SELF-CONTAINED SOFT ROBOTIC JELLYFISH WITH WATER-FILLED BENDING ACTUATORS AND POSITIONAL FEEDBACK CONTROL

by

Jennifer Frame

A Thesis Submitted to the Faculty of

The College of Engineering and Computer Science

In Partial Fulfillment of the Requirements for the Degree of

Master of Science

Florida Atlantic University

Boca Raton, Florida

August 2016

SELF-CONTAINED SOFT ROBOTIC JELLYFISH WITH WATER-FILLED BENDING ACTUATORS AND POSITIONAL FEEDBACK CONTROL

by

Jennifer Frame

This thesis was prepared under the direction of the candidate's thesis advisor, Dr. Erik Engeberg, Department of Ocean and Mechanical Engineering, and has been approved by the members of her supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Master of Science.

SUPERVISORY COMMITTEE:

Erik Engeberg, Ph.D.
Thesis Advisor

Edgar An, Ph.D.

Oscar Curet, Ph.D.

Javad Hashemi, Ph.D.
Chair, Department of Ocean and Mechanical Engineering

Mohammad Ilyas, Ph.D.
Dean, College of Engineering and Computer Science

Deborah L. Floyd, Ed.D.
Dean, Graduate College

07/19/2016
Date

ACKNOWLEDGEMENTS

Many thanks to all who have helped make the JenniFish a reality, especially: Chris Nunes for design, modeling, and testing; Ed Henderson and John Kielbasa for electronics layout; Dr. Curet and Dr. Engeberg for partial funding; and my fellow graduate students in the BioRobotics lab for their advice throughout the process. Finally, I would like to sincerely thank my family for all of the much needed support and encouragement they provided.

ABSTRACT

Author:	Jennifer Frame
Title:	Self-Contained Soft Robotic Jellyfish with Water-Filled Bending Actuators and Positional Feedback Control
Institution:	Florida Atlantic University
Thesis Advisor:	Dr. Erik Engeberg
Degree:	Master of Science
Year:	2016

This thesis concerns the design, construction, control, and testing of a novel self-contained soft robotic vehicle; the JenniFish is a free-swimming jellyfish-like soft robot that could be adapted for a variety of uses, including: low frequency, low power sensing applications; swarm robotics; a STEM classroom learning resource; etc. The final vehicle design contains eight PneuNet-type actuators radially situated around a 3D printed electronics canister. These propel the vehicle when inflated with water from its surroundings by impeller pumps; since the actuators are connected in two neighboring groups of four, the JenniFish has bi-directional movement capabilities. Imbedded resistive flex sensors provide actuator position to the vehicle's PD controller. Other onboard sensors include an IMU and an external temperature sensor. Quantitative constrained load cell tests, both in-line and bending, as well as qualitative free-swimming video tests were conducted to find baseline vehicle performance capabilities. Collected metrics compare well with existing robotic jellyfish.

SELF-CONTAINED SOFT ROBOTIC JELLYFISH WITH WATER-FILLED BENDING ACTUATORS AND POSITIONAL FEEDBACK CONTROL

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

1 INTRODUCTION

Fig. 1. The JenniFish during ocean testing off Delray Beach, FL.

This thesis work was started during the fall 2015 semester and pertains to the design, fabrication, control, and testing of a free-swimming soft robot (Fig. 1). Christened the JenniFish, this vehicle was first conceived as an eventual solution in low frequency, low power sensing applications. Current options for low power marine monitoring devices, such as gliders and various deployable buoy systems, provide researchers with much needed ocean data. Fresh water monitoring systems help maintain the health of natural and man-made resources such as lakes, rivers, water treatment plants, and aquatic fitness centers. Properly adapted, the JenniFish could be a small, cost-effective addition to current choices in both of these markets. Since its inception, the possible uses for future JenniFish iterations have grown beyond water quality monitoring; one day they could be implemented in swarm robotics or used as a STEM classroom learning resource (Fig. 2).

Most importantly, as soft robotics is a relatively new research field, the full potential of soft robotic vehicles is still being discovered. Hopefully, this thesis work will help contribute to the development of other novel soft robotic platforms.

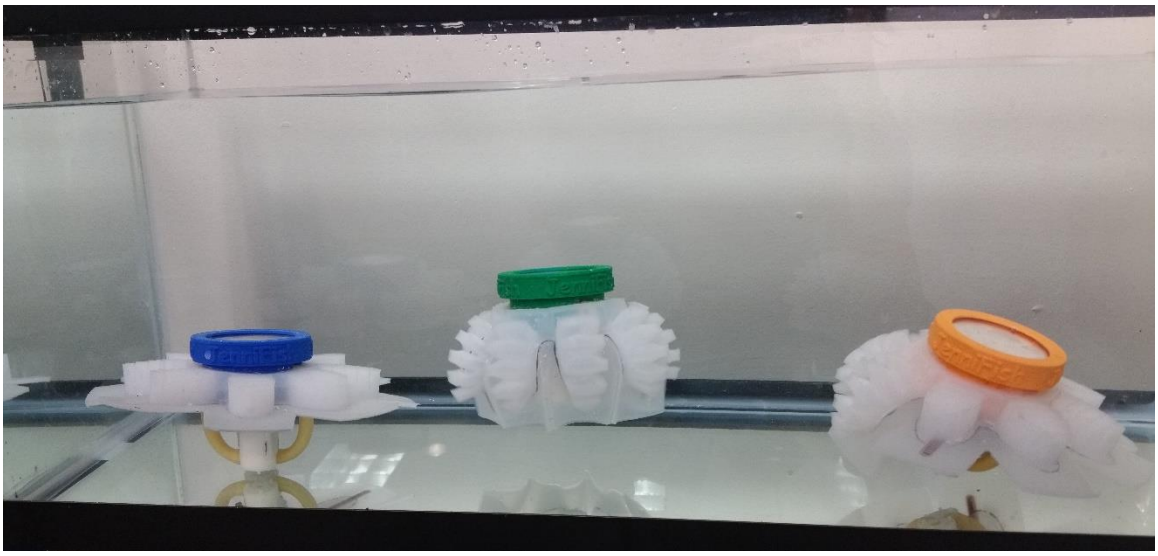

Fig. 2. JenniFish are relatively easy and affordable to manufacture.

1.1 Goals and Objectives

The purpose of this thesis project was to create a novel soft robotics platform. The goal was to design, build, and test a self-contained proof of concept robot. Its two main objectives were: (1) to implement a modified form of soft actuation to mimic an underwater propulsion method seen in nature and (2) then to test its capabilities, both qualitatively and quantitatively, for comparison to other propulsive methods used in similar systems.

1.2 Literature Review

A literature review was conducted to facilitate vehicle design, construction, and testing. It focused largely on how to (1) utilize current soft robotics knowledge, especially prior research into PneuNet actuator geometry/design, (2) achieve similar

swimming characteristics exhibited by a jellyfish, and (3) employ prior research done regarding the optimum size, shape, etc. chosen for existing robotic jellyfish.

1.2.1 Soft Robotics and PneuNet Actuators

Soft robotics was the chosen research area for this thesis work because it is a relatively new field with ample undiscovered potential. Much of this field's initial motivation centered on applications involving humans in close proximity to machines; the compliant-material nature of soft robots lessens the difference in elastic modulus, and often weight, between humans and robots, helping abate the robot's inherent ability to cause injury through contact [1]. This provides an additional advantage of reducing the amount of computing power required to maintain safe human-robot interaction and opens up numerous opportunities in wearable industries [2]. Researchers have also realized that compliant materials can make abstruse tasks become more viable because they require less complex control to accomplish delicate maneuvers, leading to significant work being done in areas involving grasping and other assistive gestures [3]. Furthermore, as researchers widen their focus, soft robotics is pushing the envelope in biomimicry [4]. Customarily, rigid robots do not enjoy the same kinematic freedom as soft robots [5]. Complex natural movements, such as those attained by animal muscles and marine life, are now starting to be imitated by soft robotic actuation [6]. Soft robots are successfully maneuvering through difficult terrain and obstacles, such as tight spaces, that challenge less materially diverse machines [1]. These great accomplishments are made possible by the multitude of soft actuation methods in existence today.

The number and type of soft actuation methods available today encompasses a diverse selection. Some of this selection includes: fiber-reinforced (McKibben-type) and

PneuNet-type ‘pneumatic muscle’ actuators, dielectric elastomer actuators (DEA), and ionic polymer metal composites (IPMC) [1]. Significant progress has been seen since the first usage and testing of artificial muscle actuators [7]. Utilization of the many soft actuation methods will continue to improve as new modeling and control protocols are created [8]. More efficient and cost effective production methods combined with user-friendly interfaces will help to further increase the popularity of soft robotics [9]. This research, which focuses on modifying pneumatic network (PneuNet) type actuators to simulate the underwater propulsion style emblematic of jellyfish, will hopefully contribute to such advances.

As a fundamental type of soft actuators, PneuNets have proven to be very versatile and propitious. Developed and publicized by Harvard University’s Whitesides Research Group, flexible pneumatic network actuators display relatively fast actuation times and perform well when tasked with bending mechanics such as grasping [10]. They consist of one or more elastomeric materials, have a protractile top layer, a reinforced inextensible bottom layer, and internal networks for fluid distribution [10]. A combination of geometry and material properties dictate these actuators’ range and variety of motion, so intended implementation is of primary consideration during design. Lamentably, predictive simulation of this actuator can be very complex due to the elastic nature of the actuators and the hard-to-characterize actuation medium flow; sometimes, trial and error is the most reliable design tool [10]. Fortunately, PneuNets and other types of soft actuation have been tested in underwater designs.

Soft Robotics has been seen applied to mimic swimming propulsion seen in nature. Fiber-reinforced bending actuators have been applied to reproduce undulating and

jetting propulsive methods seen in marine species such as Manta rays [11]. Octopus-like grasping was achieved by a semi-soft robot with silicone encased tentacles [12]. The flipper-like leg motions of a sea turtle were mimicked with a smart soft composite structure that incorporated shape memory alloy technology [13]. The PneuNet actuator type has been applied to underwater propulsion to mimic the wriggling motion of fish [14]. However, the author is not aware of PneuNets having been applied to mimic Jellyfish movement despite numerous other type of actuation methods having been employed in robotic jellyfish designs (see section 1.2.3). For this reason, among others, it was chosen as the basis for actuation in this research.

1.2.2 Jellyfish Swimming Characteristics

Portraying jellyfish propulsion accurately was of primary interest in this thesis work, so several theories were investigated to gain a general understanding of how jellyfish swim. It is commonly accepted that many jellyfish species use jet propulsion to generate thrust; however, it is not the only useful thrust generating mechanism despite some contest [15]. Jellyfish propulsion has been linked to bell shape and can be separated into two categories: jetting and rowing [16]. Jetting, witnessed in oblate medusae, is considered to be the faster of the two propulsion methods [16]. The slower rowing, or paddling, method, observed in oblates, is considered to be drag-based and contributed to the generation of vortices [16]. According to relatively recent research, jellyfish propulsion is enhanced by passive energy recapture [17]. Jellyfish have been reclassified as very efficient swimmers, especially since the definition of an energy consumption coefficient independent of body size [18]. Interestingly, it has also been suggested that swimming propulsion, including the type exhibited by jellyfish, is actually suction-based

[19]. This analysis in conjunction with the exploration of current robotic jellyfish systems was considered when debating different methods of actuation to implement in this project.

1.2.3 Existing Robotic Jellyfish

Past research has applied several methods of actuation to mimic the appearance, motion, and underwater propulsion methods seen in jellyfish species, most commonly the A. victoria and A. aurita jellyfish species. Papers were read on the following methods: shape memory alloys (SMAs) [20] [21] [22], hydrogen-fuel-powered SMA [23], ionic polymer metal composites (IPMCs) [24] [25] [26], engineered tissue [27], electromagnetic actuation (EMA) [28], tension-spring mechanisms [29], and iris mechanisms [30]. Conclusions made by these researchers were considered while designing the JenniFish.

Regard was given to past designs for their experience optimizing robot attributes, such as size, number of actuators, etc. Scaling was considered to be an important factor in many of the robotic jellyfish designs. The Robojelly has a bell diameter of 164mm, and the A. aurita species can be as large as 260mm [20]. Actuator placement was also given attention in most designs. Both the Robojelly and the IPMC vehicle used eight actuators extending radially outward, because this was found to most accurately mimic jellyfish bell contraction [20]. Furthermore, this type of actuator placement was determined to mimic the more efficient but slower form of jellyfish locomotion known as “rowing” [24]. Prior research also determined that passive flap inclusion increased swimming efficiency [20]. Moreover, it was found that a segmented flap design was the most

proficient [20]. The literature review on robotic jellyfish also revealed common tabulated vehicle characteristics.

Common metrics for comparison included: swimming speed, proficiency, thrust force, and power consumption. The swimming speed of A. victoria jellyfish averages 20 mms^{-1} [22]. Comparatively, the study done on a robotic jellyfish with four IPMC actuators reported an average swimming speed of 0.77 mms^{-1} which nearly doubled upon the addition of four more IPMC actuators to 1.5mms^{-1} [22]. A. aurita jellyfish attain a 0.25s^{-1} proficiency whereas Robojelly attained a 0.19s^{-1} proficiency [20]. Past research was also very valuable in determining the types of testing procedures that should be used.

Two methods of determining thrust production were seen during literature review. The first involved theoretical modeling based on experimental measurements [20]. The second involved exact thrust force measurement via load cell testing [24]. Virginia Tech reported the average thrust produced by different configuration of its shape memory alloy actuated robotic jellyfish (Robojelly); the configuration including a segmented bell with flap produced a maximum average thrust of 3.90mN [20]. Without a segmented bell but still including a passive flap, the maximum average thrust reported was 1.80mN [20]. Power consumption varied depending on the biomimetic jellyfish system, from as little as 1.14W to as much as 17W. Also, the effects of implementing a control method on Robojelly performance, including thrust production and power consumption, were examined [31].

2 VEHICLE DESIGN

After completing a literature review, several design hypotheses were formed and used to construct a vehicle body model in SolidWorks. Then, the SolidWorks Mold Tools Toolbox was used to generate a three-part mold. This mold was 3D printed and used to make a preliminary vehicle body for design verification. Following preliminary testing, improvements were made to the vehicle's body and modifications were made to accommodate vehicle electronics in Revision #1. After both constrained and free-swimming tests were conducted on Revision #1, a final set of design upgrades were made. Revision #2 was the last vehicle design that the scope of this thesis warranted. The following sections provide more details regarding each of these three design stages.

2.1 Preliminary Design

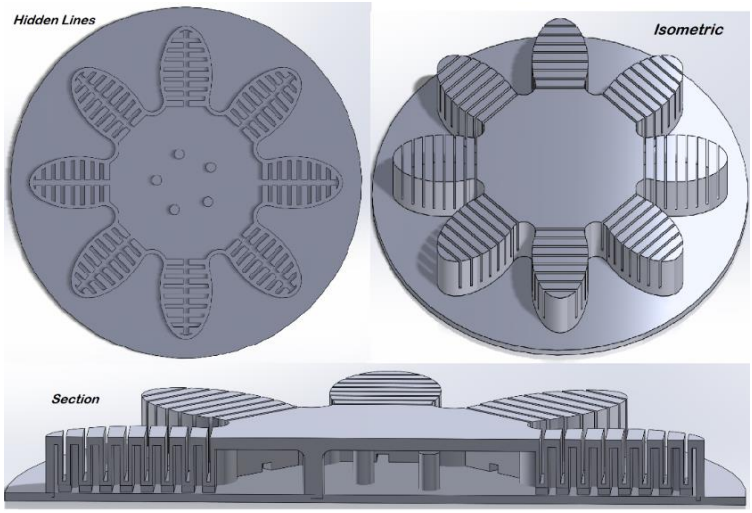

Fig. 3. Views of the preliminary SolidWorks model.

The SolidWorks model views in Fig. 3 display the Preliminary JenniFish Design. This model reflects several major design considerations, including: vehicle size, actuator

type, actuator number, actuator placement, actuator size, and cavity dimensioning. These choices were made based on prior work regarding biomimetic jellyfish-like vehicles and PneuNet actuators. Many vehicle attributes, including the JenniFish's maximum diameter, were desired to be on the same scale as prior research and actual jellyfish. For this reason, an initial total diameter of 180mm (including flap) was chosen. Prior research provided evidence supporting passive flap inclusion and segmentation. Therefore, the material that connects the eight actuators was extended slightly beyond the length of the actuators and only partial testing was conducted without flap segmentation. Bell diameter was used as a reference when determining actuator length. Previous studies using linear actuation methods influenced the decision to use eight actuators with a radial placement pattern. When designing actuator shape, existing knowledge on PneuNet architecture was utilized.

One of this thesis' objectives was to implement a modified form of soft actuation to mimic an underwater propulsion method seen in nature. Actuators tested by the Whitesides Research Group were uniform in width and bent equally along their length [10]. It was hypothesized that using an oblong-shaped actuator, rather than a rectangular-shaped one, would produce less curvature at the actuator ends. Reduced bending at the actuator tips would better mimic actual jellyfish bell shape and swimming motions, theoretically enhancing their propulsive characteristics underwater. The desire to mimic two other jellyfish characteristics, passive relaxation and neutral buoyancy, facilitated the selection of actuation medium, actuation method, and body material.

Many of the design aspects are interrelated including actuation medium, actuation method, and body material. PneuNet actuators relax passively due to their elasticity,

which accurately mimics Jellyfish relaxation [20]. As jellyfish are neutrally buoyant creatures, water was the desired actuation medium of the JenniFish [20]. Body material selection directly effects elasticity, buoyancy, and cavity dimensioning. These factors combined with practical concerns of availability and ease of use made Ecoflex 00-30 reinforced with paper an advantageous choice. The performance of the materials detailed in "Pneumatic Networks for Soft Robotics that Actuate Rapidly" as well as classmate fabrication experience with 3D printer tolerance and soft materials were considered when assigning cavity dimensions [10]. The preliminary design was fabricated and wave tank tested (Fig. 4) to verify the many design and fabrication hypotheses made.

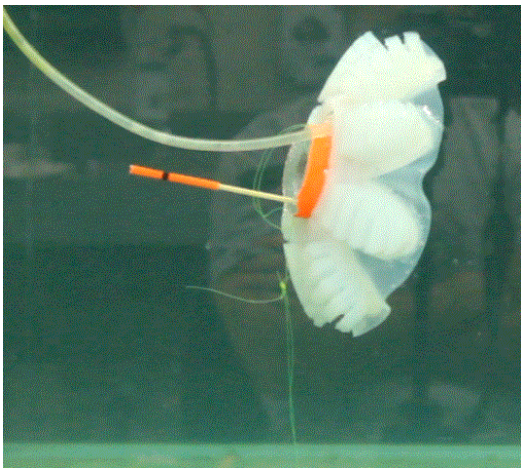

Fig. 4. Preliminary design verification.

2.2 Revision #1 Design

After experiencing the preliminary construction process and conducting preliminary wave tank tests to ensure the design was feasible, it was concluded that the basic design would work. Since the fundamental design was confirmed, the bulk of Revision #1 Design focused on making the JenniFish self-contained and capable of two-sided actuation. First, final details regarding actuation method were decided. Water was the desired actuation medium due to buoyancy, but it was not immediately apparent if the

water source should be contained or external. It was determined that impeller pumps would reasonably portray the passive relaxation seen in jellyfish, conveniently remove any need for valves (thus reducing electronic complexity and space requirements), and keep system power consumption to a minimum. Basic characterization of pump requirements associated with the preliminary design were determined during testing. Thus when inexpensive, commercially available self-priming submersible impeller pumps were found capable of meeting minimum performance requirements, using an external water source became the favored design choice.

Impeller pumps fill the JenniFish actuators when powered and let water vent to the surroundings when not powered. The simplistic nature of this design helps reduce vehicle cost and logistics; since only a proof of concept vehicle was within the scope of this thesis, concerns for long-term vehicle maintenance and usage associated with an external water source, like bio-fouling and corrosion, were less important. Future iterations might be able to utilize the actuation means as a way to sample water quality monitoring or they could be adapted to use a contained water source for actuation. The volume and fill rates of the current system could be used to design alternative systems if deemed necessary.

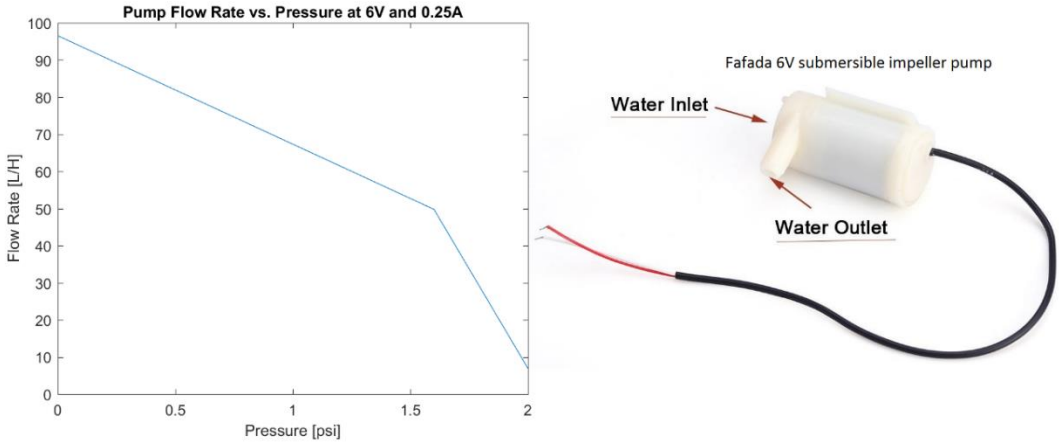

Fig. 5. Pump selected for use in this project and its measured performance.

Fig. 5 contains a simple graph of the chosen pumps' performance during testing. After pump selection, it was determined that most of the central fluid chamber present in the preliminary design should be replaced by a void space. This space was then allocated for the electronics housing to fill. Ideally, the JenniFish would have two-sided actuation controlled by position feedback, which gave rise to the idea of imbedded resistive flex sensors. Then, fluid distribution was divided in two, creating separate fluid chambers each connecting to four adjacent actuators. This was designed to allow for two-sided actuation, theoretically providing the JenniFish with directional control. The author has not seen this in other studies. An external temperature sensor was also necessitated at this point in the design phase as a way to facilitate water quality monitoring.

