LOCOMOTION OF A CYLINDRICAL ROLLING ROBOT WITH A SHAPE CHANGING OUTER SURFACE

By

MICHAEL GERALD PUOPOLO

Bachelor of Science in Physics
University of California, Riverside
Riverside, CA
1994

Master of Science in Engineering
California Polytechnic State University, San Luis Obispo
San Luis Obispo, CA
1998

Submitted to the Faculty of the
Graduate College of
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
March, 2017

LOCOMOTION OF A CYLINDRICAL ROLLING ROBOT WITH A SHAPE CHANGING OUTER SURFACE

Dissertation Approved:

Dr. Jamey Jacob

Dissertation Adviser

Dr. Ron Delahoussaye

Committee Member

Dr. James Kidd

Committee Member

Dr. Russ Rhinehart

Committee Member

Dr. Leticia Barchini

Outside Committee Member

Name: MICHAEL GERALD PUOPOLO

Date of Degree: MARCH 2017

Title of Study: LOCOMOTION OF A CYLINDRICAL ROLLING ROBOT WITH A SHAPE CHANGING OUTER SURFACE

Major Field: MECHANICAL AND AEROSPACE ENGINEERING

ABSTRACT: A cylindrical rolling robot is developed that generates torque by changing shape (eccentricity) of its elliptical outer surface. Shape change commences four times per revolution, whenever one of four elliptical axes rotates past an inclination called trigger angle. The robot is equipped with a sensing/control system by which it measures angular position and angular velocity, computes error with respect to a desired step velocity profile, and changes shape of the outer surface at these times accordingly. A series of trial rolls are conducted using various trigger angles, and energy consumed by the robot's singular servo motor is measured and used to calculate energy consumed per unit roll distance. Results show that for each of three desired velocity profiles tested, there exists one or more trigger angle values that result in relatively low energy consumption per unit roll distance. When the robot operates within this optimal trigger angle range, it undergoes minimal actuation burdening and inadvertent braking, both of which are inherent to the robot's control system and act to lower the robot's energy economy during locomotion. Moreover, this optimal range generally shifts towards the vertical as angular velocity of the robot increases. A mathematical model of the robot's motion is developed and applied in a simulation program that is used to predict, analyze and further understand the robot's locomotion.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Ground based robots typically move from place to place using wheels, legs, or by changing shape in a biomimetic fashion, as with peristaltic or slithering locomotion [1, 2, 3]. Wheeled robots are the most common of these three locomotion styles, because in general, wheeled robots are efficient and they can move faster than other types of ground based robots [1]. A special class of wheeled robots is the rolling robot, which rolls exclusively on an outer, driven surface that entirely envelopes the system [4]. Rolling robots, whether spherical or cylindrical in form, have several advantages over traditional wheeled and legged robots. The outer surface of a rolling robot restricts the possibility of tipping over on a side from which the robot cannot recover, and it provides mechanical and environmental protection for the vulnerable systems on the robot. Also, the outer surface is typically large in diameter and can therefore act as an effective traction device for traversing paths of varied terrains, and can even be equipped to jump and bounce over otherwise insurmountable obstacles [5].

A rolling robot that moves by its own energy, in contrast to one moved by wind, for example, must somehow generate torque to make itself roll. The study herein investigates velocity control of an autonomous rolling robot, developed at the Unmanned Systems Laboratory at Oklahoma State University (OSU) and pictured in Fig. 1.1, that generates torque through changing shape of its outer surface, which is flexible and can be morphed to retain oblongness about one of two notional elliptical axes that are fixed to the robot and roll with it. When the OSU rolling robot,

Figure 1.1: The OSU rolling robot has a mass of 0.950 kg with a perimeter of 2.095 m.

as it is entitled, is configured as a tilted, upright ellipse, traction and gravity forces generate torque about the robot center, causing the robot to simultaneously fall and roll, similar to the motion of a cam or egg when placed on end. Once in motion, the robot repeatedly and automatically changes its outer surface shape in order to continue rolling forward.

In addition to being a shape changing and a partially gravity powered robot, locomotion of the OSU rolling robot is categorized as dynamic [6], meaning it has a natural rocking tendency and exhibits inertial motion. In other words, if the outer surface were to suddenly stop changing shape in the middle of a roll, the robot would likely require significant time to come to rest. Non-dynamic rollers, such as crawling rolling robots, move slowly in comparison and do not exhibit dramatic inertial effects after surface transformation or other motion input has ceased [7, 8]. The OSU rolling robot is also an underactuated system [9], referring to how the robot exploits its own natural dynamics in order to achieve steady, rolling locomotion.

1.1 Motivation

Various previous investigations have shown shape change to be a viable method for controlling rolling robots [10, 6, 11, 7]. Rolling loops, through shape change alone, have been controlled to commence from rest and move forward quickly and efficiently [6, 11]. Up to now, however, the challenge of controlling a dynamic rolling robot to follow a desired velocity profile through shape change alone has largely been unexplored by robotics researchers. In a New Scientist article that showcased the rolling robot of Sugiyama and Hirai [7], a roboticist at the University of Manchester alluded to this fact when he commented that rolling robots using shape change to produce locomotion will be difficult to control accurately [12, 13]. One reason why this next step in the research has not yet been taken is that such robots require lightweight yet nimble actuation, along with accurate sensing of position and velocity, in order to achieve satisfactory onboard control. It is only during the last ten years that advances in sensor technology have made it possible for gyroscopes, for example, to be lightweight, low-power and accurate enough for employment onboard rolling robots that simultaneously translate and rotate [14]. Actuating devices, too, such as servo motors, have become lighter and stronger. Thus, implementation of velocity control onboard a shape changing rolling robot is not only an attractive challenge, but an opportune beckoning. The work presented herein is motivated by this challenge and the potential applications that could result from successfully meeting it.

At least three previously documented research projects [6, 11, 15] involving shape changing rolling robots have used the modular loop configuration, in which six or more servo modules are stacked end-to-end to form a loop robot. These modular loops typically have a high number of degrees of freedom [13], allowing them to potentially attain divers shapes and thereby experiment with various rolling gaits [11]. In addition, modular loops can easily be scaled by simply adding or subtracting one or more modules. Yet with all the benefits of the modular platform, rolling modular loops are bulky and overly complicated. An attractive challenge – and one that is undertaken by the study herein – is to strive for velocity control of a dynamic rolling robot whose outer surface is, in contrast to the modular loops, thin and lightweight and can be morphed using a singular servo motor. Such a design is simpler and more elegant than the modular loop design. A conceivable, future extraction of this idea is a dynamic roller whose outer surface is composed of an adaptive material such as Nitinol, pneumatic muscles, or a thermally tunable foam that acts as frame, wheel and actuator. Imagining a robot such as this one, which is perhaps not too far-off in the future, highlights how shape changing rolling robots have potential to be simple, ultralightweight, efficient robots for use in various applications.

As an example of possible application, shape changing rolling robots offer a compelling architecture for robots that would be used as rovers in space exploration. NASA has expressed interest in developing exploration rovers for the Martian surface whose mobility would be derived from the planet's powerful surface winds. According to a NASA technical document [16], the concept that NASA would like to investigate is a lightweight, low-power data collection robot that would be blown about the surface of Mars like a terrestrial tumbleweed. A worthwhile candidate design for NASA's vision is a robot akin to the rolling robot investigated herein; one imagines a spherical, dynamic roller with a shape changing outer surface, several feet in diameter, that fully envelopes the payload, such as sensors and power source, protecting it from the harsh Martian environment. When winds are high, the robot would roll like a tumbleweed, but in low winds the robot could move to locations of interest by changing the shape of its outer surface to generate roll torque. Possibly the roller could be empowered with a self-induced bouncing ability in order to hop over otherwise insurmountable obstacles.

1.2 Contribution

Compared to other shape changing, dynamic rolling robots, the OSU rolling robot is unique in that it performs autonomous velocity control relative to a desired profile using all onboard sensing of angular position and velocity. Also, the robot is considerably simpler in design and lighter than the modular loops that have been developed in recent years. It possesses a flexible and continuous outer surface made of everyday plastic whose shape is periodically changed using the same servomotor model that's used on toy boats and planes. Power is supplied by two 9 V batteries that are sufficient for dozens of 30-meter trial rolls. Although the robot itself is a new and simple design, the subtle aspects associated with its shape changing, rolling locomotion are complicated, and such aspects of this locomotion style have previously not been unearthed. The main contribution of the research contained herein is that it provides insight concerning these aspects. It is the author's wish that the lessons learned herein will be applied to future research projects involving shape changing rolling robots that would improve upon the design of the OSU rolling robot.

1.3 Goals and Objectives

Three goals are associated with the research contained herein. The first goal is to develop an autonomous rolling robot whose only means of generating roll torque is to change shape of its outer surface in order to accurately follow a desired, step angular velocity profile. The second goal is to identify shape change actuation timing strategies that minimize energy per unit roll distance of the robot for a range of desired velocity profiles. Finally, the author wishes to understand and communicate why, given the locomotion dynamics of the robot and its mechanical system, certain actuation timing strategies are superior in terms of energy economy.

Associated with these goals are six objectives that partition the goals into specific, tractable steps with each subsequent step building upon the last. The six objectives of the research are:

1. Design and develop the rolling robot, including outer surface, actuator, sensing systems, control system, and communication system.

2. Define stability for robot locomotion, and demonstrate that the robot is stable while rolling in a controlled manner on a flat and level surface.

3. Develop a mathematical model for the robot and a related simulation computer program that predicts robot behavior and can be used to analyze and further understand it.

4. Perform a series of trial rolls of the robot using three different desired step angular velocities, in which a control parameter that governs timing of shape change actuation is varied from roll to roll, while energy consumed by the actuator is measured.

5. Using results from 4, identify a value or range of values of the control parameter for each desired steady state velocity that results in minimal energy consumed per unit distance rolled.

6. Provide an explanation for why the range of optimality for the timing parameter exists.

1.4 Outline

This study herein is outlined as follows. Chapter two is titled Previous Research. It presents several previous research projects that are relevant to the OSU rolling robot. Some of these previous works provide creative or technical incentive, and others are included because they illustrate the limits of past achievements. Chapter three, System Design, describes the OSU rolling robot in detail, including the robot outer surface, actuator, onboard sensors and general behavior. It also describes formulation of the mathematical model and explains how the robot control system works. Chapter four, Experimental Methodology, explains how and where the primary experiment is conducted. Chapter five, Results, presents results from the primary experiment and also contains a section that compares simulation predictions to trial rolls. The last section, entitled Discussion and Conclusion, describes the lessons of the research, gives suggestions for future work, and summarizes the research as a whole.

CHAPTER 2

Previous Work

The OSU rolling robot is characterized by three principal traits that describe its form and function. Namely, the robot is (a) a loop that generates torque by (b) changing its shape in order to (c) achieve locomotion that is partially gravity powered. The first of these traits refers to the ribbon-like shape of the robot outer surface, which causes it to roll in a straight line as if it were on a restrictive track. This trait differentiates motion of the robot from that of spherical rolling robots, for example, that are typically not restricted to roll in any one direction. The second trait, shape changing, means the OSU robot generates roll torque by morphing it's overall form, something that sets the OSU rolling robot apart from traditional walking or wheeled robots whose parts are essentially spinning, rigid bodies. The third trait refers to the falling/rolling motion that is germane to controlled locomotion of the robot. This trait differentiates the robot from other locomotion styles that do not leverage the gravity force in a significant way to produce motion. These traits designate areas of previous work upon which the OSU rolling robot draws, and they are reviewed in this chapter.

Locomotion via gravity power has previously been studied as a field of interest in biomechanics. In [17], Mochon and McMahon presented a mathematical model of human gait using a notional three-link machine with muscles. It was assumed the muscles acted only to periodically configure the gait, leaving the machine to move entirely under the force of gravity for most of its motion. Others [18, 19] investigated gaits in which gravity alone generated motion of simple link-composed machines down a slanted ramp. Authors of [17]-[19] point to the simplicity and efficiency of gravity powered gaits, and the authors of [18] and [19] suggest that power and control could be added using small, strategically timed energy inputs that do not interrupt the machine's natural motion. Although the OSU rolling robot is not a simple, linked machine, the concept of applying strategically timed inputs during a gravity-powered gait is, in essence, how the OSU rolling robot works.

One of the earliest projects that pursued the idea of changing robot shape to produce a gravity-powered gait was performed by Lee and Sanderson in [20]. In that study, researchers formulated a model of a notional rolling truss, dubbed Tetrobot, that was composed of interlinked strut-and-node sections. By changing lengths of certain struts, the truss was repeatedly reconfigured into a shape that tipped and fell onto an adjacent side, thereby sustaining a continuous yet stuttered rolling locomotion. Timing of the reconfiguration was automatic and was based on node position feedback. Although the research performed by Lee and Sanderson did not include documented implementation, the idea of driving a rolling robot via shape change had largely been unexplored in previous robot studies.

Since Lee and Sanderson's work, researchers have developed various systems that change shape as a means to leverage the gravity force and ultimately produce rolling locomotion [6, 15, 13, 21]. Most of these newer systems are faster and more sophisticated than the Tetrobot. What's remarkable about shape changing rolling robots is that traditional actuation systems, such as motorized wheels or limbs, are foregone in favor of newly developed actuation systems, such as electroactive polymers [21] or SuperBot modules [15], that bring about profound changes to robot shape in order to produce locomotion. Rather than having a part of the robot dedicated to spinning while another part is dedicated to actuation or framing, the deformable parts of shape changing robots can act, for example, as frame, wheel and actuator all-in-one. The resulting styles of locomotion exhibited by shape changing, rolling robots are often

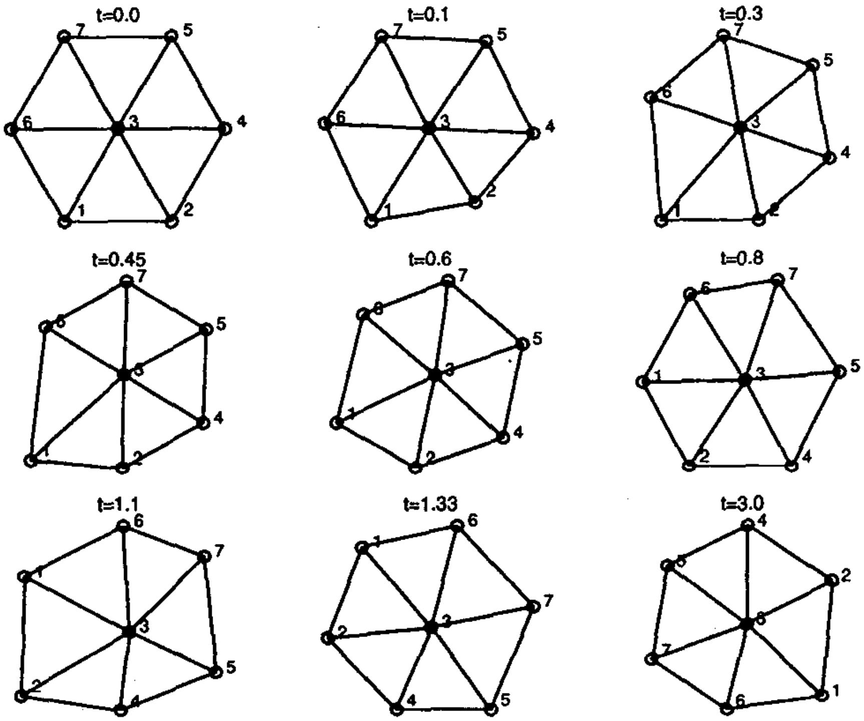


Figure 2.1: A representative illustration of the rolling motion of the Tetrobot truss. By changing lengths of certain struts, the truss changed into a shape that repeatedly tipped and fell onto an adjacent side. Image is from [20].

awe-inspiring as well as operationally efficient [6, 21].

With the rise in popularity of modular robotics, several rolling robots have been developed using SuperBot [2, 15] or CKBot [6] servo motor modules. In these module-composed robots, six or more modules are stacked end-to-end to form a loop. Each module in the loop has a set of sensors, and each module moves with respect to neighboring modules via individually controlled servo motors [2, 15] that are set into each module. When merged into a loop, the resulting system of stacked modules is highly configurable and can retain various shapes, allowing roboticists to experiment with different locomotion strategies and gaits.

Recognizing the potential of the modular platform, Shen *et al.* investigated motion of a notional rolling robot composed of six SuperBot modules connected end-to-end in a loop[10]. The robot had the capability to retain two shapes: a regular hexagon and a deformed hexagon with a tip-over tendency. Using orientation of the modular sections

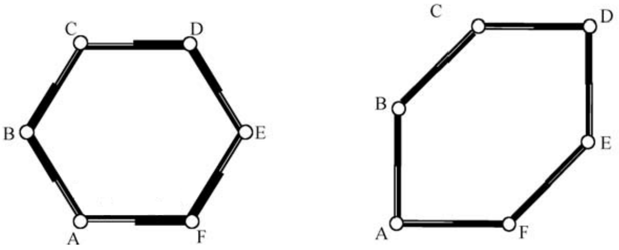


Figure 2.2: The modular robot envisioned by Shen could retain the shape of a hexagon with a tip-over tendency. Image is from [10].

with respect to the vertical as feedback, model simulations predicted the notional robot could roll along by automatically switching between its two allowable shapes, as illustrated in Fig. 2.2. Simulation results of Shen's rolling robot mirrored those of the Tetrobot. However, Shen's work was progressive because it demonstrated notional capability of a shape changing, loop-shaped robot to perform feedback controlled locomotion.

In [6], Sastra *et al.* presented a modular robot that realized the notions proposed by Shen *et al.* Sastra's rolling robot was composed of ten individually motorized CKBot modules connected end-to-end in a decagonal loop. When configured with a tip-over tendency, the loop retained a dual-arc shape that resembled the outline of an on-end American football. On each module of the robot there was a touch sensor for knowing when a side of the loop was in contact with the ground, so that when the robot rolled and a side of the decagon touched down, the loop quickly morphed into a newly configured football shape, now with its long axis aligned closer to the vertical, and continued rolling forward until the next side of the loop touched down. In this manner, the rolling loop sustained a dynamic roll using touch sensor information as feedback. Terminal velocity during experimental rolls was arbitrarily increased or decreased by changing width of the football shape; a thin, longer football resulted in faster motion on average than a wider football. Power consumption of the robot was

Figure 2.3: The CKBot roller developed by Sastra *et al.* had ten individually motorized modules connected end-to-end in a loop. Image is from [6].

measured over several trial rolls of the robot, and it was found that efficiency of the robot locomotion decreased as terminal velocity of the robot increased. The CKBot roller developed by Sastra *et al.* is pictured in Fig. 2.3.

Mellinger *et al.* presented a CKBot roller composed of twelve modules with the capability to assume elliptical polygonal shapes as it rolled [11]. Three different gaits were envisioned and implemented on the robot. In one of these gaits, the robot maintained a constant elliptical shape with a perpetual off-balance orientation while modules moved around the loop like a tank tread. Researchers implemented two other gaits that involved changing eccentricity of the ellipse as it rolled. These gaits included start-up phases where the robot commenced from rest in a stable position and reached constant angular velocity. During experimental rolls, an external motion capture system tracked roll angle and angular velocity of the robot, which were communicated back to the robot and used as control feedback.

Melo *et al.* presented a model of a notional modular rolling loop with up to 14 modules connected end-to-end. The model assumed an elliptical shape for the loop and approximated motion of the loop by treating it as an inverted pendulum [22]. In simulations of robot motion, the loop maintained a constant elliptical shape with a tilted orientation, while modules moved around the shape like a tank tread, similar to the off-balance gait demonstrated Mellinger *et al.* in [11]. The purpose of the investigation was to see if varying orientation angle of the ellipse would produce different terminal velocities of the robot center of mass. Two shapes were used for the ellipse in the simulations, one which was slightly eccentric and one which was highly eccentric. Also, the number of modules in the loop was varied. Results showed that, in theory, orientation angle could be varied in order to produce different terminal velocities of the rolling robot.

Chiu *et al.* constructed a modular rolling loop using six SuperBot modules according to the vision of Shen in [10]. The robot was programmed to deform its shape in a manner similar to the gait style proposed in [10]. Using their modular rolling loop, researchers performed an experiment that measured efficiency by lapping the robot around an inside building corridor for a total distance of over one kilometer while voltage in the modules' batteries was monitored. The robot traveled along a straight line, and turning was done by manually redirecting the robot. Results showed that the rolling robot was able to travel 1142.5 m in 54 minutes before at least one of the modules' batteries became exhausted.

Sugiyama and Hirai presented a shape-changing rolling robot (Fig. 2.4) that resembled a bicycle wheel with a deformable outer surface that was supported by several spokes made of a shape memory alloy [7]. When some of the spokes were energized with electric current, they contracted and deformed the outer surface into an elliptical shape. By repeatedly applying current to successive spokes, researchers produced a slow and steady crawling motion. Locomotion of the robot was not categorized as dynamic; however, this research demonstrated that specialized materials could be used to compose and actuate a rolling robot with a shape changing, continuous outer surface.

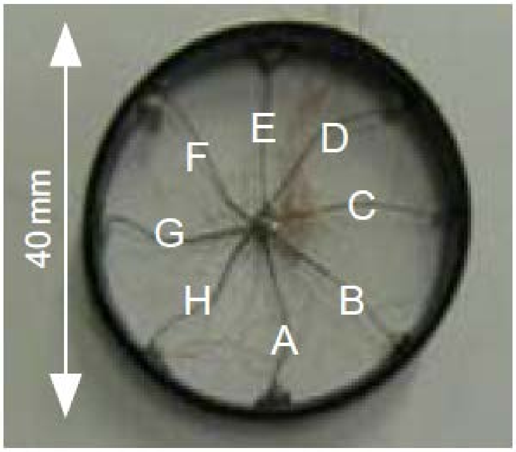


Figure 2.4: The shape changing rolling robot presented by Sugiyama and Harai had a deformable outer surface with spokes made of a shape memory alloy that contracted with current. Image is from [7].

CHAPTER 3

System Design and Modeling

This chapter describes the physical characteristics of the OSU robot, how it works, and how it is modeled. The first section provides general description of the robot, including details about its outer surface and the actuator that causes it to change shape. Because the robot control system relies on measurements taken onboard, attention is given to the robot sensors in the second section. Mathematical modeling of the robot, described in the third section, is attained through establishing a coordinate system that quantifies robot motion, constructing a free body diagram, and applying Newton's Second Law to formulate differential equations of motion. Static bending of the outer surface and rolling resistance is included in the model, as well as characterization of the actuator, which is achieved through high speed video measurement. Detailed description of the robot velocity control system is included in the fifth section, and the last section describes a computer program, based on the mathematical model of the robot, that is developed to simulate controlled motion of the robot.

3.1 Robot General Description

The outer surface of the rolling robot is a flat, 0.318 cm by 5.1 cm, strip of polyvinyl chloride plastic put into the shape of an open, elliptical cylinder with a perimeter of 2.095 m (Fig. 1.1). The cylinder is firm enough to provide a steady, dynamic roll surface for the robot yet limber enough to be significantly reshaped by the pull of a servo motor located inside the volume of the cylinder. On a smooth and level floor, the robot rolls in one direction along a straight path without leaning to the side or tipping over. Where the robot meets the floor, the outer surface bends slightly under the weight and motion of the robot. Total mass of the robot is 0.950 kg.

Contained onboard the robot and inside the cylindrical outer surface are a microprocessor board, inertial measurement unit (IMU) board, two rechargeable 9 V batteries, mechanical switches that measure angular position of the robot, a linear actuator, and an electrical energy sensor whose components are set mostly in a plastic breadboard. There is also a radio transmitter, an radio frequency module [23], that sends data regarding the robot locomotion to a receiver-equipped, external laptop computer for further analysis. The microprocessor [24] reads position, angular velocity, and energy data from the sensors. Using angular position and angular velocity as feedback, the microprocessor brings about shape change of the robot outer surface in order to affect velocity.

3.2 Batteries

The two 9 V batteries that power the robot are EBL's "high volume" $6F22$ model [25], a rechargeable lithium battery with a capacity of 600 mA. The batteries are attached to the robot via Velcro, so they can be easily removed for recharging and then be reattached. When power in the batteries is depleted, the batteries are removed from the robot and placed into a dual-receptacle recharging unit that plugs into a standard 120 V electrical outlet. There is a light-emitting diode (LED) on the charger, one for each receptacle, that turns on when a battery is placed into unit for recharging. According to instructions that come with the batteries/charger, the LED stays at full brightness until voltage of the battery reaches 8.1 V. Then the LED dims as the battery is further charged, until it finally turns off when the battery is fully charged at approximately 8.45 V. Using the charger, it takes about 3 hours to fully recharge two moderately depleted batteries simultaneously.

A series of test is performed in order to know the level of performance of the

EBL batteries and demonstrate that they are suitable for powering the robot during experimentation. With the EBL batteries fully charged and connected in parallel to the robot system as in the circuit diagram of Fig. 3.9, fifty trial rolls are performed with the robot consecutively without recharging the batteries. (The trial rolls are conducted in a manner that is identical to the procedure for trial rolls performed in the primary experiment described in Section 4.6.) Energy consumed by the servomotor for each of the fifty rolls is measured by an onboard energy sensor (Section 3.4.3) and transmitted to a laboratory computer where the data is saved. After the test is completed, the batteries are fully recharged and the motor is allowed to cool to room temperature. The test is then repeated in this manner two more times for a total of three tests. Using saved information from the three tests, energy consumed is plotted versus trial roll number for each test. In Fig. 3.1, the three plots are placed on a singular graph, in which axis grid lines are separated by one joule and the plots are arbitrarily shifted vertically with respect to one another so that each plot can be seen clearly.

Referring to Fig. 3.1, plots for the three tests show that energy consumption rises relatively quickly during the initial ten, or so, rolls. This rise is likely due to sudden temperature change of the motor [26]. After the initial ten rolls, however, energy consumption levels out to a constant value, on average, for a span of about 30 rolls. For the last 10 rolls, energy consumption slightly decreases on average in all three tests. Overall, it is clear from the plots that the EBL batteries provide a strong and steady source of energy with no sudden spikes recorded during the tests. In fact, energy consumed by the servo motor changes by no more than one fourth of a joule from roll to roll in any given test, and this change is less than one percent of the total energy consumed by the servo motor during a typical trial roll in the test.

As part of the process of choosing batteries for powering the OSU rolling robot, the testing procedure outlined above for the EBL batteries is performed using other

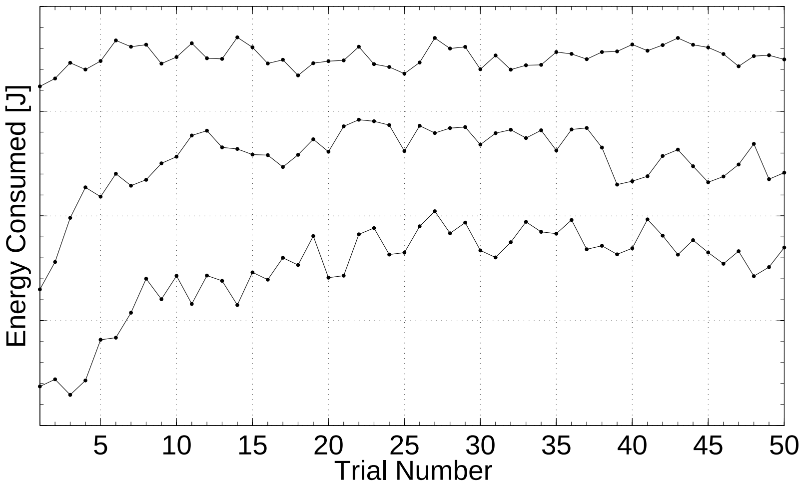


Figure 3.1: Three, fifty-roll tests of the robot powered by EBL 9 V batteries shows that energy consumed by the servo motor levels out after the initial ten trials. Plots are arbitrarily shifted, and axis grid lines in the graph are separated by one-half joule.

types of batteries: a pair of rechargeable nickel-metal hydride (NiMH) batteries, a pair of non-rechargeable lithium-ion batteries, and a pair of non-rechargeable alkaline batteries. Results from these additional battery tests are summarized as follows. The NiMH batteries are heavy and do not have sufficient energy capacity, so they are rejected. Non-rechargeable lithium-ion batteries are lightweight and have relatively high energy capacity, but they discharge capriciously and cannot be recharged/reused, so they are rejected. Regular alkaline batteries possess sufficient energy capacity and a very flat discharge characteristic, but they are heavy and cannot be recharged/reused, so they are rejected.

The EBL batteries are chosen for use on the rolling robot, because they possess a set of superior characteristics that no other singular battery, at least among the batteries that were tested, possesses. Namely, the EBL batteries have good charging cycle life [27], so their performance is repeatable over several thousand trial rolls of the robot [25]. The EBL batteries are also lightweight with a mass of only 26 grams each. In comparison to a pair of nickel-metal hydride batteries, for example, a pair of EBL batteries is 20 grams less in mass, and this difference makes a noticeable impact

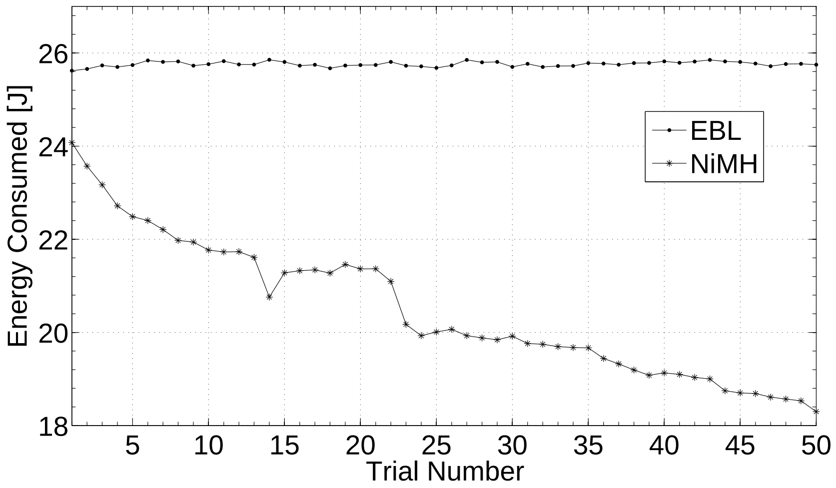


Figure 3.2: The EBL batteries provide a steady and long-lasting source of energy in comparison to the NiMH batteries tested.

on performance of the robot; even though performance design is not the aim of the research herein, having a rolling robot that is lightweight facilitates experimentation in general. Furthermore, as previously discussed in reference to Fig. 3.1, the EBL batteries have excellent energy capacity. To give an idea of the superiority of the EBL batteries in this respect, Fig. 3.2 shows a graph with plots of energy consumed versus trial number for the NiMH batteries and for the EBL batteries. Not only are the EBL batteries more energetic in general, but they provide an extraordinarily steady and long-lasting source of energy in comparison to the NiMH batteries. Lastly, the EBL batteries provide steady discharge [27] as shown by the nearly linear plot in Fig. 3.2. For the research performed herein, these latter two characteristics are imperative, because locomotion of various control strategies is compared according to energy consumed by the servo motor in the primary experiment (Section 4.6), so if energy were characterized by unpredictable spikes, or if frequent pauses were necessary for recharging the batteries, adverse uncertainties would be introduced, and the comparisons would be inconsequential.

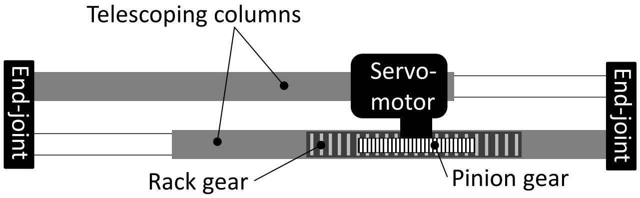


Figure 3.3: Shape change of the robot is achieved by means of a linear actuator that alters distance between end-joints that are epoxied to the inside of the outer surface.

3.3 Linear Actuator

Outer surface shape change is achieved via a linear actuator whose form is illustrated in Fig. 3.3. The actuator possesses two sets of telescoping columns comprised of interlocking male and female aluminum tubes. The tubes are secured by end-joints (Fig. 3.4) that are epoxied to the inside of the outer surface, creating an equal division of the outer surface perimeter. A servo motor, Hitec's HS-645MG [28], is fixed to a plastic support frame, which in turn is fixed to the female tube of one telescoping column. The motor is equipped with a pinion gear, whose angular position is governed by the microprocessor. Fastened to the female tube of the other column is a rack gear that meshes with the pinion gear. As the motor turns, the pinion gear displaces the rack, causing the end-joints to move away from (or towards) each other, thereby changing the outer surface shape. When the actuator is not being commanded to move, its static holding force maintains a constant cylindrical diameter of the outer surface. A third telescoping column, orientated in a perpendicular fashion relative to the linear actuator columns, is comprised of male and female tubes whose ends are secured to the inside of the outer surface. This third column, seen in Fig. 1.1 and in Fig. 3.5, is not a powered actuator. Rather, it serves as an air displacement damper to limit vibration and bending of the outer surface (Section 3.5.1).

The IMU board is mounted to the plastic support frame of the servo motor, and the 9 V batteries are attached to the servo motor housing with Velcro, so they can be easily removed for recharging and then reattached. The microprocessor and the

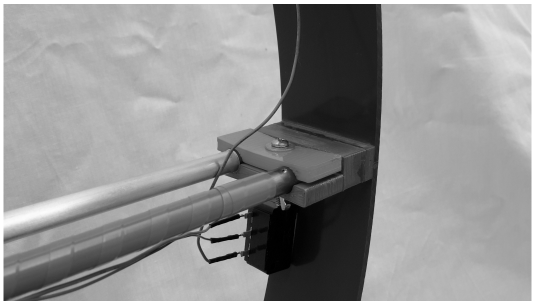


Figure 3.4: One side of an end-joint firmly holds the telescoping columns of the linear actuator, and the other side is epoxied to the inside of the outer surface.

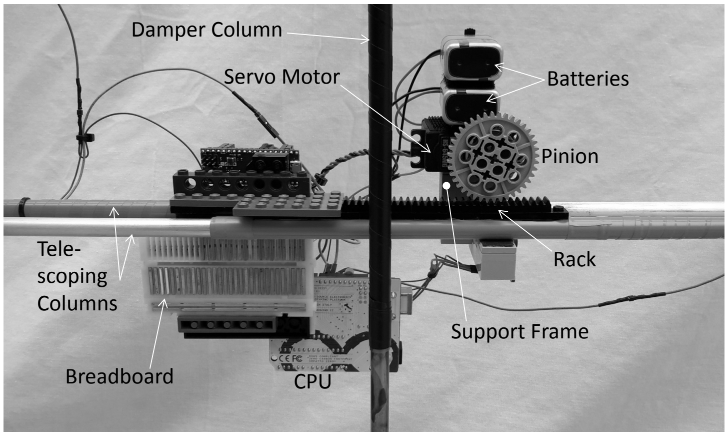
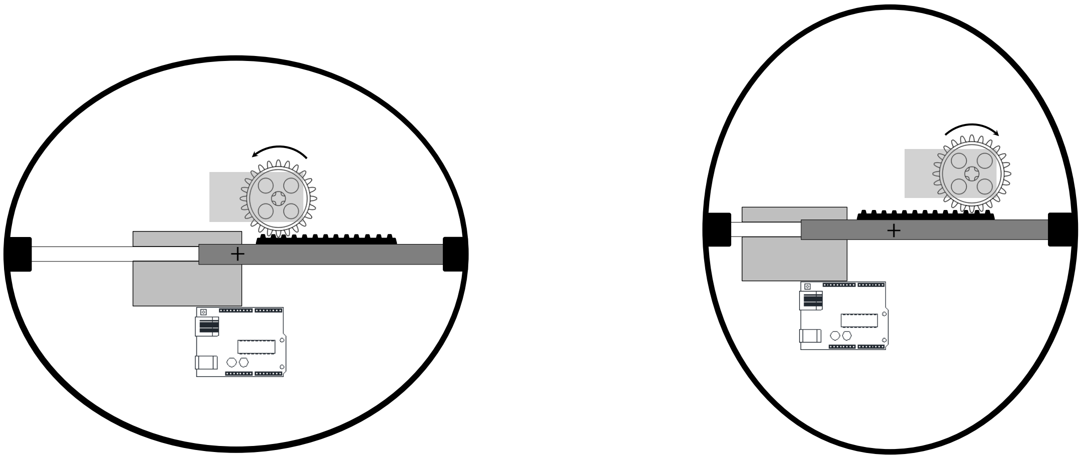


Figure 3.5: Various hardware items located inside the outer surface are fixed to the telescoping columns of the linear actuator.

plastic breadboard are fixed to the rack column. Hardware items mentioned here are pictured close-up in Fig. 3.5, and the IMU is discussed further in Section 3.4 and pictured in Fig. 3.7 and Fig. 3.8. Collective mass of the components that move with the rack column (breadboard, microprocessor) is approximately equal to the collective mass of the components that move with the pinion column (servo motor with support frame, batteries); and these component sets are fastened in such a way so their centers of mass are radially opposed in reference to the axis of the outer surface cylinder, regardless of elongation of the actuator. Consequently, the robot center of mass remains approximately at the same location, at the axis of the cylinder, even as the robot changes shape, as illustrated in Fig. 3.6. While a constant center of mass location is not necessary for locomotion of the robot, it allows motion to be rotationally consistent and simpler to model and predict.



(a) As the pinion gear turns counterclockwise, the telescoping columns push-out against the outer surface.

(b) As the pinion gear turns clockwise, the telescoping columns squeeze the outer surface together.

Figure 3.6: Simplified representation of the robot illustrating that center of mass, symbolized by a cross, remains at the axis of the outer surface cylinder regardless of linear actuator length.

3.4 Onboard Sensors

The rolling robot measures its angular velocity and angular position, and these values are used for feedback control. Angular position is measured with mechanical switches, and angular velocity is measured with a gyroscope. The robot also measures current and voltage at the servo motor by means of a specially designed circuit, and uses these values to compute energy consumed per unit roll distance, which is the central measurement made in the primary experiment described in Section 4. Measurement of current and voltage, along with computation of energy, are performed by electrical components that are collectively called the energy sensor. These three sensors – mechanical switches, gyroscope and energy sensor – are described in this section.

3.4.1 Gyroscope

The gyroscope used onboard the OSU rolling robot is ST Microelectronics' *L3GD20* angular rate sensor [29], a $3-$axis micro-electrical-mechanical gyroscope in a coin-

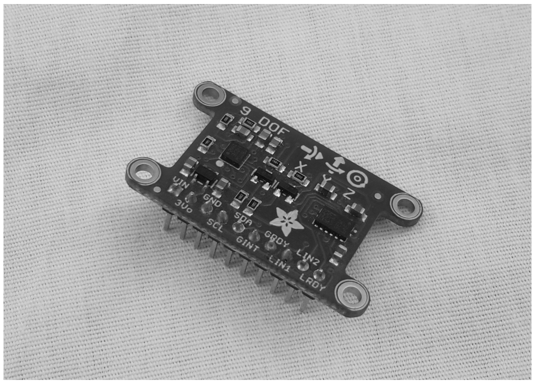


Figure 3.7: The IMU board employed on the rolling robot is coin-sized with a mass of 4 grams. Axes of the gyroscope are defined on the board.

sized IMU breakout board [30]. The *L3GD20* gyroscope is extremely lightweight, low-power and inexpensive [14]. As employed on the robot, the *L3GD20* is not altered from its factory calibrated state with a 16 bit-rate data output with a full-scale limit of 500 degrees per second [29]. According to ST Microelectronics, the *L3GD20* is insensitive to linear acceleration and vibration as it measures angular rate [31]. This characteristic heavily influenced the choice of using the *L3GD20* onboard the rolling robot, because the gyroscope employed therein will undergo a combination of rotational and translational motion, yet it must continually feedback accurate measurements in order for the control system to be effective.

The IMU board that houses the *L3GD20* gyroscope is powered by 3.3 V and communicates with the CPU via a two-pin I^{2}C bus [32]. Orientation of the gyroscope's three axes are defined on the IMU board, as pictured in Fig 3.7. The board is fixed to the rolling robot servomotor housing (Fig 3.8), so that the gyroscope's x-axis is parallel to the cylindrical axis of the outer surface, which is also the roll axis of the robot. Electronic connections between the IMU board and microprocessor are shown in the circuit diagram in Fig. 3.9.

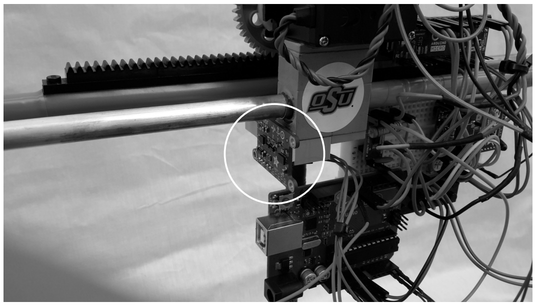


Figure 3.8: IMU board, marked here with a circle, is fixed to the servomotor housing.

