University of Nevada, Reno

Biomorphic Hyper-Redundant Snake Robot: Design, Prototyping, and Locomotion Performance Demonstration

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science in
Electrical Engineering

by

Alexandr Bajenov

Dr. Yantao Shen / Thesis Advisor

December, 2018

THE GRADUATE SCHOOL

We recommend that the thesis
prepared under our supervision by

ALEXANDR BAJENOV

Entitled

Biomorphic Hyper-Redundant Snake Robot:
Design, Prototyping, And Locomotion Performance Demonstration

be accepted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

Yantao Shen, Ph.D., Advisor

M. Sami Fadali, Ph.D., Committee Member

Kostas Alexis, Ph.D., Graduate School Representative

David W. Zeh, Ph.D., Dean, Graduate School

December, 2018

Abstract

This thesis focuses on the design, construction, and control of a robotic snake. Relevant theoretical and practical aspects of the mechanical, electrical, and algorithmic design of the robot are described. Mechanically, the robot is a modular multi-segment mechanism, in which each segment is a 2-DOF (two degrees-of-freedom) universal joint driven by two motors. Most of the structural and mechancal parts were designed in Solidworks and 3D-printed, while others were standard off-the-shelf components such as screws and bearings. The motors are RC servo motors. The angular displacement command for each servo motor is calculated by a central pattern generator from a vector of motion parameters. These parameters are transmitted wirelessly from a handheld device to the robot. By varying the motion parameters, the robot is capable of several modes of locomotion, including slithering forward or in reverse, steering to either side, rolling sideways, or side-winding sideways.

Acknowledgements

I would like to thank Dr. Yantao Shen for his patience as the necessity for redesign became apparent and as the project moved from one prototype to another. I would like to thank Weixin Yang, a fellow student on this project, for all the hard work on the simulation, and the sensors-and-feedback system side of the project. I would like to thank Yudong Luo for going out of his way to assist us with our project, Na Zhao for helping us improve the structural design, and all the other students in the lab who also helped with the project.

I would also like to thank Anthony Piazza for helping us with the practical matters involved in constructing the prototypes whenever we lacked the necessary tools or components. I would like to thank Dr. Tod Colegrove, head of the DeLaMare Science & Engineering Library, for working hard to bring rapid prototyping technology to campus and making it widely available for all students. I would also like to thank the people at the Innevation [sic] Center for making their workshops and rapid prototyping technology available to the Reno-Sparks area.

Last but not least, I would like to thank the examination committee members Dr. Kostas Alexis, Dr. Sami Fadali, and Dr. Yantao Shen for participating in the examination of my thesis.

Table of Contents

Table of Tables

Table of Figures

Chapter 1

Introduction

This thesis covers the robot snake that was built as a group project, including the advantages of snake-like locomotion, the theory behind its operation, and how the robot was designed and implemented. This chapter describes the strategic aspects of the research, including the motivation for undertaking this project, the objectives we intended to meet, and my practical contributions to this project.

1.1 Objectives

The objective of this project is to build a biomimetic robotic snake capable of demonstrating the versatility of snake-like locomotion and traversing different surfaces. The project started with simple, flat, predictable surfaces, such as the table-tops and the laboratory floor. After the capability for basic locomotion is demonstrated, additional gaits are developed to improve the robot's capabilities on a wider variety of surfaces. This is accomplished in part by hyper-redundancy – the intentional use of more degrees of freedom than deemed sufficient by a more traditional, wheeled-platform approach to robot locomotion. In other words, traditional wheeled platforms have two degrees of freedom – one for each wheel – whereas the robot snake has two degrees of freedom in every one of its seven or eight motorized X-Y joints, for a total of fourteen or sixteen degrees of freedom. Because the robot's motion as a whole is affected by each degree of freedom independently, each additional degree of freedom increases the combination-space of possible locomotion solutions by an order of magnitude. This makes the exploration of solution space challenging, because a smaller percentage of solutions is feasible; however, the total range of feasible solutions is also greater.

1.2 Motivation

The traditional solution to robot locomotion - wheeled robots – is admittedly efficient but lacks versatility. A thickly carpeted floor is usually enough to immobilize small motorized toys. On a larger scale, even ostensibly all-terrain rovers are vulnerable to becoming bogged down by difficult terrain [13] despite presumably significant engineering efforts and expenditures to prevent that very problem. In such systems, the fixed-radius wheel becomes the weak link in the chain of events that need to happen in order for the robot's locomotion to occur: if the robot finds itself propped up on an obstacle in such a position that its wheels in are unable to exert sufficient traction to overcome the gravitational potential energy of its immediate situation, then the robot is effectively rendered immobile and trapped in place, as with NASA's Spirit Rover, a replica of which is shown on Figure 1.2.1.

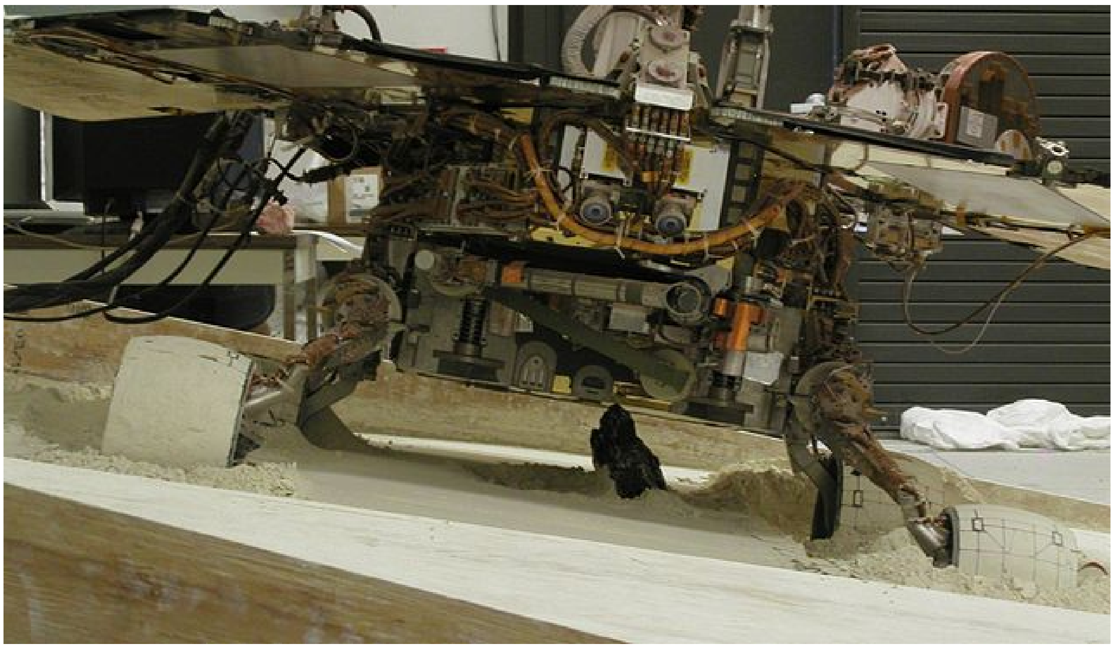


Figure 1.2.1: Laboratory attempt to replicate conditions of the accidentally immobilized Spirit Rover. Courtesy NASA/JPL-Caltech [36].

A snake-like robot, although unlikely to ever become as efficient as a wheeled vehicle, is expected to make up for it with versatility inherent to its hyper-redundant structure. Such a platform, aided by a much greater number of degrees of freedom, offers a much broader range of solutions to locomotion problems. This makes hyper-redundant robots a less risky, more reliable alternative to wheeled platforms for deployment in unpredictable environments.

A snake-like robot also benefits from biomimetics – the emulation of solutions observed in nature. In general, biomimetics offers the use of solutions developed by many thousands of years of nature's trial and error – evolution. For snake-like robots, specifically, biomimetics makes a number of locomotion solutions available from observation of live snakes. Lateral undulation and sidewinding (please see Section 3.1 for more detail) are some of the more prominent examples of effective locomotion solutions based on biomimetics.

Yet another benefit inherent (although not exclusive) to a snake-like robot is the absence of continuously-rotating external parts. Each motorized X-Y joint is limited to approximately 40° in each direction. This is not nearly enough for a wheeled platform, but it is enough for a motorized joint in a snake-like robot with many such joints. This lack of continuous rotation allows for much simpler dust- and water-resistance techniques, such as enclosing the robot in a watertight sleeve, which cannot be accommodated by a robot with continuously-rotating joints. Although such techniques have not yet been implemented in this project, they remain a feasible option.

1.3 Contribution

My contribution to the project consists primarily of hardware design to improve the physical and computational capabilities of the robot snake previously designed by other participants, by re-designing the robot's gear train and designing its central electronic hardware, respectively.

I redesigned the gear train, with considerations to desired torque and range of movement, the specifications of previously selected and purchased servo motors, as well as manufacturing constraints imposed by the specifications of Shapeways' *Raw Brass* material and geometric constraints imposed by previously designed and manufactured structural parts.

The robot's central electronic hardware has been designed several times. The first was designed by connecting its 'blocks' – simple breakout boards which were bought online. The second design was an upgrade, primarily to the robot's computational power, replacing the microcontroller with a more modern, more powerful one. Combining all parts of the control circuit on a single PCB eliminated bulky connectors, freeing up space inside the enclosure, thereby enabling future additions of advanced on-board sensors. The third design was an upgrade to its manufacturability; a need which became apparent when the second design was assembled by means of reflow soldering.

My other contributions include the design of the '*Skull*' enclosure, simple electrical interconnection hardware, as well as firmware algorithms that are used to demonstrate the basic capabilities of the robot snake's hardware before it can be adapted for use in more algorithmically advanced projects.

Chapter 2

Background

This chapter provides a brief background on snakes and snake-shaped robots. The biomechanical structure of natural snakes is briefly covered, which is relevant because we tried to immitate it to some extent with our robot. A summary of selected examples of prior academic work in the field of snake-like robots is also included to compare various parameters of our robot snake to prior accomplishments.

2.1 Snakes

Although snakes lack the efficiency of wheeled platforms on perfectly flat terrain, they are highly versatile and very adaptive to different real-world environments due to their many degrees of biomechanical freedom. Even though many of their gaits are presumably instinctive, they can, in theory, overcome previously unknown locomotion challenges on difficult terrain. Tree snakes, for example, are capable of traversing complex vertical environments, as shown on Figure 2.1.1.

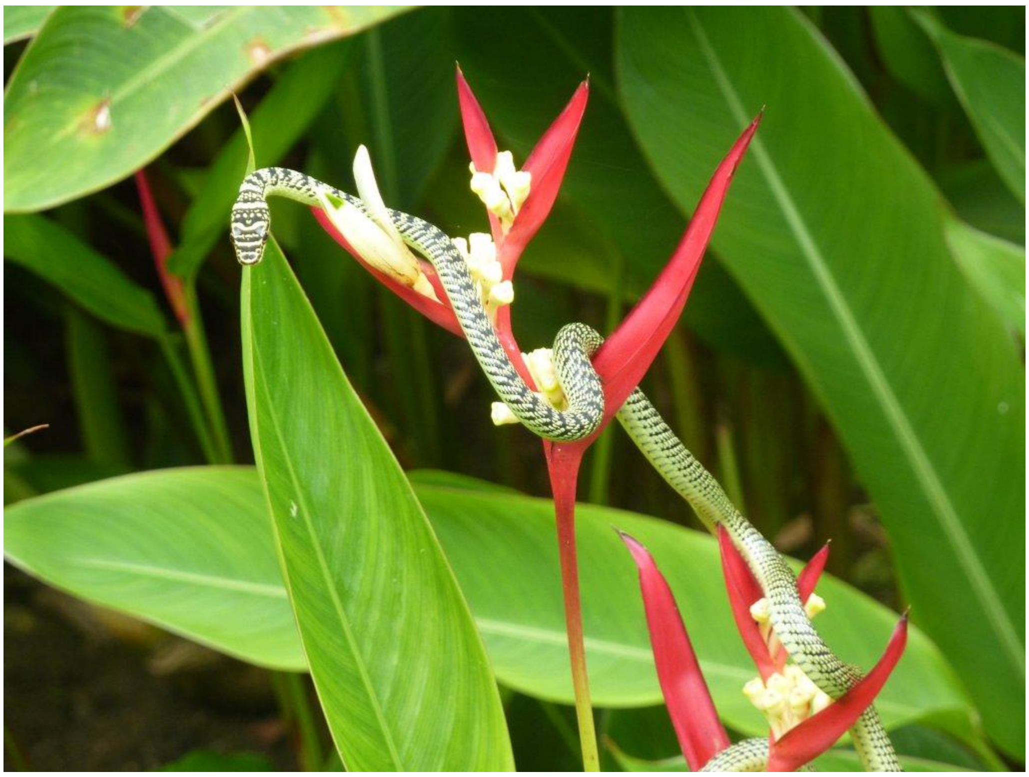


Figure 2.1.1: A small tree snake perched on top of a plant [37].

The hyper-redundancy of a snake's body is characterized by the smooth, and apparently continuous curvature along the length of its body. In reality, however, even live snakes have a skeleton consisting of effectively rigid segments. However, the vertebrae of a snake's skeleton (see Figure 2.1.2) are short and numerous enough to allow for very smooth curvature.

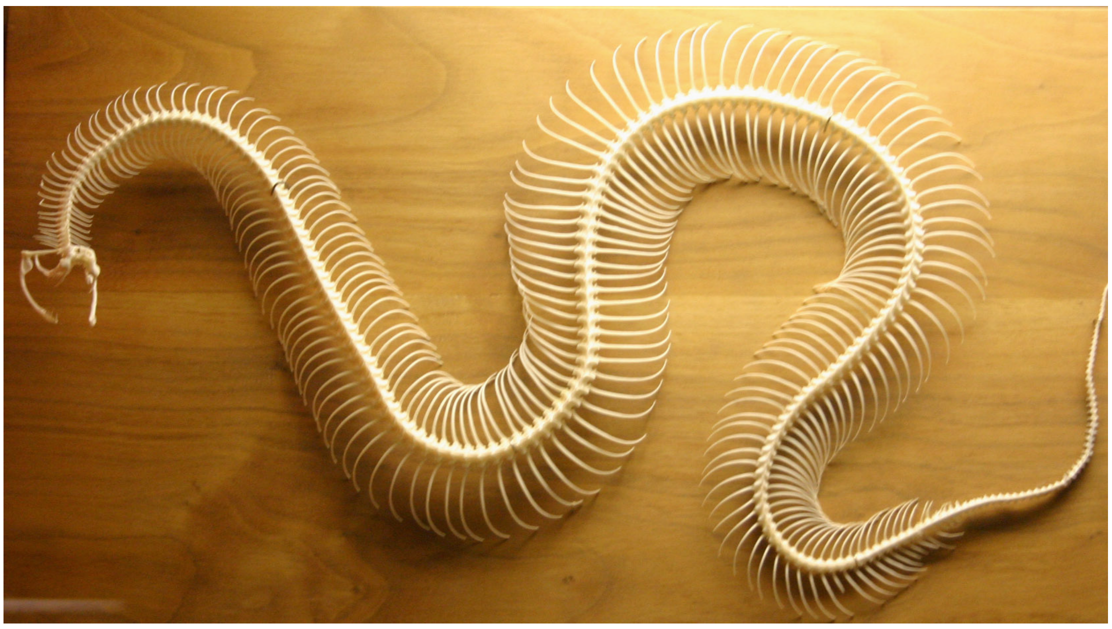


Figure 2.1.2: Snake skeleton [40].

2.2 Robot Snakes

Contrary to robot snakes, mobile robots of the simpler, more traditional schemes of locomotion drive around on wheels or tracks. They are designed around the assumption that they would traverse rigid, approximately horizontal surfaces. While this assumption may hold true on floors and paved roads, it is often false for natural terrain. The advantages of snake-like locomotion over wheeled locomotion have been apparent for a long time. Consumer, industrial, and military robot design still favors wheels or tracks for their efficiency on predictable surfaces, but there have been efforts by academic institutions to design and build snake-like robots.

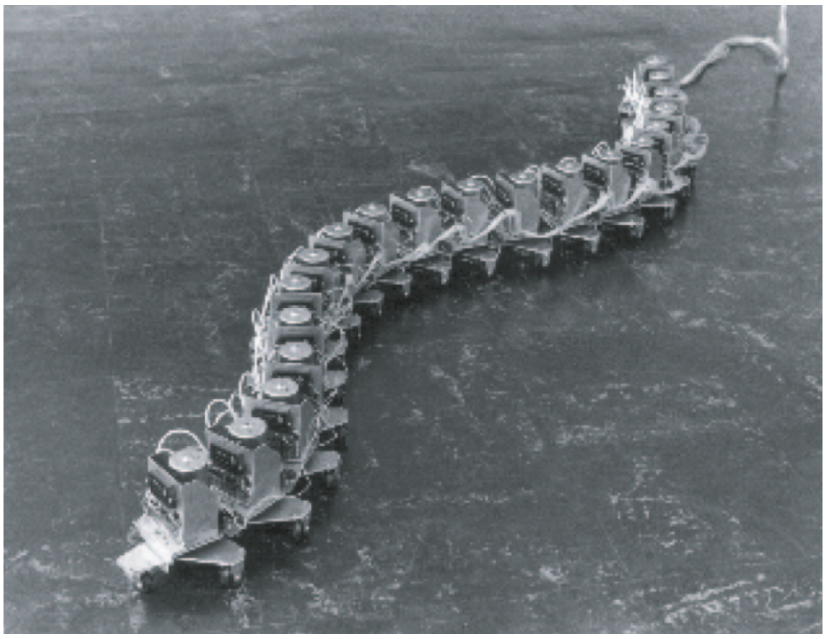


Figure 2.2.1: The world's first snake robot, ACM-III.
Courtesy of Tokyo Institute of Technology.

The world's first snake robot, "ACM-III," (see Figure 2.2.1) which stands for "Active Cord Mechanism," was developed by Professor Shigeo Hirose of the Tokyo Institute of Technology in 1972 [1][15][16]. ACM-III was arranged in 20 motorized segments that generated motion along a flat plane with 20 degrees of freedom, and demonstrated serpentine locomotion by means of two-dimensional lateral undulation. This differs from our robot snake, which is capable of three-dimensional motion; however, our robot snake has only sixteen degrees of freedom – only eight in each direction. Professor Hirose later upgraded it to a self-contained version of this robot, named ACM-R1 in 1995 [18], and went on to create more snake-like robots that explored the potential of snake-like mobility.

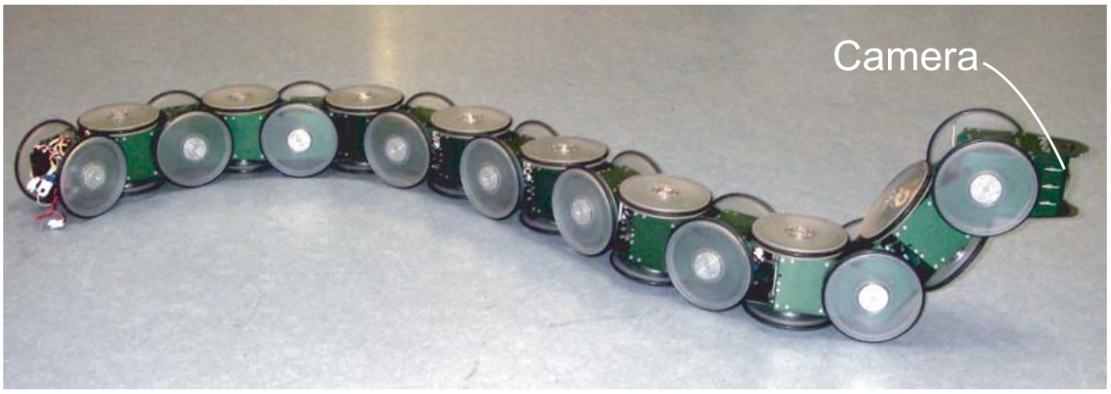


Figure 2.2.2: ACM-R3n. Courtesy of Tokyo Institute of Technology.

ACM-R2 [18], ACM-R3 [17], and ACM-R3n [19] (see Figure 2.2.2,) developed at the Tokyo Institute of Technology in 2000, 2001, and 2002, respectively, are snake-like robots capable of three-dimensional (3D) motion. Like the ACM-III, the ACM-R3 and ACM-R3n consist of 20 joints. Unlike the ACM-III, the ACM-R3's joints are offset at right angles from neighboring joints, which alternate their direction of motion between horizontal and vertical. The ACM-R3 and ACM-R3n are brought into motion by RC servo motors, which is a trait our robot has in common with these. The ACM-R3n also features an on-board camera. Although our robot shares some similarity with the ACM-R3 and

ACM-R3n in its (admittedly limited) ability for 3D motion, our robot's structure is arranged as a chain of motorized 2-DOF universal joints rather than a chain of alternating horizontal and vertical 1-DOF hinges. The structure of ACM-R3 and ACM-R3n has the advantage of a much simpler mechanical design as compared to ours. It is also worth pointing out that, for a robot of comparable size (the ACM-R3 is 1.8 m in length, whereas ours is 1.3 m in length,) the ACM-R3 managed to achieve a significantly higher output torque of 15 N·m, whereas ours achieves an output torque of no more than 0.631 N·m – a difference of over an order of magnitude.

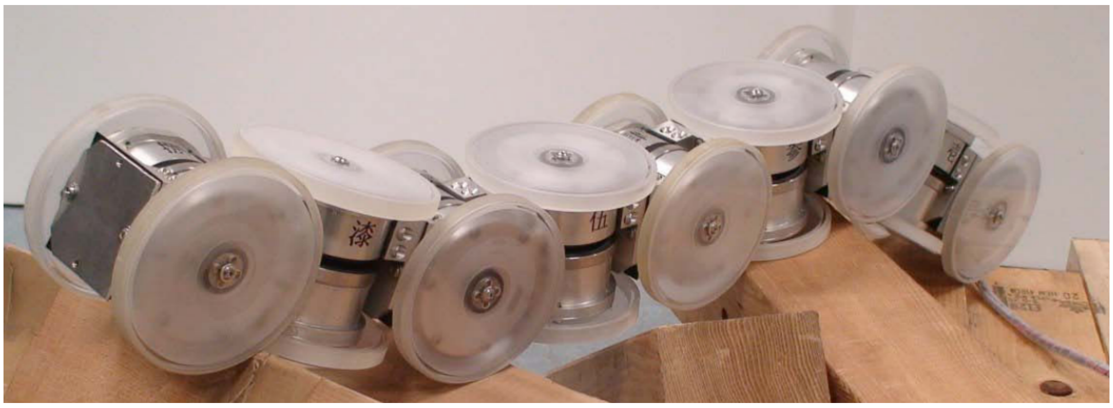


Figure 2.2.3: ACM-R4.1. Courtesy of Tokyo Institute of Technology.

ACM-R4 (and ACM-R4.1, see Figure 2.2.3) [20][21] developed at the Tokyo Institute of Technology *c*. 2006, has a close visual resemblance to the ACM-R3 and ACM-R3n, with the only visible difference being its use of metal structural components. Internally, the ACM-R4 differs from the ACM-R3 in having torque sensors and torque limiters, as well as active wheels that are arranged coaxially with, yet are driven independently from, the robot's motorized joints.

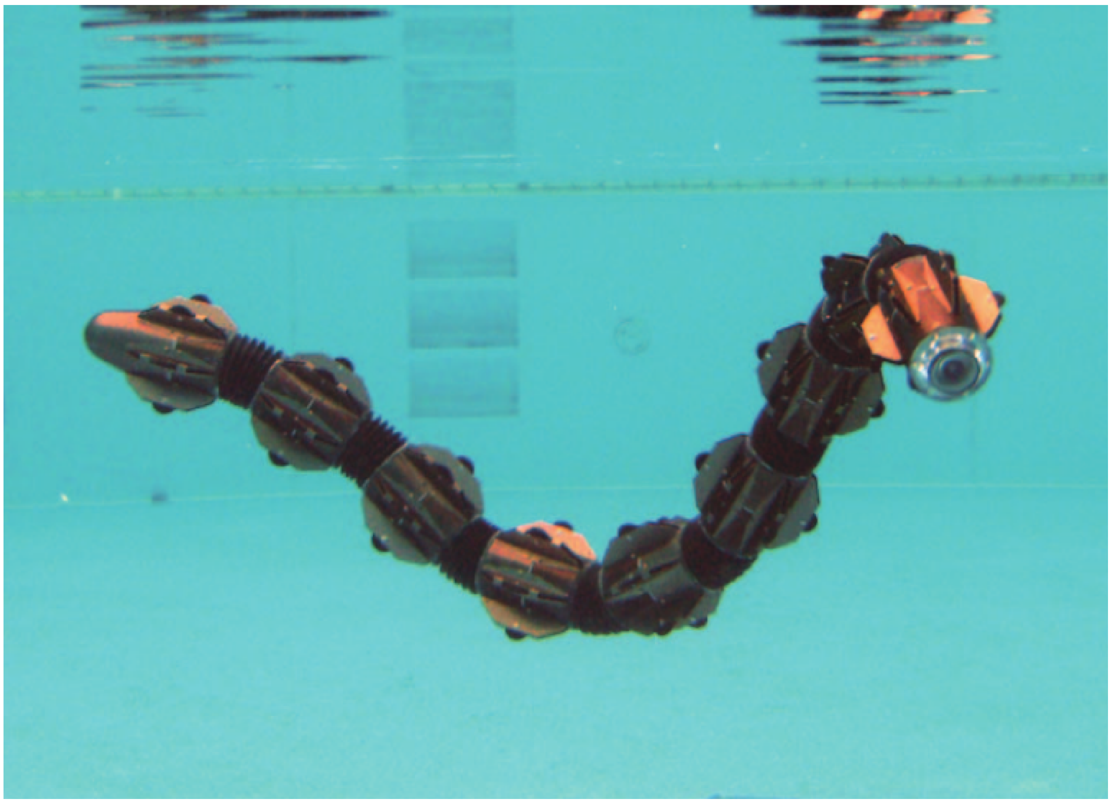


Figure 2.2.4: ACM-R5. Courtesy of Tokyo Institute of Technology.

HELIX [26] and ACM-R5 [27][31] (see Figure 2.2.4) were built *c*. 2000 and *c*. 2005, respectively, at the Tokyo Institute of Technology, to demonstrate snake-like locomotion underwater. A prominent design feature of these robots was their sealed water- and dust-proof structures, which made the complexity of a universal-joint-type of motorized joints a worthwhile investment. Because two degrees of freedom now shared one "bellows"-style waterproofing joint, this design choice halved the amount of waterproofing work. Although our robot does not share the environment-proofing features of the HELIX and ACM-R5, it does share their structure of rigid segments connected by motorized universal joints.

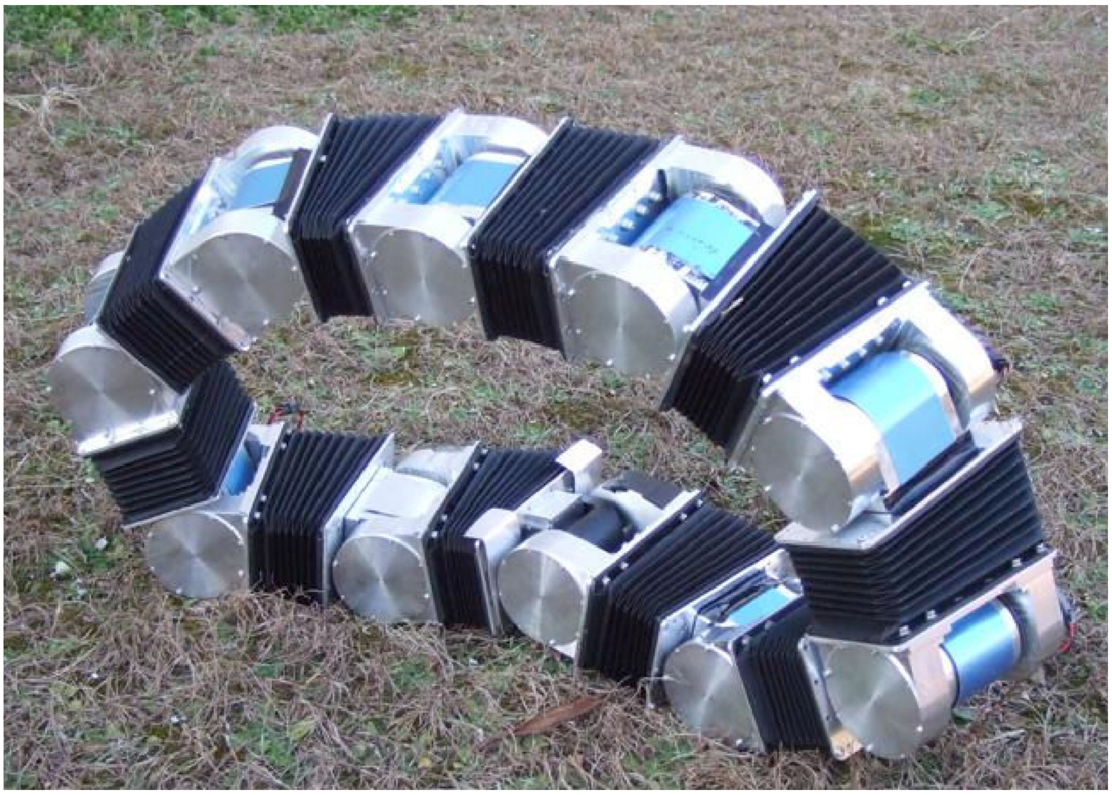


Figure 2.2.5: ACM-R7, with its body formed into a loop.
Courtesy of Tokyo Institute of Technology.

The ACM-R7 [28] (see Figure 2.2.5) was a robot snake built *c.* 2010 at the Tokyo Institute of Technology to explore a unique means of locomotion. For lack of better words, the ACM-R7 bites its own tail to form its body into a closed loop, turning into something like a wheel or a caterpillar tread that can drive around by itself. The motion in each direction depends on the common and differential motion, respectively, of each joint's two DC motors. The ACM-R7 exhibits substantially different torque and angular range in the vertical ("Pitch") and horizontal ("Yaw") motion of its joints. Although our robot snake does not offer the power-to-weight ratio necessary to demonstrate a similar mode of locomotion, the dinstinctive specialization between torque and angular velocity parameters of horizontal joints and vertical joints is useful for snake robots that have a top side and a bottom side for the purpose of contact with terrain. For instance, vertically-motorized joints benefit from a greater available torque to lift the front, middle, or rear of the robot off the ground to traverse gaps; whereas the horizontally motorized joints require less torque because they only need to overcome the forces of friction rather than the force of gravity.

The ACM-L2 [32] developed at the Tokyo Institute of Technology was another approach to snake-like robot locomotion that made use of vertically moving motorized joints, namely the *Pedal Wave*. This mode of locomotion is more akin to that of a caterpillar rather than that of a snake, but is suitable for traversing narrow spaces. Unfortunately this mode of locomotion requires shorter and more numerous segments than we have on our robot snake.