Other minor modifications to both the preliminary model and mold were implemented in the first design revision as well. Table 1 summarizes mold dimensions for the first revision. The max diameter was increased from 180mm to 210mm after experimentally determining that the passive flap could be larger (bell diameter of 160mm remained constant between iterations); this was done despite it making the JenniFish's flap percentage larger than that found in natural A. aurita and the Robojelly [20]. Considering pump selection, the fluid distribution channels were increased in width by 1mm with the intention of reducing required inflation pressure and thereby the time required to fully actuate the JenniFish. Also, space for pump hoses and sensor wires were accounted for in revision #1. Finally, mold tolerance issues were addressed after experimenting with the available 3D printer, culminating in the following SolidWorks model seen in Fig. 6. The hidden lines view shows the network of channels inside the JenniFish, with the wider middle channel being for fluid distribution.

Table 1. Summary of revision #1 mold dimensions in millimeters.

Fluid Chambers	2.50 wide x 14.5 tall
Distribution Channels	5.00 wide x 4.00 tall
Walls	2.25 thick
Ceiling	3.25 thick
Bottom	2.25 thick
Maximum Diameter	210

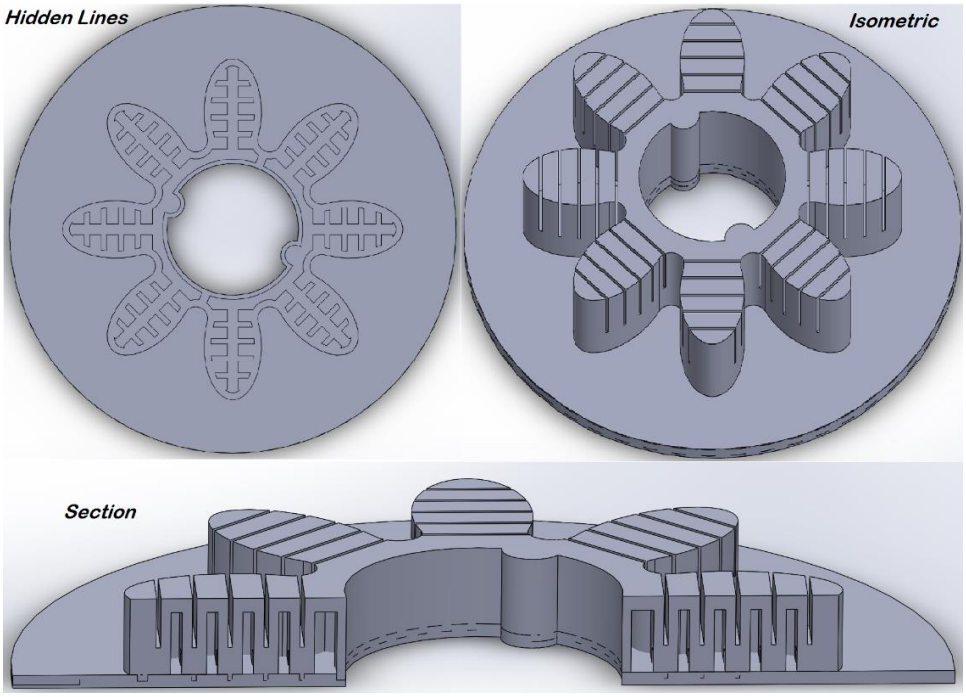

Fig. 6. Views of Revision #1 SolidWorks model.

After several failed 3D printable electronics can designs (including a vacuum seal idea and screws through the Plexiglas top) a 3D printable mason-jar-style electronics housing was designed in SolidWorks to fill the middle of the revised JenniFish design. An O-ring groove and O-ring help make a water-tight seal with the 66.5mm diameter circular Plexiglas top that is held down by a top that is threaded onto the canister. To be free-swimming, the JenniFish's battery and motherboard had to fit inside the housing. A custom electronics design was completed based on pump voltage requirements, housing

space constraints, price, and availability. Additionally, a magnetic and visual interface was devised so that limited communication with the vehicle could be accomplished without opening its electronics canister. For more details, refer to the vehicle control section.

Paper was used to reinforce the bottom layer in the preliminary design, and, because it worked, the author did not try alternative reinforcement material, such as polyester fabric. Concerns regarding the anisotropic weave of certain fabrics made it too risky to implement in the first revision due to time constraints. A small test PneuNet actuator was fabricated with polyester bottom reinforcement because it was hypothesized that the fabric would be a more durable reinforcement since the paper was prone to ripping inside of the cast silicone. Preliminary testing produced positive results, so polyester fabric was used in Revision #2. Upon completion (see Fig. 7), Revision #1 had a dry weight of 380g (plus or minus 5.67g) and a 160mm contracted diameter, which is approximately 24% contraction compared to the 50% contraction seen in A. aurita species. It has two Spectra Symbol resistive flex sensors which were imbedded in the bottom flap during the fabrication process. These flex sensors significantly reduced the bending capabilities of the two actuators they bisected. In revision #2 this was remedied by removing the stiff plastic casing on the flex sensors.

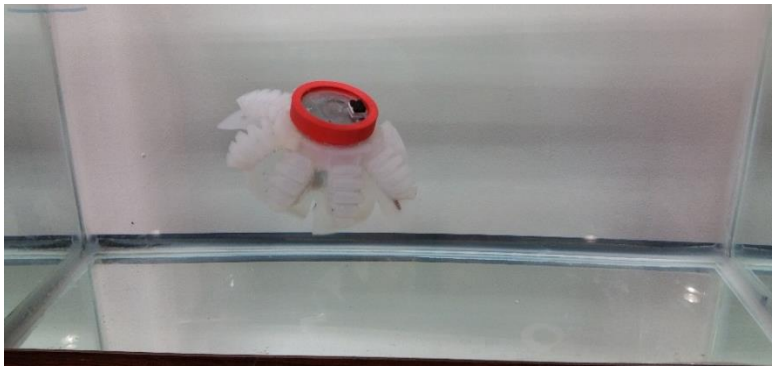

Fig. 7. Revision #1 freely swimming in lab aquarium tank.

2.3 Revision #2 (Final) Design

Polyester bottom reinforcement and uncased flex sensors were successfully implemented in the final design iteration. Apart from these two physical improvements, the upgrades in Revision #2 were all electronics and control based (Fig. 8). The motherboard was greatly enhanced with the addition of a 9 DoF IMU, onboard flash memory, battery voltage feedback, contacts for real-time serial communication during constrained testing, and repositioned light and magnetic interface sensors. A PD controller was implemented and tested on this iteration as well. See the vehicle control section for more details.

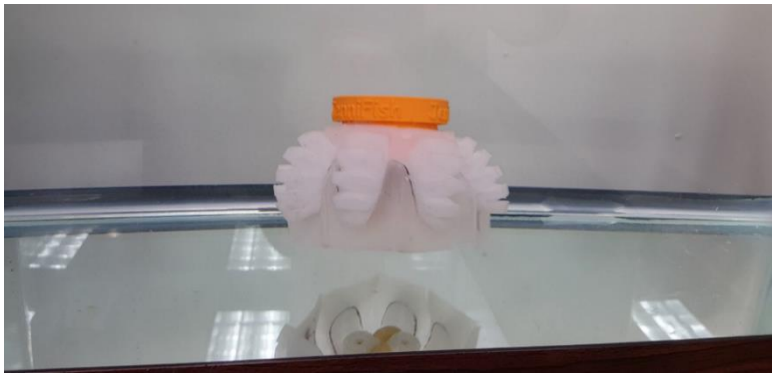

Fig. 8. Revision #2 freely swimming in lab aquarium tank.

3 VEHICLE FABRICATION

Constructing a JenniFish involves a number of stages, including: a material and part ordering process, waterjetting, 3D printing, body fabrication, motherboard population, and a waterproofing routine. This section contains details of this process in the form of a How-To guide. It has been ordered in list format for simplicity, but asterisks denote stages that can be done simultaneously.

3.1 Ordering

Table 2. Itemized JenniFish parts list.

Item	Details	Vendor	Quantity	Price
Fafada Water Pumps	DC, Mini Submersible, 120L/H Max Lift 3.6ft	eBay	2	$1.99
Buna-N O-ring	036, 70A Durometer, 2-3/8" ID, 2-1/2" OD, 1/16" width	Amazon	1	$0.15
Ecoflex 00-30	400mL cartridge (1.0lb net)	Reynolds Advanced Materials	1	$23.70
Resistive Flex Sensors	Spectra Symbol	Amazon	2	$7.95
PJRC Teensy 3.2 Microcontrollers	without pins	Amazon	1	$25.95
Plexiglas	12"x12"x1/8"	True Value	1	$5.00
Latex Tubing	Grafco, 1/4" ID, 3/8" OD, 1'	Amazon	1	$0.60
Clear RTV Silicone	Permatex, 3oz	Amazon	1	$5.72
Motherboard Rev. 2	Including Parts (IMU, Flash, temperature, 9V, wires, etc.)	Advanced Circuits	1	$110.00
Extras	3-D printing material, epoxy resin, foam, electrical tape, magnet, polyester fabric, etc.	various	1	$9.00
			Total	$200.00

This stage should be done first, as it helps organize the builder and his or her timeline; product lead time can help to decide the next build step. The COTS components required for the JenniFish, in addition to possible vendors and approximate price, are included in Table 2.

3.2 Waterjetting*

The JenniFish's electronics canister includes a circular, 66.5mm diameter Plexiglas piece, which was waterjetted in the FAU Boca Raton campus machine shop. An IGES file was delivered to the machinist along with a 305mm x 305mm x 3.18mm Plexiglas sheet. A sheet of this size contains enough material to make sixteen tops.

3.3 3D Printing*

The JenniFish requires several 3D printed components, which have been listed in Table 3 along with the type of plastic and 3D printer used in current JenniFish revisions. Printing directives vary between machines, so make sure to check bed size limitations, print quality settings, and tolerance adjustments. ABS is recommended for the electronics canister because acetone can be used to smooth plastic surface imperfections. In general, the SolidWorks model file is saved as an .STL file and uploaded into a printing software such as Cura. A G code file is then generated and sent to the printing machine.

Table 3. 3D printed JenniFish components

Item	Printer	Plastic	Quantity
Bottom Mold Piece	Ultimaker 2	PLA	1
Bottom Mold Disk	Ultimaker 2	PLA	1
Top Mold Piece	Ultimaker 2	PLA	1
Electronics Can	Axiom AirWolf	ABS	1
Electronics Can Top	Axiom AirWolf	ABS	1
In-line L.C. Mount	Axiom AirWolf	ABS	1
Bending L.C. Mount	Axiom AirWolf	ABS	1

3.4 Body Fabrication**

Body fabrication is a multi-stage process that involves layering Ecoflex-0030, reinforcement material, and any sensors that get imbedded in the body. For this stage access to a desiccator is required. Be sure to wear the proper personal protective equipment, such as gloves and eye glasses, when working with chemicals. The following list walks through the fabrication process.

1) Make stencil out of cardboard (or similarly stiff material) to trace actuator pattern onto bottom reinforcement, in this case polyester fabric. Be sure to use the true size of the pattern to make a stencil like that shown in Fig. 9.

Fig. 9. Bottom reinforcement should have this shape's outline.

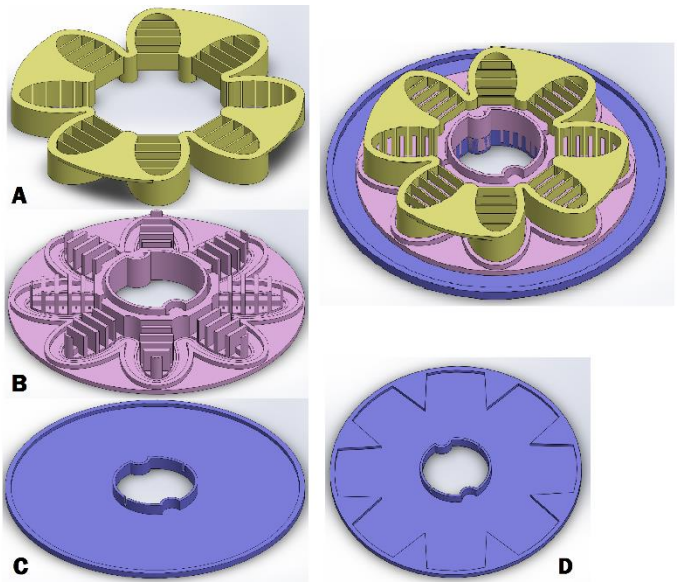

Fig. 10. SolidWorks model of three-part mold.

2) After 3D printing the three-part mold model (Fig. 10), make sure that the mold sections are smooth and free of surface defects. Sand the mold surfaces and lightly wipe with acetone if printed out of ABS. Ascertain that sensor slots were printed the appropriate size; if not, sand the edges until sensors fit well. The yellow (A) and pink (B) molds are put together and filled with Ecoflex separately from the purple (C) mold. Depending on printer tolerance, Ecoflex may leak out of the mold before curing. To prevent this, hot glue is applied along the bottom intersection of A and B. Spray all three mold parts (A, B, C) with a mold release, such as Ease Release 200. Be sure to allow appropriate drying time (about 20 minutes). Finally, prepare all sensors to be imbedded. Check to see that they are functioning and, if desired, remove any stiff casings (Fig. 11).

Fig. 11. Glue molds together (left), spray molds (middle), remove casing (right).

3) After the bottom reinforcement, sensors, and molds are ready, Ecoflex is measured, mixed, de-gassed, and poured. Ecoflex is a two-part formula that cures only when the components are mixed in a 1:1 ratio. Thoroughly mix the containers of Ecoflex and then measure out 100 mL of each Ecoflex part into separate cups. Next, pour the two parts together into a larger, around 400mL, container. Stir for at least one minute and put container into a de-gassing chamber

(Fig. 12). De-gas until no bubbles appear on the mixture's surface (approximately 5 minutes).

Fig. 12. Mix a 1:1 ratio of Ecoflex (left) and then de-gas (right).

4) Coat the bottom mold (C) with a thin layer of de-gassed Ecoflex. Place the bottom reinforcement on top of this layer. Pour additional Ecoflex only where sensors are to be added. Place sensors appropriate side up onto the extra Ecoflex. Finally, add another thin layer of Ecoflex over the sensors. Fill the previously hot-glued together tops molds (A and B) to the brim with Ecoflex. This process should use most, if not all, of the Ecoflex (Fig. 13).

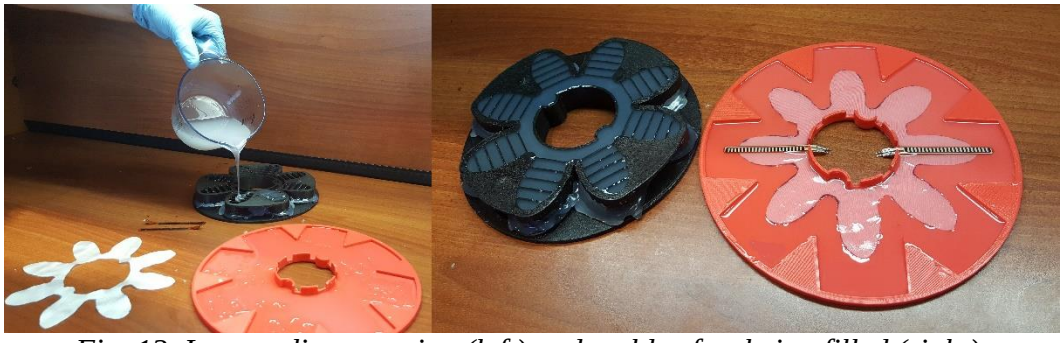

Fig. 13. Intermediate pouring (left) and molds after being filled (right).

5) Wait until the two separate pieces have cured (about 24 hours). Then, remove the hot-glue connecting the two top molds and peel them apart. Do not de-mold part (C). After the top silicone piece is de-molded, mix another 20mL total of Ecoflex in the same way as done in step 4. Then thinly coat the top of (C) with Ecoflex mixture. Carefully line up the top silicone piece with the bottom reinforcement

and lightly tap on the top of the silicone piece to ascertain contact with the wet Ecoflex. When the additional layer of Ecoflex cures, it will securely connect the two silicone pieces together, completing body fabrication (Fig. 14).

Fig. 14. De-molded top (left) and connected top and bottom pieces (right).

3.5 Motherboard Population**

Upon delivery, the motherboard will need to be tested and then populated with the Teensy microcontroller and all other components (LEDs, capacitors, resistors, flash memory chip, etc.). All soldering was done in the electronics lab under the guidance of its staff members Ed Henderson and John Kielbasa. Most of the motherboard components are surface mount parts, with the exception of the IMU which requires reflow soldering. The JenniFish electronics parts list and schematics are used to inform proper part placement. Be sure to match the board revision to the revision listed on the parts list and schematics.

3.6 Waterproofing Routine

This stage involves putting all of the previously constructed JenniFish parts together. Due to Silicone and Epoxy cure times, budget several days of time before scheduling tests. Be sure to work in a well ventilated area when using RTV silicone and epoxy. Let the following steps serve as a guide during this fabrication stage (Fig. 15).

Fig. 15. Materials needed for waterproofing routine.

1) Wires must be soldered to the three onboard sensors (one externally mounted temperature sensor and the two 55.9mm flex resistors imbedded in the silicone body). Each wire (7 total) should be approximately 102mm long. Ideally, the wires should be Teflon coated 24AWG. Apply heat shrink to all joints and strip and tin the ends that will be soldered to the motherboard later. If using the same electronics as Revision #2, it is suggested that the same wire color scheme be used.
2) The 3D printed electronics canister should be free of major surface imperfections, especially the O-ring groove. Remove any imperfections and wipe ABS with acetone. The through-holes on the bottom of the can need to be drilled open if they are printed partial closed; the print quality on the Axiom was generally not high enough to print unobstructed circular holes.
3) Sensor and pump wires enter the housing from the bottom. Silicone RTV is used to secure the silicone body to the 3D printed can and cover exposed wiring. First,

insert RTV silicone between the body and the can. Then after situating the pumps and pushing all wires through their respective through-holes, spread RTV silicone around the base of the pumps and all three sensors. Exposed wiring on any of these components could ruin them when submerged.

4) After the RTV silicone dries (wait at least 24 hours), epoxy is then poured into the bottom of the canister to ensure the through-holes are water-tight. Only a few milliliters are necessary. Make sure that the can is level during the drying process or the unequally distributed weight of the hardened epoxy will cause the JenniFish to have trim issues. A 1:1 fiber glassing epoxy was used successfully, but other pourable epoxies should work. Let the epoxy dry for about 24 hours before attempting the next stage of fabrication (Fig. 16).

Fig. 16. Coat can underside with RTV silicone (left) and pour epoxy in bottom (right).

3.7 Upload Code and Test

The last fabrication stage involves soldering a populated motherboard into the waterproofed body, uploading code to the microcontroller, and testing to ensure that fabrication was successful. All wires have an attachment point on the motherboard; refer to the electronics schematics when soldering to ensure proper connection. In order to upload code to the microcontroller, both the Arduino IDE and the Teensyduino add-on

for the Arduino IDE must be downloaded. Refer to Arduino literature on how to download the IDE and add-ons. If a user has no previous Arduino experience, this literature will also be a helpful reference for basic Arduino knowledge. A USB to micro-USB cable is needed to upload code to the JenniFish. It is not recommended that a battery be connected to the JenniFish while code is being uploaded. Connect the micro-USB as shown in Fig. 17.

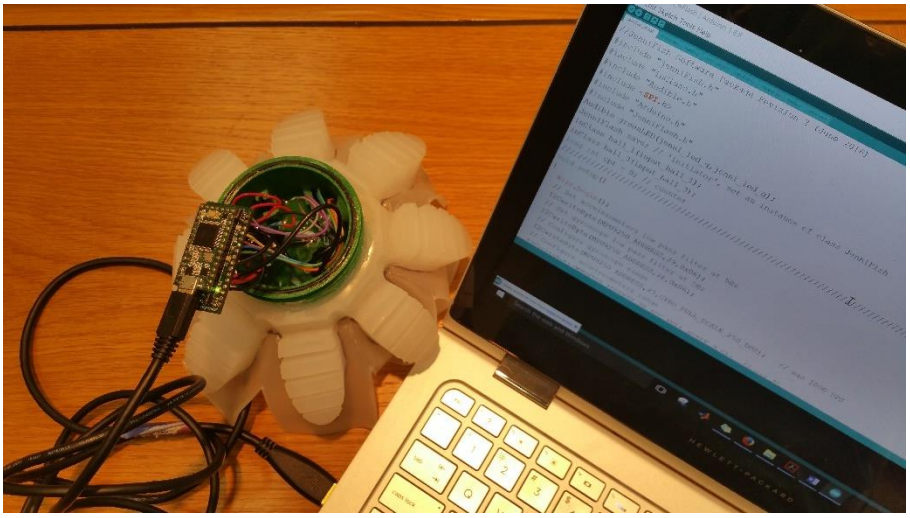

Fig. 17. Upload code to the JenniFish via a micro-USB cable.

Then add grease or toilet bowl wax to the O-ring and groove to seal the JenniFish electronics canister and water test (Fig. 18).

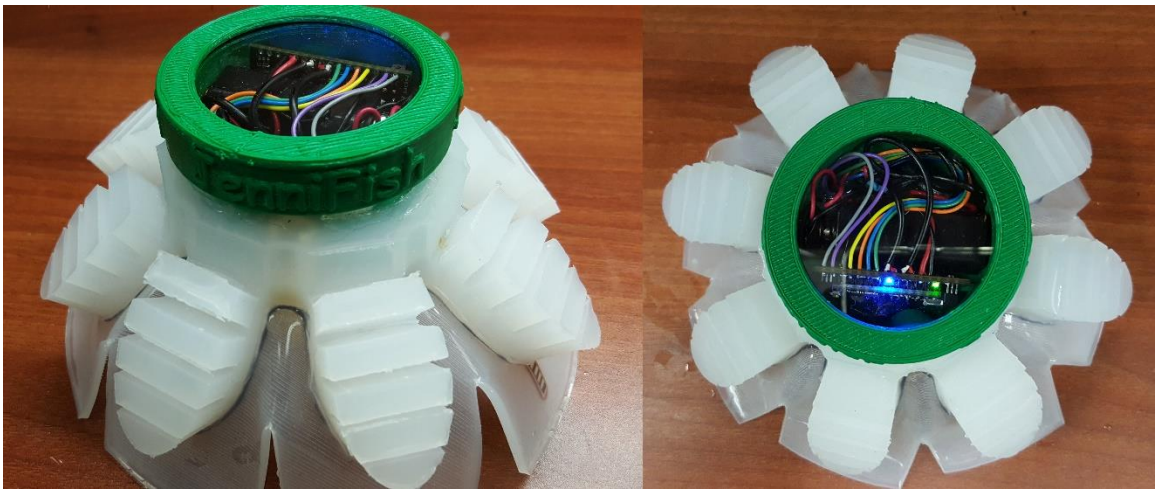

Fig. 18. Side view of sealed JenniFish (left) and top view (right).