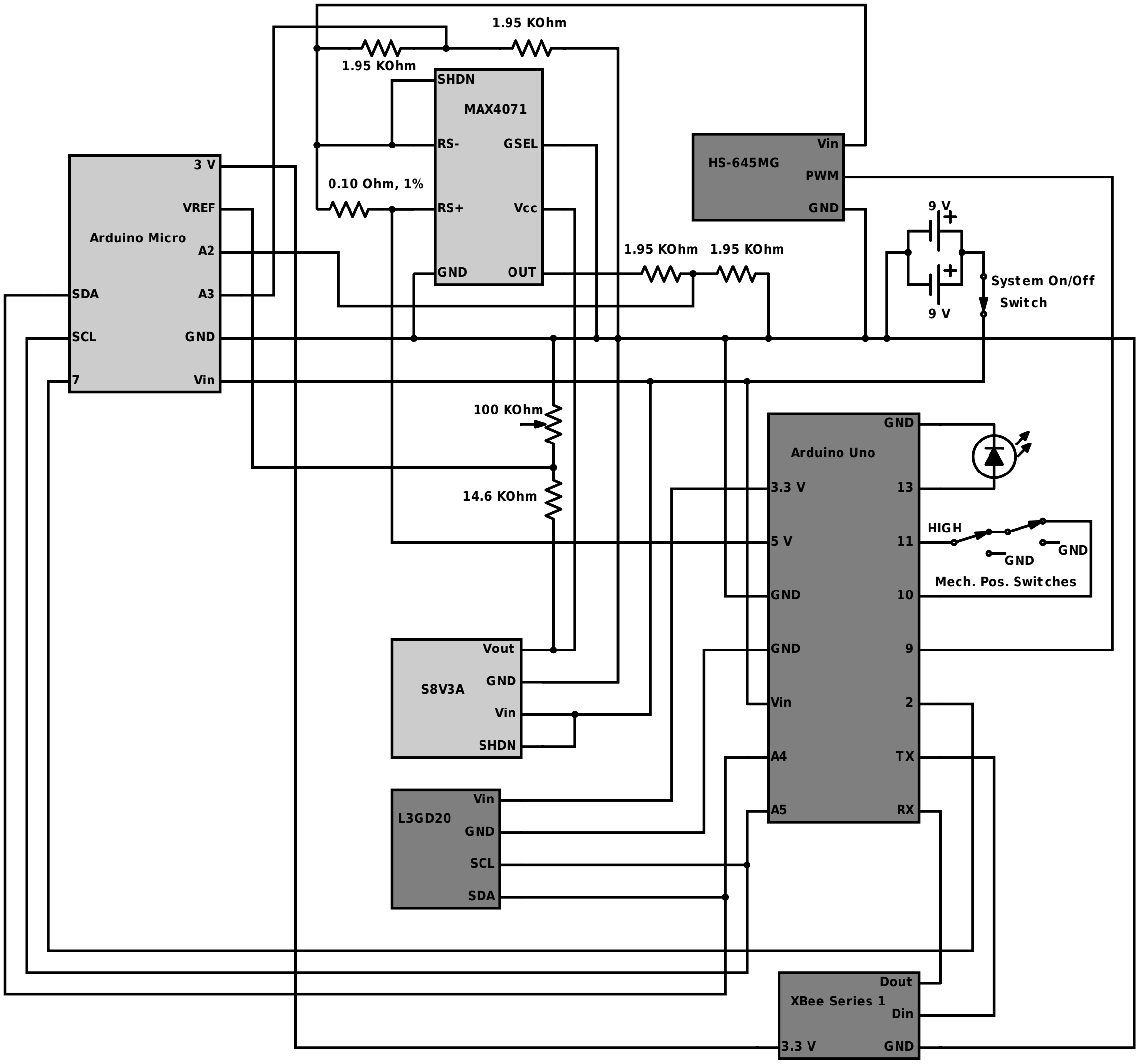
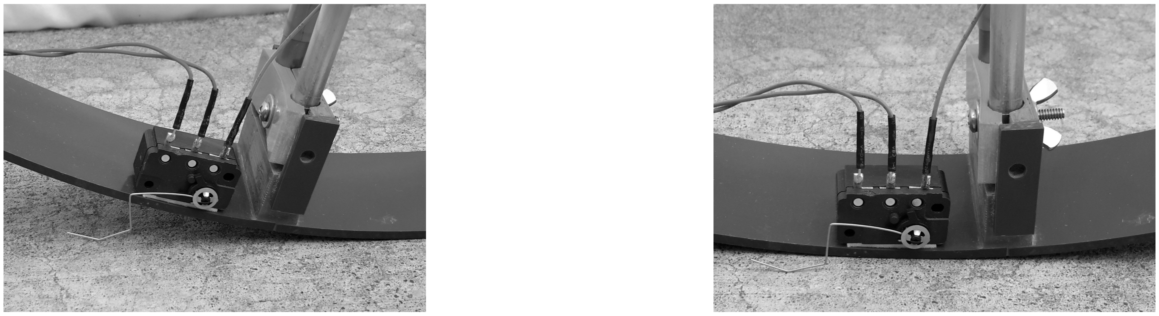
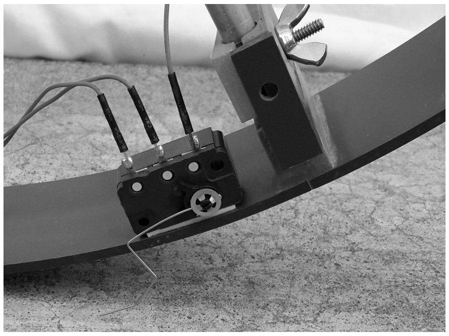


Figure 3.9: Circuit diagram showing connections of the rolling robot electronic components. Light gray-colored integrated circuits are main components of the robot energy sensor.



(a) Switch is unactivated.

(b) Switch is newly activated as A enters the vertical position.



(c) Switch snaps back and is again unactivated.

Figure 3.10: Position of the robot is established by mechanical switches fixed to the inside of the outer surface. As the robot rolls to the left (progression is a-b-c) a mechanical switch is activated, momentarily establishing angular position of the robot to the microprocessor, and then released.

3.4.2 Position Switches

Angular position of the rolling robot is measured using two mechanical snap-action switches, fastened to the inside of the outer surface, that are activated by rotating a spring-loaded lever arm. When not activated, the lever arm switch extends beyond the outer surface, so that as the floor and outer surface meet near the position of a switch, the lever arm gently presses against the floor until the switch is activated. As the robot continues to roll, the switch moves up and away from the floor, and spring action of the lever arm eventually causes it to snap back to its default, inactivated position.

The two mechanical switches are part of an electrical circuit with a digital voltage line that's read by the onboard microprocessor (Fig 3.9). The circuit is designed so that when both switches are in the default position, voltage in the digital line is high. However, when one of the switches meets the roll surface and is activated, voltage in the digital line momentarily goes low. The mechanical switches are positioned along the outer surface in a manner such that activation occurs whenever A is in the vertical position. Therefore, by keeping track of when and how many times voltage goes low in the switch circuit during a roll, and as long as initial angular position is known, the microprocessor establishes angular position, θ (defined in Section 3.5), every half rotation of the robot. By way of example, if the robot commences from the -25° position, θ is established at -180°, -360°,-540°, -720°, etc. If the robot commences from -270°, θ is established at -360° and at every half turn thereafter.

3.4.3 Electrical Energy Sensor

An electrical energy sensor that computes energy used by the servo motor is designed and employed onboard the rolling robot. At the heart of the sensor is a highly accurate current sensing chip, Maxim Integrated's $MAX4071$ [33], that works in cooperation with a 0.10 Ω, 1% tolerance sense resistor placed in series between the motor power supply, a regulated 5 V output from the microprocessor, and the motor load [34, 35]. Components and connections of the energy sensor are shown in Fig. 3.9, mostly in the top left corner of the circuit diagram. The $MAX4071$ accepts voltage across the sense resistor, amplifies it, adds a constant, and outputs the resultant voltage. This output voltage, which is representative of current that flows through the sense resistor and into the motor, is passed through a voltage divider and finally received at an analog-to-digital converter on a microprocessor [36] that is dedicated to energy computation and is separate from the main microprocessor. In a similar manner, voltage at the positive lead of the servo motor is scaled using a voltage divider and then passed to a second analog-to-digital converter on the dedicated microprocessor.

For discussion purposes, these voltages that are received by the microprocessor are dubbed V_1 and V_2.

The linear actuator on the rolling robot (Section 3.3) moves by means of a servo motor that is documented as having a pulse refresh width of 20 ms [37]. In order to verify this characteristic, a test is performed in which the robot is secured in the $\theta = 0$ orientation to a sturdy horizontal surface, and the linear actuator is programmed to perform a back-and-forth motion profile, similar to profiles that would be carried out by the actuator during controlled, rolling motion of the robot. While the test profile is executed, V_1 and V_2 are observed using a digital storage oscilloscope with a sample rate of at least $100,000$ Hz, and it is confirmed that waveforms of V_1 and V_2 have periods of 20 ms. Example portions of V_1 and V_2 saved by the digital oscilloscope at different times during the motion profile are plotted in Figures 3.11 and 3.12.

Referring to Fig. 3.11, the waveform of V_1 is a rectangular pulse with a lower baseline of approximately 0.80 V, which corresponds to a state of zero current of the servo motor. Above this baseline, servo motor current is positive, meaning current flows from the batteries into the servo motor. However, due to back electromotive force of the servo motor [38], sometimes current becomes negative, meaning it flows from the motor into the batteries. When current is negative, V_1 dips below 0.80 V. A few occurrences of negative current are seen in the plot of V_1 in Fig. 3.11 as downward voltage spikes at the leading and trailing edges of the rectangular pulses in the left side of the graph. These occurrences are so short-lived and infrequent that they are ignored by the sensing program, i.e. when $V_1 < 0.80$ V, servo motor current is assumed to be zero.

Width of the rectangular pulse in the V_1 waveform is observed to vary from roughly 3 to 20 ms over the full motion profile, and this variation depends on whether the motor is holding or spinning. When spinning, the waveform looks like the far left pulses in Fig. 3.11 that are relatively tall and wide; and when the motor is holding, the pulse gets shorter and thinner, like the pulse at the far right side of the plot. If pressure in contraction or extension is put on the linear actuator while it is holding, the waveform stays the same, except the width of the waveform increases. Similarly for V_2, the waveform varies depending on whether the motor is holding or spinning. When spinning, the V_2 waveform looks like the plot on the left side of the graph in Fig. 3.12, in which voltage periodically dips below the baseline level of approximately 3.40 V. When the motor is holding, the V_2 waveform remains higher on average, like the form of the voltage plot on the right side of the graph in Fig. 3.12. If pressure in contraction or extension is put on the linear actuator while it is holding, the waveform stays the same, except the width of the waveform increases.

During operation of the rolling robot, a computer program (Listing 1 in the Appendix) running on the dedicated microprocessor samples V_1 and V_2 five times every millisecond, amounting to 100 samples per pulse refresh width of the servo motor. The program converts the digital values of V_1 and V_2 into floating point values, and then in order to undo voltage division performed in the circuit, the program scales V_1 and V_2 to obtain output of the $MAX4071$ chip and servo motor voltage in units of volts. As explained in Section 4.6.4, servo motor current, expressed by Equation 4.29, is a function of $MAX4071$ output voltage. Using that equation, the program solves for servo motor current and subsequently multiplies current and servo motor voltage, resulting in electrical power input to the servo motor [38] in watts. This sample and power computation process is repeated every cycle of the sensor program.

Battery power supply voltage varies from about 8.45 V at full charge to about 8 V when the actuator noticeably slows and the batteries must be removed to be recharged. Due to this significant variation in power supply, Pololu's $S8V3A$ voltage regulator [39], is employed on the energy sensor circuit. The $S8V3A$ regulator accepts battery voltage as input, steps-it-up to 9 V, and delivers it to the $MAX4071$, ensuring that even when the batteries have significantly discharged, the $MAX4071$ still

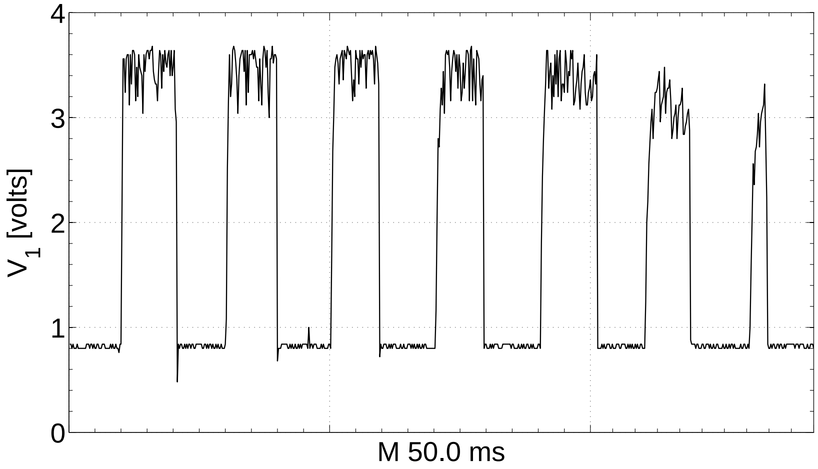


Figure 3.11: Voltage that is representative of servo motor current has a rectangular pulse waveform with a period of 20 ms. This voltage is sampled by the energy sensor 100 times per period.

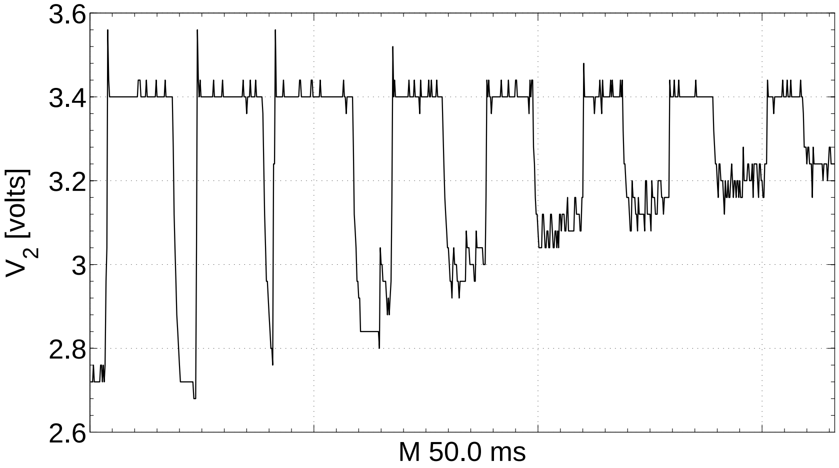


Figure 3.12: Voltage that is representative of servo motor voltage has a waveform with a period of 20 ms. This voltage is sampled by the energy sensor 100 times per period.

receives a constant, proper supply of 9 V. In fact, in extreme cases when the batteries have discharged well below 7 V, which is the lowest recommended input voltage of the main and dedicated microprocessors [24, 36], both microprocessors would automatically shut off before supply voltage to the $MAX4071$ would drop below 9 V. In as much as clipping of $MAX4071$ output voltage becomes an issue as its supply voltage dips below 8 V [33], this voltage regulation strategy guarantees that current measurement remains accurate within the error limits reported by the manufacturer for the $MAX4071$ chip [33].

At every cycle of the sensor program, average power of the servo motor is computed for the present and previous cycle of the program. The average is then multiplied by the program's sample period to give energy consumed by the motor in the span of one sample period of the sensor program [40]. A running sum of consumed energy is computed, resulting in total electrical energy used by the servo motor. At every tenth of a second after the sensor program has begun, total energy is saved as an element in an array, thereby logging total energy as a function of program runtime. When commanded by the main microprocessor program, the sensor program passes elements of the total energy array from memory on the dedicated microprocessor to the main microprocessor via an $\mathrm{I}^2\mathrm{C}$ serial bus [32] along the serial clock and serial data (SCL and SDA) lines [32] as shown in Fig. 3.9.

3.5 Robot Kinetic Model

A two-dimensional model of the robot is developed, in which the outer surface is a modified ellipse that rolls in one direction along a straight line in the horizontal-vertical plane. A laboratory-fixed Cartesian coordinate frame, XY, is established as the roll plane with X as the floor line. A dual-view illustration of the robot model situated in the XY coordinate frame is shown in Fig. 3.13. A moving coordinate frame, AB, is attached concentrically to the outer surface ellipse and rolls with it.

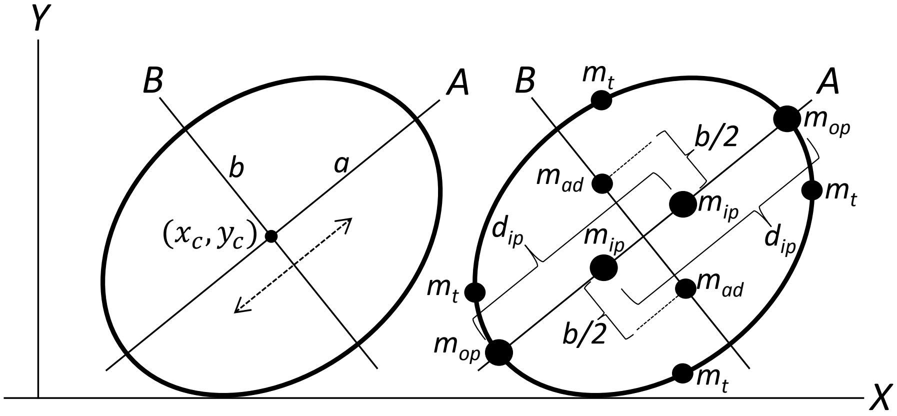


Figure 3.13: Rolling robot model is composed of an elliptical outer surface and point masses representing hardware components.

Axis A is placed along the line of motion of the actuator. Lengths a and b of the elliptical semi-major axes are measured along A and B, respectively.

Two pairs of radially opposed point masses located inside the ellipse on A represent the two hardware sets that are fixed to the columns of the linear actuator, as described in Section 3.1. The inner points on A each have a mass of $m_{ip} = 0.125$ kg and are fixed at distance, $d_{ip} = 0.022$ m, from the ellipse as shown in Fig. 3.13. The two outer points on A represent the end-joints described in Section 3.1, and each have a mass of $m_{op} = 0.015$ kg and are fixed on the ellipse. In addition, there are two points on B, each with mass $m_{ad} = 0.020$ kg, that represent the telescoping air displacement column. Points m_{ad} are placed a distance of $b/2$ from the neighboring section of the outer surface. Finally, there are four radially opposed hardware point masses placed on the outer surface ellipse. (These points are described in more detail in Section 3.5.3.) Center of mass of the robot is located at the intersection of A and B.

A free body diagram for the robot with the normal force, N, and traction, T, is shown in Fig. 3.14. Both forces act on the robot at the point where the outer surface ellipse makes contact with X, designated as the touchpoint. The value, x_e, is the horizontal position of the touchpoint with respect to the center of mass, and y_c is the vertical distance from X to the center of mass. The angle defined by the positive

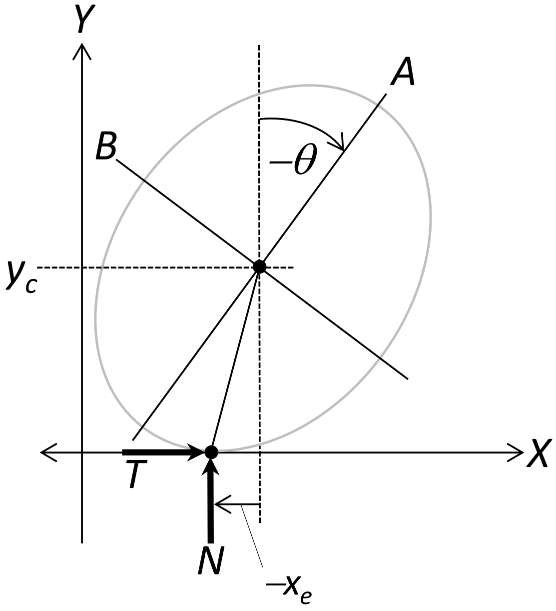


Figure 3.14: The outer surface can be modeled as a rolling ellipse with no bending.

branch of A and the vertical line passing through x_e is the roll angle of the robot, θ. (In keeping with the right-hand rule mnemonic [40], θ is negative with clockwise rotation.) By observing that X is equivalent to the tangent line of the ellipse at the touchpoint, the following expressions are derived for x_e and y_c [11]:

$$x_e = \frac{(a^2 - b^2)cos\theta sin\theta}{\sqrt{a^2cos^2\theta + b^2sin^2\theta}} \tag{3.1}$$

$$y_c = \sqrt{a^2cos^2\theta + b^2sin^2\theta} \tag{3.2}$$

3.5.1 Static Bending of the Outer Surface

When the actual robot is held by its actuator, so the outer surface is not in contact with the floor, its shape is almost perfectly elliptical. However, as soon as the robot is placed on the floor, the outer surface bends from its elliptical shape under weight of the robot. The bending is static, meaning that it occurs when the robot is not moving. Static bending of the robot is especially noticeable when the actuator is in an off-vertical angular position, in which case it is observed that in the general area where the robot contacts the floor, there is no easily distinguishable touchpoint, in contrast to the elliptical model of Fig. 3.14, where the touchpoint is easily distinguishable. Instead, the robot exhibits a touch segment, along which there is a distributed normal load, as illustrated with exaggeration in the diagram in Fig. 3.18. Also, it is observed that when the robot is placed onto the floor, the robot center of mass droops to a lower position than where it would be if the outer surface were completely rigid.

Establishing accurate placement of the normal and traction forces relative to the robot center of mass is critical to the model, and these distances are affected by bending of the outer surface. With this in mind, modifications are made relative to x_e and y_c in order to take into account the observed effects of static bending of the outer surface. Instead of being located at y_c, the center of mass is shifted to $y_c - d$ from X; and instead of x_e, the normal load – assumed to be concentrated at one central point – is shifted to $x_e + \Delta x$ from the center of mass. The Δx and d terms can be thought of as changes to x_e and y_c brought about by static bending of the outer surface. The modified locations of the center of mass and the normal force are expressed as new variables:

$$x_d = x_e + \Delta x \tag{3.3}$$

$$y_d = y_c - d \tag{3.4}$$

In order to quantify Δx and d, a test is performed. The robot is fixed in the circular configuration, and the center of the outer surface cylinder is indicated with a fiducial marker. The robot is then set on a sturdy horizontal surface and allowed to rest in the $\theta = 0$ orientation, which is measured with a highly accurate digital level. A digital camera is situated with its optical axis parallel to the cylindrical axis of the outer surface, and a picture is taken with the camera. The image is subsequently uploaded to image analysis software [41], in which a Cartesian coordinate system is assigned, and distances on the images are calibrated using a pixel-to-actual-distance ratio.

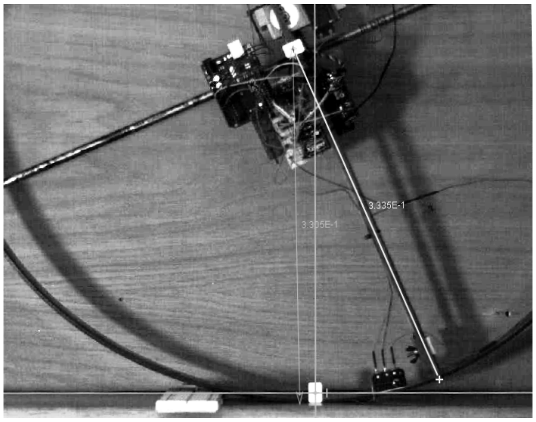


Figure 3.15: Image analysis software, through a pixel-to-actual-distance ratio, facilitates measurement of real parameters related to static bending of the robot.

Distance from the fiducial marker to the center of the touch segment is measured along the horizontal and vertical directions, giving x_d and y_d, respectively. This process is repeated for robot orientations of $-22.5°$, $-45°$, $-67.5°$, etc., up to and including $-360°$. (Figure 3.15 shows a screen shot from the image analysis software for one of these orientations.)

If the robot were to have a perfectly round outer surface in which no bending occurred, measured values of x_d and y_d would be equal to x_e and y_c given by Equations 3.1 and 3.2 for $a = b$, which are 0 and 0.3335 m, respectively. The differences, then, between these values and measured values of x_d and y_d are simply Δx and d for the case in which the outer surface is circular. Delta x and d are calculated in this manner for each orientation in the test. Results of the test are shown in Figs. 3.16 and 3.17, in which points for Δx and d are plotted versus the angular position parameter, $\psi = -rem(\theta, 360°)$.

Referring to the plotted points of d in Fig. 3.16, the vertical drooping pattern of the center of mass is shown to be approximately symmetric about $\psi = 180°$ and has a frequency of two cycles per revolution of the robot. This observed symmetry is a reasonable result, given the symmetric build of the outer surface about A. The idea of symmetric bending is further supported by the fact that average values of d between

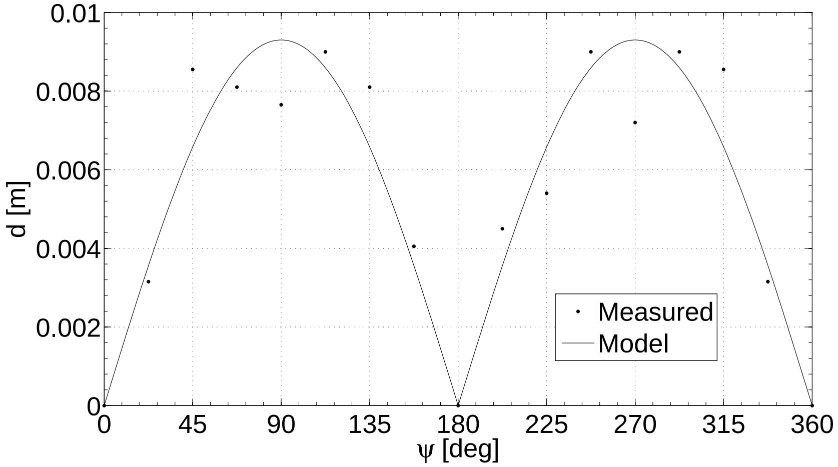


Figure 3.16: Measured values of d compared to model prediction.

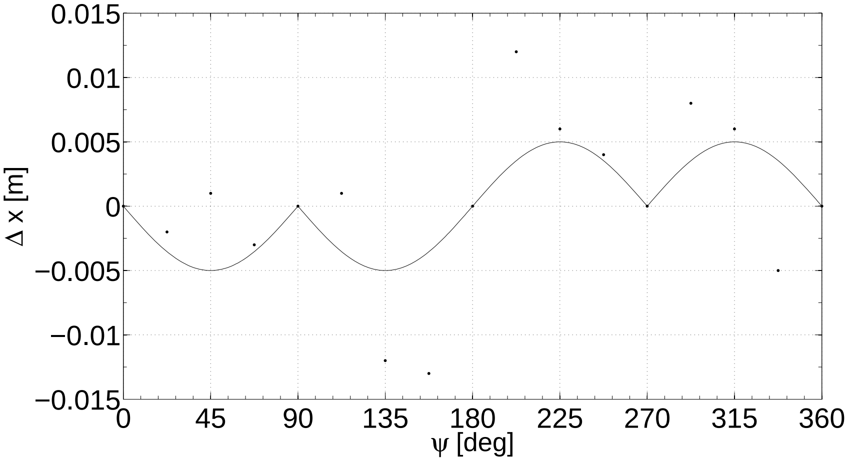


Figure 3.17: Measured values of Δx compared to model prediction.

0° and 180° and between 180° and 360°, measured to the nearest whole millimeter, are both 7 mm. In light of these results, d is modeled by the function, $\Lambda|sin\psi|$, where Λ is constant. This function is chosen because it is symmetric about 180°, and it has a plotted shape that's similar to the collection of points measured for d. In order to best-fit the function to the d data points, Λ is iteratively varied until the largest difference in magnitude between the function and measured values of d is minimized, resulting in $\Lambda = 0.0093$ and

$$d = 0.0093|sin\psi| \tag{3.5}$$

Equation 3.5 is plotted versus ψ on the same graph with measured values of d in Fig. 3.16. The largest magnitude of difference between ψ and measured values of d is 2.0 mm.

Referring to the plotted points in Fig. 3.17, Δx is mostly negative for $\psi < 180°$ and positive for $\psi > 180°$. At 90°, 180°, and 270°, Δx is always measured to be zero, and this result is due to construction of the robot that prohibits horizontal bending at these orientations of the robot. In general, the horizontal bending pattern causes the robot to be slightly clockwise torque-biased when rolling on one half of the outer surface, and slightly counterclockwise torque-biased on the other half. Perhaps this bias is due to the center of mass being located slightly to one side of A in the circular configuration, rather than precisely on A, or perhaps other construction imperfections cause the outer surface to bend unevenly over all. In any case, characterizing the Δx bending pattern is problematic due to difficulty in measuring the location of the touch segment midpoint along the horizontal; even when using magnified images from the digital camera to measure the midpoint location, is believed that individual measurements potentially have several millimeters of error. What is reassuring, however, is that overall bending is measured to be antisymmetric about $\psi = 180°$, at least on average, so that torque gains made on one half of the outer surface are lost on the other half. Specifically, the average value of Δx corresponding to $\psi < 180°$ is

-4 millimeters, and the average value of Δx corresponding to $\psi > 180°$ is 4 millimeters. For this reason, the function chosen to represent Δx, expressed in Equation 3.6 below, is antisymmetric; and it also has $\Delta x = 0$ at $\psi = 0°, 90°, 180°$ and $270°$, since Δx is measured as zero at these four orientations of the robot.

$$\Delta x = \begin{cases} 0.005|sin2\psi| & \text{if } 0 \leq \psi < 180° \\ -0.005|sin2\psi| & \text{if } 180 \leq \psi < 360° \end{cases} \tag{3.6}$$

The amplitude, 0.005, is chosen so as to minimize the sum of differences between the function and measured values of Δx over one full rotation of the robot. In Fig. 3.17, the function is plotted versus ψ, and measured values of Δx are included on the same graph.

3.5.2 Free Body Diagram

A full free body diagram for the rolling robot, with bending of the outer surface, is developed and shown in Fig. 3.18. The definition for roll angle of the robot remains the same as defined previously in this chapter. Now, however, the touchpoint position, S, is located at a horizontal distance, x_d, from the center of mass. As before, T and N act at the same location, S, on X. Vertical distance from X to the center of mass is now y_d. Touch angle, σ, is defined as the angle made by the A axis and the line from the center of mass to the touchpoint. Assuming elliptical shape of the outer surface, touch angle is related to roll angle by the following equation:

$$\sigma = \arctan(\frac{x_d}{y_d}) - \theta \tag{3.7}$$

Rolling resistance torque, τ_r, opposes roll motion of the robot and is further explained in Section 3.5.5.

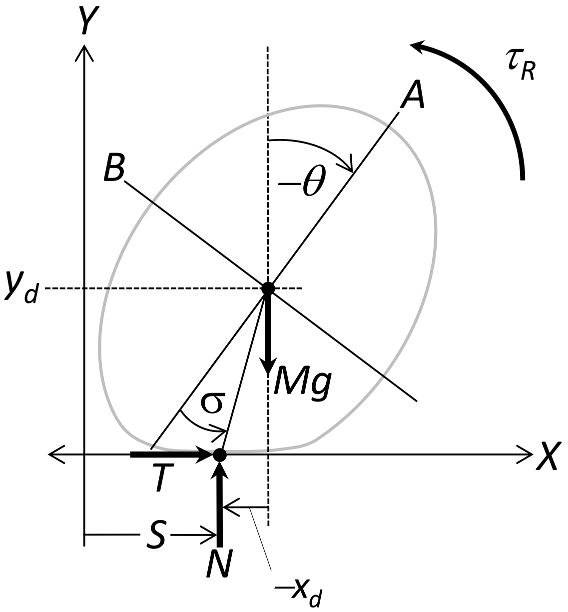


Figure 3.18: The dynamic model includes affects of bending and has three forces acting on the robot: weight, normal force and traction. The model also includes torque caused by rolling resistance.

3.5.3 Time-Derivative of Angular Momentum, $\dot{H}$

Angular momentum, H, is derived by modeling the robot as a collection of point masses that represent the entirety of its hardware components and the outer surface. In the Section 3.5, robot hardware components are modeled as ten point masses, with geometry shown in Fig. 3.13. Figure 3.19 shows the geometry of n point masses on AB representing the outer surface of the robot, indexed as $1 \le i \le n$. The points are assigned even angular spacing regardless of the shape of the surface – that is, polar angle of point mass i is expressed as

$$\alpha_i = 2\pi(i-1)/n \tag{3.8}$$

This approximation is valid as long as the outer surface is thin, retains a cylindrical shape (i.e., bending is minimal) and its eccentricity remains close to one, all of which

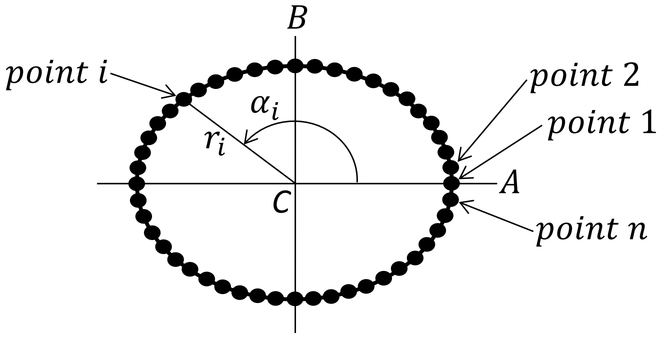


Figure 3.19: The outer surface is treated as a collection of n points in order to calculate angular momentum about the robot center of gravity at C.

are true for the rolling robot in question. Radius of point i is

$$r_i = \frac{ab}{\sqrt{a^2 \sin^2 \alpha_i + b^2 \cos^2 \alpha_i}} \tag{3.9}$$

and $\dot{r}_i$ is found by taking the time derivative. According to the previously mentioned assumption of constant radial spacing for a given point mass, terms that contain $\dot{\alpha}$ in the expression for $\dot{r}_i$ are set to zero, resulting in

$$\dot{r}_i = \frac{(a\dot{b} + b\dot{a})(ab/r_i)^2 - ab(a\dot{a}\sin^2 \alpha_i + b\dot{b}\cos^2 \alpha_i)}{(a^2 \sin^2 \alpha_i + b^2 \cos^2 \alpha_i)^{3/2}} \tag{3.10}$$

Time derivative of angular momentum for point mass i about the robot center of gravity is [42]:

$$\dot{H}_i = m_p r_i^2 \ddot{\theta} + 2 m_p r_i \dot{r}_i \dot{\theta} \tag{3.11}$$

where m_p is the mass of one point on the outer surface and is equal to $0.610/n$ kg. The time derivative of angular momentum for the entire outer surface, designated as $\dot{H}_1$, is calculated by summing contributions from n point masses:

$$\dot{H}_1 = \sum_{i=1}^{n} m_p r_i^2 \ddot{\theta} + \sum_{i=1}^{n} 2 m_p r_i \dot{r}_i \dot{\theta} \tag{3.12}$$

Time derivative of angular momentum for the six hardware point masses on A and

B in Fig. 3.13, designated as $\dot{H}_2$, is calculated in a similar manner, and is given by

$$\dot{H}_2 = (2m_{ip}(d_{ip}-a)^2+2m_{op}a^2+2m_{ad}b^2)\ddot{\theta}+(\dot{a}(4m_{ip}(a-d_{ip})+4m_{op}a)+4\dot{b}m_{ad}b)\dot{\theta} \quad (3.13)$$

Contribution of the two point masses representing the mechanical switches, designated as $\dot{H}_3$, is

$$\dot{H}_3 = \sum_{i=1}^{2} m_t[r(\alpha=\beta_i)]^2\ddot{\theta} + \sum_{i=1}^{2} 2m_t[r(\alpha=\beta_i)][\dot{r}(\alpha=\beta_i)]\dot{\theta} \quad (3.14)$$

where $m_t = 0.010$ kg is mass of one of the mechanical switches, and β_{1-2} are angular positions of the switches on AB and are equal to 75° and 355°. $\dot{H}_1$, $\dot{H}_2$ and $\dot{H}_3$ are added to give an expression for the time derivative of angular momentum for the whole robot:

$$\dot{H} = \dot{H}_1 + \dot{H}_2 + \dot{H}_3 \quad (3.15)$$

3.5.4 Equations of Motion

Referring to the free body diagram in Fig. 3.18, Newton's Second Law is applied in the horizontal and vertical directions and also about the center of gravity, resulting in three coupled differential equations of motion:

$$T = M\ddot{x}_c \quad (3.16)$$

$$N - Mg = M\ddot{y}_d \quad (3.17)$$

$$Ty_d + Nx_d + \tau_R = \dot{H} \quad (3.18)$$

where M is mass of the robot, and H is angular momentum of the robot about its center of gravity. Assuming no slipping occurs between the outer surface and the roll surface, the relationship between the robot center of mass position and arc length is

$$x_c = S - x_d \tag{3.19}$$

and $\ddot{x}_c$ is found by twice differentiating, giving

$$\ddot{x}_c = \ddot{S} - \ddot{x}_d \tag{3.20}$$

Touchpoint location, S, is equivalent to the elliptical arc length integrated over roll angle, plus a constant value [43]:

$$S = \int \sqrt{R^2 + (R')^2}\, d\sigma + C \tag{3.21}$$

with

$$R = \frac{ab}{\sqrt{a^2 \sin^2 \sigma + b^2 \cos^2 \sigma}} \tag{3.22}$$

$$R' = \frac{ab(b^2 - a^2) \sin \sigma \cos \sigma}{\sqrt{(a^2 \sin^2 \sigma + b^2 \cos^2 \sigma)^3}} \tag{3.23}$$

The first time derivative of S is found by differentiating Equation 3.21 using the "differentiation under the integral rule" [44], giving:

$$\dot{S} = \dot{\sigma} \sqrt{R^2 + (R')^2} \tag{3.24}$$

and $\dot{\sigma}$ is found by taking the time-derivative of Equation 3.7, giving:

$$\dot{\sigma} = \frac{1}{1 + (x_d/y_d)^2} \frac{y_d \dot{x}_d - x_d \dot{y}_d}{y_d^2} - \dot{\theta} \tag{3.25}$$

The second time-derivative of S in Equation 3.20 is found by numerically differentiating $\dot{S}$. Finally, Equations 3.16, 3.17, 3.18 and 3.20 are combined into one equation of motion:

$$\ddot{\theta} = \frac{1}{\rho_1}(-\rho_2\dot{\theta} + My_d\ddot{S} - My_d\ddot{x}_d + Mx_d\ddot{y}_d + Mx_dg + \tau_R) \qquad (3.26)$$

where ρ_1 and ρ_2 are coefficients of $\ddot{\theta}$ and $\dot{\theta}$, respectively, for the $\dot{H}$ term in Equation 3.15.

3.5.5 Rolling Resistance Torque

As part of an investigation into rolling resistance associated with automobiles [45], wheels with pneumatic road tires were pressed against a flat surface and spun with an applied torque. As the wheel rolled, the resultant normal force acting on the tire was measured and found to act a small distance "in front" of the wheel center, instead of directly underneath it. Given this velocity-induced shift of the normal force, the author of the investigation derived a mathematical model for a driven wheel with a pneumatic road tire when rolling on a flat, level surface. The model, illustrated in Fig. 3.20, shows the tire flattening where it makes contact with the roll surface, whereby the resultant normal force is displaced a small distance in front of the wheel center, causing a resistive moment that slows the spinning wheel.

It is observed that during trial rolls of the OSU rolling robot, it behaves like a pressurized tire in the sense that it flattens in the region of contact with the floor. For this reason, it is believed the front-shifting tendency of the normal load observed on spinning pneumatic tires also happens with the rolling robot outer surface, and this shared quality is deemed essential enough to permit a likening of the outer surface dynamics to the pneumatic tire model given in [45] and illustrated in Fig. 3.20. However, an important difference is that unlike a loaded pneumatic tire, the outer surface of the robot is not perpetually flattened as it rolls. For instance, consider the scenario in which the rolling robot is moving with a constant velocity, and its

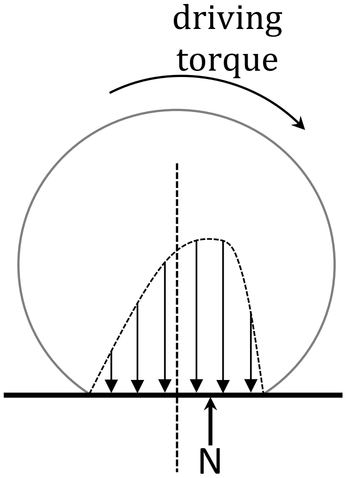


Figure 3.20: In the traditional rolling resistance model for road tires, the resultant normal force is located slightly in front of the wheel center, causing a moment that resists driven motion.

linear actuator is instantaneously oriented in the vertical orientation. At this moment, although the normal force might be very large in magnitude, rolling resistance is relatively small, because bending of the outer surface is restricted by the linear actuator, and the resultant normal force vector can therefore not be displaced. For this reason, a rolling resistance model is sought for the outer surface that maintains the notion of velocity dependence, but also bases rolling resistance on whether or not there is bending of the outer surface. When there is no bending of the outer surface (when $d = 0$), rolling resistance should be zero. In summary, it is neither bending nor angular velocity alone that brings about a resistive moment in the rolling robot; rather, it is the simultaneous presence of both.

With this idea in mind, rolling resistance torque of the outer surface is modeled as

$$\tau_R = k_R d \omega^2 \tag{3.27}$$

where k_R is a positive value that changes with ω. This model rightly predicts that $\tau_R = 0$ when $d = 0$ and when $\omega = 0$. Also, because d and ω^2 are never negative during trial rolls, the model rightly predicts that τ_R always acts in the clockwise direction, resisting rotation of the robot whenever it moves and bends. The power of two on angular velocity causes the model to capture sharp decelerations that are observed only when the robot moves with relatively high angular velocity. In order to determine appropriate values of k_R, three trial rolls of the robot are performed at three different steady state angular velocities. In the first of these rolls, the robot is controlled to reach -2.0 rad/s and continue, on average, at this velocity for eight seconds. While the roll is in progress, angular velocity of the rolling robot is sampled using onboard sensing and recorded as a function of time. Afterward, the simulation program described in Section 3.8 is run, using the same initial conditions as the trial roll. Values of k_R are iteratively used in the program until the simulated response best matches that of the actual robot. Matching is done by comparing total distance rolled and plots of angular velocity at steady state. The value of k_R used in the best match simulation is taken as the appropriate model value: 1.6 N/s^2. This process is repeated for -2.2 rad/s, and the appropriate value of k_R is found to be 2.1 N/s^2. For -2.4 rad/s, the appropriate value of k_R is found to be 1.9 N/s^2.

