CONTINUOUS WAVE PERISTALTIC MOTION IN A ROBOT

by

ALEXANDER BOXERBAUM

Submitted in Partial Fulfillment of the requirements

For the Degree of Doctor of Philosophy

Thesis Advisor: Roger Quinn

Department of Mechanical Engineering

CASE WESTERN RESERVE UNIVERSITY

May, 2012

SCHOOL OF GRADUATE STUDIES

We hereby approve the thesis/dissertation of

Alexander Boxerbaum

candidate for the Doctor of Science degree *.

(signed) Roger Quinn
(chair of the committee)

Hillel Chiel

Malcolm Cooke

Joseph Mansour

(date) 3-30-12

For My Parents:

Elliot and Felicity

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

This work would not have been possible without the support of countless friends, family and colleagues. More than anyone, my parents have helped me get to where I am today, and this work is dedicated to them. I owe almost as much to my advisor, Roger Quinn, who was willing to take a chance on an art history major who wanted to become an engineer. His guidance through the years has always been helpful and calming. My co-advisor, Hillel Chiel, has shared with me his enthusiasm and attention to detail. I would have certainly gone crazy if it weren't for Nicole Kern and her unwavering moral, intellectual and personal support. Much of this dissertation was developed through grant writing and academic publications and was funded by the Arthur P. Armington Chair and the NSF grant, IIS-1065489. My advisors and co-authors, Andrew Horchler, Kendrick Shaw and Kathryn Daltorio all have made invaluable contributions to the work presented here. Professors Joseph Mansour and Malcome Cooke have also been very helpful as members of my committee. I would also like to thank Ravi Vaidyanathan, Iwan Alexander, Joe Prahl, Kiju Lee, Roy Ritzmann, Steven Garverick, Ilya Minsky, Annette Poluse, Zac Jesse, Richard Bachmann, Mathew Klein, Kenneth Moses and the entire Biorobotics Lab, past and present.

Continuous Wave Peristaltic Motion in a Robot

Abstract

by

ALEXANDER BOXERBAUM

This dissertation is a study of peristalsis, the method of locomotion earthworms use, and how to best achieve this in a robotic platform. A technique is presented that uses a braided mesh exterior to produce smooth waves of motion along the body of a worm-like robot. This braided mesh can be powered by a one degree-of-freedom cam mechanism, which is demonstrated, or by several independent motors. A new analytical model of peristalsis is presented and predicted robot velocity is compared to a 2-D simulation and a working prototype. It has been often assumed that this motion requires strong anisotropic ground friction. However, our analysis shows that with uniform, constant velocity waves, the forces that cause accelerations within the body sum to zero. Instead, transition timing between aerial and ground phases and the ability to generate strain play a critical role in the final robot speed. Lastly, we present a soft-body controller that uses simulated neuronal populations. This controller is designed for the next generation of soft, hyper-redundant systems and can intrinsically generate waves of a desired behavior while smoothly incorporating large amounts of simulated sensory input.

Figure 1: A robot that creates peristaltic motion with a continuously deformable exterior surface.

1 Introduction

1.1 How It All Got Started

My involvement with in soft robotics began when I was asked to make a computer rendering of a possible robot for a grant solicitation called the DARPA ChemBot project. The solicitation called for a robot that could change shape dramatically and potentially navigate complex environments. When I joined the group, the concept was to make a long inchworm-like robot that consisted of several large soft segments actuated by shape memory alloy spring actuators (SMAs) and inflatable braided mesh structures (Figure 2). The robot could roll up into a tight ball in order to achieve the desired form-factor change. This concept was based strongly on an endoscopic assist robot designed by Elizabeth Mangan under Professors Chiel and Quinn (Chapter 2.3).

Figure 2: A computer rendering I made of an early concept for the DARPA ChemBot project. Two and a half segments are shown squeezing under a door. The pad at the front has directional spikes for inchworm-like attachment and locomotion. An apparatus at the front is based on the *aplysia* mouth and could take small samples from the environment.

As I worked through the design of this next-generation robot, several things stood out as being problematic. We intended to actuate each air-muscle with a micro-valve and pump system that was going to difficult to fit into a robot with a 1 cm diameter, which was

one of the requirements of the grant. We also knew that this method of locomotion generated a lot of slip. Even with an offboard power supply, the previous robot moved very slowly (Chapter 2.3) [Mangan 2002]. Lastly, the motion seemed distinctly un-wormlike to me. Worms have up to 150 segments that form only one to two waves, so the wave moves down the body very smoothly. However, the segments of this robot were relatively long, with as few as four segments along the length of the body. This created dead zones between actuated areas where the segments were attached. If in unstructured environments, these areas were to touch the ground, they would probably get stuck. These problems pushed me to rethink how worms achieve their motion.

At one of our grant meetings, it occurred to me that we could simply do away with the segments, and squeeze a continuous braided mesh using hoop-shaped SMAs. I imagined that a bolus of fluid would move along the body, squeezed by the SMAs at the trailing edge. This bolus would expand the SMAs at the leading edge of the wave (Figure 3). We did a good deal of analysis to show that this fluid could circulate through the robot effectively, and that the SMAs could be actuated fast enough to generate the desired speed.

Getting power to the SMA actuators presented challenges. The soft body of the robot is constantly changing length, and running wires along them would be tedious at any scale. However, the braided mesh itself defines a path of zero strain, so it made sense to run the wires along this path. But at that point, it makes sense to make the braided mesh out of the wiring harness itself and save on weight. This robotic concept is described in detail in Chapter 4.2.

Figure 3: Final concept robot design for the DARPA ChemBot grant. The robot could roll up into a ball, and squeeze under a door frame. Unlike our first design, the peristaltic waves would move smoothly down the body.

As we considered the design further, we began to realize the many drawbacks of SMAs. They are at best 10% efficient, and often very slow. They also need a force in order to return them to their extended state, and get as hot as 150C, hot enough to melt many materials. They may also have a short fatigue life. One possible alternative that I explored used micro-hydraulic tubing to run the hoop actuators. The hydraulic actuators would expand the hoops against a return spring. The tubing would also act as the braided mesh, similar to how the SMA concept used the wiring harness as a braided mesh. This would also allow a single hydraulic pump at the end of the robot. This concept is discussed in Chapter 4.3.

Finally, I came up with a cable driven system that would be a dynamic inverse of the hydraulic arrangement. Hoop actuator cables would squeeze the robot against a lengthwise return spring, which would be coupled due to the braided mesh. These cables could be routed through the hollow braided mesh and controlled at the end of the robot with a single cam mechanism. With the lengthwise return springs, the bolus of fluid from earlier versions was no longer needed, leaving the entire robot interior free for payload space.

While I often received enthusiastic responses when I presented our worm robot concept at conferences, some people expressed skepticism that it would work. Since I suspected this was part of the reason finding grant money was so difficult, it made sense to simply go ahead and try to build a prototype. I knew building a very small scale prototype would be much more difficult, so I chose a scale that could be assembled by hand with off the shelf components to demonstrate the principle of motion and the actuation method. This design went through several iterations, resulting in the final prototype (Figure 1). All of Chapter 5 is dedicated to describing this design process.

While several groups have studied peristalsis, we found the previously developed analytical tools lacking for an implementation like ours that could generate smooth waves of motion. We therefore began to develop a differential model of peristalsis that could be understood using integrals, as opposed to large summation techniques. Several interesting and counterintuitive conclusions can be drawn from the model, including that the internal forces due to the accelerating and decelerating segments can cancel. I also modified the idealized model to account for losses due to both slip and the mechanical coupling within

segments (loss of strain). This approach is detailed in Chapter 3, and I compare the analytical model to the motion of the robot in Chapter 9.

In contrast to the continuum assumptions of the analytical model, I also developed a 2-D dynamic simulation using Working Model 2D (Design Simulation Technologies, version 9.0). Each simulated body segment consists of a modified four-bar mechanism, where each bar is split into three pieces joined by a torsional spring. This approximates the ability of the braided mesh to bend, an essential capability for wave formation. The number of segments tested ranged between six and twelve. Each 'muscle' or actuator was simulated using a stiff spring-damper system, in which the rest length of the spring was driven by a periodic function. This method is described in Chapter 7 and compared to the analytical model in Chapter 9.

We thus have three different ways of studying peristalsis: an analytical model, a 2-D dynamic simulation and a physical model. Each approach highlights different properties of peristalsis, and I discuss and compare them in Chapter 11.

1.2 A Brief History of a Soft Body Controller

With the success of the final robot, we were strongly encouraged to continue our study of peristaltic locomotion. While the prototype worked very well on flat ground, it could only passively adapt to the terrain. More complex environments require active turning and rearing. Towards this end, I began to work on soft-bodied control architectures with two other graduate students, Andrew Horchler and Kendrick Shaw. Together with our advisors Roger Quinn and Hillel Chiel, we wrote an NSF grant aimed at developing a new kind of controller based on simulating neuronal populations. In the long term, we aim

to model these populations using a set of dynamical systems called stable heteroclinic channels (SHCs).

I had previously developed a model of the human primary visual cortex using the Wilson Cowan model of neuronal behavior (Chapter 2.6). This method has many features in common with SHCs, but characterizing a given Wilson-Cowan network as such is problematic. Wilson Cowan neuronal models use very diffuse connections among large, continuous groups of neuronal populations, which makes the process of mathematically verifying limit cycles or heteroclinic channels difficult or impossible. Nonetheless, my familiarity with the dynamical properties of Wilson Cowan models allowed me to identify a straightforward way to develop a soft-body controller that could generate a diverse and controllable set of patterned activation (Chapter 8). I was able to test this control algorithm on a quasi-static model of a worm robot with up to 150 degrees of freedom. This allowed me to simulate stretch receptors in the worm, which likely play a key role in wave propagation. By blending stretch receptor signals with the existing dynamics of the control network, I as able to show how such a signal could possibly modulate patterned muscle activations (Chapter 9) and increase efficiency over the open-loop controller. This approach will hopefully inform the morphology of a future SHC approach.

Soft-bodied robotics remains one of the most challenging areas in robotics, one that also offers great rewards. I conclude this work by outlining some avenues of future research in Chapter 12. With our recent advances in understanding, I feel that soft-bodied robots are very close to the point where they will have unprecedented locomotive abilities, and this in turn will open up many new applications.

Over the course of my research, I have transitioned from studying the mechanics of peristalsis to studying the control of it. While these two areas are somewhat distinct, it has become clear to me how one has led into the other. Many times, insights gained from studying the mechanics of the motion directly informed the structure of the control network. I hope this work will help advance the state of the art in both the mechanics and control of soft-bodied robots.

2 Background

2.1 Potential Applications

There is much to learn from studying soft-bodied invertebrates, such as leeches, worms, caterpillars and slugs. There are countless applications for a robot that could begin to approach the level of performance these animals achieve. This research will focus mainly on peristaltic crawling. At a larger scale, such a robot could be used for water main maintenance, where the robot could navigate within a functioning water main to detect leaks. It could be used for search and rescue, where the robot could move through rubble piles, and push large obstacles out of the way where need be. It could be used to diagnose large complex machinery without disassembly, such as jet engines and airplane wings, which often have fuel tanks that need to be regularly inspected and cleaned. A wormlike cable sleeve could be used to prevent robotic cable tethers from getting hung up on obstacles during search and rescue missions. Likewise, a wormlike fire hose would also benefit from this.

At a smaller scale, a wormlike robot could be used for reconnaissance within a building. It could be used to burrow through soil to install deep cables. It could be used to take samples deep underground with little disturbance to the environment. Several biomedical applications also likely exist. One could use a peristaltic robot as an endoscopic assist, so that the endoscope is being gently pulled instead of pushed through the large intestine. It could also be used to explore the small intestine in a similar manner, or swallowed as a camera pill that can actually locomote and take biopsies. One could

even possibly make a worm robot small enough to travel within arteries and act as a self-deploying stent. These are just a few of the possible applications that we have come across over the years, but there are likely many more. It is clearly an area of research worth pursuing.

There are several areas of ongoing research that have important consequences to this work. It is important to have a familiarity with the desired behavior of the animal (Chapter 2.2), and recent robotic implementations (Chapter 2.3). Some knowledge of the neuroethology of animals that exhibit the behavior is important (Chapter 2.4), as well as attempts to model the control networks in the animal (Chapter 2.5). Lastly, a familiarity with the dynamic properties of neural control circuits is needed (Chapter 2.6).

2.2 STUDIES OF PERISTALTIC LOCOMOTION

Soft bodied animals achieve their robust and flexible behaviors by deploying muscle groups arranged in ordered configurations—longitudinally, circumferentially, or helically. These animals flexibly move in a variety of ways, including peristaltic crawling, anchor-and-extend, and swimming [Brusca 1990]. These soft structures can be grouped into two categories: *hydrostatic skeletons* [Skierczynski 1996] and *muscular hydrostats* [Kier 1985]. Hydrostatic skeletons have a central fluid-filled cavity. Contraction of a muscle component of the cavity induces an expansion of other parts of the cavity and of its surrounding muscle, allowing the fluid to act as a mechanical power transmission. In contrast, muscular hydrostats, such as tongues, trunks and tentacles have no central fluid filled cavity and therefore have higher power-to-mass ratios. Robots modeled after hydrostatic skeletons could potentially replace a fluid filled cavity with large payload

space, while robots modeled after muscular hydrostats would likely have a better power to weight ratio.

Peristaltic locomotion has several interesting properties that at first might seem counter-intuitive. The waves of expansion and contraction flow in the opposite direction of the robot motion [Gray 1938]. This is a direct result of the anisotropic strain properties of the body [Alexander 2003]. When a section leaves the ground, a new ground contact point forms directly behind it. The contracting section will accelerate outward axially, but that motion is constrained on the rear side by the new ground contact point, so the segment must move forward (Figure 4).

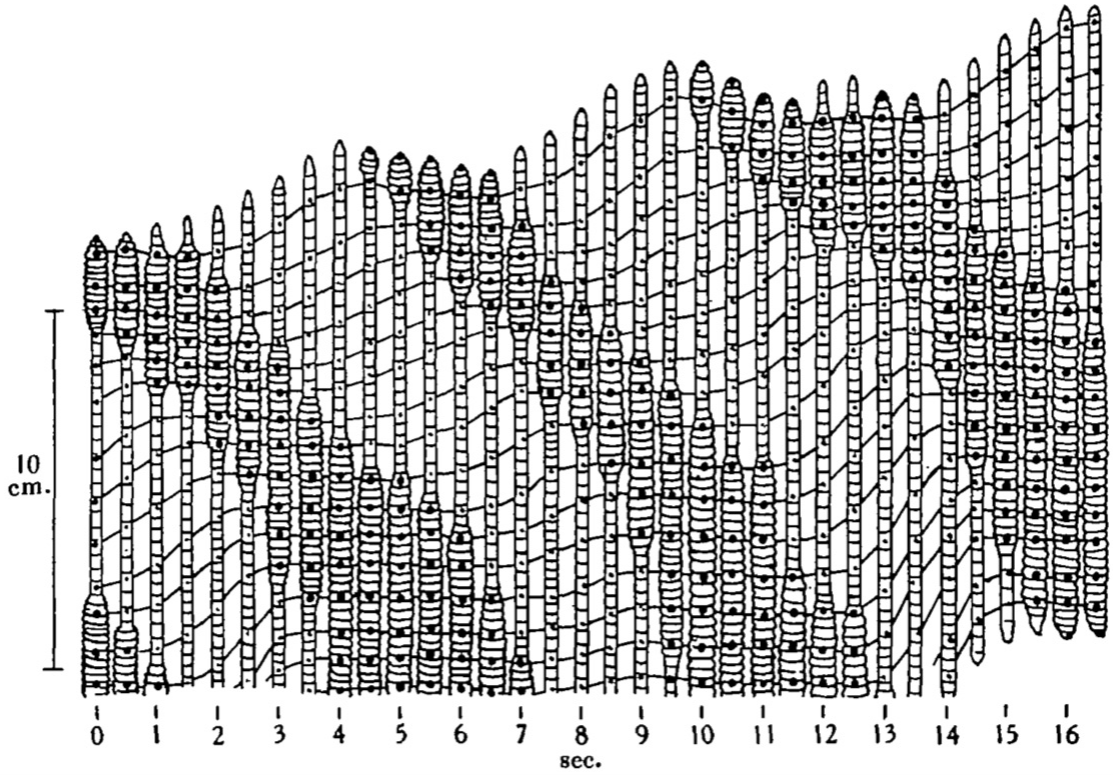

Figure 4: An illustration from the first known motion-capture of worm peristalsis (out of copyright) [Gray 1938]. Rearward travelling waves (down) create a forward progression (up) over time (horizontal axis). Note that while the head appears to slip backwards, it is likely not ever touching the ground. In contrast, the segments along the rest of the body do not slip backwards. The progression of given segments is shown in wavy lines from left to right, where segments in ground contact are shown with a dark dot.

Chapman investigated the role of the body fluid in the locomotion of the earthworm. Using a mathematical model of an incompressible fluid, he predicted the strength relationship between longitudinal and circumferential muscles. He then took measurements from earthworms of the cross-sectional areas of both and found that the longitudinal muscles have ten times the mechanical advantage as the circumferential muscles. The author suggests that this may be in part due to the need to widen the hole that the earthworm travels through, and generate friction [Chapman, 1949]. These results are also consistent with much later findings that during burrowing, the soil is pushed almost entirely perpendicular to the motion path [Barnett 2009]. Worms burrow by using their bodies as thin wedges. Longitudinal muscles may also be significantly larger because of their involvement with a thrashing escape response. Chapman's group found a theoretical relationship between length and circumference based on a hydrostatic fluid. They noted that this relationship gives the worm its maximum mechanical advantage when the circumference is large (ie, fat worms), but the strain is greatest when the worm is already very narrow and long [Chapman, 1949]. We will see later that this relationship is inverted with a braided mesh approximation of a hydrostatic body.

The kinematics [Quillin 1999] and dynamics [Quillin 2000] of earthworm locomotion have been most recently studied in detail by Quillin, who defines worm speed as:

$$\text{Speed} = \frac{\text{Stride Length}}{\text{Cycle Time}} \qquad (2\text{-}1)$$

While this observational model accurately characterizes earthworm locomotion on flat ground, it does not capture or explain the causes of slippage, and therefore tends to

overestimate the predicted speed of worm-like robots. For this reason, an observational measure of 'efficiency' is frequently used to describe the discrepancy, defined as the ratio of the effective stride length over the predicted stride length [Zarrouk, et al. 2010].

$$\text{Efficiency} = \frac{\text{Observed Stride Length}}{\text{Predicted Stride Length}} \qquad (2\text{-}2)$$

There are many factors that affect the speed and efficiency of peristalsis. Alexander has focused on the need for anisotropic friction, where the worm or robot is always touching the ground, but slides forward with less resistance than when it slides backwards. The setae of earthworms contribute to this asymmetry, as well as the decreased diameter of the worm during forward motion. He suggests that because of the small mass of earthworms (1-9 grams), these friction forces dominate over any inertial forces. [Alexander 2003]. While there are certainly times when this is the case, his analysis assumes that the forward moving coefficient of friction is on the order of 0.1, whereas it may in fact be zero if the worm is lifting the segments off the ground. Furthermore, because the robots that we are aware of weigh significantly more than an earthworm, the dynamic effects on robots are likely much more substantial.

Zarrouk et al. take a similar approach to Alexander, but also model the slip caused by the flexibility of the substrate (such as an artery wall, intestine, or possibly soil). They use a quasi-static approach, and assume that the accelerations and decelerations of the mass of the robot do not affect the motion. In this model, friction between the worm and the substrate is needed to overcome external forces such as gravity, and forces caused by dragging the forward-moving segments across the substrate. These forces push the body backwards proportional to the stiffness of the substrate [Zarrouk 2010].

Both of these analyses leave open the possibility that the dynamic forces within the worm could potentially cancel, meaning that peristalsis would only require friction (or ground reaction forces) in order to accelerate or borrow. The conditions that make this possible are explored in Chapter 3.

2.3 ROBOTS THAT ATTEMPT PERISTALSIS

Designing, building and controlling hyper-redundant robots is challenging [Tesch 2009, Transeth 2009]. Therefore, robotic implementations typically simplify the problem by reducing or grouping the degrees of freedom [Menciassi 2004, Lee 2010, Trimmer 2006], and or by replacing continuously deformable soft bodies with rigid joints [Wang 2007, Omori 2009]. These simplifications come at a cost to multifunctional flexibility and performance. Vertebrates such as snakes and salamanders are also hyper-redundant, but not soft, and for this reason, robotic implementations inspired by them have had more success [Hirose 1993, Ijspeert 2007]. However, snakelike undulations do not work in the most confined spaces or where burrowing is required. I will briefly discuss several robots, which in some cases were developed concurrently or after our robot prototypes.

The Biorobotics group at Case Western has a long history of working with soft robotics. An underwater Shape Memory Alloy robot with a hydrostatic skeleton was developed by Ravi Vaidyanathan and my co-advisors, Roger Quinn and Hillel Chiel. It consisted of three segments, each of which had the ability to actuate left-right and up-down using bands of Nitinol wire. The robot had a maximum speed of 0.6 cm/second, or 2.5 body lengths per minute, but this required a combination of angled spiked feet on

Styrofoam that allowed it to generate large anisotropic friction. The robot also required a large offboard power supply. Nonetheless, it may have been the first underwater hydrostatic robot, and it was able to turn and navigate around obstacles [Vaidyanathan 2000].

Based on this work, a wormlike robot was developed using long artificial muscles in series (Figure 5). The artificial muscle consisted of a braided mesh that was used to create a material with anisotropic strain properties. Compression along one axis caused expansion in another. In this case, the material was woven into a cylindrical shape and a bladder inflated the cylinder, pushing outward radially, which caused axial contraction [Mangan 2002]. This robot had much in common with most robots attempting peristaltic motion: a small number of identical segments (often three) attached in series, each of which could alternately contract axially and expand radially [Trivedi 2008 (review), Dario 2004, Menciassi 2004, Wang 2007; Omori 2009, Seok 2010]. In these robots, the area in between each segment is not actuated. This ubiquitous trend in robot design is consistent with the way peristaltic motion is explained in the literature, in which a small number of segments are used to approximate continuous small muscle movements for clarity [Alexander 2003]. There also may be a tendency among engineers to reduce and simplify the design as much as possible. However, earthworms have as many as 150 segments generating a single wave. This allows the wave to travel smoothly down the body [Quillin 1999]. The segmented approach is more suited to modeling animals that do have large segments, such as caterpillars [Trimmer 2006] or inchworms [Vaidyanathan 2000, Cheng 2010], where whole body motion is coupled with strong anisotropic friction or gripping in order to achieve locomotion. A notable exception to a segmented soft body approach is an

amoeba-inspired robot that does not use peristaltic motion, but has a novel whole-body method of locomotion [Ingram 2005].

The previous robot by Mangan et al. [2002] moved much more slowly than expected. It would often appear to slip backwards, or have difficulty progressing when an obstacle landed between actuators. This led to a maximum robot speed of 0.3 meters per minute, where over 80% of the theoretical speed was lost. The slipping, which may be common among all robots attempting peristaltic motion, has led investigators to conclude that friction was important for this mode of locomotion [Alexander 2003, Menciassi 2004, Zimmermann 2007, Zarrouk 2010]. Also, the robot's power requirements were substantial: it required an off-board pressurized air supply. These issues were the impetus for re-evaluating our understanding of peristaltic motion and its implementation in a robotic platform.

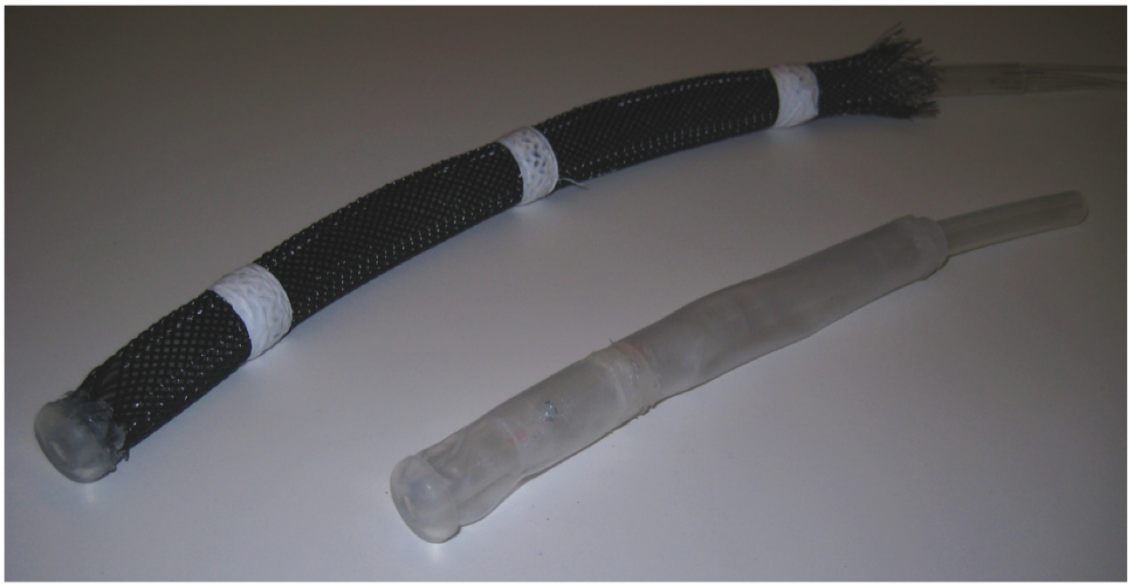

Figure 5: (Left) A previous worm-like robot with discrete actuators surrounded by a braided mesh. (Right) The inner actuator core that inflates the mesh [Mangan, et al. 2002].

After we began presenting many of the ideas developed in this thesis, another group began exploring the use of a braided mesh and SMA actuators. Instead of using a bolus of fluid to expand the SMAs as we proposed, they chose to fix the total length of the robot with a rigid structure so contracting elements forced other expanding elements open. This approach had some success, but could not return the hoop actuators to their original rest length. And instead of creating a smooth wave of motion, like many other groups they chose to simplify the design by only having five segments along the body [Seok 2010]. While their overall design proved to be very robust to rough treatment, the above design choices likely contributed to slip. Their final robot speed was 0.2 meters per minute, or 1 body length per minute.

2.4 Studies of the Neuroethology of Peristalsis

The neuroethology of earthworms has been studied in some detail. I will outline several findings that contribute to the structure of the simulated neuronal controller that is developed in Chapter 8.

Many early studies relied on novel techniques for deducing some properties of the system. In one experiment, it was shown that if a worm was cut in two sufficiently far from the head, and then a string was tied between the two parts, that the wave of motion would propagate uninterrupted from head to tail through the string (described by [Gray 1938]). Similarly, a worm suspended from its head by a string will naturally generate waves of motion. However, if the worm is dipped into a solution such that it is neutrally buoyant, this motion will stop [Gray 1938]. These findings strongly suggest that stretch receptors play a direct role in transmitting the neuro-muscular control signal down the

length of the worm. In contrast, decreasing activity has been found in the nerve cord in vitro, from head to tail during fictive locomotion [Gray, 1938]. However, this fictive locomotion did not have the same frequency, strength or regularity of an intact worm. This suggests that there may be a gradual transition from head to tail between top-down pattern generator locomotion and sensory-driven locomotion. Ultimately, the nature of the neuro-ethology of peristalsis is still not understood. Models such as the one presented in Chapter 8 could potentially play a role in understanding possible architectures.

Several sensors have been identified in the earthworm Lumbricus Terrestris. These include exteroceptors such as touch sensors (via a cuticle), chemo sensors, photo sensors, and proprioceptors such as the pressure of the coelomic fluid and stretch receptors within the muscle body. Interestingly, some studies have found that while each segment contained all of the above sensors, the ganglia within each segment respond to stimulus from several nearby segments. That is, sensory input is likely diffuse and continuous. It is also interesting to note that the pressure of one segment directly affects the pressure of an adjacent segment [Chapman 1949], providing a mechanical coupling that has consequences for the dynamics of motion, and must also affect proprioception. Lastly, the neuro-morphology of each segment is highly regular, increasing the likelihood that organized reflexes could play a key role in locomotion [Gardner 1975].

2.5 RHYTHMIC BIOLOGICALLY INSPIRED CONTROLLERS

Biologically inspired control strategies are particularly exciting because these systems in nature simply work much better than the state of the art. Animals across a broad spectrum are extremely good at integrating large amonuts of sensory input and

acting in a manner that is beneficial to the survival of the species. This method of control may be inherently non-logical. That is to say, a worm does not reason its way along as it burrows. Nor do humans, for that matter, use their logical facilities to walk down a hallway. While animals do learn, they do not have to use logic to do so. So what exactly is it that we all are doing when we navigate our enviornments? Whatever it is, it has the ability to run in the background, which makes it very powerful. We can actually reason and think *while* we walk. It is only through the development of habbits, or in this case, habitual motions, that we learn these tasks and do them well [Aristotle 1962]. This is all to say that if we want to make robotic platforms that navigate even moderately complex environments, then we should be learning how to program in this other language, the incredibly robust and adaptive language of neuro-muscular systems.

There are many major challenges to developing these control systems. It is often said that a controller needs to be robust yet adaptable [Rabinovich 2008]. That is to say, robust against noise, and still able to change behaviors. These two goals can be difficult to achieve at the same time. At one extreme, you have a controller that does one thing robustly, regardless of its environment (which can only be useful in a highly controlled environment), and at the other, you have a controller that can do anything, but never knows when to do what. As robots move out of the factory and into far more complex environments, they will need to be far more adaptable than the state of the art.

While the above distinction is useful, the situation is far more complex at the ground level of the controller. One problem is that noise is implicitly defined either as information that doesn't matter, or doesn't correlate to something in the world. This assumes one knows what matters. The random firings of neurons can play an important

role in the dynamics of the system. Furthermore, adaptablity simply means changing behaviors when it is optimal to do so (under some standard of optimality). So all of this could be simplified to say that a controller is robust and adaptable when it does what you want it to. Slightly less pessimistically, we can say that given all the information that could possibily be available to a controller, that a good controller has the ability to sense the environment enough to act in some optimal way.

