. ($i,j,k$ is the fixed coordinate system)

As with single stance, summation of forces and moments for each component are used to solve for the equations of motion for double stance.
\[ \ddot{\vec{d}}_{CM} = 0\hat{i} + 0\hat{j} \] (2.21)

For the frame:

\[ \sum \vec{F} = m_f \vec{d}_{CM} = F_y\hat{j} + F_x\hat{i} - m_fg\hat{j} + R_{1y}\hat{j} + R_{2y}\hat{j} + R_{1x}\hat{i} \] (2.22)

\[ \sum \vec{M}_{f/CM} = I_{f/CM}\vec{d}_f = \tau_m\hat{k} + R_{2y}l\sin(\pi/n)\hat{k} + R_{1x}l\cos(\pi/n)\hat{k} - R_{1y}l\sin(\pi/n)\hat{k} \] (2.23)

For the inertia wheel:

\[ \sum \vec{F} = m_i \vec{d}_{CM} = -S_x\hat{i} - S_y\hat{j} - m_ig\hat{j} \] (2.24)

\[ \sum \vec{M}_{i/CM} = I_{i/CM}\vec{d}_i = \tau_d\hat{k} + \tau_s\hat{k} \] (2.25)

For the drum:

\[ \sum \vec{F} = m_d \vec{d}_{CM} = -F_x\hat{i} - F_y\hat{j} + S_x\hat{i} + S_y\hat{j} - m_dg\hat{j} \] (2.26)

\[ \sum \vec{M}_{d/CM} = I_{d/CM}\vec{d}_d = -\tau_m\hat{k} - \tau_s\hat{k} \] (2.27)

Writing the equations as separate \( \hat{i}, \hat{j} \) and \( \hat{k} \) components produces the following nine scalar equations:

\( \hat{i} : \)

\[ F_x + R_{1x} = 0 \] (2.28)

\[ -S_x = 0 \] (2.29)

\[ S_x - F_x = 0 \] (2.30)

\( \hat{j} : \)

\[ R_{1y} + R_{2y} + F_y - m_fg = 0 \] (2.31)
\[-S_y - m_i g = 0 \quad (2.32)\]
\[S_y - F_y - m_d g = 0 \quad (2.33)\]

\[\dot{k} : \]
\[\tau_m + R2_y l \sin(\pi/n) + R1_x l \cos(\pi/n) - R1_y l \sin(\pi/n) = 0 \quad (2.34)\]
\[\tau_s + \tau_d = -I_i (\ddot{\phi} + \ddot{\theta}) \quad (2.35)\]
\[\tau_s - \tau_m = -I_d \ddot{\phi} \quad (2.36)\]

Some unknowns can be solved for by inspection. The forces \(F_x, S_x, R1_x\) from equations 2.28, 2.29 and 2.30 are all equal to zero, along with \(\dot{\beta}\) and \(\ddot{\beta}\) since the frame is stationary. Therefore, all of the equations for the \(i\) components have already been solved. Additionally, \(F_y\) and \(S_y\) can be eliminated by adding all of the \(j\) component equations together. The resulting matrix consists of only four unknowns \((R1_y, R2_y, \tau_m, \ddot{\theta})\). The following matrix equation in the form \([C] \vec{y} = \vec{d}\) is used to solve for the four unknowns.

\[
\begin{bmatrix}
1 & 1 & 0 & 0 \\
-l \sin(\pi/n) & l \sin(\pi/n) & 1 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & I_i \\
\end{bmatrix}
\begin{bmatrix}
R1_y \\
R2_y \\
\tau_m \\
\ddot{\theta} \\
\end{bmatrix}
= 
\begin{bmatrix}
g(m_d + m_f + m_i) \\
0 \\
I_d \ddot{\phi} - \tau_s \\
-\tau_s - \tau_d - I_i \ddot{\phi} \\
\end{bmatrix} 
\quad (2.37)
\]

The unknown variables can be determined by solving the linear system for \(\vec{y}\).

\[
\vec{y} = [C]^{-1} \vec{d} 
\quad (2.38)
\]
2.6 Event Detection

There are two critical events that determine the transitions between single and double stance. When the angle of the frame corresponds to the collision of the leading foot, the collision transition equations are applied and the system transitions from single to double stance as previously discussed. When the vertical reaction force on the trailing leg becomes zero, the system transitions back to single stance as the trailing leg lifts off the ground. Event detection is needed in order to switch between the governing equations for the forces and accelerations of single and double stance.

The frame angle is calculated based on the number of legs specified and determined based on the equations of motion for single stance. The vertical reaction force of the trailing leg is determined based on the equations of motion for double stance. Since there is no collision during lift off, all angular velocities remain continuous and are calculated using the equations of motion.

MATLAB’s built in event detection for the ode45 integration technique was used to solve for the velocity and position values. These events are stored in separate functions which automatically account for changes in system parameters.

2.7 Collision Transition Equations

The transition from single to double stance occurs when the leading foot hits the ground. The collision is modeled as an instantaneous perfectly plastic collision. Because the device enters double stance directly from single stance, momentum is not conserved about the point where the leading foot hits the ground. The angular velocity of the frame before it hits the ground is known, and is defined to be zero after the collision. The relative angular velocity of the drum is defined as a continuous function of time. Since the collision is instantaneous, the relative angular velocity of the drum after the collision is the same as before the collision. The last angular velocity that
remains unknown is that of the inertia wheel. Isolating the inertia wheel, angular momentum is conserved about the axle located at the center of mass as shown in equation 2.39. The plus and minus sign superscripts indicate values just before (-) and after (+) the collision.

\[ -I_i(\dot{\beta}^- + \dot{\phi}^- + \dot{\theta}^-)\hat{k} = -I_i(\dot{\beta}^+ + \dot{\phi}^+ + \dot{\theta}^+)\hat{k} \]  
\[ \dot{\beta}^+ = 0 \]  
\[ \dot{\phi}^+ = \dot{\phi}^- \]  

Given equations 2.40 and 2.41, equation 2.39 simplifies to the following:

\[ \dot{\theta}^+ = \dot{\theta}^- + \dot{\beta}^- \]  

The collection of equations 2.40, 2.41 and 2.42 are used to solve for the values of angular velocity after the collision. These values are used as initial conditions for the double stance phase of motion.

### 2.8 Energy Calculations

In a study of energy efficiency, it is important to define how energy is calculated. This model contains both gravitational and spring potential energy. Total potential energy at any given point in time is calculated using equation 2.44.

\[ h = l \sin(\beta) \]  
\[ PE = (m_f + m_d + m_i)gh + k\theta^2/2 \]
Total kinetic energy accounts for the motion of the center of mass of each component in addition to its absolute rotation as shown in equation 2.45.

\[ KE = (m_f + m_d + m_i)(\dot{\beta})^2/2 + (I_f\ddot{\beta}^2 + I_d(\dot{\beta} + \phi)^2 + I_i(\dot{\beta} + \phi + \theta)^2)/2 \] (2.45)

Total energy is found by adding potential and kinetic energy.

\[ TE = PE + KE \] (2.46)

Power provided by the motor to the system is calculated by multiplying motor torque by the angular velocity of the drum relative to the frame. Integrating power with respect to time gives the energy provided by the motor during that time interval.

\[ P_m = \tau_m \dot{\phi} \] (2.47)

\[ E_m = \int_{t_0}^{t_1} P_m dt = \int_{t_0}^{t_1} \tau_m \dot{\phi} dt \] (2.48)

Energy required as used in calculating the cost of transport is found by integrating the absolute value of power provided by the motor.

\[ E_{req} = \int_{t_0}^{t_1} |P_m| dt = \int_{t_0}^{t_1} |\tau_m \dot{\phi}| dt \] (2.49)

If power is negative at any given point in time, the system is performing work on the motor at that moment. Work performed on the motor by the system is called negative work and is considered an energy loss.

\[ W_{neg} = \begin{cases} \int_{t_0}^{t_1} P_m dt, & \text{if } P_m \leq 0 \\ 0, & \text{if } P_m > 0 \end{cases} \] (2.50)
Positive work is work performed by the motor on the system.

\[
W_{pos} = \begin{cases} 
\int_{t_0}^{t_1} P_m \, dt, & \text{if } P_m \geq 0 \\
0, & \text{if } P_m < 0
\end{cases}
\] (2.51)

Energy lost due to air drag is calculated by integrating the product of the drag torque and the absolute angular velocity of the inertia wheel.

\[
E_{\text{drag}} = \int_{t_0}^{t_1} T_d (\dot{\beta} + \dot{\phi} + \dot{\theta}) \, dt
\] (2.52)

Energy lost during collisions is calculated by subtracting the total energy of the system before the collision from the total energy after the collision. The time right before the collision is denoted as \( t^+ \) and the time right after as \( t^- \).

\[
E_{\text{col}} = TE(t^+) - TE(t^-)
\] (2.53)

For positional control, the motor torque instantaneously goes to infinity during the collision. Simply calculating the total energy before and after the collision does not account for the power expended by the motor during the collision. In order to account for the energy lost by the motor in supplying infinite torque, the power needs to be integrated with respect to the time of collision.

\[
E_m = \int_{t^-}^{t^+} P_m \, dt = \int_{t^-}^{t^+} \tau_m \dot{\phi} \, dt
\] (2.54)

As previously defined in equation 2.41, the angular velocity of the drum relative to the frame remains constant and can therefore be taken outside of the integral.

\[
E_m = \dot{\phi} \int_{t^-}^{t^+} \tau_m \, dt
\] (2.55)
The remaining integral defines the angular impulse due to the motor which is equal
to the change in absolute angular momentum.

\[
\int_{t^{-}}^{t^{+}} \tau_m \, dt = I_d(\dot{\beta}^+ + \dot{\phi}^+) - I_d(\dot{\beta}^- + \dot{\phi}^-)
\] (2.56)

Therefore, the energy expended by the motor can be calculated using equation 2.57

\[
E_m = -I_d\dot{\phi}\dot{\beta}^-
\] (2.57)

2.9 Simulation Methodology

The positions of each component can be solved as a function of time by numerically
integrating \( \ddot{\beta} \) and \( \ddot{\theta} \) based on initial conditions with the assumption that \( \phi(t) \) is a
known function of time. The ode45 method built into MATLAB uses a fourth order
variable time step Runge-Kutta method of integration. This function is used with
options to set the integration accuracy and event detection. An integration tolerance
of \( \text{RelTol} = \text{AbsTol} = 10^{-8} \) is used for both single and double stance. Fig. 2.5

![Integration Tolerance Plot](image)

Fig. 2.5: Integrated values for \( \theta \) fluctuate for different tolerances values until \( \text{RelTol} = \text{AbsTol} = 10^{-8} \)
shows a convergence plot to support the use of this value for integration tolerance.

The value of \( \theta \) at the end of a step for identical initial conditions is plotted against
the logarithmic value for integration tolerance. For values smaller than $10^{-5}$, the corresponding value for $\theta$ at the end of the step remains consistent.

Actuation is added to the system by defining the angular position of the drum relative to the inertia wheel as a function of time. Instead of integrating $\ddot{\phi}$ using initial conditions to find $\dot{\phi}$ and $\phi$, appropriate equations were used for each parameter. The control is modeled as a sine function with amplitude ($A$) in radians, period ($T$) in seconds and phase shift ($P$) in radians.

$$\phi = A \sin(2\pi/Tt + P)$$ (2.58)

$$\dot{\phi} = -2A\pi/T \cos(2\pi/Tt + P)$$ (2.59)

$$\ddot{\phi} = -4A\pi^2/T^2 \sin(2\pi/Tt + P)$$ (2.60)

Energy losses are represented as a linear function of the absolute angular velocity of the inertia wheel ($\tau_m = c(\beta + \phi + \theta)$) similar to the air drag model used by Ahlin [17]. These energy losses will be referred to as air drag.

The spring torque is modeled as a linear relationship between the torsional spring constant and the relative position of the inertia wheel to the drum ($\tau_s = k\theta$).