3.6 Rolling Robot Control System

The automatic control system for the rolling robot utilizes roll angle and angular velocity (θ and ω, respectively) as feedback variables. The goal of the control system is to cause the robot, starting from rest, to reach and maintain a prescribed step angular velocity. This goal is achieved through one controlled action: changing length of the linear actuator as the robot rolls, which in turn changes shape of the outer surface. When this shape change is strategically timed and executed, robot motion is repeatedly affected in a way that is similar to how an egg rolls when it's placed on end.

Shape change actuation is triggered when a branch of A or B leans into the roll and passes a certain inclination. When this happens, the robot control system causes the linear actuator to either extend or contract, and the shape of the robot is changed. Upon completion of the shape change actuation, the linear actuator remains at the newly changed length until actuation is newly triggered. For discussion purposes, three concepts associated with this control process are designated: the axis whose position triggers actuation is the trigger axis; the axis positioned perpendicularly from the trigger axis is the sheer axis; and moments when actuation is newly triggered are called trigger moments. Every quarter turn of the robot, axes A and B switch positions (sheer/trigger) at the trigger moment. The sheer axis is thus named due to its consistently upright position; when actuation comes to completion, the sheer axis is about to pass or has just passed through the sheer (vertical) orientation.

To help understand how shape change affects speed of the robot, consider the illustration in Fig. 3.21, in which the rolling robot is shown rolling to the right when shape change actuation is triggered. The control system responds by changing linear actuator length along A (the sheer axis in this case), and consequently, the outer surface eccentricity changes as the robot continues to roll to the right. Roughly a quarter-turn after actuation commences, two scenarios are considered in the figure. If a has been made long as A leans into the roll, the robot undergoes an induced torque imbalance about its center of mass that pushes the robot forward, as illustrated by the lower ellipse in Fig. 3.21. On the other hand, if a is changed so that ellipse eccentricity is equal to one, there is no resulting torque imbalance due to offset, and average speed is not increased.

Shape change actuation is triggered using the concept of tilt. Tilt angle of the robot is

$$\gamma = -rem(\theta, 90^\circ) \tag{3.28}$$

and can be thought of as the robot forward inclination measured by A or B relative to the vertical, as illustrated in Fig. 3.22. Tilt angle is used by the control system to perform two actions. First, a special measurement of angular velocity of the robot,

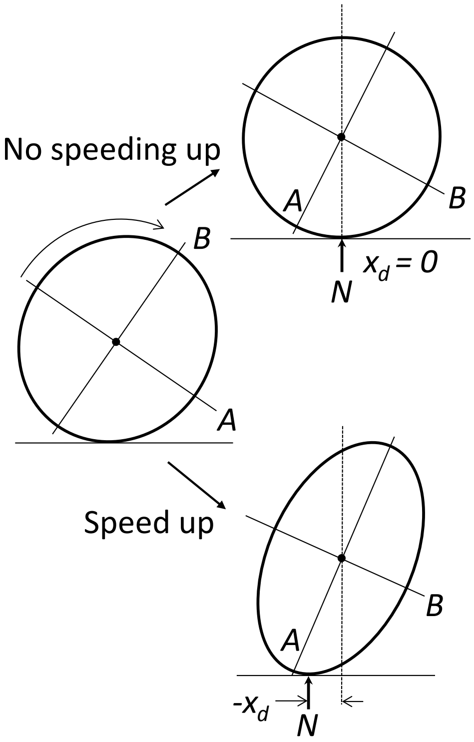


Figure 3.21: The moment, Nx_d, causes a torque imbalance about the robot center of mass, affecting roll dynamics of the robot.

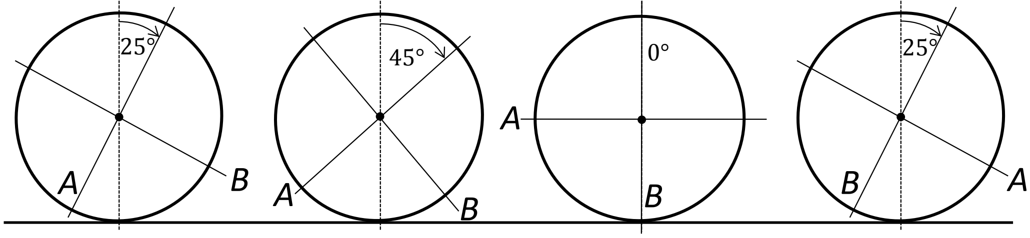


Figure 3.22: As the robot rolls to the right in this illustration, γ is always greater than or equal to 0° and less than 90°.

ω_m, is initiated when γ newly becomes greater than or equal to the set measurement angle, θ_m, which is usually 35° for the trial rolls conducted for the primary experiment documented in Section 4.6. The second action is actuation triggering, which can occur at the same time as measurement triggering but never before it. Triggering occurs when γ newly becomes greater than or equal to the set trigger angle, θ_t.

Upon actuation triggering, the microprocessor prepares for shape change by performing several computations to determine a target length of the linear actuator. The first of these computations is that of error:

$$Error = k_s \omega_d(t) - \omega_m(t) \tag{3.29}$$

where $\omega_d(t)$ is a desired velocity profile, and k_s is a positive constant, slightly larger than unity, that magnifies the desired velocity in order to compensate for the constant slowing of the robot due to rolling resistance torque (Section 3.5.5). The most noticeable control effect of k_s is to change the average value of angular velocity at steady-state. With error computed, it is in turn used to compute an intermediary value of the target length for the sheer semi-major axis, given by

$$l_t = R_c - k_f(Error) \tag{3.30}$$

where R_c is equal to 0.3335 m, the radius of the outer surface while in the circular configuration, and k_f is a positive control constant. If *Error* is negative (the robot is rolling too slowly to the right), the control system prepares to elongate the sheer semi-major axis relative to the circular configuration. If *Error* is positive (the robot is rolling too quickly to the right), k_f is temporarily set to zero until the next trigger moment. In the latter case, the robot assumes the circular configuration and is slowed by rolling resistance alone. Final target length of the sheer semi-major axis is computed according to a saturation operation that ensures the ellipse stays within the physical bounds of the system:

$$L_t = \begin{cases} s_{min} & \text{if } l_t \leq s_{min} \\ l_t & \text{if } s_{min} < l_t < s_{max} \\ s_{max} & \text{if } l_t \geq s_{max} \end{cases} \tag{3.31}$$

where s_{min} and s_{max} are the smallest and largest allowable values of the outer surface semi-major axis, and they are equal to 0.319 m and 0.349 m.

Since the linear actuator lies along A, if A is the sheer axis, the control system causes the linear actuator to elongate or shorten so that a is equal to L_t. This actuation is modeled as an isosceles trapezoidal velocity profile with upper edge length of $t_a/2$, where t_a is the time duration of the profile. Integrating the profile gives a as a function of roll time, t:

$$a(t) = \begin{cases} a_i + \frac{2V_m}{t_a}(t - t_i)^2 & \text{if } t_i < t \leq t_i + \frac{t_a}{4} \\ a_i + (t - t_i)V_m - \frac{t_a V_m}{8} & \text{if } \frac{t_a}{4} + t_i < t \leq t_i + \frac{3t_a}{4} \\ a_i - \frac{2V_m(t - t_i - \frac{3t_a}{4})^2}{t_a} - \frac{t_a V_m}{8} + V_m(t - t_i) & \text{if } \frac{3t_a}{4} + t_i < t \leq t_a \end{cases} \tag{3.32}$$

where t_i is roll time when actuation commences, a_i is the value of a when actuation commences, and $V_m = 4(L_t - a_i)/(3t_a)$. Actuation occurs only if $L_t - a_i$ is greater than 0.003 m in magnitude. Otherwise, the linear actuator remains as is, thereby avoiding moves in which a considerable percentage of the motor's motion is ineffectual due to hysteresis. If B is the sheer axis, the control system causes the linear actuator to elongate or shorten so that b is equal to L_t. In this case,

$$b(t) = \begin{cases} b_i + \frac{2V_m}{t_a}(t - t_i)^2 & \text{if } t_i < t \leq t_i + \frac{t_a}{4} \\ b_i + (t - t_i)V_m - \frac{t_a V_m}{8} & \text{if } \frac{t_a}{4} + t_i < t \leq t_i + \frac{3t_a}{4} \\ b_i - \frac{2V_m(t - t_i - \frac{3t_a}{4})^2}{t_a} - \frac{t_a V_m}{8} + V_m(t - t_i) & \text{if } \frac{3t_a}{4} + t_i < t \leq t_a \end{cases} \tag{3.33}$$

where b_i is the value of b when actuation commences, and $V_m = 4(L_t - b_i)/(3t_a)$. Again, actuation occurs only if $L_t - b_i$ is greater than 0.003 m in magnitude. Otherwise, the actuator remains as is. In order to move b to the target length, the control system must move a so that b becomes equal to L_t. Since no simple, exact formula exists that relates a, b, and ellipse circumference, the Ramanujan approximation [43] is applied to compute a as a function of time for actuation of A:

$$a(t) = -\frac{8b - 6P/\pi}{12} + \frac{1}{12}\sqrt{(8b - 6P/\pi)^2 - 24(P^2/\pi^2 - 6Pb/\pi + 6b^2)} \tag{3.34}$$

where P is ellipse circumference. The approximation is highly accurate for the range of eccentricity retained by the outer surface, so error associated with Equation 3.34 is negligible [46].

3.7 Characterization of the Linear Actuator

As described in Section 3.1, the linear actuator is a positioning device driven by Hitec's HS-645MG [28] servo motor, which generates sufficient torque to extend or shorten the actuator quickly and accurately to its commanded length by means of a rack/pinion transmission system. The servo motor has a positional range of motion of approximately 175° that translates to an approximate differential range on the actuator of 5 cm. In the present section, two aspects of the linear actuator are characterized: the relationship between commanded angular position of the servo motor and length of semi-major axis, a, after actuation is complete; and the time required for actuation.

3.7.1 Relationship Between Motor Position and a

A control program (Section 4.2) that's uploaded and run on the microprocessor generates a pulse width modulated signal that is sent to the servo motor top control its position. Upon receiving the signal, the servo motor's internal circuitry measures pulse width, which corresponds to a desired angular position of the servo motor [47], and equipped with a potentiometer that provides feedback of motor shaft position, a circuit inside the servo motor independently controls current flowing into the motor, thereby rotating the motor shaft until it arrives at the desired position. As long as pulse width of the signal does not change, the motor's position is held very close to the commanded position. In the control program, commanded position is written in units of microseconds.

A series of measurements tests are performed in order to establish a relationship between pulse width of the commanded position signal, denoted as the variable μ, and the value of a after commanded actuation is complete. The relationship is needed in order to know where to move the servo motor shaft once target position has been computed by the control system (Section 3.6). Before the tests are performed, the linear actuator is calibrated so that $a = 0.3335$ m when the servo motor is permanently set at the commanded position of $\mu = 1470$ microseconds. These values correspond to the circular configuration of the outer surface and the central position of the servo, respectively. After calibration, the outer surface of the robot – with its actuator in the vertical orientation – is clamped to a sturdy lab table and left unhandled. A program is run on the robot microprocessor in which the servo motor shaft is commanded to rotate to its positional limit in the counterclockwise direction ($\mu = 570$ microseconds) and hold. At this point, the elliptical diameter along A, which is equal to $2a$, is measured using a ruler, and the motor is commanded to go back to its central position. In this manner, measurements are obtained for several other angular positions, including the positional limit in the clockwise direction ($\mu = 2400$ microseconds) and six other, in-between values of μ.

After the eight data points have been recorded, the robot is unclamped from the lab table, and with the actuator in the horizontal orientation, the robot is reclamped.

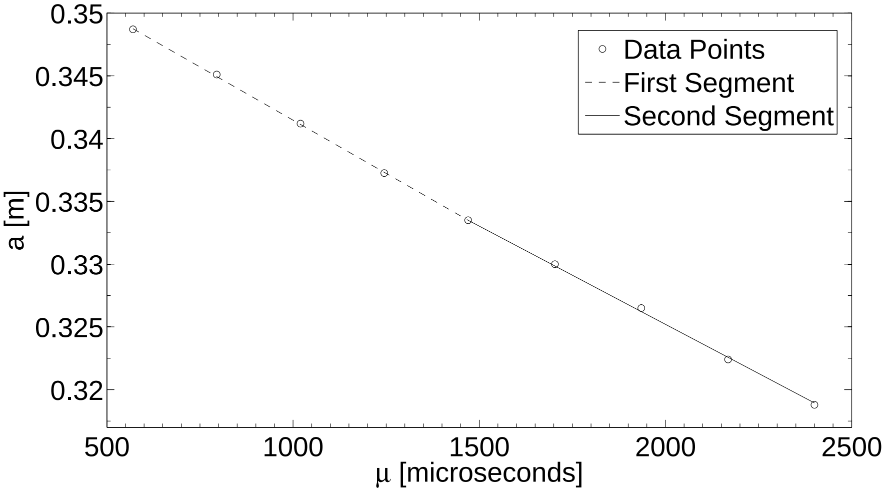


Figure 3.23: For shape change actuation, length of semi-major axis, a, is measured and plotted for nine values of servo motor angular position, μ. Shown in the graph is the piecewise linear function developed to model the relationship between a and μ .

The test is then repeated for the same eight values of μ. The process is repeated yet again with the robot clamped in the $\theta = 45°$ orientation. Thus, for each value of μ, three measurement sets of outer surface diameter, $2a$, are obtained. An average of these measurements is calculated, and then divided by two, resulting in one set of eight data points of the form, (μ, a). The calibrated central point, $(1470, 0.3335)$, is added to the set for a total of nine data points. These nine points are plotted as open circles in Fig. 3.23.

Using the method of least squares [48], a best fit line function is determined for the nine data points. The function is piecewise linear [49] with two segments: the first segment fits points where μ is less than or equal to 1470 microseconds, and the second segment fits points where μ is greater than or equal to 1470 microseconds. By design, both segments contain the point, $(0.3335, 1470)$, which is the calibrated central point, and this design causes the segmented function to be highly accurate when the outer surface is circular or nearly circular, such as when the robot is moving with controlled, steady state velocity. The best fit line function is plotted on the same graph as the nine data points in Fig. 3.23. The first segment of the function is plotted as a dashed line, and the second segment is plotted as a solid line. From the graph, it is evident that the relationship between μ and a is highly linear. In fact, the best fit line function predicts the nine values of a with error no greater than one-third of a millimeter. The function is expressed as:

$$a = \begin{cases} \frac{\mu - 1470}{63900} + 0.3335 & \text{if } 570 \geq \mu \leq 1470 \\ \frac{\mu - 1470}{59000} + 0.3335 & \text{if } 1470 > \mu \leq 2400 \end{cases} \tag{3.35}$$

3.7.2 Actuation Time Duration, t_a

During initial roll trials of the robot, it is observed that t_a is dependent on the change in a ($\Delta a = a_f - a_i$) for the actuation, and on orientation of the actuator axis, A, with respect to gravity when actuation occurs. In order to confirm these observations and quantify the effects of actuation length and orientation on t_a, a test is performed in which the outer surface is clamped to a sturdy horizontal surface with A in the horizontal orientation, as shown in Fig. 3.24. A program running on the microprocessor directs the servo motor to move to the central position ($\mu = 1470$ m), hold, and then sweep back and forth in the style of a windshield wiper. Specifically, the program commands the motor to move to the following angular positions, starting from the central position, and briefly hold after each move: 1680 ms, 1260 ms, 1943 ms, 998 ms, 2378 ms and 591 ms. As soon as the sweeping profile begins, a light emitting diode (LED) turns on. Fiducial markers are attached to the outer surface to identify the extent of the outer surface diameter along A during motion.

A high speed digital video camera is situated with its optical axis parallel to and roughly centered on the cylindrical axis of the outer surface, and a few seconds before the servo motor begins the sweeping profile, the video camera starts recording the motion at 300 frames per second and stops recording after the servo motor shaft has settled-in to the 591 ms shaft position. The recorded movie is subsequently

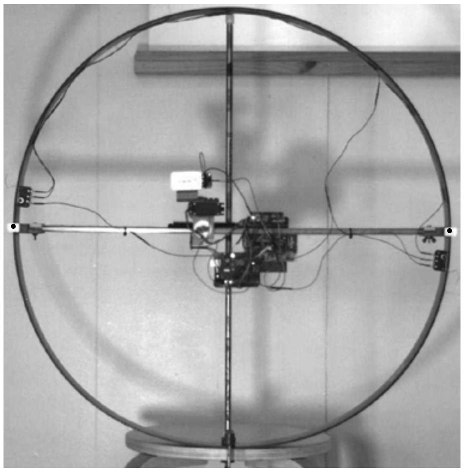


Figure 3.24: In a test to characterize motion of the actuator, the robot is clamped to a horizontal surface, and a video camera records motion. Image analysis software facilitates measurements of a by means of two fiducial markers.

uploaded to image analysis software [41] and clipped, so the frame that first shows the illuminated LED is the initial frame at which profile run time = 0 s. In the image analysis software, a consistent Cartesian coordinate system is assigned to the movie frames, and distance on the frames is calibrated using a pixel-to-actual-distance ratio.

For each frame in the movie, the image analysis software automatically locks-in on the fiducial markers and computes the distance between them, which is $2a$. The software then divides the distance by two and builds a table with a as a function of run time in 1/300 s intervals. This tabulation process is repeated two more times in the exact same manner, resulting in three values of a for every 1/300 s interval of run time. For the purpose of smoothing, the average value of a is taken as the actual value and is plotted versus run time, as shown in Fig. 3.25. Time duration required for each move of the servo motor shaft (from hold to hold) is graphically derived from the plot. For instance, referring to the plot in Fig. 3.25, time duration for the second move in which a changes from 0.330 m to 0.337 m, takes approximately 0.26 s. This time duration is indicated on the plot. Graphical measurement of time duration is repeated in this manner for the remaining moves in the sweeping profile, of which there are a total of six.

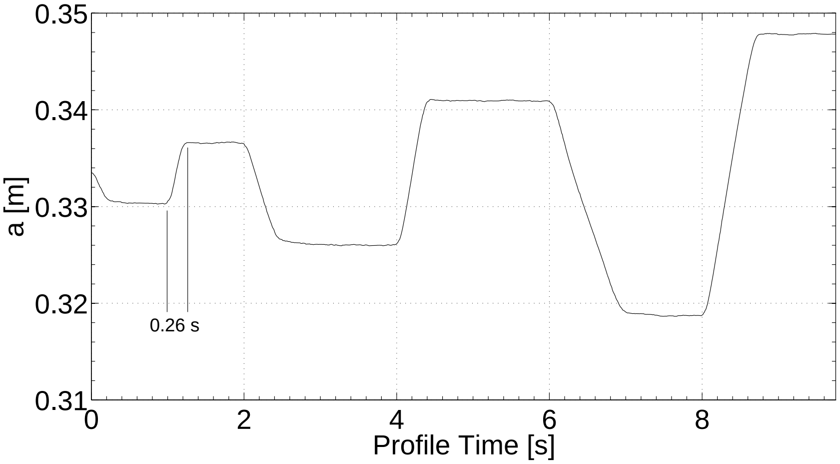


Figure 3.25: Measured values of a plotted versus actuation time while A is in the horizontal orientation. Time durations of various linear actuator moves are established using the plot, as signified for a 0.26 s move.

Table 3.1: Actuation Time Duration

	Vertical	Diagonal	Horizontal
$\Delta a = 0$	0 s	0 s	0 s
$\Delta a = -0.0033$ m	0.22 s	0.25 s	0.28 s
$\Delta a = 0.0068$ m	0.26 s	0.27 s	0.30 s
$\Delta a = -0.0109$ m	0.37 s	0.40 s	0.59 s
$\Delta a = 0.0154$ m	0.60 s	0.53 s	0.52 s
$\Delta a = -0.0219$ m	0.62 s	0.65 s	1.15 s
$\Delta a = 0.0290$ m	1.12 s	0.92 s	0.86 s

The process previously described of acquiring three measurement sets of a, averaging them, and plotting to obtain t_a for each Δa in the sweeping profile is repeated for the robot clamped with the actuator in the vertical orientation and then again with the actuator in the diagonal orientation corresponding to $\theta = 45°$. For the diagonal orientation, the outer surface is held rather than clamped in order to avoid vibration-induced stress on the end-joints. As a summary of the data collected for these three tests, Tab. 3.1 provides t_a for the various changes in a for three orientations of the linear actuator. Added to the table in each column is the obvious value of $t_a = 0$ for $\Delta a = 0$. Positive values of Δa in the table signify extension of the linear actuator, and negative values signify contraction.

It is observed from Tab. 3.1 that contractions of the actuator while in the vertical orientation take the least amount of time relative to other actuations. This result is expected, since contraction of the actuator while in the vertical orientation amounts to moving half the robot mass downward in the direction of gravity. Contraction of the actuator while the in the horizontal orientation is consistently the most time consuming of all moves, even compared to large extensions while in the vertical orientation. This seemingly odd result can be explained, in part, by the existence of bending of the telescoping columns of the linear actuator, which creates significant friction between the male and female tubes during actuation, and which is greatest when orientation of the actuator is horizontal. In addition, when the actuator is in the horizontal orientation, outer surface bending puts considerable outward force on the linear actuator, and this force makes contraction surprisingly difficult. For these reasons, contraction of the linear actuator, starting from full extension and while in the horizontal orientation, is the most burdensome and time consuming of all moves in Tab. 3.1.

Recall from Section 3.6 that actuation of the robot is triggered when axis A or B leans into the roll past a certain inclination corresponding to θ_t. For the primary experiment documented in Chapter 4, the value of θ_t is changed for each trial roll, but it is bounded by: $25^\circ \leq \theta_t \leq 65^\circ$. During controlled locomotion of the robot, therefore, actuation begins while the actuator is in-between the vertical and horizontal orientations, and so it is concluded that data points for the "Diagonal" column in Tab. 3.1 are a reasonable representation of the actuator's motion in general. The seven data points from this column are plotted as open circles in Fig. 3.26, and using the method of least squares [48], a best fit function is determined for them. The best fit function is piecewise and is comprised of two second order polynomials: the first polynomial fits points where Δa is less than or equal to 0 m, and the second polynomial fits points where Δa is greater than or equal to 0 m. The expression for

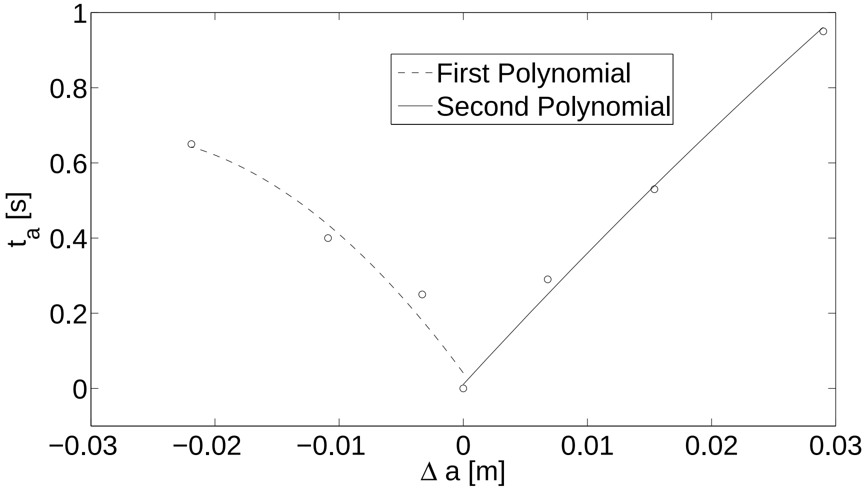


Figure 3.26: A best fit polynomial function with two segments is fitted to measured data (circles) and used to model the relationship between actuation time and change in length of the semi-major axis.

the best fit function is:

$$t_a = \begin{cases} -792.34(\Delta a)^2 - 44.82(\Delta a) + 0.0412 & \text{if } -0.0219 \leq \Delta a \leq 0 \\ -113.37(\Delta a)^2 + 36.05(\Delta a) + 0.0106 & \text{if } 0 < \Delta a \leq 0.0290 \end{cases} \tag{3.36}$$

Using Equation 3.36, t_a is plotted versus Δa in Fig. 3.26, along with the seven data points from the "Diagonal" column in Tab. 3.1.

Equation 3.32, which gives a as a function of roll time, t, is based on the assumption of a trapezoidal velocity profile for $\dot{a}$. According to this equation and also Equation 3.34, a is dependent on t_a, so now that an expression for t_a has been derived and given by Equation 3.36, t_a can be used in Equation 3.32 and in Equation 3.34 to predict a as a function of time, given Δa. With this in mind, the program that directs sweeping motion of the motor is amended to compute $\Delta a = a_f - a_i$ for each move of the sweep. In turn, Δa is used in the program to compute the value of t_a using Equation 3.36 and ultimately a at each cycle of the program. The amended program is then run, and computed values of a are written to a text file every 10 ms. Using this data, three comparison graphs are constructed. Each graph has computed values

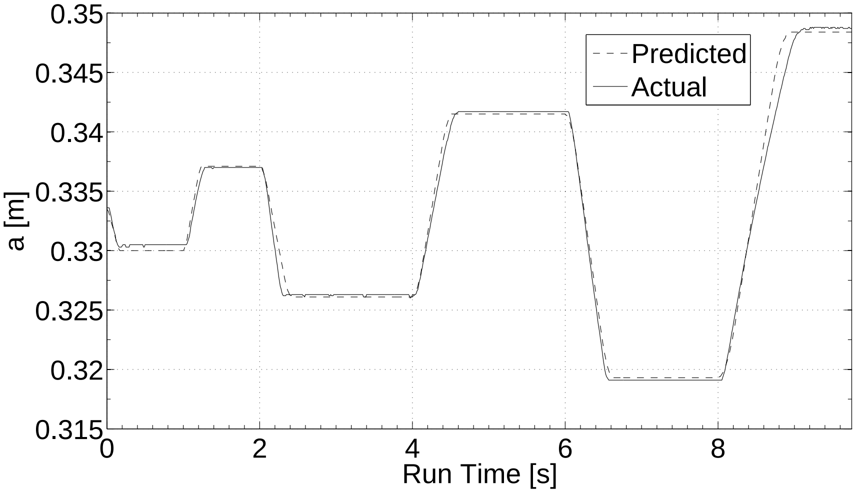


Figure 3.27: Predicted values of a, along with actual values of a measured during the vertical orientation test, are plotted versus profile run time.

of a, along with actual values of a measured previously during one of the orientation tests, plotted versus profile run time. The plots are shown in Figs. 3.27, 3.28 and 3.29, corresponding to the vertical, diagonal and horizontal orientations of the actuator, respectively.

In comparing predicted values of a to actual values of a in Figs. 3.27, 3.28 and 3.29, it is evident that predicted values of a more closely match actual values of a when motion of the actuator is along the vertical and diagonal orientations. For the several hundred values of a output by the program during the entire sweeping profile, standard deviation of the magnitude of difference between predicted and actual values of a are: 0.6 mm, 0.3 mm and 1.5 mm for the vertical, diagonal and horizontal orientations, respectively. The model, therefore, is very accurate on average, especially for actuations in the vertical and diagonal orientations. Because computed values of a are derived from data collected while the actuator is in the diagonal orientation, these result are not surprising. In addition, because friction in the telescoping columns is highest when the actuator moves along the horizontal orientation, one would expect motion therein to be relatively capricious and thus more divergent from the predicted curve than actuation along the vertical.

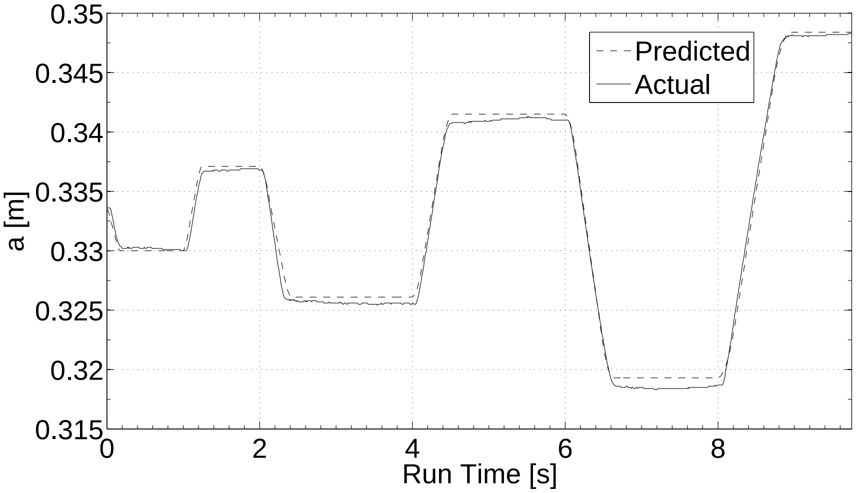


Figure 3.28: Predicted values of a, along with actual values of a measured during the diagonal orientation test, are plotted versus profile run time.

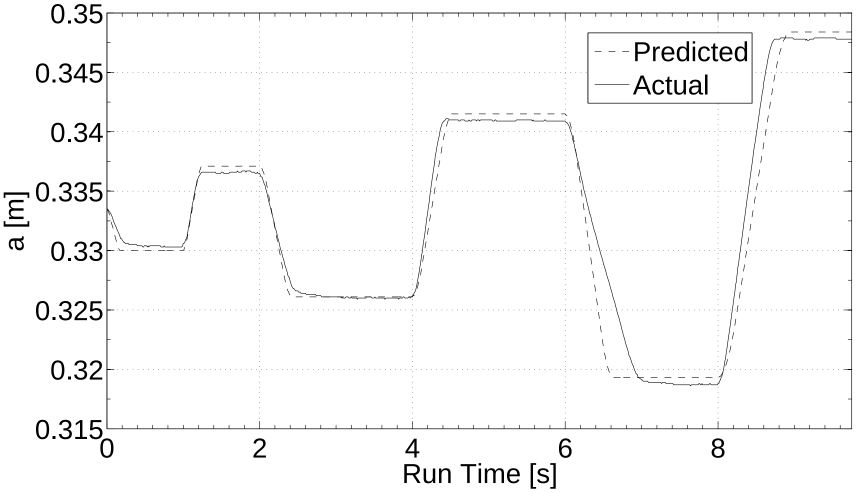


Figure 3.29: Predicted values of a, along with actual values of a measured during the horizontal orientation test, are plotted versus profile run time.

Referring again to the plots in Figs. 3.27 and 3.28, and considering only smaller moves with $\Delta a \leq 2.19$ cm, predicted values for a are significantly accurate for the vertical and diagonal orientations. In fact, for small and medium moves in the profile, difference between measured and predicted values of a for the vertical and diagonal orientations is always less than 2.6 mm in magnitude. When including larger moves in the profile, the difference between measured and predicted values of a for the vertical and diagonal orientations is always less than 3.6 mm in magnitude.

3.8 Simulation of Robot Locomotion

A Matlab/Simulink computer program is developed to solve Equation 3.26 numerically for θ as a function of roll time. The equation is repeated here for referencing convenience:

$$\ddot{\theta} = \frac{1}{\rho_1}(-\rho_2\dot{\theta} + My_c\ddot{S} - My_c\ddot{x}_d + Mx_d\ddot{y}_c + Mx_d g + \tau_R) \tag{3.37}$$

The program code, given in Listing 2 of the Appendix, is a looping structure that uses initial conditions, such as $\theta(0)$ and $a(0)$, to solve Equation 3.37 for $\ddot{\theta}$ at the first pass of the loop. After this first pass, the program "loops back" values that are needed in order to compute $\ddot{\theta}$. As part of the this process, the program checks γ at each pass to determine if actuation has been triggered, and if it has, a, b, $\dot{a}$ and $\dot{b}$ are changed according to the motion profiles given in Section 3.6; otherwise, a and b do not change. In either case, updated values of a, b, $\dot{a}$, $\dot{b}$, ρ_1, and ρ_2 are looped back, along with values for θ and $\dot{\theta}$ obtained through numerical integration, and $\ddot{\theta}$ is newly computed. Due to difficulty in finding exact expressions for some of the derivatives on the right side of Equation 3.37, a numerical derivative algorithm is employed to find the following values: $\dot{y}_c$, $\ddot{y}_c$, $\dot{x}_d$, $\ddot{x}_d$, and $\ddot{S}$. Once $\ddot{\theta}$ is newly computed, it is integrated yet again, and the process is repeated until the simulated roll is completed. At each

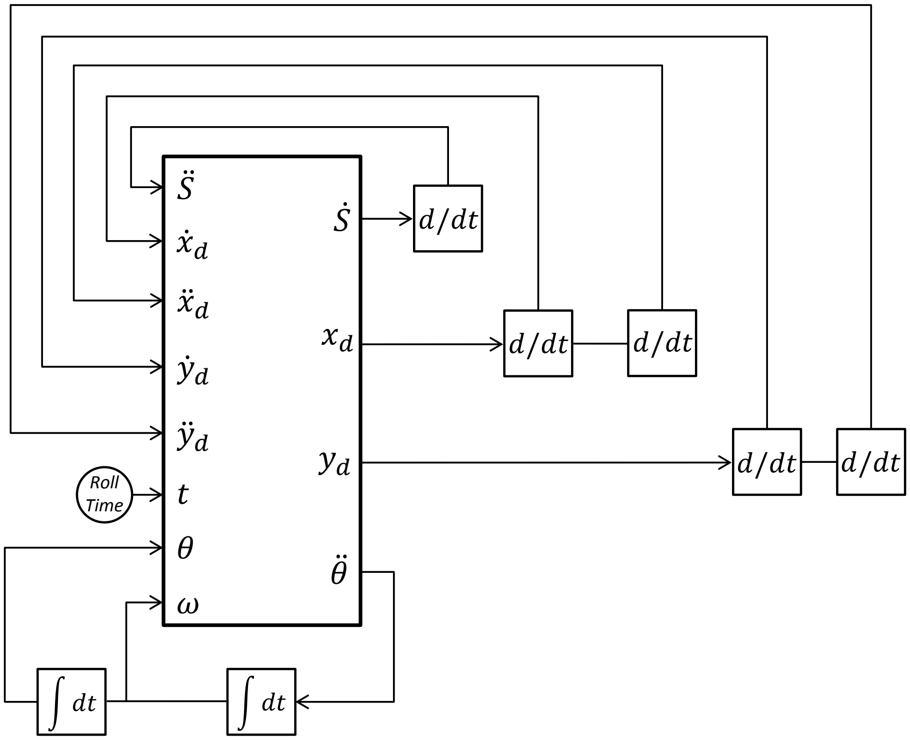


Figure 3.30: Graphical representation of the program looping structure illustrates how values on the right side of Equation 3.26 are "looped back" in order to compute $\ddot{\theta}$, which is then twice numerically integrated.

pass of the loop, motion parameters of the robot, including $\dot{\theta}$, are logged and can be plotted as a function of roll time after the program has terminated.

A graphical representation of the program's looping structure is shown in Fig. 3.30. The representation is limited to show only variables explicitly expressed in Equation 3.37. At the bottom of the figure, two boxes marked with "$\int dt$" represent numerical integration of $\ddot{\theta}$, which the program performs using Simulink's Runge-Kutta solver. Boxes with "d/dt" represent numerical derivatives. The main step in the program, in which the right side of Equation 3.37 is refreshed with updated values to compute $\ddot{\theta}$, is represented by the large central block in Fig. 3.30. A fixed step size of 0.01 s is used in the program to increment roll time, and $n = 40$ for the number of points representing the outer surface.

CHAPTER 4

Experimental Methodology

This chapter describes the preparation and procedure for the primary experiment, in which laboratory measurements are used to compute energy consumed per unit roll distance of the robot for various combinations of control parameters. The chapter is divided into six sections. The first section describes the specially designed track on which the robot rolls when the experiment is conducted. The second section describes the computer program run onboard the microprocessor that contains the control parameters that are varied during the course of the experiment. In the third section, a test is described in which the strategy for computing angular position of the robot is evaluated for accuracy. In the fourth section, it is shown that stable velocity control is achieved by the robot for a range of control parameters. Effects of outer surface vibration on measurement of velocity are investigated in the fifth section. Finally, the sixth section describes the primary experiment.

4.1 Roll Track

Trial rolls of the experiment are conducted in Oklahoma State University's Micro Aviary Laboratory, a basement room with a concrete floor that measures approximately 40 by 35 feet. The Micro Aviary room is an adequate place for experimentation with the rolling robot, because it is large enough and has sufficient open space to perform trial rolls of the rolling robot that are at least ten seconds in duration. Two drains on the floor of the Aviary cause it to be noticeably unlevel at a few spots, but everywhere else the floor is seemingly level, especially next to the walls away from the drains. Levelness, as used here, refers to how closely the floor matches the horizontal plane [50]; a marble placed at rest on a sufficiently level, glass pane floor would not freely roll to one side.

The Aviary was not designed for office space or a classroom, so the floor has not been highly finished and thus contains surface imperfections that make it somewhat rough and bumpy. Therefore, a rectangular section of laminate flooring is installed on the concrete floor as a smooth roll track for the robot. Most of the imperfections on the Aviary floor are rendered unobtrusive by a thin foam underlayment laid between the track and the laminate flooring flooring. Individual planks measuring 1.8 cm by 121.2 cm by 20.5 cm compose the flooring of the track, which is pictured under the robot in Fig. 4.1. As seen in the picture, the track is laid against a wall in the Aviary room where the concrete subfloor is apparently most level. The track is four planks wide by eight planks long, measuring 0.82 m by 9.70 m overall. Due to manufacturing imperfections of the flooring, the planks settle at slightly different elevations at the joints, resulting in several plateau-like bumps on the flooring that run across the width of the track. These bumps are tiny, and do not have a noticeable effect on the rolling robot. Wood simulation decals are factory appliquéd as the top layer on each plank of the track flooring, and the decals are not raised or embossed with faux grain texture, so the individual planks are very flat. Flatness here refers to smoothness; if one were to run his hand over a large swath of a very flat floor, he would feel neither small bumps nor a wave-like pattern of crests and troughs [50].

A test that characterizes flatness and levelness of the roll track is necessary to ensure that it is not, for example, moderately sloped, a quality that would affect results of the primary experiment and would also be difficult to confirm using only naked eye estimation. The measurement test used herein for this purpose is the F-Number system that's practiced by concrete flooring contractors who test for levelness and flatness of industrial floor slabs [50]. The system is a statistical method, typically

Figure 4.1: Roll track for the robot is a rectangular strip of laminate flooring, measuring 0.82 m by 9.70 m, that's installed on concrete.

carried-out using a special computerized machine that's rolled about the floor in a specific pattern while collecting measurements that are relevant to levelness and flatness. For the research herein, measurements are taken by hand with a ruler in a manner that emulates the measurement process performed by the machine. Since the ruler method diverges from the typical method, and also because the laminate flooring is not concrete, the method provides only an approximation of the flooring's character.

The F-Number system is based on two evaluation parameters. The first parameter communicates levelness, and it is calculated using the following formula [50]:

$$F_L = \frac{12.5}{3S_z + |z|} \tag{4.1}$$

where z is the mean of several measurements of difference in elevation between points on the flooring in question, and S_z is the standard deviation of the measurements. To calculate F_L for the roll track, a digital level with a laser pointer is used to establish a horizontal laser datum, located not more than six inches above the surface of the track. Elevation at any point on the track is measured with a ruler by placing the ruler vertically, with its edge on the track, and noticing where the laser light strikes the ruler. In this manner, elevation of the track is initially measured at two points: at the edge/center of the track, and ten feet down the center line of the track. Difference in elevation is recorded. The two measurements are taken again, this time after moving the ruler exactly one foot down the track relative to the two previous positions, and difference in elevation is again recorded. The process is repeated for a total of 21 measurements of difference in elevation. Absolute value of the mean of the 21 measurements, $|z|$, is calculated as 0.00150 m, and standard deviation of the 21 measurements, S_z, is 0.0026 m. Plugging these values into Equation 4.1, F_L for the track along its center line is calculated as 13.3. To give an idea of what this number means, levelness of this kind is what one might measure in the concrete subfloor of a college dorm, but it is generally not level enough for a warehouse floor [51].

The second F-Number communicates flatness, and it is calculated using the following formula [50]:

$$F_F = \frac{4.57}{3S_q + |q|} \tag{4.2}$$

where q is the mean of several measurements related to slope of the flooring in question, and S_q is the standard deviation of the measurements. To calculate F_F for the track, a one-foot digital level with a tilt angle display is used. It is modified to have rests on each end of the level that are spaced exactly 12 inches from one another. The modified level is calibrated by iteratively sanding the rests so that when placed consistently in the same test spot, the level displays the exact same tilt angle with the rests as without them. The modified level is then placed on the track and oriented so the level's length is parallel to the center line of the track with one rest at the edge of the track. Displayed tilt angle of the digital level is recorded. The level is then moved one foot down the center line, and the measurement is recorded again. This process is repeated for a total of 30 measurements of tilt angle, corresponding to 30 locations on the track spaced 12 inches apart. For each location, tilt angle is converted to tilt "rise" by multiplying tangent of the tilt angle by 12 inches. Absolute value of the mean of the 30 rise values, $|q|$, is calculated as 0.00150 m, and standard deviation of the 30 values is 0.0026 m. Plugging these values into Equation 4.2, F_F for the track along its center line is calculated as 35.4. To give an idea of what this number means, flatness of this kind is what one might measure in a warehouse floor, but it is generally not flat enough for a roller rink or gymnasium floor [51].