Many groups have used neuroethological studies and neurally-inspired control to tackle the problem of generating robust motor coordination for hyper-redundant systems. By coupling central pattern generator (CPG) controllers to different degrees of freedom, it is possible to create coordinated rhythmic behavior in hyper-redundant robots (e.g., swimming in robotic lamprey [Ayers 2000] and knifefish [Zhang 2008], walking and swimming in a salamander robot [Ijspeert 2008], and undulatory locomotion in a snake robot [Matsuo 2008]). The CPGs are based on limit cycle oscillators that drive each segment with a stable rhythmic pattern. Adjacent segments are then coupled via a phase offset to achieve waves along the length of the body.

The approach of Wadden, et al. [Wadden 1997] has some similarities to our approach. While there are a finite number of CPG like elements, they are diffusely connected to as many as seven other elements in order to model swimming behavior in the lamprey. They use Hodgkin-Huxley model neurons, which are often more detailed than is needed for most engineering applications and require a great many parameters to be specified. This approach was able to generate smooth waves of varying frequencies.

Another group very recently presented a neuromechanical model of the nematode, *C. Elegans*. While this work was developed concurrently with our research, many of our

findings are similar, in spite of working on different animals that exhibit different motions. Nematodes locomote with an up-down wavelike whole-body motion. This species of nematode only has 302 neurons (it is about 1mm long) and the system has been completely mapped. Furthermore, the functions of many of the neurons have been explored through genetic studies and targeted cell killing. They model the entire neuronal structure within a simulated muscular structure that can navigate simulated environments such as water, agar, and soil. The neurons are simulated as binary units that are either on or off, and are switched when their synaptic inputs reach above threshold values. It was recently shown that this species of nematode smoothly change their body motion as the viscosity increases. This group postulated that stretch proprioceptors alone could result in this change of motion with no top-down modulation, and their simulation shows this may be feasible [Boyle 2012].

In order to get their network to generate waves of motion, they made several choices that mirror our work in wormlike peristaltic networks. They weakly inhibit opposing muscle groups to prevent co-contraction (and probably help wave propagation). The stretch receptors directly excite the local group of neurons when stretched. However, the stretch receptor influence is highly asymmetrical, in that it affects only neurons posterior to the stretch [Boyle 2012]. The reach of a given stretch receptor then determines the spatial wavelength. While a good deal is known about the neuro-morphology of nematodes there is no specific evidence regarding these design choices, so it is very interesting that both their group and ours have come to similar design choices.

They claim that the network itself is not a central pattern generator (CPG), which we take to mean that it does not show sustained patterned activity without sensory input

[Pearson]. While this may be true in the traditional sense of a CPG, it may be possible that the network as a whole would respond in a patterned way to tonic excitatory inputs - if the experimenters were attempting to find such a method of excitation. Nonetheless, it is interesting that only the sensory feedback is a-symmetrically connected to the network, and the internal excitatory connections are not, and in this model, it is sufficient to generate patterned activity under environmental stimuli.

2.6 WILSON-COWAN MODEL OF NEURONAL POPULATIONS

The spatially-extended Wilson-Cowan model of neural population activity was originally developed as a tool for modeling the primary visual cortex. The synaptic connections of this region of the brain have an interesting shift-twist symmetry, whereby the connections are similar among layers of the cortex if you both rotate *and* translate a given layer. By combining this symmetry with an approach that averages the activity of a population of neurons, they were able to find several analytical solutions to the differential equations that describe the time rate of change of the neuronal populations long before numerical methods became widely available [Ermentrout 1979]. These solutions describe certain stable spatial activity patterns that also correspond to visual phenomena that have been widely reported across cultures (Figure 6).

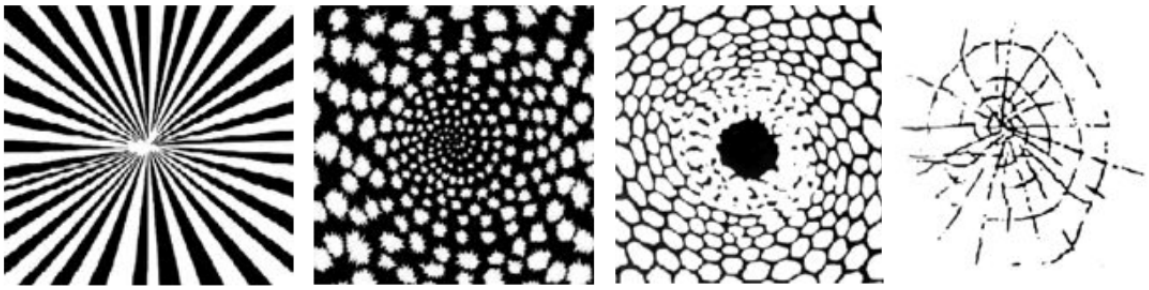

Figure 6: Graphical representations of four classes of "Form Constants" that have been observed across cultures [Bressloff 2002]. They are considered phosphenes, or visual pheonomena that do not correspond to light coming into the eye.

As part of a class in computational neuroscience, I built a numerical simulation of the primary visual cortex using Wilson-Cowan dynamics that also incorporates stochastic phenomena and simulated sensory input from the retina (Figure 7). This allowed us to find non-symmetric steady-state solutions to the state equations that also correspond to observed phenomena. While exploring this system, I discovered that shifted inhibitory connections caused lines to move over time. This gave me the idea that a similar dynamical system could be applied to generating continuous peristaltic locomotion. Since the neuro-morphology of the earthworm is highly regular, we can assume that it is shift-symmetrical, and thus use these powerful modeling techniques.

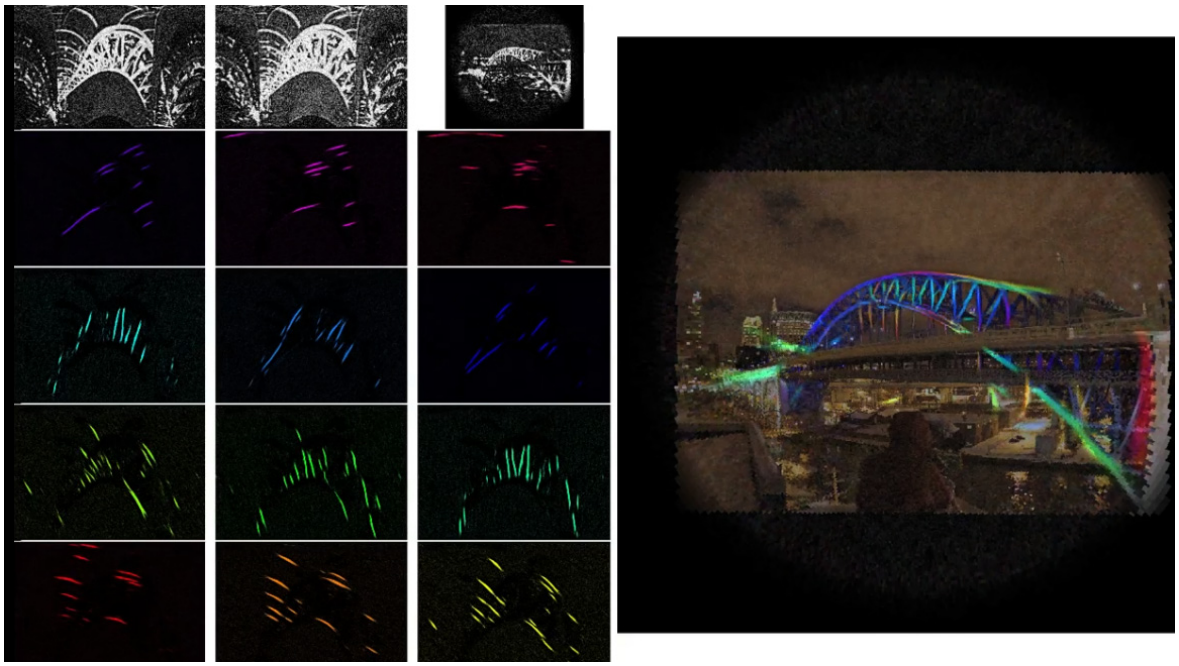

Figure 7: A screen shot of a simulation of the primary visual cortex as it is being exposed to an image of a bridge. This region of the brain finds and begins to connect line segments within an image, and tracks them over time as they move. The color is representative of the angle of the line in cortical cordinates, which the brain is able to sense.

3 A New Theory of Peristaltic Locomotion

3.1 Introduction

Once we began looking at ways of making continuously deformable structures, we realized that previous analyses of peristalsis had focused on the segmentation of earthworms. Even the most accurate previous models of earthworm locomotion [Alexander 2003] were limited in their conclusions by assuming segments were in either one of two states. While these models can have arbitrarily large numbers of segments, there has been little consideration for treating the body as a continuum structure. Earthworms can have up to 150 segments, but typically only express one to two waves per body length. In addition, the mechanical coupling of the segments tends to further smooth the waveform. For these reasons, we have developed a continuum model of peristalsis that accounts for the internal kinematics and dynamics of the motion. The derivations below begin with any waveform (strain function) and then proceed to examine temporally periodic waveforms.

3.2 A Differential Model of Position And Velocity

We can develop a differential model of peristalsis by first looking at discrete segments, and then taking the limit of small segments to reach the continuum limit. This approach was originally developed with the help of my co-advisors, Roger Quinn and Hillel Chiel, and later refined with the help of Kendrick Shaw.

Consider a small axial element on the front of the body of a worm or wormlike robot with an undeformed (resting) length of Δl (shown in Figure 8 as a small rectangle). Let us assume that the strain this element will experience can be defined by a function, $\varepsilon(l,t)$, where l is a distance from the back of the element to the head of the robot when all of the segments are experiencing zero strain (which we will refer to as undeformed coordinates). This strain is defined according to the engineering convention as

$$\frac{\Delta l^* - \Delta l}{\Delta l} \text{ , which becomes } \quad \frac{\partial l^*}{\partial l} - 1 \text{ ,}$$

(3-1)

in the continuum limit. Here Δl^* is the new deformed length of the segment under strain and l^* is the new distance to the head of the deformed robot (which we will refer to as deformed coordinates, and signify using an asterisk).

We next calculate how a deformation changes the position of the element at position L_i in undeformed coordinates. The change in length of an individual segment is just the strain times its original length, i.e. $\varepsilon(L_i,t)\cdot\Delta l$. The change in distance to the head of the worm, which we will call P^*, is the sum of the changes in length of all of the segments between this segment and the head of the worm (see figure 3, which shows how this sum is built up as the wave of deformation proceeds along the length of the robot),

$$P^*(L_i,t) = L_i^* - L_i = \sum_{j=1}^{i} \varepsilon(L_j,t)\Delta l = \sum_{j=1}^{L_i/\Delta l} \varepsilon(j\,\Delta l,t)\Delta l\,.$$

(3-2)

The rightmost equality is due to the identical undeformed length, Δl, of all of the segments, so that $L_j = j\ \Delta l$. A similar summation approach was used by Alexander

[Alexander 2003]. However, here we will take the limit as the segment size becomes small to create a continuum model. In this limit, we see that the change in position induced by the deformation is

$$P^*(L_i,t) = \lim_{\Delta l \to 0+} \sum_{j=1}^{L_i/\Delta l} \varepsilon(j\,\Delta l,t)\Delta l = \int_0^{L_i} \varepsilon(l,t)\, dl \quad . \tag{3-3}$$

Although many types of deformations are possible, traveling waves are of particular interest for peristaltic locomotion, so we will examine them in detail. We will focus on smooth, bounded deformations with a constant shape and wave velocity V_{wave} relative to the head of the worm in undeformed coordinates, so that

$$\varepsilon(l,t) = \varepsilon(l - t\,V_{\text{wave}}) \quad . \tag{3-4}$$

This is equivalent to saying that a given segment will change shape in a manner identical to the segment right in front of it, but at a later time. One can think of it as a fixed deformation wave that passes through the worm or robot at a constant speed, V_{wave}.

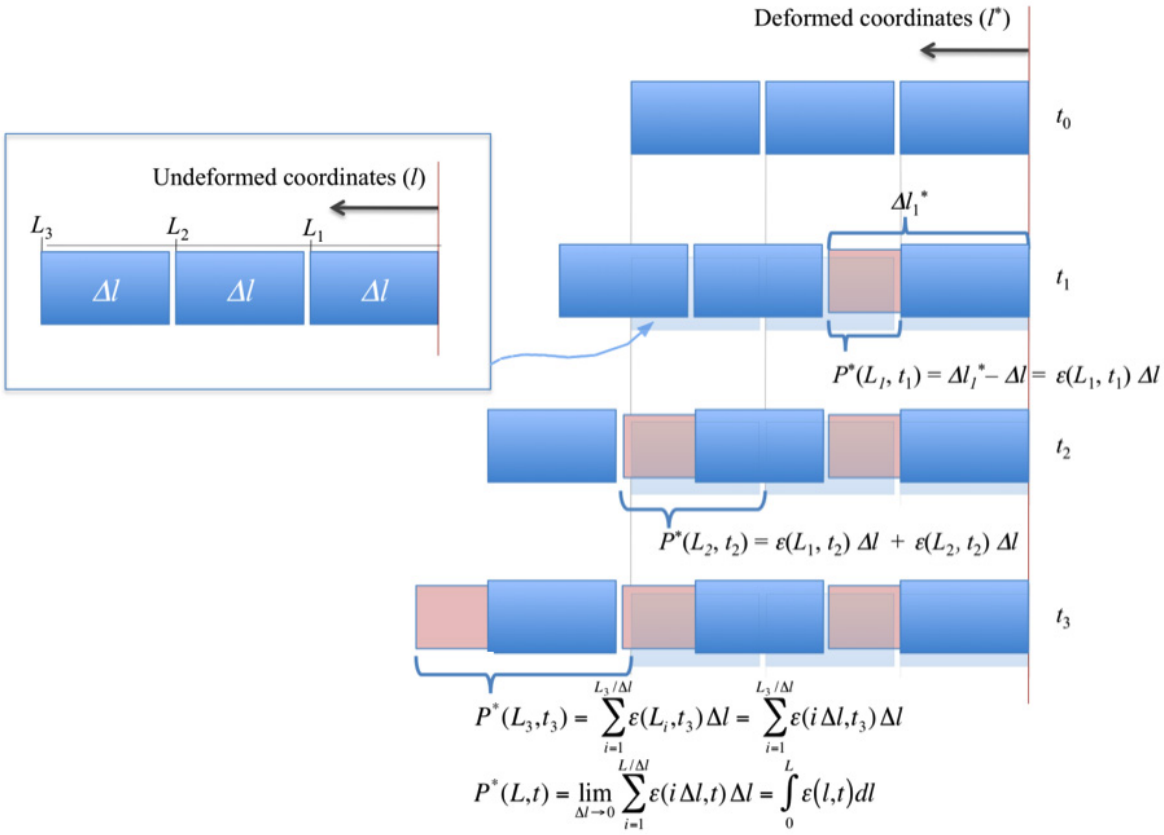

Figure 8: The first three segments of a wormlike body and their deformations over four time steps. Two different coordinate systems are used, one measuring distance from the head of the robot to a point on the body in the undeformed robot, and the other measuring distance from the head to a point on the body in the robot during the deformation (signified by a '*'). The difference between the two is the sum of the deformations between the point on the body and the head (equation 3-2). This sum becomes the integral of the strain function as the segments become differentially small and the structure becomes a continuously deformable body (equation 3-3). Thanks to Kendrick Shaw for helping with this figure.

Substituting equation 3-4 into equation 3-3 and differentiating with respect to time, we can find the velocity in deformed coordinates of a point on the worm relative to the head at the (undeformed) position L at time t,

$$V_P^*(L,t) = \frac{d}{dt}P^*(L,t) = \frac{\partial P^*}{\partial t} + \frac{\partial P^*}{\partial L}\frac{\partial L}{\partial t} = \frac{\partial P^*}{\partial t} + 0 = \frac{\partial}{\partial t}\int_0^L \varepsilon(l - t\,V_{\text{wave}})\,dl = \frac{\partial}{\partial t}\int_{-t\,V_{\text{wave}}}^{L-t\,V_{\text{wave}}} \varepsilon(u)\,du$$
$$= V_{\text{wave}}(\varepsilon(-t\,V_{\text{wave}}) - \varepsilon(L - t\,V_{\text{wave}}))$$

(3-5)

Since L is constant, the partial derivative of P^* with respect to time is the only term that carries through. Lastly, a substitution of variables allows us to evaluate the integral since we have assumed a constant wave speed, $V_{wave,}$. The result is a product of the wave velocity and the difference of strains at the two points. Thus increasing the differential strain or increasing the wave speed will proportionally increase all velocities between points on the worm and thus increase the speed of the worm.

If we choose any two points on the robot L_1 and L_2 (in undeformed coordinates), we can calculate the relative velocity between the two points in deformed coordinates using equation 3-5:

$$V_{\text{diff}}^*(L_1,L_2,t) = V_P^*(L_1,t) - V_P^*(L_2,t) = V_{\text{wave}}(\varepsilon(L_2 - t\,V_{\text{wave}}) - \varepsilon(L_1 - t\,V_{\text{wave}})). \quad (3\text{-}6)$$

Note that this relative velocity will be zero when the strain is equal at the two points. This has important consequences for the design and performance of a peristaltic robot. If two ground contact points are traveling at different velocities on a rigid substrate, one or both of the contacts must slip. Because the diameter of a point on the robot is a function of the strain at that point, all points of ground contact must maintain identical diameters to avoid slipping. Thus any contact that occurs at a point of different diameter due to tipping or drooping under gravity will result in some degree of slip.

3.3 CALCULATING POSITION, ACCELERATION, AND GROUND CONTACT FORCES

We will use a few additional results from the analytical model to evaluate the performance of the robot. The position of the head of the robot as a function of time is easy to measure on the robot, and thus a useful point of comparison with the analytical model. Using equation 3-6, if one of the two points is in contact with the ground and not slipping, and the other point is the head, then the velocity of the head can be described as,

$$V_p^*(t) = \varepsilon(-t\,V_{wave})V_{wave} \ . \qquad (3\text{-}7)$$

By taking another derivative with respect to time, the acceleration between the front of the robot and the nearest ground contact point is

$$A_P^*(t) = \frac{dV_p}{dt} = \frac{d}{dt}\varepsilon(-t\,V_{wave})V_{wave} = -V_{wave}^2\varepsilon'(-t\,V_{wave}) \ . \qquad (3\text{-}8)$$

Note that posterior parts of the robot will experience these same velocities and accelerations after a delay of L/V_{wave}. So the maximum acceleration of a given segment scales with the square of the velocity of the wave, and the time derivative of the strain function. This has implications for the scalability of worm robots. It is not surprising that even with smooth traveling waves, the profile of the deformation wave affects the maximum accelerations that are experienced. For instance, a square wave would require infinite acceleration.

The sum of the forces that produce the accelerations given in (3-8) must be equal and opposite to the ground reaction forces at the ground contact points. Therefore we can find the total ground reaction force, F, by integrating spatially over the segments that have left the ground.

$$F = \int_0^{tV_{wave}} \left(-V_{wave}^2 \varepsilon'\left(-(t - l/V_{wave})\, V_{wave}\right)\right)\rho\, dl$$
$$= \rho V_{wave}^2 \varepsilon(-tV_{wave}) \quad , \qquad (3\text{-}9)$$

where ρ is the linear density of the robot. Thus, the required ground reaction forces are proportional to the linear density, the square of the velocity of the deformation wave and the strain. Note that even if the deformation wave, ε, is not periodic, it is likely to return to zero at some point, i.e, the differential segment returns to its initial length. When it does, the velocity of that segment will also return to zero according to (3-7). Likewise with the forces, if ε returns to zero as new ground contact occurs, then, as (3-9) shows, the force at both ground contact points will also go to zero. In effect, the internal forces of the waveform must cancel. An important condition for this is that the structure is continuously deformable.

Looking at a single wave, one can see that (3-9) must hold in order to not have an over constrained system, but only for each wave independently. Thus, once a waveform has started, it is best for it to pass through the robot unchanged. The next waveform, however, can be different, but if its total displacement is different, then there will be forces acting on the body, and those forces will be proportional to acceleration of the body relative to the last wave's resultant velocity. This means that given these circumstances, external forces are not needed to maintain locomotion. If one draws a box around the worm and considers it as a free body, the ground contact forces are the forces that will accelerate the center of mass of the worm. So by showing in Equation 3-9 that the ground contact forces equal zero, by Newton's second law, the center of mass of the worm must not be accelerating - it must have a constant velocity. Kendrick Shaw was able to find the velocity of the center of mass in a more extensive derivation and verified that this is

indeed the case [Boxerbaum 2012]. In this way, the motion is analogous to a wheel rolling on flat ground: points along the circumference are accelerating, but the wheel's center of mass rolls at a constant velocity and requires no external forces. Unlike a wheel, peristalsis still requires internal forces due to the accelerating and decelerating mass of the body. Therefore, mechanical work is still required for the system.

This analysis has made several assumptions, such as that there are no errors in slip or strain. To make the model more useful for analyzing the performance of an actual robot, slip and strain losses will be incorporated into the model in the next section.

3.4 ADDING SLIP AND STRAIN ERROR TO THE ANALYTICAL MODEL

While our analytical model shows that good ground contact timing can eliminate a cause of slippage, it cannot yet predict the magnitude of that slippage. Future incorporation of friction models similar to [Zarrouk 2010] will help to that end. Also, the strain that occurs in a soft body may be less than predicted. We can measure the slippage and the strain that occurs in a robot prototype or simulation. Here, we propose a straightforward modification of the analytical model that allows us to describe both the slip and strain loss. Given a periodic deformation wave expressed by ε and a constant wave velocity, the position of a point P^* in deformed coordinates can be found as a function of time by replacing L with $t*V_{wave}$ and dl with $dt*V_{wave}$ in equation 3-3. Now,

$$P^*(t) = V_{wave} \int_0^t \varepsilon(t\,V_{wave})\,dt \ . \qquad (3\text{-}9)$$

If there were no slip or strain losses, this equation would describe the motion of our robot described later. However, we can account for these losses with minor changes:

$$P^*(t) = V_{wave} \int_0^t \left(Q_{strain} \left(\varepsilon \left(t V_{wave} \right) - Q_{slip} \right) \right) dt \,. \qquad (3\text{-}10)$$

Q_{strain} is the differential equivalent of the traditional efficiency function. If a differential element only expands to 80% of its intended maximum length, then $Q_{strain} = 0.8$. This value can be measured directly from the robot or simulation. If you know the actual strain that is being produced, either from motion capture data or simulation outputs, then Q_{strain} is not needed. Q_{slip} is the factor that accounts for slippage. The integrand of Equation 3-10 is still the velocity of the point on the body, so Q_{slip} can be found experimentally by finding the time at which the point has zero velocity, and starts to move backwards. Both factors can vary with time, depending on external forces and variations in morphology or control. Since our prototypes will be continuous and shift symmetrical, we will assume the values remain constant. If measurements of Q from the prototype or simulation are not straightforward or reliable, these factors can also be found by comparing the analytical model to a simulation or prototype and applying a least squares curve fit method. This can help diagnose the kind of inefficiency that is occurring that reduces the robot's speed.

4 A New Robotic Platform that Generates Continuous Waves of Peristalsis

4.1 Anisotropic Strain Via A Braided Mesh

Earthworms have continuous sheets of both axial and circumferential muscle fibers that work together to create waves of peristaltic motion. During forward locomotion, these two muscle groups are coupled by segments of hydrostatic fluid, creating a hydrostatic skeleton, and are typically activated in alternation at a given location along the body. In the new robot design, we use a braided mesh similar to that used in pneumatically-powered artificial muscles [Quinn 2003] to create this coupling between axial and circumferential motion with a single hoop actuator. The robot is still cylindrical in shape, but the outer wall now consists of a single continuous braided mesh (Figure 1). Any location along the braided mesh can be fully expanded or contracted. Hoop actuators are located at intervals along the long axis, close enough together that smooth, continuous waves can be formed. When these hoop actuators are activated in series, a waveform travels down the length of the body. The result is a fluid motion more akin to peristaltic motion than that generated by previous robots [Boxerbaum 2009].

4.2 Actuation Using Shape Memory Alloys

The kinematics of peristaltic motion are entirely scale invariant. At any given scale, a cross-hatched mesh needs to be constructed with the correct stiffness, and a suitable actuation method found. Here, I briefly propose two methods of construction at a very small scale.

A robot with a diameter on the order of one centimeter would have applications in medicine, including examination of the entire gastrointestinal tract, as well as applications in search and rescue environments and military reconnaissance. Shape Memory Alloys (SMAs) are a good candidate for actuation at this scale. Micro helix SMAs have strain ratios of up to 200% and can be actuated in 0.2 seconds [Menciassi 2004]. One challenge of working at such a small scale is how to get the SMA actuator to return to its initial position. It must cool down, and a force must be applied to restore it. Seok, et al. address this problem of re-extending the actuators by mechanically fixing the length of the robot, so one constricting SMA forces others open. This method succeeded in producing locomotion, with losses due to the uneven expansion cycle [Seok 2010]. Here, we propose a method of returning the SMA to its original state that relies on a hydrostatic fluid. This fluid could also serve to cool the SMAs, and speed up the cycle, as other groups have shown [Mascaro 2003]. We also propose that the wiring for the SMA actuators can also constitute the braided mesh (Figure 9) [Boxerbaum 2009].

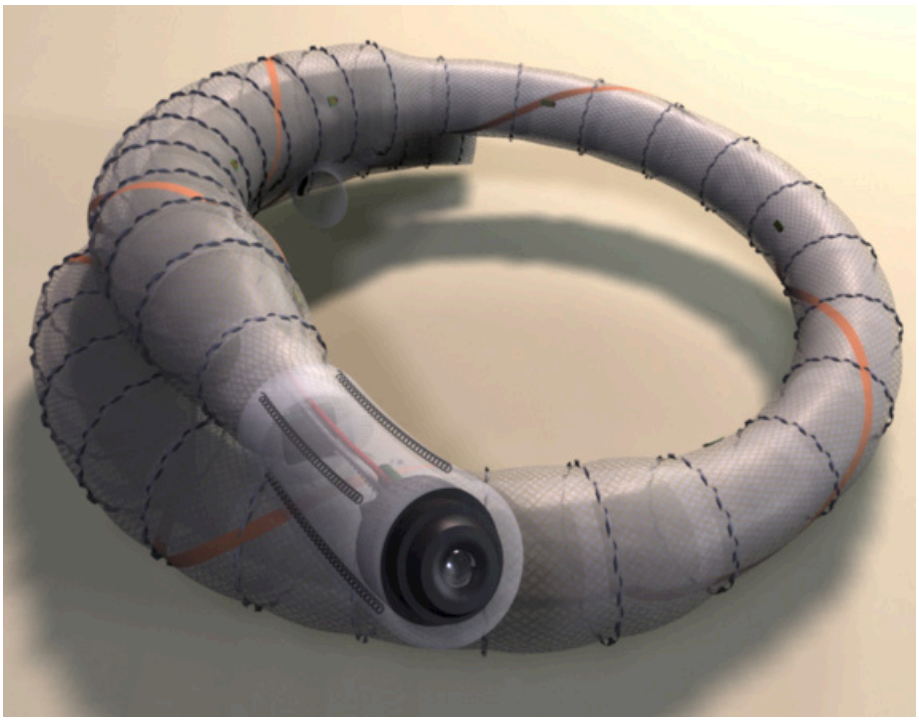

Figure 9: SoftWorm robotic concept using shape memory alloys and a hydrostatic fluid as a return spring.

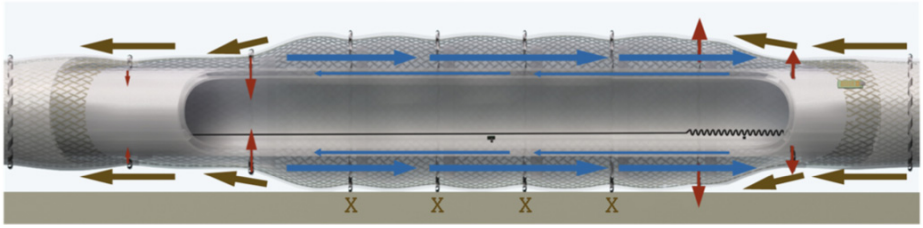

Figure 10: A cross sectional view of the SMA concept. The brown arrows indicate the flow of the exterior braided mesh. The blue arrows indicate the flow of the bolus of fluid that expands the contracted sections. The red arrows indicate expanding and contracting hoop actuators.

In this implementation, shown in Figure 9, a bolus of fluid (large blue arrows) moves between the outer skin and the inner payload of the robot by the sequential constriction of hoop SMA actuators (red inward-pointing arrows). As the fluid is squeezed at the trailing edge of the wave, it causes radial expansion at the leading edge of the wave (red outward-pointing arrows). The result is the generation of continuous peristaltic waves along the robot, causing it to move in the opposite direction of the wave (brown arrows).

4.3 ACTUATION USING HYDRAULICS

An alternative method of actuation at small scales is being explored as well. The braided mesh of the robot could be made of hollow tubing and serve as hydraulic lines for micro-hydraulic actuators at each hoop (Figure 11). Hydraulic actuators are generally only effective as pushing actuators, requiring the natural state of the robot to be elongated and narrow. Expansion at one of the hoop actuators would be achieved by applying pressure at the end of the hydraulic line. This would also allow for mechanical coupling of the hoop actuators, and allow them to be driven by a single end-mounted motor. This setup could achieve faster waves, and therefore faster robot speeds than the SMA implementation, but it would require an effective micro-hydraulic piston to be developed.