For the purpose of animation and tracking the motion of individual steps, the angles in the coordinate system were reset at the end of every step as shown in Fig. 2.6.
Fig. 2.6: Definition of angles before a step, after a step and after the coordinate reset.
Chapter 3

Research Approach

The goal of this simulation is to find motions for an inertia-coupled rimless wheel with energy losses due to air drag and actuation. After defining the equations and series of events, reasonable initial conditions must be selected to get the device to take a step. The first task is simply getting off the ground and out of single stance. From previous simulations and experiments, it is known that winding up the inertia wheel a large enough amount will produce a moment on the frame causing it to lift off the ground. So the input function for $\phi$ which defines the position of the drum relative to the frame was set to zero and $\dot{\theta}_0$ was given a large value. For sufficiently large values of $\dot{\theta}_0$, device successfully lifts off the ground and enters double stance.

However, the step is not complete unless it can successfully get out of double stance by lifting off the ground again. By adjusting the initial conditions through trial and error, values were found that achieved a complete step. Although a single step is found, there is no guarantee that a second step will occur.

For the purposes of this research, it is not particularly useful to find a single step until multiple steps can be achieved. The system can take multiple steps if given an initial push. This is achieved by giving an large initial $\dot{\beta}_0$ value. In the absence of air drag and actuation ($c = 0, A = 0$), the system is capable of converging to periodic motions for certain sets of initial conditions and parameters listed in Table 3.1.

For a system with no drag and no power, the angular velocity of the drum is fixed
Table 3.1: Initial conditions that converge to periodic motion for no drag and no power at zero as shown in Fig. 3.1. As expected, there is a jump discontinuity in $\dot{\beta}$ and $\dot{\theta}$ at each collision.

![Component velocities for five steps with no drag and no power](image)

As expected, the total energy of the system remains constant except at collisions where the energy drops as shown in Fig. 3.2.

The work up to this point has only validated previous simulations of the inertia-coupled rimless wheel, although for a different set of parameters and slight variations in model dynamics [17]. After adding the air drag model ($c = 0.004$), a sinusoidal function is provided as the control function for $\phi$ as shown in equation 3.1. The period of the function is selected to match that of the period of the inertia wheel oscillation.

$$\phi = A \sin\left(2\pi/T + P\right) \rightarrow \phi = 0.1 \sin\left(2\pi/1 + 2\right)$$  \hspace{1cm} (3.1)
The system is wound up and given a push as was done without air drag or actuation. After some trial and error, initial conditions were found that caused the device to take several steps as listed in Table 3.2.

<table>
<thead>
<tr>
<th>Initial Condition</th>
<th>Value</th>
<th>Units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta_0$</td>
<td>$(0.5 - 1/n)\pi$</td>
<td>rad</td>
<td>Frame initial angle</td>
</tr>
<tr>
<td>$\dot{\beta}_0$</td>
<td>10</td>
<td>rad/s</td>
<td>Frame initial angular velocity</td>
</tr>
<tr>
<td>$\theta_0$</td>
<td>1.3</td>
<td>rad</td>
<td>Inertia wheel initial angle</td>
</tr>
<tr>
<td>$\dot{\theta}_0$</td>
<td>13</td>
<td>rad/s</td>
<td>Inertia wheel initial angular velocity</td>
</tr>
<tr>
<td>$\phi_0$</td>
<td>0.0909</td>
<td>rad</td>
<td>Drum initial angle</td>
</tr>
<tr>
<td>$\dot{\phi}_0$</td>
<td>$-0.2615$</td>
<td>rad/s</td>
<td>Drum initial angular velocity</td>
</tr>
</tbody>
</table>

Table 3.2: Initial conditions with drag and power that converge to periodic motion

For a system with drag and power, $\dot{\phi}$ is defined as a sinusoidal function of time as shown in Fig. 3.3. The motion transitions from a transient phase towards periodic motion as shown by the repetitive pattern of $\dot{\theta}$. There is still a jump discontinuity in the velocity of the frame and inertia wheel at each collision.

Unlike the system with no drag and no power, the total energy of the system does not remain constant between each collision as shown in Fig. 3.4. The total energy even increases between collisions due to the power input to the system from the motor between the drum and the frame.

It can be observed that the last several steps seem to be repetitive. The initial
conditions after fifty steps, as listed in Table 3.3, are used as the initial conditions for the simulation.

Plotting the relative inertia wheel velocity against position shows a projection of the phase plane trajectory as shown in Fig. 3.5. The data repeats in a circular pattern, but looking closer at the place where the collision repeats shows that each step does not quite line up with the previous step. Looking at the total energy across fifty steps as seen in Fig. 3.6 shows that the initial conditions do not match the next consecutive step, but are similar to steps later on.

The repetitive nature of the steps eludes to what is known as periodic motion.
<table>
<thead>
<tr>
<th>Initial Condition</th>
<th>Value</th>
<th>Units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>β₀</td>
<td>(0.5 − 1/n)π</td>
<td>rad</td>
<td>Frame initial angle</td>
</tr>
<tr>
<td>β̇₀</td>
<td>0</td>
<td>rad/s</td>
<td>Frame initial angular velocity</td>
</tr>
<tr>
<td>θ₀</td>
<td>1.3071</td>
<td>rad</td>
<td>Inertia wheel initial angle</td>
</tr>
<tr>
<td>θ̇₀</td>
<td>14.9486</td>
<td>rad/s</td>
<td>Inertia wheel initial angular velocity</td>
</tr>
<tr>
<td>φ₀</td>
<td>0.0909</td>
<td>rad</td>
<td>Drum initial angle</td>
</tr>
<tr>
<td>φ̇₀</td>
<td>−0.2615</td>
<td>rad/s</td>
<td>Drum initial angular velocity</td>
</tr>
</tbody>
</table>

Table 3.3: Initial conditions after convergence with drag and power

Fig. 3.5: Phase plane trajectory for fifty steps after convergence. Part (a) shows the entire trajectory while part (b) shows the zoomed in section where the collision occurs.

Fig. 3.6: Total energy for fifty steps after convergence

This is defined when the all the conditions at the end of the step match the initial conditions. Certain variables are periodic by definition of the step. The frame position
at the beginning and end of the step is known as a function of the number of legs, and the velocity of the frame at both instances in time is defined to be zero. It is also assumed that none of the physical parameters of the system change from step to step. There are only four remaining variables that must be matched. The relative angular position and velocity of the inertia wheel must match. The relative position and velocity of the drum are defined by the control function. Therefore, the control function itself must be periodic. This known periodicity must match the period of each step.

While the method of giving the wheel an initial shove and looking for convergence works well intuitively and visually, there are other mathematical approaches to looking for periodic motions that can be more computationally efficient. One such method is the multidimensional numerical root find. In this case, a root defines the initial conditions that produce periodic motions for a given set of parameters. Using \( \theta, \dot{\theta} \) and the period \( (T) \) as conditions for convergence, the following three equations are defined where subscript \( i \) and \( f \) indicate values at the beginning and end of the step respectively:

\[
\begin{align*}
  f &= \theta_f - \theta_i \\
  g &= \dot{\theta}_f - \dot{\theta}_i \\
  h &= T - (t_f - t_i)
\end{align*}
\] (3.2) (3.3) (3.4)

The goal is to bring the value of these three equations to zero by varying other parameters. The parameters selected to vary are \( \theta, A \) and \( T \). Any parameters could be varied so long as they stay within realistic conditions and have some affect on the output values of all the functions. The analytical approach uses the Jacobian matrix as defined in equation 3.5.
However, because the functions are difficult to define analytically, a numerical approach is used instead.

\[
J = \begin{bmatrix}
\frac{\delta f}{\theta} & \frac{\delta f}{A} & \frac{\delta f}{T} \\
\frac{\delta g}{\theta} & \frac{\delta g}{A} & \frac{\delta g}{T} \\
\frac{\delta h}{\theta} & \frac{\delta h}{A} & \frac{\delta h}{T}
\end{bmatrix}
\]  

(3.5)

The values of the functions are first evaluated and then reevaluated after making small variations to the parameters independently. The difference between these function evaluations divided by the amount each parameter was varied by results in a forward difference approximation to the specified partial derivative. This is represented as the numerical approximation of the Jacobian shown in equation 3.9. Inverting the Jacobian, multiplying by the step changes and adding back the initial conditions moves a single iteration closer to the root by sliding down the steepest part of the three dimensional gradient. This process is repeated until the values of the functions reach zero or within some reasonable tolerance [21].
\[
\begin{bmatrix}
\theta^* \\
A^* \\
T^*
\end{bmatrix} = -J^{-1}
\begin{bmatrix}
\theta_f - \theta_i \\
\dot{\theta}_f - \dot{\theta}_i \\
T - (t_f - t_i)
\end{bmatrix}
+ \begin{bmatrix}
\theta_i \\
A_i \\
T_i
\end{bmatrix}
\]

(3.10)

Using the initial conditions from the converged steps as the starting point for the root find yielded a root with conditions listed in Table 3.4 and Table 3.5. The phase plane trajectory shown in Fig. 3.7 shows that initial conditions exactly match final conditions for a single step.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.0805</td>
<td>rad</td>
<td>Amplitude for sine function</td>
</tr>
<tr>
<td>T</td>
<td>0.9467</td>
<td>s</td>
<td>Period for sine function</td>
</tr>
<tr>
<td>P</td>
<td>2.3562</td>
<td>rad</td>
<td>Phase shift for sine function</td>
</tr>
</tbody>
</table>

Table 3.4: Control parameters found by root find

<table>
<thead>
<tr>
<th>Initial Condition</th>
<th>Value</th>
<th>Units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \beta_0 )</td>
<td>( (0.5 - 1/n)\pi )</td>
<td>rad</td>
<td>Frame initial angle</td>
</tr>
<tr>
<td>( \dot{\beta}_0 )</td>
<td>0</td>
<td>rad/s</td>
<td>Frame initial angular velocity</td>
</tr>
<tr>
<td>( \theta_0 )</td>
<td>1.3077</td>
<td>rad</td>
<td>Inertia wheel initial angle</td>
</tr>
<tr>
<td>( \dot{\theta}_0 )</td>
<td>14.9488</td>
<td>rad/s</td>
<td>Inertia wheel initial angular velocity</td>
</tr>
<tr>
<td>( \phi_0 )</td>
<td>0.0569</td>
<td>rad</td>
<td>Drum initial angle</td>
</tr>
<tr>
<td>( \dot{\phi}_0 )</td>
<td>-0.3778</td>
<td>rad/s</td>
<td>Drum initial angular velocity</td>
</tr>
</tbody>
</table>

Table 3.5: Initial conditions found by root find

It is possible that there are other roots for the given set of parameters. If other roots do exist, they can be found by choosing different starting points that are closer to those new roots than to the root that was found.
Fig. 3.7: Phase plane trajectory of motion found using root find
Chapter 4

Torque Control

During collisions using positional control, an infinite instantaneous torque is required in order to maintain the drum position and velocity. This requirement is not physically possible to maintain. An alternative method of modeling the actuation is by directly specifying the torque supplied by the motor between the frame and the drum instead of specifying the relative position of the drum with respect to the frame. This method has some key differences from the position control simulation. Firstly, the equations of motion need to be modified to move $\ddot{\phi}$ into the state vector and $\tau_m$ out as a known variable. The modified matrices for single and double stance are shown in equations 4.1 and 4.2 respectively. Defining motor torque as a function of time, $\ddot{\phi}$ needs to be integrated along with the other state vectors in order to solve for $\dot{\phi}$ and $\phi$ at any given point in time.
\[
\begin{bmatrix}
  m_f l \sin(\beta) & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 \\
  m_f l \cos(\beta) & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\
  -I_{f/CM} & 0 & 0 & 0 & -l \sin(\beta) & -l \cos(\beta) & 0 & 0 \\
  m_d l \sin(\beta) & 0 & 0 & 1 & 0 & 0 & 0 & -1 \\
  m_d l \cos(\beta) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  -I_{d/CM} & -I_{d/CM} & 0 & 0 & 0 & 0 & 0 & 0 \\
  m_i l \sin(\beta) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  m_i l \cos(\beta) & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  -I_{i/CM} & -I_{i/CM} & -I_{i/CM} & 0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
  \ddot{\beta} \\
  \ddot{\phi} \\
  F_x \\
  R_x \\
  S_x \\
  F_y \\
  R_y \\
  S_y \\
\end{bmatrix}
= 
\begin{bmatrix}
  -m_f l \beta^2 \cos\beta \\
  m_f l \beta^2 \sin\beta - m_f g \\
  -m_d l \beta^2 \cos\beta \\
  m_d l \beta^2 \sin\beta - m_d g \\
  -m_i l \beta^2 \cos\beta \\
  m_i l \beta^2 \sin\beta - m_i g \\
  \tau_s + \tau_d \\
\end{bmatrix}
\] (4.1)