4.2 Control Program

The primary experiment is composed of a series of trial rolls, in which parameters that affect locomotion energy economy of the robot are varied while energy consumed by the servo motor is measured. During these trial rolls, the robot moves under control of a program (Listing 3 in the Appendix) that is run onboard the microprocessor. The control program is written in C++ programming language and is developed in the Arduino software environment, where the code is compiled and uploaded to the microprocessor as a set of readable instructions. By turning on power and then pushing a reset button on the microprocessor board, the control program is executed. Four primary tasks are performed by the microprocessor when the control program is executed: interface sensors, compute various parameters related to the robot motion, implement the shape changing strategy described in Section 3.6, and output parameters of interest. Each of these tasks is detailed in the present section.

As with any C++ program, the robot control program is essentially an invocation of one primary function [52]. In the case of the control program, that function is named `loop`, and it repeats every 10 ms. Prior to `loop`, the control program has two precursor sections of code that each run only once when the program is executed. The first is a variable definition section, where several variables are set and, if need be, changed by the user to form a combination of control values and initial conditions that define the trial roll. The variables that can be set are: θ_m, θ_t, k_f, k_s, ω_d and $a(0)$. After the variable definition section, a second precursor section of the program performs various housekeeping tasks, including initiation of microprocessor ports, sensors, communication lines, and moving the actuator to the set value of $a(0)$. After the precursor sections have run, the `loop` function is finally invoked, but before `loop` is executed for the first time, the control program sends the power sensor a digital high voltage signal, causing the sensor to begin the process of measuring current and voltage into the servo motor to compute energy consumed. Roll time of a trial roll, t, begins when this signal is sent, and an LED on the microprocessor board is illuminated to indicate that $t > 0$.

Recall from Section 3.4.2 that position switches are placed on the outer surface in a manner such that voltage in the switch circuit goes low when one of the position switches is activated, and activation occurs whenever A is in the vertical position, i.e., $-180°$, $-360°$, $-540°$, etc. In the `loop` function, the control program checks the digital line connected to the switch circuit to see if voltage has gone low since the last pass of `loop`. If it has, a flag is momentarily turned on in the program, and a tally, n_t, is kept of how many times the flag has been turned on since the trial roll began. If `loop` sees the flag presently on, then roll angle of the robot is computed from the flag tally:

$$\theta = -180 n_t \tag{4.3}$$

and this equation assumes robot initial orientation is between 0 and $-180°$, which is always the case for all trial rolls performed in the primary experiment. By way of example, if the robot were to commence from $\theta = -15°$ and roll until the flag variable turns on in `loop` with $n_t = 3$, then θ would be computed as $-540°$ at that pass of `loop`.

If `loop` sees the flag presently off, θ is computed in the sensor program by numerically integrating the sampled angular velocity, according to the trapezoidal method [53], starting from the last angle measured by the switch circuit. For instance, if the robot were rolling with $n_t = 3$, and the flag turned on at the last pass of `loop`, then θ

would be computed as: $0.010(\omega+\omega_{last})/2-540^{\circ}$, where ω is robot angular velocity as measured by the gyroscope at the current pass of `loop`, ω_{last} is robot angular velocity as measured by the gyroscope at the last pass of `loop`, and 0.010 is the amount of time (s) in-between measurements, equal to the sample period of the `loop` function.

With θ computed at every pass of `loop`, the control program uses θ to compute γ with Equation 3.28 and checks for triggering of measurement and actuation (Section 3.6). If measurement has been triggered, the current reading of the gyroscope is saved as ω_m. If actuation has been triggered, the control program computes $Error$ and target length, L_t, and initiates shape change actuation of the robot as described in Section 3.6. Regardless of triggering, at every pass of `loop` the program computes the predicted value of a using Equation 3.32 or Equation 3.34. Also, starting at the first pass of `loop` at which $\omega \geq \omega_d$, the control program computes a running average of measured angular velocity values:

$$\omega_r = \frac{\omega_1 + \omega_2 + \cdots + \omega_n}{n} \tag{4.4}$$

where $\omega_1 + \omega_2 + \cdots + \omega_n$ is the sequence of n values of angular velocity measured consecutively in `loop` from the initial measurement, ω_1, up to the current measurement, ω_n. This parameter is used to determine stability, as described in Section 4.4.

Roll distance, S, is also computed at every pass of `loop` using the approximation

$$S \approx R_c(\pi + \theta_0 - \theta - n_t\pi) + \frac{P}{2}(n_t - 1) \tag{4.5}$$

where R_c is equal to 0.3335 m, radius of the outer surface in the circular configuration; $\theta_0 = \theta(0)$; and P is perimeter of the outer surface, 2.095 m. Equation 4.5 is based on the idea that touch position can be approximated by treating the outer surface of the robot as a rolling, circular disc that doesn't slip against the roll track. The equation incurs error, because the outer surface of the robot often slips, bends, and usually retains a non-circular shape. Nevertheless, the magnitude of error incurred by Equation 4.5 is negligible, largely for two reasons. The first reason is slippage of the outer surface, quantified by an experimental test in Section 4.6.4, is found to be minimal. The second reason is that Equation 4.5 is based on n_t, an integer that updates twice per revolution, that momentarily forces Equation 4.5 to be highly accurate regardless of bending or eccentricity of the outer surface; and in-between updates, there is simply not enough time for error to accumulate to a significant magnitude.

The rolling robot possesses a radio transmitter, by which the microprocessor communicates with a receiver-equipped laptop computer located nearby. Every 100 ms (or every tenth pass of `loop`), the control program transmits several locomotion-related parameters to the laptop, where the parameters are printed as a line of comma separated values to a serial monitor window. The transmitted/printed parameters are: θ, ω, S, a, and ω_r. When run time goes beyond 15 seconds, the control program stops sending the parameters, electronically detaches the servo motor, and commands the power sensor to send elements of its total energy array in a consecutive manner, starting at the first element for $t = 0$ s and continuing to the 151^{st} element for $t = 15$ s. After all elements have been received, the control program sends them via radio transmission to the laptop serial monitor window, at which point the microprocessor enters an idle state until it is manually turned off when the robot is retrieved.

4.3 Accuracy of Onboard Angular Position Computation

A test is performed in order to learn the accuracy of the computation strategy (described in Section 4.2), by which angular position of the robot is computed through a combination of switch activation updates and integration of angular rate as measured by the onboard gyroscope. For the test, the control program is modified, so that in addition to updating θ using Equation 4.3 when a switch activation occurs, another value is computed: θ', which is what θ would be if it were updated at the previous switch activation but not at the present one. For example, say a trial roll of the robot is performed using the modified control program, and at a certain pass of `loop`, voltage is low on the switch circuit, meaning one of the position switches has activated, causing n_t to augment from 0 to 1. As such, the modified program uses Equation 4.3 to update θ and θ' to $-180°$, and at subsequent passes of `loop`, the modified control program calculates θ' according to the strategy described in Section 4.2 – that is, by integrating angular velocity over time and adding the result to $-180°$ until switch activation newly occurs. When it does, θ is updated to $-360°$ according to Equation 4.3, but θ' is not, and at this moment, the modified control program transmits values of t, θ, and θ' to the laptop. Afterward, θ' is updated to $-360°$ and the entire process repeats. In this way, values of t, θ, and θ' are transmitted to the laptop at every pass of `loop` when a switch activation occurs. The value of θ at these passes is 540°, 720°, 900°, etc.

With the modified control program uploaded to the microprocessor, a trial roll is performed with $\omega_d = -2.0$ rad/s, and output values from the roll are imported to a spreadsheet in three columns, one for t and the other two for θ and θ'. After the trial roll is completed, θ minus θ' is computed at every output interval. This difference is the error that accumulates over a half-turn of the robot between switch activations when employing the angular computation strategy described in Section 4.2, the strategy used to compute angular position of the robot in the primary experiment. The test is repeated for $\omega_d = -2.2$ rad/s and $\omega_d = -2.4$ rad/s. Results from the three tests are included in Table 4.1, in which angle error is listed for the trial rolls, corresponding to the three desired velocities, within the first 6.3 meters of travel.

Errors listed in Table 4.1 are all positive except for one value, which means the angular position computation strategy employed by the control program is biased towards overvaluing the magnitude of robot angular position. As a result of the bias,

Table 4.1: Angular Position Measurement Error

	$\omega_d = -2.0u_s(t)$ rad/s	$\omega_d = -2.2u_s(t)$ rad/s	$\omega_d = -2.4u_s(t)$ rad/s
$\theta = 180°$	0.21°	0.30°	0.77°
$\theta = 360°$	1.23°	0.91°	0.73°
$\theta = 540°$	1.40°	1.23°	1.81°
$\theta = 720°$	0.84°	1.20°	0.82°
$\theta = 900°$	0.27°	1.27°	2.37°
$\theta = 1080°$	2.04°	0.96°	1.40°

reported values of θ' are almost always a few degrees less than θ, such as $-181.54°$, $-362.51°$, and $-543.62°$, for example. One seemingly possible explanation for these discrepancies is that θ' is accurate, and the position switches are activating later than they should. This possibility is improbable, however, because the switches are calibrated by slightly bending the lever arms (Section 3.4.2) in order to bring about switch activation when A is observed to line-up with a digital level that's fixed in the vertical orientation. This calibration is performed regularly, so that measurement of angular position by the switches is certainly highly accurate.

A more likely explanation for the negative errors in Tab. 4.1 is that the gyroscope used onboard the rolling robot is slightly negatively biased, resulting in noticeable overvaluation of angular position magnitude after numerical integration of angular velocity has been performed over time. If the largest magnitude of these errors in Tab. 4.1, 4.74° over 180° of travel, is assumed to be the largest possible magnitude of error associated the angular computation strategy, and if 4.74° is assumed to be divided equally over 180° of travel between switch activations, then at $\theta_t = 65°$ beyond switch activation, there is uncertainty of $\pm(65/180)4.74 = \pm 1.71°$ associated with the computation strategy. Furthermore, if constant angular velocity of the robot is assumed between switch activations, then for an error of 4.74° between switch activations, maximum error magnitude in gyroscope measurement can be backed-out, giving 3.31 degrees per second, or 0.0578 radians per second, during the trial rolls.

4.4 Demonstration of Stable, Controlled Locomotion

This section describes a test to find the approximate range of k_f that results in stable, controlled locomotion of the robot for a trial roll defined by $\theta_t = 45°$, $\theta_m = 35°$, $\omega_d = -2.2u_s(t)$ rad/s, $\omega(0) = 0$, $\theta(0) \approx -15°$ and $a(0) = 0.349$ m. In the test, k_f is varied in steps, while k_s is varied only to maintain stability in response to changes in k_f. There is a two-fold purpose in performing this test. The first purpose is to demonstrate that stable locomotion of the robot is achieved by the robot running under influence of the control system described in Section 3.6. The second purpose is related to design of the primary experiment, outlined in Section 4.6, in which four control constants, including k_f, are varied in order to affect the robot steady state velocity. Once the stable range of k_f is found in the test described here, it is then used to design an experiment with appropriate limits on k_f, thus streamlining the experiment.

Stability, as used herein, refers to two conditions being met relative to robot controlled angular velocity. The first condition is the robot must speed-up quickly from rest, so that rise time – that is, how long it takes for velocity of the robot to reach the desired velocity starting from rest – is no more than 6 s. If rise time is longer than 6 s, there is not enough length on the roll track to facilitate a sufficient span of steady state motion for analysis. Because the primary experiment is an investigation of locomotion energy economy at steady state, this condition is essential. The second condition is that after robot velocity has risen to ω_d, the running average of its angular velocity, ω_r, must thereafter remain close to the desired constant velocity. Some reasonable variation within several degrees per second is permissible, but large and fluctuating dips in angular velocity or a steady increase of ω_r over time are considered unstable response characteristics. This latter stability condition ensures that trial rolls with different desired velocity profiles will have different average steady state velocities. Without being assured of this difference, it would be impossible to draw conclusions about the combined effects that ω_d and other control parameters have on locomotion economy of the robot as a function of its velocity.

The test begins by preparing the simulation program described in Section 3.8 to simulate the aforementioned trial roll. The simulation is run repeatedly, and k_f and k_s are changed before each run. Plots of ω versus trial roll time are generated by the simulation program and observed. Through trial and error, a combination of k_f and k_s is obtained that provides stable locomotion, at least in simulation. That combination is then coded into the control program, along with other control constants and initial conditions, and uploaded to the microprocessor. Then on the roll track described in Section 4.1, a trial roll of the robot is performed, and ω and ω_r are plotted versus run time. If ω_r is observed to be shifted high relative to ω_d, then k_s is iteratively raised until the response is stable by inspection, as in the plot in Fig. 4.2. Then k_f is stepped down by 0.005, and the process is repeated. Trial rolls are continued in this manner until a value of k_f is reached where robot velocity response cannot seemingly be made stable by changing k_s. At this value of k_f (which is found to be 0.005), lower values of k_s cause robot velocity to remain well below ω_d until late in the run, as in the plot in Fig. 4.3. Upon raising k_s, the robot generally moves a bit faster, but rise time remains too long. Trial rolls for lower values of k_f give similar results relative to changes in k_s. Thus, it is likely impossible to find a combination of control constants that result in stable velocity response for $\omega_d = 2.0$ rad/s when $k_f \leq 0.005$.

After the lower limit of k_f is established, trial rolls are resumed on the roll track starting with the initial combination of k_f and k_s provided before by simulation. Now, however, k_f is consecutively increased rather than decreased in steps of 0.005, and k_s is again iteratively changed to maintain stability. A limit on k_f is eventually reached at which robot velocity response cannot seemingly be made stable by changing k_s. At

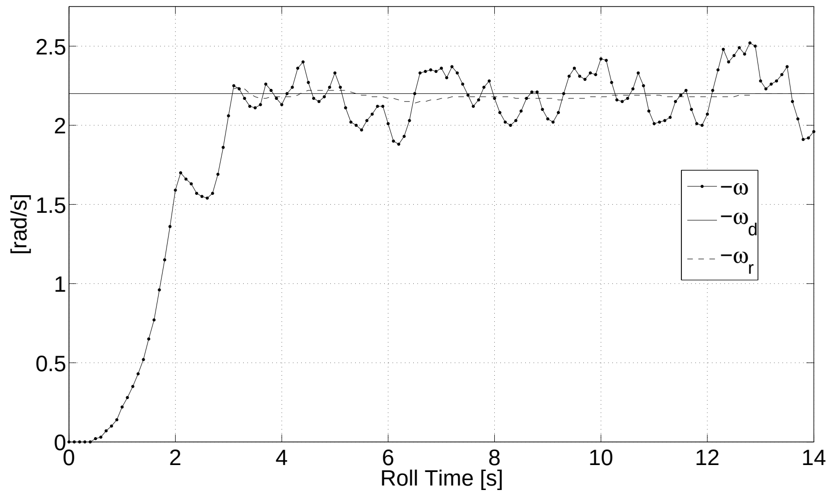


Figure 4.2: An example of stable locomotion of the robot for $\omega_d = -2.2$ rad/s. Angular velocity rises quickly to the desired level, -2.2 rad/s, and ω_r remains close to -2.2 rad/s thereafter.

this value of k_s (which is found to be 0.050), lower values of k_s cause robot velocity to remain well below ω_d for the entire trial roll. Upon raising k_s, suddenly robot velocity fluctuates in large magnitude and becomes unstable, as in the plot in Fig. 4.4. Trial rolls for higher values of k_f give similar results relative to changes in k_s. Thus, it is difficult, if not impossible, to find a combination of control constants that result in stable velocity response when $k_f \geq 0.050$.

With a stable range of k_f established, several additional trial rolls are performed for $\omega_d = -2.0$ rad/s and $\omega_d = -2.4$. For each of these desired velocities, trial rolls are performed with k_f chosen randomly within the stability range for k_f previously established: $0.005 \leq k_f \leq 0.050$. Results of these additional trials demonstrate that, in addition to stability for $\omega_d = -2.2$ rad/s, the rolling robot also achieves stable velocity control with $\omega_d = -2.0$ rad/s and $\omega_d = -2.4$ rad/s. Figs. 4.5 and 4.6 show plots of velocity versus run time for stable trial rolls for $\omega_d = -2.0$ rad/s and -2.4 rad/s.

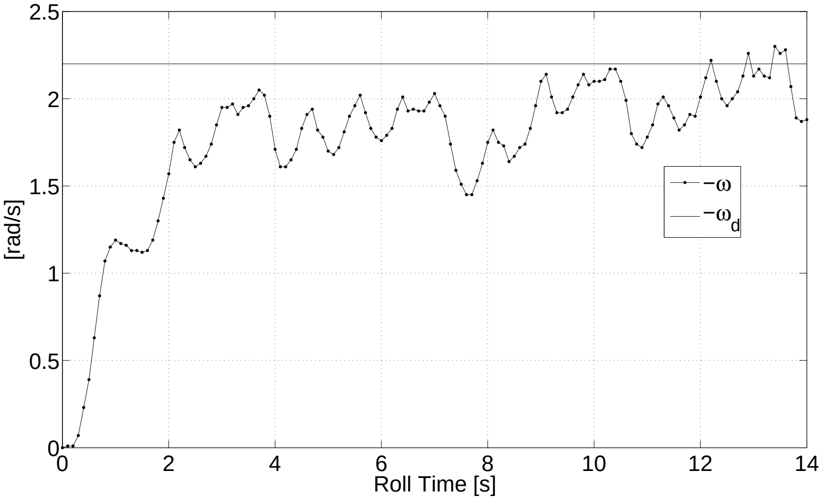


Figure 4.3: An example of unstable locomotion for the robot for $\omega_d = -2.2$ rad/s. Angular velocity reaches the desired level too late in the trial roll.

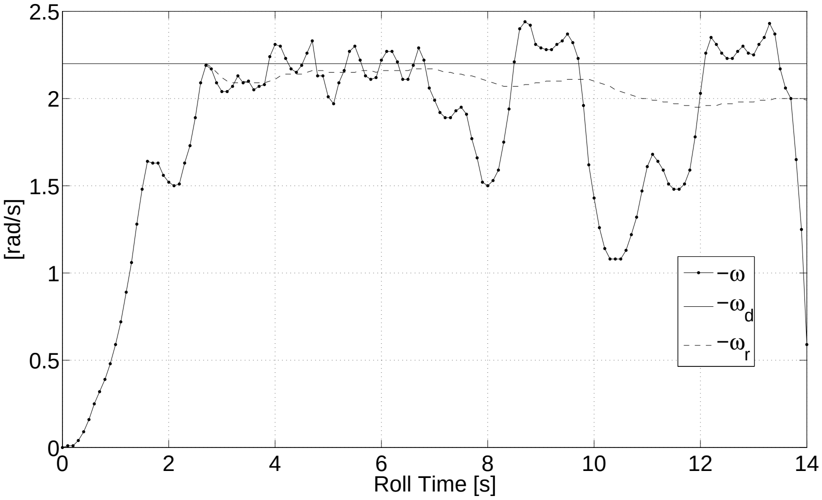


Figure 4.4: An example of unstable locomotion for the robot for $\omega_d = -2.2$ rad/s. Average angular velocity dips far below the desired level during the latter half of the trial roll.

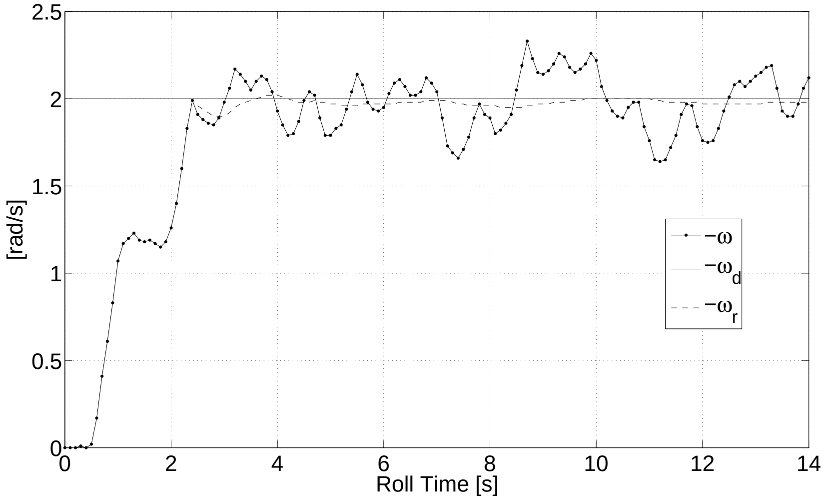


Figure 4.5: An example of stable locomotion of the robot for $\omega_d = -2.0$ rad/s. Angular velocity rises quickly to -2.0 rad/s, and ω_r remains close to -2.0 rad/s thereafter.

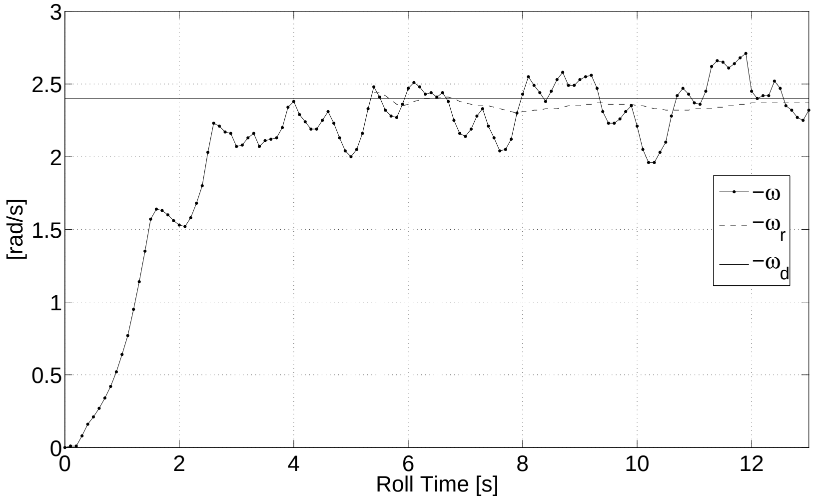


Figure 4.6: An example of stable locomotion of the robot for $\omega_d = -2.4$ rad/s. Angular velocity rises quickly to -2.4 rad/s, and ω_r remains close to -2.4 rad/s thereafter.

4.5 Effect of Vibration on Onboard Angular Velocity Measurement

Angular velocity of the rolling robot is measured by a gyroscope, which is attached to one column of the linear actuator, as described in Section 3.4.1. This location is deemed satisfactory for gyroscope measurement, because columns of the linear actuator are relatively stiff and do not easily bend. However, the actuator is connected to the robot at its outer surface, which obviously bends and vibrates as the robot rolls in controlled locomotion, and this vibration of the outer surface affects motion of the linear actuator, which in turn affects gyroscope measurement. In fact, during trial rolls of the robot, it is observed that the linear actuator exhibits a rotational shudder, presumably brought on by vibration of the outer surface, that can be described as a quick, back-and-forth rotation of the actuator, lasting approximately three-quarters of a second. It is also observed that when shudder occurs, the linear actuator (A-axis) is at or near the horizontal orientation, and then only when actuation is near or has just reached completion. Frequency of shudder occurrence is observed to be about the same for trial rolls in which desired velocity of the robot varies from -2.0 to -2.4 rad/s at steady state.

While such vibration of the robot is to be expected, the shudder phenomenon is disconcerting because it visibly affects angular velocity at the location where the gyroscope is attached; and since angular velocity measurements of the gyroscope are used as feedback in the control system, shudder of the robot is potentially a control problem. The specific concern is that if measurement of angular velocity occurs at the same time as a shudder, the measurement could be raised or lowered relative to what the measured velocity would be if there were no shudder, thereby causing the control system to overreact or underreact in a manner that renders it inaccurate.

In light of these observations and considerations, a test is performed to better understand the impact of shudder on angular velocity measurements of the gyroscope. For the test, the control program described in Section 4.2 is modified to transmit θ

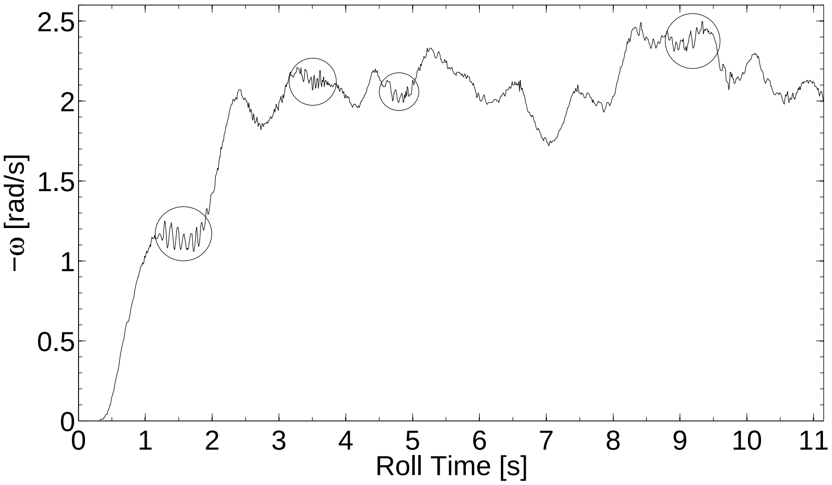


Figure 4.7: Shutter occurrences for the $\theta_t = 55°$ trial roll are manifested as angular velocity fluctuations with a frequency of approximately 12 Hz.

and ω to the laptop serial monitor at every pass of `loop`, or every 10 ms. Then five trial rolls of the robot are conducted for the following values of θ_t: $35°, 45°, 55°, 65°$, and 75°; and the rolls are further defined by $\theta_m = 35°$, $\omega_d = -2.2$ rad/s, $\omega(0) = 0$, $\theta(0) \approx -15°$ and $a(0) = 0.349$ m. Robot locomotion for each trial roll is video recorded using a high speed video camera, wherein shudder occurrences are identified and their approximate roll times are logged. Subsequently, values of θ and ω for each trial roll are copied from the laptop serial monitor and pasted into a spreadsheet, and ω is plotted versus roll time. At the sample rate of 100 Hz, high frequency shudder vibrations are manifested in the plots; then by comparing video footage with corresponding plots, shutter occurrences are confirmed to be pockets of relatively large and persistent angular velocity fluctuations, having a duration of approximately 0.75 s with a frequency of 12 Hz on average. Fig. 4.7 shows the plot of ω versus roll time for the $\theta_t = 55°$ trial roll, during which four shudders, identified in the plot with circles, occur during 11.5 seconds of roll time.

Plots of angular velocity versus roll time for the five trial rolls in the test are scrutinized, and shudder occurrences are identified and tallied. Using data that is collected for each trial roll with corresponding plots, orientation of A relative to the horizontal is computed (to the nearest degree) at run times during which shudder occurs. Using this record, several shudder-related statistics are calculated, and are reported as follows. Over the five trial rolls, shudder occurs 18 times for an average of approximately one shudder every 3.6 s. In addition, when shudder does occur, the probability of it occurring when A is oriented at or within two degrees of $35°$ clockwise or $55°$ counterclockwise relative to the horizontal (these orientations correspond to $\theta_m = 35°$) are 2/18 and 0/18, respectively. In conclusion, for trial rolls of the robot in which $\theta_m = 35°$, shudder has a negligible effect on gyroscope measurement of angular velocity, mostly because shudder occurs infrequently at this measurement orientation; and when it does occur, the effect is small, raising or lowering the measurement by 3 deg/s, at most, relative to what robot velocity would be if no shudder were present. For a stable trial roll with $\omega_d = -2.2$ rad/s, this effect amounts to roughly 3% of robot actual velocity at steady state.

4.6 The Primary Experiment

The primary experiment involves a series of trial rolls of the robot in which select control parameters are varied. While the rolls are in progress, parameters relevant to robot locomotion are collected onboard the robot and transmitted via radio signal to a nearby laptop after each roll is completed, where the information is saved for subsequent analysis. By applying the work-energy theorem to each roll, the amount of non-conservative work that retards forward motion of the robot is calculated per unit roll distance and is denoted as ϱ with units of joules per meter. This value, which is explained in detail in Section 4.6.3, is used to identify which trial rolls are most economical in terms of energy expenditure per unit roll distance. Ultimately, a range of θ_t is identified in which the robot operates most economically for each of the following average steady state velocities of the robot: -2.0 rad/s, -2.2 rad/s and -2.4 rad/s. The experiment is divided into two phases, and experimental procedures for these phases are detailed in the two subsections that immediately follow herein. A third subsection is devoted to explaining how non-conservative work is calculated and how ϱ is used, through a statistical approach, to identify which trial rolls have the most economical locomotion. Finally, uncertainty in the calculation of ϱ is discussed in the fourth and concluding subsection.

4.6.1 Procedure, Phase 1

In Phase 1 of the experiment, trial rolls are conducted that provide information used to solve the following 15 optimization problems:

For each of $\theta_t = 25°, 35°, 45°, 55°, 65°$, find ($k_f$, k_s) that minimizes ϱ for

1.1-1.5. $\omega_d(t) = -2.0\mathrm{u}_s(\mathrm{t})$ rad/s

1.6-1.10. $\omega_d(t) = -2.2\mathrm{u}_s(\mathrm{t})$ rad/s

1.11-1.15. $\omega_d(t) = -2.4\mathrm{u}_s(\mathrm{t})$ rad/s

Based on the stable range of k_f found in Section 4.4, these problems have the following constraint: $k_f \in \{0.01, 0.02, 0.03, 0.04\}$.

Phase 1 of the experiment starts with setting various parameters in the control program. For Problem 1, ω_d is set to -2.0 rad/s, and θ_t is set to 25°. Measurement trigger angle, θ_m, is set to 25°. The `loop` function in the control program is modified so that when $S \geq S_m = 8.5$ m, the outer surface automatically actuates to the circular configuration, according to the actuation profiles given in Section 3.6, and the robot stays in that configuration until the end of the roll at $t = 15$ s. Prior to these preparations, columns of the linear actuator are dissembled and cleaned/oiled with WD-40. Also, the batteries are fully charged, and the motor is allowed to remain idle for at least three hours in order to reach the ambient temperature of the laboratory, which remains close to 70° F, regardless of time of day.

All trial rolls in Phase 1 of the experiment are conducted on the roll track described in Section 4.1. The robot is initially positioned and held by the user, so that it commences from the same location – at the beginning of one side of the roll track along the track's centerline – and with a consistent set of initial conditions: $\theta(0) \approx -10°$, $\omega(0) = 0$, and $a(0) = 0.349$ m. As described in Section 4.2, an LED on the main microprocessor illuminates when the onboard sensors commence measurement, notifying the user that the trial roll has begun and he should immediately let go of the robot and allow it to act on its own. With the given initial conditions, the robot is poised to roll forward, helping the robot accelerate to steady state velocity once it's let go.

Next, several combinations of k_f and k_s are roll tested in order to find values of k_s that result in stable velocity control for each of the four allowable values of k_f for Problem 1.1. Typically, at least ten rolls are required to do so, and in the process, the motor is sufficiently warmed-up prior to the trial rolls that are to be conducted for Problem 1.1. The procedure to find k_s is identical to the procedure outlined in Section 4.4, in which k_f is set to one of the four allowable values, and k_s is iteratively changed as the robot is repeatedly rolled and the response is inspected for stability. If stability is not deemed possible for a certain value of k_f, it is disregarded as a candidate of optimality for the problem. If stability is not deemed possible for all four allowable values of k_f, the current value of θ_t is unviable, and the entire problem is disregarded.

Assuming all allowable values of k_f result in stable velocity control for the problem at hand, four combinations of k_f and k_s are found and are denoted herein as C1, C2, C3 and C4. A random number generator is employed to shuffle the four combinations ten times, with a result that is similar to the following 40-member sequence of combinations:

{C2, C1, C3, C4; C1, C2, C3, C4; C4, C3, C3, C1; C1, C2, C4, C3; C2, C3, C4, C1;

C1, C3, C4, C2; C3, C1, C2, C4; C3, C1, C2, C4; C2, C4, C1, C3; C3, C4, C1, C2}

Then using combinations from the random sequence, 40 trial rolls are conducted in order to collect information to address the problem. The first of these trial rolls is conducted with k_f and k_s set according to the first entry in the sequence, and transmitted parameters from the roll are saved. Afterward, k_f and k_s are set according to the second entry in the sequence, and a second trial roll is conducted, timed to commence 45 s after the first roll ends, and parameters from the roll are saved. Several more trial rolls are conducted in this manner, corresponding to the first half of the combinations in the sequence. The batteries are then fully recharged, and the motor is allowed to cool for at least three hours. Afterward, the motor is sufficiently warmed, and the remaining trial rolls in the sequence are conducted in the same ordered manner using the second half of the combinations in the sequence. By the time the last combination in the sequence is used, ten trial rolls are conducted for each combination of control constants.

Trial rolls for all problems in Phase 1 of the primary experiment are shuffled in this manner relative to control constant combinations in order to avoid motor efficiency biasing that could affect energy measurements. For example, if 40 trial rolls were conducted according to the following non-randomized sequence:

{C1, C1, C1, C1, C1, C1, C1, C1, C1, C1; C2, C2 C2, C2, C2, C2 C2, C2, C2, C2;
C3, C3, C3, C3, C3, C3, C3, C3, C3, C3; C4, C4, C4, C4, C4, C4, C4, C4, C4, C4}

then the first ten rolls with C1 would be actuated by a relatively cool motor, whereas the last ten rolls with C4 would be actuated by a relatively warm motor. Efficiency of the motor changes as its temperature rises [26], so in this hypothetical situation with no shuffling of the combinations, values of ϱ corresponding to C1 and C4 would be incomparable, at least for the matter of energy consumed per unit roll distance. However, because roll sequences in the actual experiment are randomly shuffled, any would-be biasing is nullified, and energy measurements for a given control constant combination can verily be compared to the other three.

After trial rolls for Problem 1 are finished, columns of the linear actuator are newly cleaned/lubricated, the batteries are fully recharged, and the motor is allowed to cool for at least three hours. The experimental procedure for Problem 1.1 is then repeated five times with the following changes in order to address Problems 1.2 - 1.5: $\theta_m = 35°$; $\theta_t = 35°, 45°, 55°, 65°$; and the roll sequence is randomly reshuffled for each problem. For Problems 1.6 - 1.10, the process remains the same as for Problems 1.1 - 1.5, except $\omega_d = -2.2$ rad/s and $S_m = 8.8$ m. For Problems 1.11-1.15, $\omega_d = -2.4$ rad/s and S_m remains at 8.8 m.

Upon concluding trial rolls for each problem, saved roll parameters are used to calculate ϱ for each trial roll, according to the strategy described in Section 4.6.3. Resulting values of ϱ corresponding to C1, C2, C3 and C4 for a given problem form populations denoted as P1, P2, P3 and P4, respectively; and the average value of ϱ (denoted as $\overline{\varrho}$) for each of these populations is recorded. In addition, uncertainty associated with ϱ (denoted as $\delta\varrho$) is calculated for each trial roll using Equation 4.39 in Section 4.6.4, and the largest value of $\delta\varrho$ within each population is recorded. At this point in the investigation, average values of ϱ could be listed from lowest to highest as a way to rank corresponding combinations in terms of energy economy. Questions arise, however, if this ranking approach is taken. Specifically, how different would two averages have to be before their corresponding combinations should be considered different? For example, if $\overline{\varrho}$ corresponding to P1 is 2.10 N/m and $\overline{\varrho}$ corresponding to P2 is 2.13 N/m, should one conclude that C1 is superior to C2 in terms of economy, or is the magnitude of difference between them too small to know for sure? In addition, it is not clear how uncertainty would come into play if this ranking approach were taken.

Given these reservations, a preferable way to rank the combinations is to compare energy economy associated with the combinations using the testing of hypothesis approach [48]. Rather than simply comparing averages, testing of hypothesis allows for the entirety of the ϱ-populations to be compared to one another using the median test [54], a numerical procedure that indicates when there is a statistically significant difference between two population groups with regard to their measures of central tendency. The median test is chosen for the comparison above other population comparison tests, because it is valid even when the two populations being compared have small sample sizes, are not normally distributed, and have dissimilar variances [54] – a trio of conditions that is sometimes true for the Phase 1 population comparisons. In addition, the testing of hypothesis method allows for $\delta\varrho$ to be taken into consideration as part of the comparison. Testing of hypothesis is so named because it is based on a default position, or null hypothesis, that is assumed to be true before any experimentation is conducted [48]. If, after experimentation, the null hypothesis is shown to be false, then an alternative hypothesis is accepted. For the experiment herein, the null hypothesis states that for any two control constant combinations, neither combination is superior with regard to energy economy. The alternative hypothesis is that one control combination results in locomotion that is more economical – namely, the combination whose ϱ-population has less positive central tendency.

Testing of hypothesis is carried out by a computer program (Listing 4 in the Appendix) developed especially for performing comparisons of ϱ-populations. The test program accepts as input two ϱ-populations and begins by randomly perturbing them to account for uncertainty. It performs this perturbation by first choosing a number from a normal distribution with a mean of zero and a standard deviation equal to one-third of the largest uncertainty recorded for the first of the two populations. If the randomly chosen number is larger in magnitude than the largest uncertainty, the number is rechosen until its magnitude is less than or equal to the largest uncertainty. The test program then adds the randomly chosen number to a value of ϱ in the first population. Afterward, a new number is chosen at random in the same way and added to the second value of ϱ in the population, and so forth, until all values in the population have been perturbed. Perturbation is repeated in this manner on the second population, but now with a standard deviation equal to one-third of the largest uncertainty recorded for the second population. The test program then performs the median test on the two perturbed populations and passes if they are significantly different with regard to central tendency with 90% or greater confidence; otherwise the test fails. When the median test passes and the first population has less positive central tendency, a variable is augmented. When the test passes and the second population has less positive tendency, a different variable is augmented. Perturbation/testing/augmentation is repeatedly carried out by the test program in this manner by means of a looping structure until a total of $N = 100,000$ median tests have been performed. The test program concludes by outputting a value, referred to as the pass percentage, which is equal to one hundred times the largest augmented variable divided by N.

In order to determine that $N = 100,000$ is an appropriate number of comparison tests to perform in the test program, a batch process is developed that executes the test program repeatedly for a given comparison. At the first execution of the program, N is set to $5,000$ and is augmented by $1,000$ at each execution of the test program thereafter, until $N = 50,000$ when the batch process terminates. Values of N, along with corresponding pass percentage, are saved by the batch process for each execution, so that after termination, values of pass percentage are plotted versus N, from $N = 5,000$ to $50,000$, along with a 10 moving average [55] of pass percentage, as shown in the example graph in Fig. 4.8. After the graph is made, the running average plot is inspected for a value of N where it begins to display a steady state pattern characterized by horizontal flatness on average. In addition, the plot of pass percentage is inspected for a value of N where change of pass percentage is, and remains thereafter, less than 1% over each 1000-gradation of N. The greater of these two values of N is determined to be a sufficiently high number of tests, meaning that

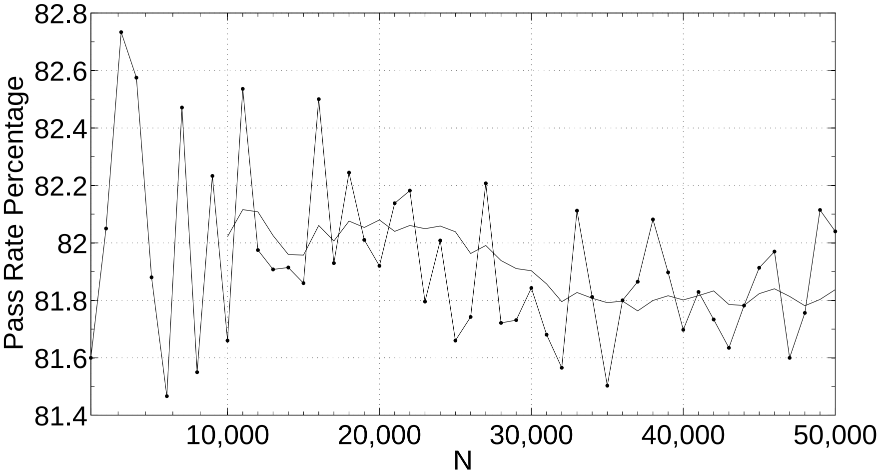


Figure 4.8: For two populations, the batch process executes the test program for various values of N, which is the number of median tests performed at each execution. Pass percentage steadies out beyond a sufficiently high value of N.

for values of N greater than the sufficiently high number, pass percentage steadies out and does not change significantly. For the example graph in Fig. 4.8, the sufficiently high value of N is about $32,000$. If the sufficiently high value of N is found to be under $50,000$ for a comparison test, the appropriate value of N is conservatively taken as $100,000$. For Phase 1 comparisons, the batch process is run for each comparison, and the appropriate value of N is determined to be $100,000$ every time.

The decision rule for comparison of two populations is as follows. If the pass percentage is greater than or equal to 90, then the null hypothesis for the two corresponding control combinations is rejected and the alternative hypothesis is accepted. In this case, the combination corresponding to the ϱ-population with less positive central tendency (higher pass rate) is determined to be superior in terms of energy economy. If, on the other hand, the pass percentage is less than 90, then the null hypothesis is kept, and neither combination is determined to be superior.

Grids are built to catalogue results of the pairwise comparisons and to help determine which control combination out of the four is superior in terms of locomotion energy economy. Two comparison grids are built for each problem. Rows and columns in the grids are titled with the combination names, as shown in Fig. 4.9. The top comparison grid in the figure shows the pass percentage of the largest augmented variable, rounded to the nearest whole number, along with the corresponding control constant combination for each pairwise comparison in the problem. If no combination is listed in a grid cell, then there are no passes of the median test for that comparison. The bottom grid shows results of the decision rule for each pairwise comparison. If the null hypothesis is kept relative to two combinations, a dash is placed in the entry where corresponding row and column intersect; or if the alternative hypothesis is accepted, name of the superior combination is placed in the entry.