4 VEHICLE CONTROL

JenniFish hardware and software details are included in this section, including schematics, axis assignment, transformation math, and pseudocode.

4.1 JenniFish Electronics

After determining that two 6V submersible impeller pumps would be sufficient to fill the actuators, a decision needed to be made on how to power the system. Considering this voltage requirement and the space constraints, in addition to price and availability, a 9V primary cell alkaline battery was chosen to provide system power via PWM control. The PJRC Teensy 3.2 microcontroller is very robust for its small size, has sufficient I/O pins, and can be coded using an Arduino IDE for familiarity, making it a good fit for this project. The Teensy 3.2 is mounted on a 45.7x22.9mm custom built motherboard designed by the FAU electronics lab; all components are surface mounted, and include a 5V linear regulator, MOSFETs, diodes, resistors, capacitors, op-amps, an Invensense MPU-9250 9 axis motion sensor, and a 16MB Flash memory chip (Fig. 19).

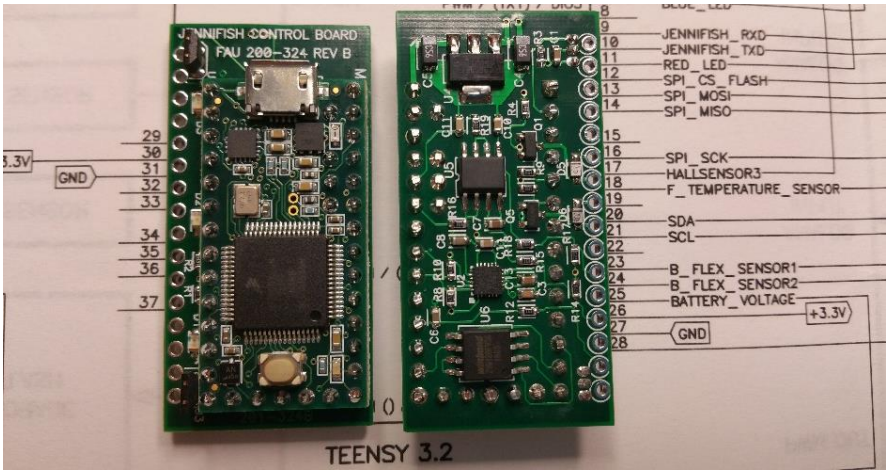

Fig. 19. Rev. 2 motherboard top view (left) and bottom view (right).

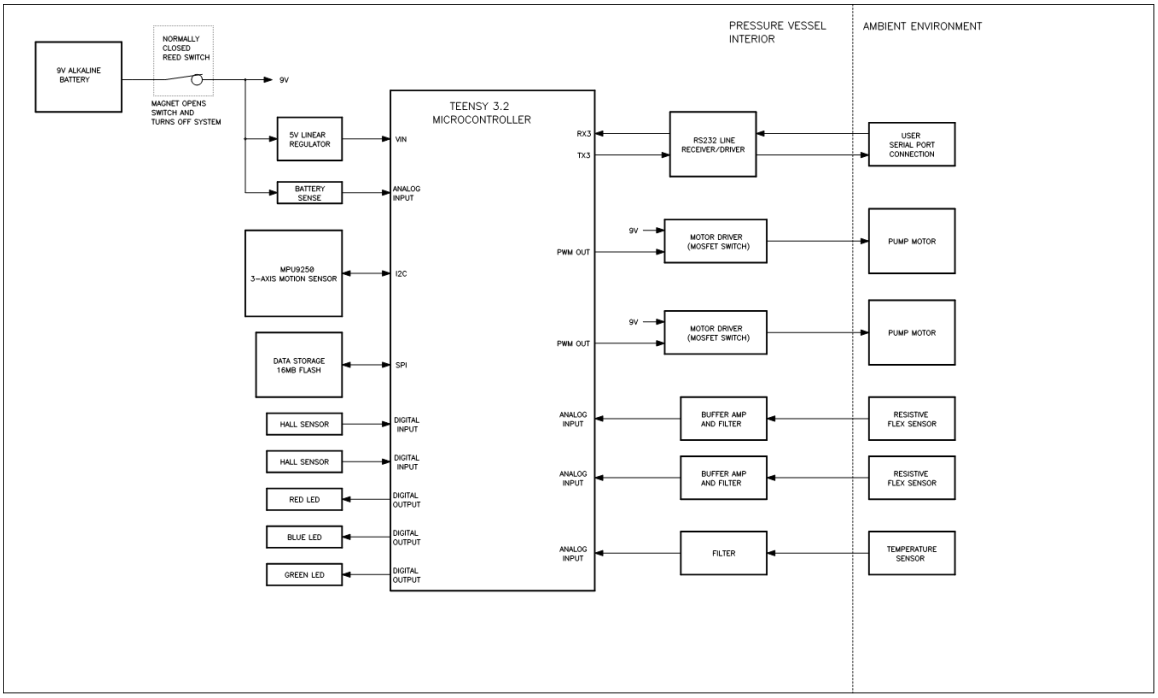

Fig. 20. Electronics block diagram

A block diagram of the electronics is included in Fig. 20 and more detailed electrical schematic are included in the appendix, section 9.1. As the schematic shows, two Allegro Microsystems, Inc. Hall Effect sensors (A3212EUA-T) were mounted on the motherboard as a way to communicate with the JenniFish without having to open the electronics housing. Several operational modes can be pre-programed onto the Teensy and chosen based on magnetic input provided by the Hall effects. Three LEDs (blue, green, and red) were included on the motherboard in addition to the Teensy's orange LED. All four LEDs provide users with a solid visual feedback interface. Currently, the blue LED is programmed to shut off when battery voltage drops below 6V. The green led is programmed to turn on when Flash memory is being recorded and blinks when the memory is full. The red LED was chosen to indicate IMU status; currently it is on if the IMU is operational and off otherwise. The Teensy's orange LED communicates program mode to the user with a blink pattern depending on magnetic input from the Hall sensors.

Temperature was collected with an MCP9701A. The IMU's accelerometer was calibrated and found to have biases within tolerance; however, the gyroscope was not. The part was not replaced on the motherboard used for Revision #2 due to time constraints. The accelerometer reports readings in relation to the IMU's own internal axis. To make the values more meaningful to the JenniFish, a transformation matrix was applied to the accelerometer data to so that it reported readings with respect to the coordinate system chosen for the vehicle. Fig. 21 shows the IMU and JenniFish coordinate systems with respect to one another and the vehicle body.

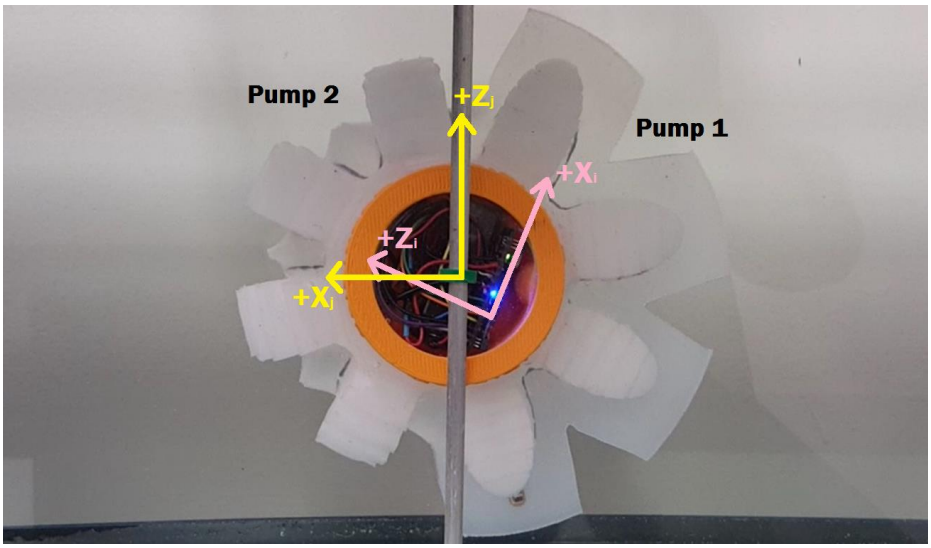

Fig. 21. IMU axis (pink) and local JenniFish axis (yellow) orientation.

The local JenniFish coordinate system was oriented by Denavit-Hartenberg convention, which is commonly used in robotics but not underwater vehicles. The transformation on the accelerometers values can be broken down into three steps (subscript 'i' represents IMU and subscript 'j' represents JenniFish local).

(1) Make $x_i = z_j$, $z_i = x_j$, and $y_i = -y_j$ with the following matrix:

$$\begin{bmatrix} x_j{}'' \\ y_j{}'' \\ z_j{}'' \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}$$

(2) Rotate by θ = 15.67° (measured in SolidWorks):

$$\begin{bmatrix} x_j{}' \\ y_j{}' \\ z_j{}' \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} x_j{}'' \\ y_j{}'' \\ z_j{}'' \end{bmatrix}$$

(3) Translate toward battery and USB connector (distances from SolidWorks in mm):

$$\begin{bmatrix} x_j \\ y_j \\ z_j \end{bmatrix} = \begin{bmatrix} 8.27 \\ 0 \\ 6.37 \end{bmatrix} + \begin{bmatrix} x_j{}' \\ y_j{}' \\ z_j{}' \end{bmatrix}$$

(4) The result was used in the Arduino code to transform accelerometer data:

$$\begin{bmatrix} x_j \\ y_j \\ z_j \end{bmatrix} = \begin{bmatrix} 8.27 \\ 0 \\ 6.37 \end{bmatrix} + \begin{bmatrix} -\sin\theta & 0 & \cos\theta \\ 0 & -1 & 0 \\ \cos\theta & 0 & \sin\theta \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \quad (1)$$

The transformed accelerometer readings from (1) were collected and post-processed in MATLAB to roll and pitch values with (2) and (3). Since the JenniFish coordinate system is atypical by conventional vehicle axis assignment, roll on the JenniFish refers to the rotation about x_j (axis perpendicular to the vehicle's centerline, which is the line of symmetry that splits the actuator groups). Pitch on the JenniFish refers to the rotation about z_j (axis along vehicle centerline).

$$Roll\ (about\ x_j\ axis)\colon \tan\emptyset = \frac{a_{jz}}{\sqrt{(a_{jx})^2 + (-a_{jy})^2}} \quad (2)$$

$$Pitch\ (about\ z_j\ axis)\colon \tan\vartheta = \frac{-a_{jx}}{-a_{jy}} \quad (3)$$

4.2 JenniFish Software

The JenniFish software package is multifaceted and was structured to make the vehicle as user-friendly as possible. One of its tasks, as mentioned earlier, is to control user interface settings, including how magnetic input gathered by the two Hall sensors is

interpreted and how visual feedback is communicated to the user via onboard LEDs. It also allows for various vehicle operation modes. This is especially useful for testing different control methodologies. Operating frequency is flexible, though it was found that frequencies faster than 100Hz hurt real-time communication with Simulink. Functions and objects were implemented to keep the main loop code organized and intelligible to others. The pseudocode in Fig. 22 gives a basic outline of the JenniFish code structure; however, the full Arduino code has been included in the appendix, section 9.2.

```
JenniFish Pseudocode
-Request values for PD gains once per power cycle
-Read battery voltage and report if low
-Read flex sensors and map values from 0 to 100
-Read temperature sensor
-Read IMU Data and report if error occurs
-Check Flash status and report if recording or full
-Check for user input to offload or erase memory
-Check Hall sensors & decide operational mode
-Run operational mode
-Send PWM commands to pumps
-Print data
-Maintain constant loop time
-Functions: controllers (timed swim and PD)
           waveforms (square, saw tooth, multistep, sine)
           flex sensor calibration and mapping
           I2C read and write for IMU
           Print values to serial
-Objects: program mode interface
         flash memory
```

Fig. 22. Software package pseudocode.

Two control methodologies were created for the JenniFish – one that was purely timing-based and another which relied both on timing and position feedback. This second methodology was realized with a PD controller and various desired control signals like square waves. Functions were made for the PD controller and desired waveforms in the Arduino code. Fig. 23 is the function that was coded for the controller. Fig. 24 contains the function that was coded to generate a square wave as the desired control signal and an example operational mode program for square wave tracking.

```
void controller(){
  e1 = thetaD1 - f1;
  e2 = thetaD2 - f2;
  eDot1 = (e1 - eLast1) / (jenni_loop_period/1000);
  eDot2 = (e2 - eLast2) / (jenni_loop_period/1000);
  p1 = (kp * e1) + (kd * eDot1);
  p2 = (kp * e2) + (kd * eDot2);
  pump1 = constrain(p1, 0, 100);
  pump2 = constrain(p2, 0, 100);
  eLast1 = e1;
  eLast2 = e2;
}
```

Fig. 23. Arduino function created for PD control.

As can be seen, a basic proportional derivative (PD) controller was implemented for each half of the JenniFish. The controller's output is limited from 0 to 100 which translates to 0 to 6V after PWM. Since the JenniFish has impeller pumps, actuator relaxation cannot be controlled. During tuning, the system displayed very underdamped characteristics. One of the desired control waveforms tested with the JenniFish's PD controller was a square wave. The function on the left was written so that square waves of different frequency and amplitude could easily be generated. In the program mode, squareWave(1000, 3000, 100, 0); produces a waveform with an amplitude of 100 for one second and then an amplitude of zero for two seconds every three second period.

```
////////////////////////////////////////////////////////.
void squareWave(int cT, int rT, int cLimit, int rLimit)
{
  if (millis() - t >= rT)
  {
    t = millis();
  }
  if (millis() - t <= cT)
  {
    thetaD = cLimit;
  }
  if (millis() - t > cT)
  {
    thetaD = rLimit;
  }
}
////////////////////////////////////////////////////////.
```

JenniFish code for square wave tracking mode

```
case 4: // Square Wave Tracking
  if (millis() - startt < 60000)
  {
    squareWave(1000, 3000, 100, 0);
    thetaD1 = thetaD;
    thetaD2 = thetaD;
    controller();
    saveVariables();
  }
  else pCase = 0;
  break;
```

Fig. 24. Code written to allow square wave tracking.

5 VEHICLE TESTING

Four hypotheses motivated vehicle testing:

(1) The JenniFish's actuation method, chosen to mimic that of oblate medusae, will prove capable of propelling the vehicle upward against slight negative buoyancy.

(2) Implementation of a positional feedback PD controller will increase net thrust when compared to purely timing based control.

(3) Increasing cycle frequency, and consequently limiting relaxation range, will increase net thrust when compared to a frequency which allows full relaxation.

(4) The JenniFish's two-sided actuation will enable bi-directional heading abilities, which will also be enhanced by controller and cycle frequency adjustment.

Due to the comprehensive nature of these hypotheses, both quantitative and qualitative testing were deemed important. Constrained load cell testing – both in-line and bending – was chosen to provide quantitative evidence due to its controlled and repeatable nature. Video recorded free-swimming tests – both tank and open-water – were the preferred method for providing qualitative evidence.

Preliminary load cell and free-swimming tests were done with JenniFish Revision #1 to discover baseline vehicle performance capabilities. These preliminary tests provided a solid foundation of evidence supporting hypothesis 1, and lead to the formulation of hypotheses 2, 3, and 4. Testing with JenniFish Revision #2 was then done to ascertain the validity of all four hypotheses. Speed, proficiency, power consumption, and net thrust production were metrics collected during testing for their usefulness in helping determine

hypotheses as well as in comparing the JenniFish with other robotic jellyfish systems. The following sub-sections provide test setup in greater detail, including how the hypotheses drove test setup and selection.

5.1 Constrained Testing

Two types of constrained load cell testing were done: in-line and bending. Since load cell testing allows for repeatable thrust force measurement, albeit at zero speed, and load cell equipment was made available, this method was chosen over the theoretical calculation of thrust production. In-line load cell testing quantified upward thrust production, providing a means to discern hypotheses 1, 2, and 3. Bending load cell testing was done to quantify hypothesis 4. The two different load cells were calibrated prior to testing and sensor drift was documented for each test set. Furthermore, all JenniFish Revision #2 constrained testing was powered with 9V from a power supply to ensure consistency in measurement collection throughout the testing process.

5.1.1 In-line Load Cell Testing

A Futek 2lb. JR S-Beam load cell was used with the mounting set-up shown in Fig. 25 for both JenniFish Revision #1 and Revision #2.

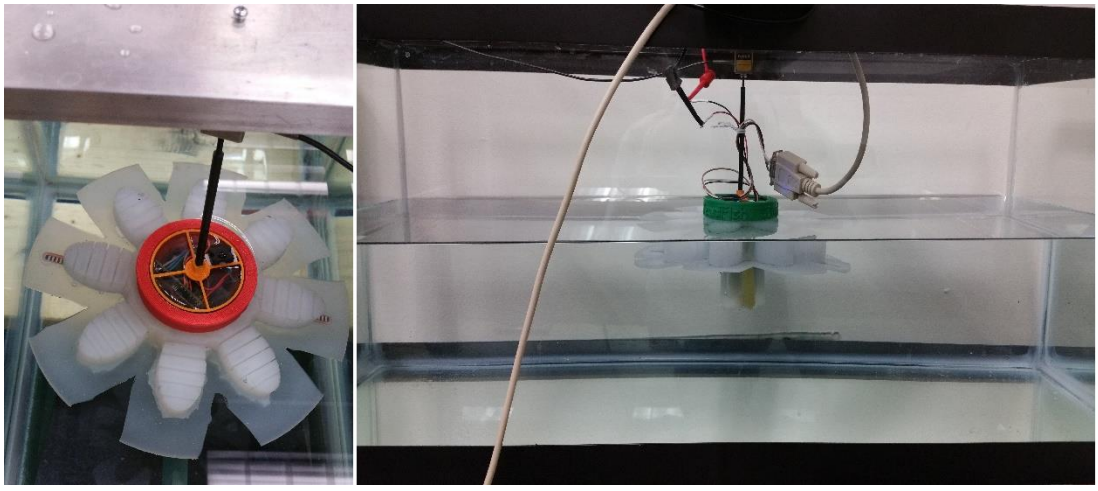

Fig. 25. Rev. 1 (left) [32] and Rev. 2 (right) during in-line load cell testing.

The load cell is connected to a Futek CSG110 Strain Gauge Universal Amplifier driven by a BK Precision 1672 Triple Output DC Power Supply set at 17V. The amplifier's signal output was brought into Simulink using a National Instruments BNC-2090A Data Acquisition board. During JenniFish #2 tests, serial data was also acquired real-time through Simulink during load cell testing so that flex sensor values, controller error, pump signal, and desired waveform could all be correlated to the load cell data, aimed at revealing the truth of hypothesis 3 (Fig. 26).

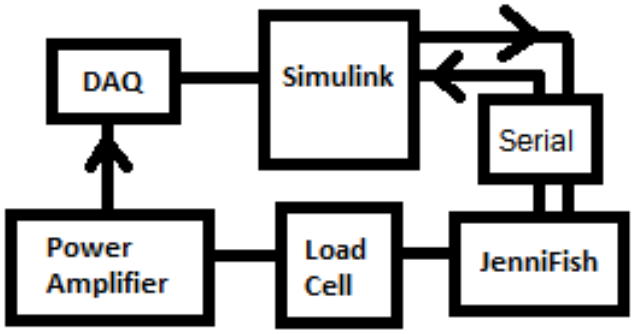

Fig. 26. Block diagram of load cell test setups.

Instrument calibration was done prior to testing with a certified weight set. A calibration curve has been included in Fig. 27.

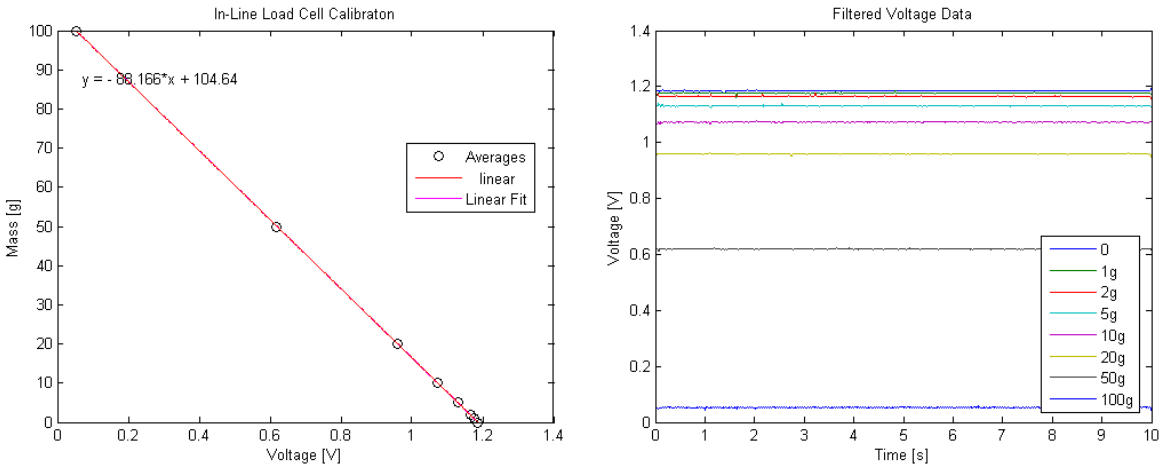

Fig. 27. Calibration curve (left) from the known weight readings (right).

The aquarium tank used for Revision #1 testing measures 762mm long by 305mm wide by 460mm tall. The aquarium tank used for Revision #2 testing measures 762mm long by

305mm wide by 320mm tall. The same aluminum mount, fabricated in-situ, was utilized during both Revision #1 and Revision #2 testing. A connecting mount for the JenniFish can was designed in SolidWorks, 3D printed, and press fit onto the square end of a Heli-coil with 4-40 thread on the opposite end (H.C. 2288-04).