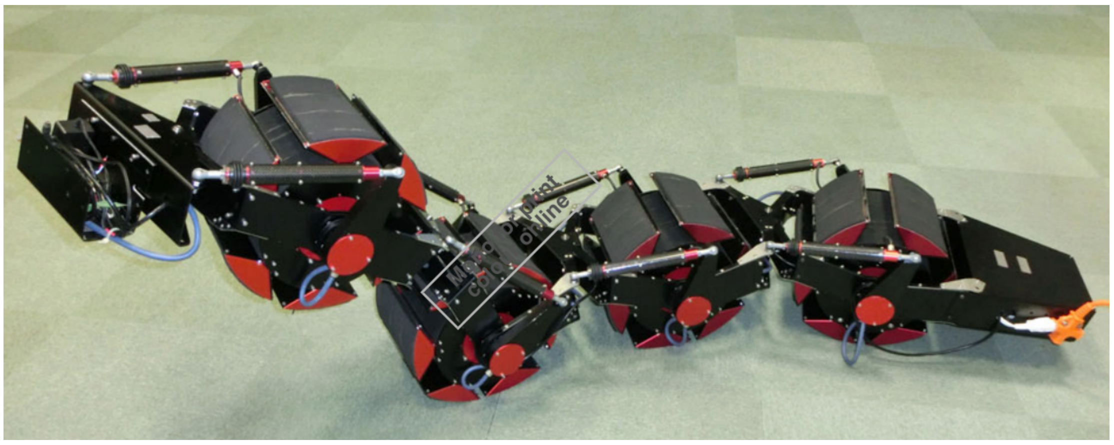


Figure 2.2.6: ACM-R8. Courtesy of Tokyo Institute of Technology.

The ACM-R8 [29] was built *c*. 2014 at the Tokyo Institute of Technology. Like the ACM-R4, the ACM-R8 combines the hyper-redundancy of a snake-like body with the flat-terrain efficiency of wheels. The wheels on the ACM-R8 are even more prominent than on the ACM-R4. Unlike the ACM-R4, the wheels on the ACM-R8 do not alternate between vertical and horizontal. Instead, the robot is designed that its wheels usually maintain a vartical orientation. Like the ACM-R7, the ACM-R8 has motorized joints brought into motion by the common and differential action of two independent actuators. However, these actuators are harmonic drives, namely the CSD-25 from Harmonic Drive System Corp., rather than the ubiquitous DC motors. Force sensors form the dual linkage between the moving parts of each motorized segment.

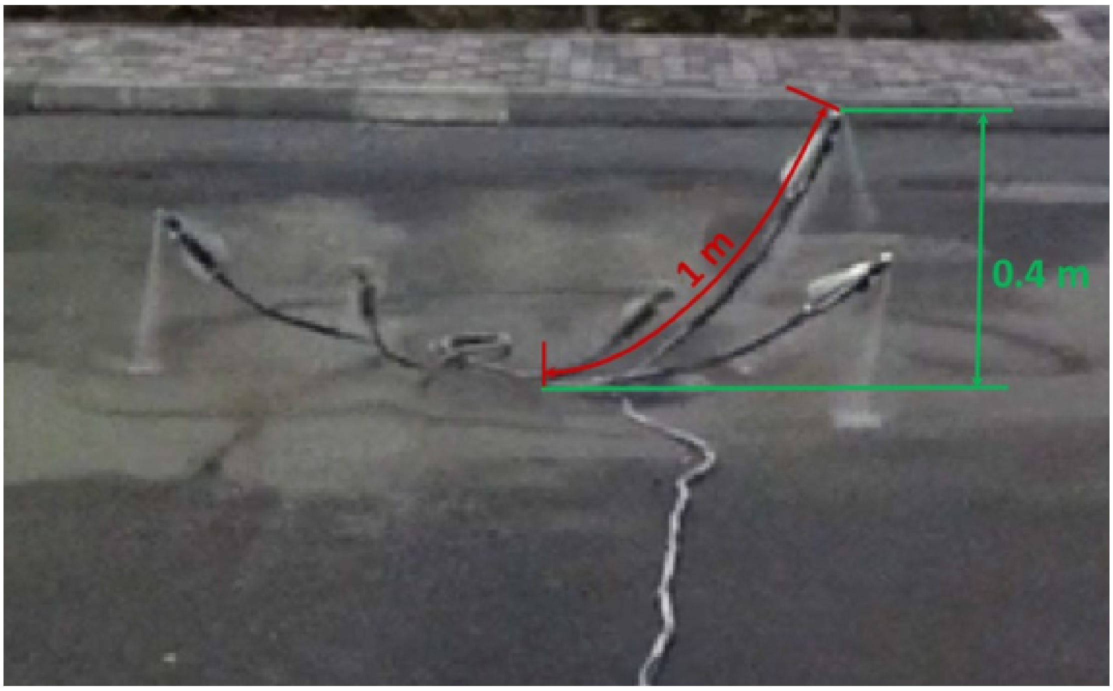


Figure 2.2.7: Sequence shot of water jet propelled long reach robot. Courtesy of Tokyo Institute of Technology.

A completely unique approach to propulsion of a slim long robot [30] was developed at the Tokyo Institute of Technology in 2016. Instead of a rigid body with motorized joints, this robot consists of a flexible hose with three nozzles at its tip, which expel jets of pressurized water to bring the tip of the hose into motion by means of reactive propulsion. This approach allows the robot to move underwater as well as hover at a low height above the surface of the water (or ground, as shown in Figure 2.2.7) although its mobility is limited by the necessary connection to a stationary high-pressure water pump.

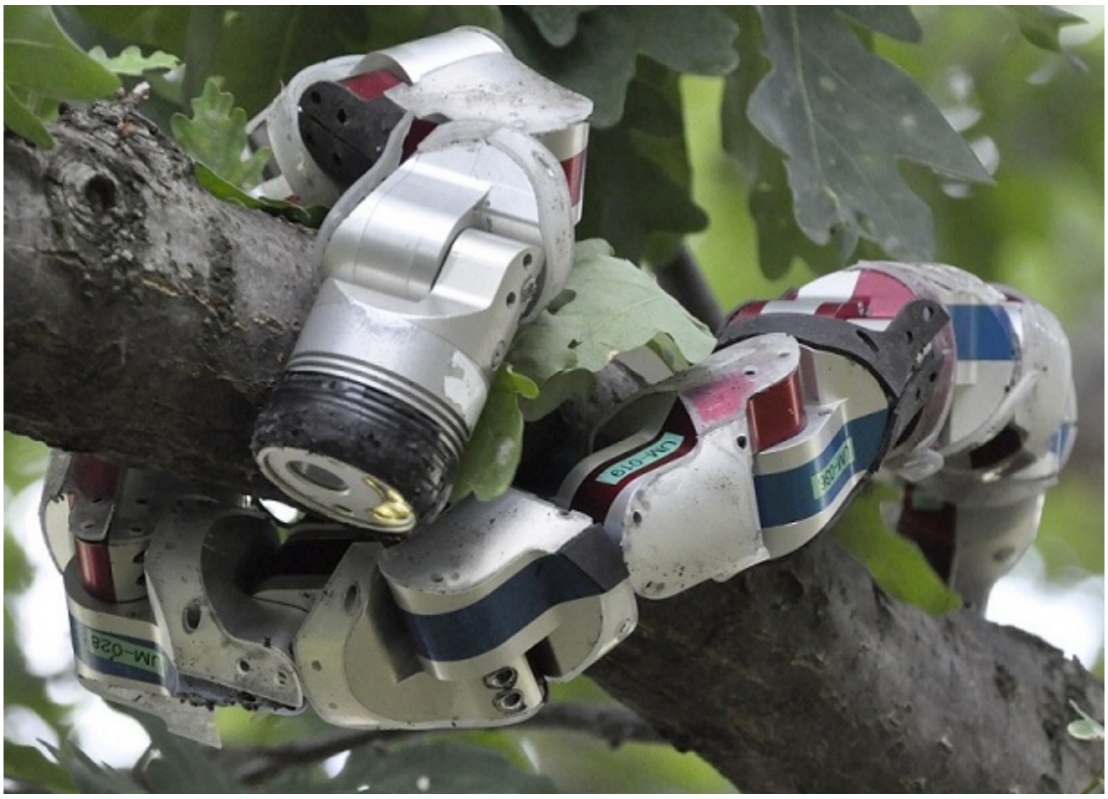


Figure 2.2.8: Unified Snake Robot. Courtesy of Carnegie Mellon University.

Perhaps one of the more famous snake-like robots is the Unified Snake Robot developed at the Carnegie Mellon University [35] over multiple iterations and refinements to its design. Mechanically, the over-all structure of the CMU Unified Snake Robot is similar to that of the ACM-R3, with 1-DOF motorized joints alternating at 90º angles, with the addition of features for improved practicality and reliability, such as slip-clutches to protect gearboxes from back-driving, and Shape Memory Alloy actuated bistable brakes. Another non-obvious improvement of the design is in its 36 V motors, which require significantly less current (as compared to, for example, our robot's 6V-rated hobby servo motors) for the same quantity of mechanical power. Over multiple iterations of the design, CMU's Unified Snake Robot has evolved into a modular design consisting of custom mechanical and electronic hardware capable of significant feats of vertical mobility, such as climbing trees and traversing vertical pipes. Unfortunately, the Unified Snake Robot's otherwise spectacular mobility is held back by the lack of an on-board power supply, mandating the use of a power tether to a nearby power supply in order to operate.

It must be admitted that mechanically, there is nothing new about our robot. Robot snakes with articulated with universal-joint linkages have been built before [26][27], robot snakes propelled with servo motors have been built before [35], and robot snakes have also been built by means of rapid prototyping technology. However, to the best of our knowledge, our robot snakes were the first to be built in the State of Nevada. Table 2.2.1 compares our robot snake to the ones mentioned above. The water-jet-propulsion robot is excluded from the list because of its radically different means of operation.

Table 2.2.1. Comparison of our robot snake to others. *(Unknown values left blank)*

Name	**Length (m)**	**Mass (kg)**	**Torque (N·m)**	**Motorized Joints DOF**	**Max Speed (m/s)**
ACM-III	2	28		20	0.6
ACM-R3	1.8	12	15	20	1
ACM-R3n	1.42	8	12	16	
ACM-R4	1.1	9.5		9	
ACM-R4.1	0.72	6.3	6.7	9	0.75
ACM-R5	1.6	6.5		16	0.4
ACM-R7	1.6	11.7	16	18	1
ACM-R8	2.01	36.5	100	8	0.4
CMU USR	0.94	2.9	1.3	16	
UNR Snake	1.1	1.35	0.63	16	0.25

Chapter 3

Theory of Operation

Snakes, both natural and robotic, make use of a versatile assortment of gaits of locomotion. This chapter describes the theoretical aspects of snake biomechanics, specifically the common gaits of locomotion employed by snakes, as well as the gaits of locomotion demonstrated by our robot snake. This chapter also briefly describes the mechanics of a linkage known as the Universal Joint because the mechanics of this linkage is the fundamental concept around which the mechanical structure of the robot is built.

3.1 Gaits of Locomotion

Snakes are able to move by means of different modes of locomotion, depending on the situation, terrain, and circumstances; see Figure 3.1.1.

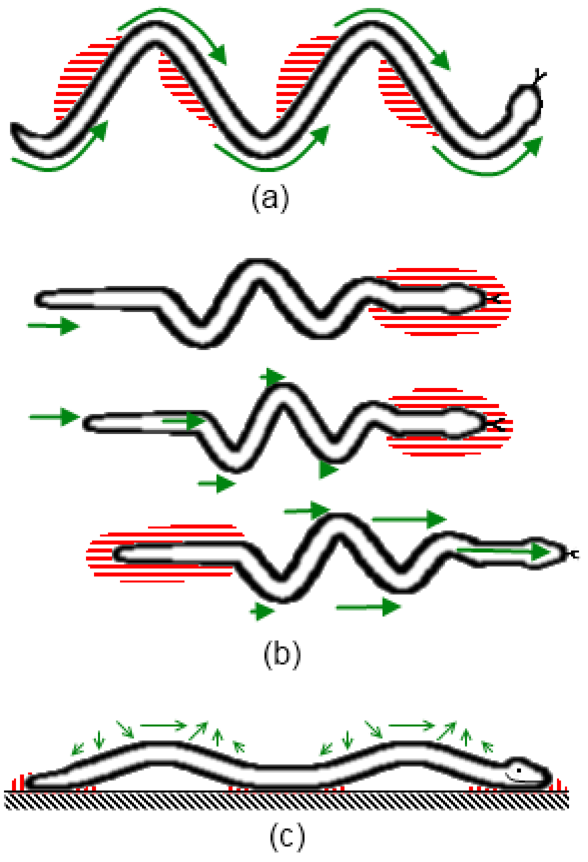


Figure 3.1.1: Some of the gaits of locomotion observed in snakes: serpentine (a), concertina (b), and rectilinear (c). Red striped areas surround areas of pressure. Green arrows denote local motion of the body of the snake.

Modes of locomotion of snakes include serpentine locomotion (*a.k.a.* lateral undulation,) sinus lifting, and sidewinding, which have been demonstrated by our robot snake. Two other modes of locomotion – rolling and differential sidewinding – are also considered. The prototypes' power-to-weight ratio is too low to experiment with the climbing locomotion observed in tree snakes, and too few degrees of freedom to demonstrate rectilinear or concertina locomotion. Therefore, those gaits remain beyond the scope of this project.

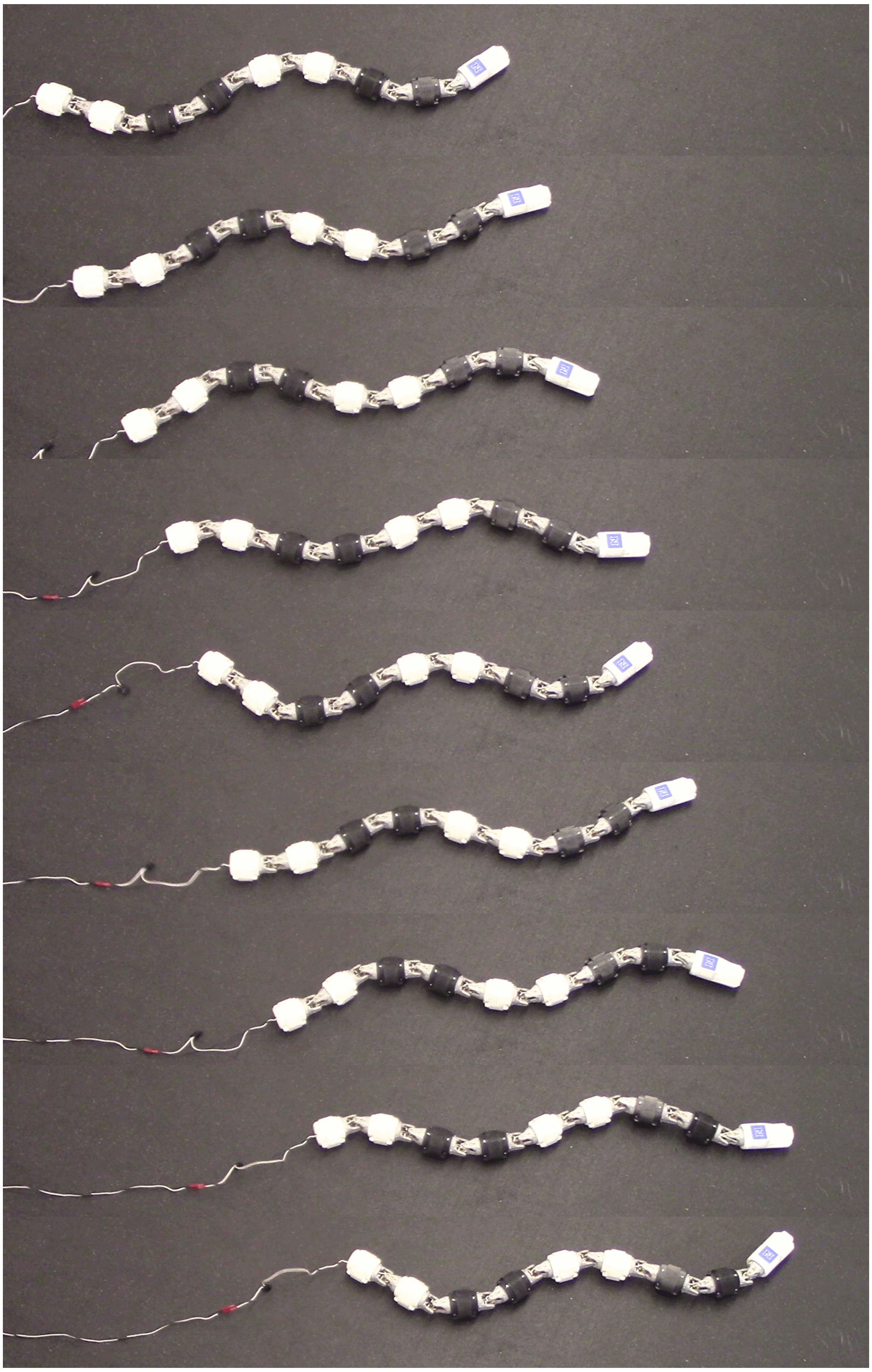


Figure 3.1.2: Lateral undulation. Nine frames recorded over a span of four seconds in increments of 0.5 seconds, from top (t=0) to bottom (t=4).

Lateral undulation (Fig. 3.1.2) is, together with sinus lifting, a commonly observed mode of locomotion for live snakes. With no live snakes immediately available, this gait of locomotion is demonstrated by our robot snake. In the process of serpentine locomotion, a snake's body forms a periodic parametric curve, visually similar to a sine wave. Unlike a sine wave, if the amplitude of this curvature was to increase far enough, the curve would touch and intersect itself.

Note from Figure 3.1.2. how the 'sinusoidal wave' propagates along the snake's body toward its tail. Anisotropic friction of natural snakes' lower scales with respect to the ground improves efficiency by improving traction *without* a proportional increase in drag. Our robot snake uses passive wheels to achieve the same effect.

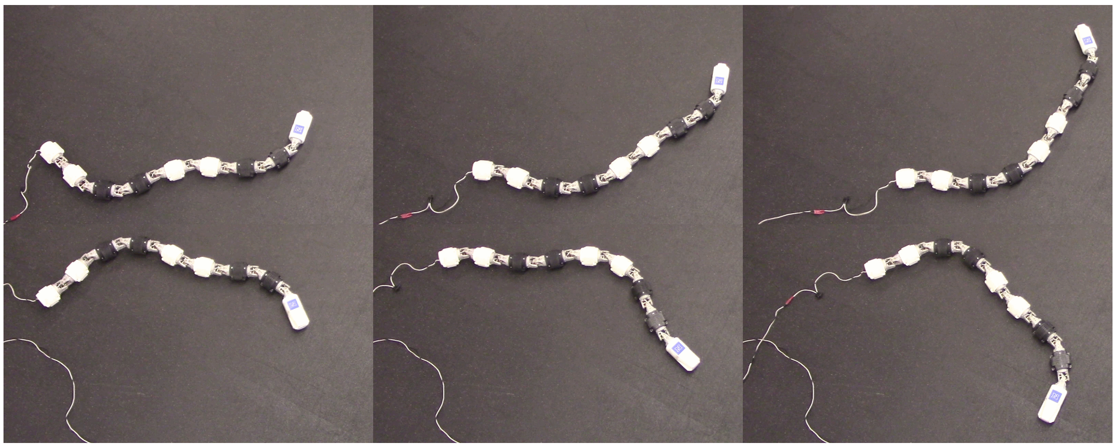


Figure 3.1.3: During the lateral undulation mode of locomotion, steering adjustments are accomplished by a constant offset to the curvature of the snake's body.

With lateral undulation, steering corrections are as simple as changing the average curvature of the snake's body. If the average curvature of the snake's body is to the snake's left, then the snake gradually turns to the left as it slithers forward via lateral undulation. Likewise, if the average curvature of the snake's body is to the snake's right, then the snake gradually turns right. This is illustrated on Figure 3.1.3, where the robot snake turns to the left (top three images) and to the right (bottom three images.)

At a glance, flat serpentine locomotion may be difficult to distinguish from sinus lifting, especially in situations in which human observers have a greater incentive to physically distance themselves from live snakes, rather than to spend the time to make any observations related to the specific details of the snakes' locomotion.

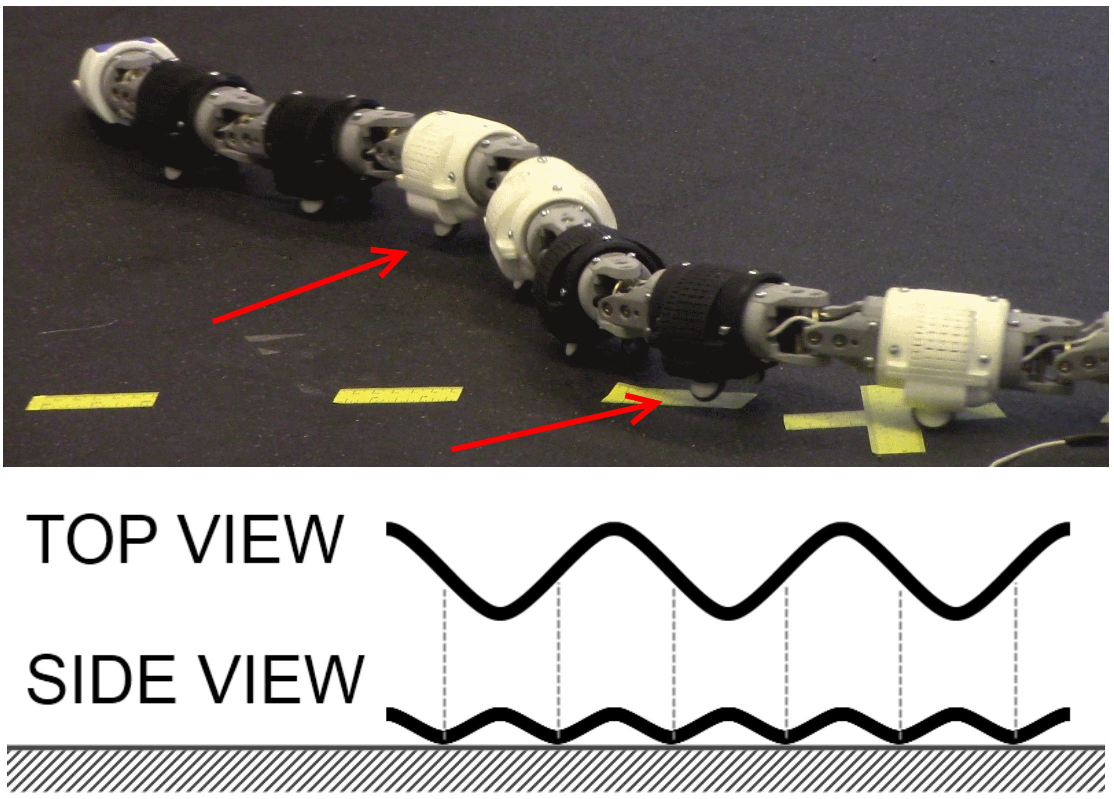


Figure 3.1.4: Sinus Lifting; a photograph (top) and an exaggerated conceptual drawing (bottom.) Note how some wheels in the top image are lifted from the floor.

Sinus lifting (Fig. 3.1.4) is a subtle modification to serpentine locomotion. If the shape of a snake's body is qualitatively compared to a sine wave, then sinus lifting may be described by the act of lifting the crests and troughs (points of local minima and maxima, or points of maximum absolute curvature, or 'anti-nodes') off the ground. Most of the traction is created by the relatively straightened out segments of the snake's body (to continue the analogy to a sine wave, the 'nodes') between the crests and troughs, while the anti-nodes create more drag than traction. By lifting the anti-nodes off the ground, the snake can eliminate the drag they create while sacrificing only a negligible amount of traction. This helps to improve efficiency of its locomotion.

Another mode of locomotion is possible if the nodes, rather than the anti-nodes, are lifted from the ground, as shown in Figure 3.1.5. In this case, the snake would 'roll' upon the anti-nodes formed by the curvature of its body. Unlike lateral undulation and sinus lifting, this mode of locomotion would propel the snake in the same direction as the propagation of the pattern of curvature along its body. However, this mode of locomotion is inefficient for snakes with anisotropic surface friction adapted for lateral undulation and sinus lifting, including ours. We have called this mode of locomotion *Sinus Rolling.* Because we have no evidence of its use by real snakes, we have no grounds to call it a biomimetic mode of locomotion.

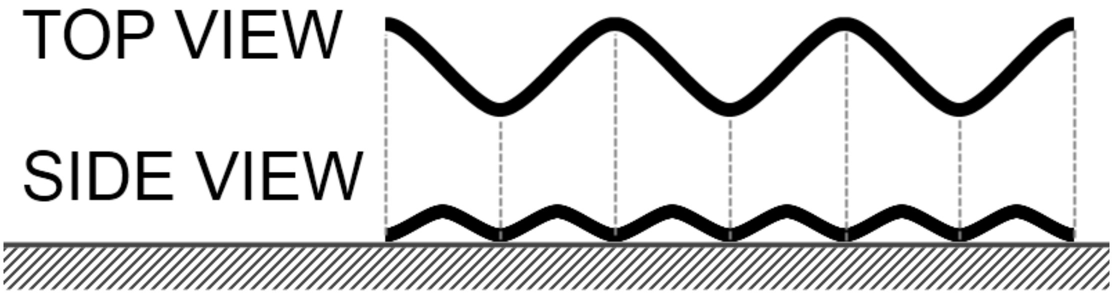


Figure 3.1.5: Sinus rolling – a hypothetical gait of locomotion.

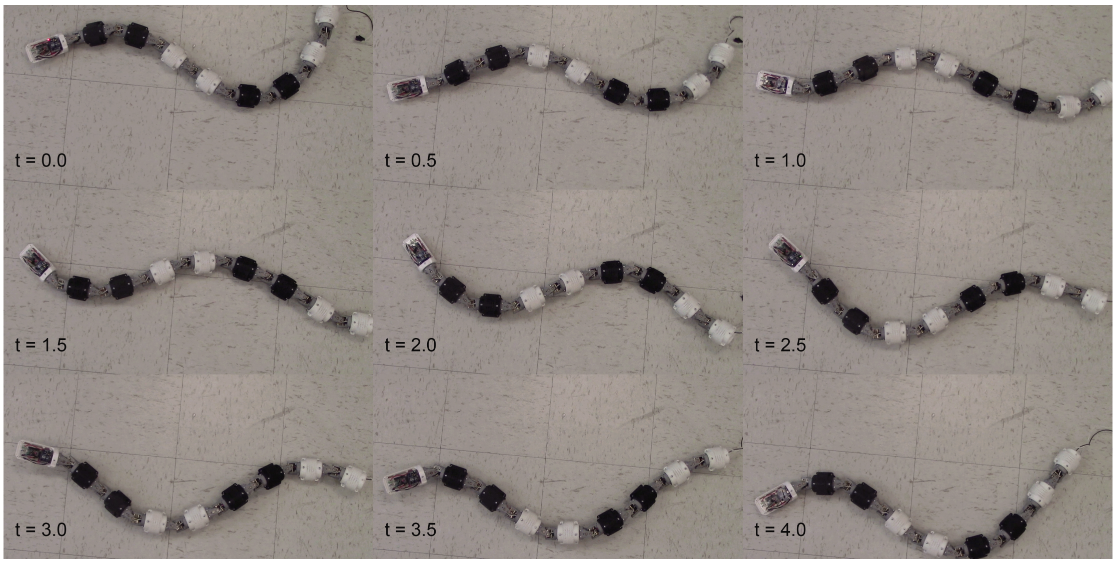


Figure 3.1.6: Sequential images of sidewinding locomotion. Nine frames recorded over a span of four seconds in increments of 0.5 seconds, from top left (t=0) to bottom right (t=4).

Sidewinding locomotion (Fig. 3.1.6) is often observed in sidewinders. It is similar to serpentine locomotion in that the snake's body takes the shape of a propagating, horizontally-oriented sine wave. It differes from serpentine locomotion in that the snake's body also forms a secondary sine wave, of the same wavelength but lesser magnitude and vertically-oriented, propagating at the same speed as the primary sine wave. The secondary sine wave is out of phase from the primary sine wave by a quarter of their wavelength. The shape of the snake's body in this mode of locomotion may be compared to an elliptical helix which 'rolls' sideways. Note that the snake itself does not have to roll to use sidewinding locomotion; only the curve formed by the shape of its body needs to roll in the desired direction of movement. However, if the magnitude of the vertical sinusoidal component is comparable to the horizontal compoment, then a rolling motion is likely to result, as the motion patern of rolling locomotion is mathematically similar to the motion pattern of sidewinding locomotion.

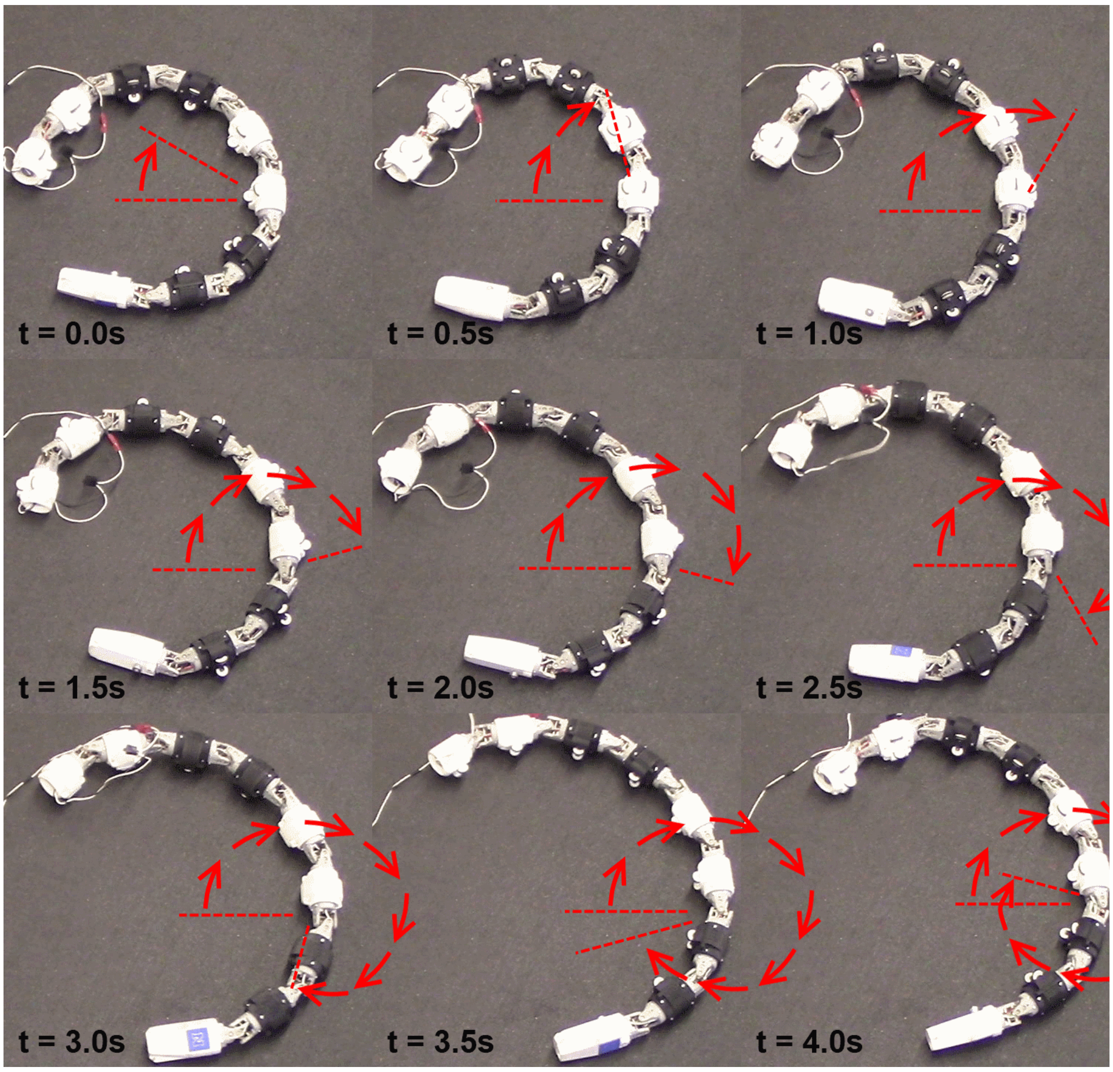


Figure 3.1.7: Rolling motion of a snake. Images were taken in sequence from left to right, then from top to bottom, in increments of 0.5 seconds.