Unlike the collision modeled using position control, the angular velocity of the drum relative to the frame is not known after the collision when using torque control. Consequently, two sets of equations are required to solve for \( \dot{\phi}^+ \) and \( \dot{\theta}^+ \). The following two equations are derived from the conservation of angular momentum at the axle located at the center of mass for the drum and the inertia wheel.

\[
\begin{bmatrix}
  1 & 1 & 0 & 0 \\
  -l \sin\pi/n & l \sin\pi/n & 0 & 0 \\
  0 & 0 & -I_i & -I_i \\
  0 & 0 & I_d & 0 \\
\end{bmatrix}
\begin{bmatrix}
  R_{1y} \\
  R_{2y} \\
  \ddot{\phi} \\
  \ddot{\theta} \\
\end{bmatrix} = 
\begin{bmatrix}
  g(m_d + m_f + m_i) \\
  -\tau_m \\
  \tau_s + \tau_d \\
  \tau_s + \tau_m \\
\end{bmatrix}
\] (4.2)

As previously defined, the angular velocity of the frame after the collision is zero:

\[
\dot{\theta}^+ = 0
\] (4.5)
Solving equations 4.3 and 4.4 given equation 4.5, yields the remaining two post collision equations:

\[ \dot{\phi}^+ = \dot{\beta}^- + \dot{\phi}^- \]  
(4.6)

\[ \dot{\theta}^+ = \dot{\theta}^- \]  
(4.7)

The collection of equations 4.5, 4.6 and 4.7 is used to solve for the values of angular velocity after the collision. These values are used as initial conditions for the double stance phase of motion.

Attempting to find stable gaits by giving the frame an initial shove is not as effective for the torque control model as it was for position control. This is due to the fact that it is difficult to know how much torque is needed at any given time in order to maintain the position between the drum and frame. A more reliable method is to take the torque profile from a periodic step found using position control and approximate the torque profile for that motion in the torque control simulation. Since the torque profile produced by the position control simulation so closely resembled a sinusoidal wave, the coefficients of a best fit line shown in Fig. 4.1 were used as the motor torque control parameters.

![Fig. 4.1: Torque profile from a single periodic step using position control along with sinusoidal curve of best fit.](image)

The input for the motor torque is defined by the function \( \tau_m = A \sin(Ft + P) \)
where the values of A, F, and P are listed in Table 4.1.

<table>
<thead>
<tr>
<th>Control parameter</th>
<th>Value</th>
<th>Units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10.51</td>
<td>N-m</td>
<td>Amplitude of torque sine function</td>
</tr>
<tr>
<td>F</td>
<td>6.651</td>
<td>rad/s</td>
<td>Frequency of torque sine function</td>
</tr>
<tr>
<td>P</td>
<td>-2.606</td>
<td>rad</td>
<td>Phase shift of torque sine function</td>
</tr>
</tbody>
</table>

Table 4.1: Parameters that define the motor torque function with respect to time.

Since the torque control function used was based on coefficients from a curve of best fit, the values produced did not correspond exactly with those calculated by the position control simulation. However, the torque control simulation successfully produced a step that was very similar. The resulting step was not periodic, as evidenced by Fig. 4.2(a). The phase plane trajectory does not repeat itself, indicating that the second step was not the same as the first step.

The root find method used previously can also be applied to look for periodic motions in the case of torque control. However, the functions used to determine a root crossing are slightly different from position control. Since the motor torque is defined, the position of the drum (\(\phi\)) does not need to match beginning to end. Also, the motor torque (\(\tau_m\)) can instantaneously change unlike angular position. However, the absolute velocity of the inertia wheel must be maintained as well as its relative position to the drum, so \(\dot{\phi}, \theta\) and \(\dot{\theta}\) must all match beginning to end. As with position control, \(\theta_0\), \(A\) and \(F\) were used as parameters to vary. The final values at the end of the root find for these three parameters are \(\theta_0 = 1.3153\), \(A = 10.3309\), and \(F = 6.5893\). Fig. 4.2(b) shows a projection indicating that final conditions match initial conditions.

In constructing a physical device demonstrating the AIRWAFT model, defining the torque as an input aids with motor specification. Using position control may lead to motions that require infinite instantaneous torque values which cannot be affectively performed by standard motors. Therefore, the torque control simulation may be of more use in constructing a physical system. Both simulations yield comparable results.
Fig. 4.2: Projection of phase plane trajectory on the $\theta - \dot{\theta}$ plane for (a) the replicated step from position control and (b) the step found using a root find in torque control.

to each other.
Chapter 5

Parameter Study

Various performance metrics can be used to evaluate the periodic motion found by the root find. The energy required to power the device is defined as the absolute value of the energy expended by the motor. It is assumed that the any work performed by the system on the motor is considered an expense rather than a gain since current is still consumed by any activated motor. As mentioned before, the cost of transport is a non-dimensional measure of efficiency for moving devices. It is also useful to compare this efficiency with the forward speed. Maximum motor torque, angular position, and angular velocity of the inertia wheel provide a good indicator of the physical feasibility of the motion by comparing them to motor specifications. It is also useful to determine how much energy is lost due to air drag and collisions.

The periodic root discovered using torque control defines periodic motion for a given set of parameters. It is helpful to see how this root might change if these parameters are modified. Specific parameters of interest include the damping coefficient, spring constant, number of feet, mass of frame, mass moment of inertia of the inertia wheel, initial velocity of inertia wheel, and phase shift of the control function. The base set of geometrical parameters can be found in Table 2.1 with initial conditions and control function parameters listed in Table 5.1 and Table 5.2.

For each parameter study, the value of the parameter is slightly altered in both directions and a root find is performed using the new value. This traces the root with
<table>
<thead>
<tr>
<th>Initial Condition</th>
<th>Value</th>
<th>Units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta_0$</td>
<td>$(0.5 - 1/n)\pi$</td>
<td>rad</td>
<td>Frame initial angle</td>
</tr>
<tr>
<td>$\dot{\beta}_0$</td>
<td>0</td>
<td>rad/s</td>
<td>Frame initial angular velocity</td>
</tr>
<tr>
<td>$\theta_0$</td>
<td>1.3153</td>
<td>rad</td>
<td>Inertia wheel initial angle</td>
</tr>
<tr>
<td>$\dot{\theta}_0$</td>
<td>14.9488</td>
<td>rad/s</td>
<td>Inertia wheel initial angular velocity</td>
</tr>
<tr>
<td>$\phi_0$</td>
<td>0.0569</td>
<td>rad</td>
<td>Drum initial angle</td>
</tr>
<tr>
<td>$\dot{\phi}_0$</td>
<td>-0.3778</td>
<td>rad/s</td>
<td>Drum initial angular velocity</td>
</tr>
</tbody>
</table>

Table 5.1: Initial conditions as used for parameter study

<table>
<thead>
<tr>
<th>Control parameter</th>
<th>Value</th>
<th>Units</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10.3309</td>
<td>N-m</td>
<td>Amplitude of torque sine function</td>
</tr>
<tr>
<td>F</td>
<td>6.5893</td>
<td>rad/s</td>
<td>Frequency of torque sine function</td>
</tr>
<tr>
<td>P</td>
<td>3.6772</td>
<td>rad</td>
<td>Phase shift of torque sine function</td>
</tr>
</tbody>
</table>

Table 5.2: Parameters that define the motor torque function with respect to time as used for parameter study

respect to that one parameter.
5.1 Number of legs

Although it would be difficult to construct a device with non-integer values for the number of legs, the analytical nature of the simulation is able to produce motions for such values. Since the initial root find is based on an eight-legged device, the lowest cost of transport remains where the number of legs is exactly eight as shown in Fig. 5.1. Periodic roots do not exist for other integer values for the number of legs. As the value for the number of legs increases, the speed decreases. The energy lost due to collisions decreases as the number of legs increases as shown in Fig. 5.2. This is because the step length is shorter and the center of mass does not vary in vertical position as much, leading to lower impacts. Energy lost due to air drag also decreases as the number of legs increases. Since the step length decreases, the oscillatory motion of the inertia wheel is reduced. Consequently, the effect of air drag on the inertia wheel is also reduced.
Fig. 5.2: Energy losses due to air drag and collisions increases as the number of legs increases

5.2 Damping coefficient

The cost of transport increases and the speed decreases as the damping coefficient increases as shown in Fig. 5.3. Increasing the damping coefficient takes more energy out of the system. Consequently, more energy is required for the device to take a step, which increases the cost of transport. Energy losses due to air drag and collisions
both increase as the damping coefficient increases as shown in Fig. 5.4. Although there is a direct linear relationship between the damping coefficient and air drag, the relationship between changes in the damping coefficient and the motions produced may not be linear. Consequently, the energy lost due to air drag is not linearly related to variation in the damping coefficient for this root.

![Graph showing energy losses due to air drag and collisions](image)

**Fig. 5.4:** Energy losses due to air drag and collisions both increase as the damping coefficient increases

### 5.3 Phase shift of control function

Similar to the trend for the number of legs, the lowest cost of transport occurs at the original value for phase shift found by the initial root find as shown in Fig. 5.5. Speed increases as the phase shift increases. Phase shift has a large impact on the energy lost due to collisions as seen in Fig. 5.6. Energy lost due to collisions increases until a value of 3.725 radians after which it begins to decrease again. Energy lost due to air drag increases as phase shift increases.
5.4 Initial velocity of inertia wheel

Giving the inertia wheel an initial velocity that is more than necessary results in higher initial kinetic energy, which is lost during motion. This results in a higher cost of transport as shown in Fig. 5.7. Increasing the initial angular velocity of the inertia wheel also increases the speed. Energy losses due to both air drag and collisions are generally greater at higher values for the initial velocity of the inertia wheel as shown in Fig. 5.8. For values of $\dot{\theta}_0$ lower than 15.24 [rad/s], energy lost due to air drag increases as $\dot{\theta}_0$ decreases.
5.5 Mass of frame

Since the cost of transport is a measure of efficiency scaled by the mass of the system, it makes sense that the cost of transport decreases as the mass of the system increases, as shown in Fig. 5.9. However, the speed also decreases as the mass of the frame increases. Adjusting the mass of the frame most likely has a similar affect as changing the mass of the entire system. The lowest cost of transport occurs when the mass of
the frame is equal to 1.455 [kg]. As the mass of the frame increases, more energy is

![Graph showing cost of transport (COT) and speed against mass of frame]  

Fig. 5.9: Cost of transport and speed both decrease as the mass of the frame increases

lost due to collisions and air drag as shown in Fig. 5.10.

![Graph showing energy lost against mass of frame]  

Fig. 5.10: Energy losses due to collisions and air drag both increase as the mass of the frame increases

5.6 Mass moment of inertia for inertia wheel

The cost of transport increases and the speed decreases as the mass moment of the inertia wheel increases as shown in Fig. 5.11. Increasing the mass moment of inertia
Fig. 5.11: Cost of transport increases and speed decreases as the mass moment of inertia for the inertia wheel is increased for the inertia wheel results in higher energy losses due to both air drag and collisions as shown in Fig. 5.12.

Fig. 5.12: Energy losses due to air drag and collisions both increase as the mass moment of inertia for the inertia wheel increases
5.7 Spring constant

Speed decreases as the mass moment of the inertia wheel increases as shown in Fig. 5.13. The cost of transport increases for spring constant values further away from \( k = 4 \) [N-m/rad] on either side. Increasing the spring constant results in lower energy losses due to both air drag and collisions as shown in Fig. 5.14.