All Revision #1 in-line load cell testing was done at a 0.25Hz cycle frequency with a 50% duty cycle; in other words, each swim cycle was four seconds in total, the first 2 seconds being fully- powered contraction, followed by two seconds of no-power relaxation). JenniFish Revision #2 in-line load cell testing focused on five different cases, motivated by hypothesis 3. Two of these cases involved testing at a frequency of 0.435Hz with a 39.1% duty cycle; one of which was purely timing based while the other used PD control with an offset square wave desired signal. The other three cases had a frequency of 0.333Hz and a 33.3% duty cycle; one case with purely timing based control, one case with PD control using a square wave desired signal, and one case with PD control using an offset square wave desired signal. All PD control was done with the same proportional and derivative gain.

5.1.2 Bending Load Cell Testing

Bending load cell testing was only done for JenniFish Revision #1 and focused on providing evidence for hypothesis 4. The same five cases done for Rev. 2 in-line testing were repeated for each actuator group. A Transducer Techniques 1Kg low capacity single point bending beam load cell was used with the mounting set-up shown in Fig. 28. The same Futek CSG110 Strain Gauge Universal Amplifier was used to connect the bending load cell. It was driven by the same BK Precision 1672 Triple Output DC Power Supply set at 17V. Signal output was brought into Simulink using the National Instruments BNC-

2090A Data Acquisition board. Serial was acquired real-time through Simulink during bending load cell testing in the same way that it was accomplished during in-line testing.

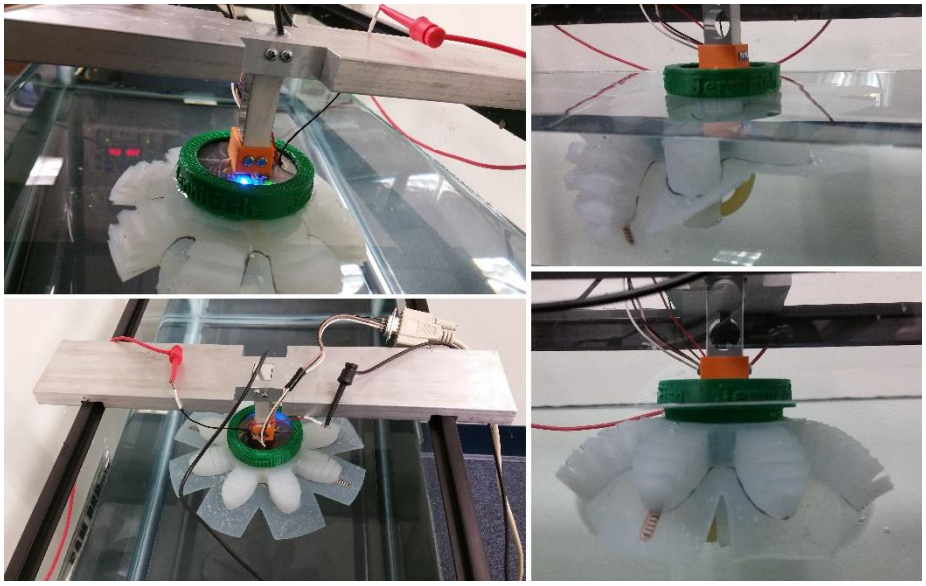

Fig. 28. Bending load cell test setup.

Both sides of the load cell were horizontally calibrated with a known weight set prior to testing; the difference in orientation between calibration and testing did not affect results as only slope, not y-intercept, was used from calibration.

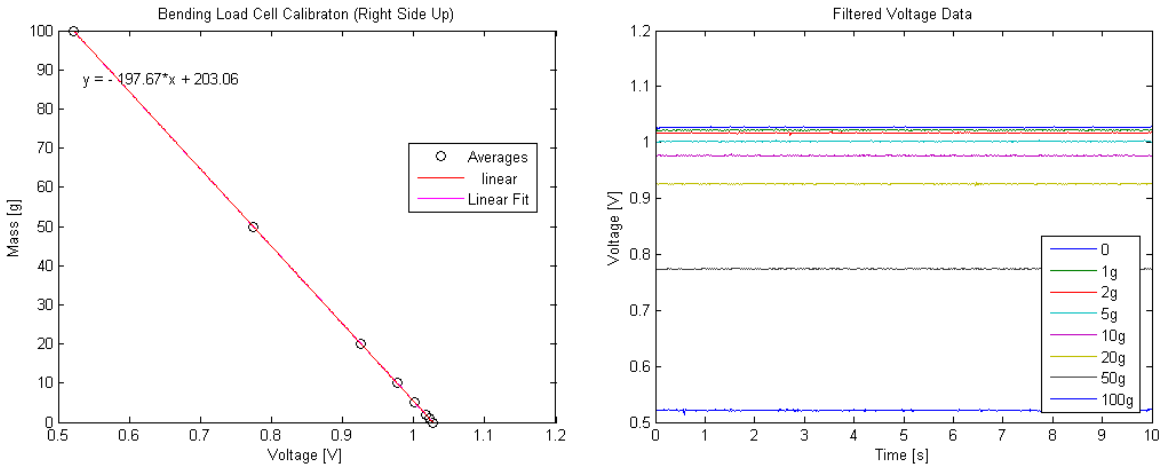

Fig. 29. Bending load cell output during right side calibration.

Fig. 29 shows the calibration output when the load cell's right side was positioned upward and Fig. 30 shows the calibration output for the left side up. The aquarium tank

used for testing measures 762mm long by 305mm wide by 320mm tall. The aluminum mount was fabricated in-situ specifically for this test setup. A connecting mount for the JenniFish can was designed in SolidWorks, 3D printed, and screwed on to the load cell.

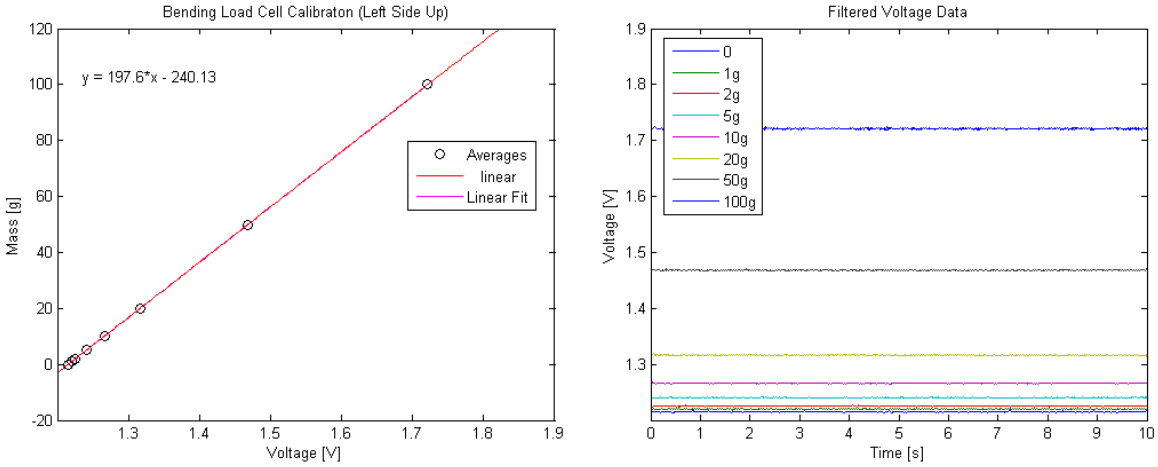

Fig. 30. Bending load cell output during left side calibration.

5.2 Free-Swimming Testing

Free-swimming tests were done in two different environments to provide qualitative vehicle assessment: aquarium tank testing and open-water testing. Tests focused on documenting the JenniFish's ability to swim upward, to change heading, and to record Flash data (including temperature sensor readings and IMU measurements). Video footage was taken of all tests, and some of this footage was then uploaded to YouTube; these video URLs were included in the appendix, section 9.3.

5.2.1 Aquarium Tank Testing

Tank testing was done for both Revision #1 and Revision #2 JenniFish in the same aquarium tanks used for load cell testing. Swimming speed and proficiency – average speed divided by bell diameter – were both characterized visually, using the open source motion tracking software Kinovea. Care was taken to keep footage used for speed characterization in-plane, including the item used for distance calibration, which is why

only tank testing could be used for this. Footage was also collected of the JenniFish producing a turning moment in two directions. Additionally, video footage was gathered of the JenniFish swimming upward through a 152mm hole, proving that it is capable of swimming through an orifice smaller than its 210mm relaxed external diameter. Finally, upward swimming with Flash data collection was completed during tank testing. Temperature variations were instigated with the addition of hot water to the tank periodically throughout data collection. The analog temperature sensor readings were correlated to specific water temperatures according to the sensor's data sheet. These Flash tests were also video recorded so that IMU data could be correlated visually with the JenniFish's motions.

5.2.2 Open-Water Testing

Open-water testing provided the valuable opportunity to test the vehicle's abilities in fresh and salt water, un-controlled environments. Two sites were visited to do ocean testing of JenniFish Revision #2 at a depth of approximately 5.00 meters (Revision #1 was not open-ocean tested). The first testing site was a buried wreck approximately 350 meters off the coast of Delray Beach, FL. The second testing site was approximately 300 meters offshore at the EroJax artificial reef in Dania Beach, FL. Currents at both sites were estimated to be less than 1knot. Additionally, JenniFish Revision #2 was pool tested.

6 RESULTS

6.1 Constrained Testing

Constrained test data was post-processed in MATLAB. The full scripts used for load cell calibration and data parsing have been included in the appendix, section 9.4. Constrained testing was done to evaluate both thrust and turning moment production under different control schemes. Again, it is important to note that all thrust and turning moment values were collected for the body at zero speed. This thesis' scope did not allow for further quantitative testing of the body in motion. Testing of this nature would benefit future work done on this project. All controller implementation utilized the same proportional and derivative gains, so controller performance is reported in this section as well.

Quantitative data concerning system power consumption was also gathered during this testing. Maximum power consumption was calculated from power supply readings during constrained testing, leading to (4). When both pumps were powered, a system current draw of 0.65A was recorded. This leads to the following calculation:

$$\text{Power} = 9.00\text{V} * (0.65\text{A}) * \text{dutyCycle} \qquad (4)$$

Two duty cycles were tested for Revision #2, 33.3% and 39.1%, leading to 1.95W and 2.29W of power consumption respectively. Swimming efficiency based on this power consumption was not calculated due to concerns regarding definition standards; future work could perhaps focus on determining energy per distance and comparing it to the force measurements connected with a load cell.

6.1.1 Controller Performance

The PD controller tuning process involved iteratively testing gain values to achieve acceptable tracking performance. During this process JenniFish Rev. 2 displayed underdamped vehicle dynamics; despite tuning, the system had difficulties reaching a steady state during multi-step tracking (Fig. 31).

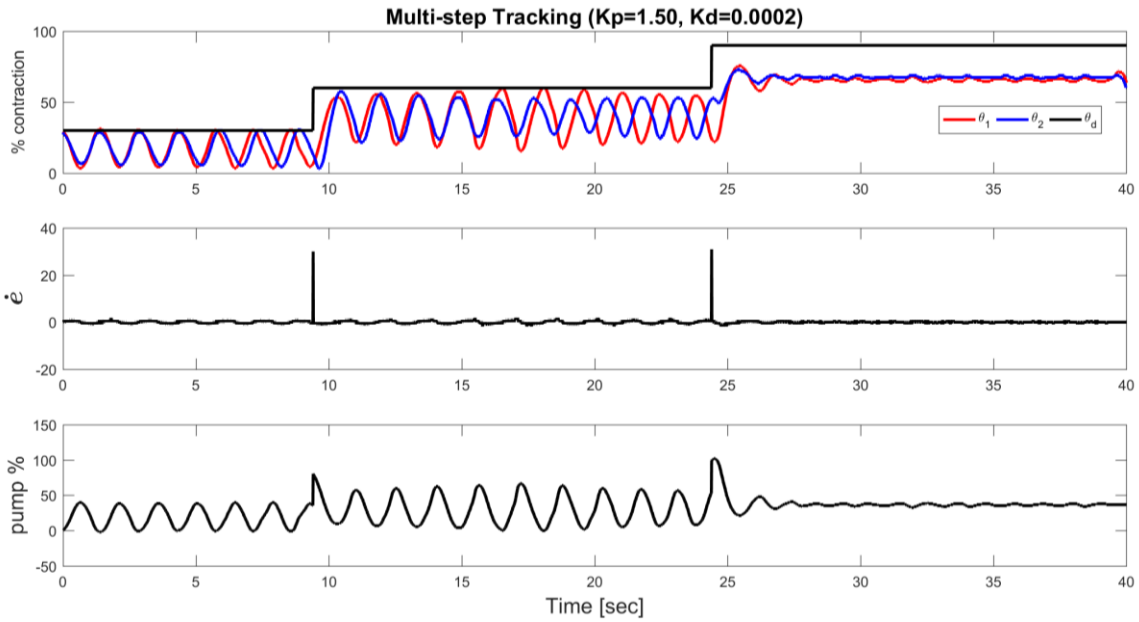

Fig. 31. Controller performance for multi-step tracking (k_p=1.50, k_d=0.0002)

Its performance during sawtooth tracking is displayed in Fig. 32 and its performance during sine wave tracking is shown in Fig. 33. Flex sensor and control signal are plotted in terms of percent contraction, with 0 correlating to fully relaxed actuator position and 100 representing fully contracted actuator position. Flex sensor calibration was done prior to data collection so that sensor values associated with fully relaxed and fully contracted were as accurate as possible. Due to constraints imposed by Simulink's real-time data collection capacity, the system was run at 100Hz and only five parameters could be reliably gathered via serial each loop. For this reason, pump and error data associated with the controller based on flex sensor 1 feedback was collected. Pump signal is

displayed in percentage of max voltage and was constrained between 0 and 100 before use.

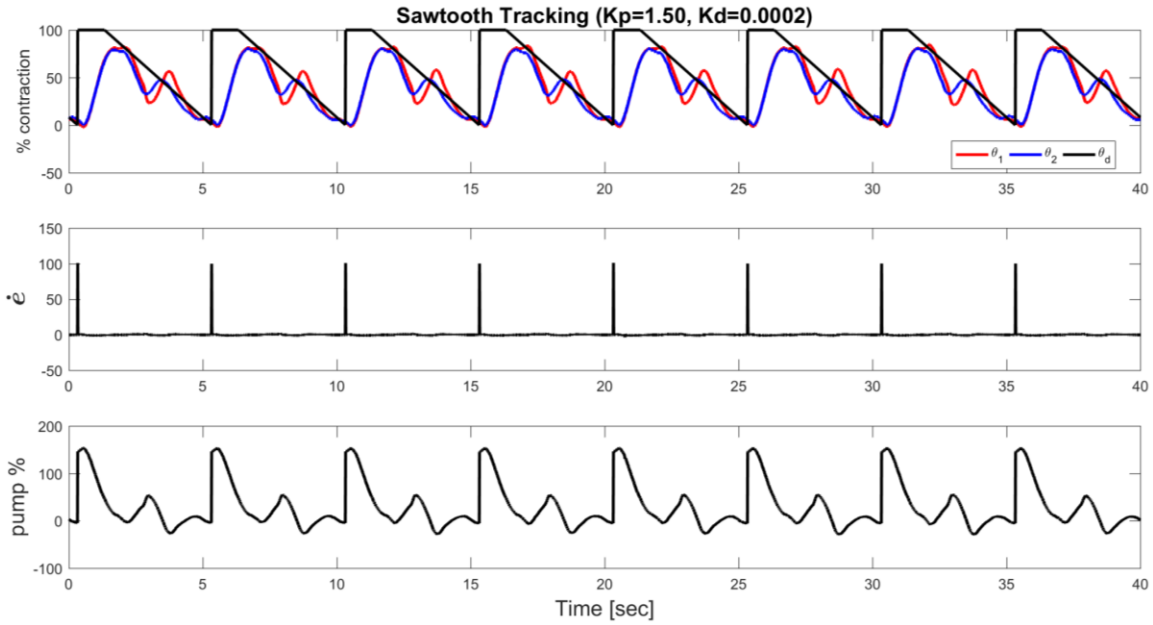

Fig. 32. Controller performance during sawtooth tracking (k_p=1.50, k_d=0.0002).

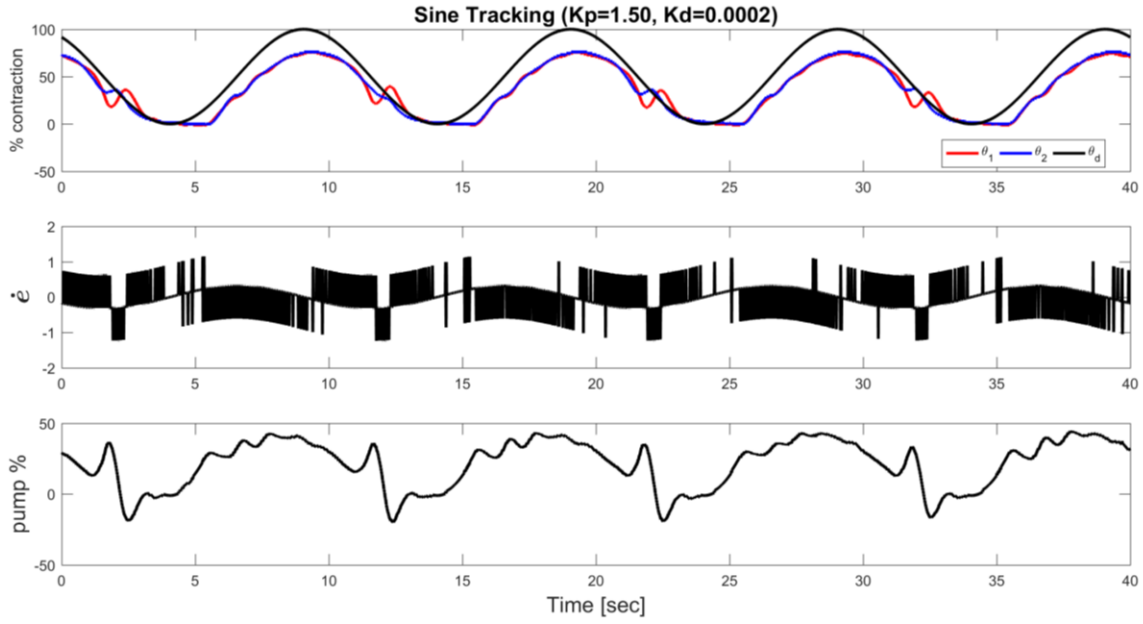

Fig. 33. Controller performance during sine wave tracking (k_p=1.50, k_d=0.0002).

6.1.2 In-Line Load Cell Testing

Several sets of preliminary in-line load cell testing was done with Revision #1 JenniFish. Results from these tests have been summarized in Table 4 for convenience. A positive net thrust was witnessed in all testing configurations, supporting hypothesis 1 that the JenniFish is capable of swimming.

Table 4. Summary of Rev. 1 JenniFish load cell results.

4 cycle Thrust [N] (uniform flap)	4 cycle Thrust [N] (segmented flap)	14 cycle Thrust [N] (uniform flap)	14 cycle Thrust [N] (segmented flap)
7.60E-03	1.29E-02	8.20E-03	6.80E-03

To reiterate, the same testing mount was used for both Rev. 1 and Rev. 2 testing; however, battery power was used for Rev. 1 tests whereas a power supply was utilized during Rev. 2 tests. Furthermore, Rev. 1 did not have battery voltage feedback like Rev. 2 does. The differences in vehicle construction (stiffer flex sensors and paper bottom reinforcement) made it so that testing was done at a 0.25Hz swimming frequency with a 50% duty cycle. Fig. 34 contains a photo sequence taken during testing.

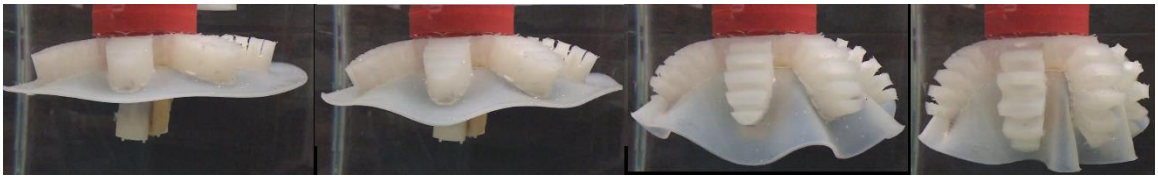

Fig. 34. Rev. 1 contraction sequence (source: Jen Frame, first published in [32]).

The first set of Rev. 1 tests consisted of 20 trials to collect net upward thrust force produced over 14 cycles with a uniform flap and waterline at top of electronics cap. Number of cycles was chosen to facilitate comparison to prior work [20]. Fig. 35 shows 4 of these trials and the unloaded readings before and after data collection. The average thrust force produced over all 20 trials was 15.5mN with a drift of 7.30mN.

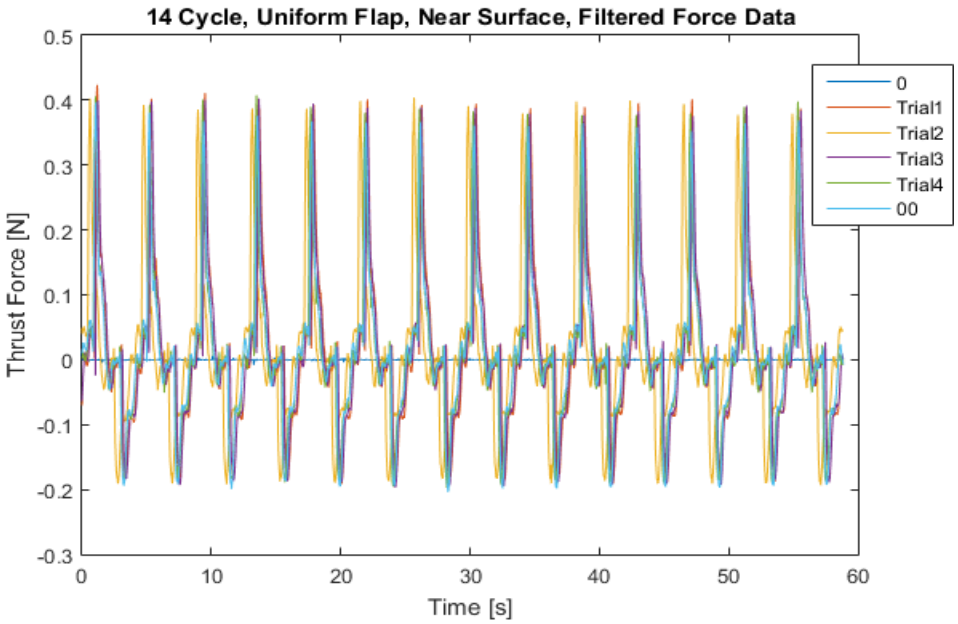

Fig. 35. Four trials of Rev. 1 filtered force data.