After all entries in the grids are filled, attention is directed to the bottom grid's columns. If a column contains anything other than a dash or the title of that column, the combination is rejected, signified by a cross-out. If only one column remains after rejections have been made, the corresponding combination is determined to be superior in terms of energy economy of the robot. If more than one column remains after rejections have been made, corresponding combinations are determined to be equal in terms of locomotion energy economy. In this case, the combination corresponding to the lowest value of $\overline{\varrho}$ among the remaining combinations is selected as superior, signified by parenthesis. If no columns remain after rejections have been made, then all combinations are determined to be equal in terms of energy economy. Again, in this case, the combination corresponding to the lowest value of $\overline{\varrho}$ among the remaining combinations is selected as superior. Selection according to $\overline{\varrho}$ is largely arbitrary, and it is made in order to have a control combination from the problem represented in Phase 2 of the experiment.

4.6.2 Procedure, Phase 2

In Phase 2, trial rolls are conducted that provide information used to solve the following three optimization problems:

	C1	C2	C3	C4
C1		C2, 3%	C3, 0%	C4, 100%
C2	C2, 3%		C2, 24%	C4, 76%
C3	C3, 0%	C2, 24%		C4, 100%
C4	C4, 100%	C4, 76%	C4, 100%	

	~~C1~~	C2	~~C3~~	(C4)
C1		-	-	C4
C2	-		-	-
C3	-	-		C4
C4	C4	-	C4	

Figure 4.9: Examples of pairwise comparison grids for a problem in Phase 1 of the primary experiment. The grids communicate which control constant combinations are deemed superior in terms of locomotion energy economy.

Find θ_t that minimizes ϱ for

2.1. $\omega_d(t) = -2.0\mathrm{u}_s(\mathrm{t})$ rad/s with $\theta_t \in \{35°, 45°, 55°, 65°\}$

2.2. $\omega_d(t) = -2.2\mathrm{u}_s(\mathrm{t})$ rad/s with $\theta_t \in \{25°, 35°, 45°, 55°\}$

2.3. $\omega_d(t) = -2.4\mathrm{u}_s(\mathrm{t})$ rad/s with $\theta_t \in \{25°, 35°, 45°\}$

Optimal combinations of k_f and k_s for given values of θ_t determined in Phase 1 are used in Phase 2. By the time Phase 2 of the experiment is conducted, combinations of θ_t and ω_d that are not viable for stability have been identified in Phase 1, and so these exclusions are reflected in the constraints for the above problems.

The second phase of the experiment starts with setting various parameters in the control program. The desired velocity, ω_d, is set to -2.0; θ_t is set to 35°, which is the lowest value of θ_t found to result in stable locomotion in Phase 1; and θ_m is set to 35°. The `loop` function in the control program is modified so that when $S \geq S_m = 8.5$ m, the outer surface automatically actuates to the circular configuration and stays in that configuration until $t = 15$ s. Prior to these preparations, columns of the linear actuator are dissembled and cleaned/lubricated with WD-40. Also, the batteries are fully charged, and the motor is allowed to remain idle for at least three hours in order to reach the ambient temperature of the laboratory. Next, ten preliminary trial rolls of the robot are conducted in order to warm-up the motor prior to the actual trial rolls that are to be conducted as part of the experiment for Problem 2.1. All trial rolls in Phase 2 of the experiment, including these preliminary rolls, are conducted on the roll track described in Section 4.1 and are given the same initial conditions as in Phase 1.

A random number generator is employed to shuffle the order of the trial rolls relative to viable values of θ_t. As in Phase 1 of the experiment, shuffling is done to avoid any would-be biasing caused by rising temperature of the servo motor. For example, if viable values of θ_t are 35°, 45°, 55°, and 65°, the random number generator would shuffle these four values ten times, resulting in a 40-member sequence of θ_t values, with units of degrees, similar to the following:

$$\{35, 55, 45, 65; 65, 45, 55, 35; 35, 65, 45, 55; 55, 35, 45, 65; 65, 45, 35, 55;$$
$$45, 35, 65, 55; 55, 65, 45, 35; 45, 55, 35, 65; 35, 55, 45, 65; 55, 45, 65, 35\}$$

Then using the order of θ_t from the shuffled sequence, along with optimally efficient values of k_f and k_s that are determined in Phase 1 of the experiment, 40 trial rolls are conducted in order to collect information to address Problem 2.1. The first of these trial rolls is conducted with θ_t set to the first entry in the sequence, and information from the trial roll is saved. Afterward, θ_t is set to the second entry in the sequence, and a second trial roll is conducted, timed to commence 45 s after the first roll ends, and information from the trial roll is saved. Several more trial rolls are conducted in this manner, corresponding to the first half of the combinations in the sequence. The batteries are then fully recharged, and the motor is allowed to cool for at least three hours. Afterward, the motor is sufficiently warmed, and the remaining trial rolls in the sequence are conducted in the same ordered manner, using the second half of the combinations in the sequence. By the time the last value in the sequence is used, ten trial rolls are conducted for each value of θ_t.

After trial rolls for Problem 2.1 are finished, columns of the linear actuator are newly cleaned/lubricated, the batteries are fully recharged, and the motor is allowed to cool for at least three hours. The experimental procedure for Problem 2.1 is then repeated with the following changes in order to address Problem 2.2: $\omega_d = -2.2$ rad/s, $S_m = 8.8$ m, and the roll sequence is reshuffled. For Problem 2.3, $\omega_d = -2.4$ rad/s, S_m remains at 8.8 m, and the roll sequence is again reshuffled. For these last two problems, the value of θ_m is set to 25° when $\theta_t = 25$, otherwise it is set to 35°.

Upon concluding trial rolls for problems in Phase 2, saved roll parameters are used to calculate ϱ for each trial roll, according to the strategy described in Section 4.6.3. The resulting ten values of ϱ corresponding to each value of θ_t for a given problem form populations denoted as P1, P2, etc., and the average value of ϱ for each of these populations is recorded. In addition, $\delta\varrho$ is calculated for each roll according to Equation 4.39 in Section 4.6.4, and the largest value of $\delta\varrho$ within each population is recorded. Rho populations associated with values of θ_t for each problem are compared one-by-one as in Phase 1, whereby a computer program perturbs the populations and applies the median test. The value of N, which is the number of median tests performed in the computer program, is set to $100,000$ as in Phase 1 (Section 4.6.1), and the decision rule for comparison of two populations is also the same as in Phase 1, and is as follows. If the pass percentage of the largest augmented variable is greater than 90, then the null hypothesis is rejected and the alternative hypothesis is accepted. In this case, the value of θ_t corresponding to the ϱ-population with less positive central tendency is determined to be superior in terms of energy economy. If, on the other hand, the pass percentage of the largest augmented variable is less than or equal to 90, then the null hypothesis is kept, and neither value of θ_t is determined to be superior.

Two comparison grids are built to catalogue results of the population comparisons and to help determine the best value of θ_t for each problem, if possible. Rows and columns in the grids are titled with a value of θ_t, as shown in Fig. 4.9. The top grid

	35°	45°	55°	65°
35°		45°, 100%	55°, 49%	35°, 100%
45°	45°, 100%		45°, 21.3%	45°, 100%
55°	55°, 49%	45°, 21%		55°, 100%
65°	35°, 100%	45°, 100%	55°, 100%	

	~~35°~~	45°	55°	~~65°~~
35°		45°	-	35°
45°	45°		-	45°
55°	-	-		55°
65°	35°	45°	55°	

Figure 4.10: Examples of pairwise comparison grids for a problem in Phase 2 of the primary experiment. The grids communicate which trigger angles are deemed superior in terms of locomotion energy economy.

in the figure shows the pass percentage of the largest augmented variable, rounded to the nearest whole number, for each comparison, along with the corresponding value of θ_t. The bottom grid shows results of the decision rule for each comparison. If the null hypothesis is kept relative to two combinations, a dash is placed in the entry where corresponding row and column intersect; or if the alternative hypothesis is accepted, the superior value of θ_t is placed in the entry. After all entries in the grids are filled, attention is directed to the bottom grid's columns. If a column contains anything other than a dash or the title of that column, the corresponding value of θ_t is rejected, signified by a cross-out. If only one column remains after rejections have been made, the corresponding value of θ_t is determined to be superior in terms of energy economy of robot locomotion. If more than one column remains after rejections have been made, corresponding values of θ_t are determined to be superior to other values of θ_t in terms of energy economy.

4.6.3 Energy Per Unit Roll Distance, ϱ

As the robot rolls in a controlled manner, various resistive agents perform non-conservative work that retards robot motion, causing it to move slower than it would if these agents were not present. The primary resistive agents that slow the robot are: servo motor inefficiency, friction in the columns of the linear actuator, and rolling resistance torque. For a given roll distance, ΔS, of the robot, the sum of non-conservative work performed by these agents is a negative value denoted herein as W_{NC}, and non-conservative work performed per unit distance rolled is defined as

$$\varrho \equiv -\frac{W_{NC}}{\Delta S} \tag{4.6}$$

The negative sign used in the definition is meant to ensure that ϱ is always positive, a convention adopted merely for convenience in reporting results. In the primary experiment, energy economy of robot locomotion is evaluated using this parameter.

In order to help understand the significance of ϱ in relation to robot locomotion, imagine two trial rolls of the robot, each with different combinations of control constants (k_f, k_s, θ_t) yet with identical values of ω_d and θ_m and with identical initial conditions. In these hypothetical rolls, say the robot exhibits stable, steady state average velocity equal to ω_d. An obvious approach to compare energy economy of these rolls would be to measure the work done by the servo motor for both rolls after the robot has achieved steady state velocity. This measurement could be done by coupling a torque sensor and encoder to the servo motor, and integrating the motor's measured torque over the motor's angular displacement. This approach would be mechanically inconvenient, however, because attaching two mechanical sensors to the servo motor would add significant weight and complication to the robot.

Instead, an equally valid method to evaluate energy economy of the robot – and the approach used for the primary experiment documented herein – is to calculate the non-conservative work, W_{NC}, performed on the robot through a process of electrical measurement at the servo motor. This approach, which is explained in the following paragraphs, is based on the simple idea that non-conservative work causes the linear actuator to work more than it would if resistive agents were not present in the system;

therefore, when comparing locomotion economy of two trial rolls, the roll that registers a smaller magnitude of non-conservative work to achieve comparable results actually works less, and is therefore the more economical roll. In other words, the control combination corresponding to a smaller value of ϱ is superior in terms of energy economy.

In addition to being an indicator of how freely the robot moves, ϱ can also be thought of as an indicator of the electrical energy required to move the robot over a certain roll distance. Continuing with the hypothetical rolls mentioned previously, imagine that for each roll, angle of the robot and height of robot center of mass are the same at the start as the end of the roll, so there is no change in kinetic energy and potential energy. In this case, application of the work-energy theorem [40] requires that energy put into the robot during a roll is equal to the magnitude of non-conservative work performed on the robot from beginning to end. Therefore, the magnitude of W_{NC} is exactly equal to the amount of electrical energy required to move the robot down the track. Of course, if there were changes in kinetic and/or potential energy from beginning to end of the roll, then a slightly more complicated approach would be taken to calculate non-conservative work, but the general idea remains the same: W_{NC} indicates the amount of electrical energy required for the roll. It follows that when comparing locomotion of the two hypothetical trial rolls, the roll that registers a smaller magnitude of non-conservative work per unit roll distance also pulls less energy per roll distance from the batteries. Clearly, then, that roll is more economical. According to Equation 4.6, this statement is equivalent to saying the control combination corresponding to a smaller value of ϱ is superior in terms of energy economy.

Recall from Section 4.2 that after a trial roll is finished, the robot is programmed to report locomotion-related parameters for every 100 ms (every tenth of a second) of the 15 second trial roll to a laptop computer. Reported parameters from each trial roll in the primary experiment are saved in a digital spreadsheet as 151 successive rows containing values of θ, ω, S, a, ω_r and total energy consumed by the servo motor. To facilitate calculation of ϱ each trial roll, two roll time points are chosen from the spreadsheet. Point 1 is chosen from the beginning of the trial roll after the robot has achieved steady state velocity and when the robot is in, or is very close to, the circular configuration (with $a = 0.3335$ m). Typically, Point 1 is about 4.5 seconds into the roll, depending on the desired velocity of the trial roll. Point 2 is chosen from points at the end of the roll after the robot has reached S_m and settled-in to the circular configuration, as directed by the control program. Typically, Point 2 is about 14 seconds into the trial roll. Even though selection of Points 1 and 2 vary from roll to roll, they almost always envelop about nine seconds of steady state velocity of the robot.

After Points 1 and 2 have been chosen, the work energy theorem [40] is applied to the trial roll using these two reference points, by which the following energy conservation equation is derived:

$$E_m + W_{NC} = \Delta K + \Delta U \tag{4.7}$$

where ΔK and ΔU are changes in kinetic and potential energy, respectively, of the robot from Point 1 to Point 2, and E_m is total energy supplied to the servo motor between Points 1 and 2. Here, W_{NC} is non-conservative work performed on the linear actuator, servo motor, and outer surface of the robot. It does not include energy consumed by the microprocessor and sensors, because energy into these components is not part of E_m, and energy out of these components does not affect mechanics of the robot. E_m is calculated by subtracting total energy used by the servo motor at Point 1 from total energy used at Point 2:

$$E_m = E_2 - E_1 \tag{4.8}$$

Change in kinetic energy, ΔK, in Equation 4.7 is calculated according to the assumption that the robot is a rigid body. As such, kinetic energy is calculated according to angular velocity of robot axis, A, as well as linear velocity, v, of robot center of mass. Change in kinetic energy from Point 1 to Point 2 is therefore expressed as

$$\Delta K = \frac{1}{2}Mv_2^2 + \frac{1}{2}I\omega_2^2 - \frac{1}{2}Mv_1^2 - \frac{1}{2}I\omega_1^2 \tag{4.9}$$

where I is mass moment of inertia of the robot in the circular configuration, and v and ω are subscripted to signify velocities at Points 1 and 2. Because the robot is in, or nearly in, the circular configuration at Points 1 and 2, height of robot center of mass is equal to $h - d$, where h is radius of the unloaded outer surface cylinder in the circular configuration, and d is a small change in height of robot center of mass due to bending of the outer surface (see Section 3.5.1). Treating $h - d$ as the radius of the rolling robot, the arc length formula is applied in order to calculate velocity of robot center of mass at Points 1 and 2, resulting in: $v_1 = (h - d_1)\omega_1$ and $v_2 = (h - d_2)\omega_2$, where d_1 and d_2 are changes in height of robot center of mass due to bending at Points 1 and 2, respectively. Substituting for v_1 and v_2, Equation 4.9 becomes

$$\Delta K = \frac{M}{2}[(h - d_2)^2\omega_2^2 - (h - d_1)^2\omega_1^2] + \frac{I}{2}(\omega_2^2 - \omega_1^2) \tag{4.10}$$

As an aside, it is possible that at Point 1 the robot is in the midst of changing shape as it rolls. In this case, the robot is not a solid body, and the actual linear velocity of robot center of mass is slightly different from v_1 in Equation 4.9 due to the potentially additional velocity component of robot center of mass during actuation. However, the kinetic energy potentially brought on by this unaccounted for velocity component is tiny compared to the entire kinetic energy of the rolling robot, and so it is not included in Equation 4.9.

Potential energy of the robot at any point is equal to gravity potential of robot center of mass, plus potential stored in deflection of the flexible outer surface. For the purpose of calculating these values, an imaginary reference datum is placed at the surface of the roll track, and the outer surface is treated as a linear spring with stiffness, k_s. Accordingly, potential energy of the robot at Point 1 is

$$U_1 = mg(h - d_1) + \frac{1}{2}k_s d_1^2 \tag{4.11}$$

where the quantity on the left of the addition sign is gravity potential of robot center of mass, and the quantity on the right of the addition sign is outer surface spring potential. Recall from Section 3.5.1 that d is a function of robot orientation, a condition resulting from the linear actuator running along the diameter of the outer surface that acts as a stiff beam that firmly holds the outer surface and restricts its bending. Consequently, when the robot is oriented such that the linear actuator is nearly horizontal, the outer surface bends considerably. In this case, the outer surface behaves as a spring with a relatively small spring constant. In contrast, when the actuator is nearly vertical, the outer surface bends only a tiny amount. In this case, the outer surface behaves as a very stiff spring. To quantify this variable energy storage capacity of the outer surface, stiffness k_s is modeled as being equal to robot weight divided by deflection of robot center of mass:

$$k_s = \frac{Mg}{d} \tag{4.12}$$

Substituting Equation 4.12 into Equation 4.11 and simplifying gives the following expression for potential energy of the robot at Point 1:

$$U_1 = Mg(h - d_1/2) \tag{4.13}$$

Likewise, potential energy at Point 2 is

$$U_2 = Mg(h - d_2/2) \tag{4.14}$$

and the change in potential energy between Points 1 and 2 is

$$\Delta U = U_2 - U_1 = \frac{Mg}{2}(d_1 - d_2) \tag{4.15}$$

As defined previously, ϱ is equal to the negative of non-conservative work, W_{NC}, performed between Points 1 and 2 divided by the distance, $\Delta S = S_2 - S_1$, traveled between Points 1 and 2. Combining this definition with Equation 4.7, 4.10 and 4.15 gives

$$\begin{multline}\varrho = -\frac{Mg}{2\Delta S}(d_1 - d_2) - \frac{M}{2\Delta S}[(h - d_2)^2\omega_2^2 - (h - d_1)^2\omega_1^2] \\ - \frac{I}{2\Delta S}(\omega_2^2 - \omega_1^2) + \frac{E_m}{\Delta S} \tag{4.16}\end{multline}$$

4.6.4 Uncertainty in ϱ

It is possible that the value of ϱ calculated by Equation 4.16 for a given trial roll is slightly different from the actual value of work per unit distance between Points 1 and 2. The magnitude of the possible difference is the uncertainty associated with ϱ, and it is brought about by imperfect measurement and imperfect modeling of the parameters that go into calculating ϱ, which according to Equation 4.16 are: M, I, h, ω_1, ω_2, d_1, d_2, ΔS and E_m. Concerning measurements of the first two, M is obtained with a scale, and I is found using the swinging pendulum method [56]; and no significant difference is found in I when semi-major axis of the outer surface is changed from $a = 0.3335$ m to $a = 0.3490$ m, so that I can safely be considered a constant, regardless of outer surface eccentricity. Because M and I are carefully measured, they are deemed highly accurate and therefore essentially noncontributing to uncertainty in ϱ. In addition, it will be shown in this section that uncertainty in measurement of ΔS is small enough to be considered noncontributing. Each of the six remaining parameters, however, has significant measurement uncertainty and therefore contributes to uncertainty in ϱ. With these parameters denoted as p_{1-6}, the square-root sum method [57] for propagation of uncertainty gives the following expression for uncertainty in ϱ:

$$\delta\varrho = \sqrt{(\frac{\partial\varrho}{\partial p_1}\delta p_1)^2 + (\frac{\partial\varrho}{\partial p_2}\delta p_2)^2 + ... + (\frac{\partial\varrho}{\partial p_6}\delta p_6)^2} \tag{4.17}$$

where $\delta p_1 - \delta p_6$ are uncertainties associated with $p_1 - p_6$, respectively, and they are given or derived in the following paragraphs of this section. The six partial derivatives in Equation 4.17 are derived and given as follows:

$$\frac{\delta\varrho}{\delta h} = \frac{M}{\Delta S}(h - d_1)\omega_1^2 - \frac{M}{\Delta S}(h - d_2)\omega_2^2 \tag{4.18}$$

$$\frac{\delta\varrho}{\delta\omega_1} = \frac{M}{\Delta S}(h_1 - d_1)^2\omega_1 + \frac{I}{\Delta S}\omega_1 \tag{4.19}$$

$$\frac{\delta\varrho}{\delta\omega_2} = -\frac{M}{\Delta S}(h_2 - d_2)^2\omega_2 - \frac{I}{\Delta S}\omega_2 \tag{4.20}$$

$$\frac{\delta\varrho}{\delta d_1} = -\frac{Mg}{2\Delta S} - \frac{M}{\Delta S}(h_1 - d_1)\omega_1^2 \tag{4.21}$$

$$\frac{\delta\varrho}{\delta d_2} = \frac{Mg}{2\Delta S} + \frac{M}{\Delta S}(h_2 - d_2)\omega_2^2 \tag{4.22}$$

$$\frac{\delta\varrho}{\delta E_m} = \frac{1}{\Delta S} \tag{4.23}$$

Uncertainties in h_1, h_2, d_1 and d_2

In Section 3.1, center of mass of the robot is said to be approximately located at the axis of the outer surface cylinder of the robot when it is unloaded. This approximation, by which the height, h, of robot center of mass above the roll surface is equated to 0.3335 m for computing energy of the robot in the circular configuration (Equation 4.10, Equation 4.11, Equation 4.13, and Equation 4.16), is estimated to have a margin of error of ±0.005 m due to difficulty in placing and confirming the exact center of gravity of the robot. In other words, the center of mass of the robot could be located up to a distance of 0.005 m in any direction from its presumed position, possibly making h smaller or larger by the same magnitude. In addition, due to imperfect modeling of a as described in Section 3.7.2, it's possible that at Points 1 and 2, the robot might not be in the circular configuration, even though it could be reported as such. This latter discrepancy leads to the actual center of mass being located up to a distance of 0.0026 m relative to its presumed position. In total then, the center of mass of the robot could be off by as much as 0.005 m + 0.0026 m = 0.0076 m in either direction along A from its presumed position. The expression for uncertainty in h is therefore given by: $\delta h_1 = \delta h_2 = 0.0076$ m.

Uncertainty in d is due to error in the vertical bending model, given by Equation 3.5, applied to predict it. In Section 3.5.1, predicted values of d obtained using Equation 3.5 are compared to actual static bending measurements for orientations of the robot spanning $0° - 360°$. Difference between predicted values and actual measurements is shown to be no greater than 2.0 millimeters in magnitude. This maximum value is used as uncertainty in d, so that $\delta d_1 = \delta d_2 = 0.0020$ m.

Uncertainties in ω_1, ω_2 and ΔS

Uncertainty in angular velocity is derived by performing a test, as described in Section 4.2, in which the largest possible discrepancy is found between the computed roll angle using gyroscope measurements and roll angle as measured by the mechanical switches at every half turn of the robot. Differences in these two values of roll angle provide a reference by which uncertainty for ω is derived. This uncertainty is explained in Section 4.2, and it is restated here as: $\delta\omega_1 = \delta\omega_1 = 0.0578$ rad/s.

In order to determine uncertainty in roll distance, S, a test is conducted in which measured values of S are compared to actual values. For this test, several rolls of the robot are performed, in which robot initial position is accurately established by starting each roll from a marked position on the track. At a distance down the track, a bumper is placed on the roll surface, so that when the robot makes contact with the bumper, it has rolled a distance of 8.50 m. At or just after contact, the reported angular velocity of the robot sharply drops, and at this report time, roll distance of the robot as computed by the control program using Equation 4.5 is saved to a laptop computer. Five such rolls are performed in this manner for $\omega_d = 2.4$ rad/s, and for each roll the computed value of S is compared to the actual roll distance, 8.50 m. Results of the test reveal that reported roll distance of the robot is always greater than the actual roll distance, yet it is never greater by more than 2.5 cm. In terms of error per unit roll distance, this discrepancy is equivalent to 0.025 m/8.5 m = 0.0029 meters of error per meter rolled.

Because the computed value of roll distance in the test is always greater than the actual value, it is deduced that the discrepancy is due to slippage of the outer surface against the surface of the roll track. Assuming the slippage occurs evenly about the distance of the track, it follows that the amount of error in the measurement of ΔS reported by the control program for two points on the roll track separated by six meters – at 9.70 m and 3.70 m, for instance – would never be more than: $[9.70(0.0029+1) - 3.70(0.0029+1) - 6.00]$ m = 0.017 m. This uncertainty for ΔS, if it were included in Equation 4.17, would add only a tiny amount to $\delta\varrho$ in comparison to uncertainty from other parameters, and it is therefore treated as noncontributing.

Uncertainty in E_m

As described in Section 3.4.3, the energy sensor samples current and voltage at the servo motor, multiplies them to obtain power consumed by the servo motor, and then integrates power over time to give total energy used by the servo motor. Hence, δE_m is a consequence of uncertainties in current and voltage measurements as well as propagation of these uncertainties when using the measurements to compute power and energy consumed by the servo motor from Point 1 to Point 2. Uncertainty in energy consumed could be computed onboard the energy sensor during each experimental trial roll, but this is not the preferred approach, and the reason has to do with the computation load of the sensing program; in order to achieve the most accurate computation of energy possible, sample period of the energy sensor program is made to be as small as possible, and consequently, the number of computations the sensor program is allowed to perform is strictly limited. For this reason, instead of performing uncertainty-related computations in the sensing program, δE_m is calculated after trial rolls are conducted, using information that is reported by the robot to the laboratory laptop computer and saved.

This post-roll calculation of δP is a simplification strategy in which the concept of average power is invoked to approximate uncertainty in the reported value of E_m. The strategy relies on the assumption that there is no significant error introduced by the integration process used to compute energy in the sensor program (Section 3.4.3). Thus, if t_{1-2} is given as roll duration between Points 1 and 2, energy used by the servo motor is approximated by applying the definition of average power [40]:

$$E_m = \overline{P} t_{1-2} \tag{4.24}$$

and applying the square-root sum method [57] to this equation for propagation of uncertainty gives

$$\delta E_m = (\delta \overline{P}) t_{1-2} \tag{4.25}$$

According to the power transfer equation [40], power of any device is equal to the current going into the device multiplied by potential across the device's terminals.

For the servo motor, then, average power is

$$\overline{P} = (\overline{i_m})(\overline{V_m}) \tag{4.26}$$

where $\overline{i_m}$ and $\overline{V_m}$ are average current and voltage, respectively, for the servo motor. By monitoring servo motor voltage with an oscilloscope during actuation of the robot, V_m is found to be 4.96 volts on average with little overall change, allowing Equation 4.26 to be rewritten as

$$\overline{P} = 4.96(\overline{i_m}) \tag{4.27}$$

and

$$\delta\overline{P} = 4.96(\delta\overline{i_m}) \tag{4.28}$$

Servo motor current, i_m is a function of $MAX4071$ output voltage, V_{out} [33]:

$$i_m = \frac{V_{out} - 1.5}{50R} \tag{4.29}$$

where R is ohmage of the sense resistor placed in series between the motor power supply and the motor load (Fig. 3.9). Applying the square-root sum method [57] to Equation 4.29 for propagation of uncertainty gives

$$\delta i_m = \sqrt{(\frac{\partial i_m}{\partial V_{out}}\delta V_{out})^2 + (\frac{\partial i_m}{\partial R}\delta R)^2} \tag{4.30}$$

where δV_{out} is uncertainty in output voltage of the $MAX4071$ chip, and δR is uncertainty of the sense resistor ohmage. Although δR is non-zero, it is tiny compared to δV_{out}, and it can therefore be ignored, whereby Equation 4.30 simplifies to

$$\delta i_m = \frac{1}{50R}\delta V_{out} \tag{4.31}$$

For the configuration of the $MAX4071$ chip as applied on the energy sensor, uncertainty in output voltage is due to imperfect chip performance and is no greater than 1% [33]. Additional uncertainty of 0.00781 volts is introduced to by possibility of imperfect analog-to-digital conversion [58]. Total uncertainty in output voltage of the $MAX4071$ is therefore

$$\delta V_{out} = \frac{V_{out}}{100} + 0.00781 \tag{4.32}$$

and this equation is combined with Equation 4.31 to give

$$\delta i_m = \frac{1}{50R}(\frac{V_{out}}{100} + 0.00781) \tag{4.33}$$

From Section 3.4.3, $R = 0.10\ \Omega$, so that uncertainty in motor current is

$$\delta i_m = \frac{V_{out}}{500} + 0.00156 \tag{4.34}$$

It is assumed that Equation 4.34 provides a good approximation of the uncertainty in average current when V_{out} is replaced therein by $\overline{V_{out}}$, which is the average value of V_{out} over t_{1-2}. As such,

$$\delta \overline{i_m} = \frac{\overline{V_{out}}}{500} + 0.00156 \tag{4.35}$$

In the same vein, it is assumed that Equation 4.29 provides good approximation of average current when V_{out} is replaced by $\overline{V_{out}}$, so that

$$\overline{i_m} = \frac{\overline{V_{out}} - 1.5}{5} \tag{4.36}$$

Combining these two equations results in

$$\delta \overline{i_m} = \frac{5\overline{i_m} + 1.5}{500} + 0.00156 \tag{4.37}$$

and combining this equation with Equation 4.27 and 4.28 gives

$$\delta\overline{P} = 0.0100\overline{P} + 0.00774 \tag{4.38}$$

Finally, Equations 4.24 and 4.25 are combined with Equation 4.38 to give

$$\delta E_m = 0.0100E_m + 0.0226t_{1-2} \tag{4.39}$$

CHAPTER 5

Results

This chapter presents results of the primary experiment described in Section 4.6. The chapter is divided into three sections. The first and second sections present results of Phase 1 and Phase 2 of the primary experiment, and the third section presents results from running the simulation program described in Section 3.5. In the third section, simulated parameters are compared to parameters that are measured/computed during trial rolls of the robot.

5.1 Phase 1

This section presents results of Phase 1 problems that are not disregarded for being unviable. Results for each problem are fully contained on a singular page, and each page begins with a problem summary that includes desired angular velocity, trigger angle, and values of k_f and k_s that define the problem. Following each problem summary is a table (Tables 5.1 through 5.10) that lists ϱ and $\delta\varrho$ for the trial rolls conducted in the problem. The last row of this table gives $\overline{\varrho}$ and the maximum value of $\delta\varrho$ for each ϱ-population. Finally, comparison grids as described in Section 4.6.1 are presented in Fig. 5.1 through Fig. 5.10. Results from Problem 1.13, although it is a viable problem, are not included in this section, because only one value of k_f for Problem 1.13 results in stability. Instead, results of Problem 1.13 are included in Section 5.2 as part of Problem 2.3 results.

Problem 1.2

$\omega_d = -2.0$ rad/s
$\theta_t = 35°$
C1: $k_f = 0.010$, $k_s = 1.27$
C2: $k_f = 0.020$, $k_s = 1.18$
C3: $k_f = 0.030$, $k_s = 1.12$
C4: $k_f = 0.040$, $k_s = 1.08$

Table 5.1: ϱ, $\delta\varrho$ for trial rolls of Problem 1.2.

	P1	P2	P3	P4
	$2.777, 0.065$	$2.494, 0.061$	$2.476, 0.060$	$2.466, 0.062$
	$2.731, 0.065$	$2.593, 0.061$	$2.860, 0.064$	$2.702, 0.063$
	$3.044, 0.068$	$3.092, 0.066$	$3.076, 0.066$	$2.671, 0.064$
	$2.896, 0, 066$	$3.057, 0.066$	$3.125, 0.066$	$3.150, 0.068$
	$3.057, 0.068$	$2.831, 0.063$	$3.211, 0.068$	$2.919, 0.065$
	$2.527, 0.062$	$2.781, 0.063$	$2.697, 0.062$	$3.002, 0.066$
	$2.779, 0.066$	$2.478, 0.060$	$2.917, 0.065$	$2.497, 0.060$
	$2.884, 0.066$	$2.836, 0.063$	$2.779, 0.063$	$2.646, 0.063$
	$2.901, 0.067$	$2.732, 0.063$	$2.846, 0.064$	$2.687, 0.064$
	$2.786, 0.065$	$2.755, 0.063$	$2.802, 0.064$	$2.565, 0.062$
$\overline{\varrho}$:	2.838	2.765	2.879	2.730
max. $\delta\varrho$:	0.068	0.066	0.068	0.068

	C1	C2	C3	C4
C1		C2, 3%	C3, 0%	C4, 100%
C2	C2, 3%		C2, 24%	C4, 76%
C3	C3, 0%	C2, 24%		C4, 100%
C4	C4, 100%	C4, 76%	C4, 100%	

	~~C1~~	C2	~~C3~~	(C4)
C1		-	-	C4
C2	-		-	-
C3	-	-		C4
C4	C4	-	C4	

Figure 5.1: Pairwise comparison grids for Problem 1.2.

Problem 1.3
$\omega_d = -2.0$ rad/s
$\theta_t = 45°$
C1: $k_f = 0.010$, $k_s = 1.22$
C2: $k_f = 0.020$, $k_s = 1.14$
C3: $k_f = 0.030$, $k_s = 1.09$
C4: $k_f = 0.040$, $k_s = 1.07$

Table 5.2: ϱ, $\delta\varrho$ for trial rolls of Problem 1.3.

	P1	P2	P3	P4
	$2.431, 0.061$	$2.348, 0.059$	$2.443, 0.061$	$2.327, 0.060$
	$2.501, 0.062$	$2.298, 0.058$	$2.616, 0.062$	$2.592, 0.062$
	$2.597, 0.063$	$2.579, 0.061$	$2.670, 0.062$	$2.693, 0.063$
	$2.778, 0.065$	$2.861, 0.065$	$2.526, 0.061$	$2.718, 0.063$
	$3.065, 0.068$	$2.783, 0.063$	$2.868, 0.064$	$2.710, 0.063$
	$2.907, 0.067$	$2.867, 0.065$	$2.759, 0.064$	$2.854, 0.065$
	$2.926, 0.067$	$2.634, 0.063$	$2.526, 0.062$	$2.567, 0.062$
	$2.975, 0.067$	$2.836, 0.064$	$2.754, 0.064$	$2.605, 0.063$
	$3.076, 0.068$	$3.005, 0.067$	$2.941, 0.065$	$2.770, 0.064$
	$2.364, 0.062$	$2.999, 0.066$	$3.059, 0.067$	$2.974, 0.066$
$\overline{\varrho}$:	2.762	2.721	2.716	2.681
max $\delta\varrho$:	0.068	0.067	0.067	0.066

	C1	C2	C3	C4
C1		0%	0%	0%
C2	0%		0%	C4, 0%
C3	0%	0%		0%
C4	0%	C4, 0%	0%	

	C1	C2	C3	(C4)
C1		-	-	-
C2	-		-	-
C3	-	-		-
C4	-	-	-	

Figure 5.2: Pairwise comparison grids for Problem 1.3.

Problem 1.4

$\omega_d = -2.0$ rad/s
$\theta_t = 55^\circ$
C1: $k_f = 0.010$, $k_s = 1.21$
C2: $k_f = 0.020$, $k_s = 1.13$
C3: $k_f = 0.030$, $k_s = 1.09$
C4: $k_f = 0.040$, $k_s = 1.07$

Table 5.3: ϱ, $\delta\varrho$ for trial rolls of Problem 1.4.

	P1	P2	P3	P4
	$2.843, 0.066$	$3.146, 0.068$	$3.263, 0.068$	$3.243, 0.073$
	$3.105, 0.068$	$2.617, 0.057$	$3.300, 0.068$	$3.331, 0.069$
	$2.939, 0.067$	$3.020, 0.066$	$3.239, 0.068$	$3.321, 0.069$
	$3.129, 0.069$	$2.805, 0.064$	$3.143, 0.067$	$3.213, 0.068$
	$2.873, 0.066$	$2.738, 0.063$	$3.054, 0.066$	$3.190, 0.067$
	$2.287, 0.061$	$2.355, 0.060$	$2.613, 0.063$	$2.599, 0.063$
	$2.348, 0.061$	$2.512, 0.062$	$2.841, 0.066$	$2.710, 0.064$
	$2.464, 0.064$	$2.537, 0.063$	$2.797, 0.065$	$2.697, 0.064$
	$2.230, 0.061$	$2.357, 0.061$	$2.718, 0.070$	$2.648, 0.064$
	$2.324, 0.062$	$2.410, 0.061$	$2.779, 0.065$	$2.711, 0.066$
$\overline{\varrho}$:	2.654	2.650	2.975	2.966
max $\delta\varrho$:	0.069	0.068	0.070	0.073

	C1	C2	C3	C4
C1	▨	0%	C3, 0%	0%
C2	0%	▨	C2, 99%	C2, 13%
C3	C3, 0%	C2, 99%	▨	0%
C4	0%	C2, 13%	0%	▨

	C1	(C2)	~~C3~~	C4
C1	▨	-	-	-
C2	-	▨	C2	-
C3	-	C2	▨	-
C4	-	-	-	▨

Figure 5.3: Pairwise comparison grids for Problem 1.4.

Problem 1.5
$\omega_d = -2.0$ rad/s
$\theta_t = 65°$
C1: $k_f = 0.010$, $k_s = 1.21$
C2: $k_f = 0.020$, $k_s = 1.12$
C3: $k_f = 0.030$, $k_s = 1.11$

Table 5.4: ϱ, $\delta\varrho$ for trial rolls of Problem 1.5.

	P_1	P_2	P_3
	$3.145, 0.069$	$3.483, 0.071$	$3.743, 0.074$
	$3.212, 0.070$	$3.494, 0.072$	$4.109, 0.078$
	$3.218, 0.069$	$3.157, 0.068$	$4.711, 0.085$
	$3.429, 0.073$	$3.623, 0.073$	$4.339, 0.080$
	$3.259, 0.070$	$3.698, 0.074$	$5.240, 0.092$
	$3.222, 0.070$	$3.531, 0.070$	$3.711, 0.077$
	$3.278, 0.070$	$3.290, 0.067$	$3.869, 0.076$
	$3.244, 0.069$	$3.157, 0.068$	$4.107, 0.081$
	$3.311, 0.071$	$3.567, 0.070$	$4.304, 0.082$
	$3.490, 0.068$	$3.598, 0.070$	$4.799, 0.091$
$\overline{\varrho}$:	3.281	3.460	4.293
max $\delta\varrho$:	0.073	0.074	0.092

	C1	C2	C3
C1	▨	C1, 100%	C1, 100%
C2	C1, 100%	▨	C2, 100%
C3	C1, 100%	C2, 100%	▨

	(C1)	~~C2~~	~~C3~~
C1	▨	C1	C1
C2	C1	▨	C2
C3	C1	C2	▨

Figure 5.4: Pairwise comparison grids for Problem 1.5.

Problem 1.6
$\omega_d = -2.2$ rad/s
$\theta_t = 25°$
C1: $k_f = 0.020$, $k_s = 1.16$
C2: $k_f = 0.030$, $k_s = 1.10$
C3: $k_f = 0.040$, $k_s = 1.09$

Table 5.5: ϱ, $\delta\varrho$ for trial rolls of Problem 1.6.

	P1	P2	P3
	$2.428, 0.057$	$2.399, 0.057$	$2.234, 0.057$
	$2.429, 0.057$	$2.461, 0.058$	$2.429, 0.057$
	$2.418, 0.057$	$2.421, 0.058$	$2.323, 0.056$
	$2.383, 0.057$	$2.433, 0.058$	$2.396, 0.056$
	$2.286, 0.056$	$2.052, 0.054$	$2.243, 0.056$
	$2.467, 0.057$	$2.365, 0.057$	$2.229, 0.056$
	$2.655, 0.060$	$2.276, 0.056$	$2.385, 0.058$
	$2.364, 0.057$	$2.230, 0.056$	$2.186, 0.051$
	$2.285, 0.056$	$2.257, 0.057$	$2.441, 0.058$
	$2.404, 0.057$	$2.280, 0.056$	$2.226, 0.056$
$\overline{\varrho}$:	2.412	2.317	2.309
max $\delta\varrho$:	0.060	0.058	0.058

	C1	C2	C3
C1	▨	C2, 24%	C3, 56%
C2	C2, 24%	▨	0%
C3	C3, 56%	0%	▨

	C1	(C2)	C3
C1	▨	-	-
C2	-	▨	-
C3	-	-	▨

Figure 5.5: Pairwise comparison grids for Problem 1.6.

Problem 1.7

$\omega_d = -2.2$ rad/s
$\theta_t = 35^\circ$
C1: $k_f = 0.010$, $k_s = 1.27$
C2: $k_f = 0.020$, $k_s = 1.15$
C3: $k_f = 0.030$, $k_s = 1.10$
C4: $k_f = 0.040$, $k_s = 1.09$

Table 5.6: ϱ, $\delta\varrho$ for trial rolls of Problem 1.7.

	P1	P2	P3	P4
	$2.111, 0.054$	$2.258, 0.055$	$2.183, 0.055$	$2.167, 0.055$
	$2.115, 0.054$	$2.148, 0.055$	$2.205, 0.055$	$2.334, 0.056$
	$2.143, 0.055$	$2.340, 0.056$	$2.224, 0.055$	$2.460, 0.058$
	$2.230, 0.055$	$2.301, 0.056$	$2.192, 0.055$	$2.340, 0.056$
	$2.236, 0.055$	$2.305, 0.056$	$2.321, 0.057$	$2.402, 0.057$
	$2.267, 0.056$	$2.275, 0.056$	$2.358, 0.061$	$2.497, 0.058$
	$2.356, 0.057$	$2.440, 0.058$	$2.455, 0.062$	$2.456, 0.058$
	$2.506, 0.059$	$2.437, 0.058$	$2.577, 0.059$	$2.512, 0.058$
	$2.372, 0.058$	$2.453, 0.058$	$2.495, 0.058$	$2.308, 0.056$
	$2.257, 0.051$	$2.392, 0.057$	$2.316, 0.056$	$2.450, 0.058$
$\overline{\varrho}$:	2.259	2.335	2.333	2.393
max $\delta\varrho$:	0.059	0.058	0.061	0.058

	C1	C2	C3	C4
C1	▨	C1, 84%	C1, 2%	C1, 100%
C2	C1, 84%	▨	C3, 0%	C2, 4%
C3	C1, 2%	C3, 0%	▨	C3, 32%
C4	C1, 100%	C2, 4%	C3, 32%	▨

	C1	(C2)	C3	~~C4~~
C1	▨	-	-	C1
C2	-	▨	-	-
C3	-	-	▨	-
C4	C1	-	-	▨

Figure 5.6: Pairwise comparison grids for Problem 1.7.