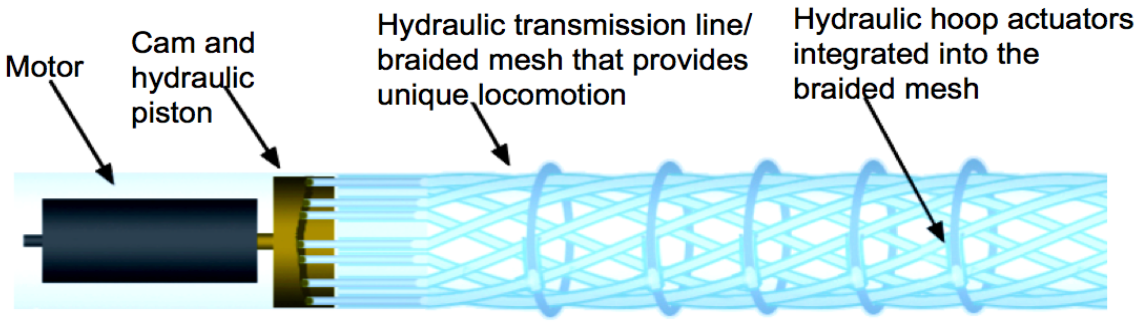

Figure 11: Micro-hydraulic actuator concept. Here, the hoop actuators expand against a contractile force to create the wave motion.

5 Actuation Using A Motor, Cam and Cables

5.1 INTRODUCTION

The question then became, how do we build a soft wormlike robot that demonstrates the new principle of locomotion described in Chapter 3 as simply as possible? The answer is to make it bigger. This allowed me to assemble the robot by hand, and use conventional, off the shelf materials. Not including the cost of the motor, which was left over from a previous project, the first prototype cost under $200.

5.2 HOW IT WORKS

Two large-scale prototypes of this new robotic concept have been completed and tested. With a maximum diameter of 25 cm, they are scaled to function in fresh water mains (Table 1). Like the micro-hydraulic concept, the robot is hollow. This would allow inspection and servicing to be performed without shutting off water flow. But, instead of expanding segments, the actuators contract radially, causing axial expansion. Furthermore, instead of using a bolus of fluid to return the hoop actuators to their initial state, latex springs are used as a return spring along the length of the robot (Figure 12). This greatly simplifies the design.

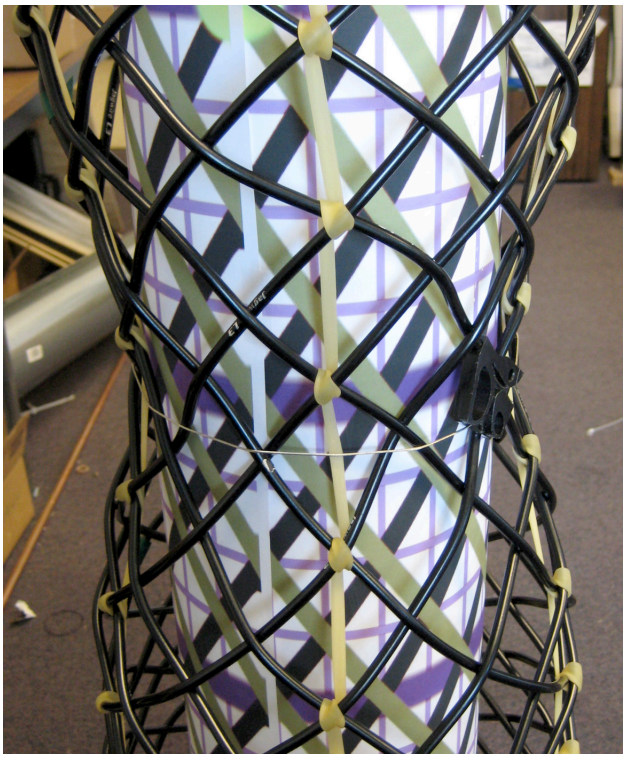

Figure 12: The first prototype keeps its anisotropic strain properties by weaving bicycle brake cable sheathing and securing it with latex tubing that also acts as a return spring. This method of construction did not keep the mesh in good alignment after several trials. In this figure, a single hoop actuator has been installed and is in the constricted position.

Table 1: Robot Properties

Total mass (without batteries)	3.8 kg
Motor mass	0.9 kg
Total rest length	50 cm
Rest length of section with actuators	33 cm
Length of section with actuators under a 2kg load	59 cm
Outer diameter	22 cm
Inner diameter	18 cm
Cam diameter	16 cm
Cam arm radius	5 cm
Hoop actuator stroke length	20 cm

Similar to the previous concepts presented in Chapter 4, the braided mesh that provides the anisotropic strain properties has an elegant dual function – it also routes steel

actuator cables which transmit power through the robot. The mesh is made of bicycle brake cable sheathing, which is hollow and rigid along its long axis. Steel cables run through the sheathing out to individual hoop actuators where there is a mechanism that interrupts the brake cable sheathing and routes the cable around the circumference (Figure 14). Two cables run through each sheath and split in opposite directions to meet on the far side, creating a hoop actuator. This doubles the stroke length of the actuator compared with a single cable wrapped around the whole circumference.

In both prototypes, the steel actuator cables are pulled in sequence by a cam driven by a single drive motor at one end of the robot (Figure 15). While future versions will have individually controlled actuators in order to study sensorimotor wave propagation and adaptive behavior (Chapter 8), this mechanism creates peristaltic motion with no computational overhead and with a waveform that provides good speed. In this way, forward and backward motion is controlled as a single degree of freedom using a single DC motor.

5.3 JOINTS, CLAMPS AND CABLE GUIDES

In the first prototype, the mesh of brake cable sheathing keeps its shape because it is woven and secured with the latex tubing at most junctures (Figure 12). This technique did not prove robust in repeated testing, and the uneven bending of the sheathing increased the maximum cable friction. The second prototype remedies this problem by using swivel joints at each crossing (Figure 13). These joints are specialized to provide several functions. Wherein the first prototype had hoop actuator cables that simply wrapped around the robot, in the second prototype, specialized swivel joints guide the hoop

actuator cable around the circumference and transmit its forces to the mesh in a much more controlled fashion. Other swivel joints secure the latex return spring along the long axis of the robot. In future versions, this latex return spring can be replaced with actuated cables to add additional degrees of freedom and turning ability. Securing the mesh with swivel joints essentially braids the mesh, and allows the mesh material to have a constant bending radius. Alternatively, encasing the mesh in a soft polymer skin would also preserve the alignment of the strands and act as a return spring.

Each hoop actuator has a special swivel joint with a clamping mechanism (Figure 13 A). This secures both strands of the hoop actuator, and allows the rest length of each actuator to be tuned. This addresses a mechanical limitation we discovered in the first prototype. The most distant actuators require more force to actuate because of their long runs through the brake cable sheathing. In the first prototype, the cable clamping mechanism was integrated into the cam mechanism head at one end of the robot, and the cable was pulled by a drive motor (Figure 15, top). The clamp was frequently not strong enough to resist the large forces in the cables for the most distal actuators. So, typically after a few waveforms, the cables slipped out, causing the most distant actuators to fail. Even with several failed actuators, the robot still moved forward at a slower speed, suggesting that the design is robust to partial actuator failure.

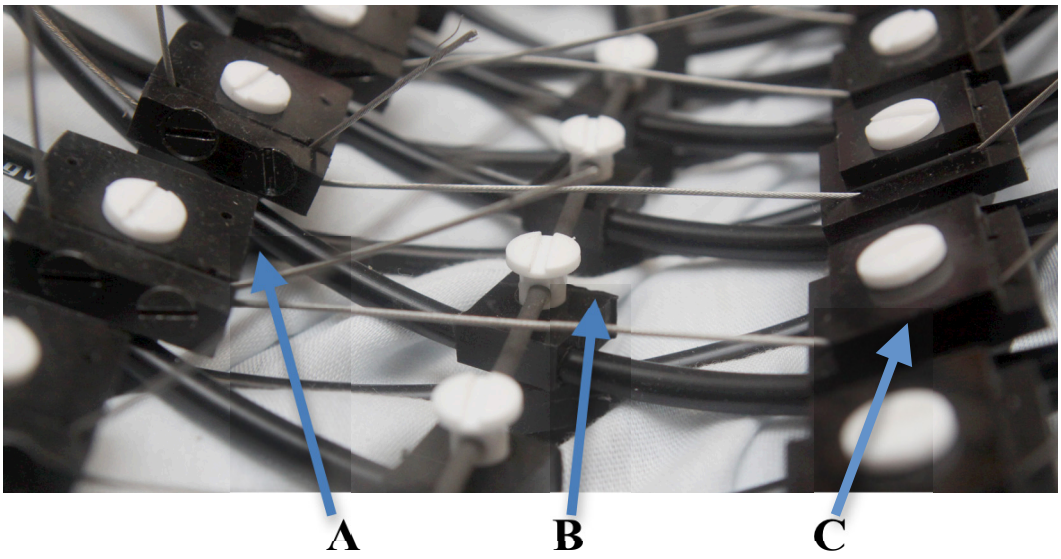

Figure 13: The second prototype secures the braided mesh, the hoop actuators and the latex return spring with several hundred specialized swivel joints. They secure the end of the hoop actuator cables (**A**), guide and secure the latex return spring (**B**), and guide the hoop actuator cables (**C**).

5.4 ACTUATOR HEADS

It was a challenge to create a mechanism to route the actuator cables out from inside their sheaths and around the robot circumference. Because of the way the body deformed, care had to be taken to stay above or at the minimum bending radius, no matter what the orientation of the robot (Figure 14). Ilya Minsky, a Senior Project student, played a role in the final design and manufacturing of this part.

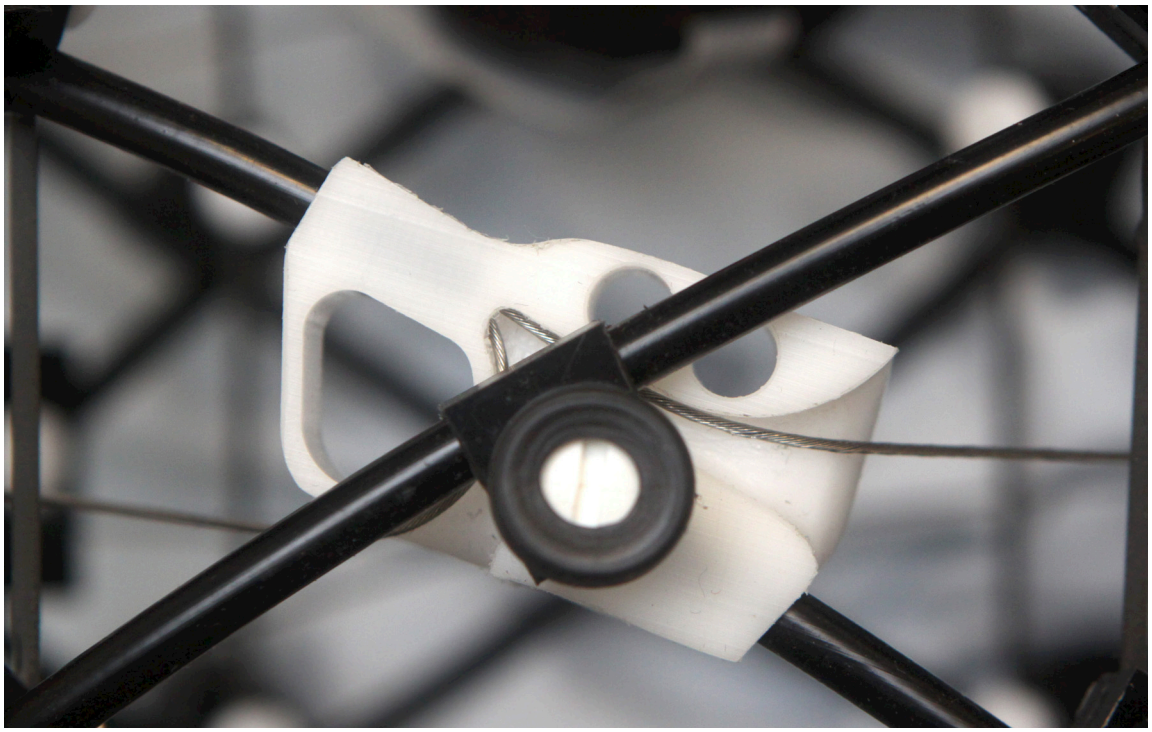

Figure 14: A specialized swivel joint terminates the brake cable sheathing and routes the nested cable in two directions to form the hoop actuator around the circumference of the body. The rubber foot in the foreground provides traction with the ground.

5.5 Motor and Cam Mechanism

The cam mechanism is designed to pull the cables with a waveform that is roughly sinusoidal in both time and space. The exact waveform is a combination of both sine and cosine waves that is cycloid-like, and has a near singularity due to the geometry (Figure 22). The shape of the waveform can be adjusted easily by changing the length of the cam arm. In the current setup, two waves are present at all times. Closely paired cables visible in Figure 5-4 are routed to two hoop actuators spaced apart by half the length of the robot. Their proximity to each other on the perimeter of the cam indicates that these two actuators will have nearly identical states at any given time. With this style of cam mechanism, any whole number of waves along the body is possible. Both prototypes were

designed such that two full waves propagated along the length of the body in order to prevent early ground contact, while still providing at least five hoop actuators per wave. Three waves would run the risk of not being able to deform enough, which increases inter-segmental mechanical coupling. This can decrease total strain, as is shown in simulation (Chapter 7). In the first prototype, ten hoop actuators were distributed along the length of the robot, utilizing only half the available brake cable sheathings. The second prototype had twelve hoop actuators, and the mesh was constructed of only twelve strands in order to reduce weight and the number of swivel joints (Figure 16).

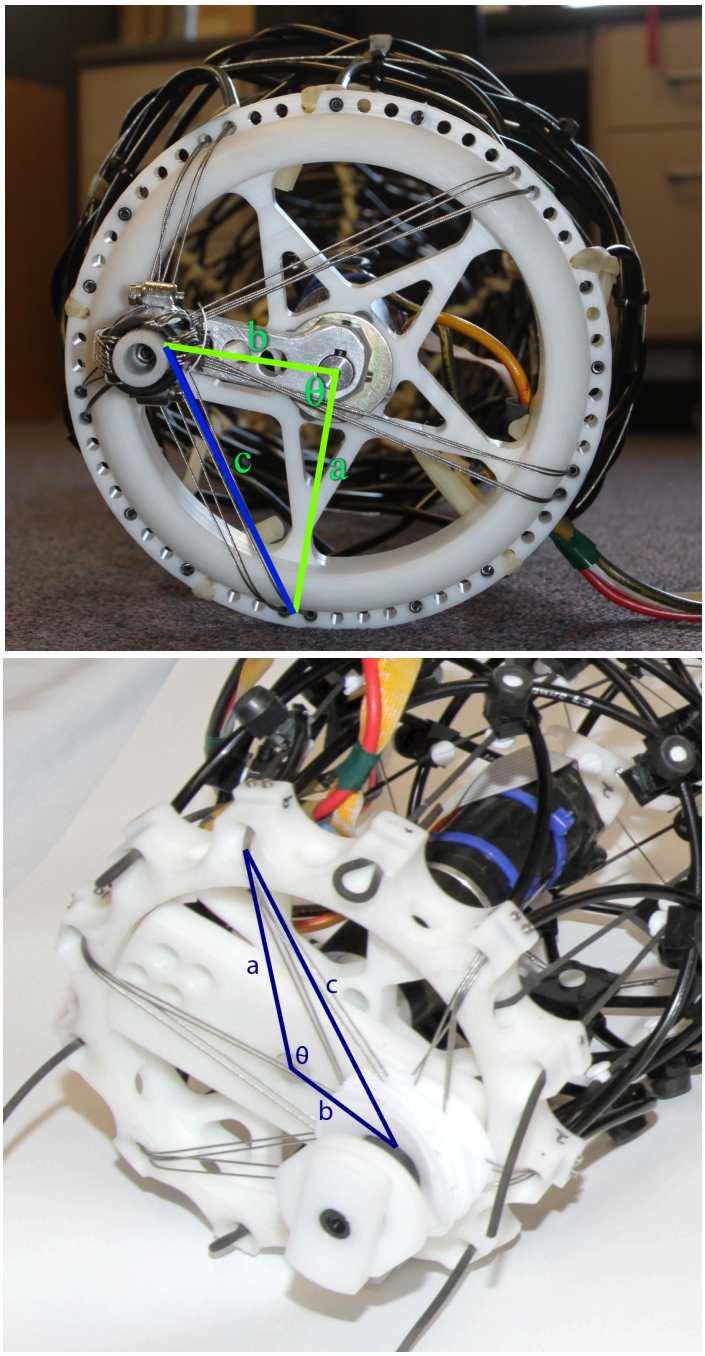

Figure 15: Two versions of the cam mechanism that drives all actuators and creates two traveling waves along the length of the robot. The location of the cable port about the circumference indicates the phase shift relative to the other actuators. In the first prototype (top), the cables are secured at the cam head and they experience a sharp turn when entering the cable sheathing. In the second prototype (bottom) the cables are secured at the actuator, and only wrap around the cam head, and the cable ports have been redesigned to prevent sharp bending of the cable.

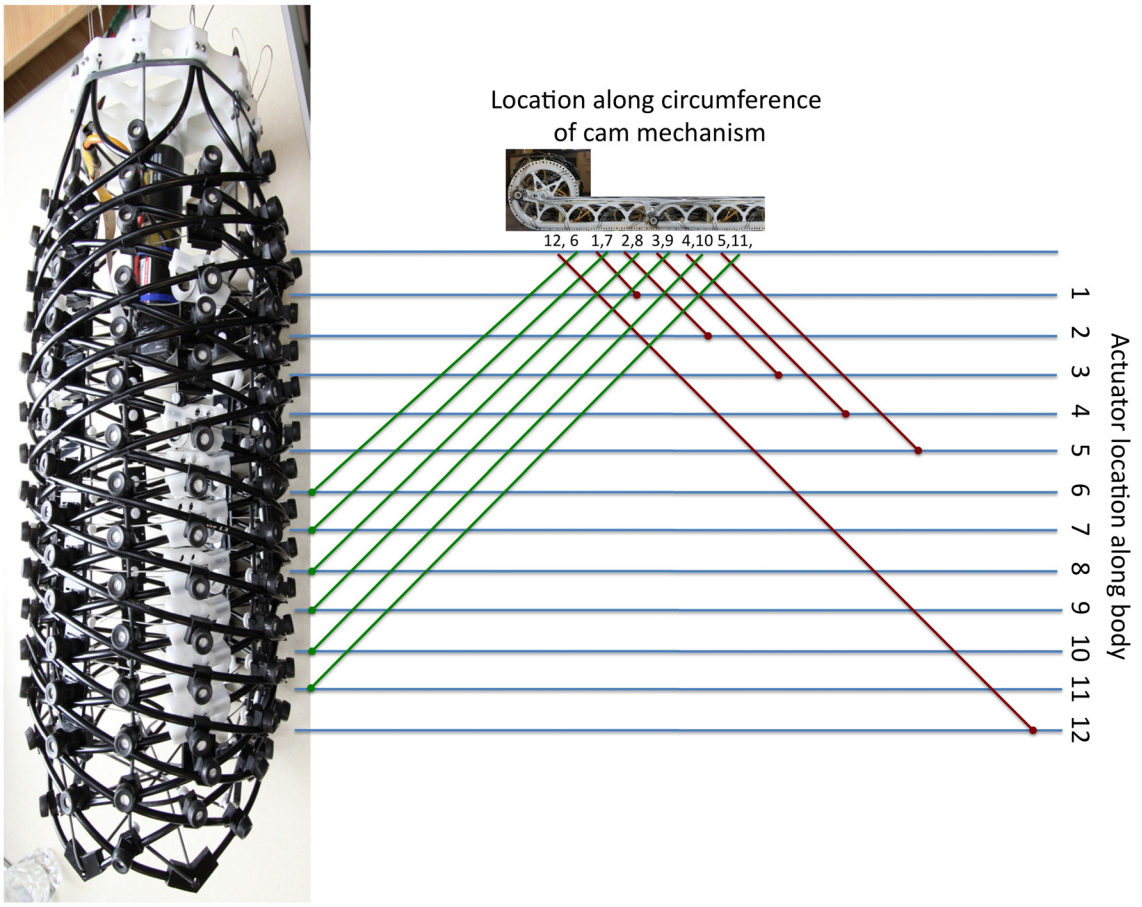

Figure 16: Cable wiring diagram for the second prototype with twelve actuators and twelve strands. The green strands wrap clockwise relative to the cam mechanism, while the red strands wrap counterclockwise. Each actuator location is paired at the cam head with a location half the length of the robot away.

We initially attempted to use polyester string as an actuator cable, because of its very small minimum bend radius. These strings repeatedly broke under loading. While Kevlar or Spectra string may still be good alternatives, steel cable was chosen specifically for its strength and its natural pairing with the Teflon lining of the brake cable sheathing. The larger minimum bend radius of steel cable meant that special care had to be taken in how the cables were routed.

The cam mechanism went through several design revisions. The first design was able to produce a travelling waveform in the robot, but had two major drawbacks. The

design I constructed to keep the cables within the minimum bending radius was geometrically incorrect; it only worked in one of the two dimensions the cable moved in (Figure 15, top). This caused a tight bend in the cables, which quickly damaged them. This problem was fixed in the second design (Figure 15, bottom).

The other challenge took three iterations to find a robust solution. The head of the cam mechanism translates in a curved path, but does not necessarily rotate. This device is what pulls the cables, and in the first iteration, it also acted as the cable clamp location (Figure 15, top). However, the tension in the cable is actually greatest at the cam head, which made them prone to slipping. In the second iteration, the cable clamps were moved to the hoop actuators, as discussed in the previous section. Instead, at the cam head, the cables wrapped around a mandrel. In this arrangement, a new problem developed. When the motor is not actively pulling on a cable, the return spring that is integrated into the braided mesh is pulling the cable back through the sheathing. However, this does not always happen, either due to a lack of tension in the return spring, or to environmental factors such as crawling through a tube that is smaller than the maximum diameter of the robot. In this case, the cable bends outward at the cam mechanism, and sometimes other cables will get entangled in it. I attempted to fix this problem by inserting thin sheets of Teflon between each cable (Figure 15, bottom). This approach worked for a while, but with Teflon being inherently plastic, it eventually deformed and became ineffective.

The final iteration of the cam head resolves this problem by integrating six mandrels into a single cam head (Figure 17). Each mandrel holds two sets of cables that belong to actuator pairs separated by one wavelength. This device effectively keeps the cables separate even when large amounts of slack in the cable exist (Figure 18).

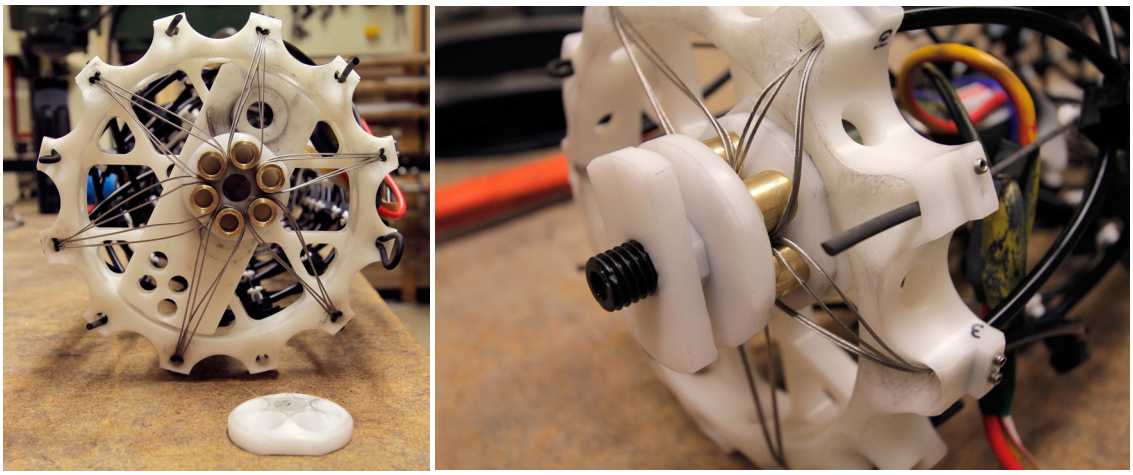

Figure 17: The final cam head design. The close up (right) shows the assembled device that keeps all wires separate by using several brass mandrels shown in the disassembled device (left). When the cables are not pulled back through the sheathing, either due to environmental conditions or due to a lack of proper tension in the cable, this new cam head prevents any tangling of the cables.

Figure 18: A still taken from high-speed video of the final worm cam mechanism. There is slack in the top four cable pairs, but the cam head prevents them from tangling.

Video of all trials was recorded from an angle orthogonal to the direction of motion. The second prototype trials were processed using WINanalyse (Mikromak) software to extract position and velocity data. The white plastic fasteners with the rubber feet were used as fiducials for this task.

5.6 A Few Final Fixes

After several hours of continuous operation, I noticed that the two hoop actuators most distant from the cam mechanism were sometimes not operating correctly. The frictional forces that develop in the tube grow exponentially with length because the shape of the tube is helical, causing positive locking through the capstan effect. The most distant actuator cable wraps around the worm twice before getting to the hoop actuator, and then once around the robot. Including the turns within the actuator head and cam mechanism, this constituted almost four full loops. As a result, the force required to actuate the most distant actuators is significantly higher, and one can see the robot rock slightly as it pulls on these cables. These large forces eventually caused the brake cable sheathing to fail at the cam mechanism. Conversely, when the motor was not pulling on the cables, the return springs were not strong enough to fully pull the cable back through the sheathing, causing the cable buckling discussed in the previous section.

There are several ways to address this problem. One can make the robot smaller. While scaling the robot would preserve the number of loops, it would also lessen the distance the cable is in contact with the sheathing. One could also decrease the number of actuators per wave. This would decrease the number of loops. With the existing design, these fixes would mean starting over. Based on a suggestion from Nicole Kern, I decided to rewire the last actuator cable sheaths so they had a nearly straight shot from the cam mechanism to the hoop actuators. I did this by simply modifying the hoop actuator head so that it still attached to the braided mesh, but a separate sheath was added to run straight back to the cam mechanism, bypassing the mesh entirely. Because the distance between the hoop actuator and the cam is always changing, this cable still needed to bend, but the force required to actuate the robot was reduced significantly. This greatly improved robot

performance by increasing the effective strain rate and decreasing the required motor torque. The robot stopped rocking altogether.

5.7 Turning and Future Steering

Worms have a set of longitudinal muscles per segment that can cause turning in both the horizontal and vertical plane. These muscles are activated in a coordinated way such that whole body turning is implemented. Often the head of the worm feels out the path of least resistance, which is then followed by the body. For this reason, we propose controlling a future wormlike robot by using a combination of whole body turning and articulated head movements.

In order to test the feasibility of whole-body turning, we added an extra elastic cord to one side of the braid and tensioned it. This tensioning caused one side of the braid to remain more circumferentially expanded, while the opposite side was more contracted. This bends the robot towards the side that is circumferentially expanded. This kind of turning is mechanically coupled to forward locomotion, so one would expect that maximum speed and turning radius are related. Future work could explore this relationship both with the robot and in simulation. Despite the complexity of the geometry, this approach did not appear to interfere with the curved forward motion of the robot (Figure 19). With the tested asymmetrical tension, a turning radius of 0.65 meters was observed. The average robot speed was 3.5 meters per minute, only slightly less than the speed when going straight forward of 4 meters per minute.

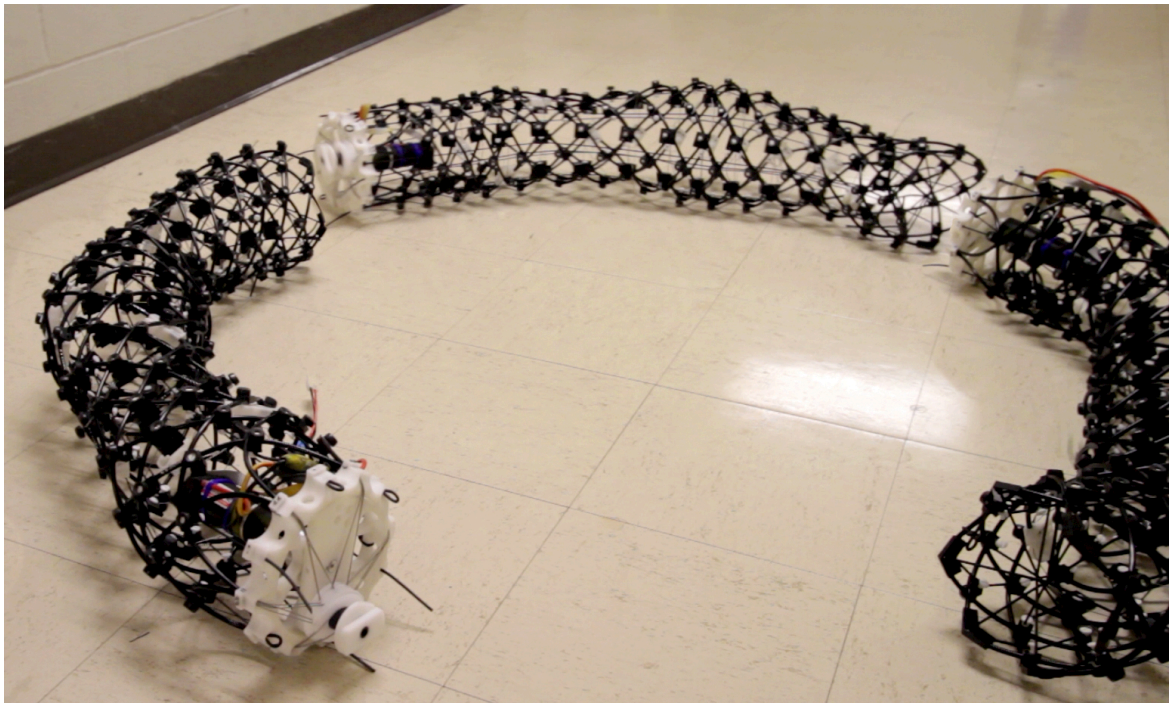

Figure 19: A composite image taken from three stills of a video of the robot turning. With the given asymmetrical tension, the average turning radius was 0.65 meters. The Images from left to right were taken at t=0, 22 and 44 seconds.