The next set of Rev. 1 tests consisted of 20 trials to collect net thrust force produced over 14 cycles with a segmented flap and waterline at top of electronics cap. Fig. 36 shows 4 of these trials. The average thrust force produced over all 20 trials was 15.5mN with a drift of 8.70mN. Considering only the first four cycles yielded an average thrust force of 12.9mN and a corresponding drift of 5.70mN.

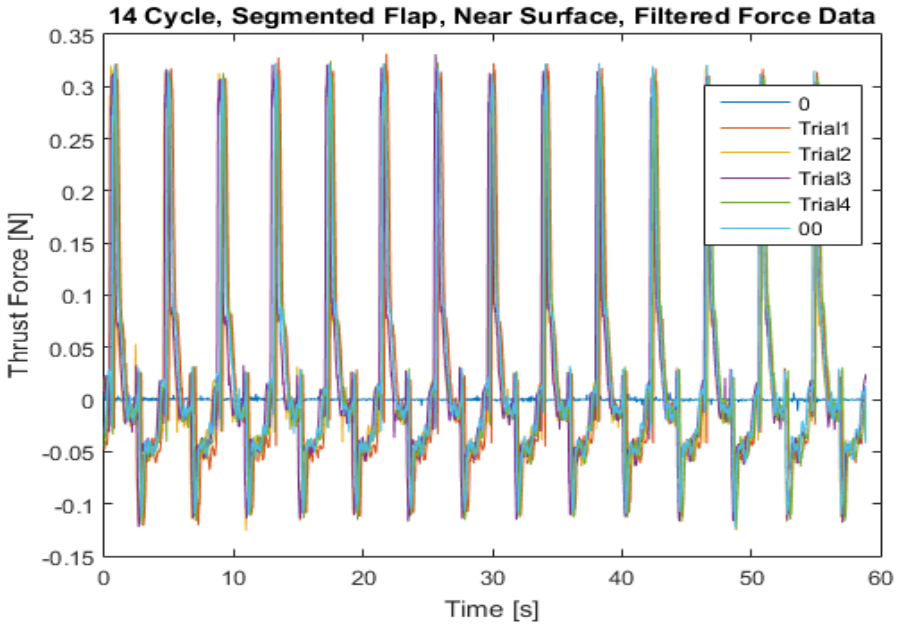

Fig. 36. Four trials of Rev. 1 filtered force data.

Lastly, a test set of 20 trials was done to collect net thrust force produced over 14 cycles with a segmented flap and waterline approximately 80mm above the top of the electronics cap. This test was meant to reveal if the JenniFish's location in prior tests caused unwanted surface effects. Below is a graph (Fig. 37) of 4 of these trials and the unloaded readings before and after data collection. The average thrust force produced over all 20 trials was14.5mN with a drift of 6.25mN. The very small difference between tests done at the surface and submerged 80mm indicates that either surface effects are negligible or that 80mm was not a large enough distance to remove them.

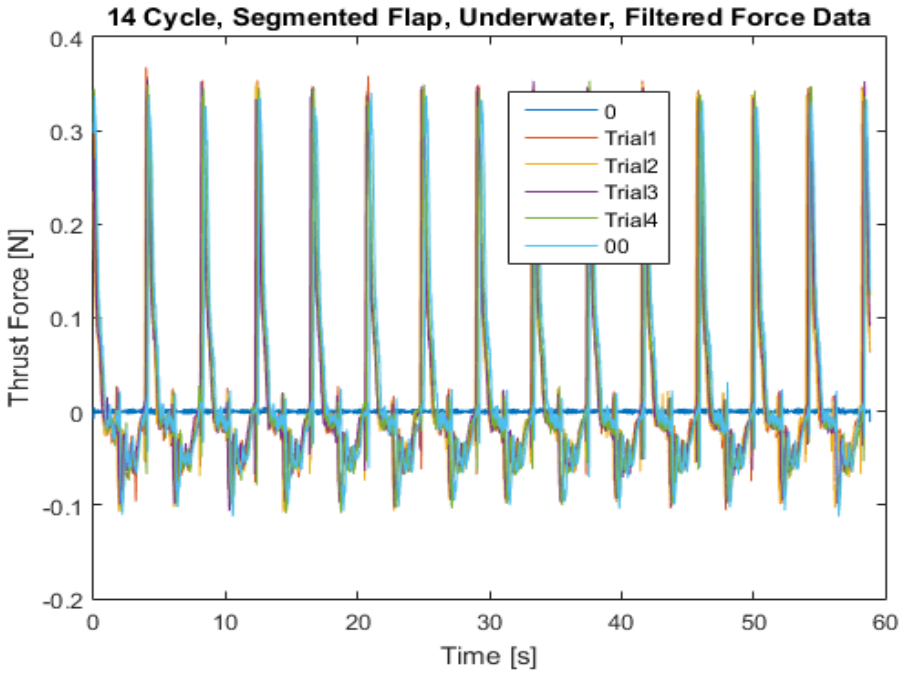

Fig. 37. Four trials of the fully submerged Rev. 1 test set.

Three sets of JenniFish Revision #2 testing were done; however, the bulk of graphs pertaining to the first two sets are located in the appendix, section 9.5. Although 18 cycles of data were collected per case, only the last 14 were analyzed in this section; this was done to ensure that steady state was reached for 0.435Hz tests and to maintain consistency with Revision #1 tests. The full 18 cycle analysis can be found in the

appendix, section 9.5. Table 5 summarizes average net thrust per cycle for each case and all three data sets.

Table 5. Summary of average cycle-net-force per case (only last 14 cycles).

Data Set	No Control 0.333Hz, 33.1%	No Control 0.435Hz, 39.1%	Square Wave 0.333Hz, 33.3%	Offset Sq Wave 0.333Hz, 33.3%	Offset Sq Wave 0.435Hz, 39.1%	Drift
1	0.0218	0.0241	0.0194	0.0176	0.0217	0.0035
2	0.0217	0.0221	0.0188	0.0164	0.0210	0.0027
3	0.0223	0.0250	0.0210	0.0195	0.0232	0.0014
average	0.0220	0.0237	0.0197	0.0179	0.0220	0.0025

Table 6 summarizes net thrust production during the last 14 cycles of all cases and all data sets. The positive net thrusts in both tables support hypothesis 1. Additionally, the case that produced the largest net thrust production contributes to hypotheses 2 and 3.

Table 6. Summary of net force per case (last 14 cycles only)

Data Set	No Control 0.333Hz, 33.1%	No Control 0.435Hz, 39.1%	Square Wave 0.333Hz, 33.3%	Offset Sq Wave 0.333Hz, 33.3%	Offset Sq Wave 0.435Hz, 39.1%	Drift
1	0.0216	0.0237	0.0191	0.0174	0.0213	0.0035
2	0.0214	0.0217	0.0185	0.0165	0.0207	0.0027
3	0.0221	0.0246	0.0207	0.0192	0.0233	0.0014
average	0.0217	0.0233	0.0194	0.0177	0.0218	0.0025

A two-way analysis of variance (ANOVA2) was completed for the max thrust per cycle as well as the net thrust per cycle. Table 7 contains the results of the max thrust evaluation for all data sets. A value smaller than 0.05 for Prob>F Rows indicates statistical difference between the two cases being compared. A value larger than 0.05 for Prob>F Columns indicates that the data points within a case are statistically repeatable. This analysis was done to contribute to hypotheses 2 and 3.

Table 8 contains the results of the net thrust per cycle ANOVA2 for the last 14 cycles of all three data sets. Again, a P>F Rows value smaller than 0.05 indicates statistical independence between cases. This analysis was done to contribute to hypotheses 2 and 3 as well.

Table 7. Results of max thrust per cycle ANOVA2 of all data sets (last 14 cycles only).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.9543
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.8506
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.2493
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0007	0.2742
Offset Sq Wave (0.333Hz) vs. Offset Sq Wave (0.435Hz)	0	0.9991

Table 8. Results of net thrust per cycle ANOVA2 of all data sets (last 14 cycles only).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0.1832	0.4107
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.5984
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0003	0.669
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0894	0.3897
Offset Sq Wave (0.333Hz) vs. Offset Sq Wave (0.435Hz)	0.0383	0.8772

The following graphs included in this section reinforce what was summarized in the previous tables. The next two plots and two tables pertain to JenniFish Rev. 2, data set 1, last 14 cycles only. Fig. 38 contains the first set of load cell measurements at a frequency of 0.435Hz with a 39.1% duty cycle. Load cell drift for set 1 was 3.50mN.

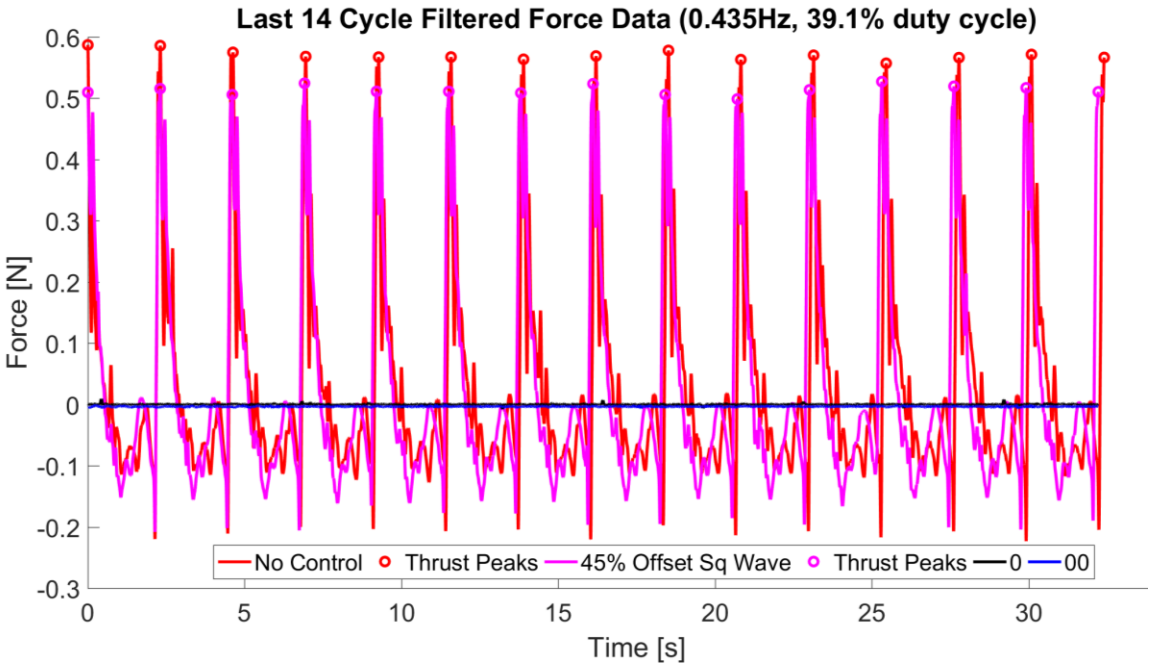

Fig. 38. Set 1 last 14 cycle filtered force data (0.435Hz, 39.1% duty cycle).

Fig. 39 contains the first set of load cell measurements at a frequency of 0.333Hz with a 33.3% duty cycle. Table 9 and Table 10 contain theANOVA2 results of data set 1 (last 14 cycles only).

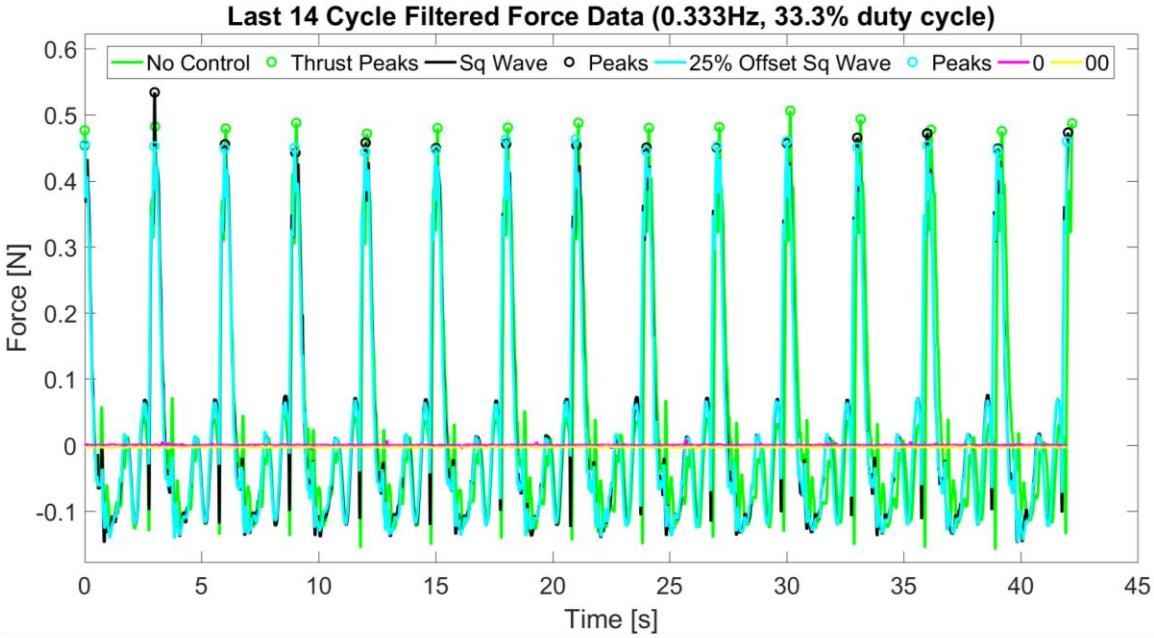

Fig. 39. Set 1 last 14 cycle filtered force data (0.333Hz, 33.3% duty cycle).

Table 9. Results of max thrust per cycle ANOVA2 of data set 1 (last 14 cycles).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.7936
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0.0027	0.4722
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.0302
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.1522	0.4155

Table 10. Results of net thrust per cycle ANOVA2 of data set 1 (last 14 cycles).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.1252
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.1089
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0001	0.724
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0453	0.8913

The following two plots and two tables pertain to JenniFish Rev. 2, data set 2, last 14 cycles only. Fig. 40 contains the second set of load cell measurements at a frequency of 0.435Hz with a 39.1% duty cycle. Load cell drift for set 2 was 2.70mN.

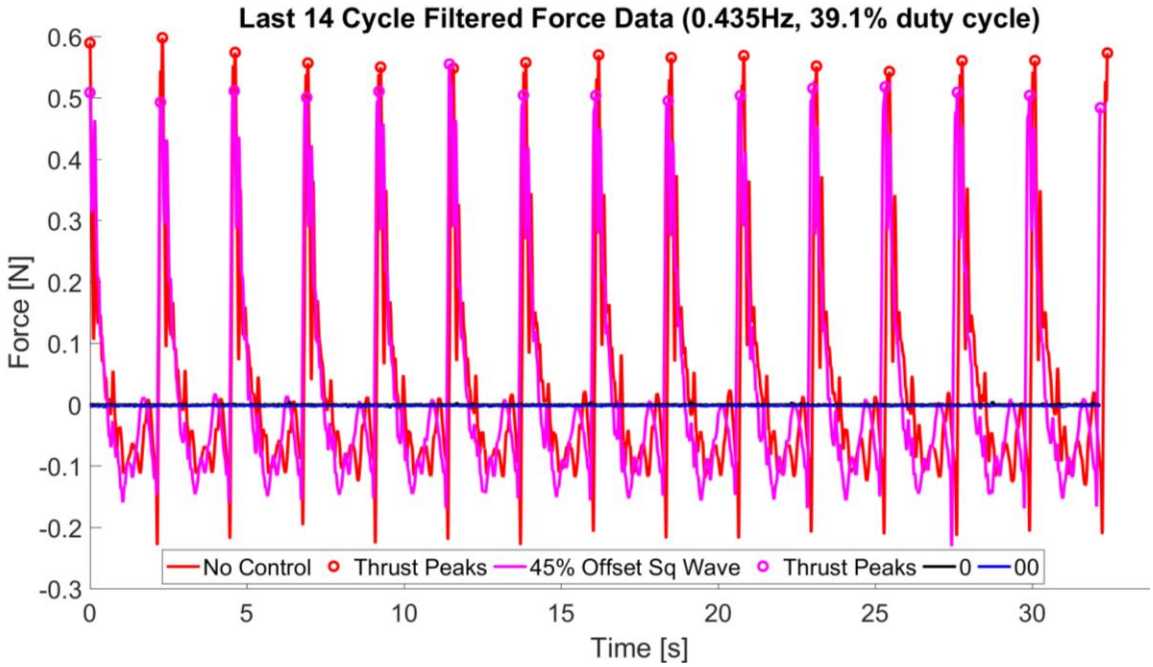

Fig. 40. Set 2 last 14 cycle filtered force data (0.435Hz, 39.1% duty cycle).

Fig. 41 contains set 2 in-line load cell measurements at a frequency of 0.333Hz with a 33.3% duty cycle. Table 11 and Table 12 contain theANOVA2 results of data set 2.

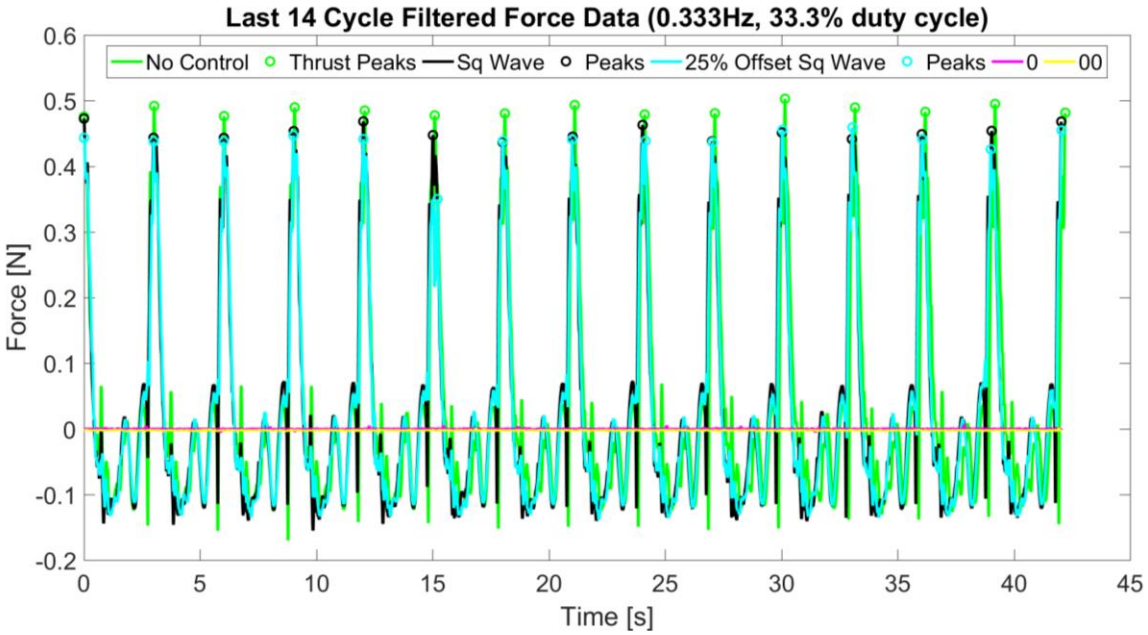

Fig. 41. Set 2 last 14 cycle filtered force data (0.333Hz, 33.3% duty cycle).

Table 11. Max thrust per cycle set 2 ANOVA2 results (last 14 cycles only).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.98
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.7234
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.2476
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.044	0.349

Table 12. Net thrust per cycle set 2 ANOVA2 results (last 14 cycles only).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0.5408	0.4559
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0.0001	0.9375
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0981	0.5685
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.4203	0.4107

The remaining in-line load cell plots pertain to JenniFish Rev. 2, data set 3, last 14 cycles only. Fig. 42 contains the third and final set of load cell measurements at a frequency of 0.435Hz with a 39.1% duty cycle. Load cell drift for set 3 was 1.40mN.

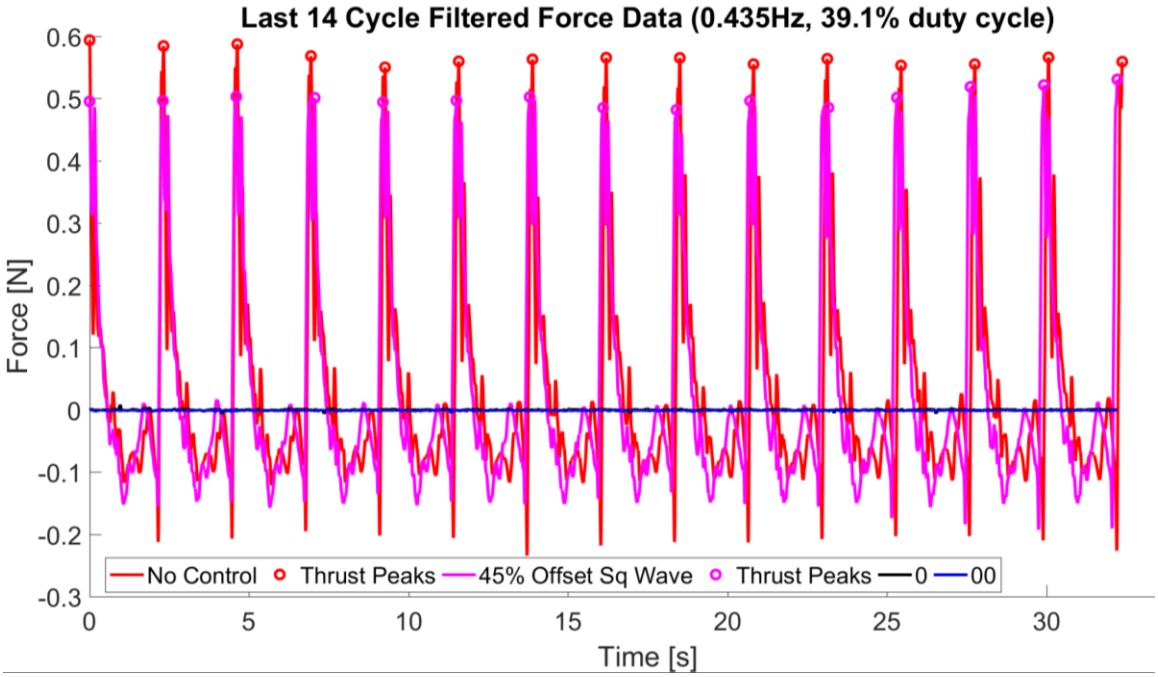

Fig. 42. Set 3 last 14 cycle filtered force data (0.435Hz, 39.1% duty cycle).