Rolling (Fig. 3.1.7) is rarely observed in live snakes, apart from rolling to rectify their orientation if they find themselves upside-down. The exact reasons for its disuse in live snakes is beyond the scope of this paper. The robot snake described in this paper, however, has demonstrated the ability to utilize this form of locomotion. Ordinarily, a snake's orientation with respect to the directions of up and down is stable owing to its mostly-flat sinusoidal shape against the approximately flat ground plane. The snake can, however, change the aforementioned point of stability by gradually changing its sinusoidal shape from one parallel with a plane that is horizontal relative to the snake's head, to a sinusoidal shape parallel with a plane that is vertical relative to the snake's head. The plane parallel to the shape of the snake's body remains approximately parallel with the ground, near mechanical equilibrium, while the snake's body rotates in the opposite direction.

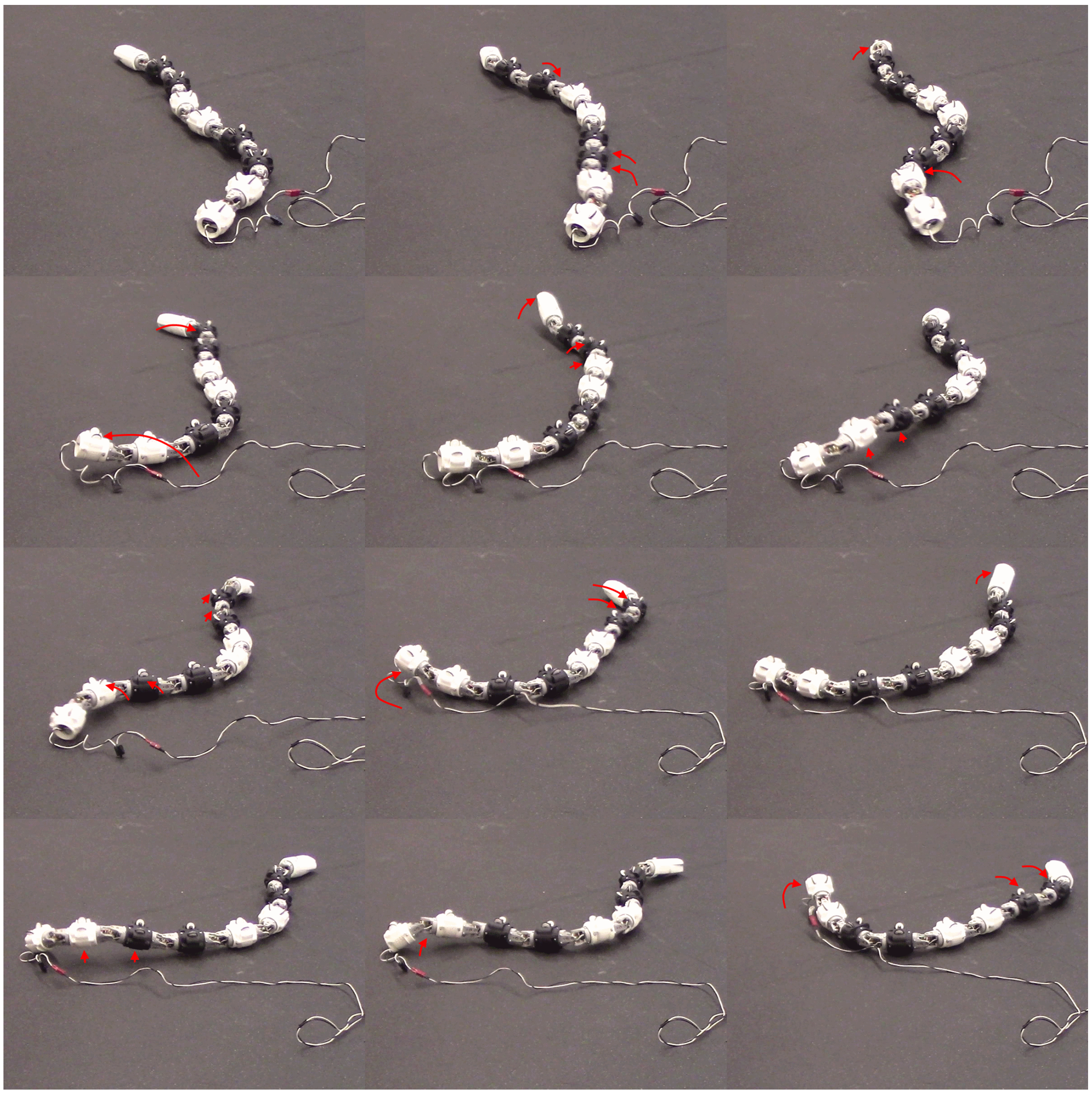


Figure 3.1.8: Differential sidewinding. Images were taken in increments of 0.5 sec, from left to right, then top to bottom.

Another gait of locomotion has been developed based on the sidewinding gait (see Figure 3.1.8.) Like the sidewinding gait, it uses a phase-offset vertical compoment to make the snake's body assume a helical shape. Unlike the sidewinding gait, this motion varies the amplitude of the vertical component from the head to the tail of the snake. This difference causes the snake to rotate along the horizontal plane rather than to move sideways. We called this gait of locomotion *Differential Sidewinding*. Lifting parts of the snake off the ground makes Differential Sidewinding inefficient in terms of energy, but it may be useful in situations where terrain prohibits a more efficient turn by means of a sideways curvature offset to the lateral undulation gait.

3.2 Mechanics of Universal Joint

The center-lines of two rotational constraints intersect orthogonally in three-dimensional space. This allows some rotation about two hinges, but one constraint remains for the remaining direction of rotation. The Universal Joint is built around this concept. The seven or eight (depending on which prototype) motorized joints of the robot snake are universal joints, each powered with two self-contained servo motors with closed-loop control systems (also known as RC servos.)

3.3 Gear Design

Rapid prototyping methods, including laser cutting and various forms of 3D printing, have been utilized to manufacture various structural and mechanical parts for the robot and related systems. This allowed the actual manufacturing process to be out-sourced, while the STL files may be archived as purely digital information, or transmitted as such over the Internet. This approach has its advantages and disadvantages. The advantages are in the ease of storage and transfer of electronic geometry files, as well as the versatility the rapid prototyping technology – 3D printed parts are not generated by rapidly rotating cutting tools, and are therefore free from constraints inherent to design for traditional manufacture. The disadvantages of the use of this relatively new technology is that it is still maturing, and is not yet capable of precision comparable to traditional manufacturing methods. Although this section focuses mostly on the design of gears, it must be noted that the specifics of available manufacturing technology had to be taken into account for the sake of delivering a design that is realistically manufacturable.

There are two kinds of gears as classified by tooth profiles; these are involute [22][23][10] and cycloidal [11][12] gears. The involute gear is the most commonly used one today, with a tooth profile based on the involute curve. The cycloidal gear has its advantages, but requires very high manufacturing precision [24]. Both gear profiles were considered for the design of the robot snake's drive train, but ultimately the involute gear profile was selected. This decision was made in part due to constraints on manufacturing precision of Shapeways' Raw Brass parts [25] and in part due to greater abundance of relevant information available on involute gears compared to cycloidal gears.

The fundamental law of gearing states that the angular velocity ratio between two meshing gears must remain constant as the gears turn. This is trivial if gears were cylinders with infinitesimal teeth made of some ideal unbreakable material and arranged at a perfect distance apart within the mechanism. In reality, gears must be constructed of real materials with finite material strength, and mounted at some distance with subtle variations due to realistic imperfections of the manufacturing process. This necessitates gear teeth to be larger than the limits of manufacturing precision and thick enough to withstand the force they are menat to transfer. Shaping the contact surfaces of gear teeth based on the involute curve is one way to make a pair of meshing gears obey (or more realistically, to make them approximate) the fundamental law of gearing. This is the principle behind involute gears.

The sequential outline of the geometric construction process of the involute spur gear is found in Appendix D.

Chapter 4

Practical Implementation

This chapter describes the details of implementation of the robot and its user interface. These include the robot's mechanical gear train, several versions of the robot's central electronic hardware, the electronic hardware of the physical implementation of the user interface, virtual implementation of the user interface, the software algorithms used by the robot snake and its user interface, and the signal protocol that is used to transmit commands for motion pattern parameter updates.

The system consists of a user interface (which can exist either as dedicated hardware or as an app running on a general-purpose personal computer) that receives input from the user and transmits it as radio signals via XBee radio module; and the robot snake itself, which receives radio signals through another XBee radio module and interprets the signals to update its motion pattern parameters. This arrangement is illustrated in Figure 4.0.1.

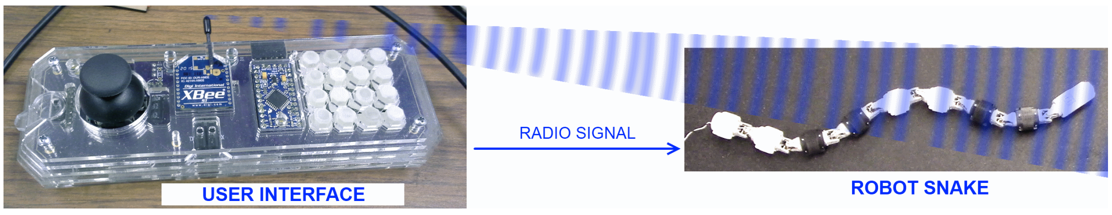


Figure 4.0.1: Graphical representation of robot snake receiving radio signals from the remote command unit.

4.1 Mechanical Design: Prototype I

For Prototype I (shown in part on Figure 4.1.1,) I designed the black and white tubes that connect the motorized joints. As such, this section is an analysis rather than a firsthand description of its design.

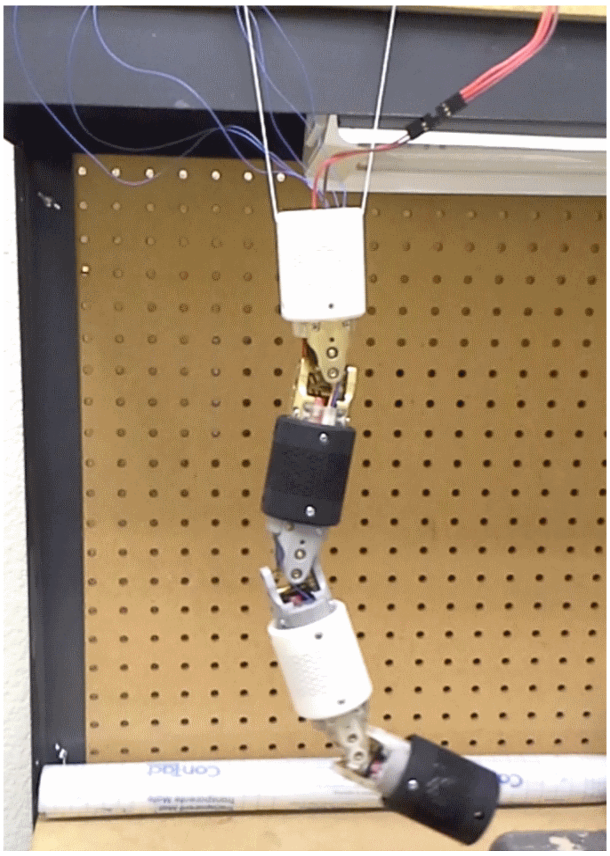


Figure 4.1.1: Three segments consisting of original (Prototype I) robot snake parts suspended on string for testing.

At 5.0 V, the RC servo used (namely, the MKS DS450) has an inconsistent given torque specification ranging from 0.299 Nm to 0.316 Nm (for simplicity of calculations, let us assume it is 0.300 Nm,) a speed specification of 9.106 rad/s, and an angular range of ±80º (1.396 rad) for a total of 160º (2.793 rad.) The "small gear" directly on the servo had 25 teeth and meshed with an even smaller gear, which had 17 teeth, integrated into the compound "mid gear," for a gear ratio of 25:17. The other gear on the "mid gear" meshed with the core of the 2-DOF joint – the "ball gear" – with a gear ratio of 1:1, which neither increased nor reduced the total gear ratio. The total ratio of the gear train was 25:17. Thus, at 5.0 V, the output torque (neglecting for friction) was 0.204 Nm, the output speed (neglecting for friction and other loading effects) was 13.4 rad/s, and the output angular range (neglecting angular constraints of the system) was ±117.65º (2.053 rad) for a total of 235.29º (4.107 rad.) Unfortunately, natural mechanical constraints limited the actual range of motion for each joint to only about ±45º (0.785 rad) for a total of about 90º (1.571 rad). This was sub-optimal: both the maximum angular displacement (as well as its time derivative, angular velocity) of each joint is significantly greater than that which would provide any additional benefit, whereas the output torque of each joint is insufficient to achieve locomotion. This configuration is summarized in Figure 4.1.2. In the earliest prototype, it was also observed that resin gear located between the RC servo and the rest of the gear train – the only non-metal gear in the whole mechanism – was not sufficiently durable to withstand the mechanical load.

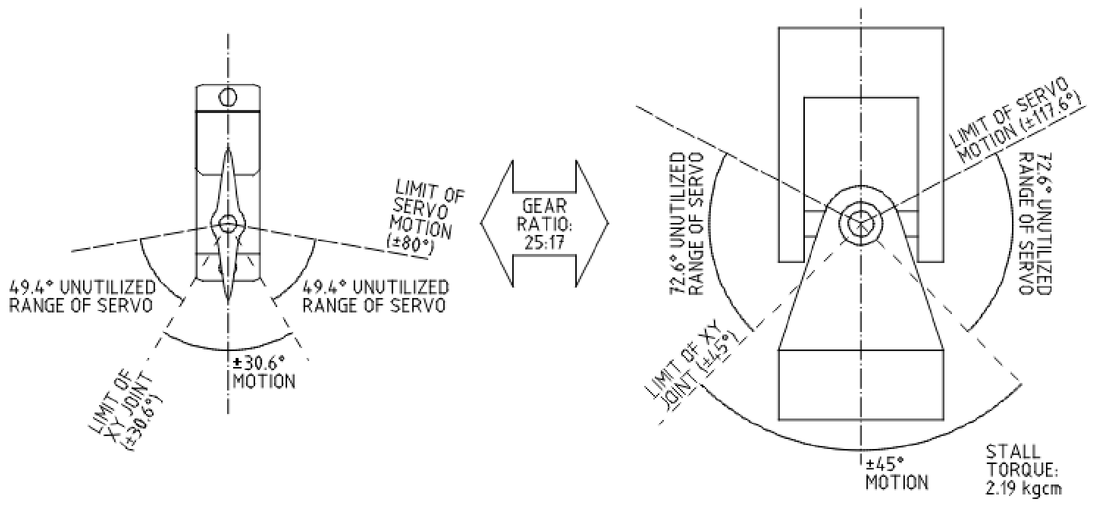
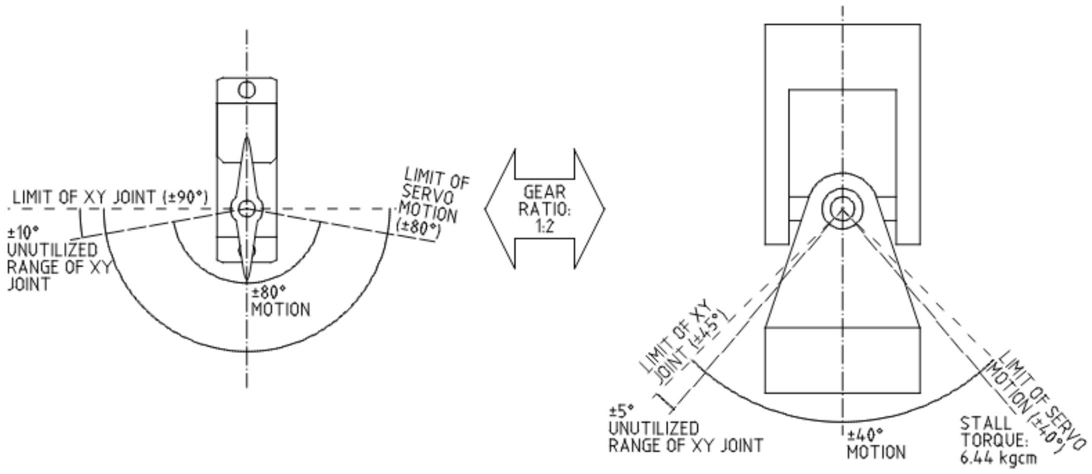

LIMIT OF SERVO MOTION (±80°)
49.4° UNUTILIZED RANGE OF SERVO
49.4° UNUTILIZED RANGE OF SERVO
±30.6° MOTION
LIMIT OF XY JOINT (±30.6°)
GEAR RATIO: 25:17
72.6° UNUTILIZED RANGE OF SERVO
LIMIT OF SERVO MOTION (±117.6°)
72.6° UNUTILIZED RANGE OF SERVO
LIMIT OF XY JOINT (±45°)
±45° MOTION
STALL TORQUE: 2.19 kgcm

Figure 4.1.2: Summary diagram of gear train of Prototype I.

4.2 Mechanical Design: Prototype II

The obvious objective of the next design iteration of the gear train, before even engineering trade-offs were considered, was the redesign of the gear ratio to utilize a greater range of motion of the motors. Normally, this would come at the expense of maximum angular velocity; and with RC servos, this would also come at the expense of the range of angular displacement of the motorized mechanical joint. In this case, however, the superfluous range of motion of the previous gear train was nullified by the innate angular displacement limitations of the mechanical joint in its former configuration. Rather than a trade-off, this improvement was effectively free. The new configuration is summarized in Figure 4.2.2. The details of the original gear train and two considered variants for the redesign are listed in Table 4.2.1. One may notice that the *lesser mid gear* has a non-integer number of teeth – 25.2. This may appear unusual because it would cause issues with continuously-rotating gears, but gears rotating within angular constraints do not turn far enough to necessitate an integer number of teeth.



LIMIT OF XY JOINT (±90°)
±10° UNUTILIZED RANGE OF XY JOINT
LIMIT OF SERVO MOTION (±80°)
±80° MOTION
GEAR RATIO: 1:2
LIMIT OF XY JOINT (±45°)
±5° UNUTILIZED RANGE OF XY JOINT
±40° MOTION
LIMIT OF SERVO MOTION (±40°)
STALL TORQUE: 6.44 kgcm

Figure 4.2.2: Summary diagram of gear train of Prototype II.

Table 4.2.1. Numerical details of design of gear train

	Prototype 1 (initial)	**Prototype 2 (considered)**	**Prototype 2 (selected)**
Small gear pitch radius (mm)	5.00	4.00	4.50
Lesser mid gear pitch radius (mm)	3.40	5.00	5.40
Greater mid gear pitch radius (mm)	6.00	4.50	4.50
Ball gear pitch radius (mm)	6.00	7.50	7.50
Adjustment: small:mid gears centers	0.00	0.60	1.50
Adjustment: mid:ball gears centers	0.00	0.00	0.00
Pressure angle: small:mid teeth (deg)	21.50	25.00	20.00
Pressure angle: mid:ball teeth (deg)	21.00	25.00	20.00
Torque efficiency of small:mid interface	93.04%	90.63%	93.97%
Torque efficiency of mid:ball interface	93.36%	90.63%	93.97%
Total torque efficiency of gear train	**86.86%**	**82.14%**	**88.30%**
Comparison of the above to original design		*94.56%*	*101.66%*
Base circle radius: small gear (mm)	4.6521	3.6252	4.2286
Base circle radius: lesser mid gear (mm)	3.1634	4.5315	5.0743
Base circle radius: greater mid gear (mm)	5.6015	4.0784	4.2286
Base circle radius: ball gear (mm)	5.6015	6.7973	7.0477
Contact depth: small:mid gears (mm)	**0.58**	**0.84**	**0.60**
Comparison of the above to original design		*144.27%*	*102.15%*
Contact depth: mid:ball gears (mm)	**0.80**	**1.12**	**0.72**
Comparison of the above to original design		*141.06%*	*90.80%*
Gear ratio: small:mid	1.47	0.80	0.83
Gear ratio: mid:ball	1.00	0.60	0.60
Total gear ratio	1.47	0.48	0.50
Pitch force at small/mid teeth, N/Nm	**294.12**	**120.00**	**111.11**
Comparison of the above to original design		*40.80%*	*37.78%*
Pitch force at mid/ball teeth, N/Nm	**166.67**	**133.33**	**133.33**
Comparison of the above to original design		*80.00%*	*80.00%*
Load force at small/mid teeth, N/Nm-load	**338.60**	**146.09**	**125.83**
Comparison of the above to original design		*43.15%*	*37.16%*
Load force at mid/ball teeth, N/Nm-load	**178.52**	**147.12**	**141.89**
Comparison of the above to original design		*82.41%*	*79.48%*
Stall torque of servo (Nm)	0.31556	0.31556	0.31556
Stall pitch force of smal:mid teeth (N)	58.72	71.50	65.90
Stall pitch force of mid:ball teeth (N)	30.96	72.00	74.31
Stall torque of robot snake joint (Nm)	0.18576	0.54000	0.55729
Comparison of the above to original design		*290.70%*	*300.01%*
Movement range of servo and small gear (±degrees)	80.00	80.00	80.00
Movement range of mid gear (±degrees)	117.65	64.00	66.67
Movement range of ball gear and segment (±degrees)	**117.65**	**38.40**	**40.00**
Maximum number of teeth: small gear	26.18	20.94	23.56
Maximum number of teeth: lesser mid gear	17.80	26.18	28.27
Maximum number of teeth: greater mid gear	31.42	23.56	23.56
Maximum number of teeth: ball gear	31.42	39.27	39.27
Maximum pitch of small gear alone (teeth/mmpr)	5.236	5.236	5.236
Maximum pitch of lesser mid gear alone (teeth/mmpr)	5.236	5.236	5.236
Maximum pitch of greater mid gear alone (teeth/mmpr)	5.236	5.236	5.236
Maximum pitch of ball gear alone (teeth/mmpr)	5.236	5.236	5.236
Maximum pitch of small:mid gears (teeth/mmpr)	5.236	5.236	5.236
Maximum pitch of mid:ball gears (teeth/mmpr)	5.236	5.236	5.236
Teeth: small gear			21
Teeth: lesser mid gear			25.2
Teeth: greater mid gear			21
Teeth: ball gear			35
Degrees per tooth: small gear			17.142857143
Degrees per tooth: lesser mid gear			14.285714286
Degrees per tooth: greater mid gear			17.142857143
Degrees per tooth: ball gear			10.285714286

The functional specifications of the new gears were computed by entering the parameters into a spreadsheet and incrementally changing them until the resulting specifications were deemed satisfactory.

Working under the design constraint of compatibility with structural components designed previously by other participants made the sum of pitch radii of each pair of meshing gears a hard design constraint. The manufacturing precision limits of the Shapeways' Raw Brass manufacturing process was another hard design constraint, because making the gears' teeth profiles too fine would result in poor meshing due to manufacturing imperfections. Similarly, the contact depth (the depth of overlap between addendum circles of meshing gears) of the gears' teeth had to be sufficiently great to accommodate any slack between the structural parts and the bearings that held the pairs of gears meshed together to prevent their teeth from slipping past each other.

The gear ratio (which governs the torque vs. speed trade-off) was a firm design requirement. The objective was to map the natural range of motion of the structural components of the U-Joint linkage (which was approximately ±45º) to roughly correspond with the range of motion of the servo motor (which was specified as ±80º.)

The thickness of the gears' teeth was another firm requirement, as making them too thin increased the risk of breaking under high torques.

The pressure angle of the meshing gears was a soft design objective. A lower pressure angle would mean less friction and therefore greater efficiency of the gear train, and a higher efficiency is almost always a desirable feature.

A mechanical drawing summarizing the pitch radii and arrangement of gears in the new gear train may be found in Appendix A.

Gear	**R_{pitch}**	**Addendum**	**Dedendum**	**Tooth Thickness**
Servo Gear	4.5 mm	0.4257 mm	0.3714 mm	1.3264 mm
Mid-Gear, Servo Side	5.4 mm	0.2714 mm	0.4257 mm	1.3325 mm
Mid-Gear, U-Joint Side	4.5 mm	0.4523 mm	0.3714 mm	1.3264 mm
U-Joint Core Gear	7.5 mm	0.2714 mm	0.5523 mm	1.3392 mm

Gear Pair	**Ratio**	**Contact Depth**	**Pressure Angle**
Servo Gear and Mid-Gear	5/6	0.6971 mm	20º
Mid-Gear and U-Joint Core	3/5	0.7237 mm	20º
Whole gear train	1/2	*N/A*	*N/A*

Once the specifications for the gears (*i.e.*, gear ratio, pitch radius, pressure angle, number of teeth,) were selected, the outlines of the gears were drafted in QCAD, a freeware 2D CAD program. The gears' involute curves, which would have required to QCAD's premium features to be drafted in QCAD, were instead drafted in LibreCAD, another freeware 2D CAD program. The outlines were then saved in DXF files and imported into Solidworks as 2D sketches, and then extruded into 3D solid models.

4.3 Electrical Design: Robot Snake Brain 1.0

The electrical hardware of the robot snake was initially bult around the Spark Fun Electronics' Arduino Pro Mini[41] microcontroller board, which consists of an ATMega328 microcontroller chip with special bootloader firmware and minimal peripheral hardware. The microcontroller board may be programmed via the Adafruit FTDI cable, and possibly other USB-to-serial adapters that support hardware flow control. The microcontroller board is situated in the middle of the top half of the robot snake's skull. A photograph of the top half of the Robot Snake Brain 1.0 is shown on Figure 4.3.1.

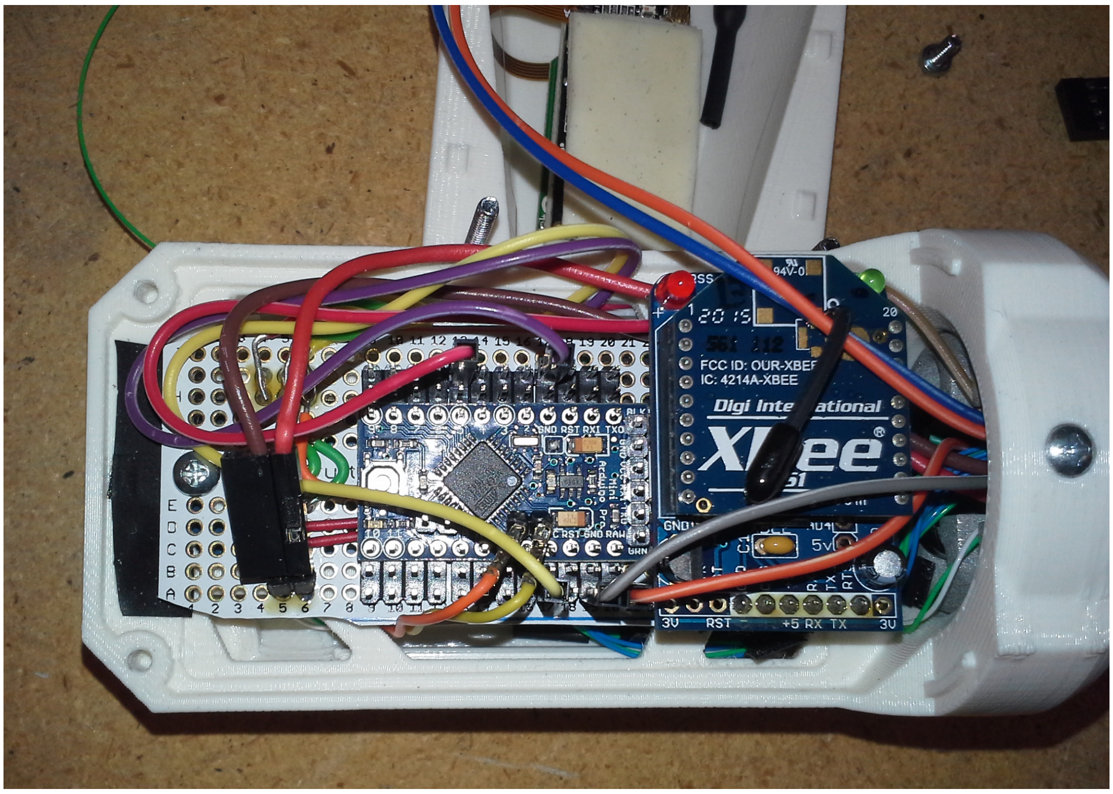


Figure 4.3.1: Top half of Robot Snake Brain 1.0 is visible when the top cover of the robot snake's skull is removed. Microcontroller board, XBee, XBee adapter, and general-purpose protoboard (used for interconnection) are visible in this photograph.

An XBee wireless modem was utilized to receive a wireless command serial signal from the Remote Command Unit (RCU) and to pass the aforementioned signal to the microcontroller. The XBee modem was located on a level-shifting adapter board [43] in the upper rear of the top half of the robot snake's skull. The adapter board was necessary to shift I/O signal levels between 5V logic (for the ATMega328) and 3.3V logic (for the XBee,) and also because the XBee modem has 2-mm-pitch headers, whereas the general-purpose PCB used for interconnection has 2.54-mm (0.100 inch) spacing between its plated holes.

Although the ATMega328 is theoretically capable of generating Pulse-Width-Modulated (PWM) signals that could serve as reference inputs to the robot's self-contained RC servo motors, the ATMega328 has too few PWM-capable I/O pins for all fourteen servos, and such a minimalist design would scale poorly if the number of motorized joints was to be increased. Instead, a dedicated 16-channel PWM driver chip, the NXP PCA9685, was utilized to generate the PWM signals to be sent to the servos. The Adafruit Industries PCA9685 breakout board [44] was used for ease of connectivity and debugging. The PCA9685 chip on the breakout board receives commands from the ATMega328 microcontroller via I2C link and sends an appropriate PWM command signals to each servo over a dedicated electrical connection. The PCA9685 breakout board is located in the center of the bottom half of the robot snake's skull.

An on-board Inertial Measurement Unit (IMU) was required [14]. An IMU, namely the Bosch BNO055 chip on a breakout board [45], was included in the rear of the bottom half of the robot snake's skull. Although the BNO055 has proven to be a very capable IMU, we were unable make it function alongside the PCA9685 PWM control chip on ATMega328's I2C line. Because the ATMega328 in the Arduino Pro Mini [41] has only one dedicated I2C channel, a second Arduino had to be utilized to receive signals from the BNO055 to be transmitted through the XBee modem. [33] Unfortunately, this does not allow for the Arduino responsible for the robot snake's motion to have real-time access to the data from the IMU. Because we had initially assumed that we could have the BNO055 chip and the PCA9685 chip share one I2C line, the necessity of adding the second Arduino was unanticipated, and no provisions were made for its placement within the robot snake's skull. Instead, it was located in the connector tube behind the first motorized X-Y joint.