The next generation of wormlike robots could use two elastic or semi-elastic cords, one left and right of center, and the other top and bottom. The tension between the two sides could be adjusted by a motor that could spool out cord from one side to the other.

6 Derivation of the Kinematics of the Large Scale Prototype

6.1 The Kinematics of the Braided Mesh

The mechanical strain that occurs with the simple braided mesh described above can be directly calculated from the geometry of four crossing strands (Figure 20). We will assume the strands are rigid in order to treat them as a four-bar mechanism. However, there must be bending in these fibers in order for distinct waves to form. The scale of the weave is not important for this derivation, as it only describes the anisotropic properties of a continuous ideal material. The hoop actuator contracts along the circumferential length d by a displacement c, changing its length to d':

$$d' = (d - c) \qquad (6\text{-}1)$$

The input c is often a periodic function that describes the contractions as a function of time or position. While the anisotropic strain properties do not depend on the scale of the braiding in this kinematic approach, the two values d and c must be scaled appropriately to one another. Here, we define d as the maximum circumference of the entire braided mesh, so c is the total change in length of the hoop actuator.

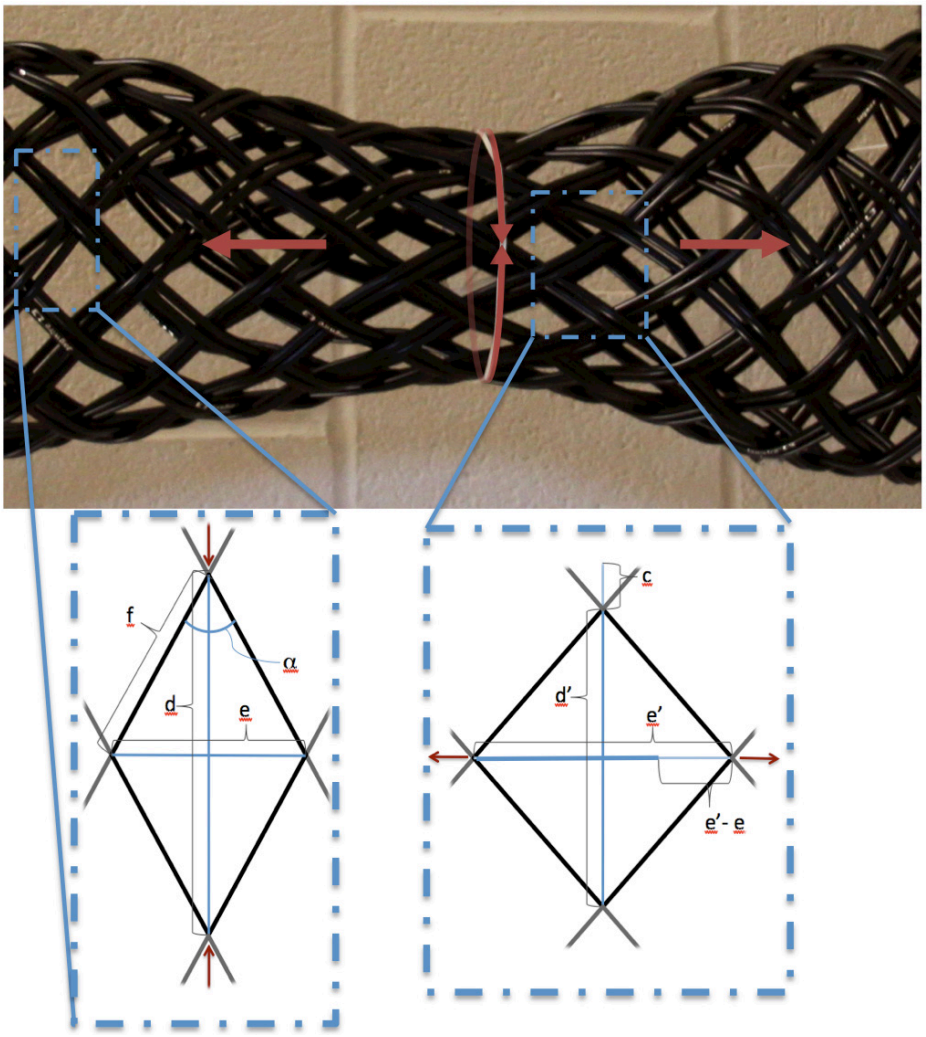

Figure 20: A single element of the braided mesh is used to derive the anisotropic strain properties of the material. The dimension *c* is the input, the change in circumferential length due to the hoop actuator. The change in axial length is *e'-e*.

The dimension along *e* will expand by an amount that is a function of the initial shape of the diagonal element, defined here by the angle α. From the Pythagorean theorem and the law of sines, we have:

$$d'^2 + e'^2 = (2f)^2 \longrightarrow e' = \sqrt{(2f)^2 - d'^2} \tag{6-2}$$

$$\frac{f}{\sin(\pi/2)} = \frac{d/2}{\sin(\pi/2-\alpha/2)} \longrightarrow f = \frac{d}{2\cos(\alpha/2)} \quad (6\text{-}3)$$

The above equations can be combined to find the new axial length e':

$$e' = \sqrt{\left(\frac{d}{\cos(\alpha/2)}\right)^2 - (d-c)^2} \quad (6\text{-}4)$$

Lastly, we will define the strain of the material as:

$$\varepsilon = \frac{e'-e}{e}, \quad (6\text{-}5)$$

where

$$e = d\tan(\alpha/2) \quad (6\text{-}6)$$

Combining (6-4), (6-5), and (6-6) we now have an equation for the axial strain of the braided mesh as a function of the hoop actuator activation c and the geometry of the mesh defined by the rest circumference d and start angle α:

$$\varepsilon = \frac{\sqrt{\left(\frac{d}{\cos(\alpha/2)}\right)^2 - (d-c)^2} - d\tan(\alpha/2)}{d\tan(\alpha/2)} \quad (6\text{-}7)$$

We have seen that a strain function of this kind plays a critical role in determining the motion of the robot or animal.

6.2 The Kinematics of the Cam Mechanism

The cam mechanism is designed to pull on the cables with a waveform that is roughly sinusoidal in both time and space. The exact waveform is a combination of both sine and cosine waves that has a near singularity due to the geometry (Figure 22). The shape of the waveform can be adjusted easily by changing the length of the cam arm (Figure 21, line *b*). In the current setup, two waves are present at all times. Closely paired cables visible in Figure 21 are routed to two hoop actuators spaced apart by half the length of the robot. Their proximity to each other on the perimeter of the cam indicates that these two actuators will have nearly identical states at any given time. With this style of cam mechanism, any whole number of waves along the body is possible. Both prototypes were designed such that two full waves propagated along the length of the body in order to prevent early ground contact, while still providing at least five hoop actuators per wave. In the first prototype, ten hoop actuators are distributed along the length of the robot, utilizing only half the available brake cable sheathings. The second prototype has twelve hoop actuators, but the mesh is constructed of only twelve strands in order to reduce weight and the number of swivel joints.

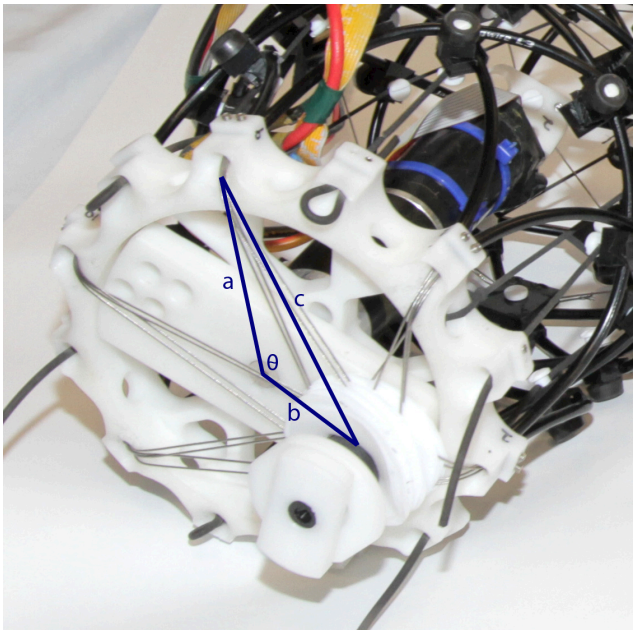

Figure 21: The cam mechanism that drives all actuators and creates two traveling waves along the length of the robot. The location of the cable origin about the circumference indicates the phase shift relative to the other actuators. The distance *a* is the radius of the cam mechanism. The distance *b* is the length of the crank arm that pulls the cables. The distance *c* is the length the actuator cables have been pulled from their rest length.

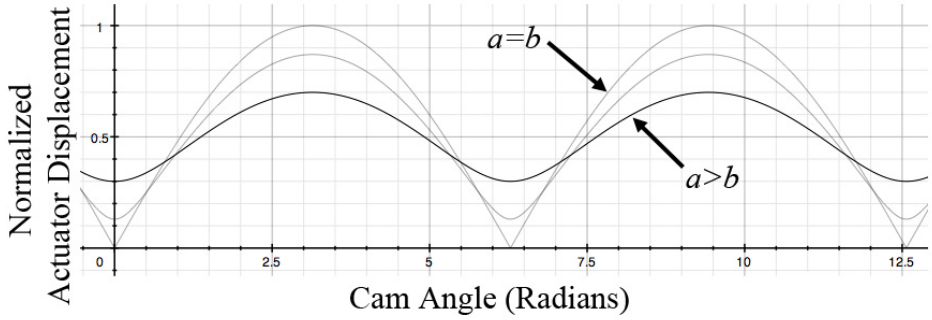

Figure 22: Three possible actuator displacement waveforms (*c*), created by the cam mechanism with different relationships between *a* and *b*. We chose a cam mechanism design that would maximize strain, so *b* is as close to *a* in length as possible, given mechanical limitations.

7 A 2-D Dynamic Simulation of The Robot Using 8 "Working Model"

8.1 METHODS

A 2-D dynamic simulation was created using Working Model 2D (Design Simulation Technologies, version 9.0) to evaluate this method of locomotion, and to capture the discrete nature of individual segments that are not represented in the analytical model. Each simulated body segment consists of a modified four-bar mechanism, where each bar is split into three pieces joined by a torsional spring (Figure 23). This approximates the ability of the braided mesh to bend, an essential capability for wave formation. The number of segments tested ranged between six and twelve. Each 'muscle' or actuator was simulated using a stiff spring-damper system, in which the rest length of the spring was driven by a periodic function:

$$\text{hoop actuator length} = d - 2\,sqrt(a^2 + b^2 - 2ab\cos\left(V_{\text{wave}}\,t - \varphi\left(\frac{N_{\text{waves}}\,2pi}{N_{\text{segments}}}\right)\right)$$

(7-1)

where *d, a* and *b* are constants (*d* =.5 m; *a* = 0.08 m; *b* = 0.07 m). N_{waves} and $N_{segments}$ are the number of waves and segments along the body (N_{waves} =1,2,3; $N_{segments}$=12), and φ is the number of the actuator that provides the phase shift (φ =1 through 12). This function is based on a cam mechanism in the robot that is used to activate all the actuators, and is described in Section VI. The density of the robot and the stiffness of the braided mesh and the actuators were chosen prior to the design of the final prototype robot, so comparisons between the two will remain qualitative (Table 2). One of the advantages of this

simulation is access to a large amount of data, including the positions, velocities, and accelerations of points on the robot, including its center of mass.

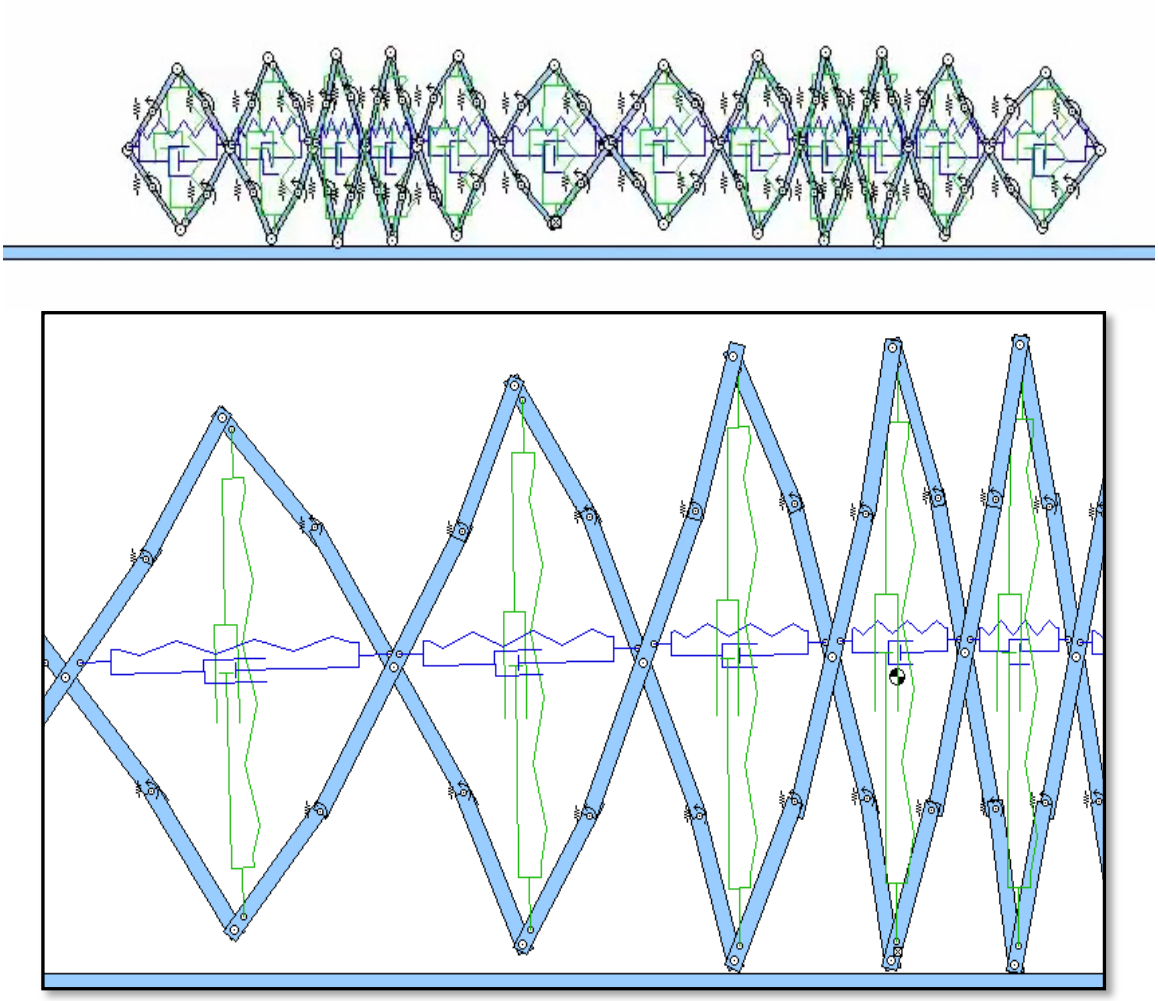

Figure 23: A 2-D simulation of the robotic concept with two whole waves. The actuators are shown in green, and are modeled as stiff spring-dampers, where the rest length of the spring is varied as the control input via Equation 27. The blue spring-dampers are the return springs, which are not actuated. The flexibility of the braided mesh is simulated using rigid bodies and torsion springs. Please see Appendix A, Extension 1 for examples of the motion.

Table 2: Simulation Properties

Total mass	0.19 kg
Torsional stiffness used to simulate flexion of fibers	6 N-m/radian
Coefficient of static friction Coefficient of kinetic friction	0.3 0.2
Actuator stiffness Damping	640 N/m 1 N-s/m
Axial spring stiffness Rest length	127 N/m 0.15 m
One Wave, Minimum length, Maximum length	0.073 m/segment, 0.27 m/segment
Integration method	Runge Kutta

Because this simulation does not have a continuous exterior wall, the ground contact transitions are imperfect. Here, the deformation of each segment is the result of interactions between many actuators, as is the case with true soft-body dynamics. By changing the number of waves over the length of the body, and therefore the number of actuators per wave, we were able to study the effects of segmentation and soft-body interactions on locomotion speed.

9 Using Simulated Neuronal Populations as a Control Network

9.1 INTRODUCTION

The robot described in the first half of this dissertation demonstrates many principles of effective peristaltic locomotion. However, with the twelve circumferential actuators effectively coupled into a single degree-of-freedom, only wave speed can be modulated and the prototype is unable to adapt to environmental changes. Our analysis highlights the importance of transition timing between aerial and ground phases (Chapter 3), something that cannot be optimized without a more flexible control scheme. To attain this, it would be desirable to have independent control of each actuator and thus the ability to also modulate wave amplitude and wave frequency as functions of time and position along the body. This, of course, greatly complicates the control problem.

This chapter explores a biologically inspired method of soft-body robotic control. Our ultimate goal is to test these control strategies on a soft robotic platform with many degrees of freedom, and to show that the control strategy is robust and can adapt to its surroundings. Here, I demonstrate one possible control architecture using simulated neuronal populations. I first show how this architecture can naturally generate rhythmic patterns that would cause peristalsis in a robot. Both the spatial and temporal frequency of these patterns can be easily modulated with two variables. This would be akin to descending signals that control the worm's desired behavior. I then show that this top-down signal can be modulated by strain sensor input, which has been shown to play a role in wave propagation in worms [Gray 1938, Gardner 1975]. This second network

architecture is only a slightly modified version of the open loop network, so I will present the open loop network first, and then show how it was modified to incorporate sensory feedback.

Designing neural control networks is very challenging. We have found three things are essential to the process: a familiarity with the desired behavior, some knowledge of the neuroethology of animals that exhibit the behavior, and lastly a familiarity with the dynamic properties of neural control circuits. Our experience studying peristaltic motion has led to principles for effective motion, which have been laid out in the first half of this dissertation. Combined with our limited knowledge of earthworm neural circuitry (Chapter 2.4), we will proceed to develop a neural model for the adaptive control of peristaltic motion.

9.2 AN OPEN LOOP NEURONAL CONTROLLER

To control a soft wormlike robot, the circumferential and longitudinal actuators must be coordinated to allow peristaltic locomotion, turning, burrowing, and adaptation to the surroundings. In our model, the longitudinal and circumferential actuators are controlled by separate arrays of simulated neuronal populations. These populations have a spatial configuration that mirrors the muscle architecture of a worm, and populations closer to each other have stronger connections. The simulation presented here consists of a column of 150 actuator pairs along the body length.

The dynamics of our system are based on the spatially-extended Wilson-Cowan model of the primary visual cortex [Ermentrout 1979]. In the classical model, excitatory and inhibitory populations are arranged in two stacked 2-D arrays, and populations near

each other innervate one another more strongly. Such a system has been shown to produce statically stable patterns, as opposed to oscillatory behaviors (Chapter 2.6). Here, we modify the classic cortical arrangement by replacing the single inhibitory population array with two separate arrays of inhibitory populations, I_a and I_p, for each excitatory array, whose connections are shifted anteriorly and posteriorly, respectively. There are two excitatory arrays, E_{circ} *and* E_{long}, which drive the circular and longitudinal actuators, and have a spatial configuration that mirrors the muscle architecture of a worm (Figure 24). The simulation presented here consists of a column of 150 actuator pairs along the body length.

The general form of the state variable equations can be described as:

$$\tau \dot{U} = -U + \sigma\left(\sum_{i=1}^{n} w_i\left(U_i \otimes \hat{U}_i\right)\right), \qquad (8\text{-}1)$$

where *Udot* is the time rate of change of the vector *U,* which represents the average level of activity in a given array which stretches along the body of the worm. Within an array, all constants are the same, and connections are shift-symmetrical. Under no other external input, an active region of U will decay at a rate of $-U/\tau$, where τ is a time constant of the population. The function $\sigma(x) = 1.05/(1 + 20\exp(-7x)) - 0.05$ is a generalized logistic, which has a y-intercept of $\sigma(0) = 0$ and is bounded by the range $-0.1 < \sigma(x) < 1$; strong negative inputs yield weak bounded negative outputs, which contributes to the stability of the system. The $\otimes$ operator convolves a population vector U_i with an associated kernel, $\hat{U}_i$, a Gaussian distribution that defines the spatial connectivity of each population. In general, the excitatory kernels ($\hat{E}$) have a much tighter spread than the inhibitory kernels ($\hat{I}$), corresponding to the more localized excitation in the network. The constant n is the total

number of arrays in the simulation. All kernels are normalized, and the total weight of the connection is determined by the scalar constant, w_i, which provides the easiest way to tune the behavior of the system. The sign of w indicates whether the connection is excitatory or inhibitory.

Our open loop controller can be described more specifically with the following state equations that describe one of the two E matrices, E_{circ}:

$$\begin{aligned}
\tau_e \dot{E} &= -E + \sigma\left(w_{ee}\left(E \otimes \hat{E}\right) - w_{iae}\left(I_a \otimes \hat{I}_a\right) - w_{ipe}\left(I_p \otimes \hat{I}_p\right) - w_{ice} C\right) \\
\tau_i \dot{I}_a &= -I_a + \sigma\left(w_{ei}\left(E \otimes \hat{E}\right) - w_{ii}\left(I_a \otimes \hat{I}_c\right)\right) \\
\tau_i \dot{I}_p &= -I_p + \sigma\left(w_{ei}\left(E \otimes \hat{E}\right) - w_{ii}\left(I_p \otimes \hat{I}_c\right)\right)
\end{aligned} \tag{8-2}$$

The vector, $C = (0.5(I_{a\text{-}long} + I_{p\text{-}long})) \otimes \hat{I}_c$, is a coupling term that connects this group with three additional mirrored equations that describe the state of the opposing set of actuators. This keeps the two excitatory networks phase locked with each other. We assume that excitatory neurons have a quicker response time, so $\tau_e = 1$ and $\tau_i = 1.5$. The scalar constants are $w_{ee} = 3.2$, $w_{ei} = 2.4$, $w_{ice} = 1.6$, $w_{ii} = 0.6$ and are also shown in Figure 24.

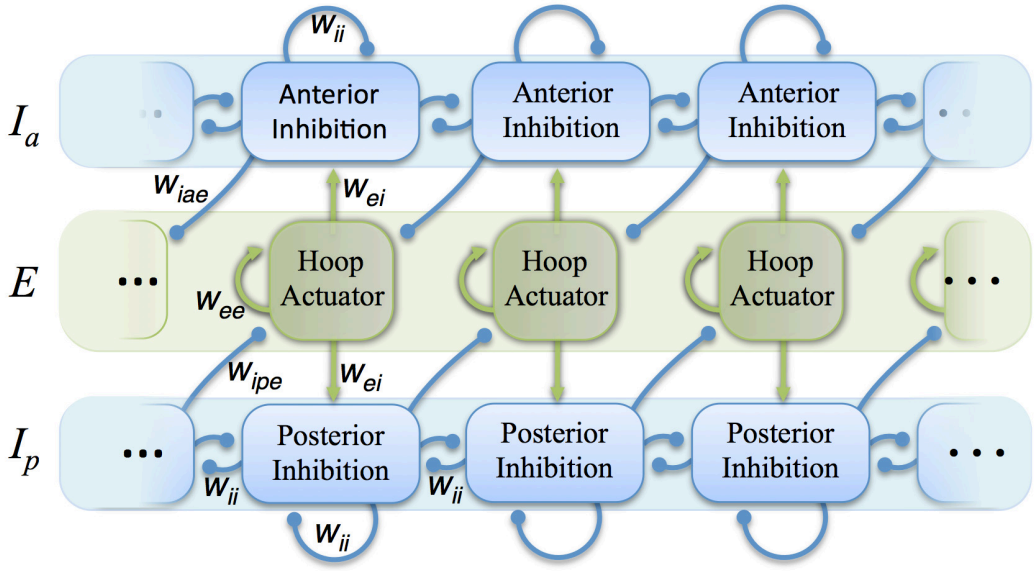

Figure 24: For the purpose of showing several connection trends, the nearly continuous dynamics of a single muscle activation group are represented as three nodes along the body. It is important to note that each synaptic connection shown here is applied to multiple populations through a Gaussian distribution to its neighbors. The excitatory populations (green) directly stimulate the actuators, while also locally stimulating two inhibitory networks (blue). These two networks in turn inhibit the excitatory network more broadly, but are shifted in opposite directions along the network.

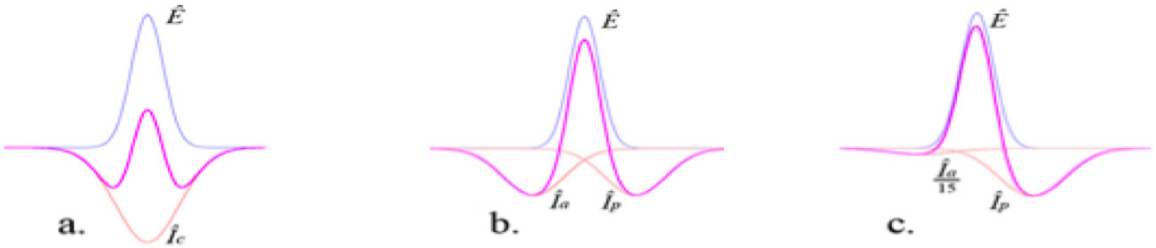

Figure 25: An excitatory (blue) and inhibitory (red) Gaussian kernel combine when excited to produce a "Mexican hat" influence (magenta) (**a**). Similar influences can be created with two offset inhibitory kernels (**b**). If the inhibitory populations have asymmetrical influence, a variety of patterns can emerge (**c**). In **a** and **b**, motion is not sustained without external influence. In **c**, waves of motion naturally occur.

In the classical Wilson-Cowan model, a single inhibitory population has broader yet weaker connections than the excitatory population. In this case, the combined influence of the connections to the excitatory populations forms what has been called a "Mexican hat" distribution (Figure 25 a). In our variation, the two kernels, $\hat{I}_a$ and $\hat{I}_p$, are respectively offset negatively and positively by equal and opposite amounts relative to zero mean and combine to form a "Mexican hat" as well (Figure 25 b). However, biasing the weights of the shifted inhibitory populations gives rise to a variety of additional spatial influences (Figure 25 c). Many of the effects of these various arrangements were discovered inadvertently while building a model of the visual cortex (Chapter 2.6).

Because the state equations use convolutions, boundary conditions need to be set at both the head and tail to define the values just outside the valid simulation space. Both zero-padded and reflective (mirror image) boundary conditions resulted in sustained excitatory activity where the waveform terminated. In some cases, this edge activity propagated through the simulation and caused a noticeable shift in the temporal frequency. A circular boundary condition, so that the head connects to the tail, eliminated these edge effects, but locked the simulation into patterns that have a whole number of waves over the length of the body. The boundary condition that qualitatively most reduced edge effects was a soft zero-padding that linearly forced only the excitatory populations to zero over the first and last 10% of the length of the simulation at the beginning of each time step. This is equivalent to saying that while the density of excitatory neurons remains constant over the length of the body, it decreases to zero at the head and tail.

The initial conditions of all populations were set to zero. Stable patterns would typically emerge from these initial conditions within 50 time-steps. The results of this

simulation are presented in Chapter 10.1.

9.3 A Closed Loop Neuronal Controller

There are many ways that sensory input can be integrated into the system. The effects can be excitatory, inhibitory, or even logic-style operators, and they can innervate any layer of the network. We can glean many hints about the neuroethology of earthworms from the literature (Chapter 2.4), but we do not have enough of a picture to simply build a model from the research. To select among the many alternatives, I compared simulated strain sensor data to the specific contributions of various network components (Figure 26) and found a strong correlation between strain sensor activity of a given kind of muscle and its excitatory signal. This is not surprising since when the body is not under load, muscle excitation causes a deformation to the soft body.

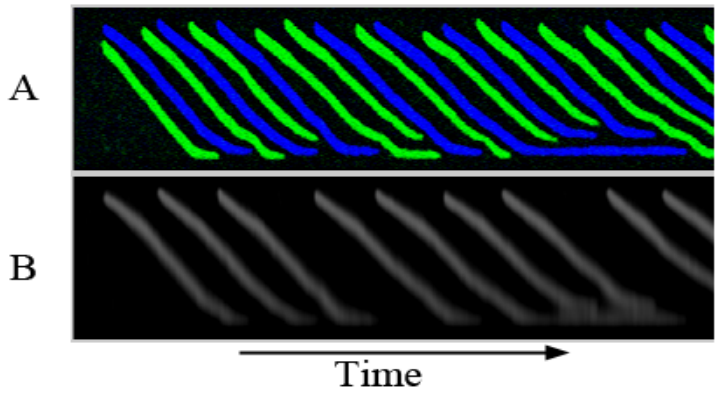

Figure 26: (A) is the excitatory array activation of E_{hoop} (blue) and E_{circ} (Green) over time. (B) is the signal from the hoop muscle strain sensor (see Chapter 8.4). When not under external loading, the hoop muscle strain sensor correlates strongly to the E_{hoop} array.

We had success by blending the effects of sensory input and excitatory dynamics in different ratios as a function of position along the body. Compared to our previous model described in Chapter 8.2, all of the innervations of an excitatory array have been adjusted by adding sensory feedback from the strain sensor corresponding to that array.

The amount of feedback depends on the position of the neuronal population along the body. Several relationships have been tested. This blending of inputs is achieved by applying separate weighted arrays that are the length of the simulated body and in the range of 0 to 1 ([*G*] and [1-*G*]) to both the sensory feedback and the excitatory connections (Figure 27). [*G*] scales the local strength of the excitatory connection via the ° operator, which is the Hadamard product, or entrywise product. [*G*] and [1-*G*] are defined such that they sum to unity at any point along the length of the worm. [G] can have many different characteristic profiles, and several are explored in Chapter 10.2. The new state equation for a single excitatory layer now takes the form:

$$\tau\ \dot{E} = -E + \sigma\left(w_1\,[G]\circ\left(E\otimes\hat{E}\right) + w_2\,[1-G]\circ\left(S\otimes\hat{S}\right) + \sum_{i=3}^{n} w_i\left(I_i\otimes\hat{I}_i\right)\right), \quad (8\text{-}3)$$

where S is the stretch receptor input, which has broad connectivity identical in scale to an inhibitory connection, but without an anterior or posterior offset. While we currently only have a linear muscle model, this broad connectivity provides a similar low-pass filter that a more elaborate muscle model would provide.