Fig. 43 contains the third set of load cell measurements at a frequency of 0.333Hz with a 33.3% duty cycle. Table 13 and Table 14 contain theANOVA2 results of data set 3 (last 14 cycles only).

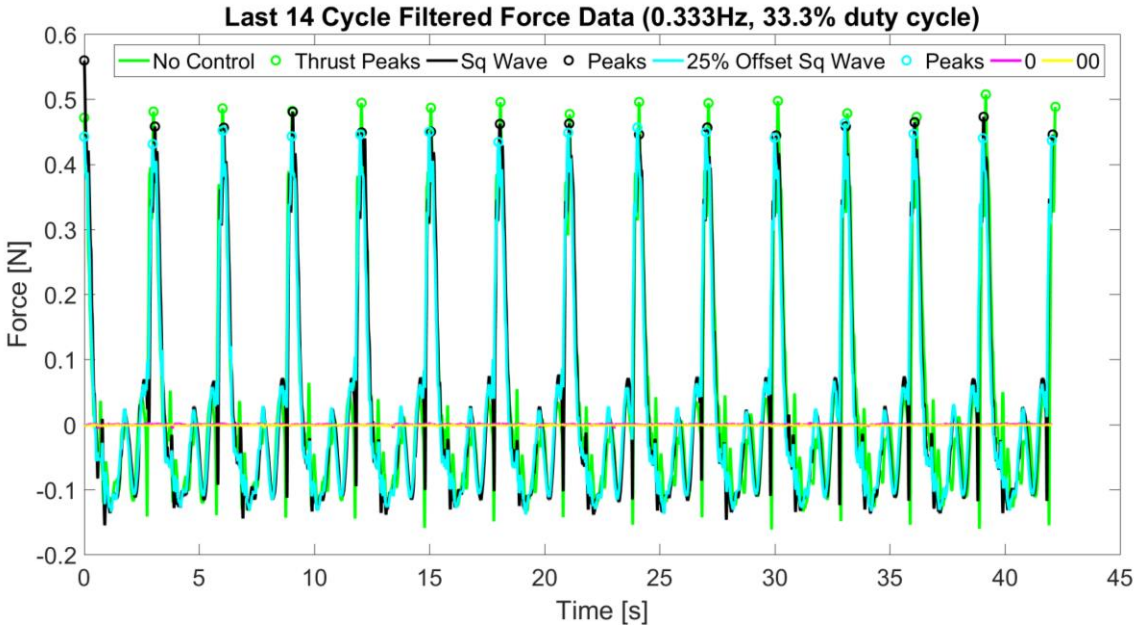

Fig. 43. Set3 last 14 cycle filtered force data (0.333Hz, 33.3% duty cycle).

Table 13. Max thrust per cycle set 3 ANOVA2 results (last 14 cycles only).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.7031
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0.0211	0.8634
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.7303
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.03	0.6121

Table 14. Net thrust per cycle set 3 ANOVA 2 results (last 14 cycles only).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0.62	0.5028
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0.3099	0.6785
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0133	0.9736
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.3334	0.7957

For the JenniFish Rev. 2 third and final data set, last 14 cycles only, serial data collected during load cell testing is also plotted. Fig. 44 shows in-line load cell data plotted over flex sensor 1 data for no control at 0.333Hz, 33.3% duty cycle. The signal to both pump 1 and pump 2 during no control cases are square waves from 0 to 100 with the same duty cycle and frequency as the case (either 0.333Hz or 0.435Hz).

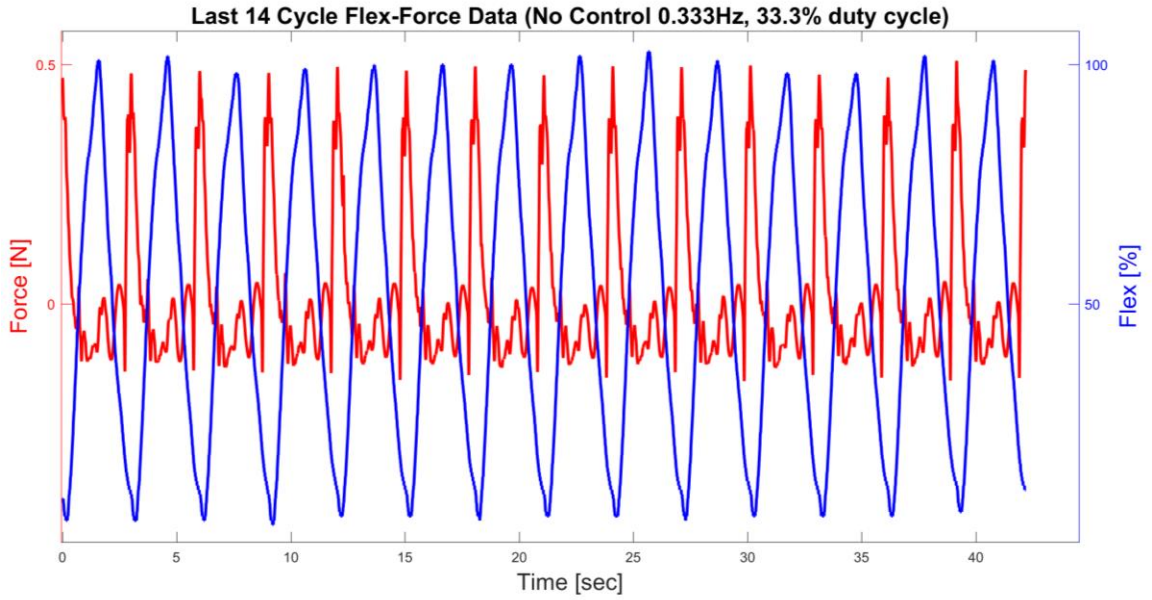

Fig. 44. Set 3 last 14 cycle flex-force data (no control, 0.333Hz, 33.3% duty cycle).

Fig. 45 shows in-line load cell data plotted over flex sensor 1 data for no control at 0.435Hz, 39.1% duty cycle. Again, both pump signals during no control cases are square waves from 0 to 100 with the same duty cycle and frequency as the case (here 0.435Hz).

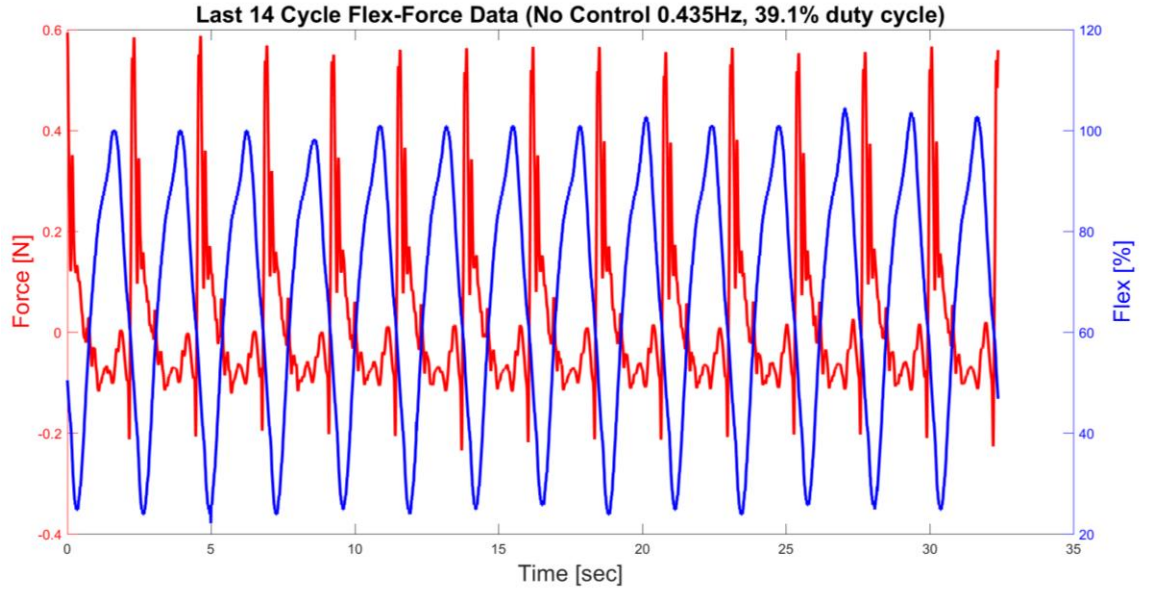

Fig. 45. Set 3 last 14 cycle flex-force data (no control, 0.435Hz, 39.1% duty cycle).

Fig. 46 shows in-line load cell data plotted over flex sensor 1 data for square wave control at 0.435Hz, 39.1% duty cycle. Fig. 47 contains the remaining controller data for

this case. Remember, real-time serial data collection issues with Simulink limited data gathering to only one controller; however, the other controller outputs should be similar.

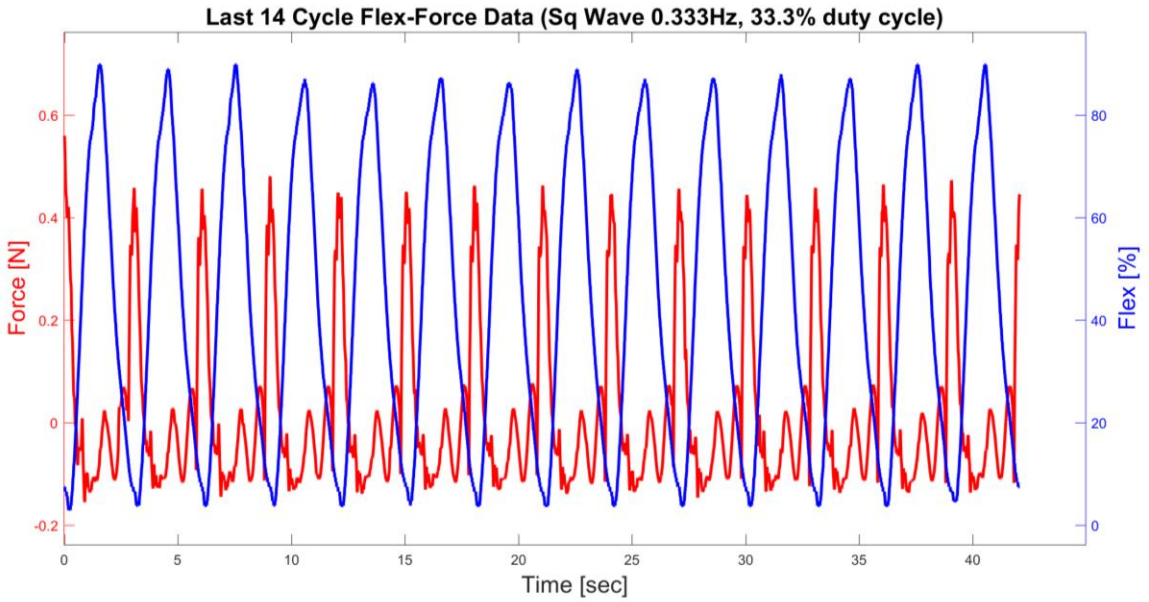

Fig. 46. Set 3 last 14 cycle flex-force data (square wave, 0.333Hz, 33.3% duty cycle).

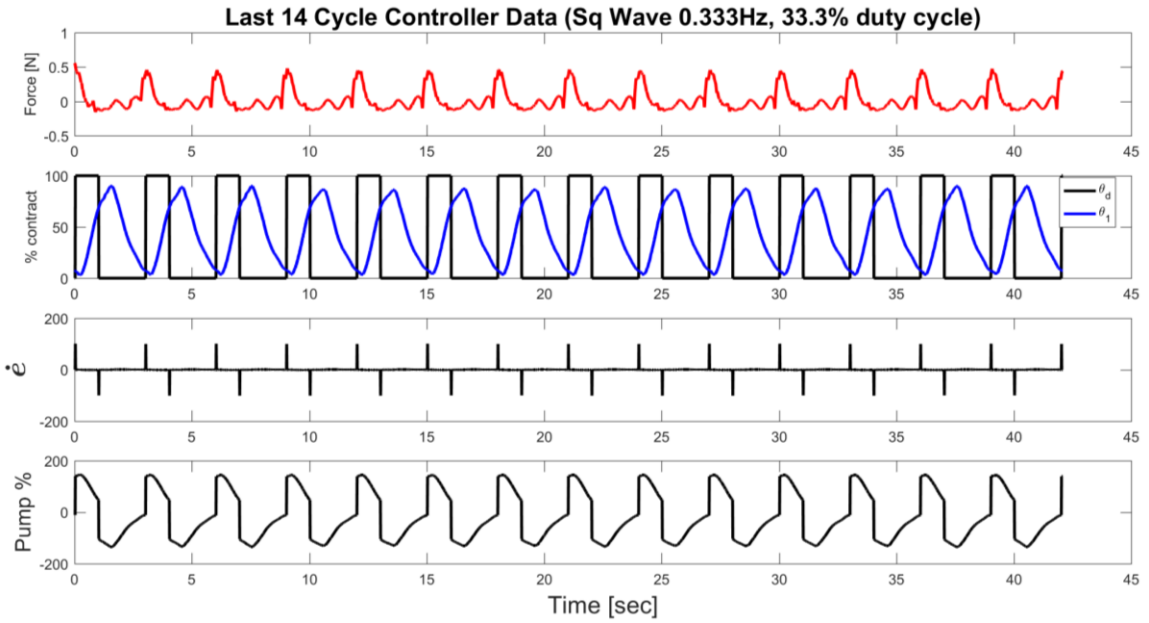

Fig. 47. Set 3 last 14 cycle controller 1 data (square wave, 0.333Hz, 33.3% duty cycle).

Fig. 48 shows in-line load cell data plotted over flex sensor 1 data for a 25% offset square wave control at 0.333Hz, 33.3% duty cycle. Fig. 49 contains the remaining controller data for this case. Again, real-time serial data collection issues with Simulink

limited data gathering to only one controller; however, the other controller outputs should be similar.

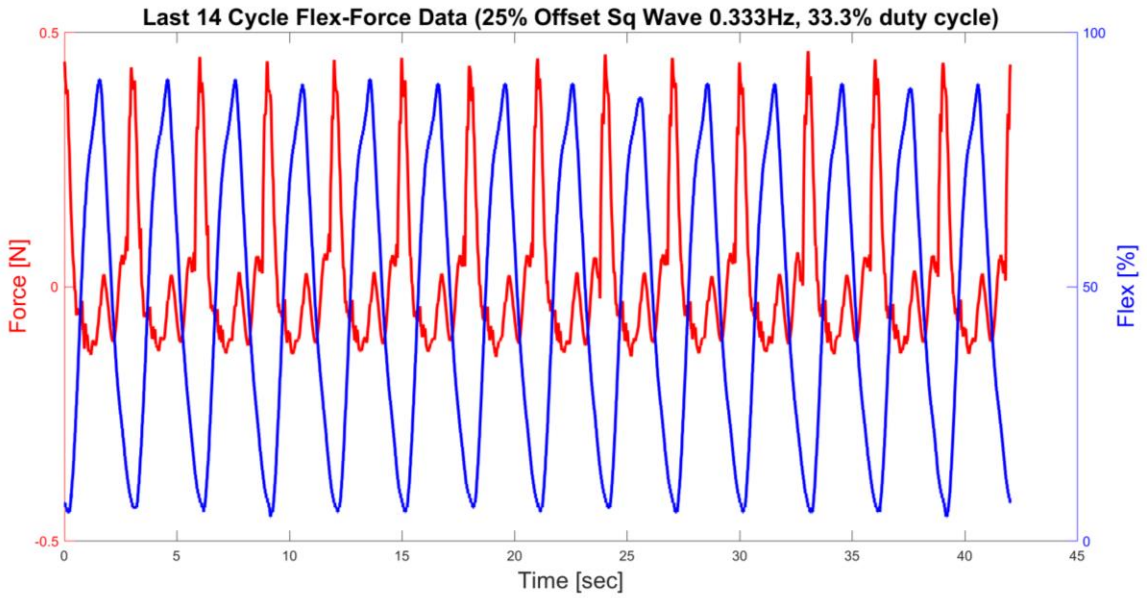

Fig. 48. Set 3 last 14 cycle flex-force data (25% offset sq. wave, 0.333Hz, 33.3% DC).

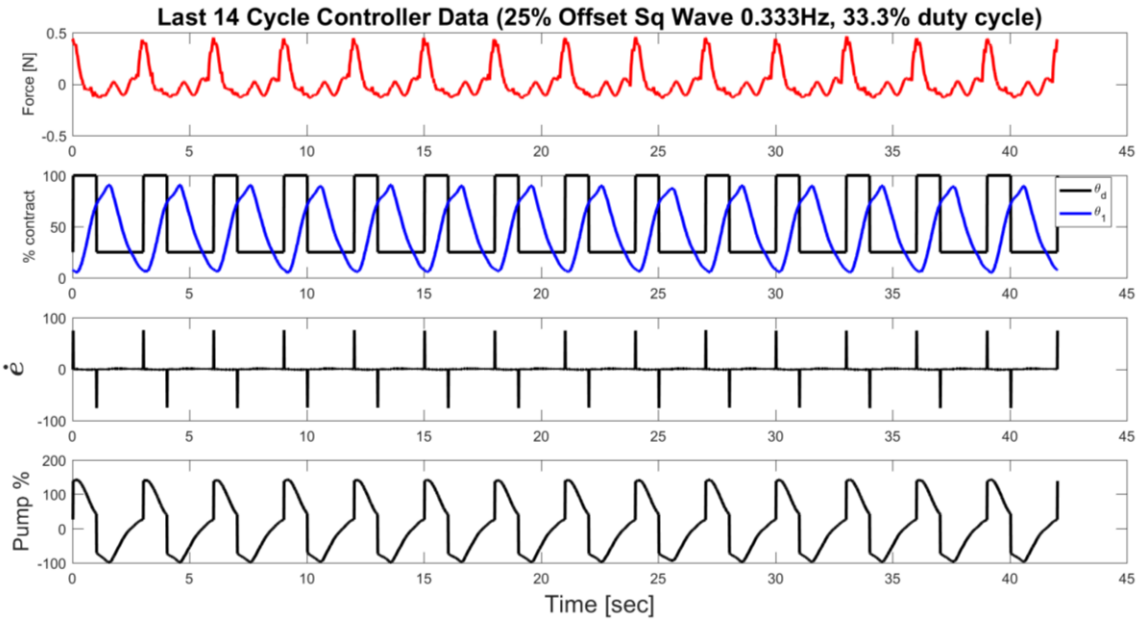

Fig. 49. Set 3 last 14 cycle flex-force data (25% offset sq. wave, 0.333Hz, 33.3% DC).

Fig. 50 shows the last in-line load cell data plotted over flex sensor 1 data for a 45% offset square wave control at 0.435Hz, 39.1% duty cycle. Fig. 51 contains the remaining controller data for this case.

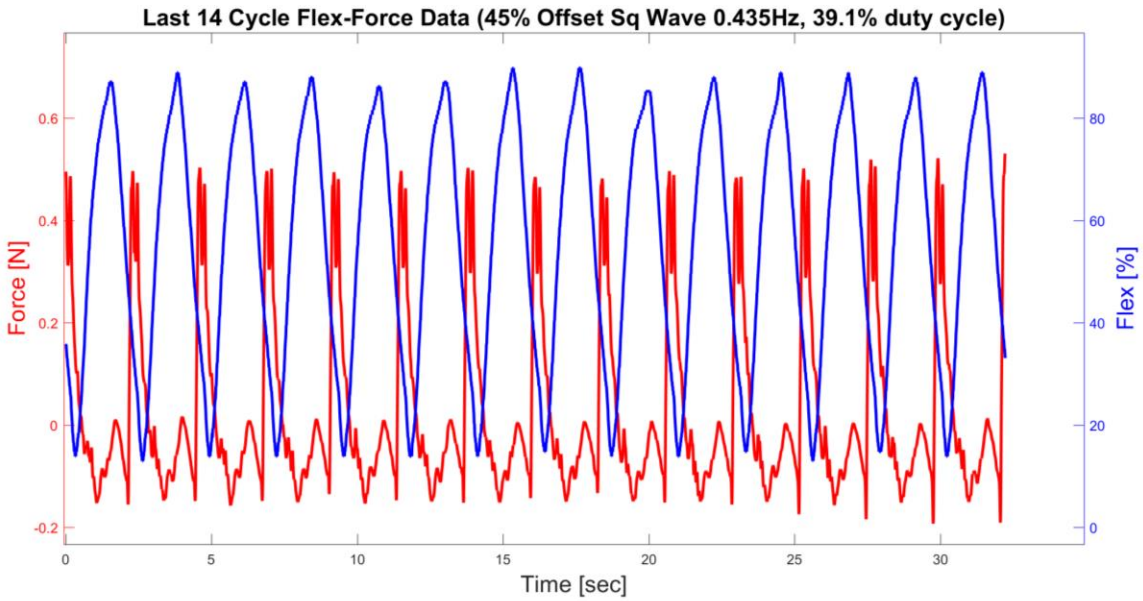

Fig. 50. Set 3 last 14 cycle flex-force data (45% offset sq. wave, 0.435Hz, 39.1% DC).