Central electrical interconnection within the robot snake's skull was accomplished mainly by a general-purpose PCB cut to sufficiently compact size with a rotary tool and augmented with jumper wires. To save space, the Arduino microcontroller board was connected to the general-purpose PCB by means of two rows of male headers intentionally deformed so that each pin of the headers makes reliable electrical contact with the plated holes of the microcontroller board. The XBee modem adapter board [43] was connected by means of a male header mated with a female header. The electrical schematic of general electrical interconnection within the robot snake's skull (*a.k.a.* Robot Snake Brain 1.0) may be found in Appendix B.

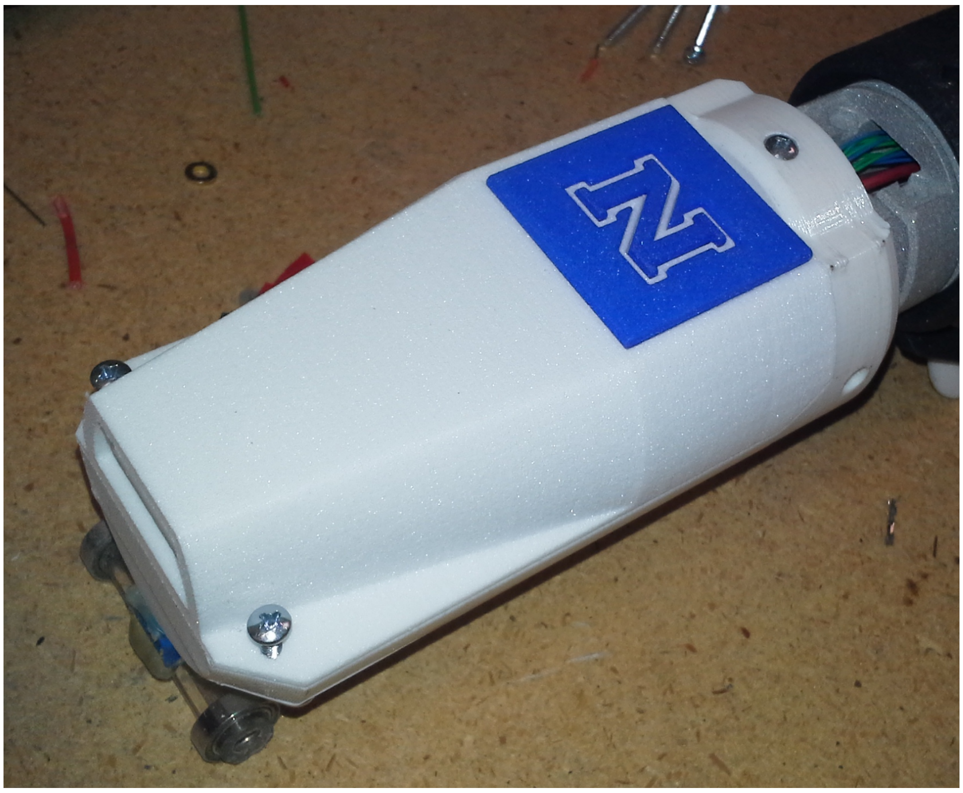


Figure 4.3.2: Robot Snake's skull, with a later variant of the top cover.

A brief note on the robot snake's control unit enclosure, also known as its *'Skull.'* The design of the Skull was more art than science, although it had to be geometrically compatible with all adjacent hardware. Later versions of the robot snake's control unit circuit board likewise had to be designed for geometric compatibility with the existing skull. Later iterations of the top cover of the Skull included the university logo as an identifying feature as shown in Figure 4.3.2. Because high-durability parts are more practical to 3D-print in single color, the university logo was designed to be printed as two parts of blue plastic with small protrusions on their underside, which would be inserted into matching apertures in the white skull cover, then fixed in place by deformation via localized application of heat from the inner side of the skull cover in a manner similar to rivets.

4.4 Electrical Design: Robot Snake Brain 2.0 and 2.1

As the scope of the project drifted to include self-contained closed-loop control algorithms, it became necessary to integrate all sensors and actuators on the robot into a single control system. This was not possible with Robot Snake Brain 1.0; therefore, a new brain had to be designed to meet these objectives.

The ATSAMD21G18A microcontroller [3] was selected primarily for the availability of proven open-source hardware designs and bootloader firmware by Adafruit Industries [2]. Other advantages include higher speed, as well as greater capacity of SRAM and flash memory.

However, this microcontroller is not available in a DIP package. To use this MCU with breakout boards, the robot snake skull must be redesigned to be bigger, which would intensify power-to-weight challenges in the long run. To use this MCU without breakout boards, the microcontroller must be reflow-soldered, which would be a challenge in the short run. Due to steadily decreasing availability of modern integrated circuits in DIP packages, the option that offered reflow soldering experience was deemed more lucrative.

The electrical design of Robot Snake Brain 2.0 was an iterative process starting with breakout boards connected on a prototyping breadboard. A custom breakout board (see Figure 4.4.1) was designed and built for the ATSAMD21G18A MCU, based on the open-source Adafruit Feather M0 Basic Proto [2] and its bootloader firmware. Unlike the Adafruit Feather, this breakout board exposed all IMU pins that were not dedicated to specific support hardware and were not otherwise reserved.

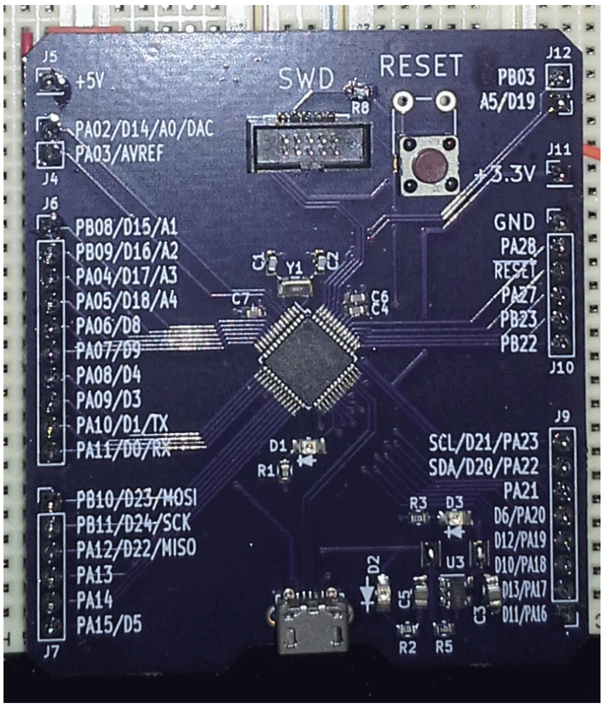


Figure 4.4.1: Top side of the ATSAMD21G18A breakout board.

The electrical design of Robot Snake Brain 2.0 was very similar to the electrical arrangement of breakout boards on the prototyping breadboard. However, lack of available space on the component-side of the PCB made it necessary to replace the SWD programming header (which is expected to be used only once) with solder pads on the opposite side of the PCB. For the same reason, fewer general-purpose I/O pins were exposed on the Robot Snake Brain 2.0 than on the breakout boards.

Robot Snake Brain 2.0 was designed in KiCAD as a two-layer circuit board, featuring the ATSAMD21G18A MCU running at 48 Mhz with 32 kB of SRAM and 256 kB of program flash memory, Bosch BNO055 IMU, NXP 9685 PWM generator, and the Digi XBee3 Micro. The top and bottom renders of Robot Snake Brain 2.0 are shown on Figure 4.4.2 and 4.4.3, respectively. A successfully assembled Robot Snake Brain 2.0 is shown on Figure 4.4.4.

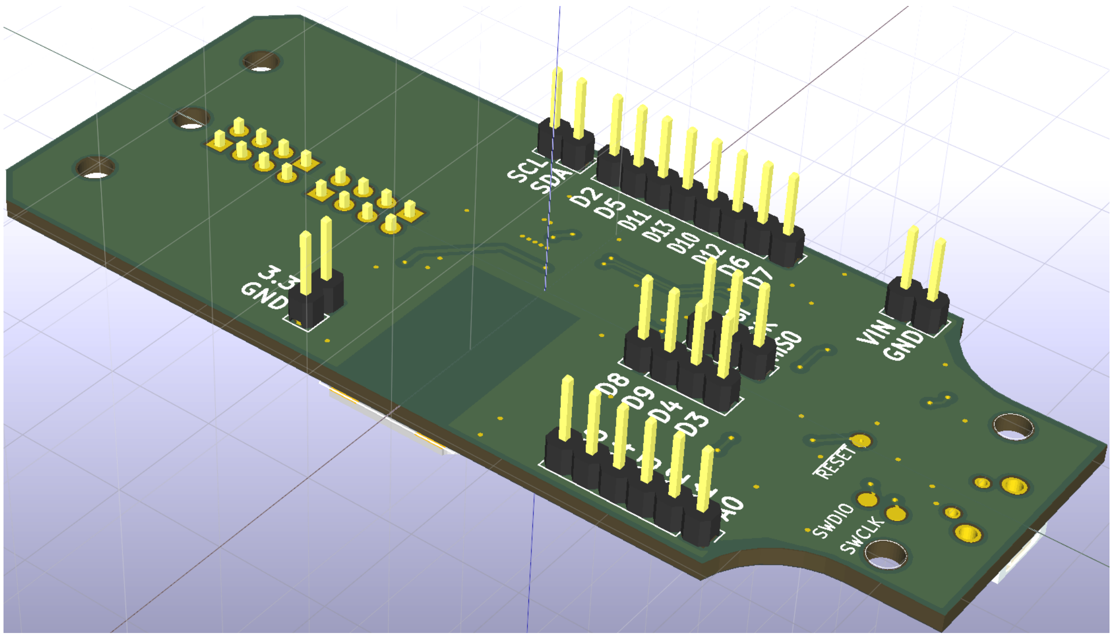


Figure 4.4.2: Top 3D render of Robot Snake Brain 2.0.

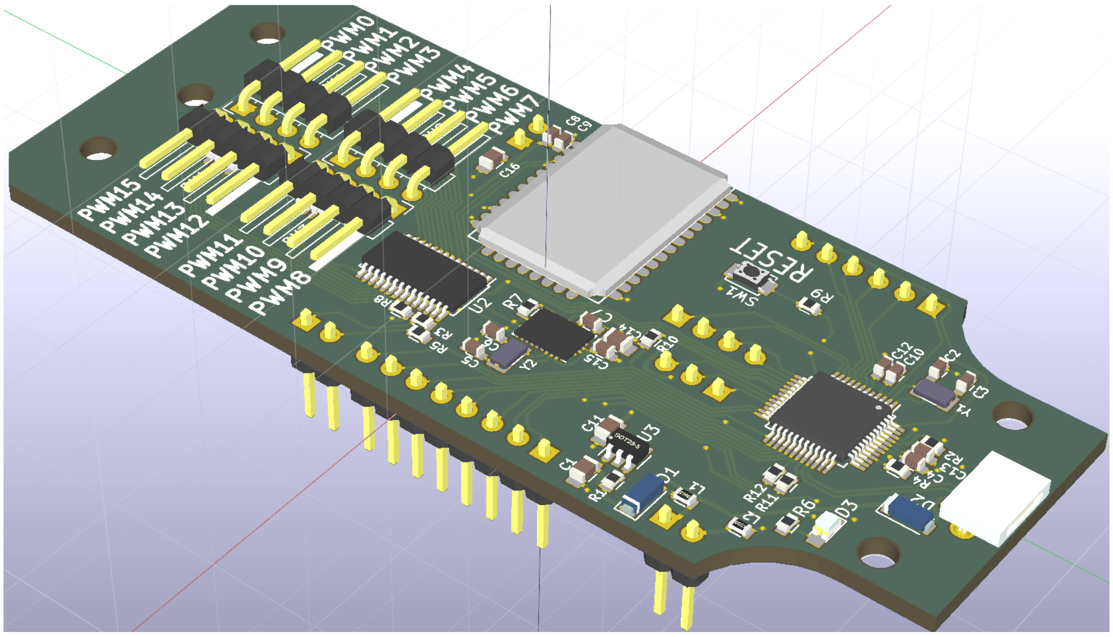


Figure 4.4.3: Bottom 3D render of Robot Snake Brain 2.0.

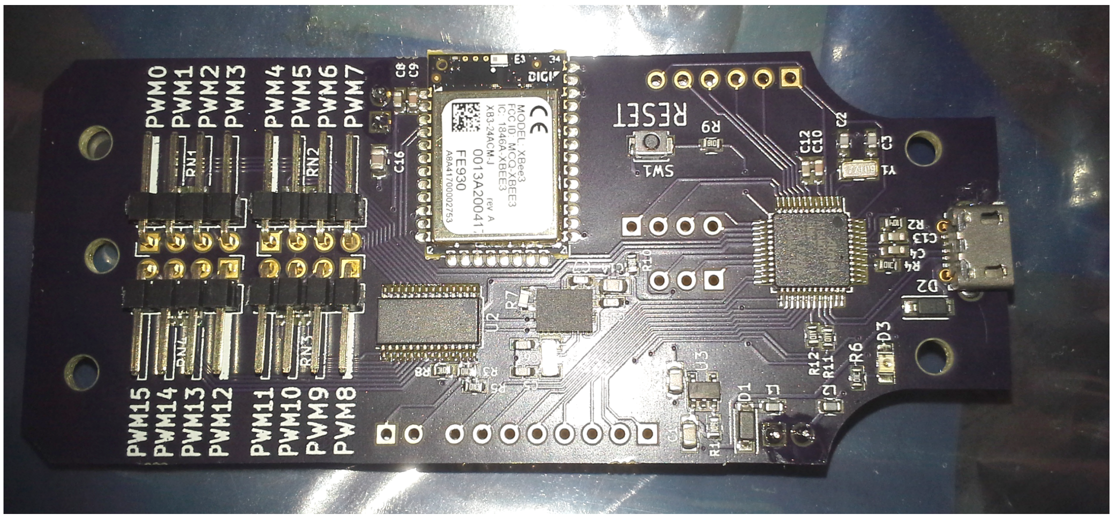


Figure 4.4.4: Bottom side of successfully assembled Robot Snake Brain 2.0.

It may appear counter-intuitive that all the SMD parts are mounted to what is referred to as the "bottom side" of the circuit board. However, there is a reason for this seemingly arbitrary choice: the circuit board is designed to be mounted in the bottom half of the robot snake's skull as shown in Figure 4.4.5. In this orientation, the side with all SMD parts is indeed facing down, and is therefore referred to as the bottom.

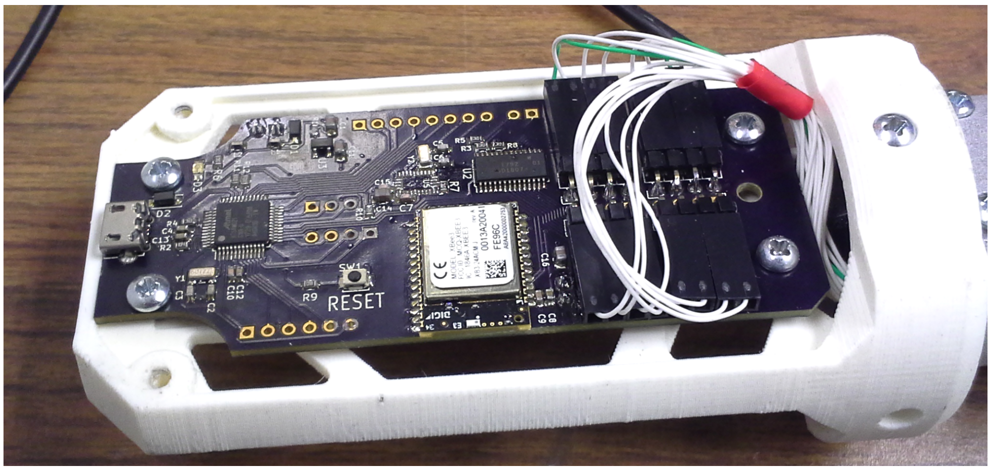


Figure 4.4.5: Bottom view of Robot Snake Brain 2.0, installed. (The IMU chip is absent because it was soldered unsuccessfully and had to be removed.)

Aside from SMD reflow difficulties and minor design flaws related to physical locations of I/O headers and the Reset button, the Robot Snake Brain 2.0 has demonstrated the desired electrical functionality in tests both outside and inside the robot snake's skull.

To facilitate more convenient programming, the bootloader firmware from the open-source Adafruit Feather M0 was re-used for the Robot Snake Brain 2.0. Therefore, when connected to a computer via USB cable, the Robot Snake Brain 2.0 is identified and treated exactly the same as the Adafruit Feather M0. This bypassed the necessity of soldering wires to the programming pads and completely replacing the robot's firmware every time it is programmed. Rather, the bootloader firmware needs to be programmed only once by means of specialized programming hardware (in this specific case, the Segger J-Link EDU Mini was used.) Once programmed with the bootloader firmware, the Robot Snake Brain 2.0 may be programmed through a simple USB cable, as the Adafruit Feather M0, by means of the Arduino IDE software.

The reflow soldering process, by means of which the Robot Snake Brain 2.0 was assembled, proved to be no less challenging than initially anticipated. Numerous problems have arisen in the efforts of reflow-soldering the Robot Snake Brain 2.0 circuit board.

Solder bridges were the most common failure mode. Solder bridges that occured on the TQFP package of the IMU IC and SOIC package of the PWM generator IC could be repaired manually with a soldering iron and some solder flux, although this carried the risk of damaging the IC by overheating it. The ATSAMD21G18A proved especially vulnerable to damage from overheating. From the experience of reflow-soldering the ATSAMD21G18A breakout boards, it was already known that the USB Micro connector was a common failure point, namely for solder bridging, due to narrow pitch between electrical pins.

A closer analysis of the Robot Snake Brain 2.0 circuit boards revealed that the solder resist mask was offset outward from copper pads by a distance of 0.2mm on each side. This is the default solder mask offset setting for KiCAD PCB design software. Because the distance between adjacent pins on the ATSAMD21G18A microcontroller's TQFP package and on the USB Micro B connector is only 0.25mm, which is less than twice the default solder mask offset, solder resist between adjacent pads under these components was entirely absent. It was inferred that the absense of solder resist between adjacent pins contributed to reflow soldering failure by solder bridging.

To address these problems, the design of the circuit board was modified slightly from Robot Snake Brain 2.0 to Robot Snake Brain 2.1, with the following measures taken to facilitate more reliable reflow soldering.

To address the problem of missing solder resist between adjacent narrow-pitch solder pads, solder mask clearance had to be adjusted in KiCAD's (PCB design software) settings. The nominal pitch between the closest of adjacent pins is 0.5mm, and the pins themselves appear to be approximately half of the pitch – that is, 0.25mm in width. It follows that the distance between adjacent pins is approximately 0.25mm. However, despite problems caused by excess solder mask clearance, some solder mask clearance is desirable to compensate for non-ideal alignment of copper and solder mask as a result of non-ideal PCB manufacturing process. As stated by our OSH Park, our supplier of raw circuit boards: *"Maximum soldermask expansion, retraction, or shift is 3mil"* [6] which is equal to 0.0762mm. It was decided to assign equal importance to both factors – the presence and clearance of solder mask – allocating half the space between adjacent pins (that is, 0.125mm) to each. Because solder mask clearance is measured outwards from both solder pads at the same time, solder mask clearance would have to be further reduced to one-half of that amount, or to one-quarter of the distance between adjacent pads. As such, solder mask clearance was reduced from 0.2mm to 0.0625mm. A close-up of the layouts of a few 0.5mm-pitch pins within the design of Robot Snake Brain 2.0 PCB and resulting 2.1 PCB, including the solder paste and solder mask, is shown on Figure 4.4.6 (a) and (b), respectively.

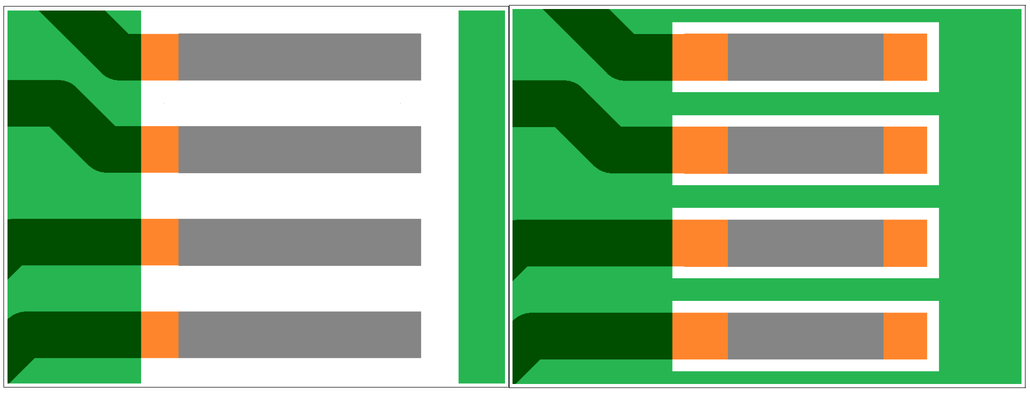


Figure 4.4.6: Close-up of circuit board layouts around microcontroller (pins 30-33) on **(a)** Robot Snake Brain 2.0 and on **(b)** Robot Snake Brain 2.1. Each pin is 0.25mm wide; distance between adjacent pins is also 0.25mm.

Legend:

- Dark-green: solder mask on copper (trace.)
- Light-green: solder mask on dielectric PCB substrate.
- Orange: exposed copper (solder pad.)
- Gray: solder paste on copper solder pad.
- White: exposed dielectric PCB substrate.

To reduce the risk of solder bridges, which proved to be the most common problem so far, the design of the solder paste stencil was modified in such a way as to reduce the volume of solder paste that would be applied to narrow-pitch pins. By default, each aperture in the solder paste stencil is identical in size and shape to the solder pad. It was theorized that, as surface tension pulls excess molten solder from the solder pad into a more spherical shape, molten solder assumes a shorter but wider shape than the solder pad, becoming wider than the 0.5mm pitch, thereby coming into contact with the molten solder on adjacent solder pads and causing solder bridging. FCT Assembly, Inc. asserts that the solution to this problem is to reduce the quantity of solder paste [7][8].

The objective for the redesign of the solder stencil was a reduction of deposited solder paste by an amount ranging from 25% to 50% of the original volume, according to the frequency of solder bridges encountered with prior designs. It was noted from Dervaes [8] and Smith [7] that the inner surface area of the aperture of the solder stencil should be minimized for optimal results. It follows that solder paste stencil apertures with rounded edges are inherently superior to rectangular-edged apertures of the same area. For this reason, solder stencil apertures corresponding to the solder pads of the USB connector were designed to be oval-shaped (or to be more precise, in a shape known as a Stadium.) However, because it was not certain as to whether oval pads would result in empirical improvement to soldering reliability, they were introduced only for the USB connector. For other pads in the design, the width of the solder paste stencil apertures was kept the same, while their length was reduced proportionally to the desired reduction of volume of solder paste to be deposited.

For improved robustness, a small ground fill was added around the USB connector, and a 100kΩ resistor (designated as R13 in the schematic) was added between the USB shield and the circuit board's ground. The mechanical drawing of the raw circuit board of Robot Snake Brain 2.1 may be found in Appendix A. The electrical schematic of Robot Snake Brain 2.1 may be found in Appendix B.

Three circuit boards of Robot Snake Brain 2.1 were assembled based on the resulting design, and exhibited no problems related to solder bridging.

The highlighted features of the two versions of the robot snake brain are compared on Table 4.4.1. Robot Snake Brain 2.0 has been omitted from this table because it is identical to Robot Snake Brain 2.1 in every way except for the resulting manufacturing yield. Manufacture of three boards of each (Robot Snake Brains 2.0 and Robot Snake Brain 2.1) was attempted, of which only one Robot Snake Brain 2.0 turned out fully functional, whereas all three Robot Snake Brains 2.1 were reflow-soldered successfully.

Table 4.4.1. Comparison between robot snake brains.

Brain Version	**Snake Brain 1.0**	**Snake Brain 2.1**
MCU IC	ATMega328	ATSAMD21G18A
MCU Core	8-bit AVR	32-bit ARM Cortex M0
MCU Clock	20 MHz	48 MHz
Program Flash	32 kB	256 kB
SRAM	2 kB	32 kB
Assembly Strategy	Interconnected breakout boards and other modules, fully manual soldering.	Custom PCB, manual pick-and-place, reflow SMD soldering.

4.5 Electrical Design: Robot Snake Interconnection

The design of the peripheral electrical interconnection along the length of the robot snake's body was constrained by mechanical parts which were designed prior to the robot's electrical system. Specifically, all wires that were routed past the robot's motorized X-Y joints had to pass through either of two 6-mm-wide holes, which placed a bottleneck on the width and thickness of wires to be routed lengthwise along the robot. This constraint limited the number of thick power wires to two, while the thinnest (*c.* 30 AWG) of wire stocked in the lab was utilized for low-current PWM signals. The power lines and PWM signals were adapted to standard RC servo connectors by means of small custom PCBs, which we dubbed 'segment nodes.' One segment node can route PWM signals to up to four servos. As such, one segment node is located within every other connector tube between motorized X-Y joints. The electrical schematic of the segment node may be found in Appendix B.

The prototype described in this paper is powered from an external power supply through a wire requiring several Amps (depending on mechanical effort) of current at 5 Volts. In the first version of the robot snake brain, the ATMega328 microcontroller is rated up to 6.0V [4], so the external power supply was set to 5.2V to offset the voltage drop in the wire at times of high current draw. In Robot Snake Brain 2.0 and 2.1, the ATSAMD21G18A microcontroller is rated only up to 3.8V [3], but is powered through the Exar SPX3819 linear voltage regulator rated for up to 16V [5], which is above the upper limit of the servo motors' voltage range - 6.0V.

Although current losses can be reduced by transmitting a higher voltage over thinner wire, this requires the addition of at least one switching step-down DC-DC converter powerful enough to handle the high current draw. This design option was considered, including the possibility of adding several step-down converters (capable of supplying roughly one Amp each,) one in place of each segment node, *vs.* adding one single step-down converter (capable of supplying roughly a dozen Amps) in one place within the robot. These design options were rejected in favor of a simple thick wire because it was understood that the robot would eventually have to be battery powered with 3.7 V Lithium-Polymer cells, and its power system would have to be redesigned anyway.

4.6 Electrical Design: Remote Control Unit

The remote control unit consists of two separate systems: the Remote Command Unit (RCU) and the remote feedback unit. These are electrically separate, although a linear voltage regulator within the RCU (a 7805 in a TO-220 package) allows it to be powered from an external power supply at any voltage between 6V and 28V as a backup power supply in case of LiPo battery failure. The 7805 could be used to power the remote command unit from the remote feedback unit's power supply and omit the LiPo battery management module.

Electrically, the Remote Command Unit consists of an Arduino Pro Mini [41], a PowerBoost500 LiPo battery management module [42], a 3.7 Volt Lithium Polymer Ion battery, a matrix keypad, and a 2D analog joystick, as well as the aforementioned 7805 voltage regulator, all interconnected by a custom circuit board shown on Figure 4.6.1. The electrical schematic for the RCU is found in Appendix B.

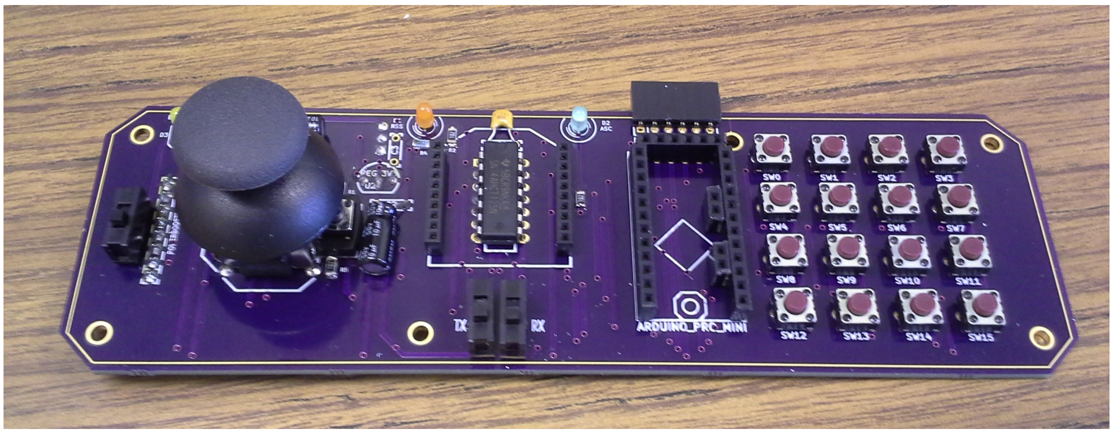


Figure 4.6.1: Custom circuit board of Remote Command Unit. Only soldered components, including headers for Arduino and XBee modules, are visible.

4.7 Algorithm Design: Remote Command Protocol

The command packet is 48 bits long. The first octet is the address header, the next 32 bits are the payload data, and the last octet is the check-sum. The address header determines the location in the parameter vector to which the 32-bit data is to be written. The purpose of the eight-bit check-sum is to verify the integrity of the data. It must be admitted that, due to the queue-like serial buffer structure, there is a 1/256 chance accepting a false packet because its check-sum was correct by accident. An example of a valid command packet is shown below, in which every hexadecimal digit represents four bits:

Address		**Data**								**Check-Sum**	
0	5	3	F	8	0	0	0	0	0	3	C

In this example, the validity of the command packet is first verified by adding the six octets of data together and making sure that their eight-bit sum is zero:

$$0x05+ 0x3F+ 0x80+ 0x00+ 0x00+ 0x3C=0x00$$

If the validity of the check-sum is verified, and if the address is valid, the four-byte payload data is written to the address (in this case, 0x05) of the '*Parameter Vector*,' an array allocated specifically for the purpose of receiving and storing remote commands from the command transmitter. This array consists of eleven 32-bit words; each word is the same length as the payload data of a command packet. For the current firmware version, the command in the above example sets the propagation paramter to a floating-point value of 1.0, which is responsible for the propagation of the motion pattern along the array of motorized X-Y joints.

4.8 Algorithm Design: Robot Snake

The firmware of this robot may be considered as one serial-input-multiple-output block on a signal-flow graph of the system as a whole. The input is a serial channel from the wireless XBee modem, and the output consists of fourteen PWM signals (two for each X-Y motorized joint,) plus one simple digital logic signal. The microcontroller in the robot snake's head receives serial commands from the remote command unit through a wireless XBee link, validates and decodes them, then updates its parameter vector accordingly. The parameter vector is used in continuous generation of a pattern of PWM signals which are sent to the servos. Because information flows through this system in only one direction, from the input to the outputs, it may be conveniently divided into smaller blocks as illustrated on Figure 4.8.1. This section focuses examines the theoretical and practical aspects of the firmware functionality by means of a detailed look within each of these blocks.