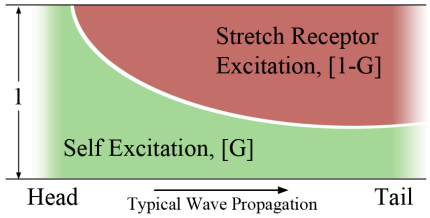

Figure 27: One possible blending of excitatory connections ([*G*]) and stretch receptor inputs ([1-*G*]).

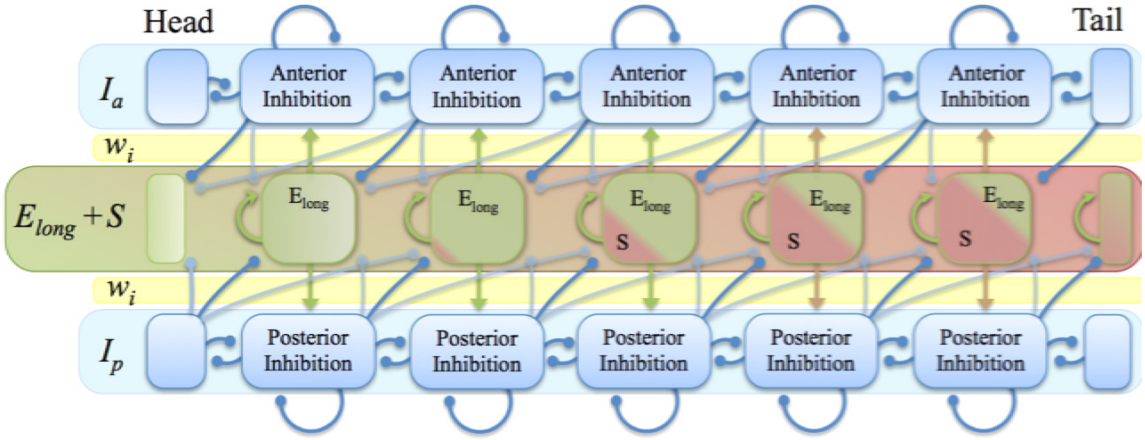

Figure 28: For the purpose of showing several connection trends, the nearly continuous dynamics of a single muscle activation group are represented as five nodes along the body. It is important to note that each synaptic connection shown here is applied to multiple populations through a Gaussian distribution to its neighbors. The excitatory populations (E_{long}, green middle layer) directly stimulate the lengthwise actuators (not shown), while also locally stimulating the anterior and posterior inhibition networks (blue). These two networks in turn inhibit the excitatory network more broadly, but are shifted in opposite directions along the network. Towards the tail, the excitatory connections are combined with strain sensor input (S, in red) from the muscle that the network controls. The weights of the inhibitory connections, w_i, can control the direction, speed and wavelength of the naturally occurring waves in a top-down manner. Not shown are the weak inhibition across the E_{long} and E_{circ} networks that keep the networks phase locked.

9.4 Model of Worm Posture And Stretch Receptors

In order to model the effect of stretch receptors on our control system, Kathryn Daltorio and I made a basic model of worm posture as a function of a given set of muscle activations and vertical environmental constraints, i.e., under the assumption that the environment is essentially frictionless. These forces we ignore here probably do play a very important role in managing properties of the wave, especially during burrowing, where the worm must prevent slip as it applies a force. However, we have shown that tangential ground reactions are small if the wave is symmetric and steady state, so this model can begin to look at how stretch receptors can play a role in wave propagation.

Each segment's position is solved by balancing the forces on a braided mesh

element (Figure 29). We assume that the excitatory layers E_{long} and E_{circ} correlate linearly with muscle force generation, F_x and F_y. These forces are resisted by a non-linear spring with a rest length, L. We assume the weight of the structure can be ignored. If there are no other opposing forces, the posture of each mesh element can be solved independently since all the forces are internal.

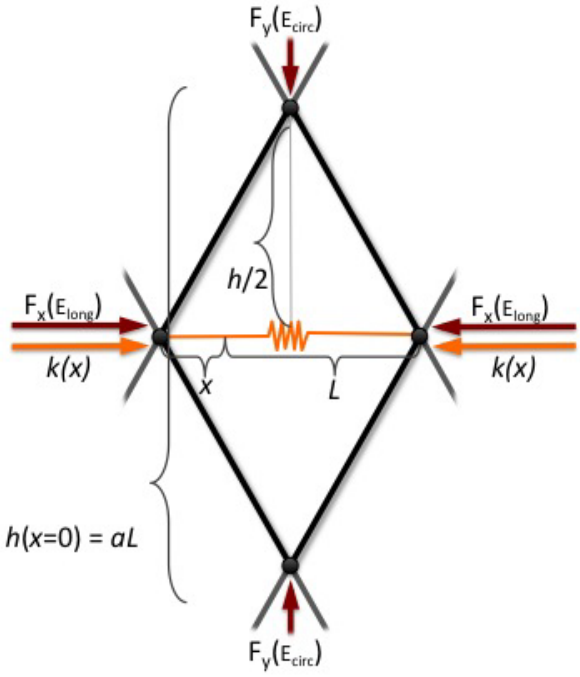

Figure 29: A single braided mesh element used to derive robot pose as a function of muscle activity.

Because this is a constrained system with known forces we can use the principle of virtual work. We use the Pythagorean theorem on the rigid lengths to define the coupling, where *h* is the height of the element and *x* is the extension of the spring. *L* is the resting length of the element and *aL* is the resting height of the element.

$$h = \sqrt{\left(1 + a^2\right)L^2 - \left(x + L\right)^2} \qquad (8\text{-}4)$$

Note that if we want to be able to extend to $x = L$, a must be greater than sqrt(3) in order

to maintain nonzero height. Taking the variation and rearranging results in

$$-\left(\frac{x+L}{h}\right)\delta x = \delta h \quad . \tag{8-5}$$

If $k(x)$ is the spring force (shown schematically in Figure 29), we can model it as k(x) = A tan(x/L · pi/2), where A is a constant. This nonlinear spring ensures that the position of the resultant element is bounded regardless of the input forces. Thus the equation to be numerically solved at each time step is:

$$-F_x\delta x - k(x)\delta x - F_y\delta h = 0$$

$$-F_x - A\tan\left(\frac{x}{L}\frac{\pi}{2}\right) + \left(\frac{x+L}{h}\right)F_y = 0 \tag{8-6}$$

We assume the stretch receptor measures x, the displacement from the rest length (Figure 30 for a plot of x, and Figure 29 for diagram of x relative to a braided mesh element). E_{long} is only stimulated by positive values of x, and E_{circ} only by negative values. We used constants L = .55 mm, a =sqrt(3.1).

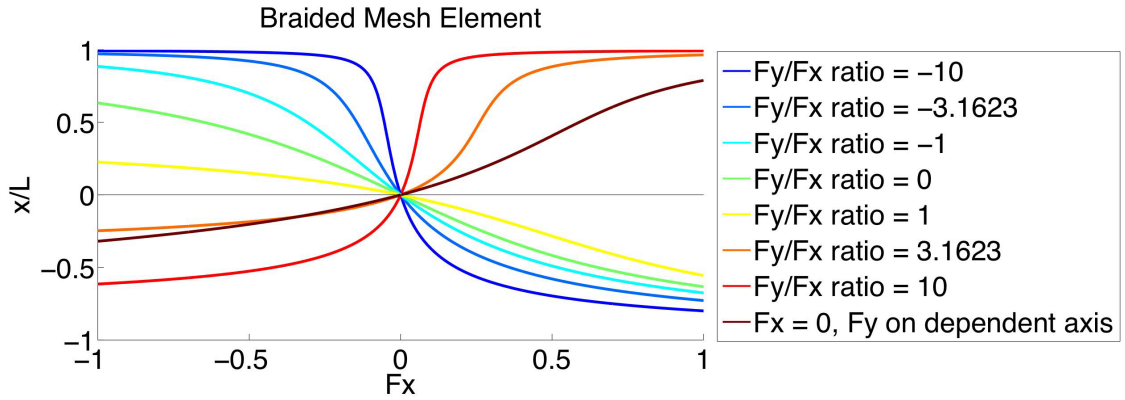

Figure 30: The extension in the length of an element x normalized by the rest length L is demonstrated for several constant ratios of Fx to Fy. Fx is plotted on the lower axis, and Fy is directly proportional. This provides a characterization of the effective properties of the element. Due to the anisotropic properties of the braided mesh and the non-linear spring, the kinematics of very large forces are bounded. In the case of the constraining tube in Figure 9, the length was not permitted to shrink below the rest length (x/L>0).

For the purposes of simulating the position of each segment, the above method was used to create a lookup table whose inputs were the Fx and Fy values from the muscle model at each time step. External forces and kinematic constraints can be added as well. In order to test the simulated worm behavior in a tube that decreases diameter, we applied a kinematic filter such that the maximum height of a worm segment was less than a given value. The results of this experiment are presented in Chapter 10.2.

10 Results From the Prototypes and Mechanical Simulations

10.1 Comparison of the 2D simulation to the Analytical Model

In soft structures, the segments are highly mechanically coupled. To study this, we compared the Working Model simulations described in Chapter 7 of a twelve-segmented worm robot that had one, two, or three waves over the length of the body at any time (Figure 31). The velocity of the deformation waveform, V_{wave}, is the same because the leading edge of the waveform takes the same time to cross the whole body. However, since the number of waves per body length increases, the frequency of undulation increases as well. Nonetheless, the area under the predicted strain curve is the same, so their predicted average speeds are also the same (Equation 3-6).

In contrast to the analytical prediction, we see that the fewer the number of waves along the body, the faster the simulated robot. But the fastest result, at one wave per body length, also has the most retrograde motion, or slip. So there are two phenomena affecting the simulated body speed, and they are effectively described by the correction factors, Q_{strain} and Q_{slip} defined above (Equation 3-10).

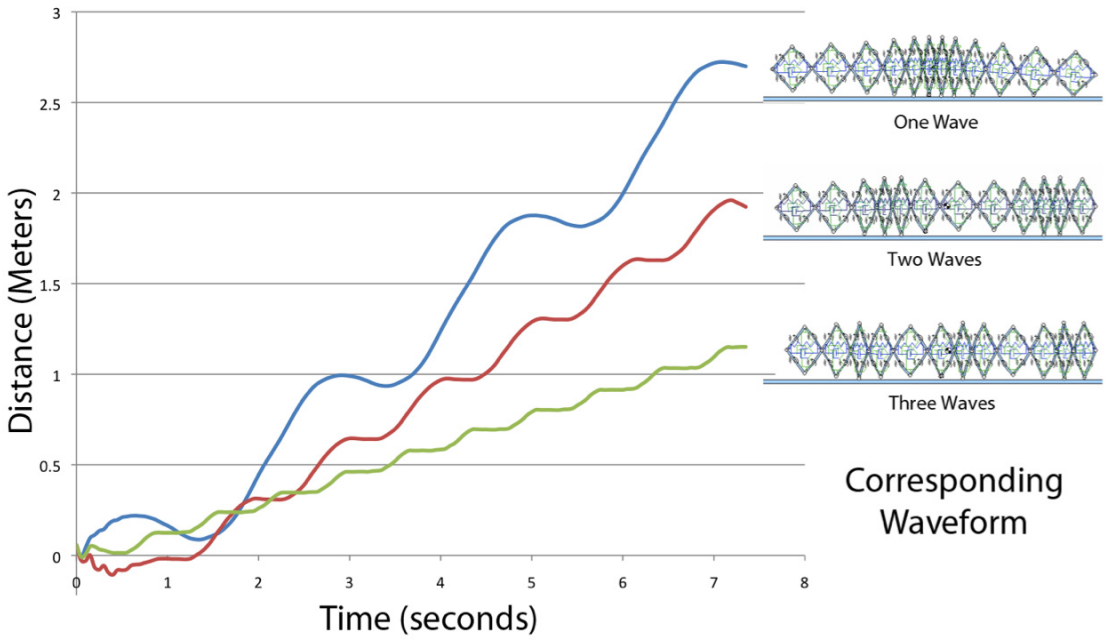

Figure 31: The simulated horizontal displacement of a single segment as a function of time for three different waveforms. Because all three trials have the same deformation wave speed, V_{wave}, and the area under the theoretical strain curve is the same, these trials would have the same speed if it were not for soft body effects such as strain loss and slipping. The body length is an average of 1.5 meters, giving average speeds of 0.17 body-lengths/s for the one-wave trial (top trace), 0.12 body-lengths/s for the two-wave trial (middle trace), and 0.078 body-lengths/s for the three-wave trial (bottom trace).

We compared our analytical model to the simulation by applying a least squares curve fit for the coefficients Q_{strain} and Q_{slip} described in Equation 3-10. The kinematics of the braided mesh were taken from the derivation Chapter 6.1. The hoop actuator position was found using the kinematics of the cam mechanism described in Chapter 6.2.

We found that combining both coefficients provided a good agreement (Figure 32 and Figure 33). Taken independently, the Q_{strain} coefficient was better at adjusting the analytical models of all three waveforms to match the simulation results (one and three waves shown). This suggested the final strain was significantly less than the theoretical 4-bar model predicted (Equation 6-7). We then measured the maximum strain in the one-

wave 2D simulation manually, and found it to be 1.8, compared to the predicted a peak value of 4.2 (Equation 3-1). This is a strain efficiency of 0.43, which is very close to the Q_{strain} value of 0.4 found using the least squares method.

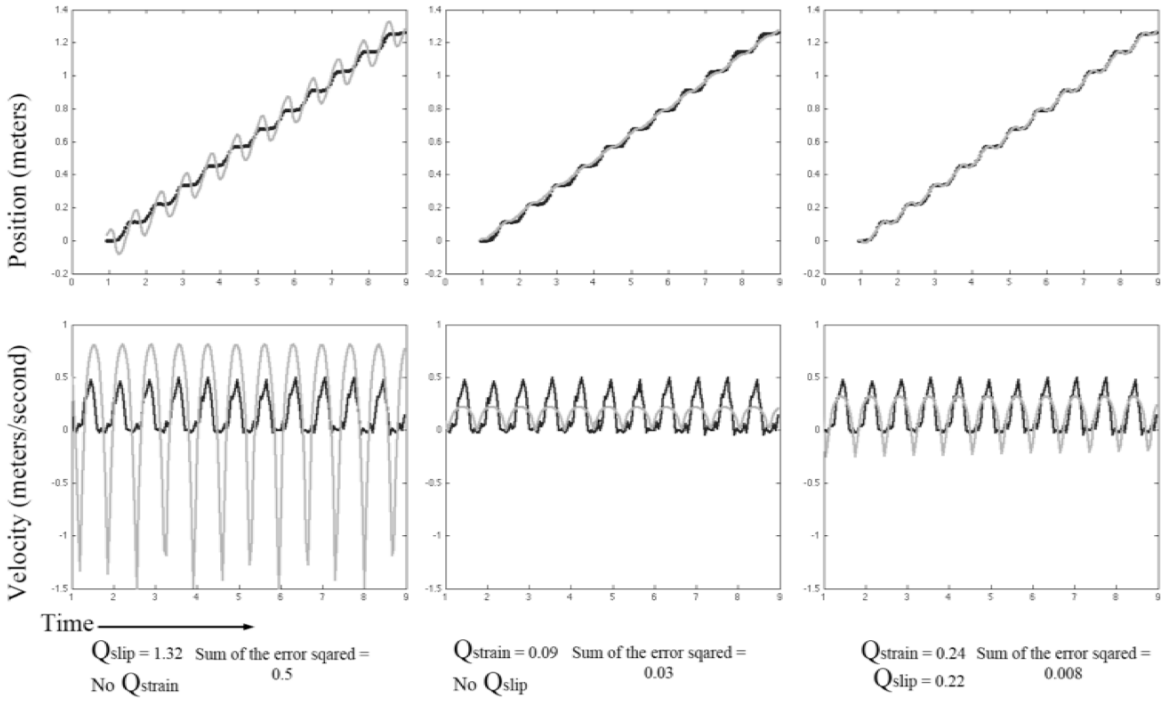

Figure 32: Position and velocity data taken from the three-wave simulation (dark trace) is compared to the analytical model (light trace). The data cannot be modeled well by using the slip coefficient alone (left). Using the strain coefficient instead greatly improves the fit (center), but the best fit was achieved by using both coefficients (right).

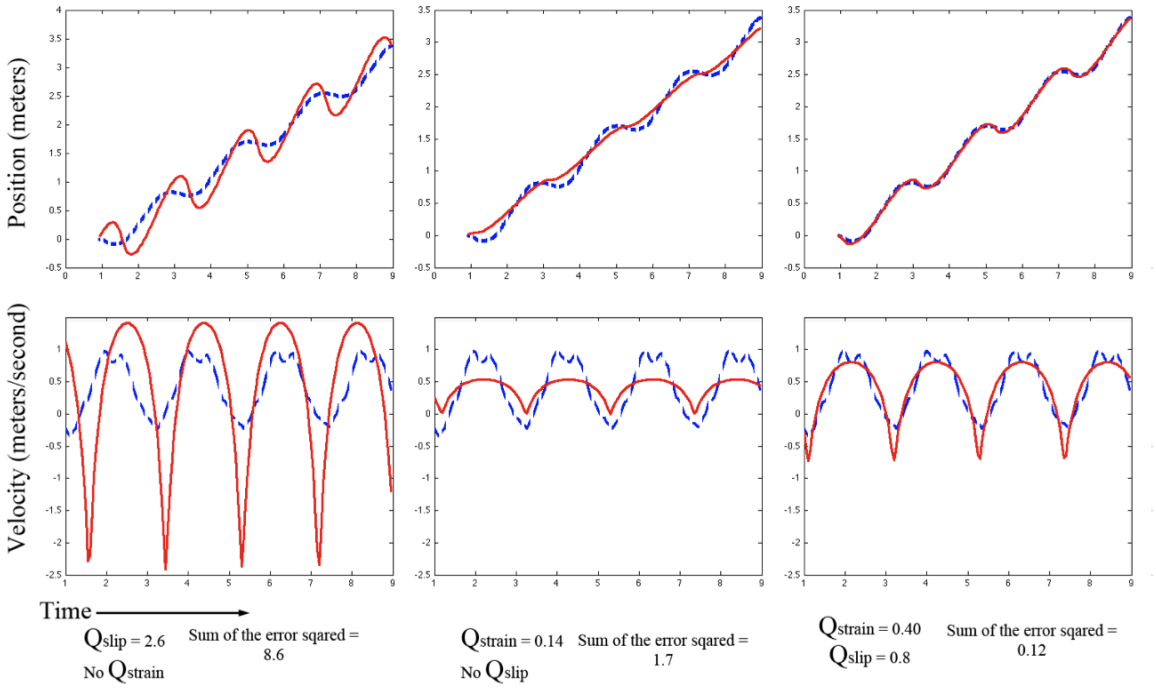

Figure 33: Position and velocity data taken from the one-wave simulation (dashed trace) is compared to the analytical model (solid trace). Like the three-wave simulation, the one-wave simulation is modeled most accurately using both Q coefficients (Right), as opposed to one or the other (Left and Center). However, here slip losses dominate instead of strain losses. The maximum slip is over-estimated due to inaccuracies in modeling the strain function.

This method of adjusting the Q values still tends to over-estimate the slip that occurs because the theoretical strain function has a sharp transition about the ground contact point that does not occur in the soft-bodied structure (Figure 33, right). In other words, while the slip rate may be constant over a cycle, the strain efficiency is not. The strain efficiency is the worst around the time of ground contact. In simulation, it is straightforward to measure the strain directly, so we used this data to compare the measured strain rate at a given location to the velocity at that location. The analytical model states that the velocity of a segment will be equal to the strain at that segment times the speed of the deformation wave, minus a slip constant (Equation 3-10). Figure 34

shows that the analytical model does an excellent job of predicting the velocity of a segment once the simulated robot has reached a maximum average speed after two cycles.

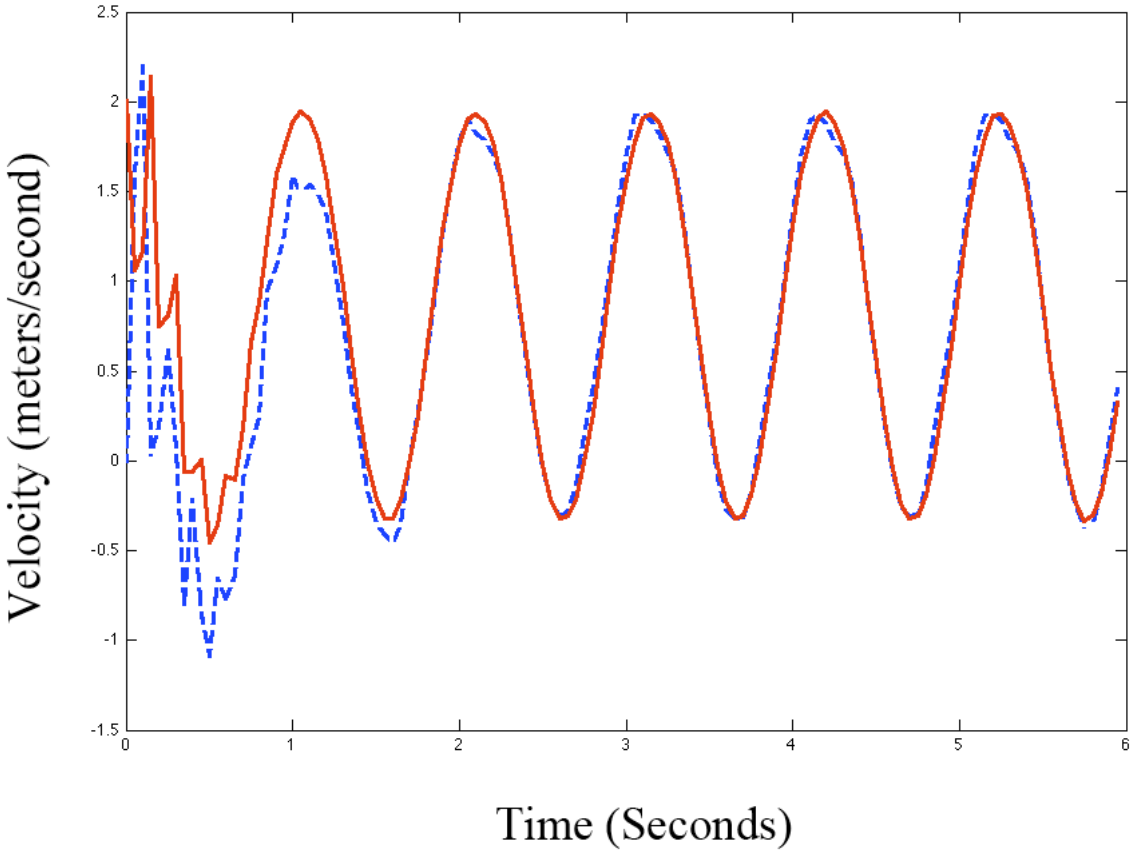

Figure 34: A comparison between the velocity predicted by the measured strain of a segment (solid trace) to the velocity of that segment in simulation (dashed trace). Since we are measuring the strain directly, Q_{strain} is not needed to adjust the curve profile. Initially, the velocity is less than predicted, as the robot slips more during the initial acceleration. In the first half second, the body is also adjusting its pose to the initial position commands. Once a maximum average speed is achieved, the velocity and strain data correlate very well (average error squared =0.0054, Q_{slip} =0.31).

Comparing the Q values of the one-wave and three-wave simulations, we can see that the three-wave model has much more strain loss (Q_{strain}= 0.24, as opposed to Q_{strain}= 0.4) and the one-wave model has more slip loss (Q_{slip} / Q_{strain} = 2 for one wave as opposed to 0.92 for three waves). Note that the two coefficients scale differently because the strain coefficient is multiplicative, whereas the slip coefficient is subtractive (Q_{strain}=1, Q_{slip}=0

means 100% efficiency). Because of the slip and strain, the three-wave model has difficulty generating the desired strain. This may be because of coupling across segments, since only four actuators are defining each of the three waves, as opposed to twelve actuators used for a single wave. On the other hand, the one-wave simulation slips more both in absolute terms and as a percentage of the stride length. Examination of the simulation video suggests this is due to the fact that a single wave cannot effectively keep the swing phase segments off the ground. As segments contact the ground at inappropriate times, they apply forces on the stance phase segments that counteract forward motion, as our analytical model predicted. This negative motion is reduced when there are two or more ground contact points, as it prevents the body from sagging. These results suggested using two waves in the robot prototypes.

10.2 ROBOT PROTOTYPES

The first prototype generated the desired waveforms successfully for short periods of time. A speed of 0.97 m/min was achieved over a distance of 0.9 meters. The speed was intentionally slow in order to help diagnose problems. After that distance, individual actuator cables began to slip, causing non-uniform wave propagation. The robot still moved forward until four of the ten actuators had become non-functional. This suggests that in more rugged environments, the device will be robust to individual actuator failures.

The second prototype addressed a number of issues that arose during testing of the first prototype, most of which involved refining the method of guiding the actuator cables. The swivel joints at each mesh juncture greatly improved the reliability of the device and increased the speed of the robot by significantly reducing the friction along the actuator

cables. By moving the cable clamping location from the cam mechanism to the far side of the hoop actuator, cable slippage was completely eliminated.

These improvements allowed the drive motor to operate at 54 rpm, allowing the robot to travel at 4 meters per minute (Figure 35). Fluid waves of motion were observed with little retrograde motion. The small rubber feet that the worm moved on were completely symmetrical, and did not noticeably resist backwards motion any more than forward motion. Because the body was rigid enough to prevent early ground contact and the task did not involve burrowing, strong anisotropic friction forces were not necessary to achieve good speeds.

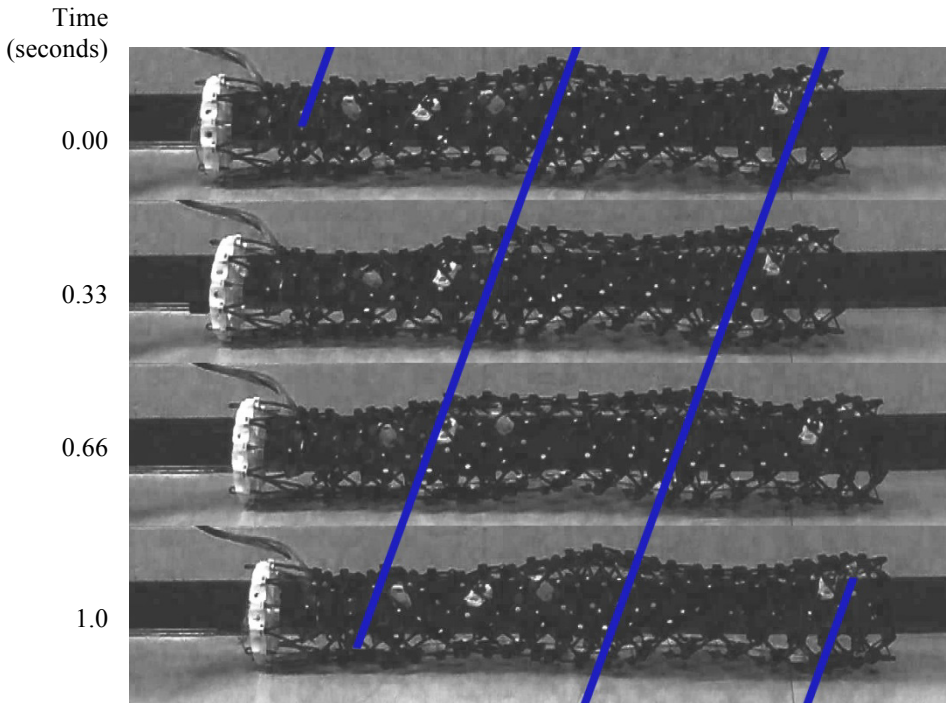

Figure 35: Stills from a video of the second prototype moving over 1.0 seconds. The diagonal lines indicate the smooth continuous rearward progression of waves.

The analytical model was compared to motion capture data taken from the second prototype trials. In the same procedure used to model the simulation, we took dimensions

directly from the robot and found the Q values. The deformation wave speed, V_{wave} was measured directly from the motion capture data by observing the rotation of the cam mechanism. It takes two full rotations of the cam mechanism for a wave to travel from the front to the back of the robot. (Figure 36, Left).