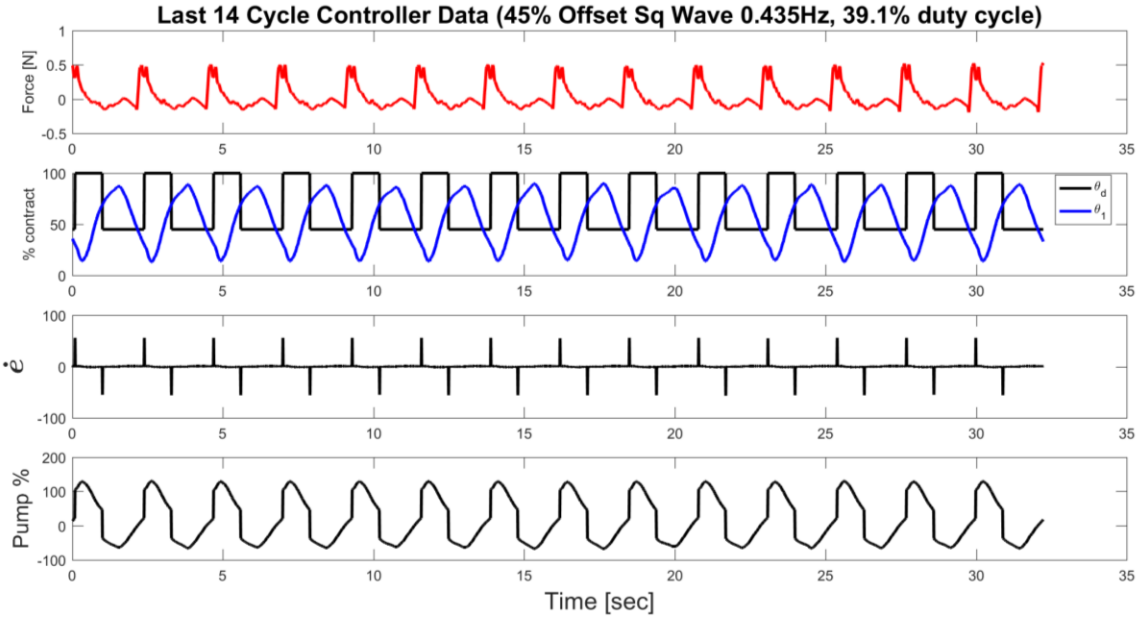

Fig. 51. Set 3 last 14 cycle flex-force data (45% offset sq. wave, 0.435Hz, 39.1% DC).

6.1.3 Bending Load Cell Testing

One set of bending load cell measurements was collected for each of the two pumps on JenniFish Revision #2. If hypothesis 4 is correct, running each side independently should create a turning moment. Net force was registered by the bending load cell while testing both sides (see Table 15 and Table 16), which supports the

hypothesis that the JenniFish's two-sided actuation permits bi-directional heading control. Net force per cycle was calculated for each case as well as net thrust production thrust per case (last 14 cycles reported here).

Table 15. Summary of net force per case (only last 14 cycles).

Pump	No Control 0.333Hz, 33.1%	No Control 0.435Hz, 39.1%	Square Wave 0.333Hz, 33.3%	Offset Sq Wave 0.333Hz, 33.3%	Offset Sq Wave 0.435Hz, 39.1%	Drift
1	0.0018	0.0013	0.0029	0.0013	0.0006	0.0001
2	0.0009	0.0002	0.0024	0.0007	0.0003	-0.0001

Table 16. Summary of average cycle net force per case (only last 14 cycles).

Pump	No Control 0.333Hz, 33.1%	No Control 0.435Hz, 39.1%	Square Wave 0.333Hz, 33.3%	Offset Sq Wave 0.333Hz, 33.3%	Offset Sq Wave 0.435Hz, 39.1%	Drift
1	0.0018	0.0014	0.0030	0.0013	0.0007	0.0001
2	0.0009	0.0003	0.0024	0.0008	0.0004	-0.0001

The same five cases used in JenniFish Rev. 2 in-line testing were done during bending load cell tests (18 full cycles each). Two of these cases involved testing at a frequency of 0.435Hz with a 39.1% duty cycle; one of which was purely timing based while the other used PD control with an offset square wave desired signal. The other three cases had a frequency of 0.333Hz and a 33.3% duty cycle; one case with purely timing based control, one case with PD control using a square wave desired signal, and one case with PD control using an offset square wave desired signal. All PD control was done with a proportional gain of 1.50 and a derivative gain of 0.0002. Two-way analysis of variance was completed for the max thrust per cycle as well as the net thrust production per cycle. Fig. 52 contains the pump1 load cell measurements at a frequency of 0.435Hz with a 39.1% duty cycle. Load cell drift for pump 1 was 0.10mN.

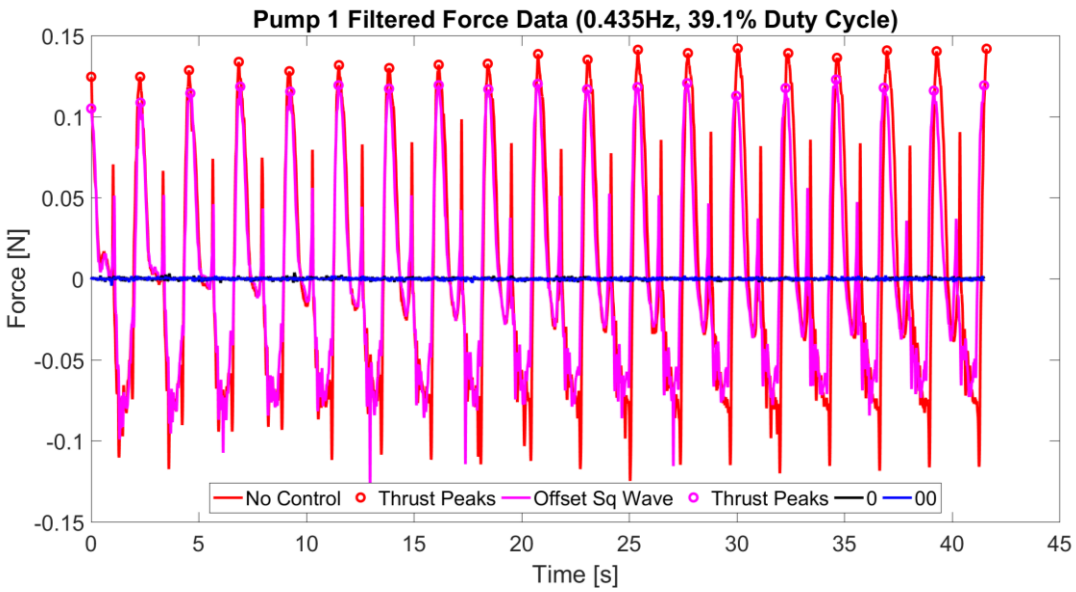

Fig. 52. All 18 cycles of pump 1 filtered force data (0.435Hz, 39.1% duty cycle).

Fig. 53 contains pump 1 load cell measurements at a frequency of 0.333Hz with a 33.3% duty cycle.

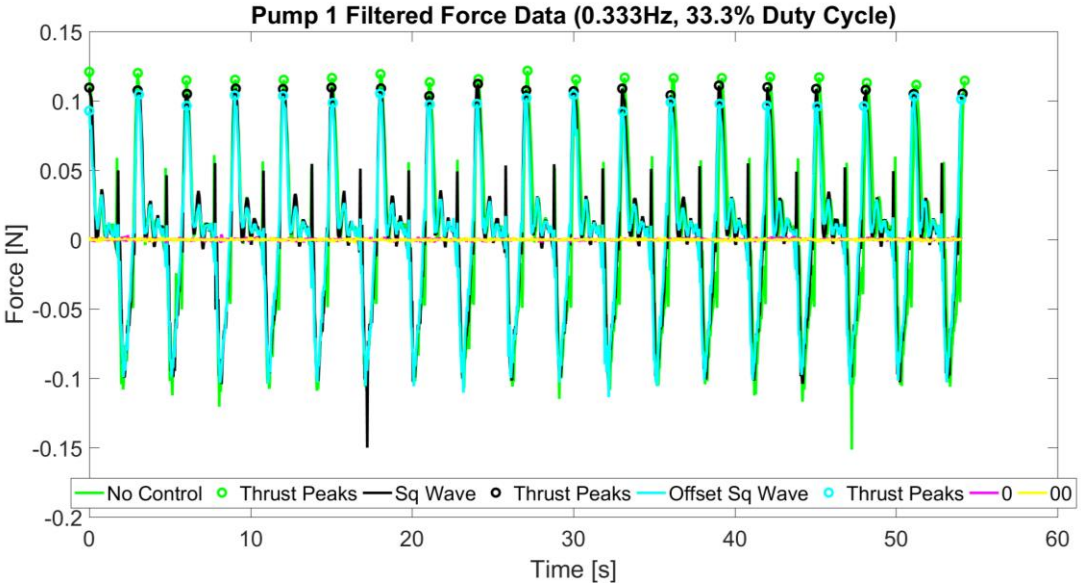

Fig. 53. All 18 cycles of pump 1 filtered force data (0.333Hz, 33.3% duty cycle).

Table 17 and Table 18 summarize the ANOVA2 results from MATLAB done for pump 1 max thrust per cycle and net thrust per cycle respectively (last 14 cycles only). Remember that Prob>F values less than 0.05 represent statistical difference between compared data points (either in rows or columns).

Table 17. Pump 1 max thrust per cycle ANOVA2 results (last 14 cycles).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.4993
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.0771
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.3762
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.7791

Table 18. Pump 1 net thrust per cycle ANOVA2 results (last 14 cycles).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0.2746	0.3599
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0.0661	0.9969
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.3078	0.0767
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0467	0.8708

Fig. 54 contains the pump 2 load cell measurements at a frequency of 0.435Hz with a 39.1% duty cycle. Load cell drift for pump 1 was -0.10mN.

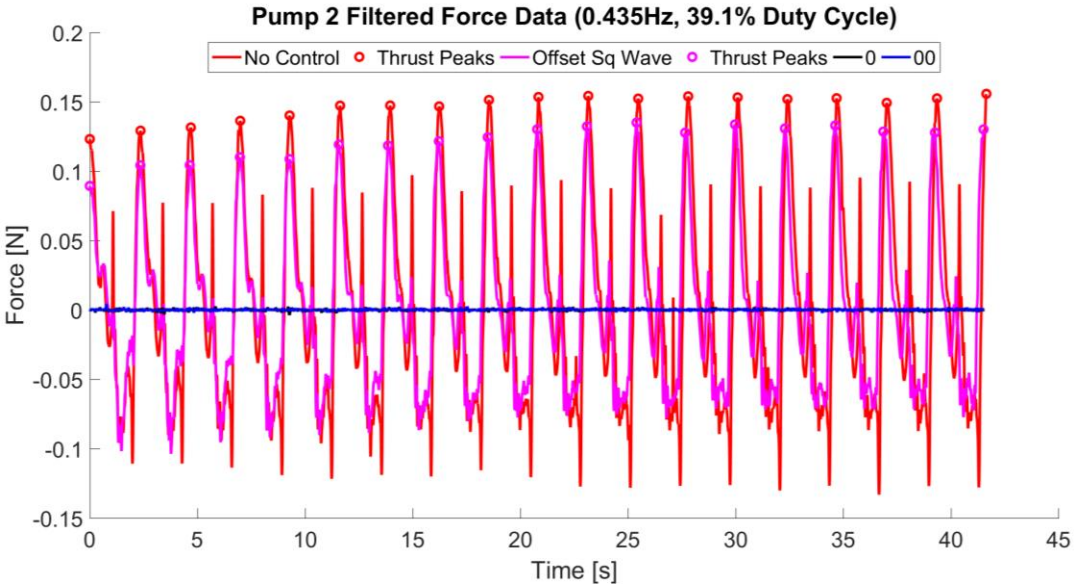

Fig. 54. All 18 cycles of pump 2 filtered force data (0.435Hz, 39.1% duty cycle).

Fig. 55 contains the pump 2 load cell measurements at a frequency of 0.333Hz with a 33.3% duty cycle.

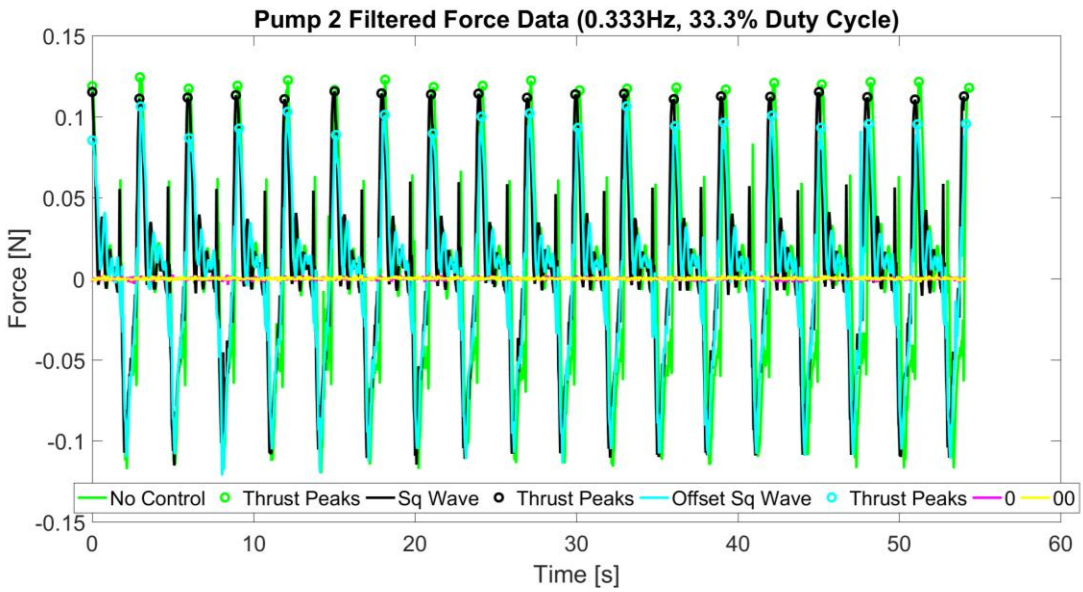

Fig. 55. All 18 cycles of pump 2 filtered force data (0.333Hz, 33.3% duty cycle).

Table 19 and Table 20 summarize the ANOVA2 results from MATLAB done for pump 2 max thrust per cycle and net thrust per cycle respectively (last 14 cycles only).

Table 19. Pump 2 max thrust per cycle ANOVA2 results (last 14 cycles only).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.0003
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.9271
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.0903
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.6984

Table 20. Pump 2 net thrust per cycle ANOVA2 results (last 14 cycles only).

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0.9268	0.5636
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0.1445	0.8718
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.8908	0.4984
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0189	0.384

6.2 Free-Swimming Testing

Free-swimming tests were used to qualitatively show that the JenniFish is capable of forward and bi-directional movement as well as the ability to swim through an orifice smaller than its external diameter. Additionally, sensor capabilities and performance in

uncontrolled environments were shown for proof of concept. Table 21 summarizes the metrics used for comparison with other robotic jellyfish platforms.

Table 21. Summary of metrics compared to other robotic jellyfish.

	JenniFish Rev. 1	JenniFish Rev. 2	RoboJelly	IPMC	A. aurita	A. victoria
Proficiency [s^(-1)]	0.13	0.19	0.19	–	0.25	0.20-0.40
Speed [mm/s]	20	30	54.2	1.5	13	20
14 cycle net thrust [mN] (segmented)	6.8	23.3	3.9	–	–	–
Avg. Power [W]	3	2.29	16.74	1.14	–	–

6.2.1 Aquarium Tank Testing

Upward speed in the JenniFish Revision #1 was measured to be 0.02m/s while upward speed in the JenniFish Revision #2 was measured to be 0.03m/s. Consequently, JenniFish Revision #1 was found to have a 0.125s^{-1} proficiency and the JenniFish Revision #2 was found to have a 0.188s^{-1} proficiency (both based on their160mm bell diameter). Turning and pass-through capabilities were also qualitatively ascertained during tank testing (Fig. 56).

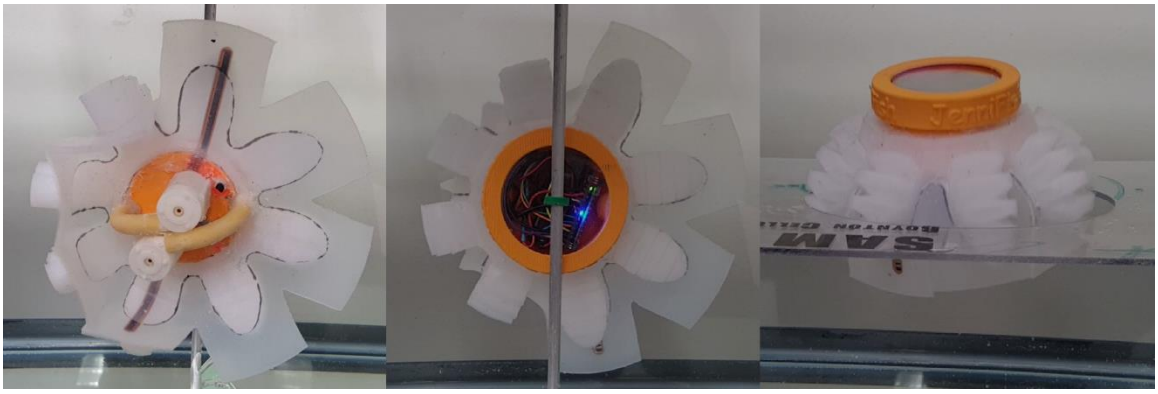

Fig. 56: Pump 1 turning (left), pump 2 turning (middle), 152mm pass-through (right).

Finally, Flash data was collected during several tank tests; roll, pitch, and temperature sensor data are shown in Fig. 57. Flex sensor, control waveform, and pump signals are shown in Fig. 58.

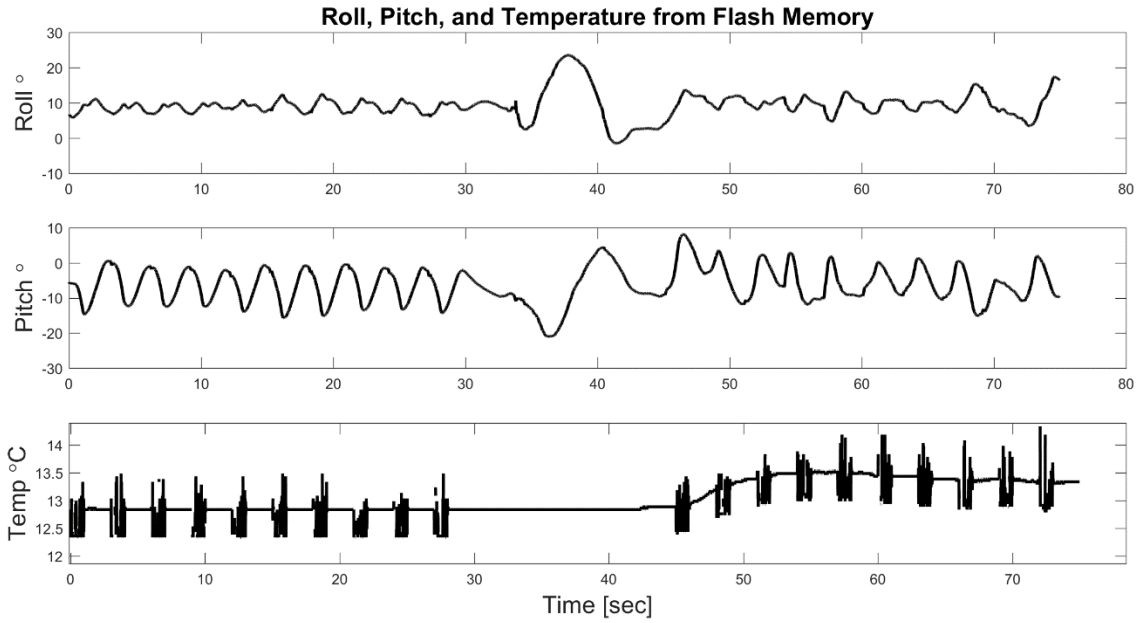

Fig. 57. Roll, pitch, and analog temperature reading from flash memory.

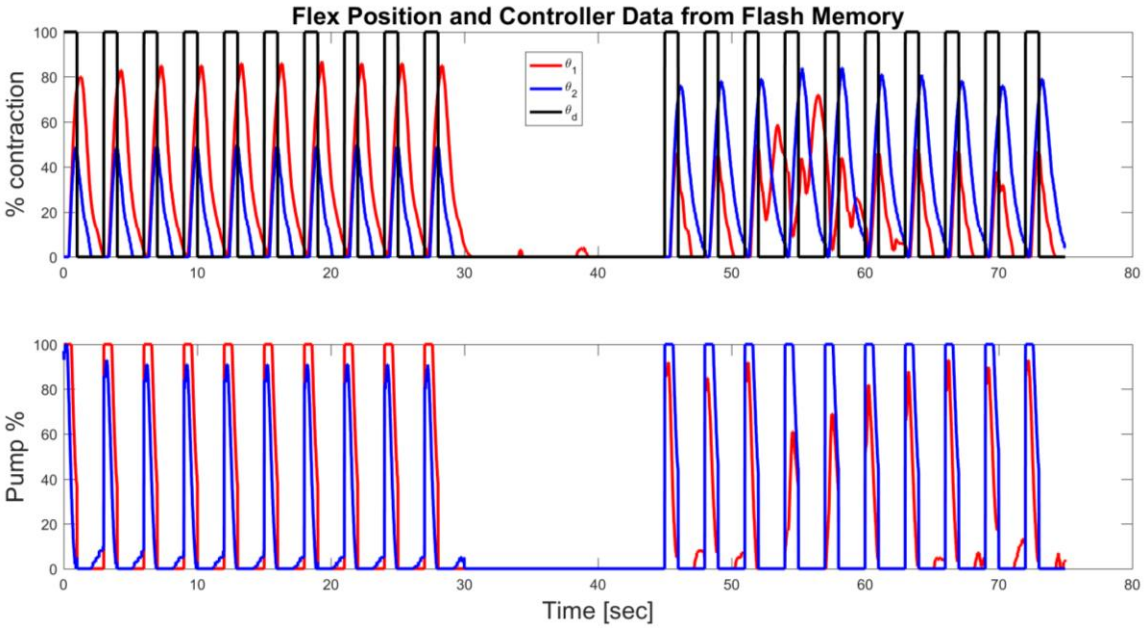

Fig. 58. Flex sensor, control waveform, and pump signals collected with flash memory.

6.2.2 Open-Water Testing

Video collected during open-water testing was not analyzed for swimming speed; it was used to show proof of concept that the JenniFish is capable of performing in un-controlled environments.