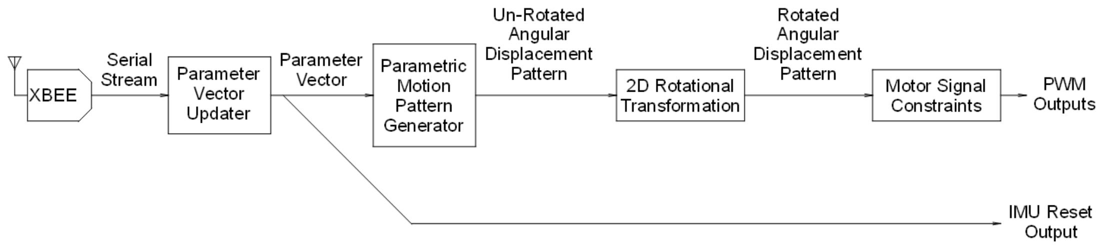

XBEE
Serial Stream
Parameter Vector Updater
Parameter Vector
Parametric Motion Pattern Generator
Un-Rotated Angular Displacement Pattern
2D Rotational Transformation
Rotated Angular Displacement Pattern
Motor Signal Constraints
PWM Outputs
IMU Reset Output

Figure 4.8.1: Simplified block diagram of robot snake firmware.

The first block, the Parameter Vector Updater, is illustrated on Figure 4.8.2. The procedure of integrity checks and de-multiplexing of incoming data packets is described in detail in Section 4.7. It is immediately noticeable that there are two, rather than one, parameter vectors, connected through Slew-Rate Limit (SRL) blocks. The second parameter vector (the one to the right) works as a non-linear Low-Pass Filter (LPF) and serves the purpose of smoothing out the changes of the parameter vector's values. Due to the serial nature of the commands received through the wireless link, values within the first parameter vector are updated incrementally rather than continuously. Prior to introduction of the SRL LPF, the physical motion of the robot was updated at wide increments, causing all servos to try to move quickly from their previous positions to their new positions at the same time. Simultaneous activation of fourteen high-power servos created current spikes which created conditions for brown-outs, and the resulting sporadic motion needlessly increased the stress on mechanical components.

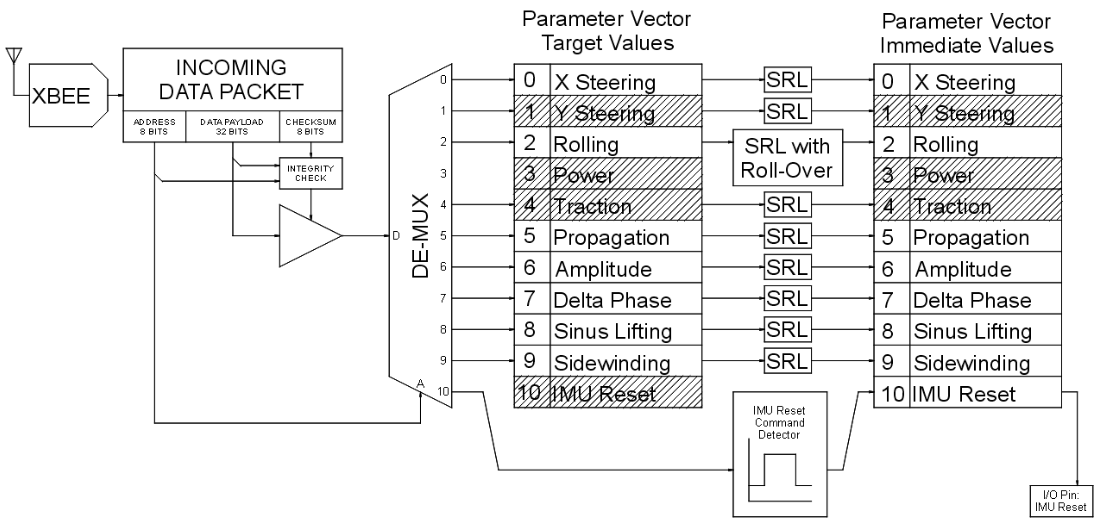


Figure 4.8.2: Flow of data from the incoming wireless signal to the parameter vector. It must be noted that, with the exception of the XBee and the I/O pin, all constructs on this diagram exist only in software.

One may also notice that some of the elements of the parameter vector are shaded. This is to indicate that, although these elements still technically exist in firmware, they are now vestigial, and no longer serve any practical purpose. The *Y Steering* element was introduced to provide a way for the robot snake to either lift or lower its head and tail with respect to the middle of its body, which would allow it to traverse uneven terrain with greater efficiency, and even to climb over low obstacles. While this functionality was written within the firmware, it has not been tested in practice, and the remote command unit has no user input programmed to invoke it. Considering the robot's current level of performance, it seems reasonable to hypothesize that the robot would require a greater power-to-weight ratio for such functionality to be truly effective.

The intent for the *Power* and *Traction* blocks was to provide a three-way trade-off between power consumption, traction against terrain, and speed of locomotion. It was hypothesized that such a trade-off exists, involving the range of angular displacement of motorized joints and the rate of propagation of the periodic motion pattern along the robot snake's motorized joints. Higher traction could supposedly be achieved by increasing the range of angular displacement of motorized joints, which would cause the lateral undulation of the robot snake's gait to become wider. All other things being equal, this would also increase the servo motors' power consumption. Similarly, higher speed of locomotion could be achieved by increasing the speed of propagation of the periodic motion pattern at the price of increased power consumption. This three-way-trade-off approach was abandoned as focus shifted toward other aspects of the project.

The *IMU Reset* element is distinguished by not having any impact upon the robot's motion. Its purpose is, upon receiving a valid IMU reset command, to send a reset signal to the on-board Inertial Measurement Unit (IMU) without powering down the rest of the robot. The state of the *IMU Reset* element directly affects the state of the *IMU Reset* output pin on the microcontroller, which is electrically connected to the *Reset* input pin on the IMU.

The *Rolling* element is also unique, because the value it contains is an angle, which intentionally *'wraps around'* from an angle of $+\pi$ radians to an angle of $-\pi$ radians (or from $-\pi$ to $+\pi$) to allow the robot to roll continuously in either direction with no artificial angular limit.

The next block in the firmware is the Parametric Motion Pattern Generator (PMPG) which is illustrated in greater detail on Figure 4.8.3.

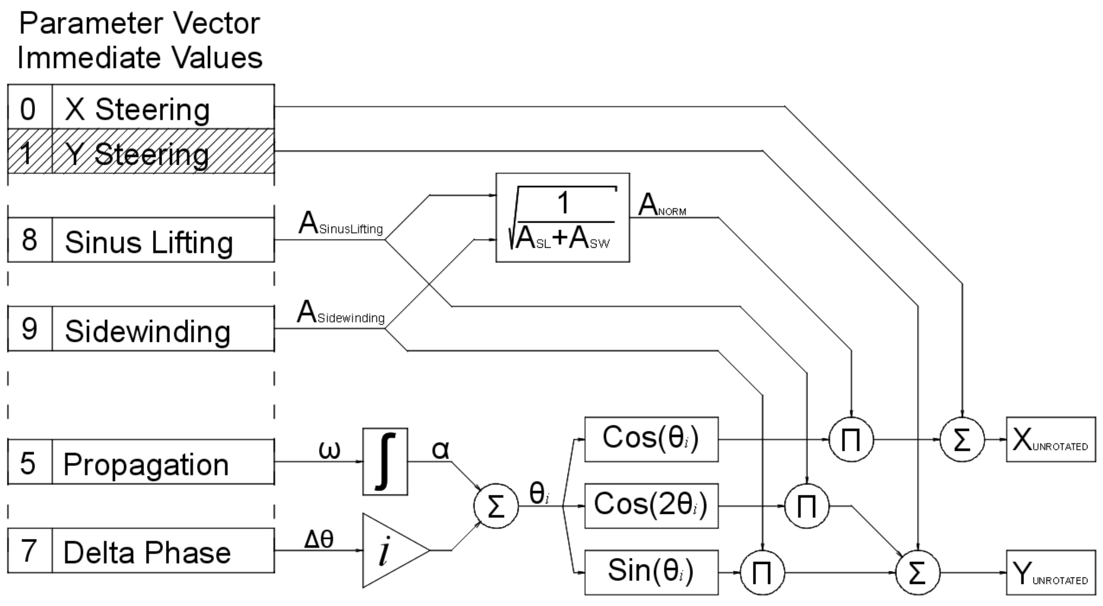


Figure 4.8.3: Parametric Motion Pattern Generator (PMPG.)

The PMPG drives all seven X-Y motorized joints identically; that only differs in the value of the index i of the joint, starting with 0 at the robot snake's head and ending with 6 at the last one near the robot snake's tail. This index creates the phase offset necessary for each X-Y joint to act according to its position on the robot snake's body. The phase offset is the product of the index i and the *Delta Phase* element of the parameter vector. The time-varying phase angle of propagation *α* is obtained by integrating the *Propagation* element of the parameter vector. While the *Propagation* element is zero, *α* remains constant and the motion pattern does not propagate in either direction along the body of the robot snake. When the *Propagation* element is positive, the motion pattern propagates forward along the body of the robot snake.

Note that the direction of propagation of the motion pattern does not necessarily correspond, positively or negatively, to the direction of locomotion of the robot. The actual direction of locomotion depends as much on other aspects of the motion pattern as it depends upon the motion pattern's direction of propagation. For instance, the elements *Sinus Lifting* and *Sidewinding* of the parameter vector may be varied to propel the robot either forward, or backwards, or sideways, even while the *Propagation* element remains constant (but non-zero.)

The three trigonometric functions at the heart of the robot snake's motion pattern generator are all functions of angle θ_i, which is the sum of the phase angle of propagation α and the phase offset $\Delta\theta \bullet i$. The cosine of this angle is responsible for lateral undulation and drives horizontal motion of the motorized joints, which may be used to propel the robot forward or backward if the coefficient of friction along the direction of its body is lower than the coefficient of friction perpendicular to the direction of its body (as demonstrated with freely rotating passive wheels on the underside of the robot.) The cosine of double of this angle is responsible for sinus lifting, which is useful on soft terrain, to propel the robot in the direction opposite of the direction of propagation of the motion pattern; or on hard terrain, to propel the robot in the direction of propagation of the motion pattern. Finally, the sine of that angle may be used to propel the robot sideways by means of a sidewinding gait. To ensure that the motion pattern is not distorted by clipping against the servo motors' range limits, a normalizing factor is applied to the amplitude of lateral undulation. This ensures that the Pythagorean sum of the vertical and horizontal amplitudes does not exceed unity. Steering offsets, which are applied afterwards, have the potential to distort the robot's motion pattern, but were deemed essential for effective navigation.

After the horizontal and vertical components of the motion pattern are computed, they are mapped to the local X and Y directions of the robot snake's motorized joints by means of a rotational transformation [9]. This is also the part that allows the robot to roll sideways.

The rotation transformation is accomplished via matrix multiplication:

$$\begin{bmatrix} X_{ROTATED} \\ Y_{ROTATED} \end{bmatrix} = \begin{bmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{bmatrix} \times \begin{bmatrix} X_{UNROTATED} \\ Y_{UNROTATED} \end{bmatrix}$$

Where φ is the Rolling element of the parameter vector.

By cycling the rolling angle all the way around, the robot snake can maintain a stable, flattened shape against terrain, while simultaneously rolling sideways.

It must be admitted that, because the rotational matrix was entered by memory as the firmware was written, the firmware was actually written with the transpose of the rotation matrix listed above. That also turned out to be the inverse of the rotation matrix, which caused the robot to roll in the opposite direction. Because the algorithm appeared to work properly during testing, this error was not detected until much later.

The final step in the flow of the block diagram is to scale the values of servo motor command signals and constrain them to a realistic range. This process is illustrated on Figure 4.8.4.

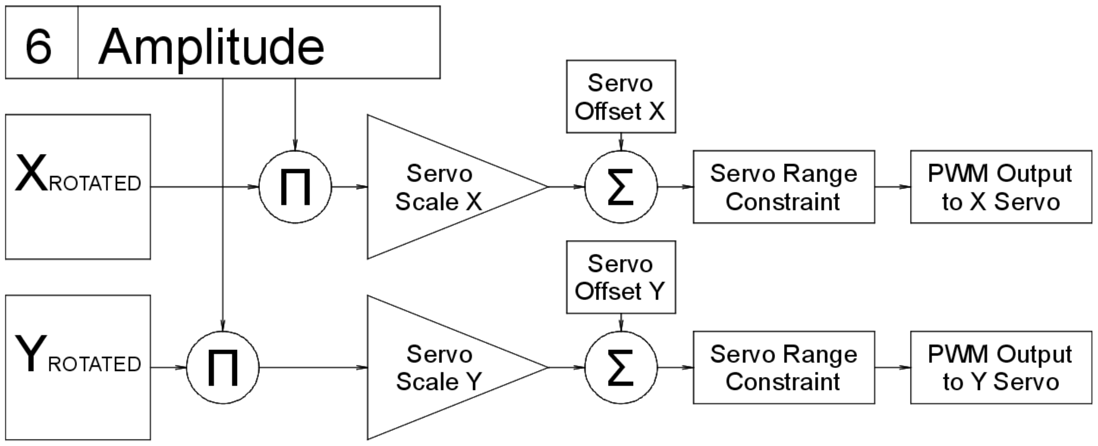


Figure 4.8.4: Motion control output stage.

The signal for each servo motor is scaled by the *Amplitude* element of the parameter vector, which is used here to conveniently bring the robot gradually out from a straight-line shape before operating, and to straighten it after operations terminate. Next, multiplicative and additive constants are applied to each motor, which provides the opportunity to compensate for internal differences between servo motors. Finally, to protect against programming errors, the servo output signals are constrained to a range deemed safe for the mechanism. The values for all fourteen servos are sent through an I2C link to a PWM control board, which updates the PWM signals sent to the servo motors.

The source code for the robot snake's firmware is found in Appendix C.

4.9 Algorithm Design: Remote Command Unit

The Remote Command Unit (RCU) is an open-loop controller and does not allow two-directional communication with the robot snake. Although physically capable of receiving serial data through the XBee, the RCU is programmed to only send motion control parameter commands.

The source code for the remote command unit firmware may be found in Appendix C. The firmware design is fairly simple: the Arduino (an Atmel ATMega328 microcontroller with an Arduino bootloader) runs a loop, in which it reads the two-dimensional analog signal from the joystick using its built-in Analog-to-Digital Converter (ADC) and scans a 4x4 matrix keypad. From these inputs, the Remote Command Unit determines which motion parameters to update, incrementing or decrementing specific motion parameters constrained to a minimum value, maximum value, and a maximum absolute rate of change. The maximum rate of change is utilized to allow the robot to gradually shift between motion parameters, which makes its motion more smooth and reduces wear on mechanical parts.

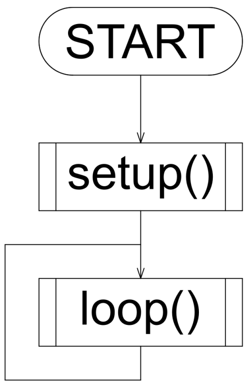


Figure 4.9.1: General structure of Arduino programs.

All Arduino programs are structured as the flowchart shown on Figure 4.9.1. Arduino programs in general are divided into two parts; setup() executes once at the beginning of the program, and then loop() executes again and again as long as the program runs. The flowchart for the setup() section of RCU firmware program is shown on Figure 4.9.2; its only purpose is to initialize the ATMega328 and tell the robot snake to assume default motion parameter values.

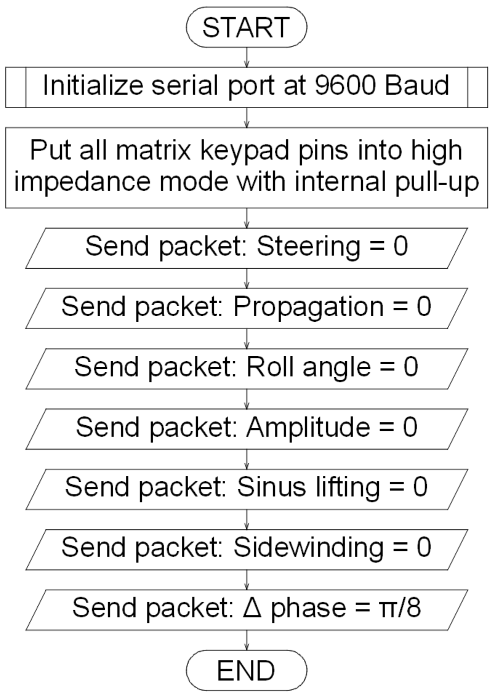

START
Initialize serial port at 9600 Baud
Put all matrix keypad pins into high impedance mode with internal pull-up
Send packet: Steering = 0
Send packet: Propagation = 0
Send packet: Roll angle = 0
Send packet: Amplitude = 0
Send packet: Sinus lifting = 0
Send packet: Sidewinding = 0
Send packet: Δ phase = π/8
END

Figure 4.9.2: Flowchart for the setup() chunk of RCU Arduino program.

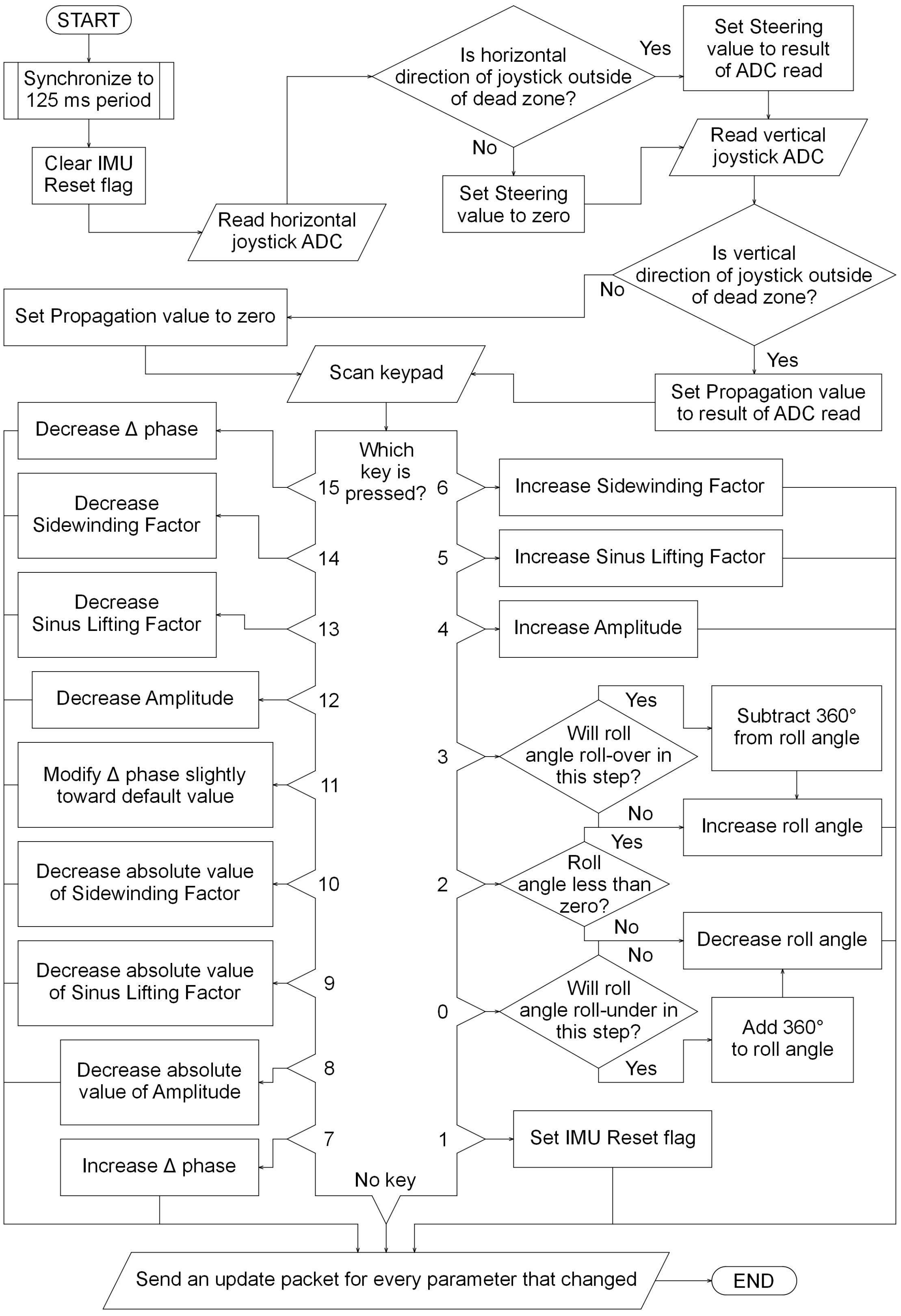

START
Synchronize to 125 ms period
Clear IMU Reset flag
Read horizontal joystick ADC
Is horizontal direction of joystick outside of dead zone?
Yes
No
Set Steering value to result of ADC read
Set Steering value to zero
Read vertical joystick ADC
Is vertical direction of joystick outside of dead zone?
No
Yes
Set Propagation value to zero
Set Propagation value to result of ADC read
Scan keypad
Which key is pressed?
15
14
13
12
11
10
9
8
7
6
5
4
3
2
0
1
No key
Decrease Δ phase
Decrease Sidewinding Factor
Decrease Sinus Lifting Factor
Decrease Amplitude
Modify Δ phase slightly toward default value
Decrease absolute value of Sidewinding Factor
Decrease absolute value of Sinus Lifting Factor
Decrease absolute value of Amplitude
Increase Δ phase
Increase Sidewinding Factor
Increase Sinus Lifting Factor
Increase Amplitude
Will roll angle roll-over in this step?
Yes
No
Subtract 360° from roll angle
Increase roll angle
Roll angle less than zero?
Yes
No
Decrease roll angle
Will roll angle roll-under in this step?
No
Yes
Add 360° to roll angle
Set IMU Reset flag
Send an update packet for every parameter that changed
END

Figure 4.9.3: The loop() section of RCU firmware program.

The loop() section of the RCU firmware program is more involved and only a simplified flowchart is shown on Figure 4.9.3 for clarity and brevity. The last element, labeled “END,” is not the end of the program, but only the end of the immediate iteration of the main loop. Once this element is reached, the process represented by this flowchart repeats from the “START” element. The widest element of this flowchart is the switch/case structure, which responds to the keys of the matrix keypad. For the sake of completeness, the enumeration of keys on the keypad is illustrated on Figure 4.9.4.

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Figure 4.9.4: Enumeration of keys on the matrix keypad.

The resulting key/command associations are shown on Figure 4.9.5. Inputs that directly influence locomotion are in green, inputs that influence locomotion indirectly are in blue, RCU system inputs are in red, and system feedback is in yellow.

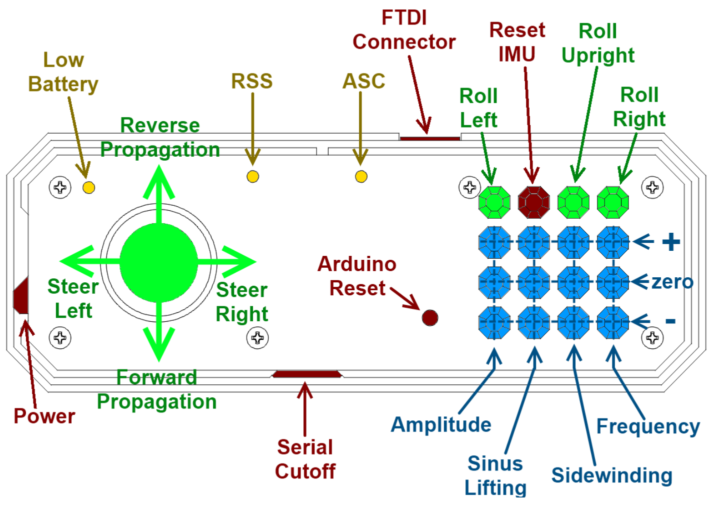


Figure 4.9.5: RCU key/command associations.

Because the user interface (UI) on the RCU is not physically labeled, a brief outline of the UI is described here.

System Inputs

- <u>Power</u>: the Acrylic lever on the left side of the device may be used to turn the RCU on or off. The lever is moved up to turn it on and down to turn it off. The Arduino may still be programmed via the Adafruit FTDI cable even if the rest of the device is turned off, which is capable of supplying enough power for programming.
- <u>Serial Cutoff</u>: the Acrylic button on the bottom side of the RCU may be pulled out (downward) to cut off the serial connection between the XBee and the Arduino, which is necessary to program the Arduino without disassembling the device. During normal operation, the Serial Cutoff button is pushed in.
- <u>FTDI Connector</u>: the single-row, six-pin, 2.54-mm-pitch female connector on the top side of the RCU is designed for use with the Adafruit FTDI cable for establishing a serial connection with the onboard Arduino. Note that a single-row, six-pin, 2.54-mm-pitch male-to-male adapter is required. The choice of a female connector is a precaution to avoid exposing the microcontroller pins unless the device needs to be programmed.
- <u>Arduino Reset</u>: a round aperture in the front surface of the RCU provides access to the Arduino's reset button. This is rarely (if ever) used, because it is faster and easier to reset the Arduino by turning the RCU off and then on, but is present in case the need to actually use it

ever arises.

- IMU Reset: the button in the first row, second column of the keypad causes the RCU to command the robot to reset its Inertial Measurement Unit.

System Feedback

- Low Battery LED: The LED in the upper-left corner of the RCU lights up when the device's battery is in need of recharging.
- RSS: The LED above and to the left of the XBee wireless modem is driven by a feedback signal from the XBee; it blinks whenever the XBee receives a wireless signal and attempts to pass it to the Arduino.
- ASC: The LED above and to the right of the XBee wireless modem is driven by a feedback signal from the XBee; it blinks whenever the XBee receives a serial signal from the Arduino and attempts to transmit it wirelessly to the robot snake. *Note: It is normal for this LED to always blink during normal operation even if the controls are left untouched, because Analog-to-Digital Converter (ADC) noise causes the least significant bits of the joystick input to change frequently enough that Steering and Propagation parameters are updated almost every 125 milliseconds.*

Direct Control Inputs

- Analog Joystick: The analog joystick on the left side of the front surface of the RCU controls the Propagation and Steering parameters.

  Pushing the joystick upward causes the robot snake to propagate its motion pattern backwards along its body, which corresponds to forward motion of the robot when the serpentine locomotion gait is utilized. Similarly, pushing the joystick downward causes the robot snake to propagate its motion pattern forwards along its body, which causes the robot to move in reverse if it is using serpentine locomotion. Note that pushing the joystick up or down does not necessarily correspond to the same forward/reverse motion if different gaits of locomotion are used; it only corresponds to the direction opposite of the propagation of the motion pattern along the body of the robot. For instance, if the joystick was pushed up or down during the sidewinding gait, the robot moves left or right rather than backward or forward.

Pushing the joystick left or right causes the robot snake to apply a sideways curvature offset to its motion pattern. During serpentine locomotion, this is useful as a form of steering. During other modes of locomotion, it still applies a sideways curvature offset but its use as a form of steering would be less effective.

- Roll Left: The button in the upper-left corner of the matrix keypad gradually decreases the Roll motion parameter, which causes the robot to gradually change the angle of X-Y mapping of its motion pattern to servo motors. The tangible effect of this change is the left-rolling motion of the robot.
- Roll Right: The button in the upper-right corner of the matrix keypad causes a qualitatively similar, quantitatively opposite effect, as compared to the Roll Left button.

- Roll Upright: Depending on the present state of the Roll motion parameter, the button in the first row, third column of the matrix keypad may have the same effect as the Roll Left button if the value of the Roll parameter is greater than zero, or it may act as the Roll Right button while the value of the Roll parameter is less than zero, or it may do nothing if the value of the Roll parameter is zero. This provides a convenient way to reset the robot's Roll motion parameter to zero.

Indirect Control Inputs

The second, third, and fourth rows of the matrix keypad are responsible for adjusting the amplitude, sinus lifting, sidewinding, and delta-phase motion parameters. Each column corresponds to one parameter. The second row increases each parameter up to some maximum, the fourth row decreases each parameter down to some minimum, and the third row 'zeroes-out' each parameter; *i.e.*, changes each parameter toward its default value.

4.10 Structural Design: Remote Command Unit

The remote command unit consists of a printed circuit board sandwiched by four layers of clear 3mm-thick Acrylic plastic on each side as shown on Figure 4.10.1. Some electrical parts, including all user interface elements such as the joystick and sixteen buttons, are soldered directly to the circuit board. However, the more complex elements – the microcontroller (Arduino Pro Mini[41], 5V, with an Atmel ATMega328 at its core,) radio modem (XBee 802.15.4,) and power management board [42] were purchased as they are and connected via 0.1” headers to the main circuit board.

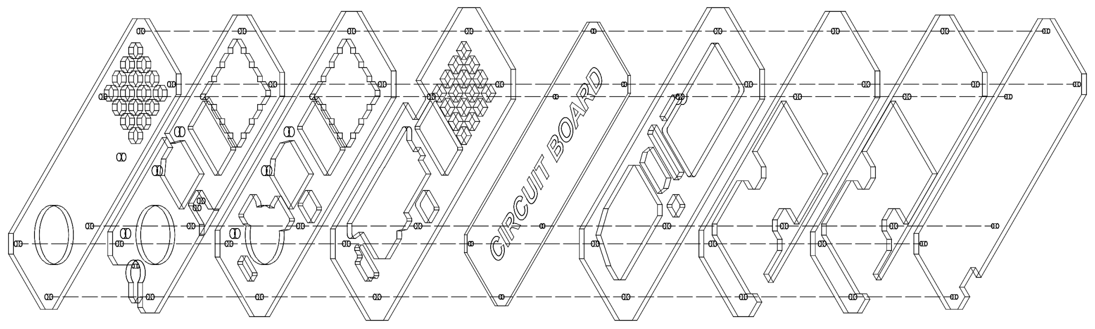


Figure 4.10.1: Remote Command Unit assembly.

4.11 Virtualization of Remote Command Unit

While the physical implementation of the Remote Command Unit has proven effective as a self-contained device for sending radio commands to the robot snake, it has one obvious disadvantage: it has to be built first.

To counter this disadvantage, an alternative means for sending radio commands to the robot snake was devised. The Virtual Remote Command Unit (VRCU) is an emulation of its physical counterpart, written in Java using the Processing 3.3.7 Integrated Development Environment (IDE.) The result is an application that can run on the Windows (and possibly Linux, but this was not tested) operating system, using only a ubiquitous Windows workstation and a USB-to-XBee adapter as hardware. A screenshot of the application window running on Microsoft Windows 7 operating system is shown on Figure 4.11.1. The application responds to both keyboard and mouse input, and allows the selection of a USB serial port through which to transmit command packets. Its source code may be found in Appendix C.

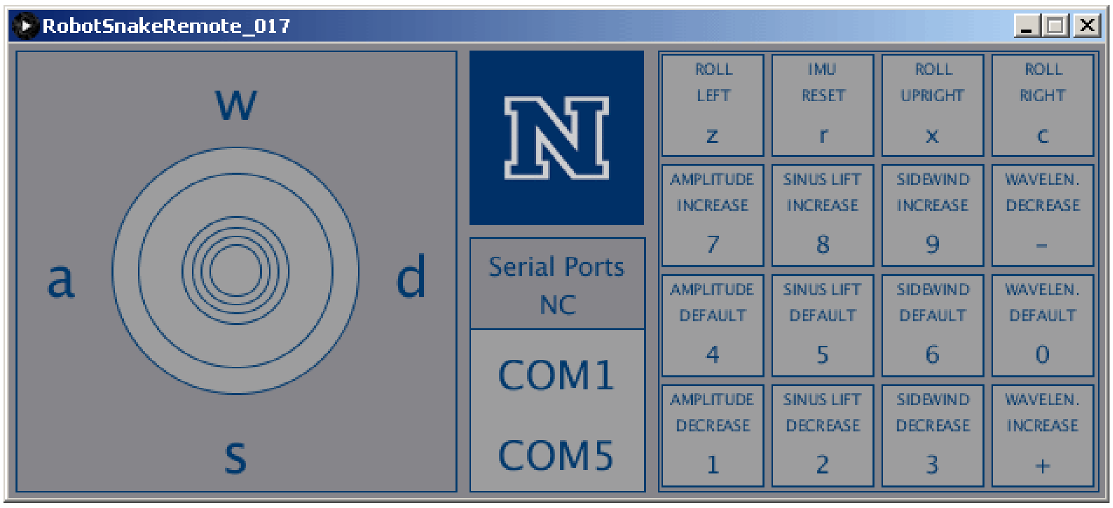


Figure 4.11.1: Virtual Remote Command Unit application window.