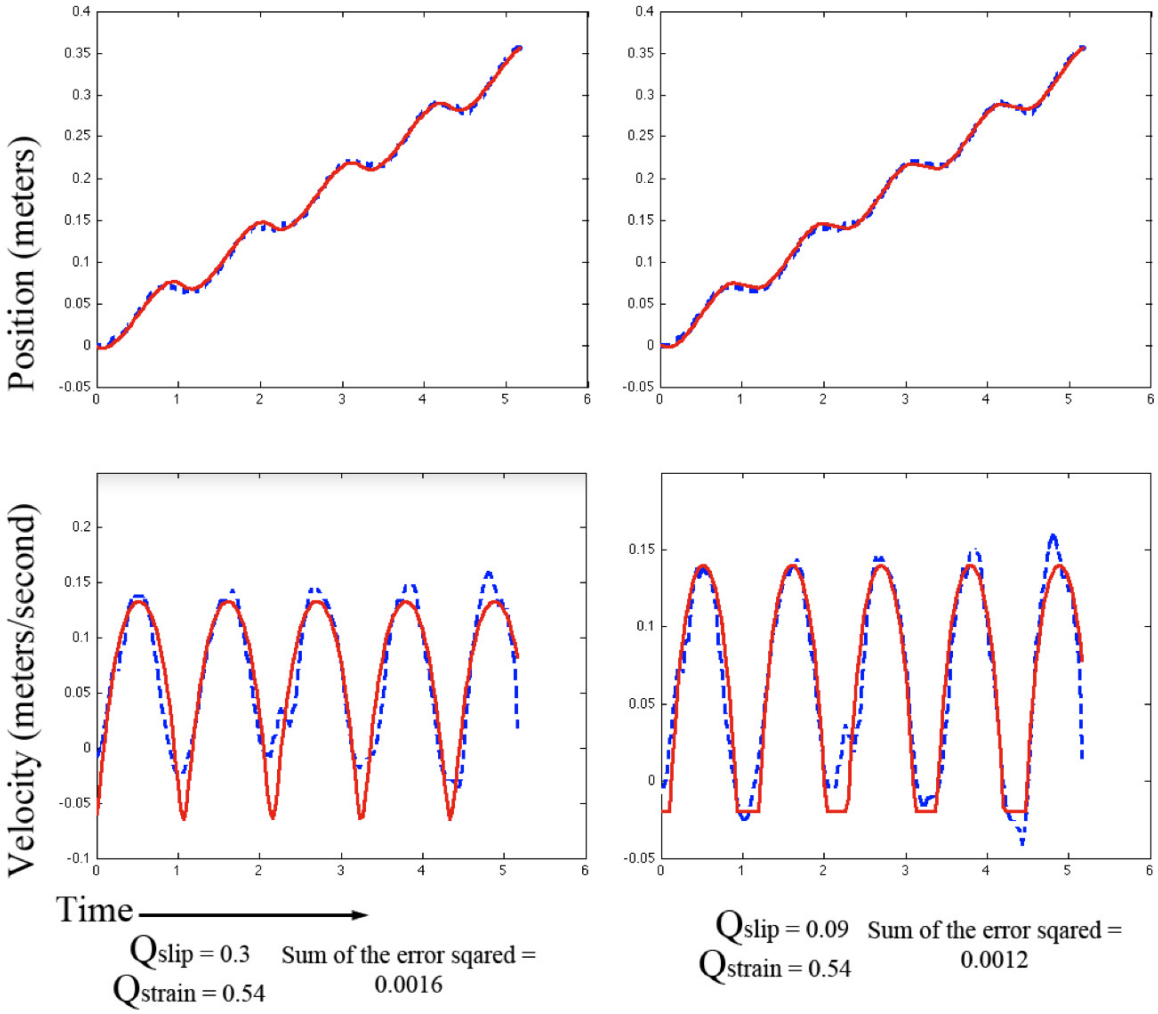

Figure 36: Comparison of velocity and position in the analytical model (solid trace) to trials of the second prototype (dashed trace). The prototype data was captured using video and WINanalyse. Left: the analytical model is based on dimensions taken from the robot. Using a least squares fit, Q_{strain} was found to be 0.54, very close to the measured value of 0.52 at the extreme strain. Q_{slip} was found to be 0.3. Both these values compare favorably to the simulation, however this approach overestimates the slip, as one can clearly see the maximum negative velocity is much less than the model suggests. Right: re-modelling the strain function to not exceed a certain value improves the data modeling.

Using a least squares fit method to find the Q values results in an analytical model that best fits the position data (Figure 36, Left). However, the velocity curve dips well below zero around the time of ground contact, to a more negative value than that recorded by the motion capture data. This prompted us to look at what was happening around the time of ground contact. While we expected the start angle α to be around 30 degrees, after the robot started moving, the angle only returned to 45 degrees. In other words, the beginning and end of every strain cycle were being clipped. There could be a number of factors contributing to this. We observed that the cable itself was slack around the cam mechanism during stance phase, so the hoop actuator had not returned to its fully expanded state. The return spring that causes axial contraction, and thus expands the hoop actuator, may not be strong enough to oppose the friction forces within the cable sheathing. Increasing the tension in this spring would improve the strain function, but make it more difficult for the hoop actuator to contract.

Since this kind of strain loss is not smooth, the Q factors have trouble modeling it. As an alternative, we added a numerical clipping plane, such that the strain function was truncated to a minimum value that was optimized along with the Q factors. This technique yielded the best results for both position and velocity accuracy (Figure 36, right), and reduced the Q_{slip} from 0.3 to 0.09, suggesting that the error in the strain function model was causing the appearance of more slip than was actually occurring.

10.3 Discussion of the Prototypes

We have identified several advantages of using a continuously deformable outer mesh to achieve peristalsis. It allows for a straightforward method of accelerating and

decelerating the segments of the body in such a way that the internal forces cancel. It simplifies ground contact timing, a critical factor in keeping the system from becoming over-constrained. And lastly, a continuously deformable structure is far less likely to be impeded by obstacles, as it has no dead band between segments. While these benefits do not preclude the need for friction in order to climb or burrow, locomoting in this manner will still be advantageous in these more challenging situations.

Given a continuously deformable robot, the analytical model defines only two ways to move faster: by building waveforms with higher strain rates, or by generating faster waves (Equation 3-5). The shape of the waveform deformation is limited by the need to have ground contact, and to prevent premature ground contact in the segments about to touch down.

Premature ground contact was frequently observed in our 2-D simulation, even in the rearmost segments. When the ground contact point switches from one segment to the next, the second segment often contacts the ground before it has fully expanded radially. Therefore, after ground contact, it will continue to expand radially, instead of contracting as part of the next cycle. This means that the wave gets unnaturally stretched due to too many kinematic constraints, and at least one of the ground contact points must slip. In this situation, anisotropic frictional properties are beneficial by forcing the robot to slip forwards, rather than backwards.

Problems can also occur during the ground-to-aerial transition. The analytical model shows that the acceleration of the segment would be greatest during the very beginning of the aerial phase when the change in strain is the greatest. Figure 37 is derived from Equation 6-7 and shows that, given a set displacement, c, the initial angle, α, is a

critical factor in the amount of axial strain that is achieved. However, in both simulation and prototype, the strain that was supposed to occur during this time period did not develop. While the most strain is achieved with small start angles, the forces required to move are high, due to the small mechanical advantage. Because the mesh is soft and flexible, high actuator forces can be impractical. The braiding along the hoop actuators will not transfer the forces to the immediately adjacent mesh before buckling. It would be advantageous to have the smallest initial angle possible that does not induce buckling.

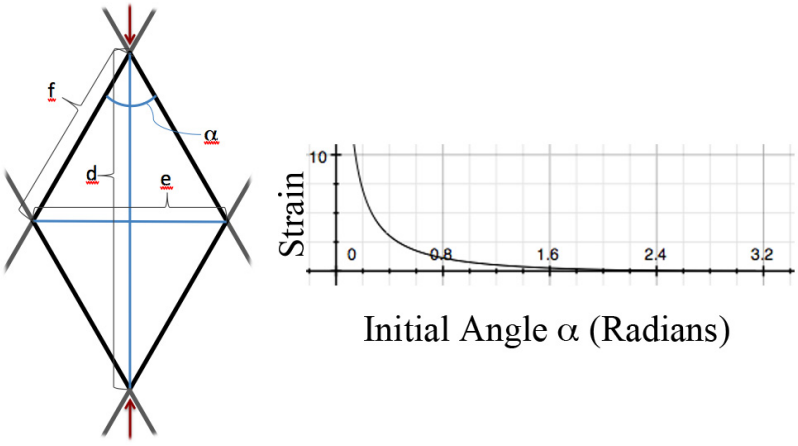

Figure 37: Strain as a function of the initial angle α, with a fixed displacement.

Our second prototype achieved a speed of 4.5 or 6 body lengths per minute, depending whether the cam mechanism is included as part of the body length. This is a fast speed for peristaltic locomotion. Comparatively, earthworms travel at speeds of 1.2 to 3.6 body lengths per minute [Quillin 1999], and the previous peristaltic robot using four discrete segments from our group traveled at 0.8 body lengths per minute [Mangan 2002]. While the Q correction factors do show that there can be some improvement in both developing the desired strain and preventing slipping, the biggest contributor to a loss of

speed was most likely the inability of the mesh to fully return to its rest position (maximum diameter) where the most axial strain is developed. As the flexible 2-D simulation showed, mechanical coupling can have an effect on effective strain rates, especially with two or three waves. Lastly, early ground contact may have played a role in reducing speed, as gravity tends to pull the robot towards the ground prematurely. Nonetheless, these obstacles were substantially overcome by a method of actuation that can produce high frequency waveforms with ease.

10.4 PROTOTYPE CONCLUSIONS

With the help of my colleagues, I have developed a novel analysis of peristaltic motion, have captured its essential predictions in an analytical model, and have tested the predictions of the model by building prototypes and comparing the model predictions to the empirical results. We found that our previous robot, and nearly all other robots that claim to use peristaltic motion, move much more slowly than predicted because of the kinematics and dynamics caused by very long actuators that greatly exaggerate the segmentation of the robot. Our study of the kinematics of peristaltic motion, both in simulation and using analytical tools, suggests a new design of a worm-like robot with a continuously deforming outer mesh. I presented several methods of constructing such a robot with a continuously deforming exterior at different scales, and reported on the completion of two large-scale prototypes. Several novel mechanisms allow for simplification of the control problem by coupling the degrees of freedom. Both an analytical model and simulation effectively describe the motion of the prototypes while suggesting ways to further improve speed and efficiency. Using a continuously

deformable outer mesh has resulted in great improvements in speed and performance over previous worm-like robotic platforms.

11 Results From the Control Network

11.1 Analysis of the Open Loop Control Network

The open loop control network described in Chapter 8.2 produced a wide variety of waveforms by modulating the strength of the connections from the two sets of inhibitory populations to the excitatory populations. As shown in Figure 38, the waveform can travel both forward and backwards at a wide range of speed and even come to a complete stop (static wave), and can change its spatial frequency.

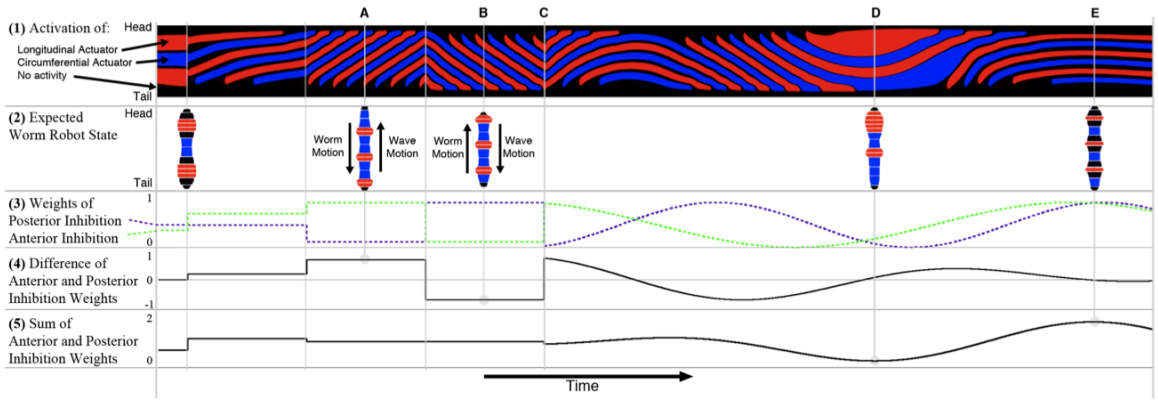

Figure 38: Waveform shape and varying inhibitory strengths. In row one, the vertical axis represents the position along the long axis of the body, and the color intensity denotes activity in either the circular (blue) or longitudinal (red) actuator controllers. The horizontal axis is time for all traces. At t=0, there are 1.5 waveforms along the body (row two). The waveform initially travels towards the head, causing backwards motion, then switches between A and B, causing forward motion. The third row shows the normalized weights of the posterior and anterior inhibitions as they change with time as a result of descending signals. The fourth and fifth rows show the *difference* and *sum* respectively of the inhibitory weights. These quantities correlate well to properties of the resultant motion. **A** and **B** are the local maximum and minimum of the *difference*, and correspond to the greatest positive and negative temporal frequency. **D** and **E** are the local maximum and minimum of the *sum*, and correspond to the least and greatest spatial frequency. First 50 time-steps not shown.

As an analogy, one can look at the excitatory layer as a heavily damped springlike element that is being pushed on both sides by the two inhibitory elements. If the forces increase (*w*'s increase), then the spring compresses more. If the forces are the same, the spring does not translate. However, if they are not the same, then the spring moves away from the greater force. The greater the difference between anterior and posterior inhibition, the faster the spring moves. This is shown in Figure 38 in the fourth and fifth rows. When the difference in anterior and posterior inhibition is high, the temporal frequency is high, and the waves travel in the direction of least inhibition (Figure 38 A, B). When there is no difference, the waveform becomes static. The sum of the two inhibitory weights positively correlates to the spatial frequency (Figure 38, E). These correlations are largely decoupled with each other, allowing for a wide range of controllable waveforms (Figure 39).

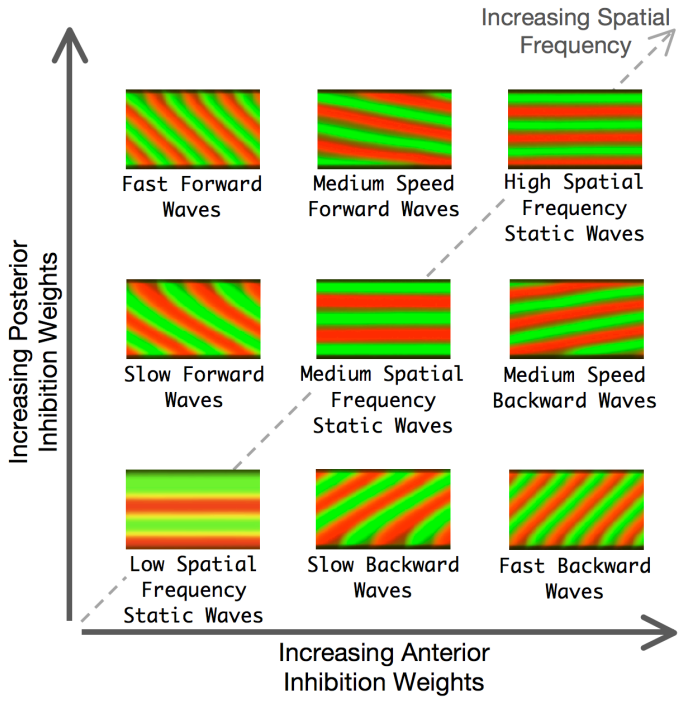

Figure 39: The decoupled effect of anterior and posterior inhibition weights on waveform shape. Thanks to Andrew Horchler for help with this figure.

11.2 ANALYSIS OF THE CLOSED LOOP CONTROL NETWORK

I tested several different techniques for integrating strain sensor information into the control of a wormlike robot. Based on our knowledge of worm neuroethology, I have focused on strain sensors. While I tried many different arrangements, the only family of arrangements I found that were stable and desirable were when the strain sensor information from a given muscle control network was integrated with the excitatory connections, E_{long} and E_{circ}.

Within this family, I tested many techniques of blending excitatory connections and strain sensor input (Figure 40). The dynamics of the network described in the previous section without sensory input had desirable properties, but were inherently open-loop (Figure 40, upper left). On the other extreme, sensory input alone was not enough to generate stable wave patterns (Figure 40, upper right), even when the wave started under the natural dynamics of the system (Figure 40, lower left). As an alternative, we reduced the excitatory connections from head to tail, such that in the absence of any sensory input, the system dropped to zero activity about half way down the body (Figure 40, lower right). However, in the second half of the body, there are still excitatory connections, and combined with the presence of strain sensor feedback, the wave stays above threshold and continues down the body (Figure 40, center). If the worm is burrowing, but making little or no progress, this would be a desirable behavior, and potentially more energy efficient.

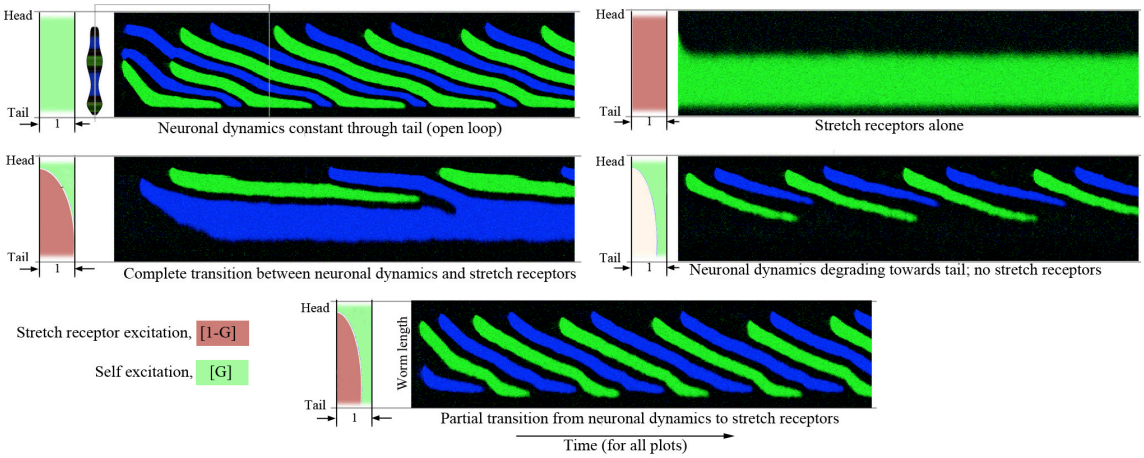

Figure 40: Five different [G] filters and their resultant behaviors. E_{long} (green) and E_{circ} (blue) are shown from head to tail on the y axis, and over time along the x axis. To the left of each simulation is a representation of the ratio of strain sensor (red) to neuronal dynamics (teal) from the head to the tail.

I was also able to demonstrate this network's ability to respond appropriately to a given sensory input. An external constraint was applied to the kinematic simulation such that the final expanded diameter of the robot would not exceed a fixed value. This is akin to a worm traveling in a fixed tube whose diameter decreases half way down the tube. This in turn affected the strain sensors. Near the tail where the wave propagation is dependent on the strain sensors, the lengthwise actuator wave drops off early (Figure 41). Because of the new kinematics of the tube constraint, this does not cause a loss in speed, and therefore it is effectively more efficient than its open loop counterpart (Figure 42).

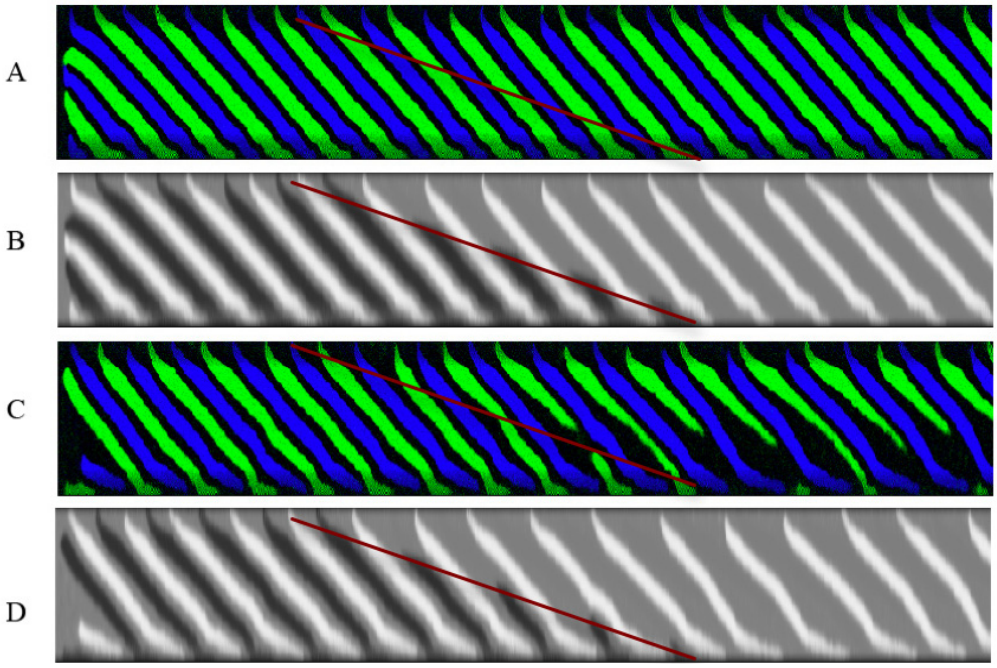

Figure 41: The system responds to a simulated environment by changing its activity. (A) is the open loop signal, and (C) is the closed loop control signal, where E_{circ} is shown in blue, and E_{long} in green. (B) and (D) are the respective body positions over time, where dark gray is fully expanded axially, and white is fully contracted. The red line in each trace indicates the gradual transition from a large tube to a smaller tube, and to the right of the line, the robot can no longer fully expand, as indicated by the absence of dark gray lines. The closed loop network (C) adapts to the change in diameter by decreasing activation of the E_{long}.

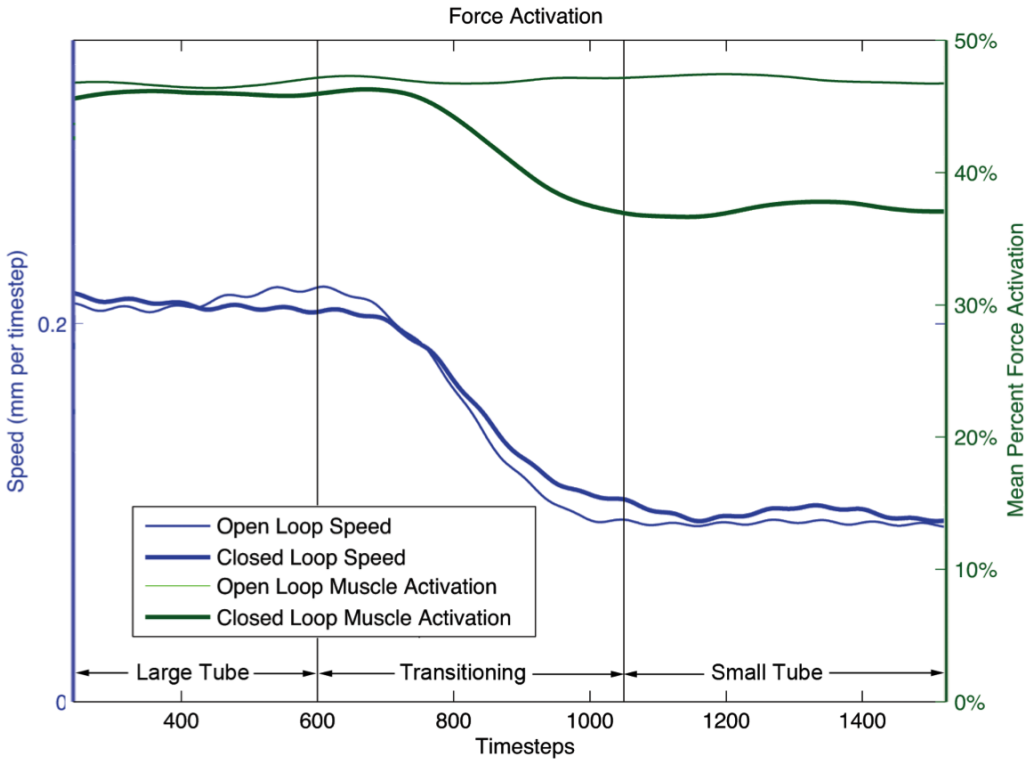

Figure 42: The speed for both open and closed loop control decreases when the constraint is applied (in between vertical lines). Only in the closed loop case does the muscle activity decrease in response to the constraint. The speed is determined by assuming the worm does not slip at the widest point, and the mean percent force activation is the sum of all activations across E_{circ} and E_{long} averaged across all segments. The data has been smoothed with a running average. Thanks to Kathryn Daltorio for help with this figure.

11.3 DISCUSSION OF THE CONTROLLER

Based on our analysis of peristaltic motion, the shape of the waveform (spatial frequency) and the speed of the waveform (temporal frequency) are essential parameters for controlling behavior in a worm robot. The open-loop controller described in Chapter 9.3 can change these parameters in a robust and straightforward way. Of particular interest is the fact that any given waveform can be brought to a complete stop (zero temporal frequency) for arbitrary amounts of time. This would be useful in a robot when it needs to process sensory input before proceeding. It would also make it trivial to activate and de-activate sub-oscillations that could dramatically transform the resultant motion.

The system is robust to sudden changes in descending signals, regardless of the timing or magnitude of the change (Figure 38 A–C). While changes in the speed of the waveform can take place almost instantaneously, changes in the shape of the waveform itself are much slower. This could prove very beneficial in the robotic platform because our analytical model showed that sudden changes in the shape of the waveform can cause slip.

Many sensors play important roles in locomotion. I chose stretch receptors as a starting point for two reasons: there are existing theories on how stretch receptors may affect wave propagation, and I was able to build a straight-forward model of the robot kinematics such that I could simulate stretch receptor signals. The latter requirement was a challenge, and it would be very beneficial to have a full dynamic or quasi-static model of the worm or robot that could incorporate ground reaction forces, and possibly even soil compaction. Without this model, it is hard to know if the system is responding effectively. Alternatively, a robot could provide a real test bed for different control architectures. With

our current stretch receptor model, the question became “What can we show with this simple model?” rather than, “how can we build a fully functioning control network?” Nonetheless, this work is an important step in the right direction.

The closed loop system described in Chapter 8.3 has some properties that are commonly found in biology: opposing muscles are applied to each segment, each repeated segment process is networked to its neighbors, and there is no explicit environmental model other than the connected network dynamics. The way in which the strain sensor propagates the wave is almost the opposite of a proportional-derivative (P-D) controller. When an axially expanded segment senses that the one in front of it is contracting, instead of fighting this change, the segment begins to contract. This is how the wave of motion propagates in simulation, and is consistent with observations of the role of stretch receptors in worms. This implies that if a segment is attempting to expand into a rigid tube, it will only exert so much force before stopping. While this may be beneficial for energy consumption, the worm must also generate enough outward force to provide friction for tasks such as burrowing. I suspect that for this reason, other sensory input, such as pressure and touch sensors may have different dynamic effects on the system, more akin to a P-D control loop, to ensure that enough burrowing force can be generated. With our next generation worm robot prototype and simulation, one can begin to explore these more complex relationships by adding internal forces and improved muscle modeling.

11.4 CONTROLLER CONCLUSIONS

With the help of my colleagues, I have built a controller for peristaltic motion in a robot that uses the Wilson-Cowan model to simulate large excitatory and inhibitory neuronal populations. By combining models of peristalsis, the neuroethology of earthworms and the dynamic properties of neural control circuits, I have created a simulated neural architecture that shows many desirable characteristics. It is robust to sudden changes in descending signals. It can quickly and accurately adjust both the shape of the deformation wave, as well as the speed of the wave, two key parameters to controlling peristaltic locomotion. This arrangement can easily tune its period and even come to a complete stop for indefinite amounts of time. The controller was modified in a straightforward way so that it could respond appropriately to simulated sensory feedback. I believe that this kind of system will be well suited to controlling the continuously deforming outer body of our novel robot.

12 Future Work

12.1 Introduction

While there are many exciting directions one can take this research, I have limited this discussion to some of the more immediate challenges and possibilities. Please also see Chapter 4, where two designs for small-scale devices are discussed that have not been built, but are presented earlier because they contributed significantly to the design process of the final prototypes.

12.2 The Next Generation Large-Scale Prototype

I see two possible directions for a next-generation large-scale prototype. One would be an extension of the reduced actuation approach, and the other would have many degrees of freedom and be used for studying soft-body control.

I was able to show how turning could be implemented in the reduced actuation prototype. It would be relatively straightforward to add steering (actuated turning) by adding lengthwise actuator cables, possibly connected in series with a spring element. It may also be beneficial to have whole body turning, and then a more articulated head with its own turning mechanism. In some situations, steering at the head may be all that is required, as the body will passively follow where the head goes. A left-right turning robot could also be used to study up-down body tension by simply shifting the center of mass. Lastly, it may also be interesting to have two lengthwise cable pairs, such that 3D turning can be explored. In this case, it may be beneficial to have the pairs be 90 degrees apart in an orthogonal arrangement. But it may also make sense to have them paired more closely

in certain directions, if, for instance, lifting requires greater force than turning (Thanks to Andrew Horchler for this last idea).

I would also suggest that the next reduced actuation robot be roughly two-thirds the size of the existing prototype – just big enough that a hand can fit down the middle for maintenance and assembly. The smaller size should reduce the friction in the cable sheathings. Like the final large-scale prototype that I built, I would suggest running the last few actuator cables separately from the braided mesh, to reduce friction. It may also be better to simply run all the cables directly to the actuator heads. This would free up the braided mesh to be designed with the best properties. The bicycle break cable sheathing has a tendency to sag with time, allowing the robot to slowly deform in undesirable ways. The reduced friction of this approach would also mean that a smaller, lighter motor could be selected (Thanks to Nicole Kern for this suggestion).

Another design concept that would allow us to make a smaller robot would be to fabricate the braided mesh such that it has a seam that runs the length of the robot. In this way, the robot could be "unzipped" in order to access the inside for maintenance or assembly. This would only be feasible if the mesh were not being used to route the actuator cables.

The lengthwise return springs could also be improved upon. In the current setup, latex cords run the length of the robot, but they are secured at each joint. In this way, each segment is independently spring-loaded. I thought this would be desirable because one could tune the stiffness of the segment independently. However, one drawback to this approach is that the spring force drops to near zero as the segment becomes fully expanded. This prevents the spring from having enough force to finish pulling the cable

through the sheathing, thus preventing the segment from fully expanding radially. The speed loss from this is substantial, because the length change would be greatest over this lost motion. As an alternative, it may be better to have lengthwise return springs that cross many segments. In this way, the pre-tension can be higher, and the restoring force would be closer to a constant value, since the total change in length would be a smaller ratio of the total length. For instance, the distance from head to tail is always constant, so a spring stretching from one end to the other would not change length, but only apply a force equal to the pre tension.

12.3 Large Scale Prototype With Many Degrees of Freedom

A large-scale prototype with individual motors at each hoop actuator would allow for a great deal more control. Such a robot could be used as a mechanical simulation platform for the control architectures outlined in Chapter 8. The weight distribution would also be more even, and the robot would have a natural low center of gravity. Also, the moment caused by the motor torques could cancel by simply placing the motors in alternating orientations.