Problem 1.8
$\omega_d = -2.2$ rad/s
$\theta_t = 45°$
C1: $k_f = 0.010$, $k_s = 1.28$
C2: $k_f = 0.020$, $k_s = 1.16$
C3: $k_f = 0.030$, $k_s = 1.13$
C4: $k_f = 0.040$, $k_s = 1.11$

Table 5.7: ϱ, $\delta\varrho$ for trial rolls of Problem 1.8.

	P1	P2	P3	P4
	$2.348, 0.053$	$2.502, 0.057$	$2.693, 0.059$	$2.512, 0.059$
	$2.483, 0.058$	$2.536, 0.059$	$2.851, 0.062$	$2.813, 0.062$
	$2.492, 0.058$	$2.640, 0.060$	$2.859, 0.062$	$2.717, 0.061$
	$2.690, 0.061$	$2.683, 0.060$	$2.578, 0.059$	$2.678, 0.061$
	$2.444, 0.058$	$2.497, 0.059$	$2.644, 0.060$	$2.491, 0.059$
	$2.421, 0.058$	$2.414, 0.058$	$2.672, 0.059$	$2.491, 0.058$
	$2.381, 0.057$	$2.288, 0.056$	$2.653, 0.060$	$2.543, 0.059$
	$2.461, 0.058$	$2.618, 0.060$	$2.832, 0.059$	$2.626, 0.060$
	$2.542, 0.059$	$2.482, 0.058$	$2.794, 0.062$	$2.627, 0.060$
	$2.604, 0.060$	$2.667, 0.060$	$2.316, 0.061$	$2.616, 0.060$
$\overline{\varrho}$:	2.487	2.533	2.722	2.611
max $\delta\varrho$:	0.061	0.060	0.062	0.062

	C1	C2	C3	C4
C1	▨	C1, 39%	C1, 100%	C1, 98%
C2	C1, 39%	▨	C2, 99%	C2, 0%
C3	C1, 100%	C2, 99%	▨	C4, 85%
C4	C1, 98%	C2, 0%	C4, 85%	▨

	(C1)	C2	~~C3~~	~~C4~~
C1	▨	-	C1	C1
C2	-	▨	C2	-
C3	C1	C2	▨	-
C4	C1	-	-	▨

Figure 5.7: Pairwise comparison grids for Problem 1.8.

Problem 1.9

$\omega_d = -2.2$ rad/s
$\theta_t = 55°$
C1: $k_f = 0.020$, $k_s = 1.18$
C2: $k_f = 0.030$, $k_s = 1.16$

Table 5.8: ϱ, $\delta\varrho$ for trial rolls of Problem 1.9.

	P1	P2
	$2.881, 0.062$	$3.008, 0.063$
	$3.099, 0.065$	$3.216, 0.066$
	$3.142, 0.065$	$3.450, 0.068$
	$3.305, 0.067$	$3.563, 0.070$
	$3.135, 0.065$	$3.398, 0.067$
	$2.884, 0.062$	$3.114, 0.065$
	$3.260, 0.066$	$3.357, 0.067$
	$3.256, 0.066$	$3.106, 0.064$
	$3.094, 0.065$	$3.030, 0.064$
	$3.039, 0.064$	$2.959, 0.064$
$\overline{\varrho}$:	3.110	3.220
max $\delta\varrho$:	0.067	0.070

	C1	C2
C1	▨	C1, 1%
C2	C1, 1%	▨

	(C1)	C2
C1	▨	-
C2	-	▨

Figure 5.8: Pairwise comparison grids for Problem 1.9.

Problem 1.11
$\omega_d = -2.4$ rad/s
$\theta_t = 25°$
C1: $k_f = 0.010$, $k_s = 1.29$
C2: $k_f = 0.020$, $k_s = 1.18$
C3: $k_f = 0.030$, $k_s = 1.14$
C4: $k_f = 0.040$, $k_s = 1.11$

Table 5.9: ϱ, $\delta\varrho$ for trial rolls of Problem 1.11.

	P1	P2	P3	P4
	$2.881, 0.060$	$2.971, 0.060$	$2.828, 0.059$	$2.709, 0.058$
	$2.884, 0.060$	$2.779, 0.058$	$2.848, 0.059$	$2.692, 0.058$
	$2.750, 0.058$	$2.821, 0.059$	$2.844, 0.059$	$3.029, 0.061$
	$2.762, 0.059$	$3.175, 0.063$	$2.908, 0.060$	$2.964, 0.060$
	$2.877, 0.060$	$3.146, 0.061$	$2.974, 0.061$	$2.679, 0.057$
	$2.982, 0.061$	$3.048, 0.061$	$2.880, 0.060$	$3.153, 0.062$
	$2.922, 0.061$	$2.913, 0.060$	$2.995, 0.061$	$3.058, 0.061$
	$3.067, 0.062$	$3.197, 0.063$	$3.083, 0.062$	$2.987, 0.060$
	$3.110, 0.062$	$3.041, 0.061$	$3.064, 0.062$	$3.190, 0.063$
	$3.160, 0.063$	$3.239, 0.063$	$3.064, 0.062$	$2.865, 0.060$
$\overline{\varrho}$:	2.939	3.033	2.949	2.933
max $\delta\varrho$:	0.063	0.063	0.062	0.063

	C1	C2	C3	C4
C1	▨	C1, 36%	C3, 0%	C1, 0%
C2	C1, 36%	▨	C3, 15%	C4, 2%
C3	C3, 0%	C3, 15%	▨	0%
C4	C1, 0%	C4, 2%	0%	▨

	C1	C2	C3	(C4)
C1	▨	-	-	-
C2	-	▨	-	-
C3	-	-	▨	-
C4	-	-	-	▨

Figure 5.9: Pairwise comparison grids for Problem 1.11.

Problem 1.12

$\omega_d = -2.4$ rad/s
$\theta_t = 35°$
C1: $k_f = 0.030$, $k_s = 1.10$
C2: $k_f = 0.040$, $k_s = 1.09$

Table 5.10: ϱ, $\delta\varrho$ for trial rolls of Problem 1.12.

	P1	P2
	$2.641, 0.058$	$2.540, 0.056$
	$2.822, 0.059$	$2.618, 0.057$
	$2.849, 0.060$	$2.512, 0.056$
	$2.622, 0.057$	$2.757, 0.058$
	$2.745, 0.058$	$2.728, 0.058$
	$3.271, 0.064$	$3.147, 0.063$
	$3.044, 0.062$	$3.214, 0.064$
	$3.174, 0.063$	$3.192, 0.063$
	$3.045, 0.063$	$3.409, 0.066$
	$3.083, 0.062$	$3.492, 0.066$
$\overline{\varrho}$:	2.930	2.961
max $\delta\varrho$:	0.064	0.066

	C1	C2
C1	■	0%
C2	0%	■

	(C1)	C2
C1	■	-
C2	-	■

Figure 5.10: Pairwise comparison grids for Problem 1.12.

5.2 Phase 2

This section presents results of trial rolls for Problem 2.1 through Problem 2.3. Results of each problem are fully contained on a singular page, and each page begins with a problem summary that includes the desired angular velocity and combinations of θ_t, k_f and k_s tested in the problem. Following each problem summary is a table (Tables 5.11 through 5.13) that lists ϱ and $\delta\varrho$ for trial rolls conducted in the problem. The last row of this table lists $\overline{\varrho}$ and the maximum value of $\delta\varrho$ for each ϱ-population. Finally, pairwise comparison grids as described in Section 4.6.2 are presented in Fig. 5.11 through Fig. 5.13 for each trial roll conducted in the problem. At the end of the section, a summary of Phase 2 results is given in Fig. 5.14, in which superior values of θ_t are identified for the various values of ω_d.

Problem 2.1

$\omega_d = -2.0$ rad/s
$\theta_t = 35°$, $k_f = 0.020$, $k_s = 1.17$
$\theta_t = 45°$, $k_f = 0.020$, $k_s = 1.14$
$\theta_t = 55°$, $k_f = 0.020$, $k_s = 1.13$
$\theta_t = 65°$, $k_f = 0.020$, $k_s = 1.12$

Table 5.11: ϱ, $\delta\varrho$ for trial rolls of Problem 2.1

	P1 ($\theta_t = 35°$)	P2 ($\theta_t = 45°$)	P3 ($\theta_t = 55°$)	P4 ($\theta_t = 65°$)
	$2.404, 0.060$	$2.331, 0.060$	$2.320, 0.059$	$2.992, 0.066$
	$2.548, 0.062$	$2.348, 0.060$	$2.431, 0.060$	$2.733, 0.064$
	$2.610, 0.062$	$2.334, 0.059$	$2.490, 0.061$	$2.761, 0.064$
	$2.465, 0.061$	$2.450, 0.061$	$2.309, 0.059$	$2.564, 0.062$
	$2.662, 0.062$	$2.447, 0.061$	$2.394, 0.060$	$2.636, 0.063$
	$2.610, 0.062$	$2.499, 0.061$	$2.377, 0.060$	$2.855, 0.065$
	$2.825, 0.064$	$2.338, 0.059$	$2.631, 0.063$	$2.726, 0.065$
	$2.738, 0.063$	$2.485, 0.061$	$2.363, 0.060$	$2.861, 0.065$
	$2.675, 0.062$	$2.452, 0.061$	$2.588, 0.062$	$2.968, 0.067$
	$2.635, 0.062$	$2.420, 0.061$	$2.658, 0.063$	$2.900, 0.066$
$\overline{\varrho}$:	2.617	2.410	2.456	2.788
max $\delta\varrho$:	0.064	0.061	0.063	0.067

	35°	45°	55°	65°
35°		45°, 100%	55°, 49%	35°, 100%
45°	45°, 100%		45°, 21.3%	45°, 100%
55°	55°, 49%	45°, 21%		55°, 100%
65°	35°, 100%	45°, 100%	55°, 100%	

	~~35°~~	45°	55°	~~65°~~
35°		45°	-	35°
45°	45°		-	45°
55°	-	-		55°
65°	35°	45°	55°	

Figure 5.11: Pairwise comparison grids for Problem 2.1.

Problem 2.2

$\omega_d = -2.2$ rad/s
$\theta_t = 25°$: $k_f = 0.020$, $k_s = 1.18$
$\theta_t = 35°$: $k_f = 0.020$, $k_s = 1.15$
$\theta_t = 45°$: $k_f = 0.020$, $k_s = 1.16$
$\theta_t = 55°$: $k_f = 0.020$, $k_s = 1.18$

Table 5.12: ϱ, $\delta\varrho$ for trial rolls of Problem 2.2

	P1 ($\theta_t = 25°$)	P2 ($\theta_t = 35°$)	P3 ($\theta_t = 45°$)	P4 ($\theta_t = 55°$)
	$2.781, 0.060$	$2.425, 0.058$	$2.621, 0.059$	$3.214, 0.065$
	$2.806, 0.060$	$2.323, 0.056$	$2.479, 0.058$	$3.079, 0.064$
	$2.856, 0.061$	$2.465, 0.058$	$2.966, 0.063$	$3.561, 0.069$
	$2.740, 0.058$	$2.511, 0.058$	$2.979, 0.063$	$3.303, 0.066$
	$2.728, 0.060$	$2.863, 0.060$	$2.891, 0.061$	$3.423, 0.068$
	$2.899, 0.061$	$2.788, 0.061$	$2.655, 0.059$	$3.859, 0.073$
	$2.849, 0.061$	$2.743, 0.061$	$2.838, 0.062$	$3.937, 0.069$
	$3.019, 0.061$	$2.993, 0.064$	$3.189, 0.065$	$3.839, 0.072$
	$2.877, 0.061$	$2.937, 0.063$	$2.953, 0.062$	$3.819, 0.072$
	$2.984, 0.063$	$2.853, 0.061$	$3.072, 0.063$	$3.880, 0.073$
$\overline{\varrho}$:	2.854	2.690	2.864	3.591
max $\delta\varrho$:	0.063	0.064	0.065	0.073

	25°	35°	45°	55°
25°	▨	35°, 6%	25°, 5%	25°, 100%
35°	35°, 6%	▨	35°, 38%	35°, 100%
45°	25°, 5%	35°, 38%	▨	45°, 100%
55°	25°, 100%	35°, 100%	45°, 100%	▨

	25°	35°	45°	~~55°~~
25°	▨	-	-	25°
35°	-	▨	-	35°
45°	-	-	▨	45°
55°	25°	35°	45°	▨

Figure 5.12: Pairwise comparison grids for Problem 2.2.

Problem 2.3

$\omega_d = -2.4$ rad/s
$\theta_t = 25°$: $k_f = 0.040$, $k_s = 1.11$
$\theta_t = 35°$: $k_f = 0.040$, $k_s = 1.09$
$\theta_t = 45°$: $k_f = 0.040$, $k_s = 1.14$

Table 5.13: ϱ, $\delta\varrho$ for trial rolls of Problem 2.3

	P1 ($\theta_t = 25°$)	P2 ($\theta_t = 35°$)	P3 ($\theta_t = 45°$)
	$3.134, 0.064$	$3.017, 0.062$	$3.388, 0.066$
	$3.193, 0.064$	$3.113, 0.063$	$3.595, 0.068$
	$3.101, 0.063$	$3.067, 0.063$	$3.466, 0.067$
	$3.286, 0.065$	$3.221, 0.061$	$3.626, 0.068$
	$3.328, 0.065$	$3.370, 0.066$	$3.430, 0.066$
	$3.368, 0.065$	$3.235, 0.064$	$3.701, 0.069$
	$3.476, 0.067$	$3.309, 0.065$	$4.125, 0.074$
	$3.488, 0.067$	$3.462, 0.067$	$3.914, 0.072$
	$3.534, 0.067$	$3.319, 0.065$	$3.845, 0.070$
	$3.603, 0.069$	$3.453, 0.067$	$3.952, 0.072$
$\overline{\varrho}$:	3.351	3.257	3.704
max $\delta\varrho$:	0.069	0.067	0.074

	25°	35°	45°
25°	▨	35°, 6%	25°, 100%
35°	35°, 6%	▨	35°, 100%
45°	25°, 100%	35°, 100%	▨

	25°	35°	~~45~~
25°	▨	-	25°
35°	-	▨	35°
45°	25°	35°	▨

Figure 5.13: Pairwise comparison grids for Problem 2.3.

	25°	35°	45°	55°	65°
-2.0 rad/s	▨	☒	●	●	☒
-2.2 rad/s	●	●	●	☒	▨
-2.4 rad/s	●	●	☒	▨	▨

Figure 5.14: Summary of Phase 2 results. Cells with a dot represent superior values of θ_t at the given value of ω_d. Gray cells represent values of θ_t that are unstable, and cells with a cross-out represent values of θ_t that are inferior.

5.3 Comparison of Simulation Predictions to Trial Rolls

In this section, the simulation program as described in Section 3.8 is configured to perform three rolls of the robot. Each of the simulated rolls has a set of initial conditions and control constants that are identical to a trial roll conducted in Phase 2 of the primary experiment. Roll 1 is from Problem 2.1 with $\theta_t = 35°$, $\theta(0) = 14°$, and $\omega_d = -2.0$ rad/s. Roll 2 is from Problem 2.2 with $\theta_t = 45°$, $\theta(0) = -14°$, and $\omega_d = -2.2$ rad/s. Roll 3 is from Problem 2.3 with $\theta_t = 35°$, $\theta(0) = -7°$, and $\omega_d = -2.4$ rad/s. By design, the rolls have desired angular velocities and trigger angles whose collective values span the ranges of these parameters tested in the primary experiment.

The program is configured to output ω, S, and a at 10 millisecond intervals of simulated roll time for the simulated trial rolls. Output values of angular velocity, ω, from the simulated rolls are gathered and plotted along with ω from actual trial rolls, and the resulting comparison plots for Rolls 1-3 are included in Figures 5.15, 5.18, and 5.21. Similarly, values of roll distance, S, from the simulated rolls are plotted along with S from actual trial rolls, and the resulting comparison plots for Rolls 1-3 are included in Figures 5.16, 5.19, and 5.22. Finally, values of semi-major axis length, a, from the simulated rolls are plotted along with a from actual trial rolls, and the resulting comparison plots for Rolls 1-3 are included in Figures 5.17, 5.20, and 5.23.

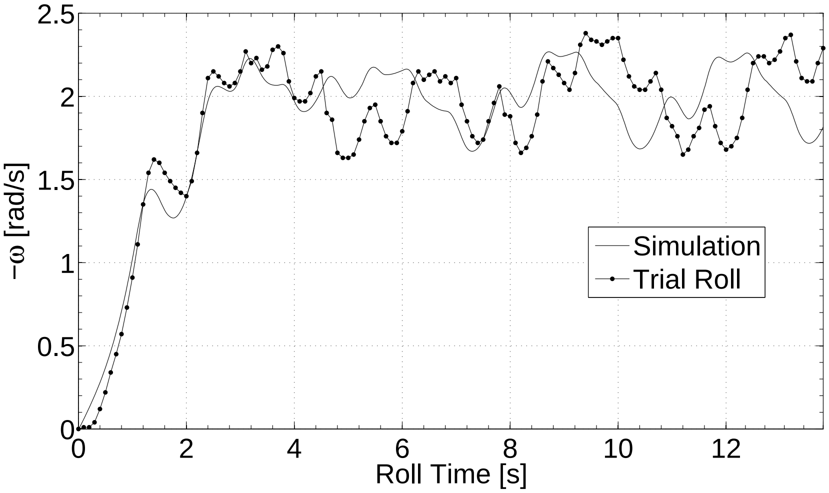


Figure 5.15: Robot angular velocity from simulation and from trial roll are plotted versus roll time for Roll 1, in which $\omega_d = -2.0$ rad/s.

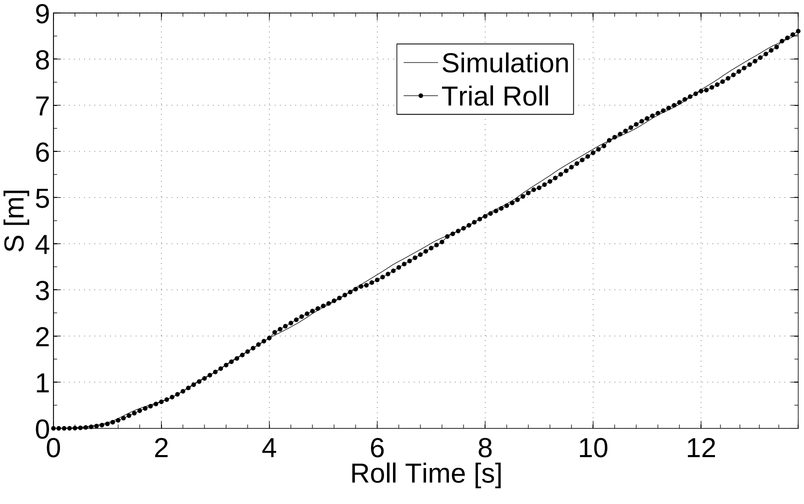


Figure 5.16: Touchpoint position from simulation and from trial roll are plotted versus roll time for Roll 1.

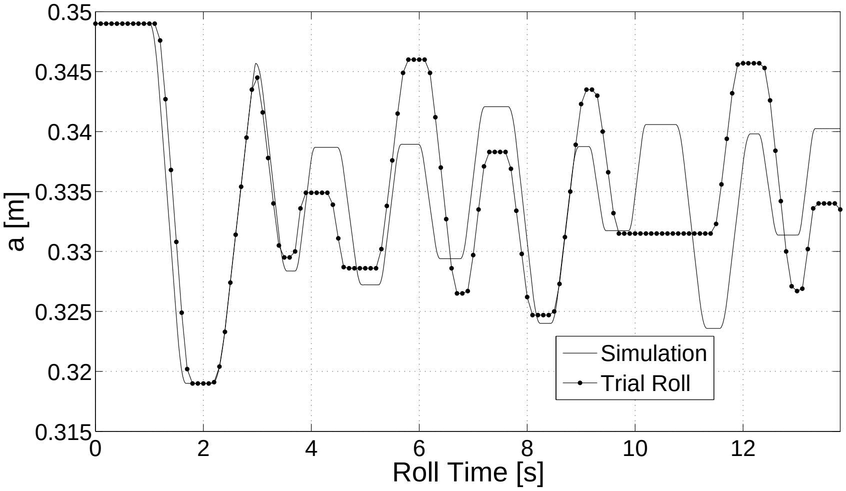


Figure 5.17: Semi-major axis length from simulation and from trial roll are plotted versus roll time for Roll 1.

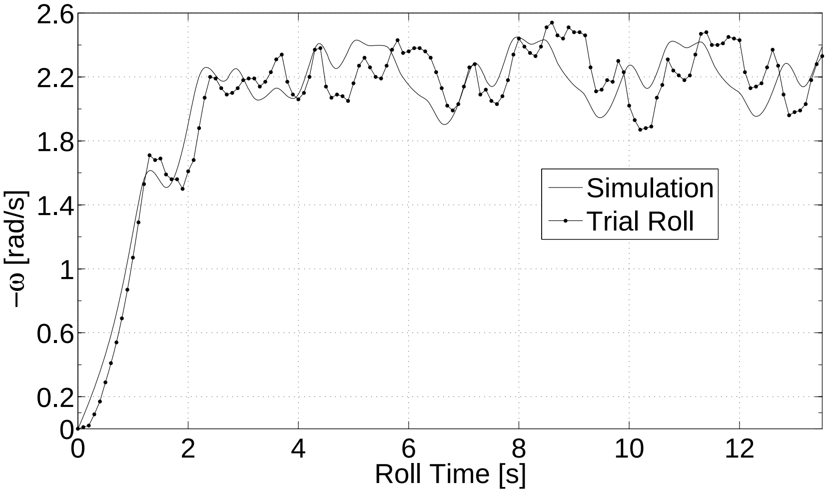


Figure 5.18: Robot angular velocity from simulation and from trial roll are plotted versus roll time for Roll 2, in which $\omega_d = -2.2$ rad/s.

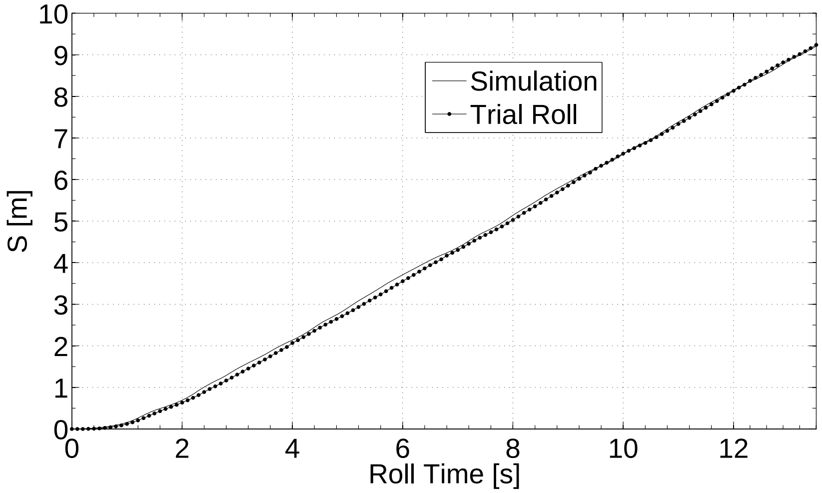


Figure 5.19: Touchpoint position from simulation and from trial roll are plotted versus roll time for Roll 2.

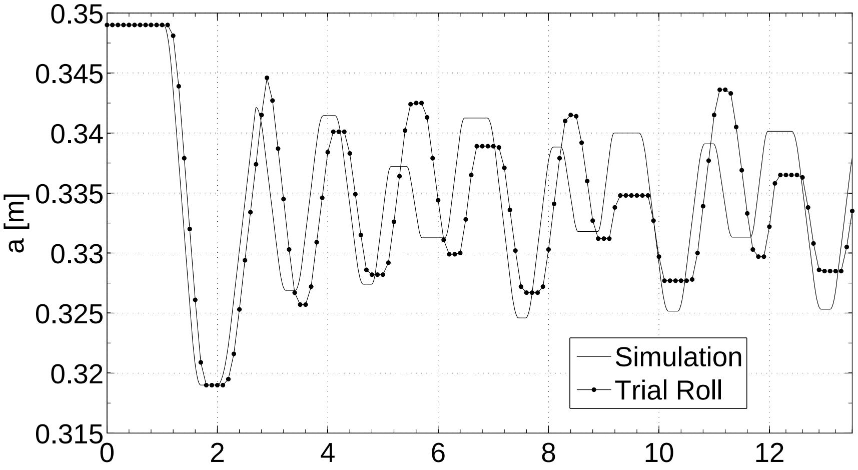


Figure 5.20: Semi-major axis length from simulation and from trial roll are plotted versus roll time for Roll 2.

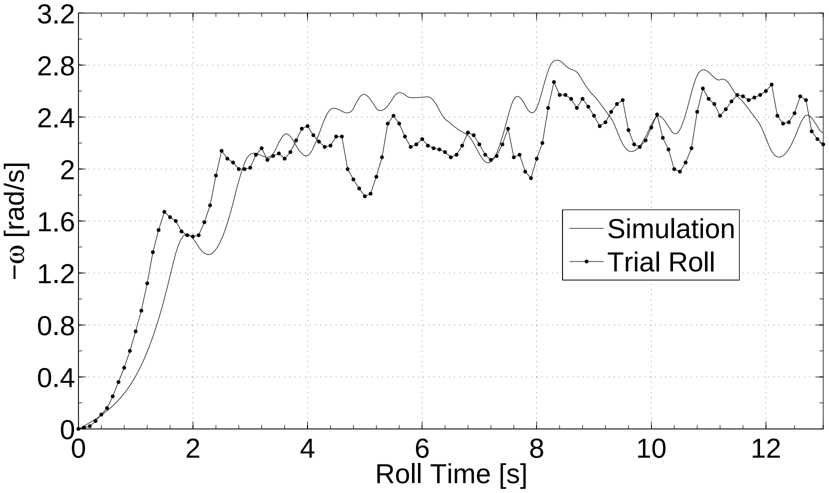


Figure 5.21: Robot angular velocity from simulation and from trial roll are plotted versus roll time for Roll 3, in which $\omega_d = -2.4$ rad/s.

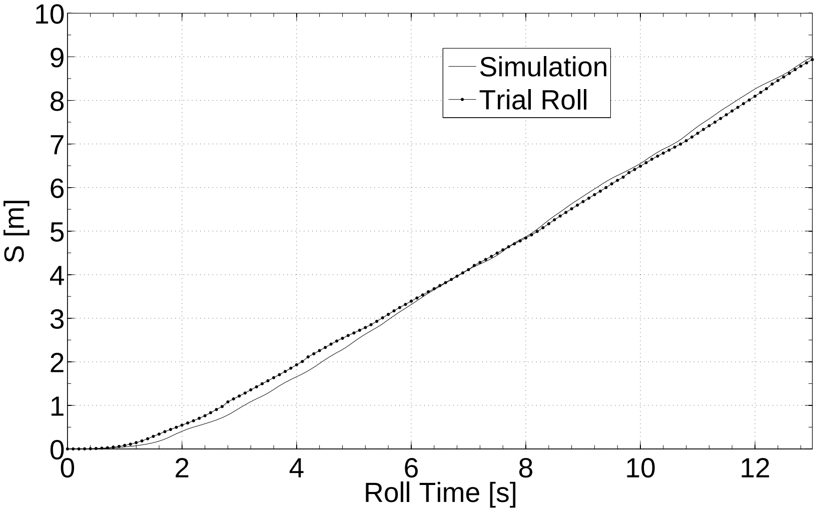


Figure 5.22: Touchpoint position from simulation and from trial roll are plotted versus roll time for Roll 3.

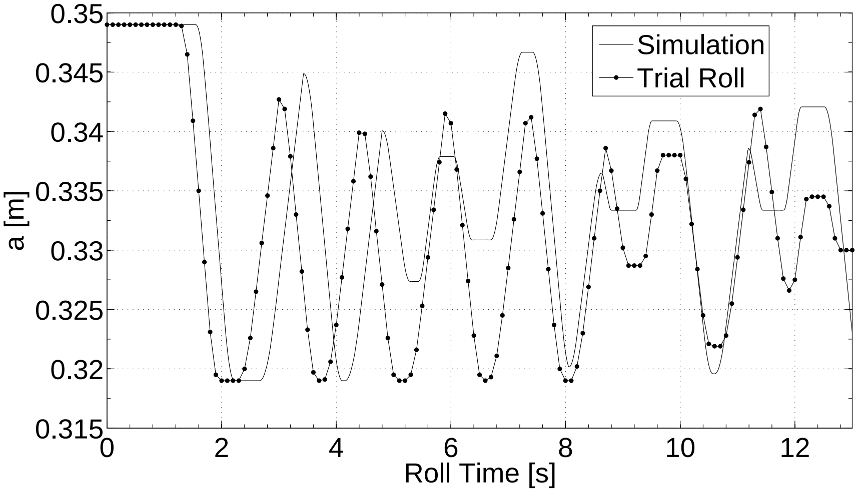


Figure 5.23: Semi-major axis length from simulation and from trial roll are plotted versus roll time for Roll 3.

CHAPTER 6

Discussion and Conclusion

This final chapter is divided into five sections. The first section is dedicated to explaining why a range of superior values of θ_t exists for each desired velocity, as seen in the results of the primary experiment in Section 5, and why this range shifts as velocity of the robot changes. In the second section in this chapter, two brief investigations are presented that take advantage of the prediction ability of the simulation program detailed in Section 3.8. In the third section, shortcomings of the simulation model are discussed. The fourth section provides suggestions for future work with the OSU rolling robot, or one like it, and the final section provides a conclusion for the research as a whole.

6.1 Explanation for the Changing Optimal Range of θ_t

One of the most interesting takeaways from the research documented herein is that for each ω_d tested in Phase 2 of the primary experiment, there exists a range of two or three values of θ_t among five tested that are determined to be superior in terms of energy economy of robot locomotion. When θ_t is decreased relative to this range, so that actuation is triggered with less tilt of the trigger axis with respect to the vertical, energy economy of robot locomotion is observed to decrease; and when θ_t is increased relative to this range, so that actuation is triggered with more tilt of the trigger axis, energy economy of robot locomotion is again observed to decrease. There is therefore an optimal range of θ_t with regard to energy economy for each ω_d tested in the primary experiment.

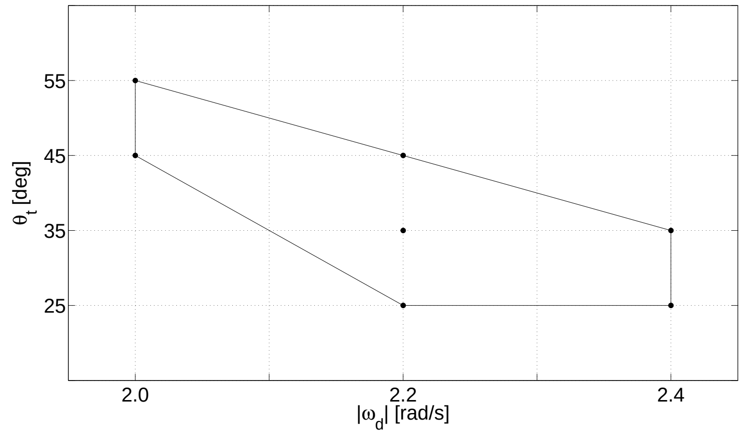


Figure 6.1: The region of optimality on the θ_t-ω_d plane has a slanted orientation, indicating that as the magnitude of ω_d is increased, the median range of optimal θ_t values decreases.

Furthermore, results of the primary experiment reveal that as desired angular velocity of the robot is changed, the optimal range of θ_t changes as well. In general, the optimal range of θ_t shifts downward as angular velocity magnitude of the robot is increased from 2.0 rad/s to 2.4 rad/s. To emphasize this downward shift, the seven superior values of θ_t represented by dots in Tab. 5.14 are plotted versus magnitude of ω_d in Fig. 6.1. In the plot, the seven values manifest as three sets of vertically aligned points that span the optimal range of θ_t for each ω_d tested in the primary experiment. Connecting the extremities of these ranges with line segments, as in Fig. 6.1, creates a polygonal region in which the optimal ranges of θ_t are determined or inferred to exist. The region has a slanted orientation, indicating that as ω_d is increased in magnitude, the median range of optimal values of θ_t decreases.

Recall from Section 3.6 that the control system repeatedly changes shape of the outer surface, and thereby changes location of the normal force relative to the robot center of mass. During stable locomotion of the robot, the control system ensures that the normal force is most often located to the left of the center of mass, causing input torque to be applied in a clockwise sense in order to drive the robot along X in the positive direction. By driving the robot in this manner, the control system causes the robot to follow, on average, a desired angular velocity. However, upon careful scrutiny of controlled motion of the robot, it is apparent that the control system has mixed consequences – that is, the control system speeds up the robot, but it sometimes slows it down, too. In fact, within ninety degrees of rotation of the robot, the control system often brings about both angular acceleration and angular deceleration of the robot.

To explicitly demonstrate this aspect of robot motion, imagine the roll scenario illustrated in Fig.6.2 where roll instances are arranged chronologically from left to right. In this scenario, θ_t is set above the optimal range identified in Section 5.2 for $\omega_d = -2.0$ rad/s, meaning that actuation occurs "late" in the γ cycle, like at 65°. At the first instance in the scenario, A has just swept past the trigger angle, and shape change actuation has begun. Accordingly, arrows signify the linear actuator is making A smaller and B longer. The moment arm, x_d, is negative in the first instance, because oblongness of the outer surface combined with the tilt of late actuation place the touchpoint to the left of robot center of mass. With a negative moment arm, Nx_d is also negative (clockwise), because N is always directed upwards (and is approximated as being constant in this mental exercise), and the robot speeds up as a result. Because actuation is late in this scenario, one expects the outer surface will not become circular until after continued rolling when B has already rotated beyond the vertical – a reasonable expectation, given the limited speed of the linear actuator. Therefore, by the time A has rotated into the horizontal orientation in the second instance, the outer surface is still oblong about A. Consequently, as A rotates beyond the horizontal as illustrated in the third instance in Fig. 6.2, x_d becomes positive and Nx_d becomes counterclockwise, and the robot slows down as a result. This slowing is referred to as inadvertent braking. By the fourth instance in Fig. 6.2, the outer surface has been reduced to the circular configuration, and the normal force location has rapidly shifted to the location right under the robot center of mass, causing x_d and Nx_d to

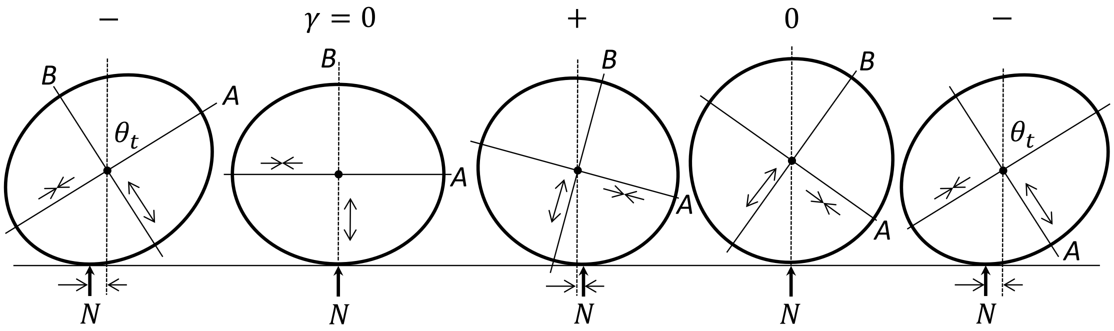


Figure 6.2: When the robot exhibits the late actuation pattern, inadvertent braking occurs after the robot has reached the $\gamma = 0$ orientation.

be zero again. By the fifth instance, the outer surface has become oblong about B, causing x_d and Nx_d to once again become positive. Actuation is triggered newly by B at the fifth instance, and the scenario subsequently repeats with A and B now in switched positions.

A symbolic representation is introduced to characterize the input torque pattern on the robot in this late actuation scenario. A plus sign is used to represent counterclockwise torque on the robot center of mass, and a negative sign is used to represent clockwise torque. The "0" digit is used to represent zero torque on the robot due to the normal force. The expression, $\gamma = 0$, is used to represent that A or B is oriented horizontally. When A or B is horizontal, not only is gamma equal to zero, but Nx_d is equal to zero, too. Starting at the first instance and using this symbolic representation, the pattern displayed by the late actuation scenario is: $-$, $\gamma = 0$, $+$, 0, $-$, $\gamma = 0$, $+$, 0, $-$, etc. A marked feature of this actuation pattern is the sequence: $-$, $\gamma = 0$, $+$; that is to say, when $\gamma = 0$, torque on the robot is changing from negative to positive (clockwise to counterclockwise).

Now imagine the roll scenario illustrated in Fig.6.3, where roll instances in the scenario are again arranged chronologically from left to right. In this scenario, θ_t is set below the optimal range identified in Section 5.2 for $\omega_d = -2.0$ rad/s, meaning that actuation occurs "early" in the γ cycle, like at 35°. At the first instance in the scenario, A has just swept past the trigger angle, and shape change actuation has begun. The moment arm, x_d, is negative, because oblongness of the outer surface, combined with the slight tilt of the robot, place the touchpoint to the left of the robot center of mass. With a negative moment arm, Nx_d is clockwise in the first instance, and the robot speeds up as a result. As shape change actuation occurs early in this scenario, the outer surface of the robot actuates to the circular configuration, as in the second instance in Fig.6.3, before A crosses the horizontal orientation. In the circular configuration, Nx_d is zero. At instance three, the robot has become oblong about B, and A still has not crossed the horizontal, so that x_d is positive and Nx_d is counterclockwise (the robot undergoes inadvertent braking). At instance four, A has rotated into the horizontal, at which point x_d and Nx_d inevitably become zero again, and the scenario subsequently repeats with A and B in switched positions by the fifth instance.

In order to characterize the input torque pattern on the robot in the early actuation scenario, the same symbolic representation is used as before, wherein a plus sign is used to represent counterclockwise torque on the robot, and a negative sign is used to represent clockwise torque. The "0" digit is used to represent zero torque on the robot, and the expression, $\gamma = 0$, is used to represent the instance when A or B is oriented horizontally, when gamma and Nx_d are both zero. Starting at the first instance and using this symbolic representation, the pattern displayed by the early actuation scenario is: $-$, 0, $+$, $\gamma = 0$, $-$, 0, $+$, $\gamma = 0$, $-$, etc. A marked feature of this actuation pattern is the sequence: $+$, $\gamma = 0$, $-$; that is to say, when $\gamma = 0$, torque on the robot is changing from counterclockwise to clockwise.

To learn if the robot exhibits early or late actuation patterning during actual locomotion, four individual trial rolls from Problem 2.1, one for each value of θ_t, are chosen and investigated. The values of ϱ for the chosen rolls are consistent with the differences between average values of ϱ for the corresponding populations listed

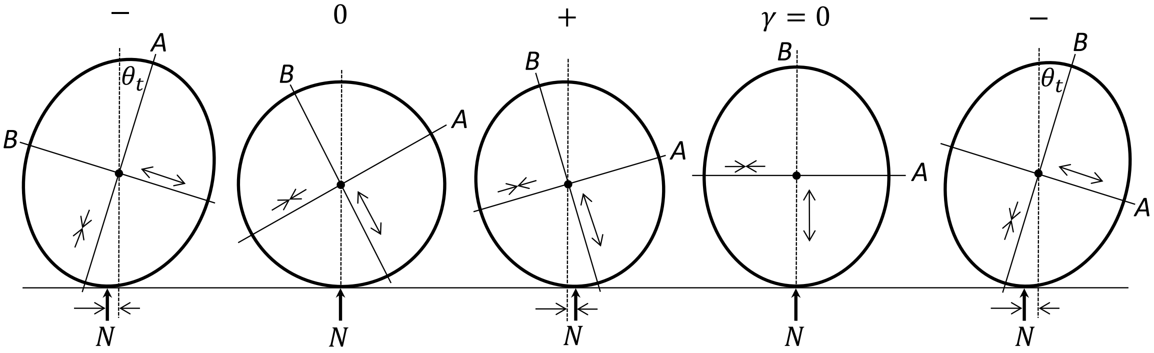


Figure 6.3: When the robot exhibits the early actuation pattern, inadvertent braking occurs before the robot has reached the $\gamma = 0$ orientation.

in Table 5.11. Problem 2.1 is chosen for this investigation, because its range of θ_t includes values below and above the region of optimality determined for the problem in Section 5.1. The four trial rolls that are investigated are referred to in this section as Roll A, Roll B, Roll C and Roll D, respectively corresponding to $\theta_t = 35°$, 45°, 55°, and 65°. Equation 3.6 is used with reported roll data to compute x_d at every tenth of a second for Roll A, and computed values of x_d for this roll are plotted versus roll time in Fig. 6.4. Dashed, vertical lines in the graph represent approximate moments during the rolls when $\gamma = 0$. A plot of angular velocity versus roll time for Roll A is included in Fig. 6.5.