![Graph showing COT vs. Spring constant](image1)

![Graph showing Speed vs. Spring constant](image2)

![Graph showing Energy loss vs. Spring constant](image3)

Fig. 5.13: Speed increases as the spring constant increases. the lowest cost of transport occurs when \( k = 4 \) [N-m/rad].

Fig. 5.14: Energy losses due to air drag and collisions decrease as the spring constant is increased.
5.8 Minimizing COT

The purpose of the variation of parameters is to look for the lowest achievable value for cost of transport with the given physical parameters. It is interesting to see the effect that the number of legs and the damping coefficient has on the performance of the system, but the model of interest is defined as an eight-legged device subject to a damping coefficient of $0.004 \ [N \cdot m \cdot s/\text{rad}]$. In order to find the lowest cost of transport value among the remaining parameters, a larger spring constant, lower initial velocity of the inertia wheel, higher mass of frame, and lower mass moment of inertia for the inertia wheel is preferred.

The performance metrics of the motion producing the lowest cost of transport found are listed in Table 5.3.

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>COT</td>
<td>0.3267</td>
<td>-</td>
</tr>
<tr>
<td>Speed</td>
<td>0.3161</td>
<td>m/s</td>
</tr>
<tr>
<td>Max. $T_m$</td>
<td>10.331</td>
<td>N-m</td>
</tr>
<tr>
<td>Max. $\theta$</td>
<td>2.5905</td>
<td>rad</td>
</tr>
<tr>
<td>Max. $\dot{\theta}$</td>
<td>16.7018</td>
<td>rad/s</td>
</tr>
<tr>
<td>Energy required</td>
<td>3.421</td>
<td>J</td>
</tr>
<tr>
<td>Energy in</td>
<td>2.9319</td>
<td>J</td>
</tr>
<tr>
<td>Energy out</td>
<td>-0.4892</td>
<td>J</td>
</tr>
<tr>
<td>Energy lost to air drag</td>
<td>-0.4875</td>
<td>J</td>
</tr>
<tr>
<td>Energy lost due to collisions</td>
<td>-1.9549</td>
<td>J</td>
</tr>
</tbody>
</table>

Table 5.3: Performance metrics for periodic motion with lowest cost of transport found

The phase plane trajectory shown in Fig. 5.15 shows that the motion is indeed periodic as final conditions match initial conditions. Total energy remains within the range of 24.5 to 27 Joules as shown in Fig. 5.16. Angular positions of all components remain within the realistic range of 3 radians in both directions as shown in Fig. 5.17. The motion produces maximum torque values of $10.33 \ [N-m]$ as shown in Fig. 5.18.
Fig. 5.15: Phase plane trajectory of motion with lowest COT found using torque control

Fig. 5.16: Total energy for one step of motion with lowest COT found using torque control

Fig. 5.17: Component angular positions of motion with lowest COT found using torque control
Fig. 5.18: Component angular velocities of motion with lowest COT found using torque control

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A$</td>
<td>10.331</td>
<td>N-m</td>
</tr>
<tr>
<td>$F$</td>
<td>6.5893</td>
<td>rad/s</td>
</tr>
<tr>
<td>$P$</td>
<td>3.6772</td>
<td>rad</td>
</tr>
<tr>
<td>$n$</td>
<td>8</td>
<td>-</td>
</tr>
<tr>
<td>$l$</td>
<td>0.3935</td>
<td>m</td>
</tr>
<tr>
<td>$r$</td>
<td>0.0646</td>
<td>m</td>
</tr>
<tr>
<td>$m_f$</td>
<td>1.438</td>
<td>kg</td>
</tr>
<tr>
<td>$m_i$</td>
<td>1.803</td>
<td>kg</td>
</tr>
<tr>
<td>$m_d$</td>
<td>0.303</td>
<td>kg</td>
</tr>
<tr>
<td>$I_f$</td>
<td>0.0851</td>
<td>kg · m$^2$</td>
</tr>
<tr>
<td>$I_i$</td>
<td>0.0984</td>
<td>kg · m$^2$</td>
</tr>
<tr>
<td>$I_d$</td>
<td>0.0018</td>
<td>kg · m$^2$</td>
</tr>
<tr>
<td>$g$</td>
<td>9.81</td>
<td>m/s$^2$</td>
</tr>
<tr>
<td>$k$</td>
<td>4</td>
<td>N · m/rad</td>
</tr>
<tr>
<td>$c$</td>
<td>0.004</td>
<td>N · m · s/rad</td>
</tr>
<tr>
<td>$\theta_0$</td>
<td>1.3153</td>
<td>rad</td>
</tr>
<tr>
<td>$\dot{\theta}_0$</td>
<td>14.9488</td>
<td>rad/s</td>
</tr>
<tr>
<td>$\beta_0$</td>
<td>1.1781</td>
<td>rad</td>
</tr>
<tr>
<td>$\dot{\beta}_0$</td>
<td>0</td>
<td>rad/s</td>
</tr>
<tr>
<td>$\phi_0$</td>
<td>0.0569</td>
<td>rad</td>
</tr>
<tr>
<td>$\dot{\phi}_0$</td>
<td>-0.3778</td>
<td>rad/s</td>
</tr>
</tbody>
</table>

Table 5.4: Control function parameters, geometric parameters and initial conditions that produce a periodic motion with a cost of transport of 0.3267
Chapter 6

Conclusions

6.1 Discussion of Results

The AIRWAFT model provides a unique method of adding actuation to an inertia-coupled rimless wheel model with energy losses. The actuation can be applied either by defining the relative position of the frame and the drum, or by specifying the torque provided by the motor. A sinusoidal function was the only type of function tested as a method of open-loop control. Stable, periodic motions exist for systems with energy losses modeled as air drag and actuation. The most efficient motion found using positional control has a cost of transport of 0.33 and a speed of 0.3161 [m/s]. This value for the COT is comparable to the lowest COT of 0.3426 found using the simulation of the Central Motor System for inertia-coupled rimless wheel model with energy losses across level ground [19].

6.2 Motor Specification

The maximum power required by the system during the step is 15.55 Watts as shown in Fig. 6.1, and the maximum speed is 16.35 RPM. The 32 RPM HD Premium Planetary Gear Motor from ServoCity outputs a power of around 17.66 Watts with a gear ratio of 264:1. The motion of the device stays below the torque-velocity curve as shown in Fig. 6.2 indicating that the motor is capable of producing that motion.
Fig. 6.1: Power required by the motor for a single periodic step with the lowest COT found. The curve is calculated using a stall torque of 21.08 [N-m] and a no load speed of 32 RPM and assumes that the motor is operating at steady state conditions [22]. As stated in the assumptions, motor dynamics are not modeled in this simulation.

Fig. 6.2: Torque-velocity curve for selected motor along with motion of the device.
6.3 Future Work

The research performed in this thesis could be further explored and extended to increase applicability. A large amount of investigation was centered around a single root that was found. Other roots may exist that could also be found and investigated. A more thorough parameter study could be performed that looks at broader ranges for all system parameters. The objective of these searches is to look for motions with even lower cost of transport values using the same model. Although the motions that were found were assumed to be stable, it would be beneficial to calculate the stability of these motions for verification. Only one type of open-loop controller function was used in this research. Further exploration using other types of functions or closed-loop control may lead to more efficient motions. Ultimately, the knowledge gained from simulating the AIRWAFT model could be used to construct a physical prototype. Additionally, the inertia wheel and actuation methods could be extended to bipedal models in order to improve energy efficiency in existing and future walking robots.
Bibliography


Appendix A

Position control code

A.1 Solving for equations of motion

1  % Single stance
2  syms mf md mi If Id Ii l beta betad betadd;
3  syms thetadd Fx Fy Rx Ry Sx Sy phidd Tm Ts Td g;
4  Ass = sym([mf*l*sin(beta) 0 -1 0 -1 0 0 0; ... 
5       mf*l*cos(beta) 0 0 -1 0 -1 0 0; ... 
6       -If 0 0 0 -l*sin(beta) -l*cos(beta) 0 0 -1; ... 
7       md*l*sin(beta) 0 1 0 0 0 -1 0 0; ... 
8       md*l*cos(beta) 0 0 1 0 0 0 -1 0; ... 
9       -Id 0 0 0 0 0 0 1; ... 
10      mi*l*sin(beta) 0 0 0 0 0 1 0 0; ... 
11      mi*l*cos(beta) 0 0 0 0 0 0 1 0; ... 
12      -Ii -Ii 0 0 0 0 0 0 0]);
13  Bss = sym([-mf*l*betad*betad*cos(beta); ... 
14       mf*l*betad*betad*sin(beta)-mf*g; 0; ... 
15       -md*l*betad*betad*cos(beta); ... 
16       md*l*betad*betad*sin(beta)-md*g; ... 
17       Id*phidd-Ts; ... 
18       -mi*l*betad*betad*cos(beta); ... 
19       mi*l*betad*betad*sin(beta)-mi*g; ... 
20       Ii*phidd+Ts+Td]);
xss = sym(Ass\Bss);

% Double stance
syms mf md mi Id Ii l thetadd R1y R2y phidd Ts Td g pi n;
Ads = sym([1 1 0 0; ...
   -l*sin(pi/n) l*sin(pi/n) 1 0; ...
   0 0 1 0; ...
   0 0 0 Ii]);
Bds = sym([g*(mf+md+mi); 0; Id*phidd - Ts; -Ts - Ii*phidd - Td]);
xds = sym(Ads\Bds);

A.2 Simulation of a single step

function [isstep, sdata] = singlestep(y0ss, step)

    global n;

    isstep = 1;
    sdata = [];

    % Single stance ODE
    sstol = 1e-8;
    tspans = [0 2];
    options = odeset('AbsTol', sstol, 'RelTol', sstol, 'Events', @legdown);
    [sst, ssy, ~, ~, ie] = ode45(@sstance, tspans, y0ss, options);

    % Collision transition equations
    stime = sst;
    time = sst+y0ss(7); % Create time vector
    theta = ssy(:,1); % Inertia wheel angle during single stance
    thetad = ssy(:,2); % Inertia wheel angular velocity during single stance
beta = ssy(:,3); % Frame angle during single stance
betad = ssy(:,4); % Frame angular velocity during single stance
phi = phival(time); % Drum angle during single stance
phid = phidval(time); % Drum angular velocity during single stance
if (1 - sst(end)/2) <= 0.001
    isstep = 0;
    return
elseif ie == 2
    isstep = 0;
    return
end
stime = vertcat(stime,stime(end));
time = vertcat(time,time(end)); % Time of collision
beta = vertcat(beta,beta(end)); % Frame angle at collision
phi = vertcat(phi,phi(end)); % Drum angle at collision
theta = vertcat(theta,theta(end)); % Inertia wheel angle at collision
phid = vertcat(phid,phid(end)); % Drum angular velocity stays the same
thetad = vertcat(thetad,betad(end) + thetad(end));
betad = vertcat(betad,0); % Frame angular velocity is zero
stance = ones(length(time),1);

% Double stance ODE
dstol = 1e-8;
tspand = [sst(end) sst(end)+2];
y0ds = [theta(end) thetad(end) beta(end) betad(end) phi(end) phid(end)
    ];
options = odeset('AbsTol',dstol,'RelTol',dstol,'Events',@legup);
[dst,dsy] = ode45(@dstance, tspand, y0ds, options);
if (1 - (dst(end) - sst(end))/2) <= 0.001
    isstep = 0;
\textbf{A.3 Single stance ODE}

\begin{verbatim}
function dydt = ssstance(t,y)
global l mf md mi If Id Ig k c;
\end{verbatim}
dydt = zeros(size(y));

% Variables
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
theta = y(1);
\[ \frac{dy}{dt}(4) = - (I_d \cdot \ddot{\phi} - Ts + g \cdot l \cdot m_d \cdot \cos(\beta) + g \cdot l \cdot m_f \cdot \cos(\beta) + g \cdot l \cdot m_i \cdot \cos(\beta)) \left/ \left( I_d + I_f + l^2 \cdot m_d \cdot \cos(\beta)^2 + l^2 \cdot m_f \cdot \cos(\beta)^2 + l^2 \cdot m_i \cdot \cos(\beta)^2 + l^2 \cdot m_d \cdot \sin(\beta)^2 + l^2 \cdot m_f \cdot \sin(\beta)^2 + l^2 \cdot m_i \cdot \sin(\beta)^2 \right) \right. \];

\[ \frac{dy}{dt}(5) = \dot{\phi}; \]

\[ \frac{dy}{dt}(6) = \ddot{\phi}; \]

\[ \text{end} \]