4.12 Data Feedback System

The feedback system is not part of this work, and is therefore covered only briefly in this thesis. It consists of two separate parts: an IMU subsystem which measures angular orientation, and the electronics from a Hubsan drone re-purposed to send a live video signal. The IMU is the BNO055 breakout board by Adafruit Industries [45], which uses a BNO055 IMU chip developed by Bosch. It was found that the BNO055 chip could not share the I2C line with the PWM control board, so it was moved to a different Arduino module. The electronics from the Hubsan drone send their video signal to the electronics from the remote control of the Hubsan drone. These were incorporated into a custom enclosure consisting of laser-cut Acrylic plastic and became known as the Video Feedback Unit (VFU.) The VFU was structurally affixed to the Remote Command Unit as shown on Figure 4.12.1. Unfortunately, the video signal was unreliable, and the robot snake's skull was too small to accommodate the electronics from the Hubsan drone. Therefore, the Video Feedback Unit was not used with later prototypes.

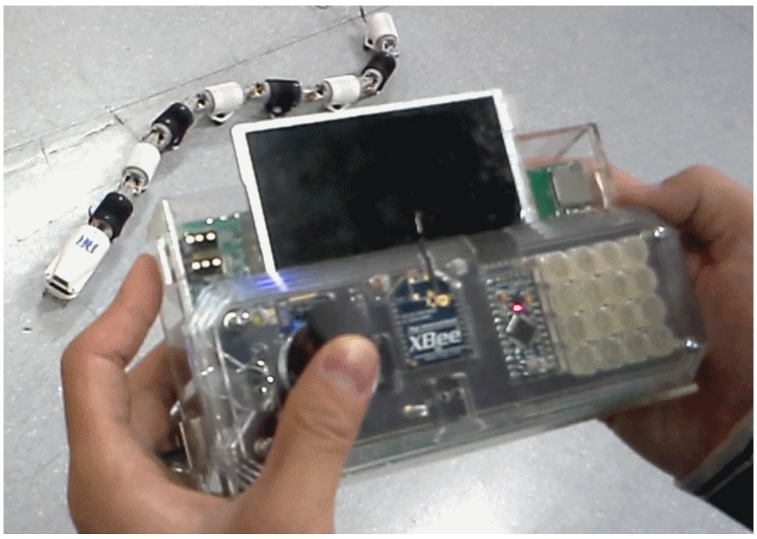


Figure 4.12.1: Remote Command Unit and (presently inactive) VFU.

Chapter 5

Results

A number of functional robot snake prototypes and a remote command unit were constructed. This chapter summarizes our accomplishments in the course of this project, and is broken down into sections that cover individual sub-systems, including the mechanical hardware of the robot, electronic hardware of the robot, and hardware of the handheld user interface device.

5.1 Robot Snake Prototype I

The first prototype, designed by students previously involved in this project, set the structural and electromechanical standards for the prototypes that would follow. Although the first prototype did not demonstrate locomotion, it served as a starting point for the mechanical design iterations and defined the design constraints that the next prototypes would be built to accommodate.

5.2 Robot Snake Prototype II

The gear train was redesigned (as shown on Figure 5.2.1) for increased torque at the expense of a narrower theoretical range of motion – a negligible expense, considering the fact that most of the original gear train's theoretical range of motion was in practice inaccessible due to the first prototype's inherent geometric constraints. As shown in Figure 5.2.2, the second mechanical design can generate sufficient torque to demonstrate a satisfactory emulation of some, but not all, snake-like gaits of locomotion. Measured by means of video tracking, our robot snake was able to achieve a velocity of 0.253 m/s [33]. The robot snake has also demonstrated the ability to roll over from the upside-down position to the rightside-up position, as shown on Figure 5.2.3. This allows the robot snake to switch between the anisotropic mode of friction against terrain offered by its passive wheels and the isotropic mode of friction offered by the surfaces of its structural components.

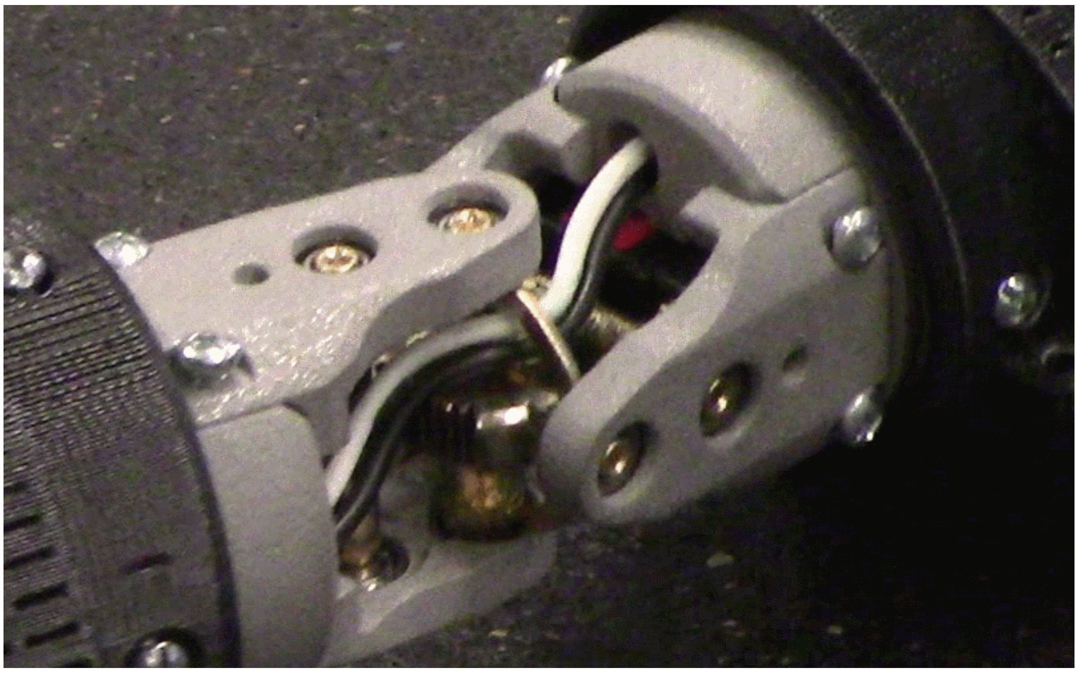


Figure 5.2.1: Close-up of a motorized joint on Robot Snake Prototype II.

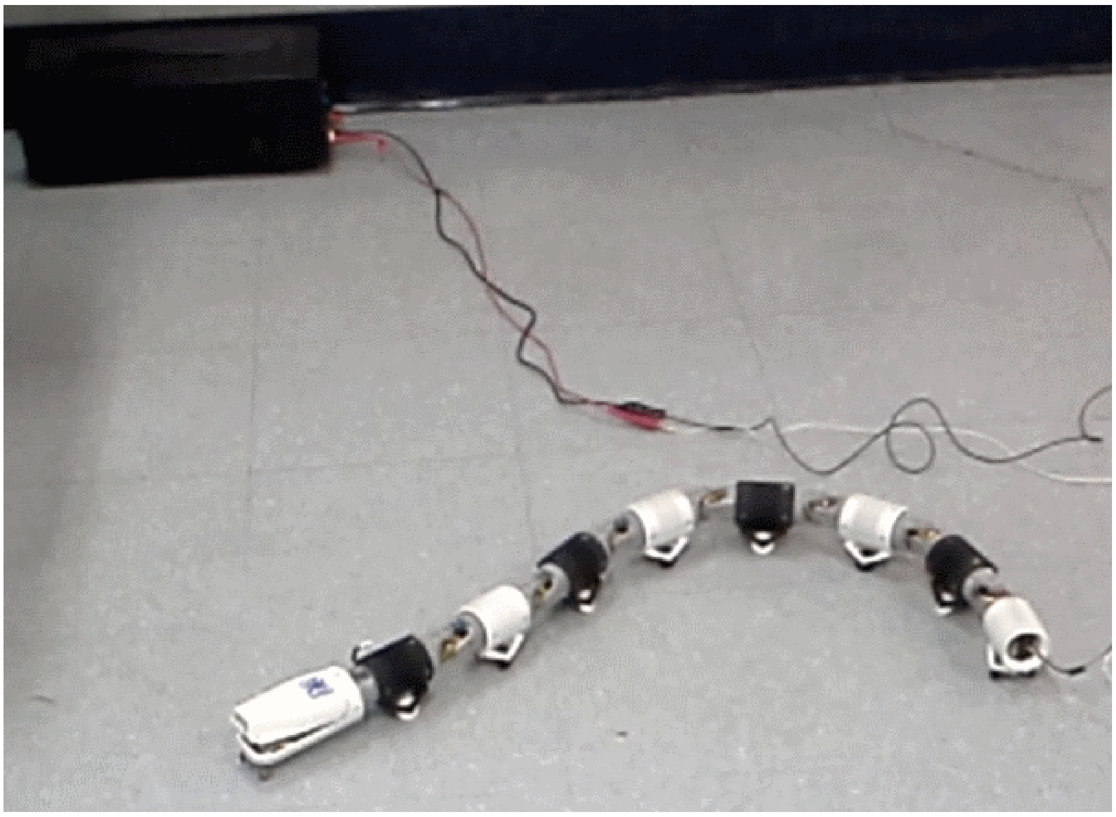


Figure 5.2.2: Robot snake demonstrates locomotion and steering.

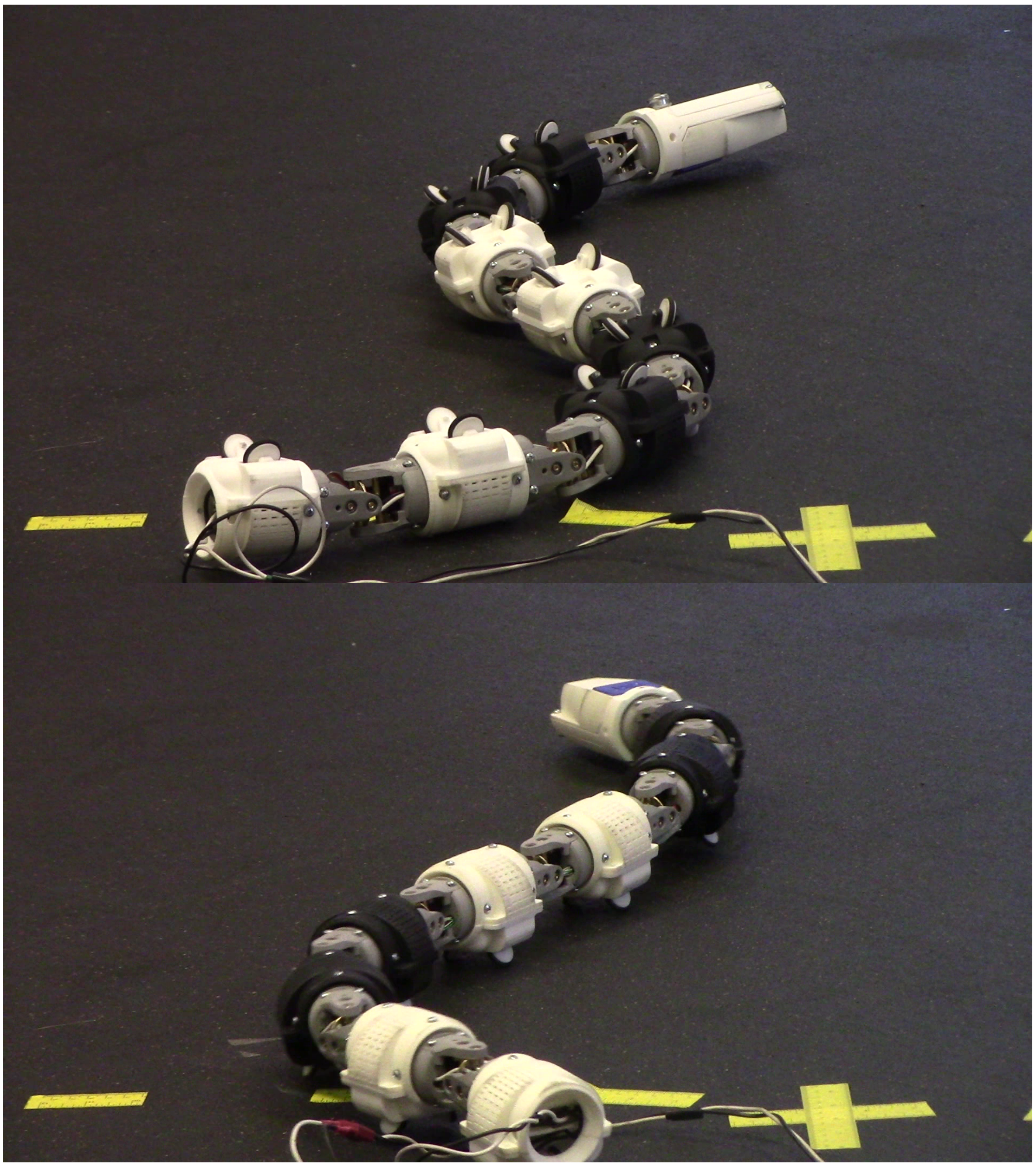


Figure 5.2.3: Robot snake demonstrates rolling from the upside-down position (top) to the rightside-up position (bottom.)

5.3 Robot Snake Brain 1.0

Robot Snake Brain 1.0 consists of breakout boards, Arduinos, and other modules interconnected by means of a perfboard, headers, and jumper wires as shown on Figure 5.3.1. It proved sufficient for demonstrating basic locomotion as well as remotely-operated motion pattern generation. However, due to the ATMega328's low computing power and inability to interface both the PWM and IMU chips on one I2C line, Snake Brain 1.0 was deemed insufficient for the implementation of self-contained closed-loop control algorithms.

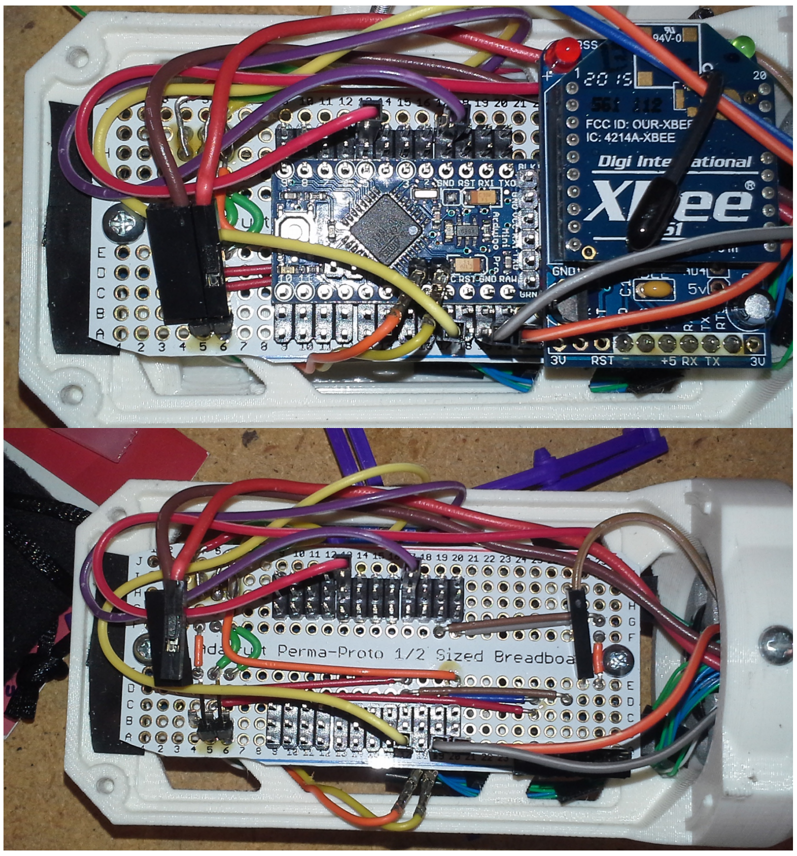


Figure 5.3.1: Robot Snake Brain 1.0, fully assembled (top) and with its Arduino Pro Mini and XBee adapter removed (bottom.)

5.4 Robot Snake Brain 2.0

Robot Snake Brain 2.0 performed at least as well as Robot Snake Brain 1.0. When successfully assembled, it has the advantage of an on-board IMU working on the same I2C line as the PWM generator chip. Combining all of those components on one PCB (as shown on Figure 5.4.1) frees up space within the robot snake's skull and allows for future addtions of new sensors and closed-loop control algorithms. However, Robot Snake Brain 2.0 was initially found to be very difficult to reflow solder. Only one of three attempted circuit boards assembled successfully, which is a manufacturing yield of approximately 33%.

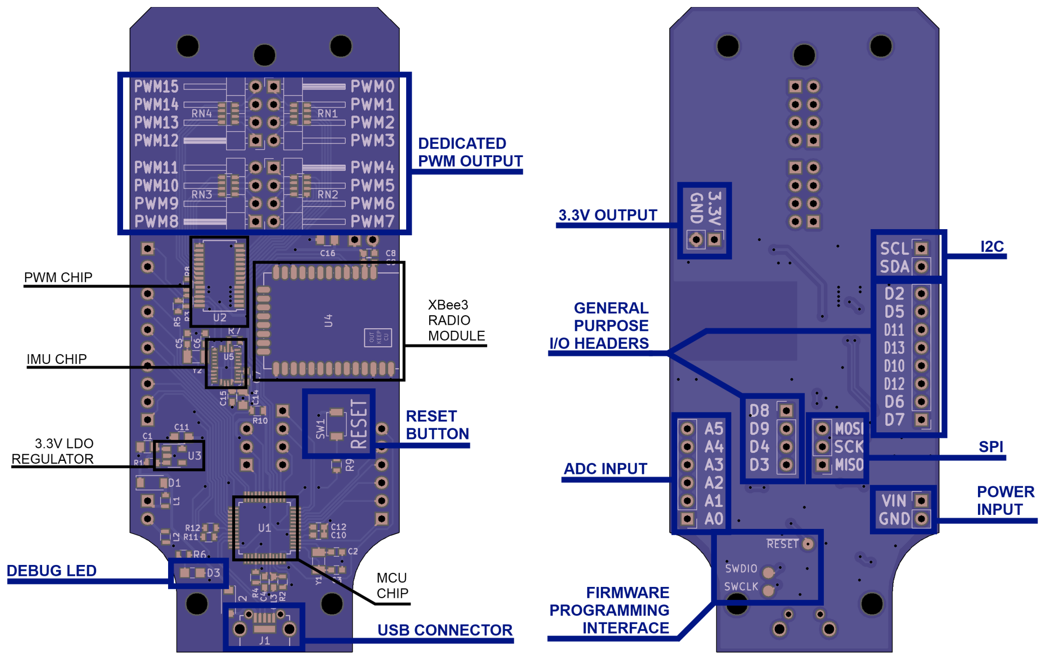
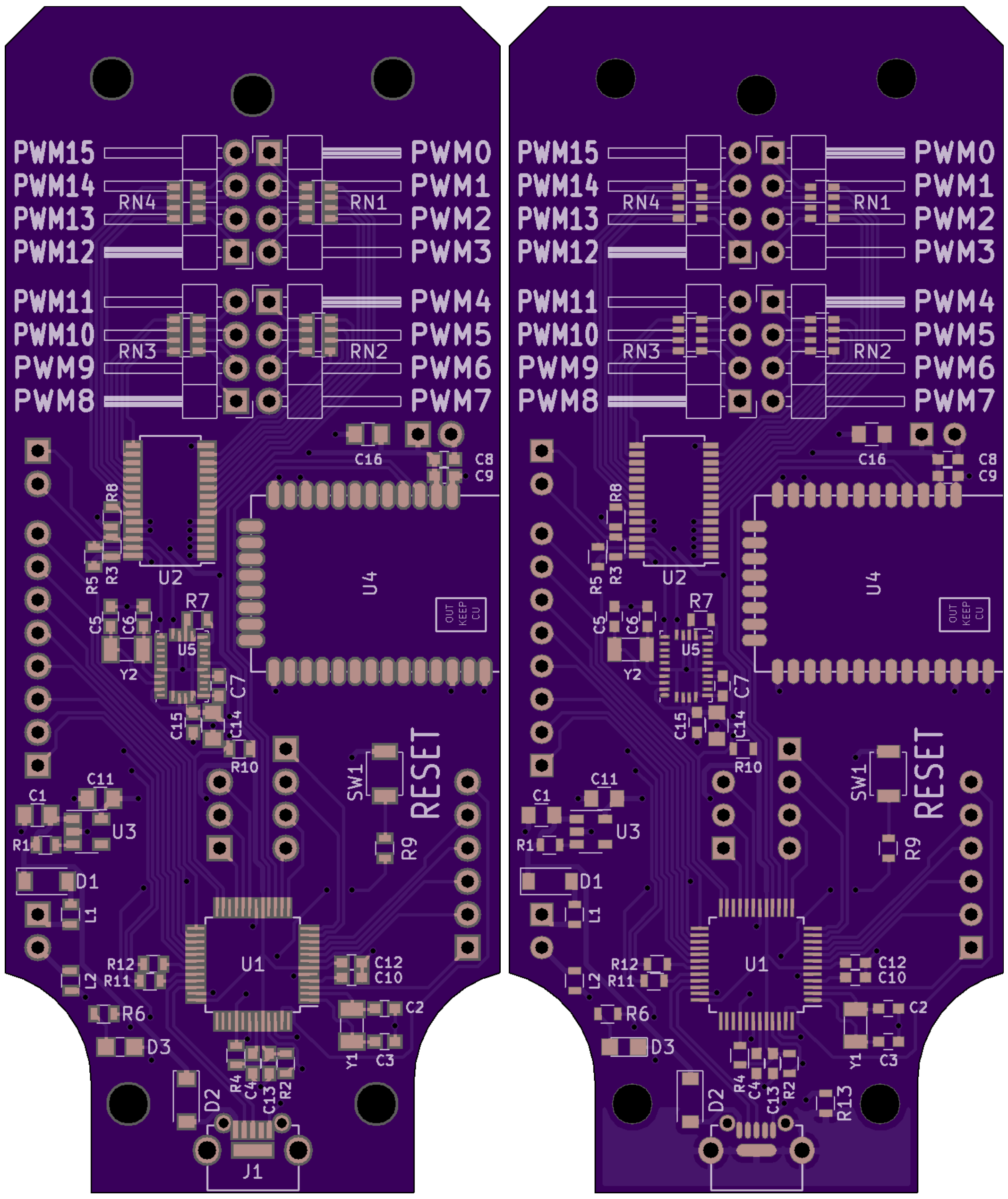


Figure 5.4.1: Component locations on Robot Snake Brain 2.0.

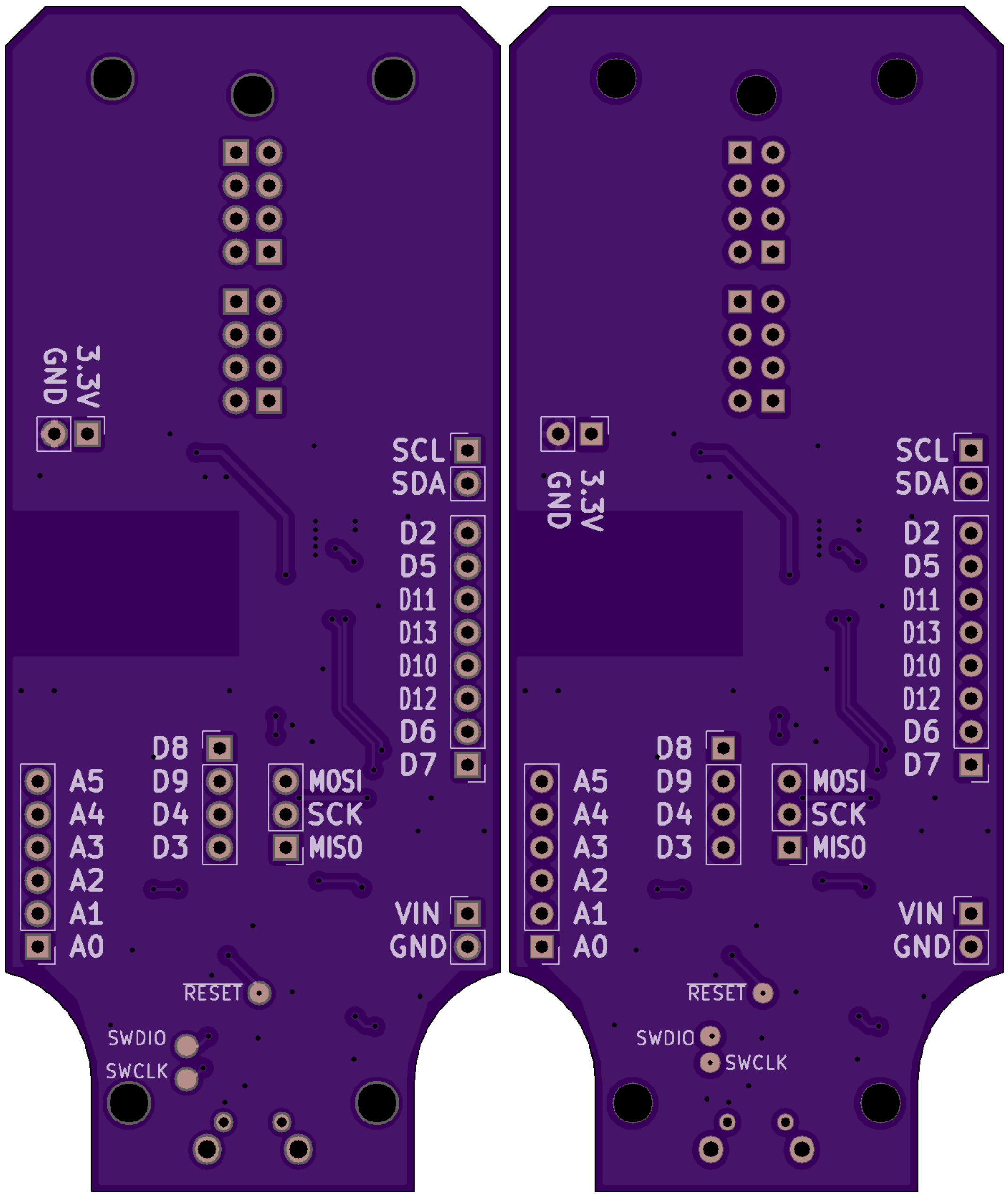
5.5 Robot Snake Brain 2.1

Although visually similar to the previous version, as shown on Figure 5.5.1 and Figure 5.5.2, version 2.1 was easier to reliably reflow-solder due to manufacturability improvements to the design of its circuit board and solder paste stencil, with a yield of 100% for a sample of three PCBs.



(a) (b)

Figure 5.5.1: Side-by-side comparison between the bottom view of the raw circuit board of **(a)** Robot Snake Brain 2.0 and **(b)** Robot Snake Brain 2.1.



(a) (b)

Figure 5.5.2: Side-by-side comparison between the top view of the raw circuit board of **(a)** Robot Snake Brain 2.0 and **(b)** Robot Snake Brain 2.1.

5.6 Remote Command Unit

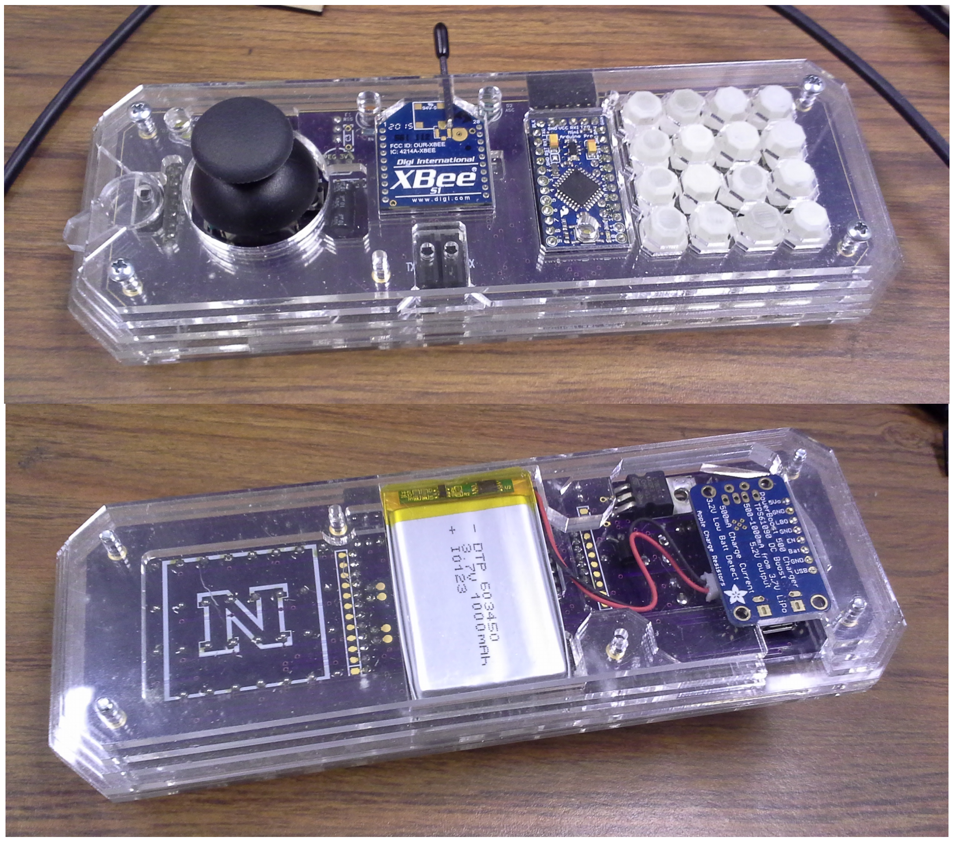


Figure 5.6.1: Remote Command Unit; front (top) and back (bottom.) Some parts of the Remote Command Unit, such as the joystick, matrix keypad buttons, XBee antenna, and Arduino programming header, protrude from the Acrylic-sandwich enclosure.

The Remote Command Unit, shown on Figure 5.6.1, is functional for control of the robot and aesthetically sufficient for public demonstrations. However, it is not without its flaws. The most prominent design flaw is the power switch on the left side of the enclosure. A small lever moving between the *on* and *off* positions in a rotary motion moves a slide switch between its 'on' and 'off' state. Although it initially worked well, friction caused both the lever and the slide switch to deteriorate over time. Now the lever requires much more force to operate. The slide switch is likely to eventually wear down, fail, and require replacement.

5.7 Virtual Remote Command Unit

The Virtual Remote Command Unit offers functionality comparable to the aforementioned Remote Command Unit at a fraction of the hardware cost. The only required hardware is an XBee to USB converter, a USB cable, and a computer capable of running a Java applet. The applet connects to a serial port and responds to both mouse and keyboard input by transmitting serial command packets. Figure 5.7.1 shows a screen capture of the Virtual Remote Command Unit applet connected to serial port “COM4” and responding to the mouse cursor (not shown) dragging its control stick upward and to the right.