The plot of x_d in Fig. 6.4 reveals that x_d is frequently positive, which means the rolling robot experiences repeated inadvertent braking during steady state angular velocity. These times are relatively short-lived, and the magnitude of x_d during inadvertent braking is generally smaller than when x_d is negative, and this result means that driving torque supplies more energy to the robot than does inadvertent braking, which is expected, since the purpose and demonstrated outcome of the control system is to maintain forward motion of the robot. Also, x_d is always roughly zero when $\gamma = 0$, a result which reflects actual motion of the robot and bolsters legitimacy of the x_d model of Equation 3.6. The fact that plotted values of x_d in Fig. 6.4 are not exactly zero at roll times when $\gamma = 0$ is not a cause of concern, as small discrepancies

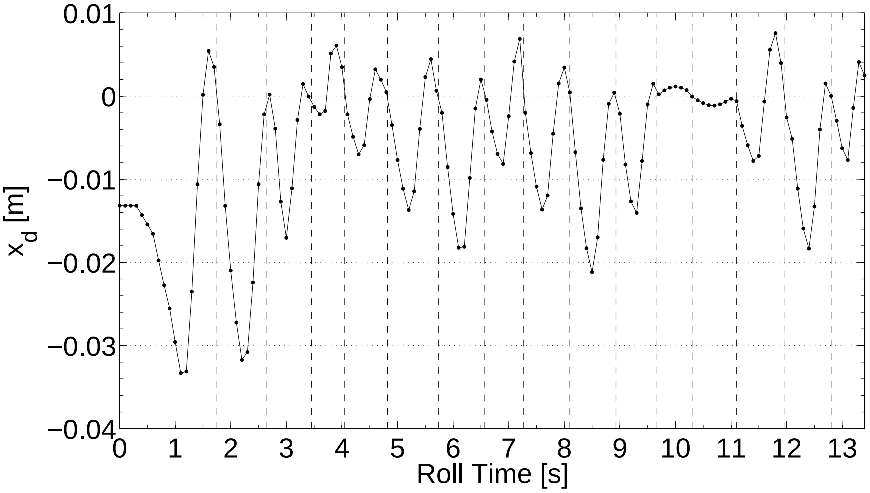


Figure 6.4: For Roll A, the robot exhibits the early actuation pattern, in which inadvertent braking (when x_d is positive) causes the roll to be relatively inefficient.

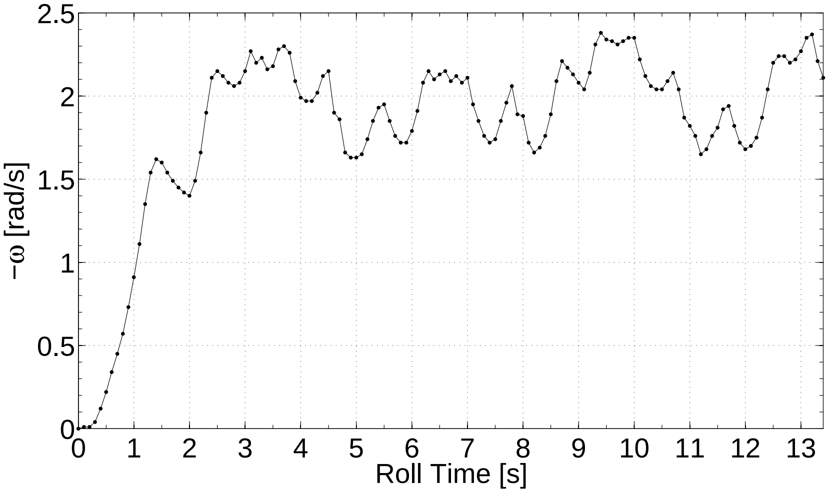


Figure 6.5: Angular velocity of the robot for Roll A, a relatively inefficient roll. At times, the robot slows itself down when it is already going too slow relative to the desired velocity of −2.0 rad/s.

arise from interpolation between reported data points.

The actuation pattern for Roll A is clearly that of early actuation. The graph in Fig. 6.4 shows 13 times at which $\gamma = 0$, and at ten or more of these times, the x_d pattern is: $+$, $\gamma = 0$, $-$. For this roll, θ_t is set to 35°, which is below the optimal range identified in Section 5.2 for $\omega = -2.0$ rad/s, so it is not surprising the early actuation pattern would manifest here. The only visual peculiarity in the x_d pattern in Fig. 6.4 occurs at roughly 10 seconds, when the plot goes nearly flat for about one second. This peculiarity is due to the computed value of $L_t - a_i$ at two consecutive trigger moments being less than 0.003 m in magnitude. At each of these trigger moments, the control system keeps the linear actuator at its current length, so as not to expend energy for a relatively inconsequential actuation, as described in Section 3.6. During this second, or so, as the outer surface maintains its configuration, it is very close to being circular, and hence the computed value of x_d (from Equation 3.3) is zero or nearly zero.

Behavior of the robot in the early actuation pattern during Roll A sheds light on why there is a lower limit on optimality of θ_t in Problem 2.1. When θ_t is set too low (actuation triggering is too early), the robot is prone to inadvertent braking, potentially at every γ cycle. Notice from Fig. 6.4 that when braking occurs, it is not always beneficial. In other words, the robot often works against itself with considerable energy during the roll, slowing itself down when it is already going too slow. A good example occurs at roughly 5.5 seconds. At this time, the robot is already rotating too slowly (see Fig. 6.5) relative to the desired velocity magnitude when inadvertent braking comes on at 5.8 seconds and slows down the robot even more. Consequently, the linear actuator is subsequently forced by the control system to move in large magnitude to create driving torque in order to speed up the robot and maintain the desired velocity. A similar thing happens between 11 and 12 seconds. With so much ill-timed braking, it is understandable why Roll A is not optimal in terms of energy

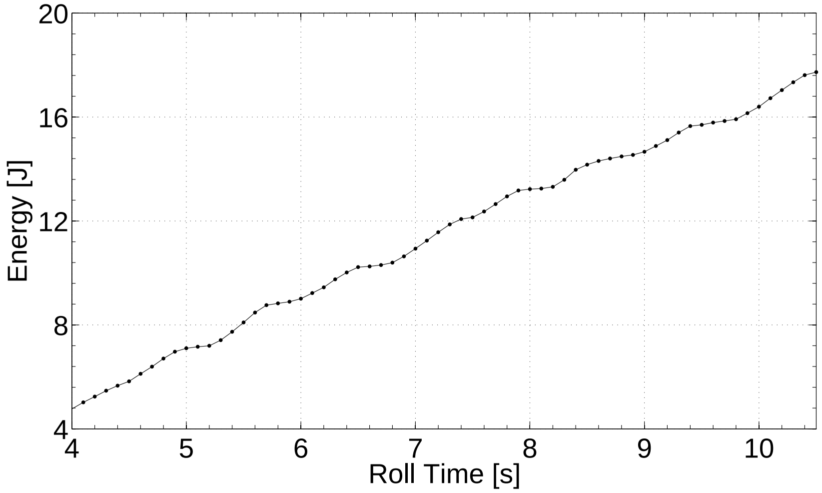


Figure 6.6: Rate of energy consumption by the servo motor during Roll D is generally higher during actuations when A is the trigger axis.

economy.

Similar investigations into Rolls B, C and D reveal these three rolls exhibit a combination of the early and late actuation patterns. In fact, there is nothing about the plots of x_d for these three rolls that would point to why robot motion during Rolls B and C is more economical than Roll D. This absence of contrast leads one to believe there is a factor separate from actuation patterning that affects energy economy of rolls in Problem 2.1. In an effort to identify this factor, energy consumed by the servo motor during Roll D is plotted in Fig. 6.6 versus roll time for several seconds during the roll. Also plotted for Roll D is length of semi-major axis, a, plotted versus roll time in Fig. 6.7. Next to the plot in Fig. 6.7 where actuation occurs, trigger axis (A or B) is noted on the graph. One immediately notices in Fig. 6.6 that slope of the energy curve varies over roll time, and by comparing Fig. 6.6 and Fig. 6.7, it is evident that periods of highest energy consumption rate correspond, roughly, to actuation periods of the robot when A is the trigger axis.

Using output data from Roll D saved from Phase 2 of the primary experiment, the average rate of energy consumption by the servo motor is calculated for actuation when A is the trigger axis, and it is found to be 2.60 J/s. When B is the trigger

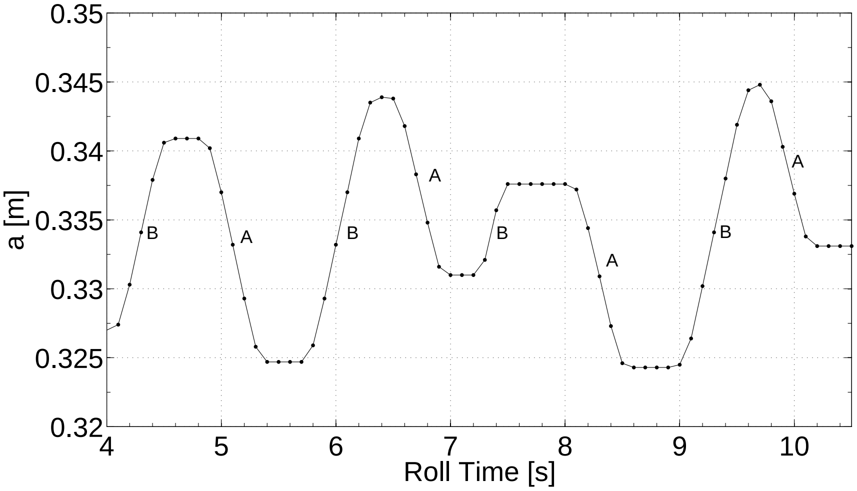


Figure 6.7: Length of a versus roll time for Roll D. Next to the plot where actuation occurs, trigger axis (A or B) is noted.

axis, rate of energy consumption during actuation is 1.90 J/s on average, and during periods when the actuator is holding a steady, rate of energy consumption is 0.98 J/s on average. These measurements are also performed for Rolls A, B and C, and it is found that energy consumption rate when B is the trigger axis is about the same, on average, for all rolls: approximately 1.90 J/s, and for holding, too: 0.98 J/s. When A is trigger axis, however, rate of energy consumption changes as a function of trigger angle. For $\theta_t = 35°$ (Roll A), the average rate is 1.79 J/s; for $\theta_t = 45°$ (Roll B), the average rate is 1.95 J/s; and for $\theta_t = 55°$ (Roll C), the average rate is 2.22 J/s. Average energy consumption rate of the servo motor is plotted versus θ_t in Fig. 6.8 for these four rolls. The difference in energy consumption rate between actuations with $\theta_t = 35°$ and $\theta_t = 65°$, for instance, is about 37%.

The reason higher trigger angle values are associated with greater actuation burdening is that bending-induced friction in the telescoping columns increases as median orientation of A during actuation moves closer to the horizontal (Section 3.7.2). Also, when median orientation of A is nearly horizontal, deformation of the outer surface is greater during actuation, which results in outwardly directed end effects that pull on the actuator and resist contraction (and recall from Section 3.6 that when actuation

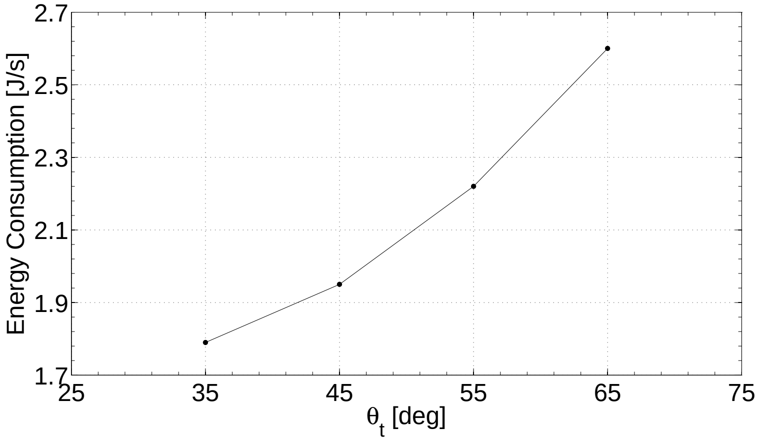


Figure 6.8: For rolls of the robot with $\omega_d = -2.0$ rad/s and when A is the trigger axis, average rate of energy consumption by the linear actuator's servo motor changes as a function of trigger angle.

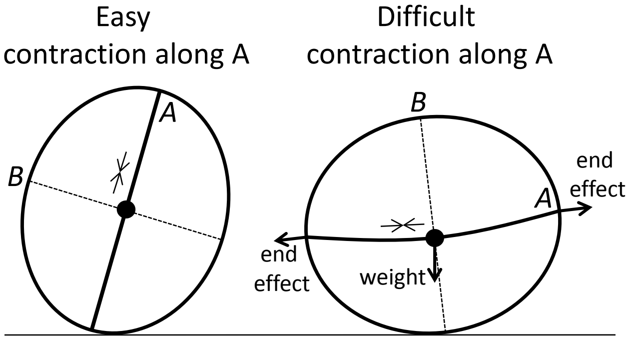


Figure 6.9: When A is oriented vertically, gravity makes contraction easy. When horizontal, bending-induced friction and outwardly directed end effects on the linear actuator make contraction difficult.

is triggered by A, the actuator always contracts). In contrast, when median orientation of A during actuation is oriented closer to the vertical, there is less bending of the telescoping columns and hence less friction; furthermore, gravity actually helps the actuator contract when it is orientated in an upright orientation. The effect of orientation of A on actuation burdening is illustrated in Fig. 6.9.

With measured rates of average energy consumption for Rolls A through D, the simulation program described in Section 3.8 is modified to recognize trigger angle and numerically integrate the appropriate consumption rate over time to compute total energy consumed, E_m, from 4 to 14 seconds for Roll A. Then using Equation 4.7, the simulation program is amended to compute W_{NC} and use this value with Equation 4.6 to compute ϱ for the roll. In the same manner, the simulation program is used to compute ϱ for the other rolls in Problem 2.1, and the results are: for $\theta_t = 35°$ (Roll A), $\varrho = 2.17$; for $\theta_t = 45°$ (Roll B), $\varrho = 2.11$; for $\theta_t = 55°$ (Roll C), $\varrho = 2.07$; and for $\theta_t = 65°$ (Roll D), $\varrho = 2.19$. These results predict that Roll B and Roll C are the most economic rolls, which is in perfect agreement with the ultimate determination regarding the optimal range of θ_t for Problem 2.1 based on experimental results.

Results for energy economy from Problem 2.1 can now be understood as stemming from a combination of inadvertent braking and actuation burdening. To summarize, at one extreme, when θ_t is set low at 35°, angular velocity of the robot, which is -2.0 rad/s on average, is small enough in magnitude so the actuating outer surface often attains the circular configuration before $\gamma = 0$. In other words, the robot exhibits the early actuation pattern, as illustrated in Fig. 6.3, in which it frequently undergoes inadvertent braking and works against itself. For this reason, robot locomotion is not the most economical for $\theta_t = 35°$. At the other extreme when θ_t is set high at 65°, actuation becomes increasingly burdensome due to the effects of gravity and bending of the outer surface, thereby nullifying gains in energy economy that might otherwise be had. In between 35° and 65°, there is an optimal range where θ_t is high enough for the robot to largely avoid the averse early actuation pattern, yet low enough so that actuation burdening does not greatly hinder the system.

Based on these ideas, a graph is provided in Fig. 6.10 that illustrates how inadvertent braking and actuation burdening act together to affect robot energy economy at different values of θ_t for trial rolls in Problem 2.1 with $\omega_d = -2.0$ rad/s. In the graph, energy economy in the form of ϱ is plotted twice as a function of θ_t. The dashed plot represents what ϱ would be if actuation burdening did not play a significant role in robot energy economy. Inadvertent braking has a primary role forming this dashed line plot, causing the robot to have relatively poor energy economy at lower trigger

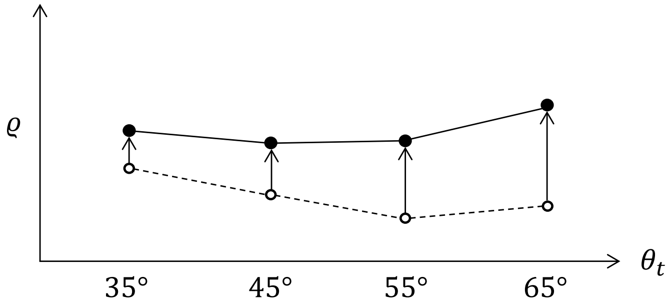


Figure 6.10: The dashed plot represents what ϱ might be if only actuation patterning played a significant role in affecting robot energy economy in Problem 2.1. When effects of actuation burdening are considered (upward arrows), ϱ is highest when $\theta_t = 65°$.

angles. However, when effects of actuation burdening are considered, represented in the graph as upward arrows, roll economy of the robot is lowered (ϱ is increased) across-the-board, resulting in lowest energy economy when $\theta_t = 65°$.

Turning attention to the results of Problem 2.2, one notices the determined optimal range of θ_t presented in Tab. 5.14 includes all low end values of θ_t tested. On the high end of θ_t, however, 55° is determined to be above the optimal range of θ_t. In order to help make sense of these results for Problem 2.2, two individual trial rolls from the problem, one on the low end of θ_t and one on the high end of θ_t, are chosen and investigated. These rolls have $\theta_t = 35°$ and $\theta_t = 55°$, and they are referred to as Roll E and Roll F, respectively. Values of ϱ for Rolls E and F are consistent with average values of ϱ for the corresponding populations listed in Table 5.11. Plots of x_d and ω versus roll time for Roll E are shown in Figures 6.11 and 6.12, and a plot of x_d for Roll F is shown in Fig. 6.13. Dashed, vertical lines in Figures 6.11 and 6.13 represent approximate moments during the rolls when $\gamma = 0$.

Looking side-by-side at plots of x_d and ω_d for Roll E in Figures 6.11 and 6.12, it is observed that something happens during Roll E that does not happen in most other measured rolls of the robot: peaks of x_d coincide almost perfectly with peaks of angular velocity, and troughs of x_d coincide almost perfectly with troughs of angular velocity. This alignment is consistent throughout the duration of the roll, and even

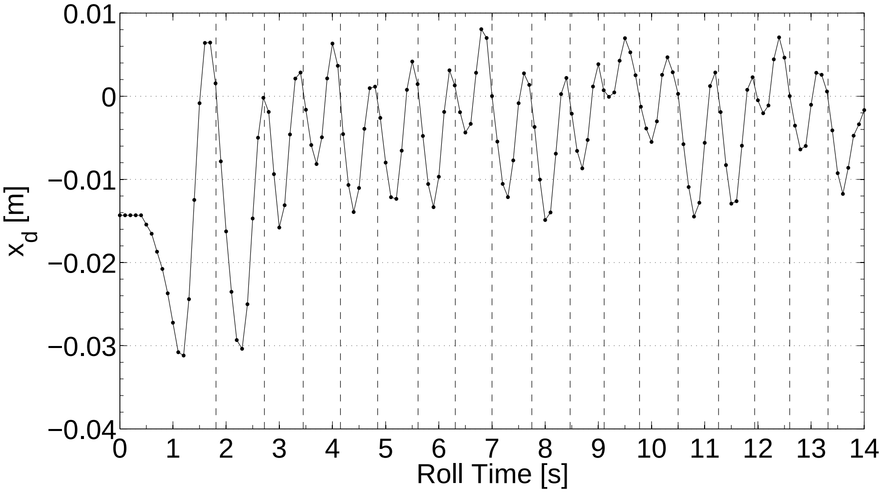


Figure 6.11: The plot of x_d for Roll E exhibits the early actuation pattern.

though Roll E clearly exhibits the early actuation pattern with considerable inadvertent braking, it is usually non-adverse. That is to say, inadvertent braking in Roll E is strongest only when robot velocity crescendos in magnitude and is too high relative to the desired velocity. In addition, driving torque is strongest when the robot is rolling too slowly. Because control of the robot is apparently synchronized in this manner, the actuator is not commanded to move in large magnitude in order to reduce large errors, and this characteristic makes the roll relatively economical in terms of energy consumed per unit roll distance.

The plot of x_d for Roll F in Fig. 6.13 differs from Roll E in that it does not show as much inadvertent braking. Also, the plot of x_d for Roll F exhibits more driving torque than Roll E. Alone, these facts might cause one to believe that Roll F is more economical than Roll E, yet because of actuation burdening, it is not. Effects of actuation burdening shift downward on θ_t as the robot rolls faster, because the A-axis is more quickly swept into the horizontal position from its initial tilt when actuation commences. Thus, Roll F with $\theta_t = 55°$ and $\omega_d = -2.2$ rad/s undergoes actuation burdening that is comparable to Roll D with $\theta_t = 65°$ and $\omega_d = -2.0$ rad/s. According to the plot in Fig. 6.9, this burden is high. Meanwhile, Roll E with a lower

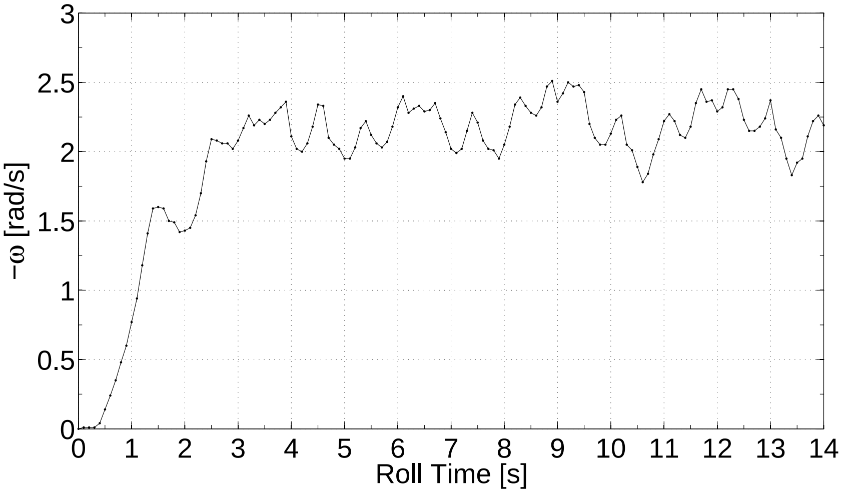


Figure 6.12: For Roll E, peaks of ω coincide with peaks of x_d, and troughs of ω coincide with troughs of x_d.

trigger angle undergoes actuation burdening that is comparable to Roll A, which is much less in magnitude than Roll D/F. The large burden undergone by Roll F causes the roll to consume more energy per unit roll distance than Roll E.

With insights provided through comparing these two rolls, results from Problem 2.2 can be understood as stemming from a combination of inadvertent braking and actuation burdening. At one extreme when θ_t is set low, inadvertent braking due to early actuation patterning causes the robot to be less economical. At the other extreme when θ_t is set high, actuation burdening significantly lowers robot energy economy. In between 35° and 55°, there is an optimal range where θ_t is high enough for the robot to largely avoid the averse early actuation pattern, yet low enough so that actuation burdening does not greatly hinder the system. Because the robot rolls faster in Problem 2.2 than in Problem 2.1, early actuation patterning is not significant enough to produce an inferior value of θ_t at the low end for Problem 2.2. Therefore, the optimal range of θ_t for Problem 2.2 includes all values below 55°.

The same explanation given here for energy economy of Problem 2.2 applies to Problem 2.3, except with the difference that actuation burdening is shifted downward in Problem 2.3 relative to Problem 2.2, so that $\theta_t = 45°$ is rendered least economical in

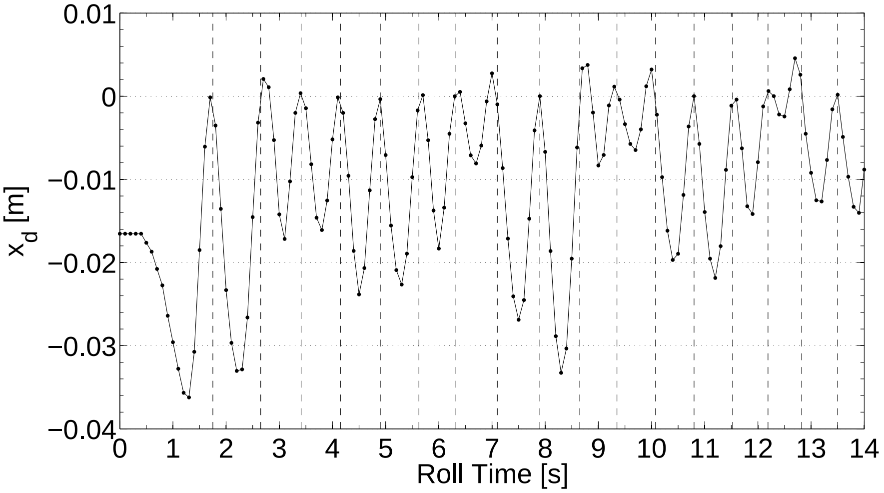


Figure 6.13: The plot of x_d for Roll F shows little inadvertent braking and more driving torque than Roll E.

Problem 2.3, while the lower trigger angles tested are most economical. Interestingly, for all Problems in Phase 2, late actuation burdening does not play a large role in affecting energy economy of the robot. For values of θ_t tested in Phase 2, the robot simply does not roll fast enough for inadvertent braking (due to late actuation) to be a major factor; although inadvertent braking is believed to be a cause of instability of the robot for $\theta_t > 45°$ in Problem 2.3.

Based on these ideas for locomotion of the robot at −2.20 rad/s and −2.4 rad/s, two graphs are composed, one for Problem 2.2 in Fig. 6.14 and one for Problem 2.3 in Fig. 6.15, that illustrate how inadvertent braking and actuation burdening act together to affect robot locomotion at different values of θ_t. In each of these plots, energy economy in the form of ϱ is plotted twice as a function of θ_t. The dashed plots represent what ϱ would be if actuation burdening did not play a significant role in robot energy economy. Inadvertent braking has a primary role forming these dashed line plots, causing the robot to have relatively poor energy economy at lower trigger angles. However, when effects of actuation burdening are considered, represented in the graph as upward arrows, roll economy of the robot is lowered (ϱ is increased) across-the-board, resulting in lowest energy economy at higher values of θ_t.

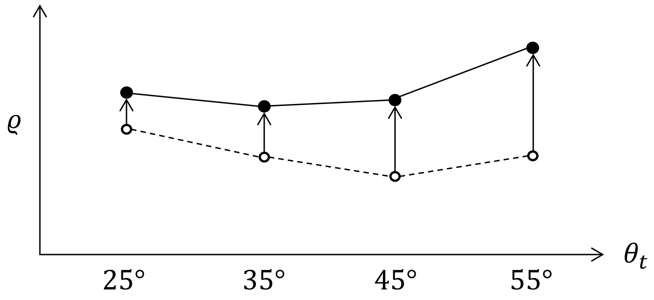


Figure 6.14: The dashed plot represents what ϱ might be if only actuation patterning played a significant role in robot energy economy in Problem 2.2. When effects of actuation burdening are considered (upward arrows), ϱ is highest when $\theta_t = 55°$.

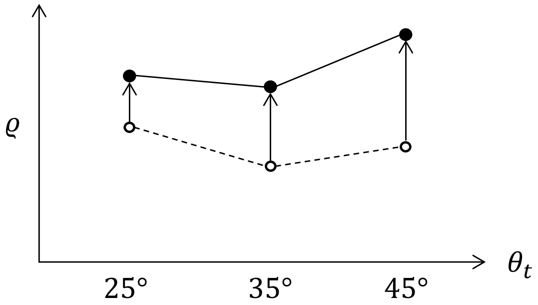


Figure 6.15: The dashed plot represents what ϱ might be if only actuation patterning played a significant role in robot energy economy in Problem 2.3. When effects of actuation burdening are considered (upward arrows), ϱ is highest when $\theta_t = 45°$.

6.2 Two Investigations Utilizing the Simulation Program

Trial rolls with duration of about 15 seconds or more are not possible using the roll track described in Section 4.1, because the track is not long enough to accommodate rolls in which the robot travels more than 9 m. As an alternative, the simulation program detailed in Section 3.8 provides accurate prediction of robot performance for longer rolls. Also, the simulation program can predict how a design change that would be difficult to implement, such as increasing size of the robot for instance, would affect performance of the robot. Two brief simulation investigations that take advantage of the simulation program in this manner are presented in this section.

The first investigation is simulation of a 30 second, controlled roll of the robot, in which desired angular velocity varies in a step-like manner, as shown in Fig. 6.16. During the simulated roll, values of θ_t, k_f, and k_s are automatically set according to the optimal control values identified for the various desired velocities in Section 6.1. Specifically, when desired velocity during the simulated roll is equal to -2.0 rad/s, θ_t is set to 45°, k_f is set to 0.02, and k_s is set to 1.14; when desired velocity is equal to -2.4 rad/s, θ_t is set to 35°, k_f is set to 0.04, and k_s is set to 1.09; and when desired velocity is equal to -2.2 rad/s, θ_t remains at 35°, k_f is set to 0.02, and k_s is set to 1.15. Velocity of the robot, output by the simulation program and plotted versus roll time in Fig. 6.16, indicates the robot can automatically operate at various velocities while maintaining locomotion that's characterized by optimal energy economy. The result also indicates the robot can accelerate/decelerate from a nonzero average velocity to a higher/lower average velocity and remain stable while doing so.

For the second investigation, two simulations are performed. In the first of these simulations, perimeter of the outer surface of the robot is doubled relative to outer surface perimeter of the robot used in the primary experiment (Section 4.6), and velocity-controlled rolls of this larger, notional robot with a perimeter of 4.19 m are performed. The larger robot is assumed to have the same general construction as the

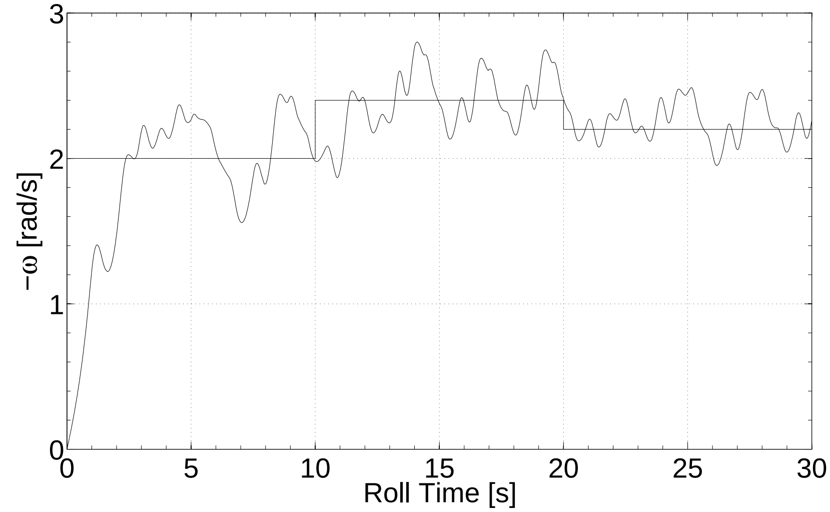


Figure 6.16: The simulation program predicts the rolling robot can be controlled to accelerate/decelerate from nonzero average velocity to a higher/lower average velocity in a stable manner. Desired velocity of the robot is represented in the plot by straight, solid lines.

robot used in the primary experiment, with an additional 72 grams of mass in order to account for the larger outer surface. Also, in order to account for increased bending of the larger outer surface, amplitudes of d and Δx are each doubled relative to their values used in simulation of the robot in the primary experiment. Desired angular velocity in simulated rolls of the larger robot is set to -1.10 rad/s, causing the robot to move with the same linear velocity in the simulation, approximately 0.734 m/s, as the actual robot in Problem 2.2 (Section 5.2).

Initial conditions for simulation of the larger robot are identical to those used for trial rolls in the primary experiment, except that in the simulation, $a(0) = 0.667$ m. Control constants for the simulation are initially set as in Problem 2.2: θ_t is 35°, k_f is 0.02, and k_s is 1.10. However, this combination of control constants does not result in stable, velocity-controlled locomotion, so further simulations are run using various combinations of constants in an effort to attain stability. In doing so, it is found that locomotion of the larger robot becomes stable with θ_t set to 65°, k_f set to 0.02, and k_s set to 1.20. Angular velocity, ω, and semi-major axis, a, for this stable roll are plotted versus simulated roll time in Fig. 6.17 and Fig. 6.18. Referring to the figures,

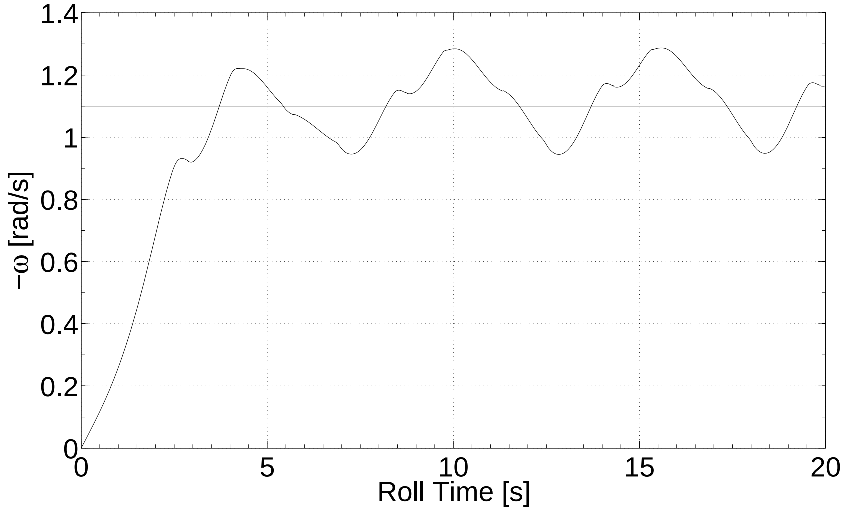


Figure 6.17: Simulated velocity of the rolling robot with a perimeter of 4.19 m, which is double the perimeter of the robot used in the primary experiment. Desired velocity for the simulated trial roll is represented by the line at $\omega_d = -1.10$ rad/s.

it is seen that semi-major axis length does not change more than one centimeter in length once the robot has reached steady state velocity. The reason relatively little actuation is needed to move the larger robot at 0.734 m/s is that increased mass of the robot brings about an increased normal force, N, which in turn increases magnitude of input torque on the robot. Although actuations are shorter during rolls of the larger robot, motion of the actuator would presumably be highly burdened due to the task of displacing increased mass during shape change actuation.

For the second simulation, perimeter of the outer surface of the robot is reduced to one-half the perimeter of the robot used in the primary experiment, and velocity-controlled rolls of this smaller, notional robot with a perimeter of 1.05 m are performed. The smaller robot is assumed to have the same general construction as the robot used in the primary experiment, but with 36 grams less mass in order to account for the smaller outer surface. Also, in order to account for reduced bending of the outer surface of the smaller robot, amplitudes of d and Δx in the simulation are each reduced by one half relative to their values in simulation of the robot used in the primary experiment. Desired angular velocity in the simulated rolls is set to -4.39

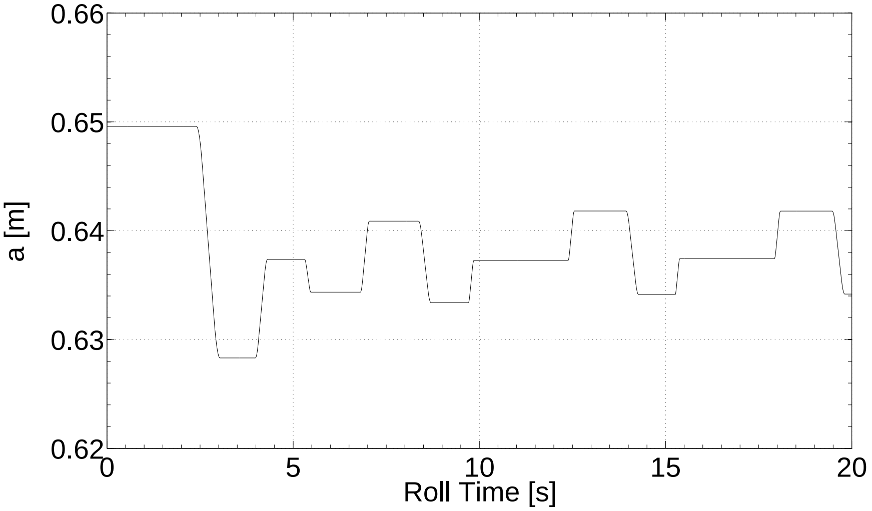


Figure 6.18: Once robot velocity reaches steady state, the semi-major axis does not change more than one centimeter in length during controlled actuation.

rad/s, corresponding to 0.734 m/s, which is roughly the average linear velocity of the actual robot in Problem 2.2 (Section 5.2).

Initial conditions for simulation of the smaller robot are identical to those used for trial rolls in the primary experiment, except that in the simulation, $a(0) = 0.182$ m. Control constants for the simulation are initially set as in Problem 2.2: θ_t is 35°, k_f is 0.02, and k_s is 1.10. However, this combination of control constants does not result in stable locomotion, so several further simulations are run using various combinations of constants in an effort to attain stability; but no matter what combination is applied, it appears the smaller robot cannot reach the desired angular velocity. Angular velocity, ω, and semi-major axis, a, are plotted versus roll time in Fig. 6.19 and in Fig. 6.20 for a simulated roll of the smaller robot. From the plot of a in Fig. 6.20, it appears there is not enough time for the actuator to extend to the full target length in between trigger moments. Because the robot spins so quickly, and because speed of the actuator is limited, length of the semi-major axis gets only so large before it is commanded to contract again at the next trigger moment. This phenomenon acts to limit the obtainable length of the semi-major axis, thereby capping input torque and terminal velocity of the robot.

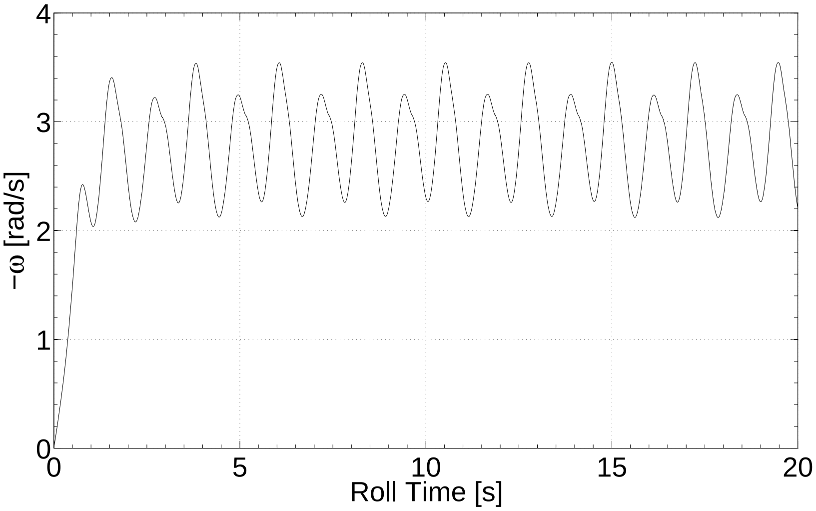


Figure 6.19: Simulated velocity of the rolling robot with a perimeter of 1.05 m, which is half the perimeter of the robot used in the primary experiment. The simulation indicates the smaller robot is unable to reach a desired angular velocity of -4.39 rad/s.

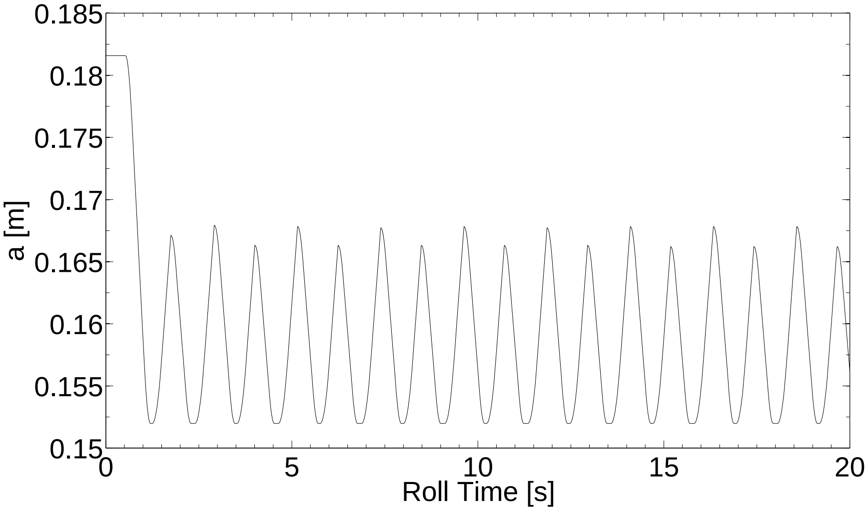


Figure 6.20: Initially, the smaller robot spins slow enough for the actuator to extend about 3 cm, but as the robot spins faster, length of the semi-major axis becomes limited to a small range, because there is not enough time in between target moments for full contraction.

6.3 Model Shortcomings

Differences between model predictions and data from actual trial runs, as presented in Section 5.3, are believed to originate in large part from imperfect characterization of actuation time duration by Equation 3.36. In Section 6.1, it was shown that t_a varies depending largely on median orientation of the A during actuation. Moves in which median orientation of the A-axis is close to the horizontal generally take longer to complete than identical actuations with less tilt. Nevertheless, Equation 3.36 is based on the assumption that all actuations occur with A at 45°, and therefore the model does not capture variation in actuation time duration that is present during trial rolls of the robot. This discrepancy causes predicted motion of the robot to move differently from what is measured during trial rolls.