A.4 Event detection for foot landing

\[ \text{function } [\text{value}, \text{isterminal}, \text{direction}] = \text{legdown}(\cdot, y) \]

```
global n;

value = [\pi - (180 - 360/n) / 360 \cdot \pi - y(3), (0.5 - 1/n) \cdot \pi - y(3)];
isterminal = [1, 1];
direction = [-1, +1];
```

end

A.5 Double stance ODE

\[ \text{function } dyt = dstance(t, y) \]

```
global I i c;

dyt = zeros(size(y));

% Variables
theta = y(1);
thetad = y(2);
```
beta = y(3);
betad = y(4);
phi = y(5);
phid = y(6);
phidd = phiddval(t);

Ts = k*theta;
Td = c*(thetad+phid);

% Equations
dydt(1) = thetad;
dydt(2) = -(Td + Ts + Ii*phidd)/Ii;
dydt(3) = 0;
dydt(4) = 0;
dydt(5) = phid;
dydt(6) = phidd;

end

A.6 Event detection for foot lift off

function [value, isterminal, direction] = legup(t, y)

global l mf md mi Id g n k;

Ts = k*y(1);
phidd = phiddval(t);

value = (Id*phidd - Ts + g*l*md*sin(pi/n) + g*l*mf*sin(pi/n) + g*l*mi*
         sin(pi/n))/(2*l*sin(pi/n));
isterminal = 1;
direction = -1;
A.7 \( \dot{\phi}(t) \)

1 function phi = phival(t)
2     global A T P;
3     phi = A* sin(2*pi/T.*t + P);
4 end

A.8 \( \ddot{\phi}(t) \)

1 function phid = phidval(t)
2     global A T P;
3     phid = -A*2*pi/T*cos(2*pi/T.*t + P);
4 end

A.9 \( \dddot{\phi}(t) \)

1 function phidd = phiddval(t)
2     global A T P;
3     phidd = -A*4*pi*pi/T/T*sin(2*pi/T.*t + P);
4 end

A.10 Root find

1 % Control parameters
2 A = 0.1;
3 T = 1;
4 P = 2;
5
6 % Constants
7 param(1) = 9; % Number of legs
param (2) = 0.3935;  % Leg length in meters
param (3) = 0.0646;  % Drum radius in meters
param (4) = 0.36;  % Inertia wheel radius in meters (drawing)
param (5) = 1.438;  % Frame mass in kilograms
param (6) = 0.303;  % Drum mass in kilograms
param (7) = 1.803;  % Inertia wheel mass in kilograms
param (8) = 0.0851;  % Frame mass moment of inertia in kg-m^2
param (9) = 0.0018;  % Drum mass moment of inertia in kg-m^2
param (10) = 0.0984;  % Inertia wheel mass moment of inertia in kg-m^2
param (11) = 9.81;  % Acceleration due to gravity in m/s^2
param (12) = 4;  % Angular spring constant in N-m/rad
param (13) = 0.004;  % Angular damping coefficient for air drag in N-m/rad

% Initial conditions
input (1) = 1.3071;  % theta0
input (2) = 14.9486;  % thetad0
input (3) = (0.5−1/param (1))*pi;  % beta0
input (4) = 0;  % betad0
input (5) = A*sin(P);  % phi0
input (6) = A*2*pi/T*cos(P);  % phid0

% Positional control parameters
input (7) = A;
input (8) = T;
input (9) = P;

% Setup
maxit = 50;
maxer = 10^(-15);
cumer = 1;
it = 0;
% Variation

dtheta = 0.01;
dA = 0.01;
dF = 0.01;

while (cumer > maxer && it < maxit)

    it = it + 1;

    % Run single step
    [isstep, dif] = singlestepparams(input, param);
    if isstep ~= 1
        break
    end

cumer = sqrt(diff(1)^2 + diff(2)^2 + diff(3)^2);

    % Variation in theta
    input(1) = input(1) + dtheta;
    [isstep, diftheta] = singlestepparams(input, param);
    if isstep ~= 1
        break
    end

    input(1) = input(1) - dtheta;

    % Variation in A
    input(7) = input(7) + dA;
    [isstep, difA] = singlestepparams(input, param);
    if isstep ~= 1
        break
    end
input(7) = input(7) - dA;

% Variation in F
input(8) = input(8) + dF;
[isstep, difF] = singlestepparams(input, param);
if isstep ~= 1
    break
end
input(8) = input(8) - dF;
difp = [diftheta; dA; difF].';
dp(1:3,1) = dtheta;
dp(1:3,2) = dA;
dp(1:3,3) = dF;
J = (difp - [dif; dif; dif].') / dp;
if cond(J) > 1e15
    isstep = 0;
disp('Singular matrix')
    break
end
steproots = -inv(J) * (dif.' ) + [input(1); input(7); input(8)];
input(1) = steproots(1);
input(7) = steproots(2);
input(8) = steproots(3);
if (it < maxit && isstep == 1)
```
function [isstep, output] = singlestepparams(input, param)

global l r w mf md mi If Id Ii g k n c A T P;

% Constants
n = param(1);    % Number of legs
l = param(2);    % Leg length in meters
r = param(3);    % Drum radius in meters
w = param(4);    % Inertia wheel radius in meters (drawing)
mf = param(5);   % Frame mass in kilograms
md = param(6);   % Drum mass in kilograms
mi = param(7);   % Inertia wheel mass in kilograms
if = param(8);   % Frame mass moment of inertia in kg-m^2
id = param(9);   % Drum mass moment of inertia in kg-m^2
ii = param(10);  % Inertia wheel mass moment of inertia in kg-m^2
G = param(11);   % Acceleration due to gravity in m/s^2
k = param(12);   % Angular spring constant in N-m/rad
c = param(13);   % Angular damping coefficient for air drag in N-m/rad

% Initial conditions
theta0 = input(1);
thetaD0 = input(2);
beta0 = input(3);
betaD0 = input(4);
phi0 = input(5);
```

A.11 Periodic conditions output
phid0 = input(6);
time0 = 0;
y0ss = [theta0 thetad0 beta0 betad0 phi0 phid0 time0];

% Positional control parameters
A = input(7); % Amplitude of input sine function
T = input(8); % Period of input sine function
P = input(9); % Phase shift of input sine function

% Run single step
[iisstep, sdata] = singlestep(y0ss,1);
if iisstep == 0
    output = [1 1 1];
    return
end

% Collect step data
theta = sdata(:,2);
thetad = sdata(:,3);
time = sdata(:,1);

% Output difference
output(:,1) = theta(end) - theta(1);
output(:,2) = thetad(end) - thetad(1);
output(:,3) = T - (time(end) - time(1));

end

A.12 Calculate performance metrics for periodic step

function [stepdata, output] = analyze(input, param)
global l r w mf md mi If Id Ii g k n c A T P;

% Constants
n = param(1);  % Number of legs
l = param(2);  % Leg length in meters
r = param(3);  % Drum radius in meters
w = param(4);  % Inertia wheel radius in meters (drawing)
mf = param(5);  % Frame mass in kilograms
md = param(6);  % Drum mass in kilograms
mi = param(7);  % Inertia wheel mass in kilograms
If = param(8);  % Frame mass moment of inertia in kg-m^2
Id = param(9);  % Drum mass moment of inertia in kg-m^2
Ii = param(10);  % Inertia wheel mass moment of inertia in kg-m^2
g = param(11);  % Acceleration due to gravity in m/s^2
k = param(12);  % Angular spring constant in N-m/rad
c = param(13);  % Angular damping coefficient for air drag in N-m/rad

% Initial conditions
theta0 = input(1);
theta0d = input(2);
beta0 = input(3);
betad0 = input(4);
phi0 = input(5);
phid0 = input(6);
time0 = 0;
y0ss = [theta0 theta0d beta0 betad0 phi0 phid0 time0];

% Motor torque parameters
A = input(7);
T = input(8);
P = input(9);
% Single stance ODE
sstol = 1e-8;
tspans = [0 2];
sopt = odeset('AbsTol', sstol, 'RelTol', sstol, 'Events', @(legdown));
[sst, ssy, ~, ~, ies] = ode45(@sstance, tspans, y0ss, sopt);

% Collision transition equations
stime = sst;
time = sst+y0ss(7); % Create time vector
theta = ssy(:,1); % Inertia wheel angle during single stance
thetad = ssy(:,2); % Inertia wheel angular velocity during single stance
beta = ssy(:,3); % Frame angle during single stance
betad = ssy(:,4); % Frame angular velocity during single stance
phi = phival(time);
phid = phidval(time);
if ies == 2
    error('No step taken');
end
stime = vertcat(stime, stime(end));
time = vertcat(time, time(end)); % Time of collision
beta = vertcat(beta, beta(end)); % Frame angle at collision
phi = vertcat(phi, phi(end)); % Drum angle at collision
theta = vertcat(theta, theta(end)); % Inertia wheel angle at collision
phid = vertcat(phid, phid(end)); % Drum angular velocity is the same
thetad = vertcat(thetad, betad(end) + thetad(end));
betad = vertcat(betad, 0); % Frame angular velocity is zero
stance = ones(length(time),1);

% Double stance ODE
dstol = 1e-8;
tspan = [sst(end) sst(end)+2];
y0ds = [theta(end) thetad(end) beta(end) betad(end) phi(end) phid(end) time(end)];
dopt = odeset('AbsTol',dstol,'RelTol',dstol,'Events',@legup);
[dst, dsy] = ode45(@dstance, tspan, y0ds, dopt);
if (1 - (dst(end) - sst(end))/2 <= 0.001)
    error('Stuck in double stance')
end

stime = vertcat(stime, dst);
time = vertcat(time, dst+y0ss(7));
theta = vertcat(theta, dsy(:,1));
thetad = vertcat(thetad, dsy(:,2));
beta = vertcat(beta, dsy(:,3));
betad = vertcat(betad, dsy(:,4));
phi = phival(time);
phid = phidval(time);
stance = vertcat(stance, zeros(length(dst),1)+2);

% Calculate torques
Ts = k.*theta;
Td = c.*(betad+thetad+phid);
sdata(:,4) = beta;
sdata(:,9) = stime;
sdata(:,11) = Ts;
sdata(:,8) = stance;
Tm = motortorque(sdata);

% Calculate energy
h = l*sin(beta);
PE = (mf+md+mi)*g*h + 0.5*k*theta.^2;
\[ KE = 0.5 \ast (mf+md+mi) \ast (1 \ast betad) \ast \hat{2} + 0.5 \ast (If \ast betad) \ast \hat{2} \\ldots \]
\[ + Id \ast (betad+phid) \ast \hat{2} + Ii \ast (betad+phid+thetad) \ast \hat{2} \];

\[ TE = PE + KE; \]

% Output step data
stepdata(:,1) = time;
stepdata(:,2) = theta;
stepdata(:,3) = thetad;
stepdata(:,4) = beta;
stepdata(:,5) = betad;
stepdata(:,6) = phi;
stepdata(:,7) = phid;
stepdata(:,8) = stance;
stepdata(:,9) = Tm;
stepdata(:,10) = Ts;
stepdata(:,11) = Td;
stepdata(:,12) = PE;
stepdata(:,13) = KE;
stepdata(:,14) = TE;