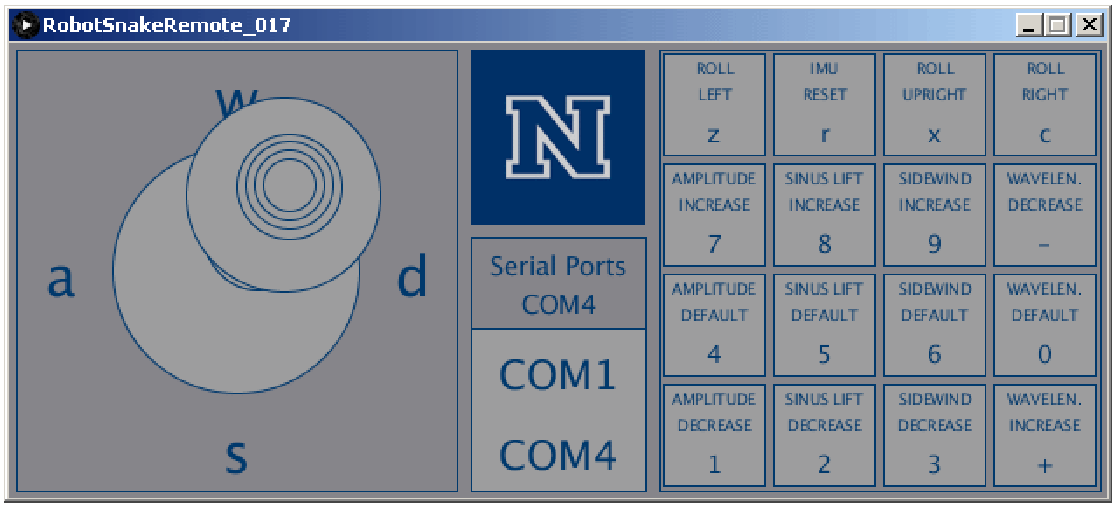


Figure 5.7.1: Virtual Remote Command Unit applet window.

Chapter 6

Future Work

Snake-like robots are a relatively recent area of study in the field of robotics. Therefore, it to follows that most of the possible improvements that can be invented in this field, have yet to be invented. This chapter suggests some of the potential future improvements that now appear to be within our reach.

6.1 Eye Lasers (LIDAR)

As cliche as it may seem to make a robot shoot laser beams from its eye(s), the integration of a LIDAR module into the robot snake could provide a greater benefit to the robot's sensing capabilities than if the same module was mounted on a traditional wheeled platform. With Robot Snake Brain 2.0, the upper half of the robot's skull has been freed for other uses, including on-board sensors. The upper cover of the skull could be re-designed to accommodate a miniature fixed-beam LIDAR module. Due to the availability of an on-board IMU and lateral undulation as the robot's primary mode of locomotion, a fixed-direction beam mounted in the robot's eye would provide the benefits of a scanning beam. Lateral undulation would provide the sideways scanning motion, while the IMU could measure the direction in which the LIDAR beam is pointing. Thus, as long as the robot is aware of its location, it can accumulate a point cloud of detection results on the move.

6.2 Geiger Counter Rattle

To demonstrate the robot's applications for radiation detection, a miniature Geiger counter could be mounted within the robot snake's tail, designed to to mimic the appearance of a rattlesnake's rattle and identified with a black-and-yellow radiation warning trifoil. Each detection could be accompanied by a loud, audible 'click' as well as a low-voltage signal sent to the MCU. Optionally, a bright LED within the rattle could flash on each detection. The advantage of mounting the geiger counter within the robot snake's 'rattle' is the intuitive association of rattling with a warning sign.

6.3 Snake (Thermal) Vision

To mimic natural snakes' ability to sense the body heat of their prey and other sources of heat, a low-resolution forward-facing thermal camera (such as AMG8833) may be installed within the robot snake's skull. Although computer vision may be beyond the scope of this project, the module's 5-bit pixel depth and very low (8 by 8 pixel) resolution means a thermal video feed would take less bandwidth than the color video feed of conventional cameras. Alternatively, the thermal camera could be mounted alongside a regular one to augment a low-resolution grayscale video feed with temperature information. To demonstrate the utility of temperature-augmented images, the visible (a), thermal (b), and simulated thermal-augmented (c) images of a person with a plastic bag draped over his hand are shown on Figure 6.3.1.

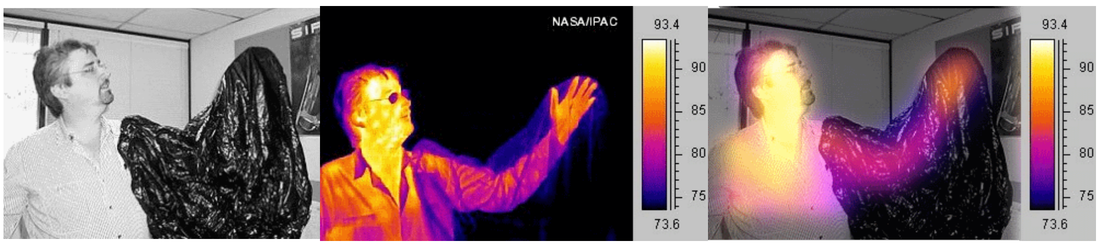


Figure 6.3.1: Comparison of visible **(a)** [38], thermal **(b)** [39], and simulated thermal-augmented **(c)** images side by side. Courtesy of NASA/IPAC.

The simulated thermal-augmented image in Figure 6.3.1 (c) is a weighted additive combination of (a) and (b). The thermal component of the image in (c) has been blurred to simulate a cheap, low-resolution thermal camera. Note how, despite the severely limited resolution of the thermal component, both the plastic bag and he source of heat are still visible in (c). If visible and components are transmitted separately, then software can switch between visible and thermal-augmented modes.

6.4 Improvements to Remote Control Interface

With the availability of the Virtual Remote Command Unit in an easy-to-use IDE, feedback data may be more easily rendered as graphical feedback without costly addition of specialized hardware.

The benefits of remote operation may be more clearly demonstrated if another iteration of the Robot Snake Brain hardware is developed with a different radio module. For example, Bluetooth, WiFi, and GSM are existing radio communication standards which can demonstrate teleoperation over short, medium, and long ranges, respectively. If the Robot Snake Brain were to be modified to receive commands through any of these channels, then the virtual remote control could be written an Android app. Alternatively, a simple USB-to-XBee adapter could be developed specifically for interfacing with a cell phone's USB OTG port; then the Android app could be written to simply send commands through the detected serial port.

6.5 On-Board Power Supply

Each of the sixteen DS450 servos within the robot snake requires up to 200 mA, for a total of 3.2 A, at 4.8 – 6.0 Volts. Normally this would be a straightforward requirement, but the long narrow shape of the robot's body makes it challenging to pack the required batteries and associated power management electronics into the already crowded tubular segments.

6.6 Structural Improvement

A significant improvement to the robot snake's space-efficiency may be achievable through a radical structural redesign. The robot snake's present structure alternates between motorized X-Y joints and tubular segments along the length of the robot, as shown on Figure 6.6.1 (a). Most of the length of the robot's body is occupied by its motorized joints, leaving very little space for additional electronics.

An alternative, proposed structure is illustrated on 6.6.1 (b). This structure fully or partially encloses most of the robot's electrical and mechanical parts in a module shaped like two halves of a cylinder of a height equal to its diameter offset by 90°, or a cube with four rounded edges. Because the length occupied by motorized joints in (b) does not exclude the same length of space from being utilized for other components as in (a), such arrangement may result in similar functionality but with shorter segments, and therefore better mechanical advantage for the motorized joints to propell the robot.

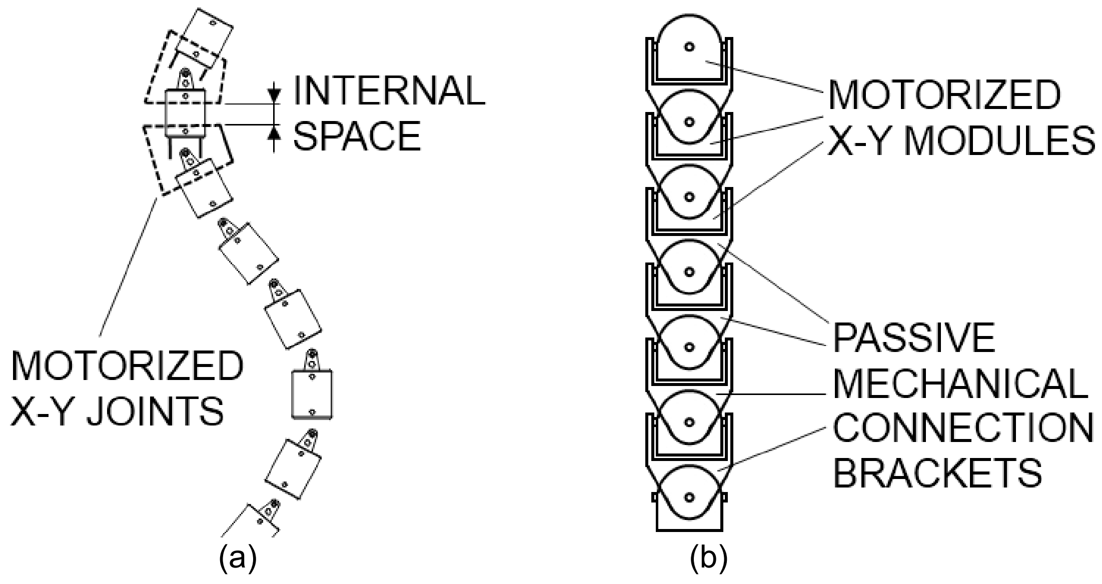


Figure 6.6.1: Structure of robot snake's body; present **(a)** and proposed **(b)**.

6.7 Environment-Proofing

To truly demonstrate the feasibility of deployment of snake-shaped robots in realistic environments, it may be necessary to enclose the robot in a water-proof (or at least dust-proof) sleeve. Any surfaces designed for contact with terrain, such as passive wheels and anisotropic friction scales, would have to be attached to the outside of the environment-proof sleeve.

6.8 Anisotropic Friction Surfaces

Current prototypes make use of passive wheels to provide the robot snake with anisotropic friction. However, it was demonstrated that it is possible to achieve anisotropic friction without moving parts, using sharp edges of low-friction metal, plastic, or other materials [34].

Bibliography

[1] Tokyo Institute of Technology. [Online].
Available: https://www.titech.ac.jp/english/research/stories/shigeo_hirose.html

[2] Adafruit Feather M0 Basic Proto. Adafruit Industries. July 2018. [Online].
Available: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-feather-m0-basic-proto.pdf

[3] Low-Power, 32-bit Cortex-M0+ MCU with Advanced Analog and PWM. ATSAMD21G18A. Rev. C. Microchip Technology Incorporated. June 2018. [Online].
Available: http://ww1.microchip.com/downloads/en/DeviceDoc/SAM-D21-Family-Datasheet-DS40001882C.pdf

[4] AVR® Microcontroller with picoPower® Technology. Atmega328/P. Rev. A. Microchip Technology Incorporated. February 2018. [Online].
Available: http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega328_P%20AVR%20MCU%20with%20picoPower%20Technology%20Data%20Sheet%2040001984A.pdf

[5] 500mA Low-Noise LDO Voltage Regulator. SPX3819M5. Rev. 2.0.3. Exar Corporation. August 2016. [Online].
Available: https://www.exar.com/ds/spx3819.pdf

[6] 2 Layer Prototype Service. Oshpark LLC. [Online].
Available: http://docs.oshpark.com/services/two-layer/

[7] G. Smith *at FCT Assembly, Inc.*, “Improve SMT Assembly Yields Using Root Cause Analysis in Stencil Design,” in *IPC Apex Expo Technical Conference, 2017 Proceedings of.* [Online].
Available: https://fctsolder.com/wp-content/uploads/2017/08/S06_01-Improve-SMT-Assembly-Yields-Using-Root-Cause-Analysis-in-Stencil-Design-Final.pdf

[8] R. Dervaes *at FCT Assembly, Inc.*, “SMT Assembly Challenges and Proven Solutions for Improving Yields,” [Online].
Available: https://fctsolder.com/wp-content/uploads/2017/08/solutionsincreasingyields.pdf

[9] E. Weisstein. "Rotation Matrix." From *MathWorld--A Wolfram Web Resource*. [Online].
Available: http://mathworld.wolfram.com/RotationMatrix.html

[10] Chironis, N. P., 1967, *Gear Design and Application*, McGraw-Hill

[11] H. Sparks. “Designing Cycloidal Gears.” April 2013. [Online].
Available: https://www.csparks.com/watchmaking/CycloidalGears/index.jxl

[12] E. Gabrielyan. “Draft of a piston-less pressure driven engine.” May 2010. [Online].
Available: https://docs.switzernet.com/people/emin-gabrielyan/100512-gear-engine/

[13] NASA Jet Propulsion Laboratory. “MISSION TO MARS: Mars Exploration Rover – Spirit.” [Online].
Available: https://www.jpl.nasa.gov/missions/mars-exploration-rover-spirit-mer/

[14] W. Yang, A. Bajenov, Y. Shen. “Improving low-cost inertial-measurement-unit (IMU)-based motion tracking accuracy for a biomorphic hyper-redundant snake robot.” SpringerOpen, December 2017.
Available: https://jrobio.springeropen.com/articles/10.1186/s40638-017-0069-z

[15] S. Hirose, A. Morishima. “Design and Control of a Mobile Robot with an Articulated Body.” *The International Journal of Robotics Research,* April 1990, pp. 99-114.

[16] S. Hirose, H. Yamada. “Machine Design of Biologically Inspired Robots.” *IEEE Robotics and Automation Magazine,* March 2009, pp. 88-98.

[17] S. Hirose, E. Fukushima. “Development of mobile robots for rescue operations.” *Advanced Robotics,* Vol. 16, No. 6, 2002, pp. 509-512.

[18] S. Hirose, E. Fukushima. “Snakes and Strings: New Robotic Components for Rescue Operations.” *The International Journal of Robotics Research,* Vol. 23, No. 4-5, April-May 2014, pp. 341-349.

[19] H. Yamada, M. Mori, S. Hirose. “Stabilization of the head of an undulating snake-like robot.” *IEEE/RSJ International Conference on Intelligent Robots and Systems, Proceedings of.* October-November 2007, pp. 3566-3571.

[20] H. Yamada, S. Takaoka, S. Hirose. “A snake-like robot for real-world inspection applications (the design and control of a practical active cord mechanism.)” *Advanced Robotics,* Vol. 27, No. 1, 2013, pp. 47-60.

[21] S. Takaoka, H. Yamada, S. Hirose. “Snake-like Active Wheel Robot ACM-R4.1 with Joint Torque Sensor

and Limiter.” *IEEE/RSJ International Conference on Intelligent Robots and Systems.* September 2011, pp. 1081-1086.

[22] Buckingham, Earle, 1887, *INVOLUTE SPUR GEARS*, Niles-Bement-Pond Co.

[23] Fellows Gear Shaper Company, c. 1950. *THE INVOLUTE CURVE AND INVOLUTE GEARING*.

[24] Buckingham, Earle, 1887, *SPUR GEARS: DESIGN, OPERATION, AND PRODUCTION*, McGraw-Hill, 1928.

[25] Shapeways, Inc. *Brass. Design Guidelines.* [Online].
Available: https://www.shapeways.com/materials/brass

[26] T. Takayama, S. Hirose. “Amphibious 3D Active Cord Mechanism “HELIX” with Helical Swimming Motion.” *IEEE/RSJ International Conference on Intelligent Robots and Systems, Proceedings of.* October 2002, pp. 775-780.

[27] H. Yamada, S. Shigisaki, M. Mori, K. Takita, K. Ogami, S. Hirose. “Development of amphibious snake-like robot ACM-R5.” *36th Int. Symp. Robotics, Proceedings of,* 2005.

[28] T. Ohashi, H. Yamada, S. Hirose. “Loop Forming Snake-like Robot ACM-R7 and Its Serpenoid Oval Control.” *IEEE/RSJ International Conference on Intelligent Robots and Systems.* October 2010, pp. 413-418.

[29] H. Komura, H. Yamada, S. Hirose. “Development of snake-like robot ACM-R8 with large and mono-tread wheel.” *Advanced Robotics,* Vol. 29, No. 17, 2015, pp. 1081-1094.

[30] J. A. Silva Rico, G. Endo, S. Hirose, H. Yamada. “Development of an actuation system based on water jet propulsion for a slim long-reach robot.” *ROBOMECH Journal.* SpringerOpen, March 2017.
Available: https://robomechjournal.springeropen.com/articles/10.1186/s40648-017-0076-4

[31] H. Yamada, S. Hirose. “Study of a 2-DOF Joint for the Small Active Cord Mechanism.” *IEEE International Conference on Robotics and Automation*. May 2009, pp. 3827-3832.

[32] H. Yamada, S. Hirose. “Steering of Pedal Wave of a Snake-like Robot by Superposition of Curvatures.” *IEEE/RSJ International Conference on Intelligent Robots and Systems.* October 2010, pp. 419-424.

[33] W. Yang. “Biomorphic Hyper-Redundant Snake Robot: Locomotion Simulation, 3D Printed Prototype and Inertial-Measurement-Unit-Based Motion Tracking.”

[34] M. Saito, M. Fukaya, T. Iwasaki. “Serpentine Locomotion with Robotic Snakes.” *IEEE Control Systems Magazine.* February 2002, pp. 64-81.

[35] C. Wright, A. Buchan, B. Brown, J. Geist, M. Schwerin, D. Rollinson, M. Tesch, H. Choset. “Design and Architecture of the Unified Modular Snake Robot.” *IEEE International Conference on Robotics and Automation*. May 2012, pp. 4347-4354.

[36] Courtesy NASA/JPL-Caltech.
Available: https://www.jpl.nasa.gov/spaceimages/images/largesize/PIA12103_hires.jpg

[37] Conrad Baetsle. “*Chrysopelea ornata* or the golden tree snake.” December 26, 2010.
Retrieved from: https://en.wikipedia.org/wiki/File:Golden_tree_snake.jpg

[38] NASA/IPAC.
Retrieved from: https://en.wikipedia.org/wiki/File:Human-Visible.jpg

[39] NASA/IPAC.
Retrieved from: https://en.wikipedia.org/wiki/File:Human-Infrared.jpg

[40] Photograph entitled ”*299_9983*”. © David “DB King”, February 28, 2006. Published under the Creative Commons Atribution 2.0 Generic (CC BY 2.0) license: https://creativecommons.org/licenses/by/2.0/legalcode
Retrieved from: https://www.flickr.com/photos/65193799@N00/106082884

[41] Spark Fun Electronics. “Arduino Pro Mini.” [Online].
Available: https://cdn.sparkfun.com/datasheets/Dev/Arduino/Boards/Arduino-Pro-Mini-v14.pdf

[42] Adafruit Industries. “PowerBoost 500 Charger.” [Online].
Available: https://learn.adafruit.com/adafruit-powerboost-500-plus-charger/downloads

[43] Adafruit Industries. “XBee v1.1 Adapter.” [Online].
Available: https://learn.adafruit.com/xbee-radios/download

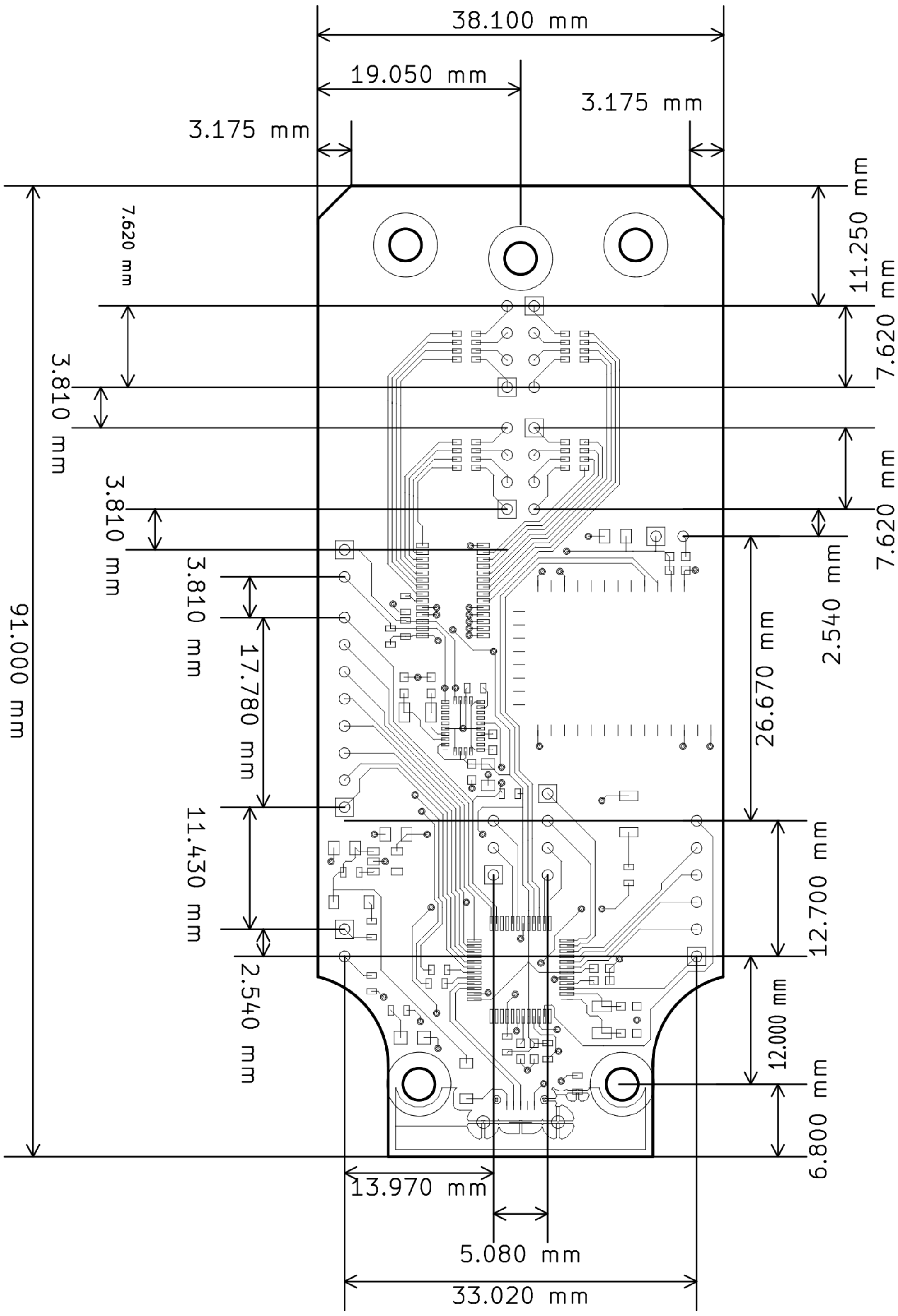
[44] Adafruit Industries. “PCA9685 16-Channel Servo Driver.” [Online].

Available: https://learn.adafruit.com/16-channel-pwm-servo-driver/downloads

[45] BNO055 IMU Breakout Board
Available: https://learn.adafruit.com/adafruit-bno055-absolute-orientation-sensor/downloads

Appendix A

Mechanical Drawings



PITCH OF ALL IN-LINE PIN HEADERS IS 2.54 MILLIMETERS

Figure A.1: Dimensions of raw circuit board of Robot Snake Brain 2.1.

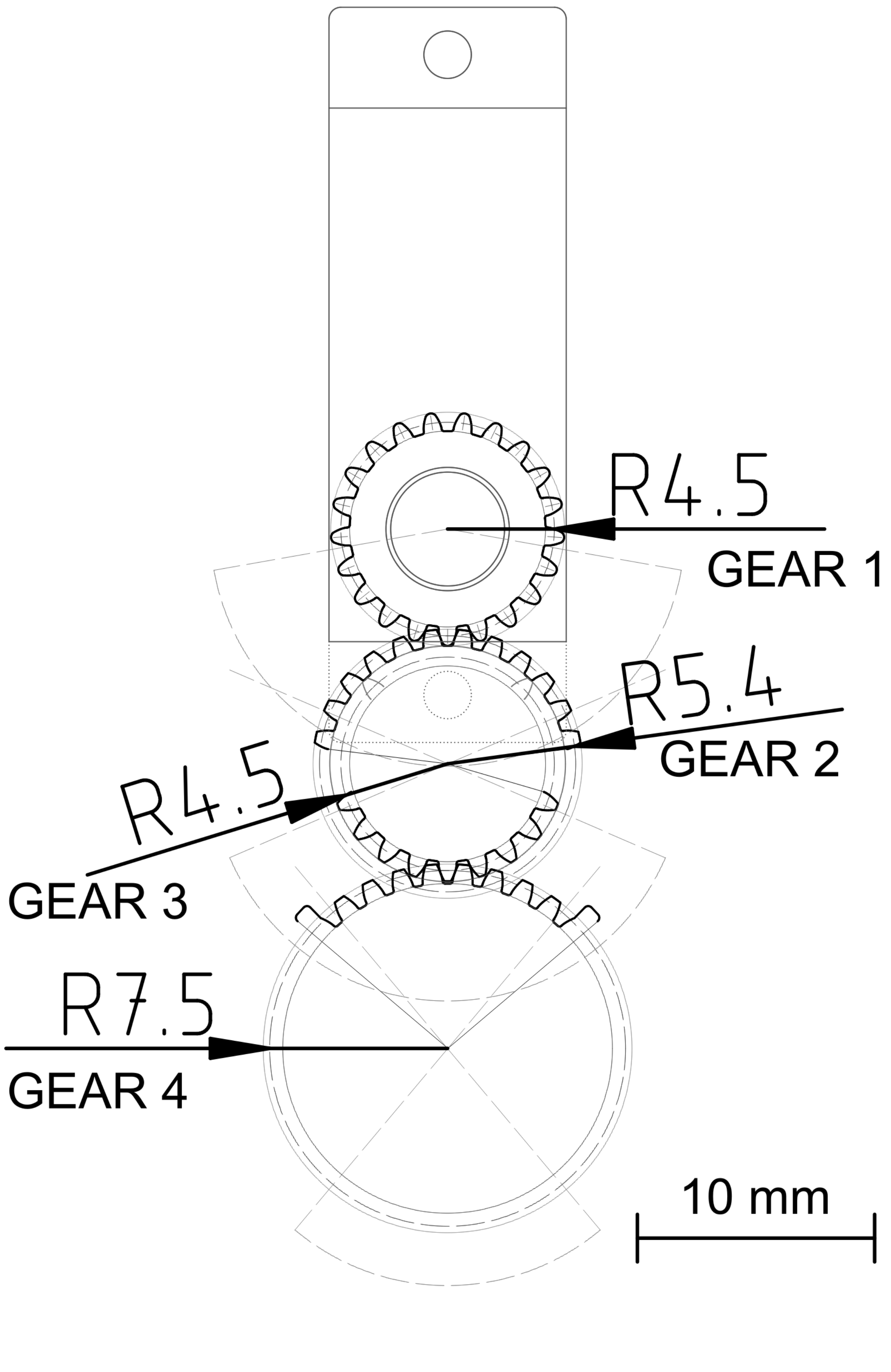


Figure A.2: Pitch radii and arrangement of gears in gear train.