Another reason for the differences between model predictions and trial roll parameters is imperfect characterization of rolling resistance torque, τ_r, by Equation 3.5.5. Accurately characterizing rolling resistance torque is extremely challenging, because the combined role that outer surface bending and angular velocity of the robot have is largely a mystery. At best, Equation 3.5.5 appears to be a good generalization of how rolling resistance affects average velocity of the robot at steady state. It is believed that imperfect characterization of rolling resistance torque is responsible for marked discrepancies between predicted and actual motion of the robot, such as the early mismatch of velocity between simulation and trial roll in Fig. 5.21 and the apparent tendency of the model to underestimate peaks in velocity magnitude in Fig. 5.15.

The equation of motion for the robot (Equation 3.26) is based on the assumption of a perfectly elliptical outer surface with a center of mass at the intersection of the elliptical axes. This assumption is yet another reason for the discrepancy between model predictions and reported trial roll parameters. In fact, due to imperfections in construction of the robot, the center of mass of the robot is likely not located exactly where the model assumes it is located as described in Section 4.6.4. Furthermore, due to unsymmetrical construction of the robot, the outer surface of the robot is not perfectly elliptical. Collectively, discrepancies such as these have a small but nonnegligible effect on actual locomotion of the robot, yet their effects are not included in the model.

6.4 Future Work

The lessons learned herein concerning the locomotion style of the OSU rolling robot call attention to new design possibilities and future areas of work. One of these possibilities is control system design. Having identified optimal trigger angles for various angular velocities of the robot in the present study, perhaps a control system could be implemented in which trigger angle changes at every trigger moment, so as to avoid inadvertent braking. Another possibility is a study focusing on the number of actuation changes per revolution of the robot. The OSU rolling robot changes shape at most four times per revolution, but perhaps six changes per revolution might be more economical, or maybe even two. In addition to control system design, it would be worthwhile to investigate whether or not there is an optimal outer surface perimeter size for a given linear velocity of the robot. A related consideration is what material(s) would best serve to compose the outer surface for a given size of the robot; bending and vibration of the outer surface would certainly be causes of major concern, and they would therefore need to be considered as part of the investigation. Perhaps the most interesting area of potential future work is exploration of alternative forms of actuation that would be faster and/or lighter than the servo motor-based actuator used on the OSU rolling robot. Along these lines, maybe an adaptive material, such as Nitinol or a thermally tunable foam, could be used in the actuation system. Lastly, the challenge of designing a three-dimensional, spherical version of the OSU rolling robot, one with turning capability, is an exciting idea that could be explored in the future.

6.5 Conclusion

A rolling robot has been developed that generates torque by changing shape of its elliptical outer surface, which is flexible and can be morphed to retain oblongness about one of two notional, elliptical axes that are fixed to the robot and roll with it. The robot has been equipped with a sensing/control system by which it measures its angular position and angular velocity, computes error with respect to a desired step velocity profile, and changes shape of its outer surface accordingly. Shape change actuation occurs four times per revolution, whenever an elliptical axis rotates past a predetermined trigger angle. The robot has demonstrated stability during roll tests, in the sense that it was able to reach a constant desired angular velocity quickly and remain close to it thereafter.

A series of trial rolls of the robot were performed using various trigger angles, while energy consumed by the servo motor was measured and used to calculate energy economy for each roll. Results of this experiment showed that, depending on velocity of the robot, there exists a range of trigger angle values that are determined to be superior in terms of energy economy. This range of optimality on trigger angle generally shifts towards the vertical as desired angular velocity is increased. In search for an explanation, it was found that economical trial rolls featured a synchronicity of actuation timing and angular velocity, wherein the robot avoided agents of inefficiency that slowed the robot or burdened the servo motor; at higher magnitude angular velocities of the robot, only actuations triggered at sufficiently low trigger angles could "keep up" in order to preserve the synchronicity.

A mathematical model was developed for the robot that resulted in a second order, nonlinear differential equation with roll time as the independent variable and angular position of the robot as a dependent variable. A computer program was written that numerically solved the differential equation of motion and was used to plot various motion parameters such as angular velocity and roll distance versus roll time. The program was configured to perform three different runs, each with a set of initial conditions that corresponded to an actual trial roll of the robot, and predicted angular velocity was compared to measured angular velocity from actual trial rolls. The model was significantly accurate in predicting angular velocity and roll distance of the robot, although there were shortcomings to the model, such as how rolling resistance torque and servo motor actuation time were characterized, that caused the model to have inaccuracies, especially for trial rolls with relatively high desired angular velocity. Two brief simulation investigations were performed that utilized the computer simulation in order to predict robot performance for roll scenarios that could not easily be performed in the laboratory.

BIBLIOGRAPHY

[1] P. J. McKerrow, *Introduction to Robotics.* Sydney: Addison Wesley Publishing Company, 1991, ch. Components and Subsystems, p. 57.

[2] R. Bogue, “Shape changing self-reconfiguring robots,” *Ind. Robot: An Int. J.*, vol. 42, no. 4, pp. 290–295, 2015.

[3] W. Watanabe *et al.*, “A fully decentralized control of a serpentine robot based on the discrepancy between body, brain and environment,” in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, St. Louis, MO, Oct 2009, pp. 2421 – 2426.

[4] R. Armour and F. Vincent, “Rolling in nature and robotics: A review,” *J. Bionic Eng.*, vol. 3, pp. 195–208, 2006.

[5] T. Ho and S. Lee, “A novel design of a robot that can jump and roll with a single actuator,” in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, Vilamoura, Algarve, Portugal, 2012, pp. 908–913.

[6] J. Sastra *et al.*, “Dynamic Rolling for a Modular Loop Robot,” *Int. J. Robotics Research*, vol. 28, no. 6, pp. 758–773, 2009.

[7] Y. Sugiyama and S. Hirai, “Crawling and Jumping by a Deformable Robot,” *Int. J. Robotics Research*, vol. 25, no. 5-6, pp. 603–620, 2006.

[8] T. Matsuda and S. Murata, “Stiffness control – locomotion of closed link robot with mechanical stiffness,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, Orlando, FL, 2006, pp. 1491–1498.

[9] R. Tedrake. (2016) Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying and Manipulation (Course Notes for MIT 6.832). Downloaded on March 8, 2017 from http://underactuated.mit.edu.

[10] W. Shen *et al.*, “Multimode locomotion via superbot reconfigurable robots,” *Auton. Robot*, vol. 20, pp. 165–177, 2006.

[11] D. Mellinger *et al.*, “Control of locomotion with with shape-changing wheels,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, Kobe, Japan, 2009, pp. 1750–1755.

[12] W. Knight, “Robotic wheels that just keep rolling along,” *New Scientist*, vol. 183, p. 22, 2004.

[13] Y. Tian *et al.*, “Design and Rolling Analysis of a Novel Deformable Polyhedron Robot,” *Int. J. Adv. Robot. Syst.*, vol. 11, 2014.

[14] J. Ganssle. (2012, Jul.) A Designer’s Guide to MEMS Sensors. Downloaded on September 15, 2017 from http://www.digikey.com/en/articles/techzone/2012/jul/a-designers-guide-to-mems-sensors.

[15] H. Chiu *et al.*, “Multifunctional superbot with rolling track configuration,” in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, San Diego, CA, Nov 2007.

[16] J. Antol *et al.*, “Low Cost Mars Surface Sxploration: The Mars Tumbleweed,” National Aeronautics and Space Administration, Tech. Rep. NASA/TM-2003-212411, Aug 2003.

[17] S. Mochon and T. A. McMahon, “Ballistic Walking,” *J. Biomechanics*, vol. 13, pp. 49–57, 1980.

[18] T. McGeer, "Passive Dynamic Walking," *Int. J. Robotics Research*, vol. 9, no. 2, pp. 62–82, 1990.

[19] A. Tavakoli, "Gravity powered locomotion and active control of three mass system," in *Proc. ASME Dynamic Systems and Control Conf.*, Cambridge, MA, 2010, pp. 141–148.

[20] W. H. Lee and A. C. Sanderson, "Dynamic rolling of modular robots," in *Proc. IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, 2000, pp. 2840–2846.

[21] I. Must *et al.*, "A power-autonomous self-rolling wheel using ionic capacitive actuators," in *Proc. SPIE 9430, Electroactive Polymer Actuators and Devices*, San Diego, CA, Mar 2015.

[22] K. Melo *et al.*, "Motion analysis of an ellipsoidal kinematic closed chain," in *Proc. IEEE IX Latin American Robotics Symposium and IEEE Colombian Conf. Automatic Control*, Bogota, Colombia, Oct 2011.

[23] *XBee/XBee-Pro S1 802.15.4 (Legacy) RF Modules User Guide*, Digi International, 2016.

[24] Arduino. (2016) UNO & Genuino UNO (Overview). https://www.arduino.cc/en/Main/ArduinoBoardUno.

[25] EBL. (2016) EBL 855 4 Bay 9V Li-ion Battery Charger with 4 Pack 600mAh 9V 6F22 Low Self-Discharge Lithium-ion Rechargeable Batteries. www.eblmall.com/ebl-855-4-bay-9v-liion-battery-charger-with-4-pack-600mah-9v-6f22-low-selfdischarge-lithiumion-rechargeable-batteries_p1367.html.

[26] D. Montone, "Temperature Effects on Motor Performance," Haydon Kerk Motion Solutions/Pittman Motors, Tech. Rep.

[27] “A Guide to Understanding Battery Specifications,” MIT Electric Vehicle Team, Terminology Summary, 2008.

[28] *Announced Specification of HS-645MG Standard Deluxe High Torque Servo*, Hitech RCD USA, Inc.

[29] *L3GD20 MEMS Motion Sensor: Three-Axis Digital Output Gyroscope (Doc ID 022116 Rev 1)*, STMicroelectronics, 2011.

[30] *Adafruit 9-DOF IMU Breakout*, Adafruit Industries, 2015.

[31] “Everything about STMicroelectronics’ 3-axis Digital MEMS Gyroscopes,” STMicroelectronics, Technical Article Doc ID 022032 Rev 1, 2011.

[32] *UM10204 I^2C-bus specification and user manual (Rev. 6)*, NXP Semiconductors, 2014.

[33] *MAX4069-MAX4072 Bideirectional, High Side, Current-Sense Amplifiers with Reference (19-2423; Rev 3)*, Maximum Integrated, 2008.

[34] Y. Zhen, “Current Sensing Circuit Concepts and Fundamentals,” Microchip Technology Inc, Tech. Rep. DS01332B, 2010-2011.

[35] S. Tiwari, “Circuit measures currents in dc servo motor,” *EDM Magazine*, p. 122, Jan. 2001.

[36] Arduino. (2016) Arduino MICRO & Genuino MICRO. https://www.arduino.cc/en/Main/ArduinoBoardMicro.

[37] *General Servo Manual (English, Version 2.00)*, Hitec RCD USA, Inc., 2002.

[38] J. Chitode and U. Bakshi, *Power Devices and Machines.* Technical Publications Pune, 2009, ch. D.C. Motors, pp. 7.20, 7.23.

[39] Pololu Corporation. (2001-2016) Pololu Adjustable Step-Up/Step-Down regulator S8V3A. https://www.pololu.com/product/2120.

[40] R. A. Serway, *Physics for Scientist and Engineers*, 3rd ed. Saunders College Publishing, 1990, ch. 7 and 27, pp. 152–173, 740–760.

[41] (2016) Tracker homepage. https://www.cabrillo.edu/∼dbrown/tracker/.

[42] R. Hibbeler, *Dynamics*, Twelfth ed. Upper Saddle River, New Jersey: Prentice Hall, 2010, ch. Kinetics of a Particle: Impulse and Momentum, pp. 262–297.

[43] T. R. Chandrupatla and T. J. Osler, "The Perimeter of an Ellipse," *Math. Scientist*, vol. 35, pp. 122–131, 2010.

[44] H. Flanders, "Differentiation Under the Integral Sign," *Amer. Math. Monthly*, vol. 80, no. 6, pp. 615–627, 1973.

[45] S. K. Clark and R. N. Dodge, "A Handbook for the Rolling Resistance of Pneumatic Tires," Industrial Development Division Institute of Science and Technology at Univeristy of Michigan, Tech. Rep.

[46] S. Ramanujan, "Modular equations and approximations to π," *Quart. J. Pure Appl. Math.*, vol. 45, pp. 350–372, 1914.

[47] *Servo Motors Control & Arduino*, Future Electronics, 2016.

[48] R. Khazanie, *Elementary Statistics in a World of Applications*, 3rd ed. Harper Collins Publishers Inc., 1990, ch. 9 and 11.

[49] J. T. Spooner *et al.*, *Stable Adaptive Control and Estimation for Nonlinear Systems.* New York: John Wiley & Sons, Inc, 2002, ch. Function Approximation, pp. 113–115.

[50] *Concrete Slab Finishes and the Use of F-number System, PDHonline Course S130(1 PDH) Notes*, PDH Online, 2013.

[51] “Guide for concrete floor and slab construction,” American Concrete Institute, Tech. Rep. 302.1R-04, 2004.

[52] W. Savitch, *Absolute C++*. Boston: Pearson, 2013, ch. C++ Basics, p. 4.

[53] S. Pal, *Numerical Methods*. Oxford University Press, 2013, ch. Numerical Integration, pp. 439–440.

[54] M. A. Pett, *Nonparametric Statistics for Health Care Research*. Thousand Oaks: Sage Publications, 1997, ch. Addressing Differences Among Groups, pp. 204–211.

[55] Y. Chou, *Statistical Analysis*. Holt, Rinehart & Winston of Canada, 1969, ch. Measures of Long-Run Growth, pp. 569–574.

[56] D. Russell, “Swinging Weights of Baseball and Softball Bats,” *The Physics Teacher*, vol. 48, pp. 471–474, 2010.

[57] S. Rabinovich, *Measurement Errors and Uncertainties Theory and Practice*. Springer-Verlag, 2000, pp. 163–196.

[58] Atmel Corporation, “AVR120: Characterization and Calibration of the ADC on an AVR,” Atmel Corporation, Application Note Rev.; Atmel-2559E, Sep 2016.

APPENDIX

This appendix contains listings of four computer programs: the energy sensor program (Listing 1), the simulation program (Listing 2), the control program (Listing 3), and the median test program (Listing 4). For the simulation program, the Simulink block diagram user interface that corresponds to the program is included in Fig. 21.

Listing 1: Energy Sensor Program

```
#include <Wire.h>
#include <I2C_Anything.h>
#define cbi(sfr, bit) (_SFR_BYTE(sfr) &= ~_BV(bit)) // set ADC clock to 1 MHz
#define sbi(sfr, bit) (_SFR_BYTE(sfr) |= _BV(bit)) // set ADC clock to 1 MHz

#define MY_ADDRESS 0x58
#define SIZE (sizeof energy)

 volatile int analog1; //variable for ADC value
 volatile int analog2; //variable for ADC value
 int flag = 0;
 int j = 0;
 int i = 1;
 int n = 0;
 int now1;
 int now2;

 float current; // amps
 float energy; // Joules
 float energy_array[151];
 float energy_last = 0;
 float power; // W
 float power_last = 0;
```

```
float Rsense = 0.100; // Ohms
float T = 200; // main loop period in microseconds
float voltage; // V

void setup()
{ sbi(ADCSRA, ADPS2); // set ADC clock to 1 MHz
  cbi(ADCSRA, ADPS1); // set ADC clock to 1 MHz
  cbi(ADCSRA, ADPS0); // set ADC clock to 1 MHz

  Serial.begin(57600);
  Wire.begin(MY_ADDRESS);
  analogReference(EXTERNAL);
  pinMode(A2, INPUT); // ADC
  pinMode(A3, INPUT); // ADC
  pinMode(7, INPUT); // signal from main microprocessor
}

void loop()
{ while (digitalRead(7) == HIGH)
  { if (j == 0) now1 = micros();
    analog1 = analogRead(A2); //read MAX4071 voltage out
    current = (2.00*analog1*(3.96/1023.0) - 1.5)/(50.0*Rsense);
    if (current < 0) current = 0;
    analog2 = analogRead(A3); //read voltage across servo motor
    voltage = 1.501*analog2*(3.96/1023.0); // digital to analog
    power = current*voltage;
    energy = energy_last + 0.50*0.0002*(power_last + power); // integrate power
    energy_last = energy;
    power_last = power;

    if(j == 0) //fill energy array
    { energy_array[0] = energy;
```

```
    }

    if ((j == 500) && (i < 151))
    { j = 0;
      energy_array[i] = energy;
      i++;
    }

    j++;
    flag = 1;

    // ensure period is equal to 200 ms
    now2 = micros();
    if ((now2 - now1) > 200) energy_array[150] = 1.01;
    int delayTime = T - (now2 - now1);
    delayMicroseconds(delayTime);
    now1 = micros();

  }

  if ((n == 0) && (flag == 1)) Wire.onRequest(onRequestEvent);
}



// send energy to Uno
void onRequestEvent(void)
{ if (n < 151)
  { energy = energy_array[n];
    I2C_writeAnything(energy);
    n++;
  }
}
```

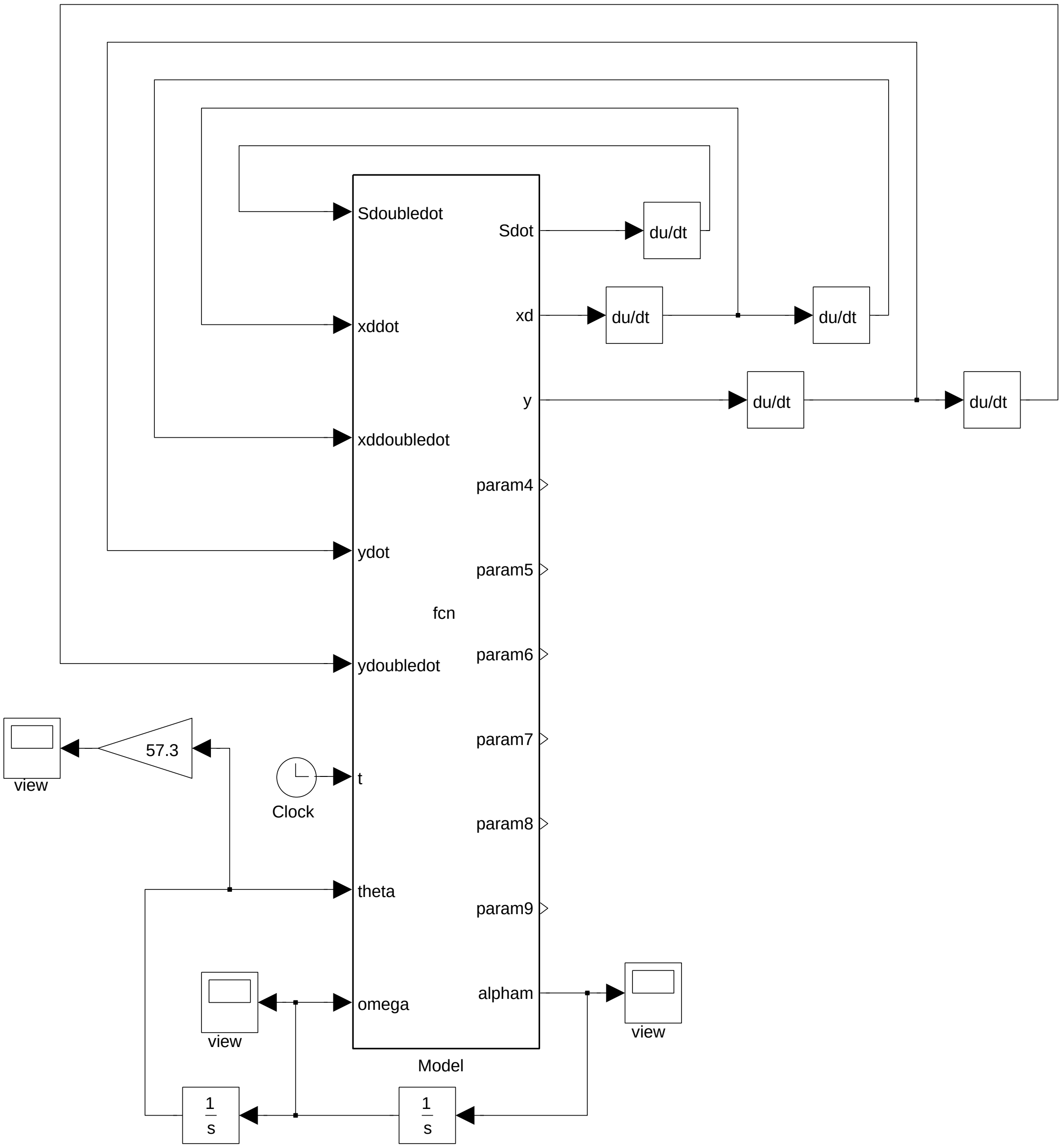


Figure 21: The incremental looping structure of the simulation program is developed using a block diagram user interface in Simulink, in which numerical derivatives and integrals are applied as blocks. Code from Listing 2 in this appendix composes the block labeled, "fcn."

Listing 2: Simulation Program

```
function [Sdot,xd,y,param4,param5,param6,param7,param8,param9,alpham] = ...
    fcn(Sdoubledot,xddot,xddoubledot,ydot,ydoubledot,t,theta,omega)

%CONSTANTS AND INITIALIZATION
axismax = 0.349; %max value of major and minor axes in m
Circumference = 2.095;
M = 0.95; %total robot mass in kg
g = 9.81; %acceleration of gravity in m/s^2
mring = 0.535; %mass of outer surface in m
kr = 1.4;

%CONTROL CONSTANTS
omegad = -2.4;
thetam = 35; %in deg
thetat = 35; %in deg
ks = 1.09 ;
kf = 0.040;

%MIN VALUE OF MAJOR AND MINOR AXES
axismin = 0.319;

%DECLARE PERSISTENT VARIABLES
persistent n
if isempty(n)
    n = 0;
end

persistent m
if isempty(m)
    m = 0;
```

```
end

persistent alast
if isempty(alast)
    alast = axismax;
end

persistent gammalast
if isempty(gammalast)
    gammalast = 0;
end

persistent a
if isempty(a)
    a = axismax;
end

persistent morphnumber
if isempty(morphnumber)
    morphnumber = 0;
end

persistent z
if isempty(z)
    z = 0;
end

persistent tactuate
if isempty(tactuate)
    tactuate = 0;
end
```

```
persistent flag
if isempty(flag)
    flag = 0;
end

persistent ts
if isempty(ts)
    ts = 0;
end

persistent adot
if isempty(adot)
    adot = 0;
end

persistent bdot
if isempty(bdot)
    bdot = 0;
end

persistent adoubledot
if isempty(adoubledot)
    adoubledot = 0;
end

persistent bdoubledot
if isempty(bdoubledot)
    bdoubledot = 0;
end

persistent Vm
if isempty(Vm)
```

```
    Vm = 0;
end

persistent amax
if isempty(amax)
    amax = axismax;
end

persistent omegam
if isempty(omegam)
    omegam = 35;
end

%SEMIMAJOR AXIS
A = 6;
B = 8*a - 6*Circumference/pi;
D = Circumference*Circumference/(pi*pi) - ...
6*Circumference*a/pi + 6*a*a;
b = -B/(2*A) + (1/(2*A))*sqrt(B*B - 4*A*D);

%xd, y
gamma = -rem(theta*57.296,90);
psi = -rem(theta*57.296,360);

dy = 0.0093*abs(sin(psi/57.3));
cs = 0.005;
if (0 <= psi) && (psi < 180)
    dxd = - cs*abs(sin(2*psi/57.3));
else
    dxd =  cs*abs(sin(2*psi/57.3));
end
```

```
xd = dxd + (a^2 - b^2)*cos(theta)*sin(theta)/ ...
    sqrt(a^2*cos(theta)*cos(theta) + b^2*sin(theta)*sin(theta));
y = dy + sqrt(a^2*cos(theta)*cos(theta) + b^2*sin(theta)*sin(theta));

%DERIVATIVE OF ARC LENGTH
phi = atan(xd/y);
sigma = -theta + phi;
R = a*b/sqrt(a^2*sin(sigma)^2 + b^2*cos(sigma)^2);
Rprime = a*b*(a^2 - b^2)*sin(sigma)*cos(sigma)/sqrt((a^2*sin(sigma)^2 + ...
    b^2*cos(sigma)^2)^3);
sigmadot = (y*xddot - xd*ydot)/(y^2*(1 + (xd/y)^2)) - omega;
Sdot = sigmadot*sqrt(R^2 + Rprime^2);

%TIME-DERIVATIVE OF ROBOT ANGULAR MOMENTUM
%outer surface first
q = 40;
p1 = 0;
p2 = 0;
for i = 0:1:q
    beta = (2*pi/q)*i;
    r = a*b/sqrt(a^2*sin(beta)^2 + b^2*cos(beta)^2);
    rdot = ((a*bdot + b*adot)*(a*b/r)^2 - a*b*(a*adot*sin(beta)^2 + ...
        b*bdot*cos(beta)^2))/(a^2*sin(beta)^2 + b^2*cos(beta)^2)^1.5;
    p1 = p1 + r^2*(mring/q);
    p2 = p2 + 2*(mring/q)*r*rdot;
end

%now add point mass contributions
dm = 0.36;
mm = 0.110;
mj = 0.018;
p1 = p1 + 2*mm*(dm - a)^2 + 2*mj*a^2;
```

```
p2 = p2 + 2*2*mm*(dm - a)*(-adot) + 2*2*mj*adot*a;

%now add contribution from switch sensors
mt = 0.012;
p1 = p1 + 2*mt*(a*b/sqrt(a^2*sin(75/57.3)^2 + b^2*cos(75/57.3)^2))^2 ...
    + 2*mt*(a*b/sqrt(a^2*sin(165/57.3)^2 + b^2*cos(165/57.3)^2))^2;
p2 = p2 + 4*mt*(a*b/sqrt(a^2*sin(75/57.3)^2 + b^2*cos(75/57.3)^2)) ...
    *(((a*bdot + b*adot)*(a*b/r)^2 - a*b*(a*adot*sin(75/57.3)^2 + ...
    b*bdot*cos(75/57.3)^2))/(a^2*sin(75/57.3)^2 + ...
    b^2*cos(75/57.3)^2)^1.5) + 4*mt*(a*b/sqrt(a^2*sin(165/57.3)^2 ...
+ b^2*cos(165/57.3)^2))*(((a*bdot + b*adot)*(a*b/r)^2 - ...
    a*b*(a*adot*sin(165/57.3)^2 + b*bdot*cos(165/57.3)^2)) ...
        /(a^2*sin(165/57.3)^2 + b^2*cos(165/57.3)^2)^1.5);

%ROLLING RESISTANCE TORQUE
N = M*(ydoubledot + g);
Tr = kr*dy*omega^2;

%UPDATE ANGULAR ACCELERATION WITH EOM
alpham = (1/p1)*(Tr + M*xd*ydoubledot + M*xd*g + y*M*Sdoubledot ...
    - y*M*xddoubledot - p2*omega);
%xcdot = Sdot - xddot;
%xcddot = Sdoubledot - xddoubledot;
%T = M*xcddot;

%CHECK FOR TRIGGER MOMENT
actuate = 0;

if (gammalast < thetam && gamma > thetam)
    omegam = omega;
end
```

```
if (gammalast < thetat && gamma > thetat)
    m = m + 1;
    actuate = 1;
    alast = a;
    n = 1;
end

gammalast = gamma;

%INITIATE ACTUATION, COMPUTE TARGET ACTUATION LENGTH
if actuate == 1 % gamma crossed threshold
    ts = t;
    error = (ks*omegad - omegam);
    morphnumber = morphnumber + 1;
    z = mod(morphnumber,2); %even

    if z == 1 % a is trigger, b is sheer axis
        b = 0.3335 - kf*error; %not going fast enough
        if error > 0 %going too fast
            b = 0.3335;
        end
        if omega > 0
            b = 0.3335  ;
        end
        if b > axismax
            b = axismax;
        end

        A = 6;
        B = 8*b - 6*Circumference/pi;
        D = Circumference*Circumference/(pi*pi) - ...
            6*Circumference*b/pi + 6*b*b;
```

```
    atarget = -B/(2*A) + (1/(2*A))*sqrt(B*B - 4*A*D);
else % b is trigger, a is sheer axis
    atarget = 0.3335  - kf*error; %not going fast enough
    if error > 0 %going too fast
        atarget = 0.3335;
    end
     if omega > 0
        atarget = 0.3335;
    end
end

if atarget > axismax
    atarget = axismax;
end

if atarget < axismin
    atarget = axismin;
end

da = atarget - a;

if (da > 0)
    tactuate = -113.37*da*da + 36.05*da + 0.0106;
end

if (da <= 0)
    tactuate = -792.34*da*da - 44.82*da + 0.0412;
end

if (abs(da) > 0.003)
    flag = 1;
    Vm = (4/3)*da/tactuate;
```

```
    end
end

%ACTUATION
if (flag == 1)
    ta = t - ts;
    if (ta <= tactuate/4)
        a = alast + 2*Vm*ta*ta/tactuate;
        adot = (4*Vm/tactuate)*ta;
        adoubledot = 4*Vm/tactuate;
    elseif (ta > tactuate/4 && ta <= 3*tactuate/4)
        a = alast - tactuate*Vm/8 + ta*Vm;
        adot = Vm;
        adoubledot = 0;
    elseif (ta > 3*tactuate/4 && ta < tactuate)
        a = -(2*Vm/tactuate)*(ta -3*tactuate/4)^2 + Vm*(ta - tactuate)...
            + alast +(7/8)*Vm*tactuate;
        adot = -2*(2*Vm/tactuate)*(ta -3*tactuate/4) + Vm;
        adoubledot = -2*(2*Vm/tactuate);
    else
        alast = a;
        adot = 0;
        flag = 0;
    end
    angle1 = atan(adot);
    bdot = tan(-angle1);
end

%PLOTTING
param4 = -omega;
param5 = Tr;
param6 = a;
```

```
param7 = gamma;
param8 = N;
param9 = xd;
```

Listing 3: Control Program

```
#include <Wire.h>
#include <I2C_Anything.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_LSM303_U.h>
#include <Adafruit_Simple_AHRS.h>
#include <Adafruit_L3GD20_U.h>
#include <VarSpeedServo.h>

Adafruit_LSM303_Accel_Unified  accel(30301);
Adafruit_LSM303_Mag_Unified  mag(30302);
Adafruit_L3GD20_Unified gyro = Adafruit_L3GD20_Unified(20);
Adafruit_Simple_AHRS  ahrs(&accel, &mag);

VarSpeedServo theServo;

#define SLAVE_ADDRESS 0x58
#define SIZE (sizeof energy)

// control constants that are changed by the user
int thetat = 45; //<--
int thetam = 35; //<--
float kf = 0.02;
float ks = 1.15; //<--
float kbr = 0;
float omegad = -2.2;
float triggerS = 8.8;

// other constants and variables
int actuate = 0;
int allowActuation = 1;
unsigned int delayTime;
```

```
int j = 0;
int m;
int morphnumber = 0;
unsigned long now1;
unsigned long now2 = 0;
unsigned long now3;
unsigned long now4;
int omegaflag = 0;
int p = 0;
int posA;
int printflag = 1;
int q = 0;
int rollSwitch;
int rollSwitch_last = 0;
int switchHitNumber = 0;
int T = 10; // sample perios in ms
float A = 6.0;
float angleStart;
float atarget;
float average[30];
float axismax = 0.349;
float axismin = 0.319;
float a = axismax;
float alast = axismax;
float b;
float B;
float yCGstart;
float Circumference = 2.10;
float D;
float da;
float energy;
float error;
```

```
float gamma;
float gamma_last;
float k;
float M = 0.940; // kg
float omega;
float omegaAvg = 0;
float omegam = 0.00;
float omega_last = 0;
float Rave = (axismin + axismax)/2.0;
float runningAverage;
float runtime = 0.00;
float S;
float sum = 0;
float sumlast = 0;
float t;
float tactuate;
float theta;
float theta_last;
float tlast = 0;
float total;
float Vm;

void setup()
{ pinMode(11, INPUT); // voltage out of touch sensor into Uno
  pinMode(2, OUTPUT); // signal from Uno to energy sensor
  pinMode(10, OUTPUT); // voltage from Uno into touch sensor
  pinMode(13, OUTPUT); // led

  digitalWrite(2, LOW);
  digitalWrite(10, HIGH); // voltage into touch sensor
  theServo.attach(9);
  theServo.slowmove(570, 40); // initial condition of a
```

```
  delay(4000);

  accel.begin();
  mag.begin();
  gyro.begin();
  sensors_vec_t orientation;
  ahrs.getOrientation(&orientation);
  theta_last = -orientation.roll;
  angleStart = theta_last;

  Serial.begin(57600);
  Wire.begin();

  for (int i = 0; i < 30; i++) average[i] = 0.00;

  Serial.println("␣");
  Serial.print("omegad␣=␣");
  Serial.print(omegad);
  Serial.print("␣␣␣thetat␣=␣");
  Serial.print(thetat);
  Serial.print("␣␣␣thetam␣=␣");
  Serial.print(thetam);
  Serial.print("␣␣␣kf␣=␣");
  Serial.print(kf, 3);
  Serial.print("␣␣␣ks␣=␣");
  Serial.println(ks);
  Serial.println("␣");
}

void loop()
{ if (j == 0)
  { digitalWrite(13, HIGH); // led
```

```
  digitalWrite(2, HIGH); // send signal to energy sensor to begin
  now1 = micros();
}

while (runtime <= 15)
{ sensors_event_t event;
  gyro.getEvent(&event);
  omega = -event.gyro.x; // gyro data in rad/sec
  if (omega > 0) omega = 0.00; // omega is always negative or zero
  if (omega <= omegad) omegaflag = 1;

  // monitor if angular velocity is less than desired
  // compute running average of angular velocity
  if (omegaflag == 1)
  { p = p + 1;
    sum = sumlast + omega;
    omegaAvg = sum/p;
    sumlast = sum;
  }

  // read position switch signal
  if (digitalRead(11) == LOW)
  { rollSwitch = 1;
  }
  else
  { rollSwitch = 0;
  }

  // compute angular position by integrating angular velocity
  // and updating with position switch data
  if ((rollSwitch == 1) && (rollSwitch_last == 0)) //1 = switch activation
  { switchHitNumber = switchHitNumber + 1;
```

```
  theta = -180.00*switchHitNumber;
}
else
{ theta = theta_last + 0.5*(0.010)*(omega_last + omega)*57.2958;
}

omega_last = omega;
rollSwitch_last = rollSwitch;
theta_last = theta;
gamma = round(theta)%90;

if (gamma <= -thetam && gamma_last > -thetam) omegam = omega;

// compute touchpoint location
if (switchHitNumber == 0)
{ S = (angleStart - theta)*(Rave/57.2958);
}
else
{ S = (PI + angleStart/57.2958)*Rave + (switchHitNumber - 1)*Circumference/2
  - (theta/57.2958)*Rave - switchHitNumber*PI*Rave;
}

//initiate actuation?
if (gamma <= -thetat && gamma_last > -thetat && allowActuation == 1)
{ error = ks*omegad - omegam;
  if (error >= 0) k = kbr;
  if (error < 0) k = kf;
  morphnumber = morphnumber + 1;
  m = morphnumber%2;

  if (m == 1) // b is actuation axis
  { b = Rave - k*error;
```

```
    B = 8.0*b - 6.0*Circumference/PI;
    D = Circumference*Circumference/(PI*PI) - 6.0*Circumference*b/PI + 6.0*b*b;
    atarget = -B/12.0 + (1.0/12.0)*sqrt(B*B - 24.0*D);
}
else // a is actuation axis
//compute target position
{ atarget = (axismax + axismin)/2 - k*error;
}

if (atarget > axismax) atarget = axismax;
if (atarget < axismin) atarget = axismin;
if (S >= triggerS)
{ atarget = 0.3335;
  allowActuation = 0;
}

if (2*atarget > 0.667)
{ posA = round(-29500*(2*atarget - 0.667) + 1470);
}
else
{ posA = round(-31950*(2*atarget - 0.667) + 1470);
}

alast = a;
da = atarget - alast;
tlast = 0;

// move only if error is big enough
if (abs(da) > 0.003 || allowActuation == 0)
{ theServo.slowmove(posA, 75);
  actuate = 1;
  // compute actuation time duration
```

```
    if (da <= 0) // for A diagonal
    { tactuate = -792.34*da*da  - 44.82*da + 0.0412;
    }
    else
    { tactuate = -113.37*da*da  + 36.05*da + 0.0106;
    }

    //if (tactuate < 0.20) tactuate = 0.20;

    Vm = 4.00*da/(3.00*tactuate);
  }
}

// compute predicted value of a
if (actuate == 1)
{ t = tlast + T/1000.00;
  tlast = t;
  if (t <= tactuate/4.00)
  { a = alast + 2.0*Vm*t*t/tactuate;
  }
  if (t > tactuate/4.00 && t <= 0.75*tactuate)
  { a = Vm*t + alast - Vm*tactuate/8.00;
  }
  if (t > (3.00/4.00)*tactuate && t < tactuate)
  { a = -2.0*Vm*(t - 0.75*tactuate)*(t - 0.75*tactuate)/tactuate
    + Vm*(t - tactuate) + alast + (7.0/8.0)*tactuate*Vm;
  }
  if (t >= tactuate)
  { actuate = 0;
    alast = a;
    tlast = 0.00;
  }
```

```
}

gamma_last = gamma;

runtime = j*T/1000.00; // in seconds

int n = j%10;
if (j == 0 || n == 0)
{ for (int i = 0; i < 29; i++)
  { average[i] = average[i+1];
    total += average[i];
  }

  runningAverage = (total + omega)/30.00;
  average[29] = omega;
  total = 0.00;

  // send data to remote laptop
  Serial.print(runtime);
  Serial.print(",␣");
  Serial.print(gamma);
  Serial.print(",␣");
  Serial.print(-omega);
  Serial.print(",␣");
  Serial.print(S, 3);
  Serial.print(",␣");
  Serial.print(a, 4);
  Serial.print(",␣");
  Serial.print(omegaAvg);
  Serial.print(",␣");
  Serial.println(runningAverage);
}
```

```
    j = j + 1;

    // make sure period is equal to 10 ms
    now2 = micros();
    delayTime = T*1000 - (now2 - now1);
    if (delayTime >= T*1000) Serial.println("sample␣period␣error");
    delayMicroseconds(T*1000 - (now2 - now1));
    now1 = micros();
  }

  // 15 s run is over
  theServo.detach();
  digitalWrite(13, LOW);
  digitalWrite(2, LOW);
  delay(15);

  // send energy values to remote laptop
  if (Wire.requestFrom(SLAVE_ADDRESS, SIZE) == 0)
  { Serial.println("energy␣sensor␣error");
  }
  else if (q < 151)
  { if (q == 0) Serial.println("␣");
    I2C_readAnything(energy);
    Serial.print(q*0.1);
    Serial.print(",␣");
    Serial.println(energy, 4);
    q++;
  }
}
```

Listing 4: Median Testing Program

```
u1 = 0.067; %largest uncertainty, first population
u2 = 0.067; %largest uncertainty, second population
nc1 = 0;
nc2 = 0;
p = 100000; %number of median tests performed

for i = 1:p %perform perturbation of populations
    matrix1 = [2.78     2.73 3.04 2.90 3.06     2.53 2.78 2.88 2.90     2.79];
    matrix2 = [2.48     2.86 3.08 3.13 3.21     2.70 2.92 2.78 2.85     2.80];
    for j = 1:10
        random1 = normrnd(0,u1/3);
        while (abs(random1) > u1)
            random1 = normrnd(0,u1/3);
        end
        random2 = normrnd(0,u2/3);
        while (abs(random2) > u2)
            random2 = normrnd(0,u2/3);
        end
        matrix1(j) = (round(100*(matrix1(j) + random1)))/100;
        matrix2(j) = (round(100*(matrix2(j) + random2)))/100;
    end

    %find grand median and perform median test
    matrix3 = horzcat(matrix1, matrix2);
    M = median(matrix3);
    count1 = 0;
    count2 = 0;

    for k = 1:10
        if (matrix1(k) <= M)
            count1 = count1 + 1;
```

```
          end
      end

      for m = 1:10
          if (matrix2(m) <= M)
              count2 = count2 + 1;
          end
      end

      ChiSquareStat = ((count1 - 5)^2)/5 + ((count2 - 5)^2)/5 ...
          + ((10 - count1 - 5)^2)/5 + ((10 - count2 - 5)^2)/5;

      if (count1 > 5 && ChiSquareStat > 2.706)
          nc1 = nc1 + 1; %augmented variable for first population
      end

      if (count2 > 5 && ChiSquareStat > 2.706)
          nc2 = nc2 + 1; %augmented variable for second population
      end
  end

  %display the largest augmented variable w corresponding pass percentage
  if (count1> count2)
      disp('martix1');
  else
      disp('matrix2')
  end

  nc = max(nc1, nc2);
  disp(100*nc/p);
```

VITA

Michael Gerald Puopolo

Candidate for the Degree of

Doctor of Philosophy

Thesis: LOCOMOTION OF A CYLINDRICAL ROLLING ROBOT WITH A SHAPE CHANGING OUTER SURFACE

Major Field: Mechanical and Aerospace Engineering

Biographical:

Michael Puopolo was born in Portland, ME, on January 13, 1972. He received the B.S. degree from University of California at Riverside in 1994 with a major in Physics, and the M.S. degree from California Polytechnic State University, San Luis Obispo in 1998 with a major in Engineering. Between 1998 and 2000, Michael worked as a Research Design Engineer with Space Systems/Loral in Palo Alta, CA. Between 2003 and 2010, he worked as a Senior Project Engineer with ACTA Service Corporation at Vandenberg Air Force Base. Michael returned to school in 2010 to pursue doctoral studies at Oklahoma State University in Mechanical and Aerospace Engineering. As a doctoral student, he served as an instructor of record for two engineering courses and authored papers on robotics and aerodynamics.