% Perform analysis
power = Tm \ast phid;
powerpos = power;
powerneg = power;
powerpos(power<=0) = 0;
powerneg(power>=0) = 0;
energyrequired = \text{trapz}(time, abs(power));
energyin = \text{trapz}(time, powerpos);
energyout = \text{trapz}(time, powerneg);
weight = g \ast (mf+md+mi);
distance = 2 \ast l \ast \sin(pi/n);
\[
\text{COT} = \frac{\text{energy required}}{\text{weight} / \text{distance}}; \\
\text{speed} = \frac{\text{distance}}{(\text{time(end)} - \text{time(1)})}; \\
\text{Tm}_{\text{max}} = \max(\text{abs}(\text{Tm})); \\
\theta_{\text{max}} = \max(\text{theta}); \\
\theta_{d\text{max}} = \max(\text{thetad}); \\
\text{energy drag} = -\text{trapz}(\text{time}, \text{abs}(\text{Td} \ast (\text{betad} + \text{phid} + \text{thetad}))); \\
\text{col}_i = \text{length(sst)}; \\
\text{energy collision} = \text{TE(\text{col}_i + 1)} - \text{TE(\text{col}_i)}; \\
\]

% Output analysis
\[
\text{output(}\cdot,1) = \text{COT}; \\
\text{output(}\cdot,2) = \text{speed}; \\
\text{output(}\cdot,3) = \text{Tm}_{\text{max}}; \\
\text{output(}\cdot,4) = \theta_{\text{max}}; \\
\text{output(}\cdot,5) = \theta_{d\text{max}}; \\
\text{output(}\cdot,6) = \text{energy required}; \\
\text{output(}\cdot,7) = \text{energy drag}; \\
\text{output(}\cdot,8) = \text{energy collision}; \\
\text{output(}\cdot,9) = \text{energy in}; \\
\text{output(}\cdot,10) = \text{energy out}; \\
\]

end

A.13 Animation

\[
\text{global l r w mf md mi If Id Ii g kn c A TP;} \\
\text{aviobj = VideoWriter('failed.avi');} \\
\text{open(aviobj);} \\
\]

% Control function
\[
\text{A} = 0; \\
\]
\[ T = 0.9571; \]
\[ P = 2; \]

\% Constants
\[ n = 8; \] \% Number of legs
\[ l = 0.3935; \] \% Leg length in meters
\[ r = 0.0646; \] \% Drum radius in meters
\[ w = 0.36; \] \% Inertia wheel radius in meters
\[ mf = 1.438; \] \% Frame mass in kilograms
\[ md = 0.303; \] \% Drum mass in kilograms
\[ mi = 1.803; \] \% Inertia wheel mass in kilograms
\[ If = 0.0851; \] \% Frame mass moment of inertia in kg\cdot m^2
\[ Id = 0.0018; \] \% Drum mass moment of inertia in kg\cdot m^2
\[ Ii = 0.0984; \] \% Inertia wheel mass moment of inertia in kg\cdot m^2
\[ g = 9.81; \] \% Acceleration due to gravity in m/s^2
\[ k = 4; \] \% Angular spring constant in N\cdot m/rad
\[ c = 0.004; \] \% Angular damping coefficient for air drag in N\cdot m/rad

\% Initial conditions
\[ \theta_0 = 2; \]
\[ \theta_d0 = 10; \]
\[ \beta_0 = (0.5 - 1/n) \pi; \]
\[ \beta_d0 = 10; \]
\[ \phi_0 = A \sin(P); \]
\[ \phi_d0 = A \cdot 2 \pi / T \cdot \cos(P); \]
\[ \text{time0} = 0; \]
\[ y0ss = [\theta_0 \ \theta_d0 \ \beta_0 \ \beta_d0 \ \phi_0 \ \phi_d0 \ \text{time0}]; \]
\[ \text{steps} = 5; \]
\[ \text{tdata} = []; \]
\[ \text{KE} = []; \]
\[ \text{PE} = []; \]
% Animation initialization
animation = 1; % Turn animation on(1) or off
rstep = 1;
if animation == 1
  anifig = figure(1);
  hold on
  beta0 = y0ss(3);
  phi0 = y0ss(5);
  theta0 = y0ss(1);

% Initial centerpoint
cn = [-l*cos(beta0); l*sin(beta0)];

% Construction of frame
R = [cos(2*pi/n) -sin(2*pi/n); sin(2*pi/n) cos(2*pi/n)];
p1 = [0; 0];
s = 1:n;
for f=1:n
  p2 = R*(p1-cn)+cn;
  s(f) = line('xdata', [p1(1) p2(1)], 'ydata', [p1(2) p2(2)]);
p1 = p2;
end

% Construction of drum
an = 0:0.01:2*pi;
rx = r*cos(an);
ry = r*sin(an);
dcir = plot(cn(1) + rx, cn(2) + ry);
dp = sqrt((l*r-r-l*r*cos(pi-phi0))^2);
da = beta0+asin(r*sin(pi-phi0)/dp);
dx = -dp*cos(da);

dy = dp*sin(da);

dl = line([dx cn(1)+0.5*(dx-cn(1))], [dy cn(2)+0.5*(dy-cn(2))]);

% Construction of inertia wheel
wx = w*cos(an);
wy = w*sin(an);
wcir = plot(cn(1)+wx, cn(2)+wy);

ip = sqrt(l*l+w*w-2*l*w*cos(pi-phi0-theta0));
ia = beta0+asin(w*sin(pi-phi0-theta0)/ip);
ix = -ip*cos(ia);
iy = ip*sin(ia);
il = line([ix cn(1)+0.8*(ix-cn(1))], [iy cn(2)+0.8*(iy-cn(2))]);
set(il, 'Color', 'red');

% Construction of spiral
sn = linspace(0,theta0);
a = r;
b = (0.8*w-r)/theta0;
sx = -(a+b.*sn).*cos(sn+beta0+phi0);
sy = (a+b.*sn).*sin(sn+beta0+phi0);
sp = plot(cn(1)+sx, cn(2)+sy);
set(sp, 'Color', 'green');

% View layout
pad = 0.1;
axis([0 inf -pad 2*l+pad])
axis equal
axis off
set(anifig, 'Color', [1 1 1]);

end

% Multiple steps
for step=1:steps

% Run single step
[isstep, sdata] = singlestep(y0ss,step);
if isstep == 0
    return
end

% Collect step data
time = sdata(:,1);
theta = sdata(:,2);
theta = sdata(:,3);
beta = sdata(:,4);
betad = sdata(:,5);
phi = sdata(:,6);
phid = sdata(:,7);
stance = sdata(:,8);
stime = sdata(:,9);

% Calculate torques
Ts = k.*theta;
Td = c.*(betad+theta+phid);
sdata(:,10) = Ts;
sdata(:,11) = Td;
Tm = motortorque(sdata);
sdata(:,12) = Tm;
sdata(:,13) = phiddval(stime);

% Accumulate data
if step >= 1
tdata = vertcat(tdata, sdata);
end

% Setup next step
theta0 = theta(end);
theta0 = theta(end);
phi0 = phi(end);
phi0 = phi(end);
beta0 = (0.5−1/n)*pi;
beta0 = 0;
y0ss = [theta0 thetad0 beta0 betad0 phi0 phid0 time(end)];

if animation == 1
for f=1:length(time)

% Step specific values
itime = time(f);
itheta = theta(f);
ithetad = thetad(f);
ibeta = beta(f);
ibetad = betad(f);
iphii = phi(f);
iphid = phid(f);

% Animation parameters
d = rstep*l*sqrt(2−2*cos(2*pi/n));
cn = [-l*cos(ibeta)+d; l*sin(ibeta)]; % Centerpoint
\[ R = \begin{bmatrix} \cos(2\pi/n) - \sin(2\pi/n); \sin(2\pi/n) \cos(2\pi/n) \end{bmatrix}; \]

\[ p1 = [d; 0]; \]

\[ \text{for } j = 1:n \]
\[ p2 = R \cdot (p1 - cn) + cn; \]
\[ \text{set}(s(j), 'xdata', [p1(1) p2(1)], 'ydata', [p1(2) p2(2)]); \]
\[ p1 = p2; \]
\[ \text{end} \]

\[ \text{hold on} \]
\[ dp = \sqrt{l*l + r*r - 2*l*r * \cos(pi - iphi)}; \]
\[ da = \text{ibeta} + \text{asin}(r * \sin(pi - iphi) / dp); \]
\[ dx = d - dp * \cos(da); \]
\[ dy = dp * \sin(da); \]
\[ \text{set}(dl, 'xdata', [dx cn(1) + 0.5*(dx-cn(1))]); \]
\[ \text{set}(dl, 'ydata', [dy cn(2) + 0.5*(dy-cn(2))]); \]
\[ ip = \sqrt{l*l + w*w - 2*l*w * \cos(pi - iphi - itheta)}; \]
\[ ia = \text{ibeta} + \text{asin}(w * \sin(pi - iphi - itheta) / ip); \]
\[ ix = d - ip * \cos(ia); \]
\[ iy = ip * \sin(ia); \]
\[ \text{set}(il, 'xdata', [ix cn(1) + 0.8*(ix-cn(1))]); \]
\[ \text{set}(il, 'ydata', [iy cn(2) + 0.8*(iy-cn(2))]); \]
\[ \text{set}(dcir, 'xdata', cn(1) + rx, 'ydata', cn(2) + ry); \]
\[ \text{set}(wcir, 'xdata', cn(1) + wx, 'ydata', cn(2) + wy); \]
\[ sn = \text{linspace}(0, itheta, 50*abs(itheta)); \]
\[ b = (0.8*w-r)/itheta; \]
\[ sx = -(a+b.*sn) .* \cos(sn+phi+ibeta); \]
\[ sy = (a+b.*sn) .* \sin(sn+phi+ibeta); \]
\[ \text{set}(sp, 'xdata', cn(1)+sx, 'ydata', cn(2)+sy); \]
\[ \text{if } itheta >= 0 \]
\[ \text{set}(sp, 'Color', 'green') \]
\[ \text{else} \]
\[ \text{set}(sp, 'Color', 'yellow') \]
end
drawnow;
writeVideo(aviobj,getframe(anigif));
pause(0.001);
end
end

% Increment actual steps
rstep = rstep + isstep;
end
close(aviobj);
Appendix B

Torque control code

B.1 Solving for equations of motion

```matlab
syms i j k mf md mi If Id Ii l beta betad betadd;
syms thetadd Fx Fy Rx Ry Sx Sy phidd Tm Ts Td g;

% Single stance
Ass = sym([mf*l*sin(beta) 0 0 -1 0 -1 0 0 0; ...
           mi*l*sin(beta) 0 0 0 0 0 1 0; ...
           md*l*sin(beta) 0 0 1 0 0 0 -1 0; ...
           mf*l*cos(beta) 0 0 0 -1 0 -1 0 0; ...
           mi*l*cos(beta) 0 0 0 0 0 0 1; ...
           md*l*cos(beta) 0 0 0 1 0 0 0 -1; ...
           -If 0 0 0 0 -1*sin(beta) -1*cos(beta) 0 0 ;
           -Ii -Ii -Ii 0 0 0 0 0; ...
           -Id -Id 0 0 0 0 0 0|]);
Bss = sym([-mf*l*betad*betad*cos(beta); ...
           -mi*l*betad*betad*cos(beta); ...
           -md*l*betad*betad*cos(beta); ...
           mf*(l*betad*betad*sin(beta) - g); ...
           mi*(l*betad*betad*sin(beta) - g); ...
           md*(l*betad*betad*sin(beta) - g); ...
           Tm; ...])
```
Ts + Td; ...  
-\(Tm - Ts\));  
\(xss = \text{sym}(\text{Ass} \setminus \text{Bss});\)  

% Double stance  
syms mf md mi Id Ii l thetadd R1y R2y phidd Tm Ts Td g pi n;  
\(\text{Ads} = \text{sym}([1 1 0 0; \ldots -l \cdot \sin(\pi/n) \ l \cdot \sin(\pi/n) 0 0; \ldots 0 0 -Ii -Ii; \ldots 0 0 Id 0]);\)  
\(\text{Bds} = \text{sym}([g \cdot (mf+mi+md); -Tm; Ts+Td; Ts+Tm]);\)  
\(\text{xds} = \text{sym}(\text{Ads} \setminus \text{Bds});\)