Appendix B

Electrical Schematics

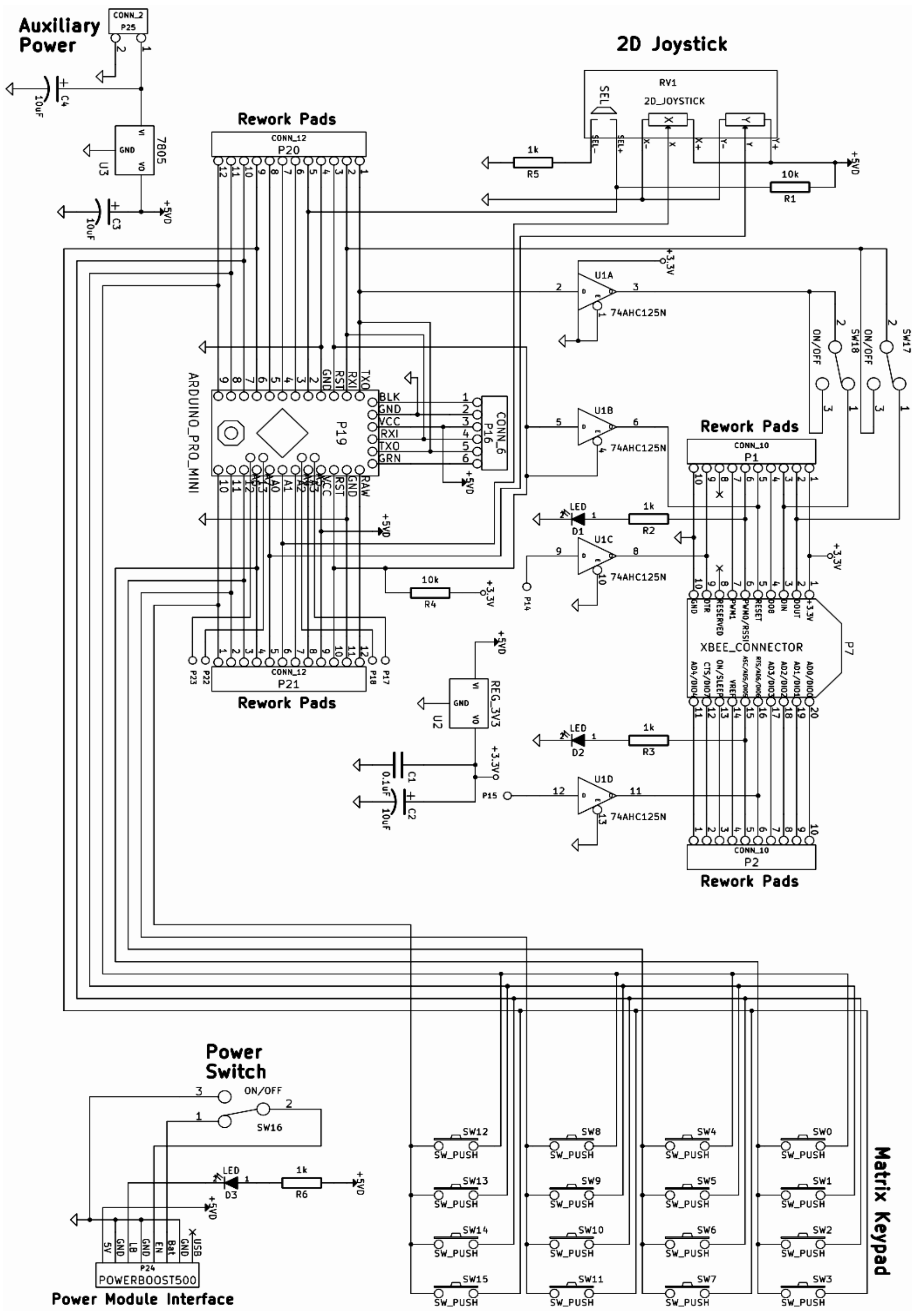


Figure B.1: Electrical schematic of remote command unit

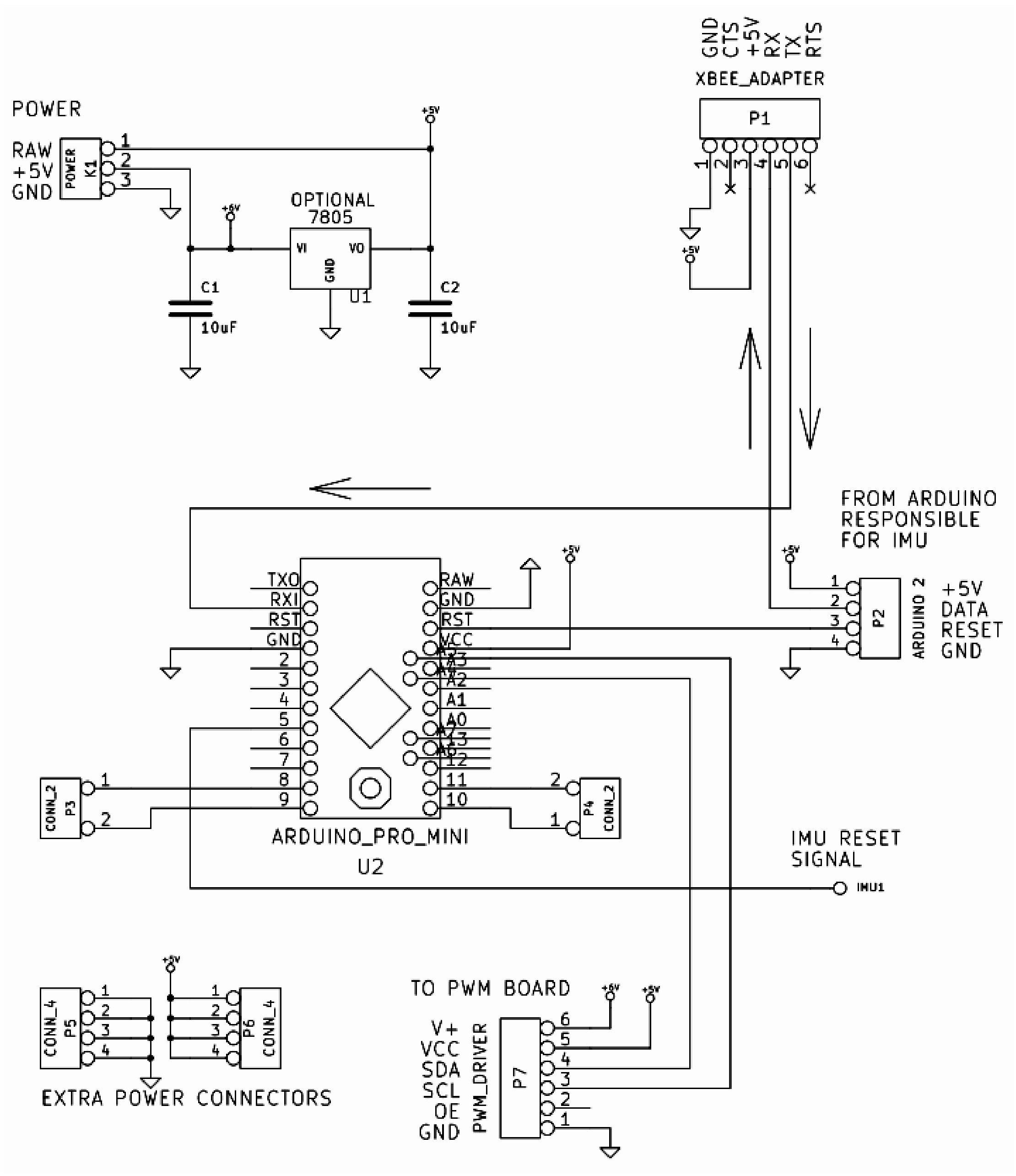


Figure B.2: Electrical schematic of Robot Snake Brain 1.0

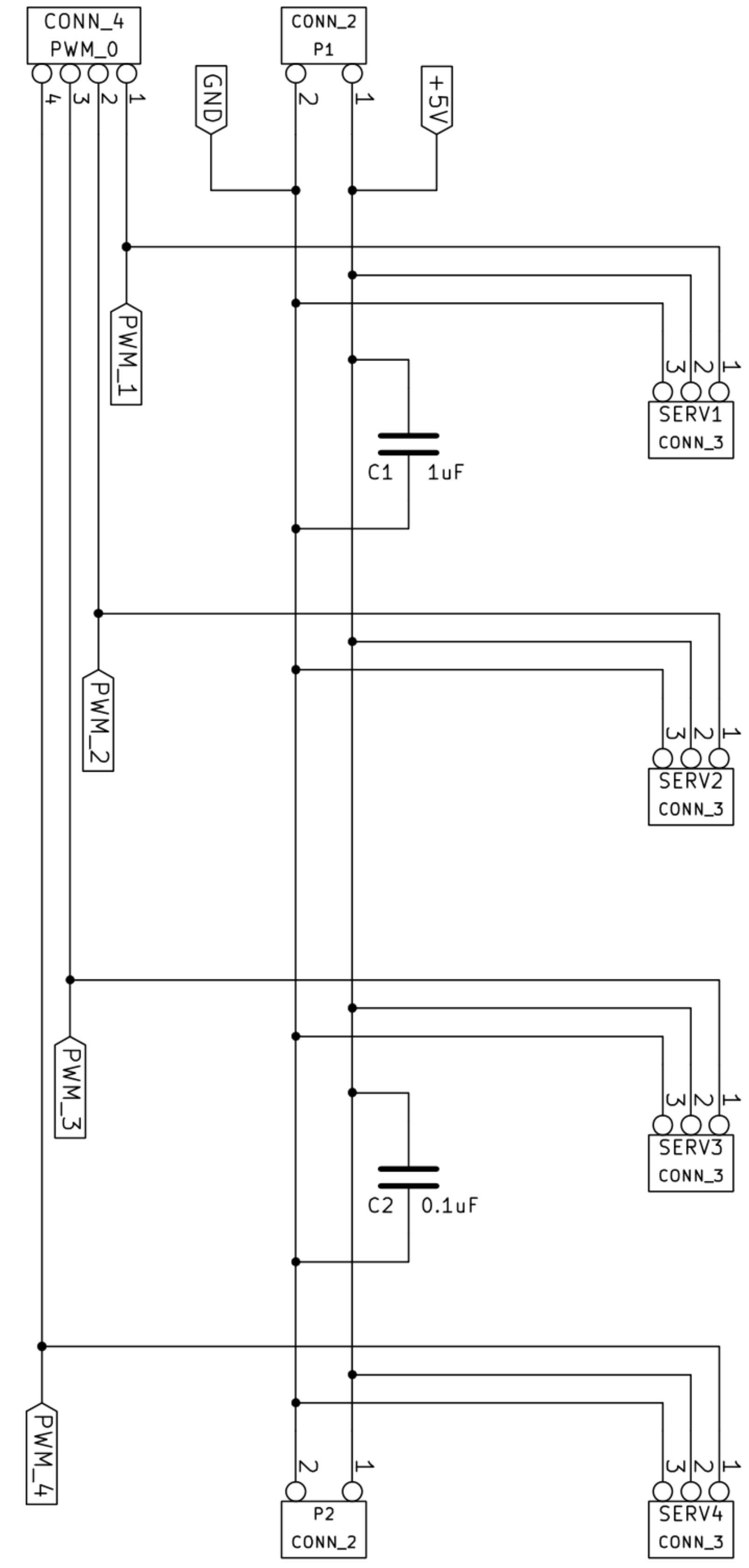


Figure B.3: Electrical schematic of robot snake interconnect node

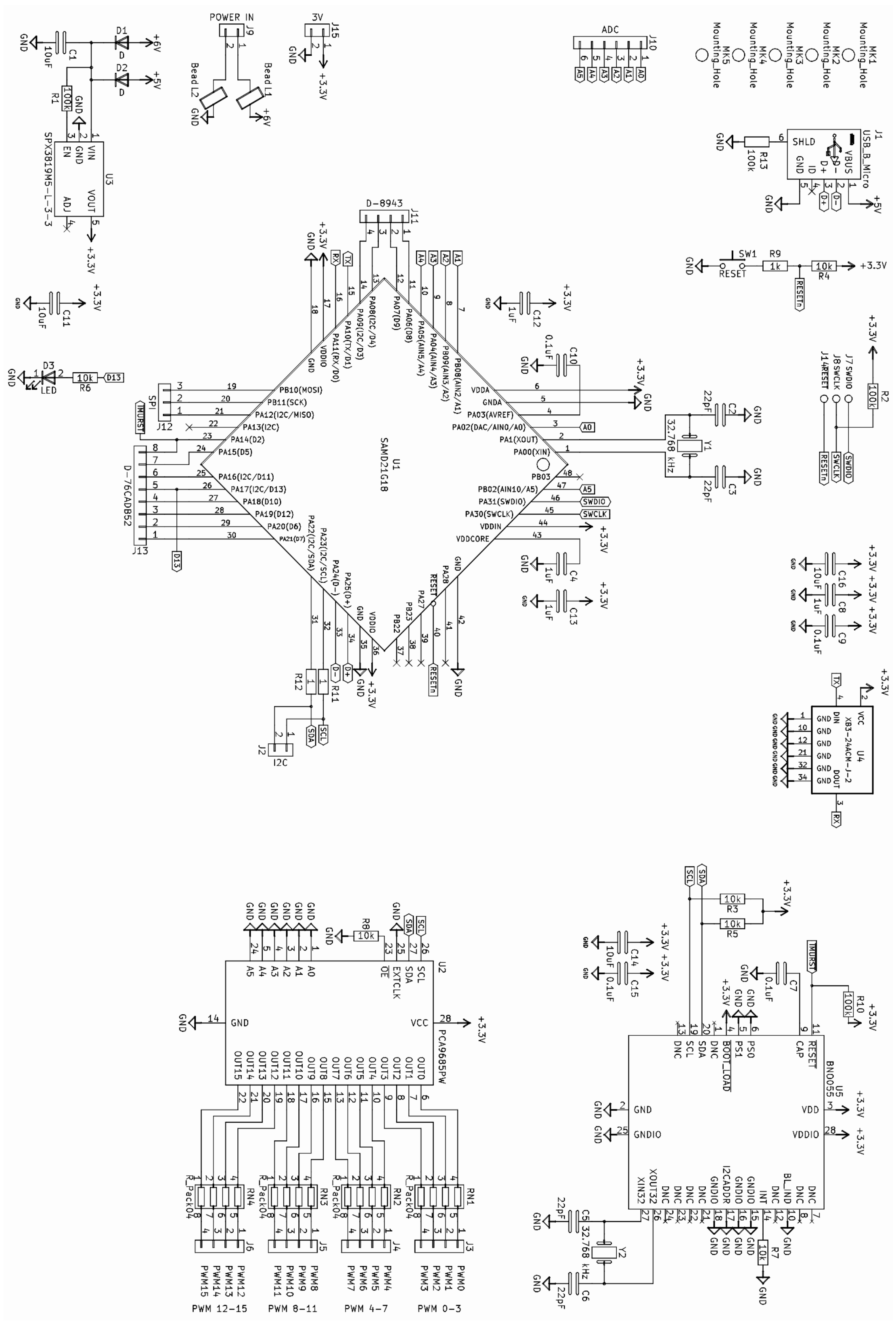


Figure B.4: Electrical schematic of Robot Snake Brain 2.1

Appendix D

Simple Involute Spur Gear Design

This section covers the practical and mathematical considerations related to design of a simple involute spur gear train. The scope of this brief outline is limited to simple gear trains consisting entirely of involute (i.e.with involute gear tooth profiles, based on the involute [1] of a circle) spur gears with constant pitch radii, and is intended for very basic applications, such as light-duty gears designed to be manufactured by means of 3D printing or laser cutting.

The simplest gear train consists of two meshing gears (although many more gears may be present in more complex gear trains,) which transfer kinetic energy from one rotating object to another, with some tradeoff between speed and torque. This is analogous to electrical transformers transfering electrical energy with some tradeoff between current and voltage.
The design specifications of a pair of meshing gears, which may be selected at the beginning of the design process, include:

- The gear ratio R, which governs the speed/torque trade-off of a meshing gear pair
- Mounting distance D_m between the axes of meshing gears
- Desired depth of teeth, which depends on the precision of the manufacturing process you intend to use

The following aspects of a pair of meshing gears are also important, and must also be determined during the design process:

- The pressure angle θ, which affects the efficiency and durability of gears
- Width of gear teeth, which govern the maximum torque the system can endure

Gear Ratio:

This may be thought of as the ratio of pitch radii of the gears:

$$R=\frac{r_{p1}}{r_{p2}}$$

For continuously rotating gears, this is also equivalent to the ratio of quantity of teeth on each gear:

$$R=\frac{N_1}{N_2}$$

The object attached to the gear with a lesser radius rotates faster than the object attached to the gear with greater radius, but has less torque, as illustrated on Figure D.1. When more than one pair of meshing gears is present in a gear train, their gear ratios and their efficiency stack multiplicatively. In terms of efficiency, that is undesirable: efficiency of real systems is almost always less than unity, and the product of efficiencies of multiple pairs of meshing gears is always less than unity.

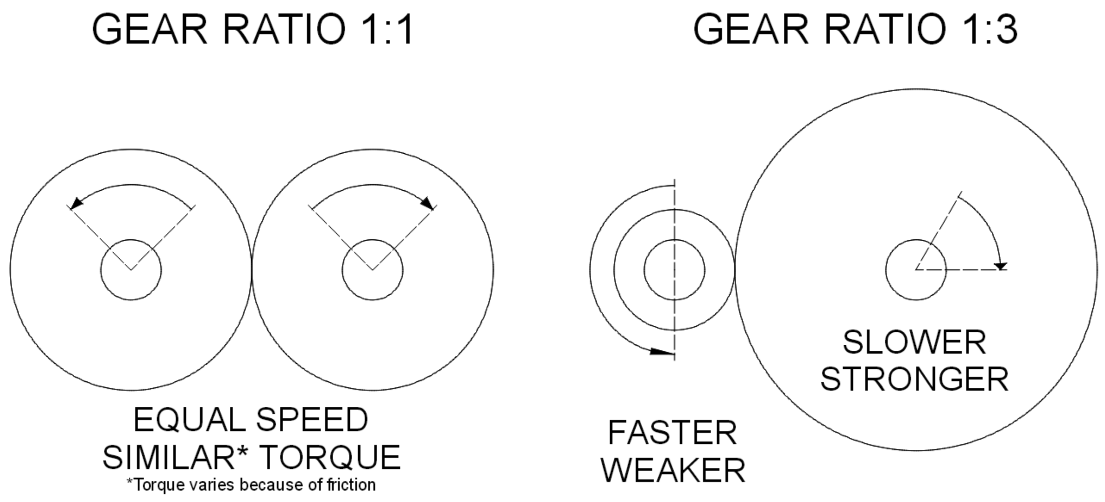


Figure D.1: Effects of variation of gear ratio. In this figure, the gears' teeth are not shown, and the gears are approximated as cylinders, with the same radius as their pitch circles, rolling against each other without slipping.

Pressure Angle:

This is the angle at which force is transferred between gear teeth as shown on Figure D.2. Because the line of force is tangent to both base circles, the pressure angle is determined by the ratio of radii of base circle r_b and pitch circle r_p and will be equal for both gears in a meshing pair:

$$\theta = \cos^{-1}\left(\frac{r_b}{r_p}\right)$$

Greater pressure angles cause slightly greater loss of power due to friction, but allow wider gear teeth to be built.

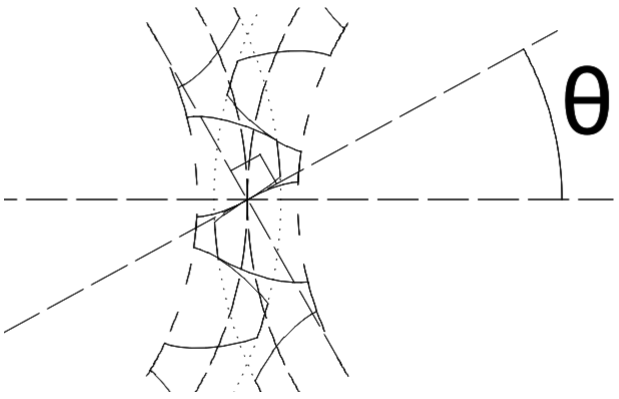


Figure D.2: Pressure angle between two gears. Note that the line of force (excluding friction) between the gears is tangent to both gears' base circles.

Width of Teeth:

The width of gear teeth (along with extrusion length and material properties) affects the maximum force each gear tooth can endure without breaking. Given the same radius of base circle, wide gear teeth can be deeper than narrow gear teeth. Depending capabilities of available prototyping technology, limits of detail and precision may impose depth and width limits to the design of gears.

For a pair of gears to mesh correctly with each other, a number of requirements must be met:

- The circular pitch of meshing gears must be equal. The circular pitch of a gear is the arc distance, measured along the pitch circle, between the same point on two adjacent teeth. For continuously rotating gears, the circular pitch is the circumference of the pitch circle divided by the number of teeth:

$$P = \frac{C}{N} = \frac{2\pi r_p}{N}$$

- For continuously meshing gears (*i.e.*, gears which are required to turn full 360° and farther,) the number of teeth must be a positive integer. However, for gears which are not required to rotate all the way around, the number of teeth just needs to be a real number.
- Inaccuracy of mounting distance between the gears must be less than the depth of contact between the meshing gears' teeth. Meshing gears tend to push away from each other, proportionally to the sine of their pressure angle. It was observed that if the gears are allowed to become pushed too far apart, their teeth slip past each other, breaking the train of transfer of kinetic energy and potentially causing damage to themselves.

The design process begins by defining the outer requirements of the gear train and limitations of materials and manufacturing technology. The pitch radii of both gears can be determined from the desired gear ratio R and the mounting distance D_m between the gears' axes:

$$r_{p1}=\frac{D_m}{R+1}; \qquad r_{p2}=\frac{R\,D_m}{R+1}$$

Next, the gears' base radii (r_b, used to calculate the involute curve) and addendum radii (r_a, the outer-most points of the gear) must be selected. As shown on Figure D.3, they must adhere to the following constraints:

$$r_{a1}>r_{p1};\quad r_{a2}>r_{p2};\quad r_{p1}>r_{b1};\quad r_{p2}>r_{b2};\quad r_{a1}-r_{p1}<r_{p2}-r_{b2};\quad r_{a2}-r_{p2}<r_{p1}-r_{b1}$$

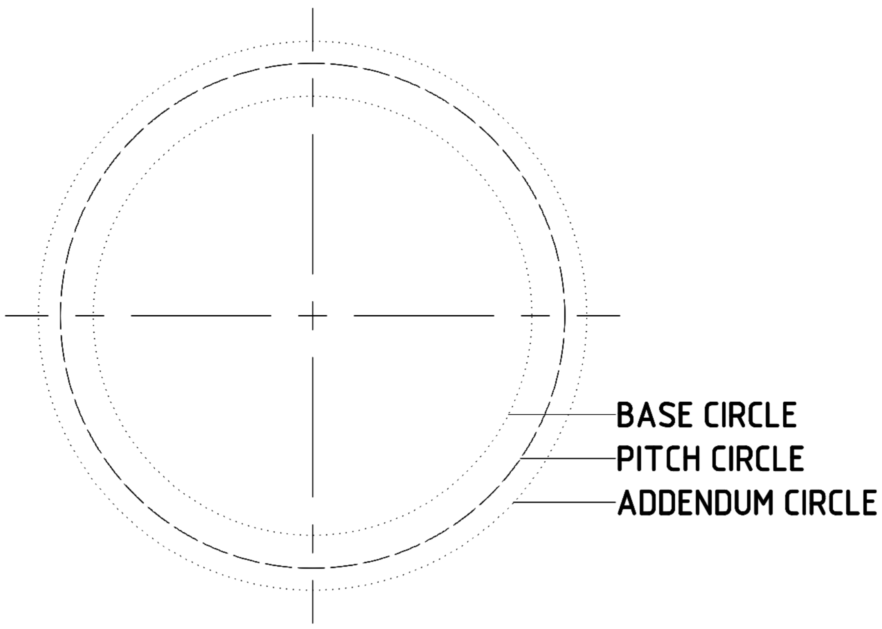


Figure D.3: Visual comparison of base circle, pitch circle, and addendum circle.

It must be noted here for completeness that the circle corresponding to the inner-most points of the gear profile is usually called the 'Dedendum Circle,' and is separate from the base circle, especially if techniques such as 'undercut' are used to decrease the pressure angle for improved efficiency. Undercut techniques involve trochoid-shaped [2] or approximately trochoic-shaped cuts closer to the center of the gear than the inner-most points of the involute contact surfaces, therefore the radius of the dedendum circle is usually less than the radius of the base circle. For this project, however, undercut techniques have not been attempted due to limits of precision of available prototyping technology; as such, for the scope of this article, the base radius and dedendum radius are effectively equal. For the sake of this article, I shall continue to call it 'base circle' unless I am referring to the dedendum circle specifically.

Several trade-offs must be considered at this step. First, the difference between the addendum circle and base circle determines how deep the gears mesh together, which must be sufficiently great that the gears' teeth engage properly. Making it too great with comparison to the base radius creates a great pressure angle, leading to reduced efficiency due to greater friction. Second, if undercut techniques are not used, the difference between the addendum circle and the pitch circle of each gear must be less than the difference between the pitch circle and the dedendum circle of the gear with which it meshes, to create just enough space between adjacent surfaces to avoid friction; this difference can be very small, but is better accomplished by means of undercut techniques. Third, the teeth must be wide enough that they do not break off, but narrow enough that at least one tooth of each gear is meshing with the opposite gear at any time.

The next step is to draw the involute curve. CAD software packages such as LibreCAD and SolidWorks [3] sometimes include tools to add equation-driven curves to a 2D sketch. The formula for the involute [1] of a circle is described by the following parametric equation:

$$x(t)=r_b\left(\cos(t)+t\sin(t)\right); \qquad y(t)=r_b\left(\sin(t)-t\cos(t)\right)$$

Even with reasonably selected numbers, the involute curve is likely to turn out much longer than necessary, as demonstrated on Figure D.4; this is acceptable, because it is much easier to shorten the curve to the addendum circle (as shown on Figure D.5) than it is to lengthen it.

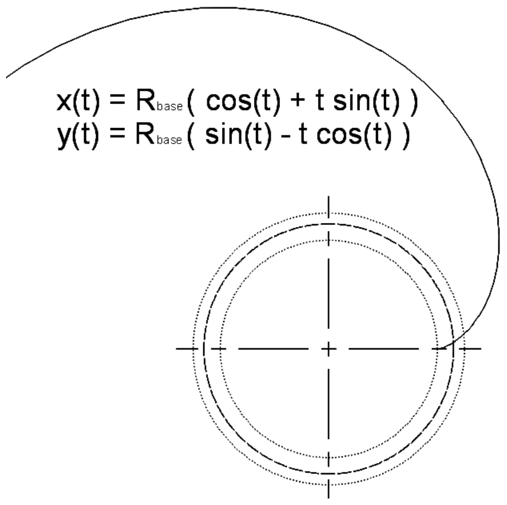

$x(t) = R_{base}(\cos(t) + t\sin(t))$
$y(t) = R_{base}(\sin(t) - t\cos(t))$

Figure D.4: Involute curve plotted from the base circle.

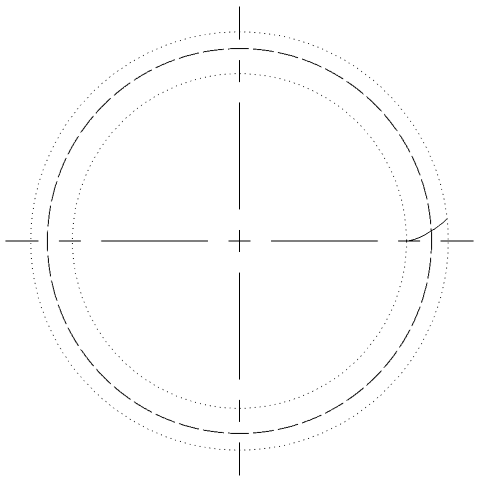


Figure D.5: Involute curve shortened to addendum circle.

The next step is to draw a line through the circle of the gear and the point of intersection of the pitch circle with the involute curve, then another line through the center of the gear but offset counter-clockwise from the previous line by an angle of 90° divided by the number of teeth on the gear, as shown on Figure D.6. This line is to bisect the the gear tooth through the middle, which conveniently allows the section of involute curve to be mirrored around it to form the opposite side of the same gear tooth, as shown on Figure. D.7, easily accomplished with quality CAD software. After this, the outer points of the involute curves can be connected by drawing a short arc along the addendum circle.

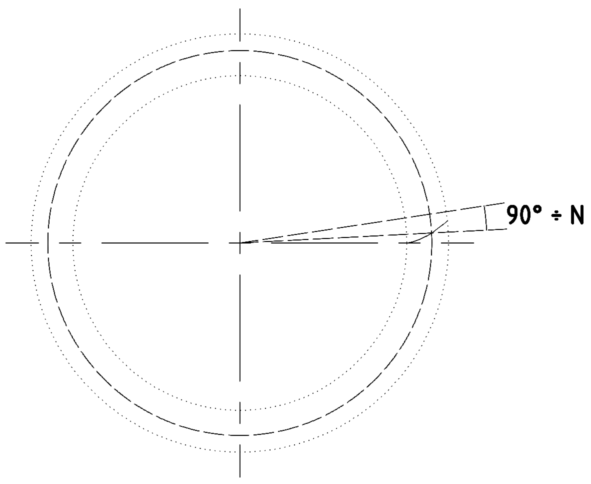


Figure D.6: Construction of the center line of the gear tooth.

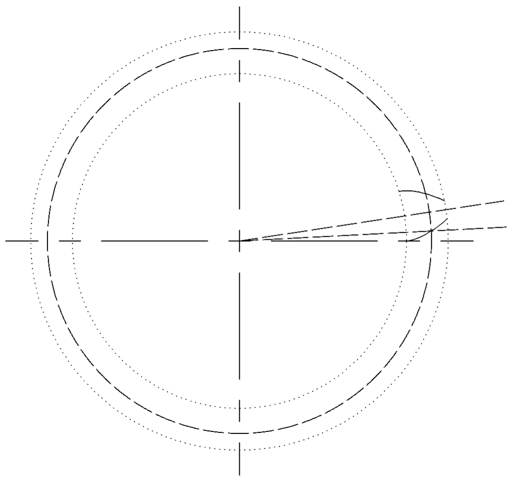


Figure D.7: Construction of the opposite side of the gear tooth by mirroring.

A similar technique is employed to complete the gear tooth and replicate it in a radial pattern to complete the gear. Another line is drawn through the center of the gear at a counter-clockwise angle of 360° divided by the number of teeth on the gear from the horizontal (if the image has been rotated since the involute curve was plotted, that angle should be measured from the starting point of the involute curve.) Another arc can be drawn along the base circle between this line and the starting point of the nearest involute curve to complete the outline of one tooth.

The result is shown on Figure D.8. Once the complete outline of the first tooth is ready, it may be replicated in a radial pattern, offset by angles of 360° divided by the number of teeth on the gear, to complete the closed outline of the entire gear as shown on Figure D.9. This shape may be extruded to form a 3D shape, which, with necessary modifications, can be exported as an STL file for 3D printing.

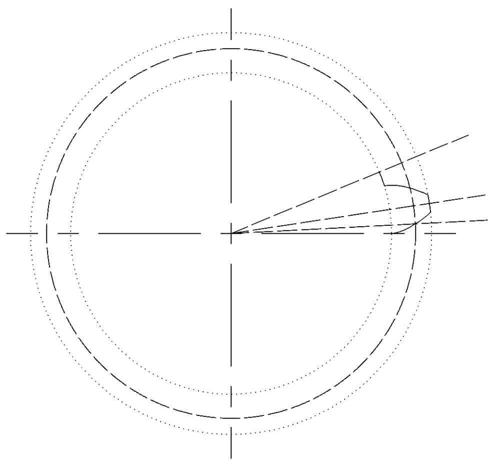


Figure D.8: Completed sketch of one of the gear's teeth.

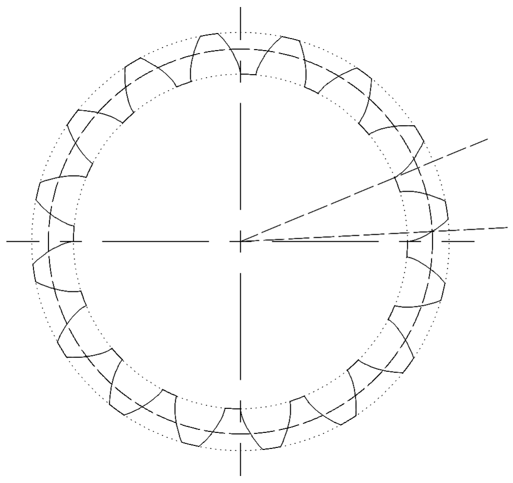


Figure D.9: Completed sketch of all of the gear's teeth.

If the gears were sketched with the purpose of visual representation, the completed sketch may need to be rotated until the straight line, which passes through the intersection of the involute curve with the pitch circle, also passes through the center of the other gear. The gears may then be illustrated together, arranged such that their pitch circles are tangent to each other, and one involute curve from each gear passes through the point where their pitch circles touch, as shown on Figure D.10.

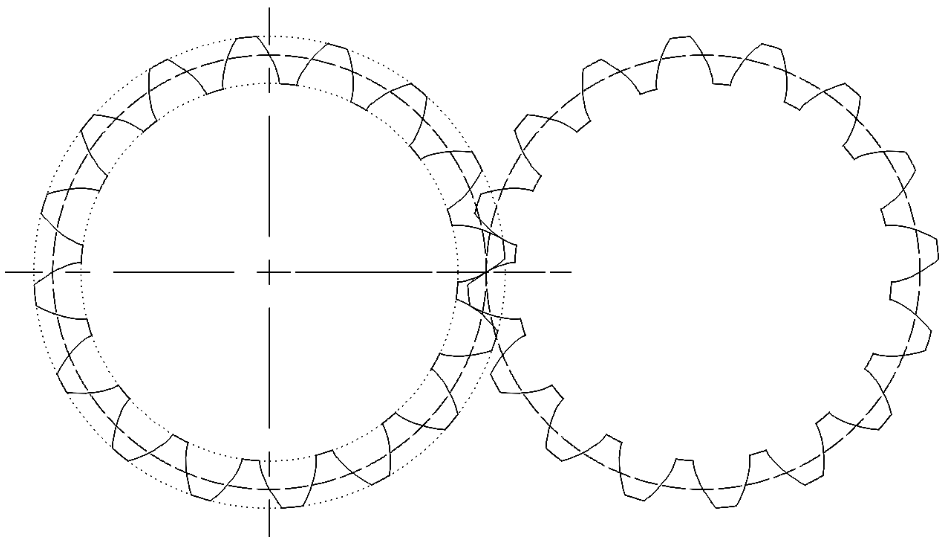


Figure D.10: Sketches of gears arranged to mesh together. Note how the sides of meshing teeth pass through the same point as where the pitch circles are in contact.

Bibliography for Appendix D

[1] Weisstein, Eric W. "Circle Involute." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/CircleInvolute.html

[2] Weisstein, Eric W. "Epitrochoid." From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/Epitrochoid.html

[3] Dassault Systemes. "Equation Driven Curves." From SOLIDWORKS Help. http://help.solidworks.com/2014/English/SolidWorks/sldworks/c_equation_driven_curves.htm#