In terms of efficiency, it is not immediately clear which of these two approaches would be more efficient. The reduced actuation approach has a very small number of large motors, which is much more efficient, but loses much more energy to friction losses in the sheathing. A survey of these tradeoffs could be useful before proceeding.

12.4 A PRINTED SMALL PROTOTYPE

The manufacture and assembly of a wormlike robot at very small scales poses several difficult challenges. A robot such as the large-scale prototype has hundreds of small moving parts, and attempting to assemble them at a small scale would be nearly impossible without special techniques. As an alternative, I began to explore rapid prototyping techniques with the help of two Senior Project students, Annette Toluse and Zac Jesse. The idea was to use the same cam driven concept as the larger prototype, but have the entire device consist of a single piece of printed ABS plastic. Instead of having swivel joints, the two helix sets would sit one inside the other, and would be connected with a soft silicone casing. One exciting application for such a robot would be as an endoscopic assist, where the robot is powered entirely by a drive cable within the tool passageway of an existing endoscope (Figure 43).

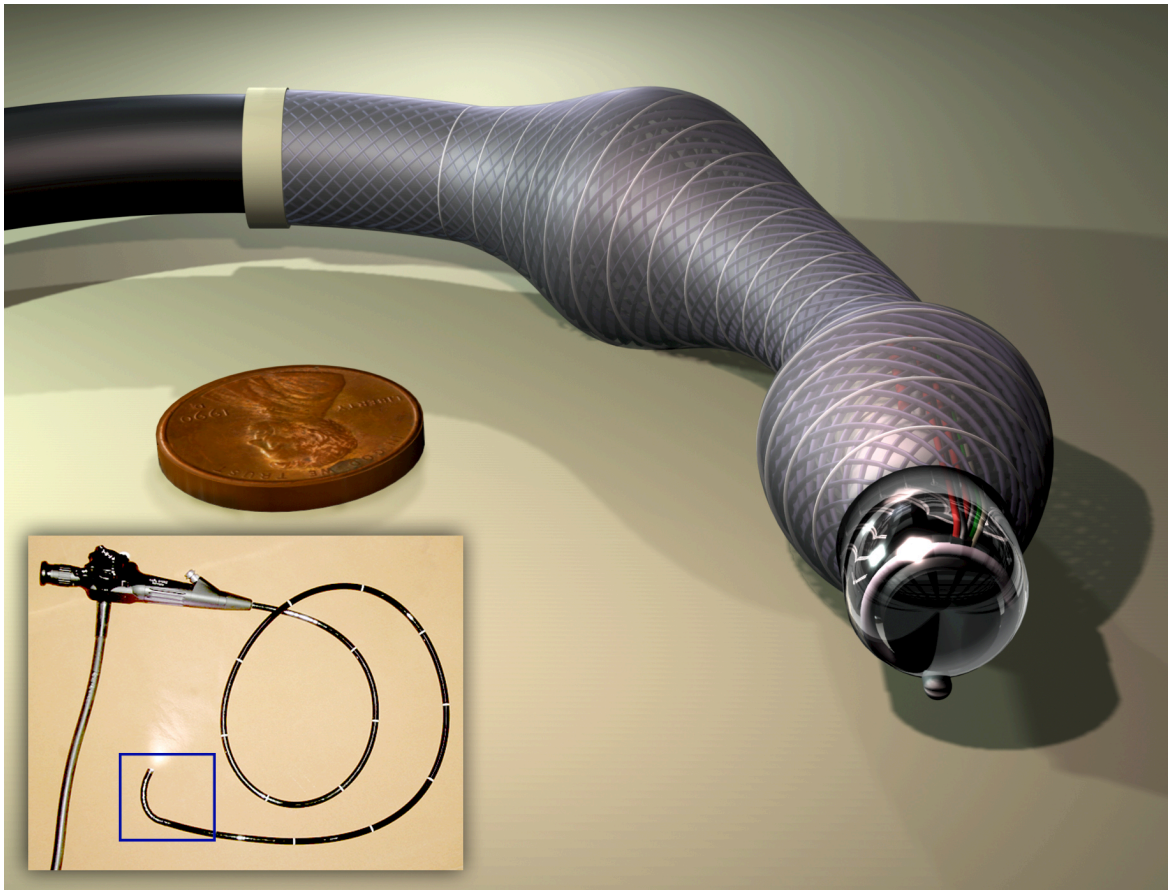

Figure 43: A rendering of a peristaltic endoscope concept. The braided mesh would be powered by a cam mechanism similar to the large-scale prototype. This cam mechanism could be powered by either a small motor, or by a cable running through the tool path in the endoscope. This arrangement has the significant benefit of applying gentle forces at the end of the endoscope, as opposed to large pushing forces applied externally.

Given the resolution limit of the best printer available to us, we predicted that the robot had to be at least 4 cm in diameter so that the smallest features were still functional. In theory, a smaller printer could make ever smaller devices. After a fair amount of design work, a test design was printed as a single piece (Figure 44). While many of the features were successful, others were too coarse to be functional. The braiding itself was also too stiff.

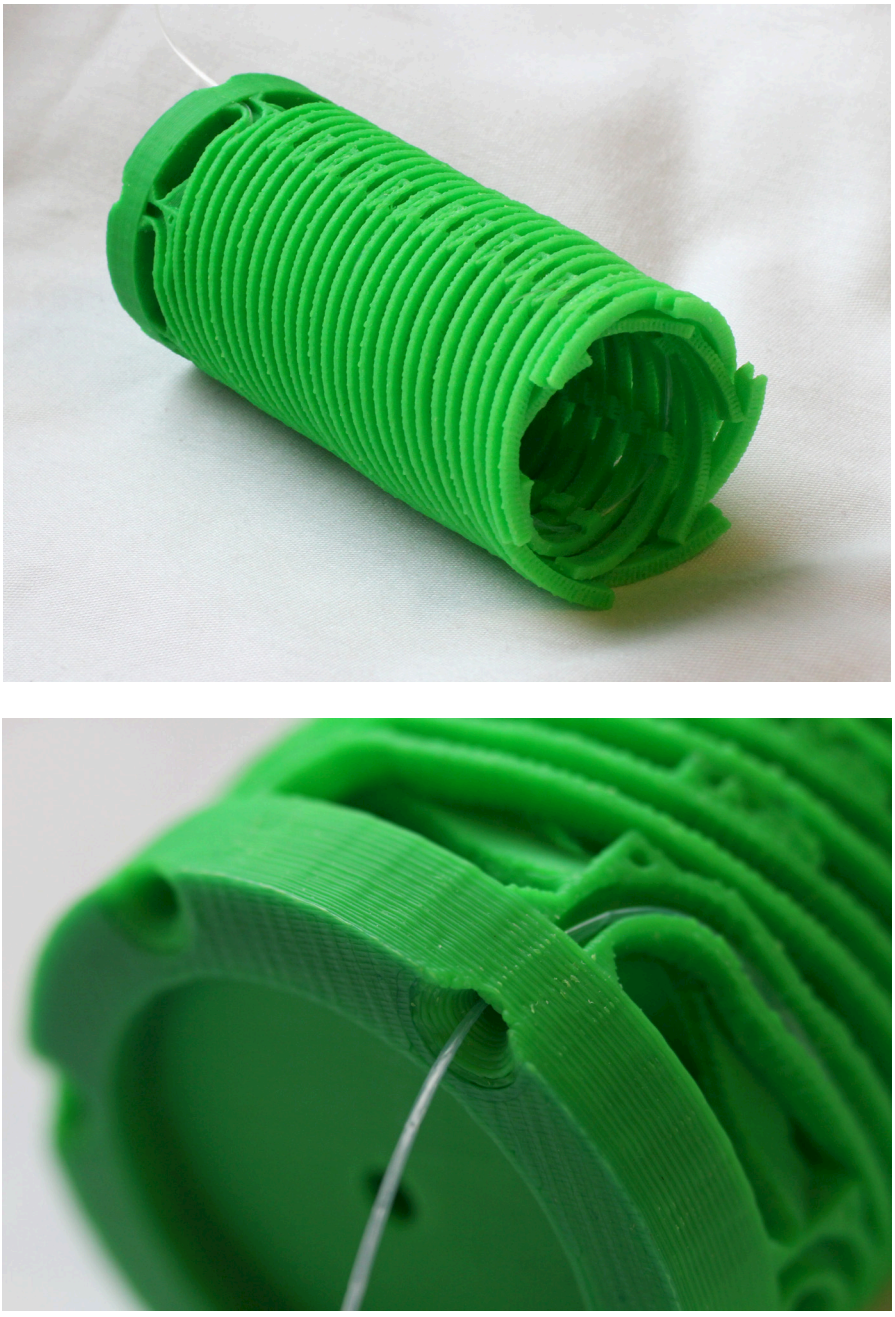

Figure 44: A first prototype of a printed wormlike robot that is 4 cm in diameter. The entire device is shown on top, while a detail of the unfinished cam mechanism is shown on bottom. Fishing wire is used as an actuation cable, which sits inside a thin Teflon tube, which in turns rests inside a groove in the ABS plastic. It can also be seen running through small guide holes inside the robot (top) to make a hoop actuator. At this scale, the texture of the printing would likely cause problems for the cables.

One of the greatest challenges to the large-scale prototype cam mechanism was friction in the cable sheathing. According to a Coulomb friction theory, it does not matter how large the robot is – only how many times the sheathing turns before reaching the hoop actuator. But initial testing done during this Senior Project suggested that the friction forces in the cable sheathing are substantially less at smaller scales than in the larger robot, likely due to the shorter length for potential friction spots to form. It also appears that a close pairing of cable and sheathing diameter is important. If the cable is significantly smaller than the sheathing, the forces are much higher. This may be due to a wedging effect, where the cable deforms the Teflon, causing it to act like a V-belt in a groove.

I think this approach has potential. Future work should focus on rapid prototyping techniques that have higher resolution and produce less scalloping regardless of resolution. Also, a satisfactory method of routing the cables needs to be found. We experimented with Teflon tubes inside the ABS structure, but Teflon will not adhere to virtually any substance, making it difficult to position. It turns out that the ribbed structure of the printed prototype did prevent the Teflon from sliding within the device, but in the long term, this will likely cause the Teflon to wear to failure. Lastly, a better technique needs to be found in order to secure the inner and outer helixes. It may be possible to design a printed joint, or make a joint out of a different polymer using shape deposition techniques [Dollar 2006].

12.5 A Passive Dynamic Pipe Crawlier

Using a braided mesh to achieve peristalsis may have many advantages for navigating pipes. The robot can be entirely hollow, potentially allowing the robot to work inside water mains without depressurizing the line. While exploring this possibility, I came across a method that may allow the robot to locomote against the flow of water in an entirely passive manner. This would be roughly analogous to the way a sailboat can sail upwind. The success of this approach relies on the answer to several very complex fluid dynamics questions, which I will not try to solve here. Instead, I will outline the proposed method and reasoning for why it may work.

This device would consist of a braided mesh embedded in a hollow polymer cylinder. This arrangement is identical to the structure of Festo brand air muscles, but made with significantly softer polymers such that the structure is very flexible. The device should have a rest diameter slightly larger than the tube through which it will crawl. When the device is in the water main, it will then exert a small passive radial force that will allow it to stay adhered to the outer wall. The device can be almost entirely shift-symmetric along its long axis. At the leading edge or lip of the device (the upstream direction) a special mechanism would be designed to encourage the edge of the device to leave the wall of the water main. This mechanism may be just the shape of the lip itself, or in more advanced actuated methods of locomotion, it could be an actuator such as a shape memory alloy. Once the lip of the device leaves the water main wall, it will be obstructing the flow of water. This will further encourage the lip to collapse inward, and water will begin to flow outside the device, as well as through it. However, since the device is made of a braided mesh, the inward collapse of the braiding will also cause the lip to move forward. This phase of the motion will be limited by a minimum diameter of the braided

mesh as it squeezes the polymer within its braid. At this minimum diameter, the lip or lip mechanism will invert such that the downstream flow of fluid will push the lip back out towards the water main wall. Once the lip has made contact with the wall, a bolus of fluid will remain trapped between the wall and the device. This bolus of fluid will be naturally pushed down the length of the device because it causes a constriction of flow within the water that is passing through the device. As the bolus travels downstream, it will cause the device to move upstream, in a manner identical to other examples of continuous wave peristalsis (Figure 45).

If this method of locomotion is successful, then it could be augmented with a minimum amount of actuators to affect other behaviors in a very efficient manner. For instance, it could quickly travel downstream by simply delaminating from the wall. It could also potentially turn corners. However in one possible turning method, the device closely resembles a "Feynman Sprinkler" (Figure 46), one of the most notorious fluid physics problems of all time [Creutz, 2005]. So predicting the motion of such a turn would be very difficult without experimentation.

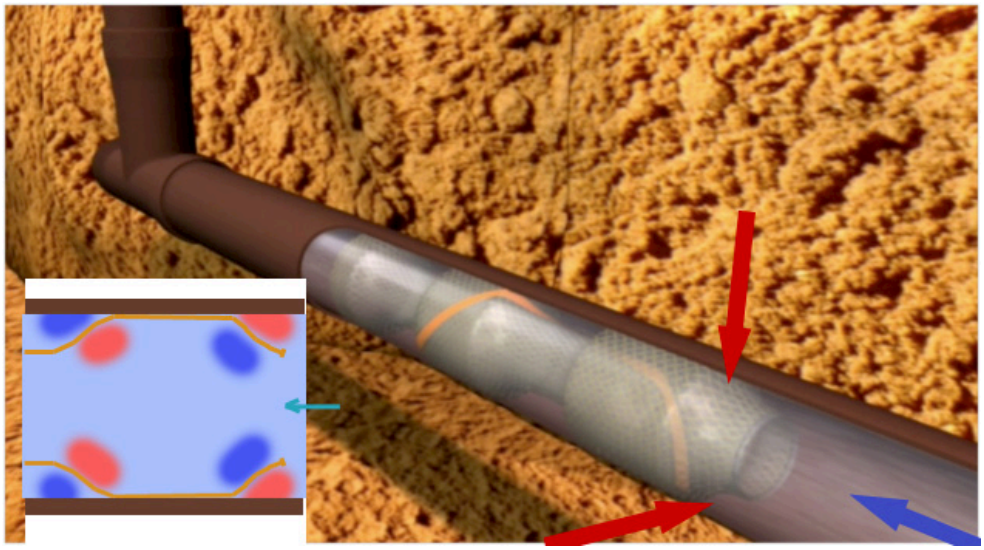

Figure 45: A concept for a passive pipe crawler that uses water flow to travel up stream. A possible pressure distribution that would cause locomotion is shown in the inset.

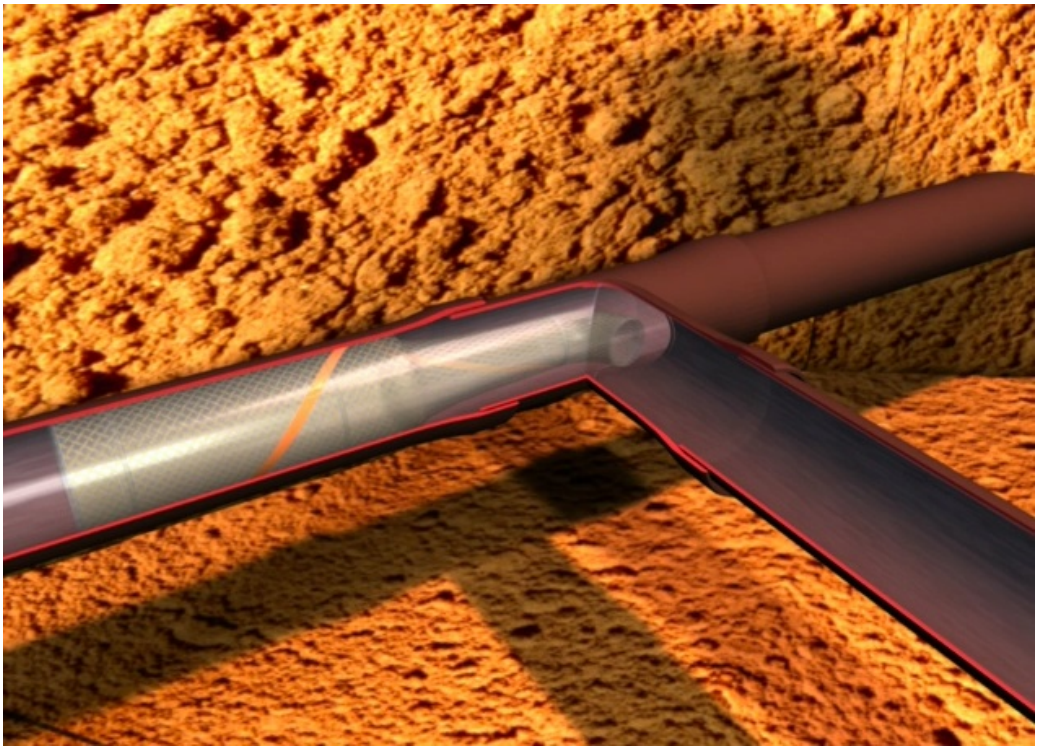

Figure 46: A pipe crawling robot navigating a corner. Such an arrangement resembles a "Feynman Sprinkler," and it is not immediately clear what forces would affect the head of the robot.

12.6 FUTURE CONTROLLER WORK

There are several future directions for the controller network. The code as I have developed it could be quickly modified to study turning. While each layer of neuronal populations is currently only one unit wide, a single line of code can widen that so that it is several units wide. In this configuration, the new dimension could represent separate lengthwise actuators. It may be that such a network would automatically turn to avoid obstacles without any further tuning of the parameters, due to the stretch receptor feedback.

The worm body simulation could use much more development. A quasi-static model is desirable that also handles horizontal forces between segments, and possibly even highly non-linear ground reaction models that could simulate soil. This would allow for a better tuning of the existing neural control network.

Lastly, this research can also inform current work being done with Stable Heteroclinic Channels (SHC's). It may be possible to convert the Wilson-Cowan dynamics to a system that is verifiably a SHC network with a minimum of effort. Even if this is not the case, I suspect that many features of the network I developed, such as the general morphology and the specific use of strain sensor feedback, will be just as important with an SHC network that models peristalsis.

13 Final Thoughts

Over the several years that I have been studying this form of locomotion, I have never tired of it. Perhaps this is because of the simple novelty of a system that appears to flow one way, but in fact flows the other. But that alone would wear off over time. I think what has kept me engaged is that time and again my assumptions have been challenged by the subject matter and by my colleagues. My intuition has been constantly refined - up to the very last day of writing this. This is startling, and kind of silly, in some regards, because sometimes it occurs to me that I have spent years studying a worm. But peristalsis is a fascinating and mesmerizing example of soft-body motion and control, and I think this area of research will be important to both robotics and biology. I hope that this work has made a contribution to our understanding in both fields.

APPENDIX:

MATLAB CODE FOR A WILSON COWAN SOFT BODY CONTROLLER

```
%% Alexander Boxerbaum
%% Dissertation Code
%% 3-30-2012

%% Outline:

% This is a simulation of postulated neuronal activity in an earthworm and
% its effect on the muscles and body pose of a worm un-affected by dynamics.
% Specifically, it generates waves of activity in two directions that could
% be used to control muscles and interface with sensory input (or
% simulaitons of it). This simulation is based on earlier work where I used
% the Wilson Cowan model of neuronal behavior to simulate the primary
% visual cortex. Please see my Dissertaion, Chapter 8, for a detailed
% description of the overall structure of the code.

% While most of the simulation is embedded in this document, it relies on
% several external functions. These function titles are listed and briefly
% described here:

% Sigmoid(x):
% This is a sigmoid function that takes a value and maps to
% a range of -0.1 to 1. The dynamics of the system rely on this function heavily.

% longFlatOffsetHat(Spread, MaxRadius,Offset):
% This is the function that builds the gaussain convolution matricies that define the
% interconnectedness of the neuronal populations.

% LengthSolver(HorizontalTension, VerticalTension, K1, K2):
% This function returns the segment length based on the forces applied to
% it and its natural spring properties. It is derived from the kinematics
% of a single braided mesh element with a spring connecting the horizontal
% nodes. It is a very slow function (because it needs to find zeros of
% nasty equations), so this code uses the function to build a lookup table
% instead of constantly calling it.


%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Start from scratch: Clear the memory; don't report processing stats
clear all
close all
profile off
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% PROGRAM FEATURES:

Noise = 1;
% if 1, turn noise machine on.

SoftEndBoundaryCondition = 1;
% if 1, then the excitatory layers loose strength at the head and tail.
% This helps limit boundary effects.

Clipped = 1;
% if 1, crop all values outside the  range to -.1<x<1. Keeps noise generation bounded.

StaticModel = 1;
% If 1, simulate the effects of the muscles on a static model

CPGDecrement=1;
% If 1, the decending signal fades near the tail.

Animation = 0;
% If 1, then animate the worm position as a function of time. This feature takes
% screen grabs, so you cannot use your computer while it animates!

MotionAnalysis = 1;

KernelDisplay = 0;
% If 1, then display the various convolution kernels that have been generated.

SoilModel =1;

ClosedLoop =1;

WieTestCycle = 0;
% WieTestCycle: If 1, then the WeiLeft and Right values are changed over time; this is
% good for diagnostic purposes, and is akin to having varying descending
% signals modulate the pattern generators.

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% TIME PARAMETERS

dt = 1 ;                    % The numerical size of a differential time element
ImageHeight = 1800;          % The display height of the simulation.
RunTime = 1*ImageHeight;    % The total simulation time is a multiple of the image height.

%This multiple (1 in this case) is the number of vertical strips of data
%that will be displayed from left to right.

Timesteps = RunTime /dt;
```

```
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% DISPLAY AND SCALE PARAMETERS

% The worm is currently one unit wide and 150 units long. Future work could
% expand the width to represent different lengthwise muscles to explore
% turning. Higher resolution can be achieved by making the simulation
% longer, but the excitatory and inhibitory convolution kernels must also
% be scaled.

Height = 1;
WormLength = 150;
Circumcision = 18;
% Circumcision is the number of populations at the head that do not map to the muscle bodies.

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% PARAMETERS for E and I layer connectivity

alpha = .220;   % This is NOT alpha from the W-C equations. It scales
                % all the W parameters equally inside the Sigmoid Function

% In general, Layer connectivity strengths begin with a capital W,
% followed by where they come from, and then where they are going. For instance
% Wie, means that it is an inhibitory connection innervating an excitatory
% layer.

WeeLong =13; %12;    % The self-excitatory effect of the ELong group (Lengthwise actuator stimulation).
WeeHoop =12;
;    % The self-excitatory effect of the EHoop group (hoop or circular actuator stimulation).

WeLML = 9;
% The strength of the effect from stretch sensor feedback. LML stands for Lengthwise Muscle Length.
% It can bre positive or negative, depending on which side of the LMLOffset it is on.

LMLOffset = .5;
% .56 The midpoint of the stretch receptor.

RestLength = .5;
%.55 The dimensionless midpoint of a single strain element when all
% forces are zero; the zero displacement of the non-linear spring.

Wei = 7;        % The excitatory effect on inhibitory cells.
WieCross =  10; % The inhibitory effect on excitatory cells across the circumferential and longatudinal layers.
Wii = 3;        % The inhibitory effect of inhibitory cells on inhibitory cells.

if ClosedLoop
    WieLeft  = 1.5;
    WieRight = 8;   % WieLeft
else
    WieLeft  = 3;
    WieRight = 8;
end
% WieLeft and WieRight are the shifted inhibitory signals that have a
% strong descending control over the shape of the waveform.


WieCenter =4;   % If WieTestCycle is on, then these three constants are
WieMag = 4;     % used to vary the WieLeft and Right values over time for
WiePeriod= 200; % testing purposes.

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% PARAMETERS for the interplay between the sensory input and cpg network

SenseStrength= 2; %
CPGStrength = 1.5;%

% The vector 'Decrement' defines the strength of the CPG network, where the
% strength of the Sensor network is (1-Decrement), evaluated at the location
% along the wormlength. These values define the shape of the vector, which
% is built later. To debug this function, the Plot command is helpful.

if ClosedLoop
    DecStart = 25;
% the number of pixels that constitutes the head, such that the signal controlling it is entirely
% CPG based. Setting this value to the worm length gets rid of any decrement of the CPG signal.
else
    DecStart = 1130;    % Effectively turns off sensory feedback by changinge the gradient blending
end
DecCenter = .7;   % .74 Multiplied by the Worm Length to define the point at which the decrement stops.
DecMin = 0.32;  % 0.32The constant minimum value of the CPG strength after the decrement.
DecMax = 1;

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% PARAMETERS for E and I convolution kernel building

OffsetLeft  = -13;    % The size of the offset used for offset convolutions.
OffsetRight  = 13;    % OffsetLeft and Right are for the posterior and anterior shifted Inhibitory connections.
OffsetCenter  = 0;    % OffsetCenter is used for the non-shifted inhibitory connections, such as IhatCenter.

ExcitatoryRadius = 32;
% ExcitatoryRadius: maximum radius (px) of influence of the E and I
% convolutions. Should be wide enough such that the farthest reaches of the
% kernels are ~0. You can visually check this with Kernel Check Figure.

Espread = 30;         % A measure of the distance of influence of each element in E, the Ehat kernel
Ispread = 6*Espread;  % A measure of the distance of influence of each element in I, the Ihat kernels.
EspreadLarge = 90;    % A measure of the distance of influence of each element in E, the EhatLarge kernel.

MaxOutput = 1;        % When "Clipped" is on, these paramaters set the
MinOutput = -.1;      % upper and lower clipping values. The dynamics of
                      % the system naturally stay within these bounds, but
                      % the noise generator can create values outside of
                      % this. Clipping ensures stability when this happens.

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% TIME CONSTANTS
```

```
tau = 1.5;            % The time constant difference between Edot and Idot.

te = 1;%.8  ;       % Time constants for E and I.
ti = 1;%.8  ;

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Other parameters

ne = 0.09;   % Noise per time step. The magnatude of the noise also depends on the timestep length.
ni = 0.09;   %

BoundaryCondSpreadInside = 1/10;    % How deep the fuzzy boundary condition goes relative to the height.
BoundaryCondSpreadOutside = 1/10;
LRspread = 25;                      % Left-Right boundary condition fuzzification

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Load Ground contact Data:
if SoilModel
GroundContactData = double(imread('GroundResistance3.bmp'));

GroundContactData = (GroundContactData/255');


end

%% HATS %% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Lets build convolutions for E and I:

Ehat =  longFlatOffsetHat(Espread, ExcitatoryRadius,0);             % The Ihat has a wider spread than Ehat.
EhatLarge = longFlatOffsetHat(EspreadLarge, ExcitatoryRadius,0);    % The Ihat has a wider spread than Ehat.

IhatLeft   = longFlatOffsetHat(Ispread, ExcitatoryRadius,OffsetLeft);
IhatRight  = longFlatOffsetHat(Ispread, ExcitatoryRadius,OffsetRight);
IhatCenter = longFlatOffsetHat(Ispread, ExcitatoryRadius,OffsetCenter);

if KernelDisplay ==1

% IMPORTANT NOTE: IhatLeft will appear shifted to the RIGHT. This is
% correct because the resultant effect will be that the excitatory cells to
% the LEFT of the inhibitory cells will have the strongest connections.

    figure(10)
    title('Various Convolution Kernels:')

    axes('Position',[0 0 1 .2])
    %axis image
    imshow(Ehat,[min(min(Ehat)),max(max(Ehat))])
    colormap(gray)

    axes('Position',[.0 .2 1 .2])
    %axis image
    imshow(IhatLeft,[min(min(IhatLeft)),max(max(IhatLeft))])
    colormap(gray)

    axes('Position',[.0 .4 1 .2])
    %axis image
    imshow(IhatRight,[min(min(IhatLeft)),max(max(IhatLeft))])
    colormap(gray)
    axes('Position',[.0 .6 1 .2])
    %axis image
    imshow(IhatCenter,[min(min(IhatLeft)),max(max(IhatLeft))])
    colormap(gray)
end

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Pre-allocation of the State Variables and Data Storage:

% Note that the current starting conditions are 0 for Excitatory and 1 for
% Inhibitory. This can be changed with little impact on the simulation.

ELong = zeros(Height, WormLength)*1; % In the absense of other defined conditions, set to 0
ILeft = ones(Height, WormLength)*1;
IRight = ones(Height, WormLength)*1;

EHoop = zeros(Height, WormLength); % In the absense of other defined conditions, set to 0
IHoopLeft = ones(Height, WormLength);
IHoopRight = ones(Height, WormLength);

MovieStorage = zeros(Height, WormLength,10,RunTime);
% MovieStorage: This array holds a lot of information that can be processed
% after the simulation is done running. The structure of the array will
% seem strange because it is designed so that the final data can be
% cut and tiled to fit the display window nicely.

%% Strain Calculations
% This provides a first order approximation of the strain in various parts
% of the robot. We assume that the dynamics are all feed forward, and only
% affected by the muscle tension.
if StaticModel

    HML = zeros(Height, WormLength);     % Hoop Muscle Length
    LML = ones(Height, WormLength)*.9;     % Lengthwise Muscle Length
    LMLold =ones(Height, WormLength)*.99;
    HMLold =zeros(Height, WormLength);

end

%% Various data logging of stretch and peak stretch locations
PeakHeight = zeros(2,RunTime);
PeakHeightLocations =zeros(2,RunTime);
Stretch = zeros(1,RunTime);

Decrement  = ones(1,WormLength);             %

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Building Two different Boundary conditions.
```

```
% BoundaryCondLR creates a soft boundary condition for the neuronal
% activity in the form of a row vector that ranges between zero and one.
% It is primarily used to damp the activity of the Excitatory
% layers so they do not dwell near the edges of the simulation space. You
% can view any boundary condition using the plot function.

BoundaryCondLR= ones((WormLength),1);    %
for i = 1:LRspread
    BoundaryCondLR(i) = ((i-1)/LRspread).^1.3;
end
for i = (WormLength):-1: (WormLength-LRspread)
   BoundaryCondLR(i) = (-(i-(WormLength))/LRspread).^1.3;
end

% MechBoundaryCond tapers the display height of the segments of the worm at the
% head and tail.
MechBoundaryCond= ones((WormLength),1);  %
for i = 1:40
    MechBoundaryCond(i) = ((i-1)/40).^.4;
end
for i = (WormLength-Circumcision):-1: (WormLength-Circumcision-30)
    MechBoundaryCond(i) = (-(i-(WormLength-Circumcision))/30)^.4;
end
for i = (WormLength-Circumcision):1: (WormLength)
    MechBoundaryCond(i) = 0;
end

% MechBoundaryCond2 tapers the height of the segments of the worm at the
% head and tail, but the tail reaches all the way to the head. This is
% sometimes used to force ground contact dominance to the front of the
% worm. It applies to the HML2 matrix, and is not always used.
MechBoundaryCond2= ones((WormLength),1);  %
for i = 1:80
    MechBoundaryCond2(i) = ((i-1)/80).^.3;
end
for i = (WormLength-Circumcision):-1: (WormLength-Circumcision-30)
    MechBoundaryCond2(i) = (-(i-(WormLength-Circumcision))/30)^.6;
end
for i = (WormLength-Circumcision):1: (WormLength)
    MechBoundaryCond2(i) = 0;
end

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Building the Decremnet function to reduce the CPG influence at the tale

if CPGDecrement==1;

    for i = DecStart:(WormLength)
        Decrement(i) = DecMin + 1*((1.8*(i - WormLength * DecCenter))^4)/(1.4*(WormLength * DecCenter))^4;
    end
    Decrement = min(max(Decrement,DecMin),DecMax);
    Decrement = fliplr(Decrement);
%Decrement = Decrement*.8    ;
end

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Building a lookup table for the inverse kinematics:
% The LengthSolver function can be built directly into the model, but by
% creating a lookup table, the simulation is twice as fast.

if StaticModel==1

    KinRes=30;  % The number of data points along each axis of the lookup table.
    % The higher the number, the higher the resolution of the lookup table.

    % KinLookup: This is the lookup table for the dynamics solver. You can
    % view it using imshow(KinLookup).
    KinLookup = zeros(KinRes);

    for j=1:KinRes
        for k=1:KinRes

    KinLookup(j,k) =LengthSolver2((j-1)/KinRes, (k-1)/KinRes, 0.65, RestLength);
    % (HorizontalTension, VerticalTension, K, RestLength)
        end
    end

end

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% RUN THE SIMULATION:
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% This simulation uses Euler's Method. This was originally because of some
%% complex matrix manipulation between timesteps. This is no longer done,
%% so it may be possible to use ode45.
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%

for t = 1:dt:RunTime

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% If WieTestCycle is on, then vary the Wie weights accordingly.
% Many of these variations are currently turned off by setting the start
% condition to something outside the time bound of the simulation.

    if WieTestCycle ==1;
        if t>30
        WieRight = (WieCenter+WieMag*sin(t/WiePeriod));
        WieLeft = (WieCenter-WieMag*sin(t/WiePeriod/.8-pi/1.5));
        end
        step = 100;
        change = 2;
        delay = 700;%-350;
```

```
        if t-delay >step
            WieRight = 6;
            WieLeft  = .5;
        end
        if t -delay >step*2
            WieRight = 6;
            WieLeft  = 4;
        end
        if t -delay >step*3
            WieRight = 8;
            WieLeft  = 2;
        end

        if t -delay >step*4
            WieRight = 3;
            WieLeft  = 3;
        end
        if t -delay >step*5
            WieRight = 1;
            WieLeft  = 8;
        end
        if t -delay >step*6
            WieRight = .7;
            WieLeft  = 2.5;
        end
        if t -delay >step*7
                WieRight = (WieCenter-WieMag*sin(t/WiePeriod));
                WieLeft = (WieCenter+WieMag*sin(t/WiePeriod/.8-pi/1.5));
        end
        if (mod((t),50))>25&& t<1
            EHoop(1:Height, 130:150) =  zeros(Height, 21);
            IHoopRight(1:Height, 130:150) =  ones(Height, 21);
            ELong(1:Height, 130:150) =  ones(Height, 21);
        elseif t<1
            ELong(1:Height, 130:150) =  zeros(Height, 21);

            IRight(1:Height, 130:150) =  ones(Height, 21);
            EHoop(1:Height, 130:150) =  ones(Height, 21);
        end
    end


%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Calculate the change in the muscle positions

    LMLNormalized = LML-LMLOffset;
    %LMLNormalized = bsxfun(@times, LMLNormalized, BoundaryCondLR');

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Differential Equations! The Heart of the Simulation.

    % All of the state variables described below are in the Wilson-Cowan
    % Form:
    % Xprime * TimeConstant = -X + Sigmoid(Influences)
    % where the influences are layers of neuronal populations spatially
    % weighted by convolution kernels.

    %%ELong is shown in Green
    ELongPrime = (      -ELong +     Sigmoid(  alpha * (CPGStrength * (WeeLong * Decrement .*  conv2(ELong, Ehat, 'same')...
    - WieCross * conv2((IHoopRight+IHoopLeft)/2, EhatLarge, 'same') - WieLeft * conv2(ILeft, IhatLeft, 'same')     ...
    - WieRight * conv2(IRight, IhatRight, 'same')) + SenseStrength *(1-Decrement) .*(-WeLML* min(conv2((LMLNormalized), ...
    Ehat, 'same'),0) ) ) ) )/te;

    IRightPrime     = ( -IRight +    Sigmoid(  alpha * (CPGStrength * (Wei * Decrement .*   conv2(ELong, Ehat, 'same') ...
    - Wii * conv2(IRight, IhatCenter, 'same') ) + SenseStrength * (1-Decrement) .*(-WieLeft  ...
    * conv2(min((LMLNormalized),0), Ehat, 'same')  ) ) ))/ti/tau;

    ILeftPrime      = ( -ILeft  +    Sigmoid(  alpha * (CPGStrength * (Wei * Decrement .*   conv2(ELong, Ehat, 'same') ...
    - Wii * conv2(ILeft, IhatCenter, 'same') )  + SenseStrength * (1-Decrement) .*(-WieRight ...
    * conv2(min((LMLNormalized),0), Ehat, 'same') ) ) ))/ti/tau;

    %%EHoop is shown in Blue
    EHoopPrime = (      -EHoop +     Sigmoid(  alpha * (CPGStrength * (WeeHoop * Decrement .* conv2(EHoop, Ehat, 'same') ...
     - WieCross * conv2((IRight+ILeft)/2, EhatLarge, 'same')         - WieLeft * conv2(IHoopLeft, IhatLeft, 'same')...
     - WieRight * conv2(IHoopRight, IhatRight, 'same')) + SenseStrength * (1-Decrement) ...
    .*(WeLML* max(conv2((LMLNormalized), Ehat, 'same'),0) ) ) ) )/te;

    IHoopRightPrime = (-IHoopRight + Sigmoid(  alpha * (CPGStrength * (Wei * Decrement .*   conv2(EHoop, Ehat, 'same')...
    - Wii * conv2(IHoopRight, IhatCenter, 'same') ) + SenseStrength * (1-Decrement) .*( WieLeft ...
    * conv2(max((LMLNormalized),0), Ehat, 'same')  ) ) ))/ti/tau;

    IHoopLeftPrime  = (-IHoopLeft  + Sigmoid(  alpha * (CPGStrength * (Wei * Decrement .*   conv2(EHoop, Ehat, 'same') ...
    - Wii * conv2(IHoopLeft, IhatCenter, 'same') )  + SenseStrength * (1-Decrement) .*( WieRight ...
    * conv2(max((LMLNormalized),0), Ehat, 'same') ) ) ))/ti/tau;

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Apply LEFT-RIGHT BOUNDARY CONDITION

     if SoftEndBoundaryCondition

        ELong = bsxfun(@times, ELong, BoundaryCondLR');
        EHoop = bsxfun(@times, EHoop, BoundaryCondLR');

        if StaticModel==1  %!!!Not sure if this should be on!!!
            HML = bsxfun(@times, HML, BoundaryCondLR');
            LML = bsxfun(@times, LML, BoundaryCondLR');
        end
     end

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Integrate with or without noise
```

```
    if Noise
        NoiseVector=randn(Height,WormLength,6);

        ELong = ELong + ELongPrime*dt + sqrt(dt)*(NoiseVector(:,:,1))*ne;
        IRight = IRight + IRightPrime*dt + sqrt(dt)*(NoiseVector(:,:,2))*ni;
        ILeft = ILeft + ILeftPrime*dt + sqrt(dt)*(NoiseVector(:,:,3))*ni;

        EHoop = EHoop + EHoopPrime*dt + sqrt(dt)*(NoiseVector(:,:,4))*ne;
        IHoopRight = IHoopRight + IHoopRightPrime*dt + sqrt(dt)*(NoiseVector(:,:,5))*ni;
        IHoopLeft = IHoopLeft + IHoopLeftPrime*dt + sqrt(dt)*(NoiseVector(:,:,6))*ni;

    else
        ELong = ELong + EPrime *dt;
        IRight = IRight + IRightPrime*dt;
        ILeft = ILeft + IRightPrime*dt;

        EHoop = EHoop + EHoopprime*dt;
        IHoopRight = IHoopRight + IHoopRightPrime*dt ;
        IHoopLeft = IHoopLeft + IHoopLeftPrime*dt ;

    end

    if Clipped
        ELong = min(max(ELong,MinOutput),MaxOutput);
        IRight = min(max(IRight, MinOutput),MaxOutput);
        ILeft = min(max(ILeft, MinOutput),MaxOutput);
        EHoop = min(max(EHoop, MinOutput),MaxOutput);
        IHoopRight = min(max(IHoopRight, MinOutput),MaxOutput);
        IHoopLeft = min(max(IHoopLeft, MinOutput),MaxOutput);

        if StaticModel
            HML = min(max(HML,0),MaxOutput);
            LML = min(max(LML,0),MaxOutput);
        end
    end
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Calculate the position of the worm segments

    if StaticModel

        LMLold=LML; %store the previous time step
        HMLold=HML;

        mark=cputime;

        for i=2:(WormLength-1)

                kk= round(max((EHoop(i) )* KinRes,1));
                jj =round(max(ELong(i)   * KinRes,1));

            if SoilModel==1  && (150-i*3)<(t-901) % starts at 600, goes to 1050

                LML(i)=max(KinLookup(jj,kk),0.49);

            else
                LML(i)=KinLookup(jj,kk);
            end


        end
        LML = conv(LML,Ehat, 'same');
        HML = 2*(0.5^2-(LML/2).^2).^0.5;
        HML = HML.*MechBoundaryCond';
        HML2 = HML.*MechBoundaryCond2';
    end

%%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%%
%% Calculate the stretch between ground contact points.
%% This can be used for diagnostic purposes, or future sensory feedback.
%
%     if t>50
%
%     [PeakHeight(:,t) PeakHeightLocations(:,t)] = findpeaks(HML,'npeaks',2, 'minpeakdistance',30);
%
%
%     if (PeakHeightLocations(2,t)-PeakHeightLocations(1,t))||0
%
%         Stretch(t) = (PeakHeightLocations(2,t)-PeakHeightLocations(1,t));
%
%     end
%
%     else
%         Stretch(t) = 0;
%     end
%%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Store current frame:

    MovieStorage(:,:,1,t)=ELong;
    MovieStorage(:,:,2,t)=EHoop;
    MovieStorage(:,:,3,t)=IRight;
    MovieStorage(:,:,4,t)=IHoopRight;
    MovieStorage(:,:,5,t)=ILeft;
    MovieStorage(:,:,6,t)=IHoopLeft;

    if StaticModel
        MovieStorage(:,:,7,t)=HML2;
        MovieStorage(:,:,8,t)=LML;
        MovieStorage(:,:,9,t)=LMLNormalized;
        MovieStorage(:,:,10,t)=HML;
    end

end  % Start New Timestep!

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
```

```
%% These parameters are used to display the data properly:
[a,b,c,d]=size(MovieStorage);

e = round(d/ImageHeight);

% Pre-allocation:
NeuronalActivity = zeros((a*ceil(d/e)),(b*e+e*2),3);
WormPosImage     = zeros((a*ceil(d/e)),(b*e+e*2),3);

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% Parsing the data for visualization.
% Here, we take the data stored in MovieStorage, and redistribute it to two
% different images, NeuronalActivity and WormPosImage. By changing the
% third index of the MovieStorage matrix, you can display different data
% sets. By changing the third index of the two images, you can change the
% effective display color.

for vv = 1:e
    for uu = 1:ImageHeight;
        %The activity of different layers of neurons. Currently displaying
        %the Excitatory layers of both the hoop and lengthwise actuators.
        NeuronalActivity((uu*a):((uu+1)*a-1),((vv-1)*b+1+vv*2):(vv*b+vv*2),2) =  ...
        MovieStorage(:,:,1,uu+ImageHeight*(vv-1));% + MovieStorage2(:,:,7,uu+ImageHeight*(vv-1))/3;
        NeuronalActivity((uu*a):((uu+1)*a-1),((vv-1)*b+1+vv*2):(vv*b+vv*2),3) = ...
        MovieStorage(:,:,2,uu+ImageHeight*(vv-1));% + MovieStorage2(:,:,7,uu+ImageHeight*(vv-1))/3;

        % worm position image
        WormPosImage((uu*a):((uu+1)*a-1),((vv-1)*b+1+vv*2):(vv*b+vv*2),1) = MovieStorage(:,:,8,uu+ImageHeight*(vv-1));%
        WormPosImage((uu*a):((uu+1)*a-1),((vv-1)*b+1+vv*2):(vv*b+vv*2),2) = MovieStorage(:,:,10,uu+ImageHeight*(vv-1));%
        WormPosImage((uu*a):((uu+1)*a-1),((vv-1)*b+1+vv*2):(vv*b+vv*2),3) = MovieStorage(:,:,7,uu+ImageHeight*(vv-1));%
    end
end

figure(3333)
set(gcf, 'Units', 'pixels')
set(gcf, 'Position', [10,10,round(1300), 1000])
%If you are working on a small screen, you can change the image size here in pixels.

axes('Position',[0.02 0.01 .24 .98])
imshow(NeuronalActivity)

axes('Position',[0.28 0.01 .24 .98])
if SoilModel
    imshow(GroundContactData,[0,1])
end

axes('Position',[.53 .01 .24 .98])
imshow(WormPosImage(:,:,1),[0,1])

axes('Position',[.77 .01 .24 .98])
imshow(WormPosImage(:,:,3),[0,1])

%%~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~%%
%% If Animation is on, animate the results of the muscle signal
%% condition the worm position data

if Animation || MotionAnalysis

    if Animation
        aviobjI = avifile('WormANimation_1.avi','fps',30)  % Open a movie file.
    end
    % IMPORTANT, If you do not change the movie file name here, it will
    % write over your old movie. ALSO, these movies are huge (1-3GB!). If I like the
    % video and I want to keep it, I immediately use Quicktime PRO 7 to
    % compress the movie and delete the original.

    %Smooth the worm position data (temporally?):
    WormPosImage(:,:,1)= conv2(WormPosImage(:,:,1),[.03,.07,.1,.2,.2,.2,.1,.07,.03]', 'same');
    WormPosImage(:,:,3)= conv2(WormPosImage(:,:,3),[.03,.07,.1,.2,.2,.2,.1,.07,.03]', 'same');
    WormPosImage(:,:,2)= conv2(WormPosImage(:,:,2),[.03,.07,.1,.2,.2,.2,.1,.07,.03]', 'same');

    if Animation
        SegHandle=zeros(1,WormLength-Circumcision);

    figure(66)
    set(gcf, 'DoubleBuffer', 'on')
    set(gcf, 'Units', 'pixels')
    set(gcf, 'Position', [10,10,1400, 70])
    axes('Position',[0 0  1 1])
    set(gca, 'XTick', 0)
    set(gca, 'YTick', 66)
    axis([2,240,-.55,.55])
    hold on
    end

    CenterPointX = zeros(RunTime,1);

    [p,NewMaxWidthLocation] =max((WormPosImage(:,:,3)),[],2)


    for tt = 50:RunTime

        if tt==50
            TotalLengthNew= 0;%200-oldAverageLength/2;
            TotalLengthOld= 0;
            WormTail=0;
            WormMovement = 0;

            for seg = 1:(WormLength-Circumcision)

                if seg==1
                    TotalLengthNew = WormTail
                else

                    TotalLengthNew = TotalLengthNew+WormPosImage(tt,seg,1);
                end
```

```
                    if Animation
                    SegHandle(seg) = fill( [TotalLengthOld,TotalLengthOld,TotalLengthNew,TotalLengthNew],...
                        [(-WormPosImage(tt,seg,2)/2),(WormPosImage(tt,seg,2)/2),(WormPosImage(tt,seg,2)/2),...
                        (-WormPosImage(tt,seg,2)/2)],(NeuronalActivity(tt,seg+1,:)*.8+.1), 'LineSmoothing', 'on');
                    end
                if seg==NewMaxWidthLocation(tt)

                    GPoint1 = TotalLengthNew;
                    GPoint2 = TotalLengthNew;

                    % Draw the ground contact point
                    if Animation
                        SegHandle(seg) = fill( [TotalLengthOld,TotalLengthOld,TotalLengthNew,TotalLengthNew],...
                        [(-WormPosImage(tt,seg,2)/2),(WormPosImage(tt,seg,2)/2),(WormPosImage(tt,seg,2)/2),...
                        (-WormPosImage(tt,seg,2)/2)],[1,0,0], 'LineSmoothing', 'on');
                    end
                end
                if seg== round(WormLength/2)
                    CenterPointX(tt)=TotalLengthNew;
                end
            end
        else

           % GPoint3 = GPoint2;
            GPoint2 = GPoint1;


            WormTail = WormTail - WormMovement;
            TotalLengthOld = WormTail;

            for seg = 1:(WormLength-Circumcision)

                if seg==1
                    TotalLengthNew = WormTail;
                else
                    TotalLengthOld = TotalLengthNew;
                    TotalLengthNew = TotalLengthNew+WormPosImage(tt,seg,1);
                end

                if Animation
                    set( SegHandle(seg), 'XData',[TotalLengthOld,TotalLengthOld,TotalLengthNew,TotalLengthNew],...
                         'YData',[(-WormPosImage(tt,seg,2)/2),(WormPosImage(tt,seg,2)/2),(WormPosImage(tt,seg+1,2)/2),...
                         (-WormPosImage(tt,seg+1,2)/2)],...
                         'FaceColor',(NeuronalActivity(tt,seg+1,:)*.8+.1));
                end

                if seg==NewMaxWidthLocation(tt)

                    GPoint1 = TotalLengthNew;
                    if Animation
                        set( SegHandle(seg), 'XData',[TotalLengthOld,TotalLengthOld,TotalLengthNew,TotalLengthNew],...
                            'YData',[(-WormPosImage(tt,seg,2)/2),(WormPosImage(tt,seg,2)/2),(WormPosImage(tt,seg+1,2)/2),...
                            (-WormPosImage(tt,seg+1,2)/2)],...
                            'FaceColor',[1,0,0]);
                    end
                end

                if seg==NewMaxWidthLocation(tt-1)

                    WormMovement = (WormMovement + (TotalLengthNew - GPoint2));
                end

                if seg== round(WormLength/2)
                    CenterPointX(tt) = TotalLengthNew;
                end

            end
                % oldAverageLength = TotalLengthNew-200+oldAverageLength/2;
        end

        if Animation
        drawnow

        frame = getframe(66);
        aviobjI = addframe(aviobjI,frame);
        end
    end
    if Animation
        aviobjI = close(aviobjI)
    end
end

if MotionAnalysis
    figure(44)
    plot(CenterPointX, 'linewidth',2)
    axis([50,RunTime,0,300])
end



~~~~~~~~~~~~~~~~~~
% Sigmoid function
% Alexander Boxerbaum
% December 1, 2008

%% This function returns a value based on a sigmoid Function. It is
%% normalized to change mostly between the domain of zero and one,
%% and maps onto the range of -0.1 to 1. The range limits have important
%% consequences for the dynamics.

function out = Sigmoid(x)
%    out=1./(1+exp(-x));

out = (1./(1+exp(-(x*7-3))) - 1/(1+exp(3))) * 1.05;

%out = max(out,0);
```

```
end


~~~~~~~~~~~~~~~~~~

% Alexander Boxerbaum
% February, 2012

% longFlatOffsetHat(Spread, MaxRadius,Offset):
% This is the function that
% builds the gaussain convolution matricies that define the
% interconnectedness of the neuronal populations.

function out = longFlatOffsetHat(se, r, shift)

%% Lets make a flexible distribution of influence for a given neuron
%

mm = 2*r + 1 ;%+ abs(shift);

% [m,n] = size(X);
out = zeros(1,mm);

for j = 1:mm

   out(j) =  exp( -((j-r-1-shift)^2)/se );

end

out = out/sum(sum(out));  % Normalize
~~~~~~~~~~~~~~~~~~

% Alexander Boxerbaum, February 2012.
% editted Kati Daltorio Feb 2012

% LengthSolver(HorizontalCompression, VerticalCompression, K, RestLength):

% This function returns the segment length based on the forces applied to
% it and its natural spring properties. It is derived from the kinematics
% of a single braided mesh element with a spring connecting the horizontal
% nodes. It is a very slow function (because it needs to find zeros of
% nasty equations), so the above code uses the function to build a lookup table
% instead of constantly calling it.

%Kati:
%I rederived equation from virtual work equations


function out = LengthSolver2(Fl, Fc, A, L) % (HorizontalTension, VerticalTension, K, RestLength)

out=fzero(@(X) ForceSum(Fl, Fc, A, L, X),[.01 (L*2-.06)]); % fzero is a matlab solver that finds the zeros of the function ForceSum
defined below.


function FS= ForceSum(Fl, Fc, A, L, X)
a = 2*sqrt(1.5^2-(.5)^2); %ratio of resting height to resting length
FS= tan((X-L)*(pi/2) /L)*A ... %part from the spring
    + Fl ...%X-direction forces that compress spring
    - Fc * (L+X)/... h where h = ...
           sqrt( (1+a^2)*L^2 - (X+L)^2 ); % but now Fc better be compressive on element, extending spring if negative like this
    %- Fc*tan(asin(X/(2*L)));  % Alex's term
```

BIBLIOGRAPHY

Alexander, R., M. (2003). Principles of Animal Locomotion. Princeton New Jersey. Princeton University Press. p. 88.

Aristotle, Nicomachean Ethics, translated by Martin Ostwald. Library of Liberal Arts, Prentice Hall, 1962.

Barnett, C. Bengough, A. G., McKenzie, B. M. Quantitative image analysis of earthworm-mediated soil displacement. Biol Fertil Soils (2009) 45:821–828.

Boxerbaum, A. S., Chiel, H. J. and Quinn, R. D. (2009). Softworm: A Soft, Biologically Inspired Worm-Like Robot. Society for Neuroscience Annual Meeting, Chicago Il.

Boxerbaum, A. S., Chiel, H. J. and Quinn, R. D. (2010). A New Theory and Methods for Creating Peristaltic Motion in a Robotic Platform. International Conference on Robotics and Automation (ICRA), 1221-1227.

A. S. Boxerbaum, H. J. Chiel, H. J., and R. D. Quinn, "Continuous Wave Peristaltic Locomotion" International Journal of Robotics Research, January 2012.

Boxerbaum, A. S., Horchler, A. D., Shaw, K,. Chiel, H. J., and Quinn, R. D. (2011). A Controller for Continuous Wave Peristaltic Locomotion. In *Proceeding of the International Conference on Intelligent Robots and Systems (IROS).*

Boyle, J., Berri, S., Cohen, N., Gait Modulation in *C. Elegans*: An Integrated Neuromechanical Model., Frontiers in Computational Neuroscience, March 2012.

Brusca, R. C. and Brusca, G. J. (1990). Invertebrates, Sinauer Associates, Sunderland, MA.

Creutz, E., (2005). "Feynman's Reverse Sprinkler". *American Journal of Physics* **73** (3): 198

Cheng, N., Ishigami, G., Hawthorne, S., Chen, H., Hansen, M., Telleria, M., Playter, R., Iagnemma, K. (2010). Design and Analysis of a Soft Mobile Robot Composed of Multiple Thermally Activated Joints Driven by a Single Actuator. In *Proceeding of the International Conference on Robotics and Automation (ICRA)*.

Collier, H. "Central nervious activity in the earthworm" Journal of Experimental Biology, 1939.

Dario, P., Ciarletta, P., Menciassi, A. and Kim, B. (2004). Modeling and experimental validation of the locomotion of endoscopic robots in the colon. *International Journal of Robotics Research*, 23(4-5), pp. 549-556.

Dollar, A., Wagner, C., Howe, R., (2006). Embedded Sensors for Biomimetic Robotics via Shape Deposition Manufacturing. Int. Conf. Biorobotics and Biomechatronics (Biorob), Pisa, Italy.

Ekeberg, Ö. and Griller, S. (1999). Simulations of neuromuscular control in lamprey swimming. *Philos. Trans. R. Soc. Lond. B. Biol. Sci.* 354:895–902.

Gardner, C. R. (1975). The neuronal Control of Locomotion in the Earthworm. Biol. Rev, 51, pp. 25-52.

Gray, J and Lissmann, W, "Studies in Animal Locomotion VII: Locomotory reflexes in the Earthworm) Journal of Experimental Biology, 15: 506-17

Hirose, S. (1993). Biologically Inspired Robots: Snake-Like Locomotors and Manipulators. Oxford, Oxford University Press.

Ijspeert, A. J., Crespi, A., Ryczko, D. and Cabelguen, J. M. (2007). From swimming to walking with a salamander robot driven by a spinal cord model. Science, 315(5817):1416–1420.

Ingram, M. and Hong, D. (2005). Whole Skin Locomotion Mechanism Inspired by Amoeboid Motility Mechanisms, *IDETC/CIE*.

Kier, W. M. and Smith K. K. (1985). Tongues, tentacles and trunks: the biomechanics of movement in muscular-hydrostats. Zool J Linn Soc 83:307–324.

Lee, E., Yang, J., Jolda, M., Wood, R. (2010). Cellular Slime Mold Robot. Workshop on "Modular Robots: The State of the Art." International Conference on Robotics and Automation (ICRA).

Mangan, E. V., Kingsley, D. A., Quinn, R. D. and Chiel, H. J. (2002) Development of a peristaltic endoscope. International Congress on Robotics and Automation (ICRA), 347–352.

Menciassi, A., Gorini, S., Pernorio, G., and Dario, P. (2004). A SMA Actuated Artificial Earthworm. International Congress on Robotics and Automation (ICRA).

Moore, A. R. "Muscle tension and reflexes in earthworm" 1923, Journal of General Physiology, 5, 327

Omori, H., Nakamura, T., Yada, T. (2009). An underground explorer robot based on peristaltic crawling of earthworms. Industrial Robot: An International Journal, Vol. 36 Iss: 4, pp.358 – 364.

Pearson, Keir, G., (1995). Proprioceptive Regulation of Locomotion. Current Opinion in Neurobiology, 5:786-791.

Quillin, K. J. (1999). Kinematic scaling of locomotion by hydrostatic animals: ontogeny of peristaltic crawiling by the earthworm lumbricusterrestris. The Journal of Experimental Biology, 202:661–674.

Quillin, K. J. (2000). Ontogenetic Scaling of Burrowing Forces in the Earthworm Lumbricus Terrestris. The Journal of Experimental Biology 203, 2757–2770.

Quinn, R. D., Nelson, G. M., Ritzmann, R. E., Bachmann, R. J., Kingsley, D. A., Offi, J. T. and Allen, T. J. (2003). Parallel Strategies for Implementing Biological Principles Into Mobile Robots. Int. Journal of Robotics Research, 22(3):169–186.

Rabinovich, M., Huerta, R., Varona, P., Afraimovich, V. (2008) Transient Cognitive Dynamics, Metastability, and Decision Making. Computational Biology, Volume 4 Issue 5.

Seok, S., Onal, C. D., Wood, R., Rus, D. and Sangbae Kim, S. (2010). Peristaltic Locomotion With Antagonistic Actuators in Soft Robotics. International Conference on Robotics and Automation (ICRA).

Skierczynski, B. A., Wilson, R. J. A., Kristan, W. B., and Skalak, R. (1996). A model of the hydrostatic skeleton of the leech. Journal of Theoretical Biology 181:329–342.

Trimmer, B., Takesian, A. and Sweet, B. (2006). Caterpillar locomotion: a new model for soft-bodied climbing and burrowing robots. 7th International Symposium on Technology and the Mine Problem, Monterey, CA.

Tesch, M., Lipkin, K., Brown, I., Hatton, R., Peck, A., Rembisz, J., Choset, H. (2009). Parameterized and Scripted Gaits for Modular Snake Robots. Advanced Robotics, Vol 23, No. 9, 2009 , pp. 1131-1158(28).

Transeth A. A., Pettersenand K. Y., Liljeback P.. (2009). AA survey on snake robot modeling and locomotion. Robotica volume 27, pp. 999–1015.

Trivedi, D., Rahn, C. D., Kier, W. M. and Walker, I. D. (2003). Soft robotics: Biological inspiration, state of the art, and future research. Applied Bionics and Biomechanics, 5(3):99–117.

Vaidyanathan, R., Chiel, H.J. and Quinn, R.D. (2000). A hydrostatic robot for marine applications. *Robotics and Autonomous Systems*, 30:103-113.

Wang, K. and Yan, G. (2007). Micro robot prototype for colonoscopy and in vitro experiments. Journal of Medical Engineering & Technology, 31(1), pp. 24-28.

Zarrouk, D., Sharf, I. and Shoham, M. (2010). Analysis of Earthworm-like Robotic Locomotion on Compliant Surfaces. International Conference on Robotics and Automation (ICRA).

Zimmermann, K. and Zeidis, I. (2007), Worm-Like Motion as a Problem of Nonlinear Dynamics, Journal of Theoretical and Applied Mechanics, 45(1):179–187.