### B.2 Simulation of a single step

```matlab
function [isstep, sdata] = singlstep(y0ss)
    sdata = [];
    isstep = 1;

    % Single stance ODE
    ssstol = 1e-8;
    tspans = [0 2];
    sopt = odeset('AbsTol', ssstol, 'RelTol', ssstol, 'Events', @legdown);
    [sst, ssy, ~, ~, ie] = ode45(@sstance, tspans, y0ss, sopt);

    % Collision transition equations
    time = sst+y0ss(7); % Create time vector
    theta = ssy(:,1); % Inertia wheel angle during single stance
    thetad = ssy(:,2); % Inertia wheel angular velocity during single stance
    beta = ssy(:,3);  % Frame angle during single stance
    betad = ssy(:,4);  % Frame angular velocity during single stance
```
phi = ssy(:,5);    % Drum angle during single stance
phid = ssy(:,6);   % Drum angular velocity during single stance

if (1 - sst(end)/2) <= 0.001
  istep = 0;
  return
elseif ie == 2
  istep = 0;
  return
end

time = vertcat(time, time(end));    % Time of collision
beta = vertcat(beta, beta(end));   % Frame angle at collision
phi = vertcat(phi, phi(end));      % Drum angle at collision
theta = vertcat(theta, theta(end)); % inertia wheel angle at collision
phid = vertcat(phid, betad(end) + phid(end));
thetad = vertcat(thetad, thetad(end));
betad = vertcat(betad, 0);

% Double stance ODE

dstol = 1e-8;
tspand = [sst(end) sst(end) +2];
y0ds = [theta(end) thetad(end) beta(end) betad(end) phi(end) phid(end) time(end)];
dopt = odeset('AbsTol', dstol, 'RelTol', dstol, 'Events', @legup);
[dst, dsy] = ode45(@dstance, tspand, y0ds, dopt);
if (1 - (dst(end) - sst(end))/2 <= 0.001)
  istep = 0;
  return
else
  istep = 1;
end
time = vertcat(time, dst+y0ss(7));
theta = vertcat(theta, dsy(:,1));
thetad = vertcat(thetad, dsy(:,2));
beta = vertcat(beta, dsy(:,3));
betad = vertcat(betad, dsy(:,4));
phi = vertcat(phi, dsy(:,5));
phid = vertcat(phid, dsy(:,6));

% Final data
sdata = time;
sdata = [sdata theta];
sdata = [sdata thetad];
sdata = [sdata beta];
sdata = [sdata betad];
sdata = [sdata phi];
sdata = [sdata phid];

end

B.3 Single stance ODE

function dydt = sstance(t,y)

global l mf md mi If Id Ii g k c;

dydt = zeros(size(y));

% Variables
theta = y(1);
thetad = y(2);
beta = y(3);
betad = y(4);
phi = y(5);
phid = y(6);

Ts = k*theta;
Td = c*(betad+thetad+phid);
Tm = motortorque(t,1);

% Equations
dydt(1) = thetad;
dydt(2) = -(Id*Td + Ii*Tm + Id*Ts + Ii*Ts)/(Id*Ii);
dydt(3) = betad;
dydt(4) = - (Tm + g*l*md*cos(beta) + g*l*mf*cos(beta) + g*l*mi*cos(beta))^(2) + g*l*md*cos(beta)^(2) + g*l*mf*cos(beta)^(2) + g*l*mi*cos(beta)^(2) + Id*Tm + Ii*Tm + Ii*Ts + Tm*ls + Ii*Ts)/(Id*Ii);
dydt(5) = phid;
dydt(6) = (Id*Tm + Ii*Tm + Ii*Ts + Tm*ls + Ii*Ts)/(Id*Ii);

end

B.4 Event detection for foot landing

function [value, isterminal, direction] = legdown(~, y)
global n;
value = [ \pi - (180 - 360/n) / 360 \star \pi - y(3), (0.5 - 1/n) \star \pi - y(3) ];

isterminal = [1, 1];
direction = [-1, +1];

end

B.5 Double stance ODE

function dydt = dstance(t,y)

global Id Ii k c;

dydt = zeros(size(y));

% Variables
theta = y(1);
theta = y(2);
beta = y(3);
betad = y(4);
phi = y(5);
phid = y(6);

Ts = k \star theta;
Td = c \star (theta + phi);
Tm = motortorque(t,2);

% Equations
dydt(1) = theta;
dydt(2) = -(Id \star Td + Ii \star Tm + Id \star Ts + Ii \star Ts) / (Id \star Ii);
dydt(3) = 0;
dydt(4) = 0;
dydt(5) = phid;
\[ \frac{dydt}{6} = \frac{(Tm + Ts)}{Id}; \]

**B.6 Event detection for foot lift off**

```matlab
function [value, isterminal, direction] = legup(t, \theta)
    global nl mf md mi g;
    Tm = motortorque(t, 2);
    value = \left(\frac{Tm + g*l*md*s\sin(\frac{\pi}{n}) + g*l*mf*s\sin(\frac{\pi}{n}) + g*l*mi*s\sin(\frac{\pi}{n})}{2*l*s\sin(\frac{\pi}{n})}\right);
    isterminal = 1;
    direction = -1;
end
```

**B.7 Root find**

```matlab
% Constants
param(1) = 8;       % Number of legs
param(2) = 0.3935;  % Leg length in meters
param(3) = 0.0646;  % Drum radius in meters
param(4) = 0.36;    % Inertia wheel radius in meters (drawing)
param(5) = 1.438;   % Frame mass in kilograms
param(6) = 0.303;   % Drum mass in kilograms
param(7) = 1.803;   % Inertia wheel mass in kilograms
param(8) = 0.0851;  % Frame mass moment of inertia in kg⋅m^2
param(9) = 0.0018;  % Drum mass moment of inertia in kg⋅m^2
param(10) = 0.0984; % Inertia wheel mass moment of inertia in kg⋅m^2
```
param (11) = 9.81; % Acceleration due to gravity in m/s^2
param (12) = 4; % Angular spring constant in N-m/rad
param (13) = 0.004; % Angular damping coefficient for air drag in N-m/rad

% Initial conditions
input (1) = 1.3153;
input (2) = 14.9488;
input (3) = (0.5 − 1/param(1))*pi;
input (4) = 0;
input (5) = 0.0569;
input (6) = −0.3778;

% Motor torque parameters
input (7) = 10.3309; % A
input (8) = 6.5893; % F
input (9) = −2.606; % P

% Setup
maxit = 50;
maxer = 10^(-10);
cumer = 1;
it = 0;

% Variation
dtheta = 0.01;
dA = 0.01;
dF = 0.01;

while (cumer > maxer && it < maxit)
    it = it + 1;
% Run single step
[isstep, dif] = singlestepparams(input, param);
if isstep ~= 1
    break
end
cumer = sqrt(dif(1)^2 + dif(2)^2 + dif(3)^2);

% Variation in theta
input(1) = input(1) + dtheta;
[isstep, diftheta] = singlestepparams(input, param);
if isstep ~= 1
    break
end
input(1) = input(1) - dtheta;

% Variation in A
input(7) = input(7) + dA;
[isstep, difA] = singlestepparams(input, param);
if isstep ~= 1
    break
end
input(7) = input(7) - dA;

% Variation in T
input(8) = input(8) + dF;
[isstep, difF] = singlestepparams(input, param);
if isstep ~= 1
    break
end
input(8) = input(8) - dF;
difp = [diftheta;difA;difF].';
dp(1:3,1) = dtheta;
dp(1:3,2) = dA;
dp(1:3,3) = dF;

J = (difp-[dif;dif;dif]).'/dp;

roots = -inv(J)*(dif.') + [input(1);input(7);input(8)];

input(1) = roots(1);
input(7) = roots(2);
input(8) = roots(3);

end

if (it < maxit & isstep == 1)
    disp('root')
    try
        [~, output] = analyze(input, param);
        print = [theta A F roots' output(1)];
    catch
        print = [theta A F roots'];
    end
    disp(print)
    dlmwrite('output.csv',print,'-append');
else
    disp([theta A F])
end

B.8 Periodic conditions output
function [isstep, output] = singlestepparams(input, param)

    global l r w mf md mi If Id Ii g k n c A F P;

% Constants
n = param(1);  % Number of legs
l = param(2);  % Leg length in meters
r = param(3);  % Drum radius in meters
w = param(4);  % Inertia wheel radius in meters (drawing)
mf = param(5);  % Frame mass in kilograms
md = param(6);  % Drum mass in kilograms
mi = param(7);  % Inertia wheel mass in kilograms
If = param(8);  % Frame mass moment of inertia in kg-m²
Id = param(9);  % Drum mass moment of inertia in kg-m²
Ii = param(10);  % Inertia wheel mass moment of inertia in kg-m²
g = param(11);  % Acceleration due to gravity in m/s²
k = param(12);  % Angular spring constant in N-m/rad
C = param(13);  % Angular damping coefficient for air drag in N-m/rad

% Initial conditions
theta0 = input(1);
theta0 = input(2);
beta0 = input(3);
betad0 = input(4);
phi0 = input(5);
phid0 = input(6);
time0 = 0;
y0ss = [theta0 thetad0 beta0 betad0 phi0 phid0 time0];

% Motor torque parameters
A = input(7);  % Amplitude of input sine function
F = input(8);  % Frequency of input sine function
P = input(9);  % Phase shift of input sine function

% Run single step
[isstep, sdata] = singlestep(y0ss);
if isstep == 0
    output = [1 1 1];
    return
end

% Collect step data
theta = sdata(:,2);
thetad = sdata(:,3);
phid = sdata(:,7);

% Output difference
output(:,1) = phid(end) - phid(1);
output(:,2) = theta(end) - theta(1);
output(:,3) = thetad(end) - thetad(1);

end

B.9 Calculate performance metrics for periodic step

function [stepdata, output] = analyze(input, param)

    global l r w mf md mi Id Ii g k n c A F P;

    % Constants
    n = param(1);  % Number of legs
    l = param(2);  % Leg length in meters
    r = param(3);  % Drum radius in meters
\[ w = \text{param}(4) ; \]  % Inertia wheel radius in meters (drawing)
\[ mf = \text{param}(5) ; \]  % Frame mass in kilograms
\[ md = \text{param}(6) ; \]  % Drum mass in kilograms
\[ mi = \text{param}(7) ; \]  % Inertia wheel mass in kilograms
\[ If = \text{param}(8) ; \]  % Frame mass moment of inertia in kg-m^2
\[ Id = \text{param}(9) ; \]  % Drum mass moment of inertia in kg-m^2
\[ Ii = \text{param}(10) ; \]  % Inertia wheel mass moment of inertia in kg-m^2
\[ g = \text{param}(11) ; \]  % Acceleration due to gravity in m/s^2
\[ k = \text{param}(12) ; \]  % Angular spring constant in N-m/rad
\[ c = \text{param}(13) ; \]  % Angular damping coefficient for air drag in N-m/rad

\% Initial conditions
\[ \theta_0 = \text{input}(1) ; \]
\[ \theta_\dot{}_0 = \text{input}(2) ; \]
\[ \beta_0 = (0.5\cdot1/\text{param}(1))*\pi ; \]
\[ \beta_\dot{}_0 = \text{input}(4) ; \]
\[ \phi_0 = \text{input}(5) ; \]
\[ \phi_\dot{}_0 = \text{input}(6) ; \]
\[ \text{time}_0 = 0 ; \]
\[ \text{y}_0\text{ss} = [\theta_0 \ \theta_\dot{}_0 \ \beta_0 \ \beta_\dot{}_0 \ \phi_0 \ \phi_\dot{}_0 \ \text{time}_0] ; \]

\% Motor torque parameters
\[ A = \text{input}(7) ; \]  % Amplitude of input sine function
\[ F = \text{input}(8) ; \]  % Frequency of input sine function
\[ P = \text{input}(9) ; \]  % Phase shift of input sine function

\% Single stance ODE
\[ \text{sstol} = 1e^{-8} ; \]
\[ \text{tspans} = [0 \ 2] ; \]
\[ \text{sopt} = \text{odeset}('\text{AbsTol}',\text{sstol} , '\text{RelTol}' , \text{sstol} , '\text{Events}' , @\text{legdown}) ; \]
\[ [\text{sst} , \text{ssy} , ~ , ~ , \text{ies}] = \text{ode45}(@\text{stance} , \text{tspans} , \text{y}_0\text{ss} , \text{sopt}) ; \]
% Collision transition equations

{\text{time}} = \text{sst} + \text{y0ss}(7); \% Create time vector

{\text{theta}} = \text{ssy}(\text{,:,1}); \% Inertia wheel angle during single stance

{\text{thetad}} = \text{ssy}(\text{,:,2}); \% Inertia wheel angular velocity during single

{\text{beta}} = \text{ssy}(\text{,:,3}); \% Frame angle during single stance

{\text{betad}} = \text{ssy}(\text{,:,4}); \% Frame angular velocity during single stance

{\text{phi}} = \text{ssy}(\text{,:,5}); \% Drum angle during single stance

{\text{phid}} = \text{ssy}(\text{,:,6}); \% Drum angular velocity during single stance

\text{if} \text{ ies} == 2
\text{  error('No step taken');}
\text{end}

\text{time} = \text{vercat(time, time(end))}; \% Time of collision

{\text{beta}} = \text{vercat(beta, beta(end))}; \% Frame angle at collision

{\text{phi}} = \text{vercat(phi, phi(end))}; \% Drum angle at collision

{\text{theta}} = \text{vercat(theta, theta(end))}; \% Inertia wheel angle at collision

{\text{phid}} = \text{vercat(phid, betad(end) + phid(end))};

{\text{thetad}} = \text{vercat(thetad, thetad(end))};

{\text{betad}} = \text{vercat(betad, 0)};

{\text{stance}} = \text{ones(length(time),1)};

\% Double stance ODE

\text{dstol} = 1e-8;

\text{tspand} = [\text{sst(end) sst(end) +2]};

\text{y0ds} = [\text{theta(end) thetad(end) beta(end) betad(end) phi(end) phid(end) time(end)}];

\text{dopt} = \text{odeset('AbsTol',dstol,'RelTol',dstol,'Events','@legup');}

[\text{dst}, \text{dsy}] = \text{ode45(@dstance, tspand, y0ds, dopt)};

\text{if} (1 - (\text{dst(end)} - \text{sst(end)])/2 <= 0.001)
\text{  disp(dst(end)-sst(end))}
\text{  error('Stuck in double stance')}
end

72 time = vertcat(time, dst+y0ss(7));
73 theta = vertcat(theta, dsy(:,1));
74 thetad = vertcat(thetad, dsy(:,2));
75 beta = vertcat(beta, dsy(:,3));
76 betad = vertcat(betad, dsy(:,4));
77 phi = vertcat(phi, dsy(:,5));
78 phid = vertcat(phid, dsy(:,6));
79 stance = vertcat(stance, zeros(length(dst),1)+2);

81 % Calculate torques
82 Tm = arrayfun(@motortorque, time, stance);
83 Ts = k.*theta;
84 Td = c.*(betad+thetad+phid);

85 % Calculate energy
86 h = l*sin(beta);
87 PE = (mf+md+mi)*g*h + 0.5*k*theta.^2;
88 KE = 0.5.*(mf+md+mi).*(1.*betad).^2 + 0.5.*(If.*betad).^2 + Id.*(betad+phid).^2 + Ii.*(betad+phid+thetad).^2);
89 TE = PE + KE;

93 % Output step data
94 stepdata(:,1) = time;
95 stepdata(:,2) = theta;
96 stepdata(:,3) = thetad;
97 stepdata(:,4) = beta;
98 stepdata(:,5) = betad;
99 stepdata(:,6) = phi;
100 stepdata(:,7) = phid;
stepdata(:,8) = stance;
stepdata(:,9) = Tm;
stepdata(:,10) = Ts;
stepdata(:,11) = Td;
stepdata(:,12) = PE;
stepdata(:,13) = KE;
stepdata(:,14) = TE;

% Perform analysis
power = Tm.*phid;
powerpos = power;
powerneg = power;
powerpos(power<=0) = 0;
powerneg(power>=0) = 0;
energyrequired = trapz(time,abs(power));
energyin = trapz(time,powerpos);
energyout = trapz(time,powerneg);
weight = g*(mf+md+mi);
distance = 2*l*sin(pi/n);
COT = energyrequired/weight/distance;
speed = distance/(time(end) - time(1));
Tm_max = max(abs(Tm));
theta_max = max(theta);
theta_d_max = max(theta_d);
energydrag = -trapz(time,abs(Td.*(betad+phid+thetad)));
collisionindex = length(sst);
energy_collision = TE(collisionindex+1) - TE(collisionindex);

% Output analysis
output(:,1) = COT;
output(:,2) = speed;
output(:,3) = Tm_max;
output(:,4) = theta_max;
output(:,5) = thetad_max;
output(:,6) = energyrequired;
output(:,7) = energydrag;
output(:,8) = energycollision;
output(:,9) = energyin;
output(:,10) = energyout;

end

B.10 Animation

global l r w mf md mi If Id Ii g k n c A F P;

aviobj = VideoWriter('torque.avi');
open(aviobj);

% Control function
A = 10.331;
F = 6.5893;
P = 3.6772;

% Constants
n = 8;                   % Number of legs
l = 0.3935;              % Leg length in meters
r = 0.0646;              % Drum radius in meters
w = 0.36;                % Inertia wheel radius in meters
mf = 1.438;              % Frame mass in kilograms
md = 0.303;              % Drum mass in kilograms
mi = 1.803;              % Inertia wheel mass in kilograms
If = 0.0851;             % Frame mass moment of inertia in kg·m^2
Id = 0.0018; % Drum mass moment of inertia in kg-m^2
Ii = 0.0984; % Inertia wheel mass moment of inertia in kg-m^2
g = 9.81; % Acceleration due to gravity in m/s^2
k = 4; % Angular spring constant in N-m/rad
c = 0.004; % Angular damping coefficient for air drag in N-m/rad

% Initial conditions
theta0 = 1.3153;
theta0 = 14.9488;
beta0 = (0.5 – 1/n)*pi;
betad0 = 0;
phi0 = 0.0569;
phid0 = -0.3778;
time0 = 0;
y0ss = [theta0 thetad0 beta0 betad0 phi0 phid0 time0];
steps = 8;
tdata = [];
KE = [];
PE = [];

% Animation initialization
animation = 1; % Turn animation on(1) or off
rstep = 1;
if animation == 1
    anifig = figure(1);
    hold on
    beta0 = y0ss(3);
    phi0 = y0ss(5);
    theta0 = y0ss(1);
    % Initial centerpoint
cn = [−l*cos(beta0); l*sin(beta0)];

% Construction of frame
R = [cos(2*pi/n) −sin(2*pi/n); sin(2*pi/n) cos(2*pi/n)];
p1 = [0; 0];
s = 1:n;
for f=1:n
    p2 = R*(p1−cn)+cn;
    s(f) = line('xdata', [p1(1) p2(1)], 'ydata', [p1(2) p2(2)]);
    p1 = p2;
end

% Construction of drum
an = 0:0.01:2*pi;
rx = r*cos(an);
ry = r*sin(an);
dcir = plot(cn(1) + rx, cn(2) + ry);
dp = sqrt(l^2+r^2−2*l*r*cos(pi−phi0));
da = beta0+asin(r*sin(pi−phi0)/dp);
dx = −dp*cos(da);
dy = dp*sin(da);
dl = line([dx cn(1)+0.5*(dx−cn(1)] ,... 
      [dy cn(2)+0.5*(dy−cn(2))] );

% Construction of inertia wheel
wx = w*cos(an);
wy = w*sin(an);
wcir = plot(cn(1) + wx, cn(2) + wy);
ip = sqrt(l^2+w^2−2*l*w*cos(pi−phi0−theta0));
ia = beta0+asin(w*sin(pi−phi0−theta0)/ip);
ix = −ip*cos(ia);
iy = ip*sin(ia);

il = line([ix cn(1)+0.8*(ix-cn(1))],...
        [iy cn(2)+0.8*(iy-cn(2))]);
set(il, 'Color', 'red');

% Construction of spiral
sn = linspace(0,theta0);
a = r;
b = (0.8*w-r)/theta0;
sx = -(a+b.*sn).*cos(sn+beta0+phi0);
sy = (a+b.*sn).*sin(sn+beta0+phi0);
sp = plot(cn(1)+sx,cn(2)+sy);
set(sp, 'Color', 'green');

% View layout
pad = 0.1;
axis([0 inf -pad 2*l+pad])
axis equal
axis off
set(anifig, 'Color', [1 1 1]);
end

% Multiple steps
for step=1:steps

% Run single step
[isstep, sdata] = singlestep(y0ss);
if isstep == 0
    return
end
% Collect step data

time = sdata(:,1);
theta = sdata(:,2);
thetad = sdata(:,3);
beta = sdata(:,4);
betad = sdata(:,5);
phi = sdata(:,6);
phid = sdata(:,7);

% Accumulate data
if step >= 1
tdata = vertcat(tdata, sdata);
end

% Setup next step
theta0 = theta(end);
theta0 = theta(end);
theta0 = theta0;
phi0 = phi(end) + (beta(end) - (0.5 - 1/n)*pi);
phid0 = phid(end);
betad0 = (0.5 - 1/n)*pi;
betad0 = betad0;
y0ss = [theta0 thetad0 beta0 betad0 phi0 phid0 time(end)];

if animation == 1
for f=1:1:length(time)

    % Step specific values
    itime = time(f);
    itheta = theta(f);
    ithetad = thetad(f);
end
i beta = \texttt{beta}(f);
ibetad = \texttt{betad}(f);
iph = \texttt{phi}(f);
iphid = \texttt{phid}(f);

% Animation parameters

d = rstep*1*sqrt(2-2*cos(2*pi/n));

\begin{align*}
\texttt{cn} &= [-l*cos(i beta)+d; l*sin(i beta)]; & \text{ % Centerpoint} \\
R &= [cos(2*pi/n) -sin(2*pi/n); sin(2*pi/n) cos(2*pi/n)]; \\
p1 &= [d; 0]; \\
\text{\textbf{for} } j = 1:n \\
p2 &= R*(p1-cn)+cn; \\
set(s(j), 'xdata', [p1(1) p2(1)], 'ydata', [p1(2) p2(2)]); \\
p1 &= p2;
\text{\textbf{end}} \\
\text{\textbf{hold on}} \\
dp &= sqrt(l*r*r-2*l*r*cos(pi-phi)); \\
da &= i beta+\texttt{asin}(r*sin(pi-phi)/dp); \\
dx &= d-dp*cos(da); \\
dy &= dp*sin(da); \\
set(dl, 'xdata', [dx cn(1)+0.5*(dx-cn(1))]); \\
set(dl, 'ydata', [dy cn(2)+0.5*(dy-cn(2))]); \\
\texttt{ip} &= sqrt(l*l+w*w-2*l*w*cos(pi-phi-itheta)); \\
\texttt{ia} &= i beta+\texttt{asin}(w*sin(pi-phi-itheta)/ip); \\
\texttt{ix} &= d-ip*cos(ia); \\
\texttt{iy} &= ip*sin(ia); \\
\texttt{set} \ (il, 'xdata', [ix cn(1)+0.8*(ix-cn(1))]); \\
\texttt{set} \ (il, 'ydata', [iy cn(2)+0.8*(iy-cn(2))]); \\
\texttt{set} \ (dcir, 'xdata', cn(1)+rx, 'ydata', cn(2)+ry); \\
\texttt{set} \ (wcir, 'xdata', cn(1)+wx, 'ydata', cn(2)+wy); \\
\texttt{sn} &= \texttt{linspace}(0,itheta,50*abs(itheta));
b = (0.8*w-r)/itheta;
sx = -(a+b.*sn).*cos(sn+phi+ibeta);
sy = (a+b.*sn).*sin(sn+phi+ibeta);
set(sp, 'xdata', cn(1)+sx, 'ydata', cn(2)+sy);
if itheta >= 0
    set(sp, 'Color', 'green')
else
    set(sp, 'Color', 'yellow')
end
drawnow;
writeVideo(aviobj, getframe(anifig));
pause(0.001);
end
end

% Increment actual steps
rstep = rstep + isstep;
end
close(aviobj);