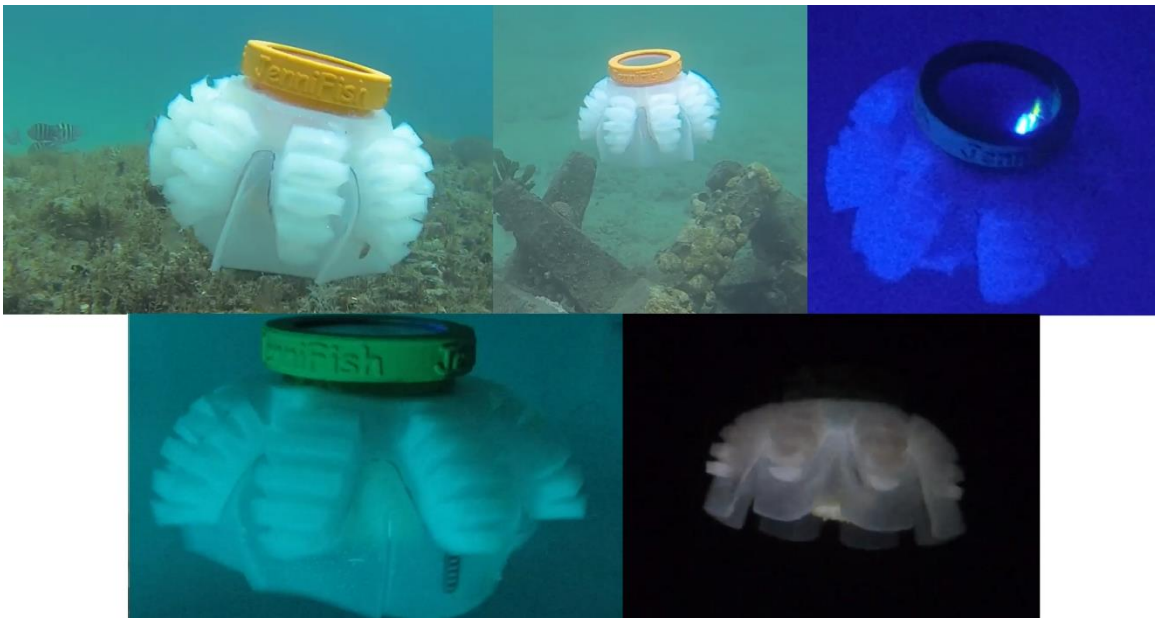

Fig. 59: Snapshots of the JenniFish during open-water testing.

7 DISCUSSION

Testing focused on four hypotheses: (1) that the JenniFish could propel itself upward, (2) that implementing position feedback control would improve net thrust production, (3) that reducing cycle frequency would improve net thrust production, and (4) that the JenniFish could produce a turning moment. Results given in the previous section have adequately proven the first hypothesis – the JenniFish's actuation method, chosen to mimic that of oblate medusae, has consistently proven capable of propelling the vehicle upward against slight negative buoyancy. Both Revision #1 and Revision #2 produced net upward thrust during constrained in-line load cell testing; and free-swimming video footage showed them swimming upward despite being negatively buoyant.

The results do not support hypothesis 2. The position feedback PD control methodology did not increase net thrust production to the degree that was expected; however, PD control only performed worse than no control at 0.333Hz. At 0.435Hz the net thrust per cycle of no control at 0.435Hz was not determined to be statistical different from the net thrust per cycle produced with offset square wave control at the same frequency. Additionally, the difference between net force production of no control and offset square wave control at 0.435Hz was not larger than the measurement sensor drift. However, net thrust for no control at 0.333Hz was larger than both control wave forms at 0.333Hz. The difference was greater than sensor drift and no control at 0.333Hz was deemed statistically different than both square wave and offset square wave control when

net cycle thrust was compared. This may be because the controller selection was not appropriate for the discovered underdamped vehicle dynamics, due to the system only being capable of controlled contraction. No control created greater peak force; examining flex data shows that full contraction is rarely achieved during square wave control. This could be a result of steady state error or poor flex calibration methods. Improved controller tuning might have helped slightly; however, the flex sensors did not prove to be nearly as reliable or durable as intended, perhaps due to their physical configuration within the system. Less than exceptional flex sensor performance certainly did not help. It is for this reason that future work is suggested to improve sensor implementation or the use of a different position feedback controller.

Another important consideration when evaluating hypotheses 2, 3, and 4 with the quantitative, constrained load cell results is that the tests were constrained to zero velocity. Thrust at zero speed (like that gathered during bollard-pull tests for propellers), is not necessarily representative of performance during motion. The JenniFish was found to be quite capable of free-swimming movement despite the very small net force numbers collected. Using constrained testing to determine optimal control methodology and cycle frequency in terms of trust might not be truly illustrative. Since thrust is generally velocity dependent, at least to some degree, the zero-speed load cell tests may not be the most accurate way to measure thrust.

Interestingly, results did support hypothesis 3, though not conclusively. The largest average net force was produced during testing with purely timing based control at 0.435Hz and 39.1% duty cycle. However, the difference in net force production between no control at 0.435Hz and no control at 0.333Hz was not greater than the sensor drift

value. Fig. 60 contains three cycles of flex sensor and load cell data that was collected simultaneously during purely timing based control at 0.333Hz and 33.3% duty cycle.

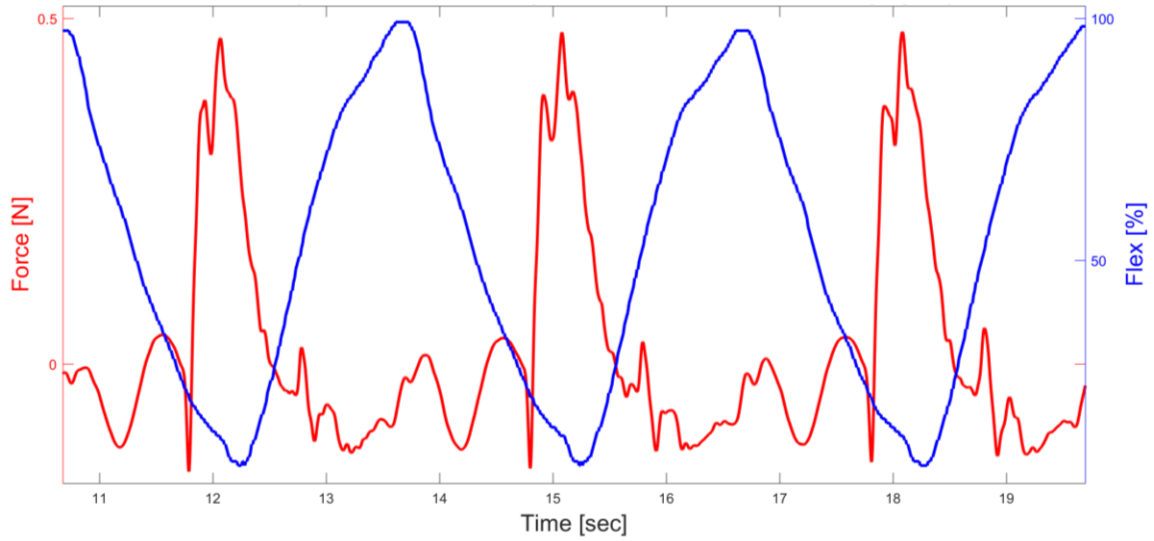

Fig. 60. 3 cycles of flex with in-line force data (no control, 0.333Hz, 33.3% DC).

Fig. 61contains three cycles of flex sensor and load cell data that was collected simultaneously during purely timing based control at 0.435Hz and 39.1% duty cycle.

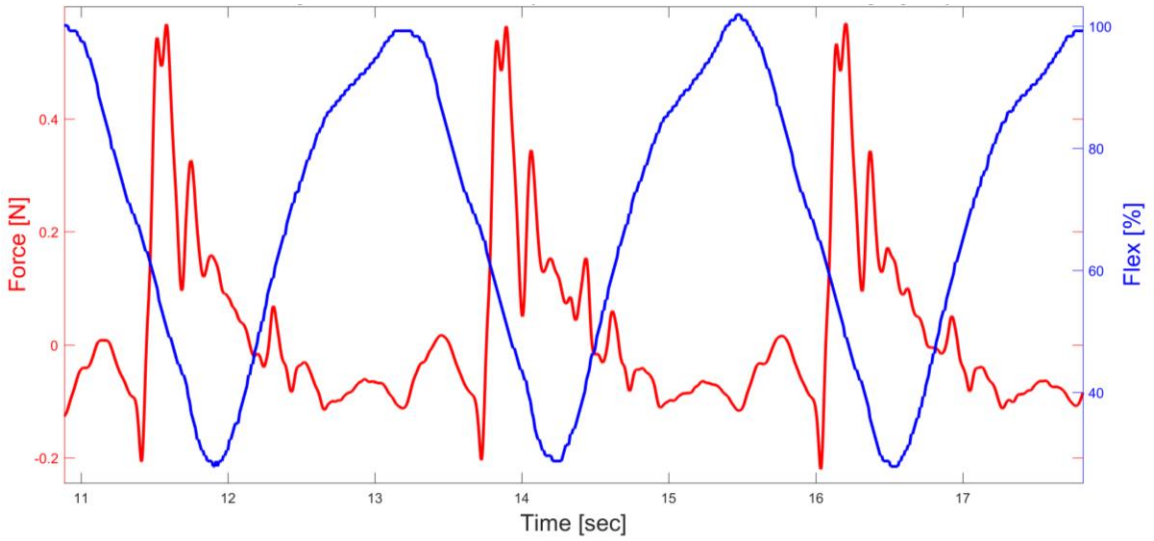

Fig. 61. 3 cycles of flex with in-line force data (no control, 0.435Hz, 39.1% DC).

Assuming that any time offset between serial and DAQ collection is negligible, the plots show when max thrust is produced during the swim cycle, which could help determine optimal cycle configuration in future work; noting that the most thrust is produced at the

end of relaxation/ beginning of contraction, further work could test how net thrust is impacted when swimming is done with cycle contraction from 0 to 50%, for example.

Finally, free-swimming tests visually proved hypotheses 4 – the JenniFish was able to turn itself bi-directionally. Positive net load cell data (though small in magnitude) was registered by the bending load cell test configuration, providing support for this hypothesis as well. However, no statistically relevant conclusions regarding controller or cycle frequency adjustment was gained from this testing. Additionally, a secondary smaller spike not seen during in-line testing was witnessed during bending tests. Further testing could be done to find out why this is a by-product of one-sided actuation.

Overall, the work accomplished with this thesis has displayed promising results thus far. The vehicle compares well with other robotic jellyfish and has proven itself to be a versatile platform with room for further development. The results displayed here will hopefully facilitate future work, including improvements on testing methodology. Ideas for additional vehicle revisions, such as outfitting the JenniFish with a pressure sensor so that it could hold a certain depth during data collection, could also fuel future work done with this system. Currently, scientists are trying to uncover the full importance of jellyfish to ocean ecology; perhaps the JenniFish platform can help them understand motion patterns of jellyfish, leading to a better knowledge of the ecosystem [33].

8 CONCLUSIONS

A novel self-contained soft robotic vehicle was designed, built, controlled and tested. It displayed the swimming motions characteristic of jellyfish while propelling itself upward despite slight negative buoyancy – during aquarium, ocean, and pool testing. The system also proved it could swim up through a hole smaller than its relaxed diameter and turn itself with one-sided actuation. Furthermore, free-swimming characteristics such as speed, proficiency, and power consumption are the same or better than other robotic jellyfish platforms. The vehicle was found to display underdamped dynamics, which should be considered when choosing any future control methods. In summary, PD position feedback control was not more effective than purely timing based control with the control waveforms and frequencies tested. However, results indicated that increasing cycle frequency to 0.435Hz was beneficial to net thrust production. Further tests should be done to verify this and to explore other cycle configurations. Although useful, further quantitative testing is also suggested since results did not conclusively demonstrate that zero-velocity testing accurately represents force or moment magnitudes. Finally, the performance baselines gathered during testing provide compelling evidence for the platform's continued use and improvement.

9 APPENDIX

9.1 Electronics Schematics

Detailed schematic figures are included in this section of the appendix for interested readers and replication of work. Only the latest motherboard (revision B) was included, since it improved on the first revision's sensor inclusion, test points, and circuitry.

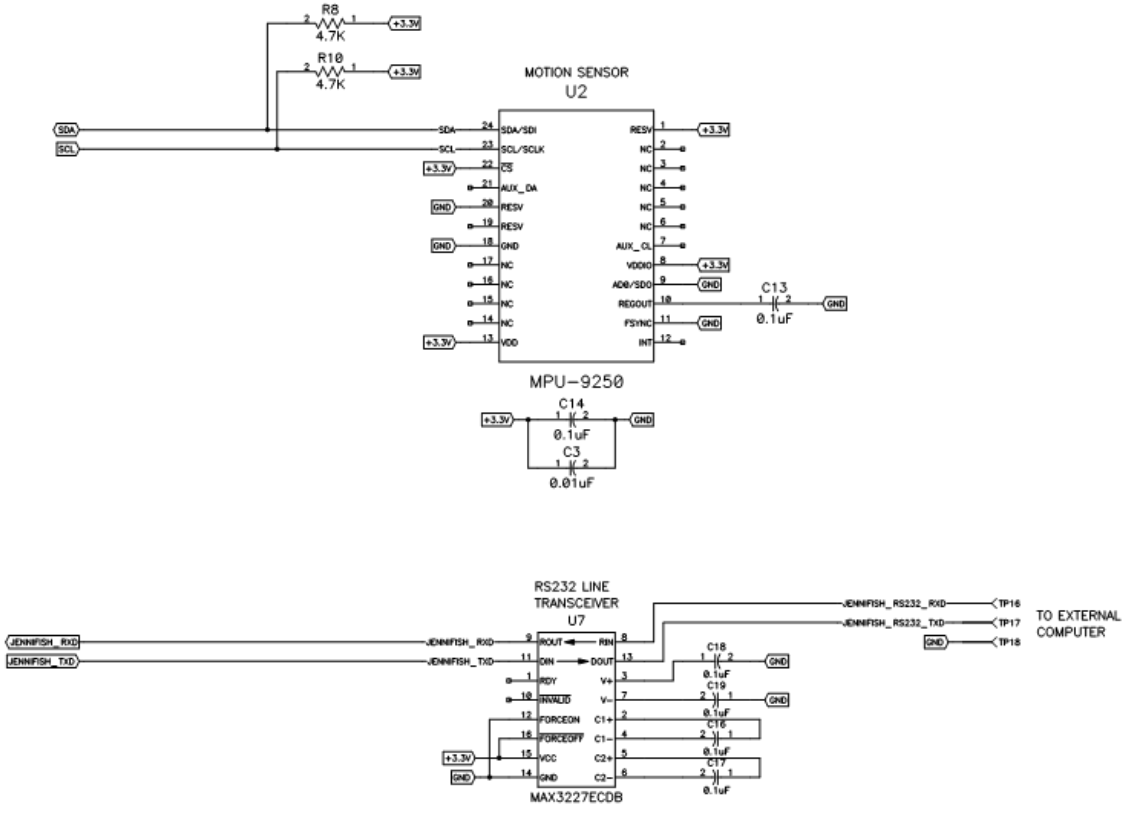

Fig. 62. Schematic of 9 DoF IMU (top) and flash memory chip (bottom).

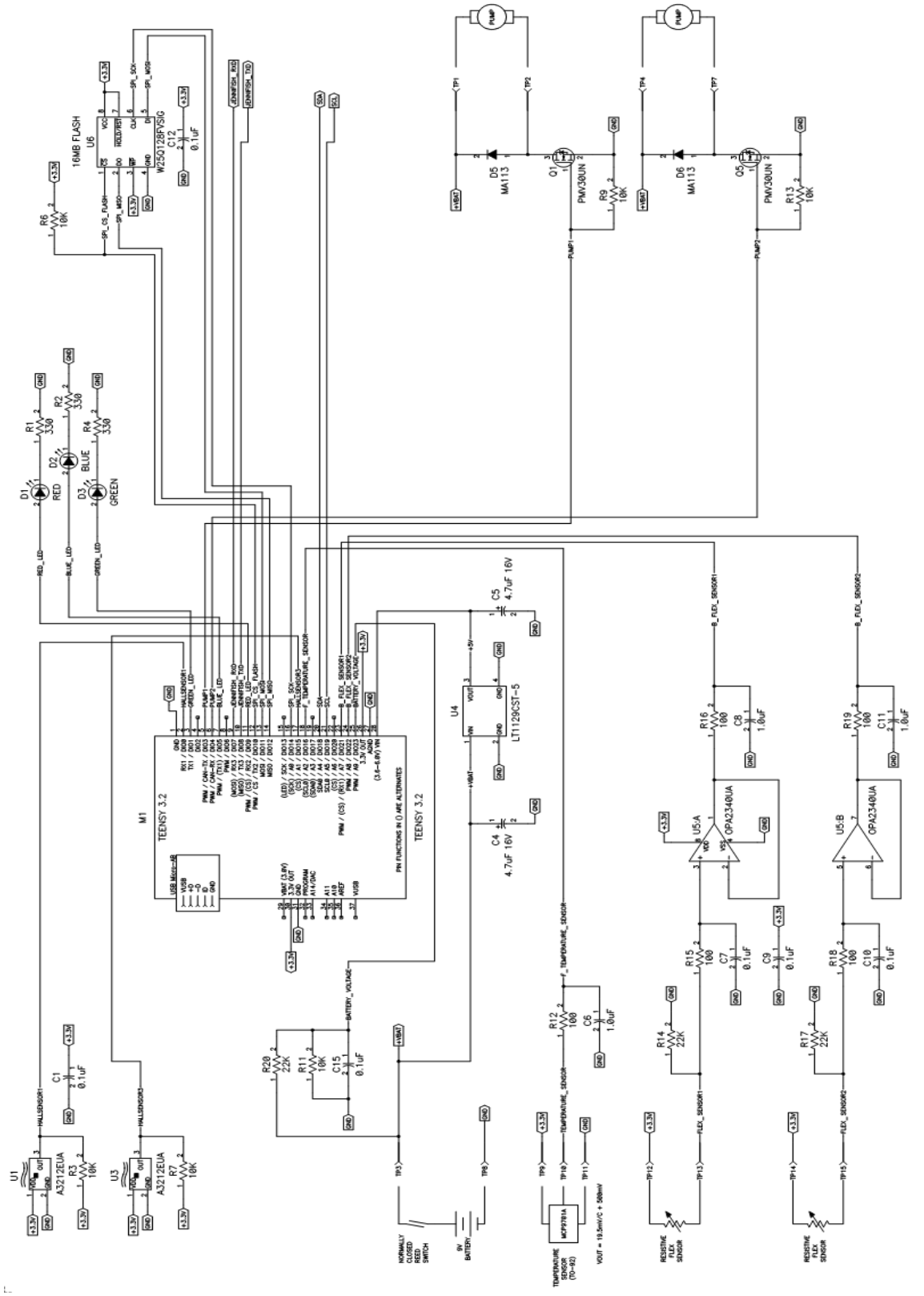

Fig. 63. Schematic including Teensy microcontroller and circuits connecting sensors.

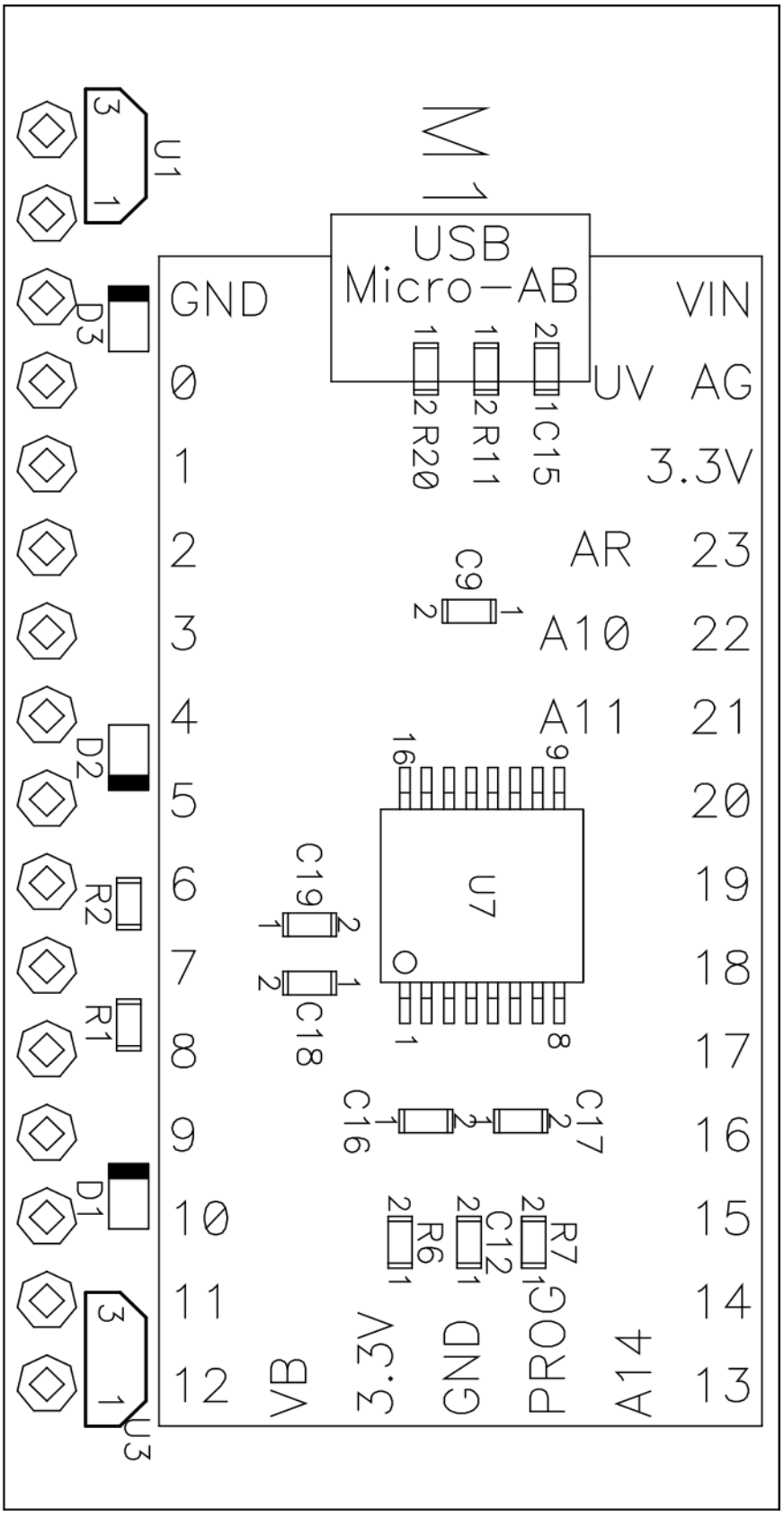

Fig. 64. Motherboard top assembly schematic.

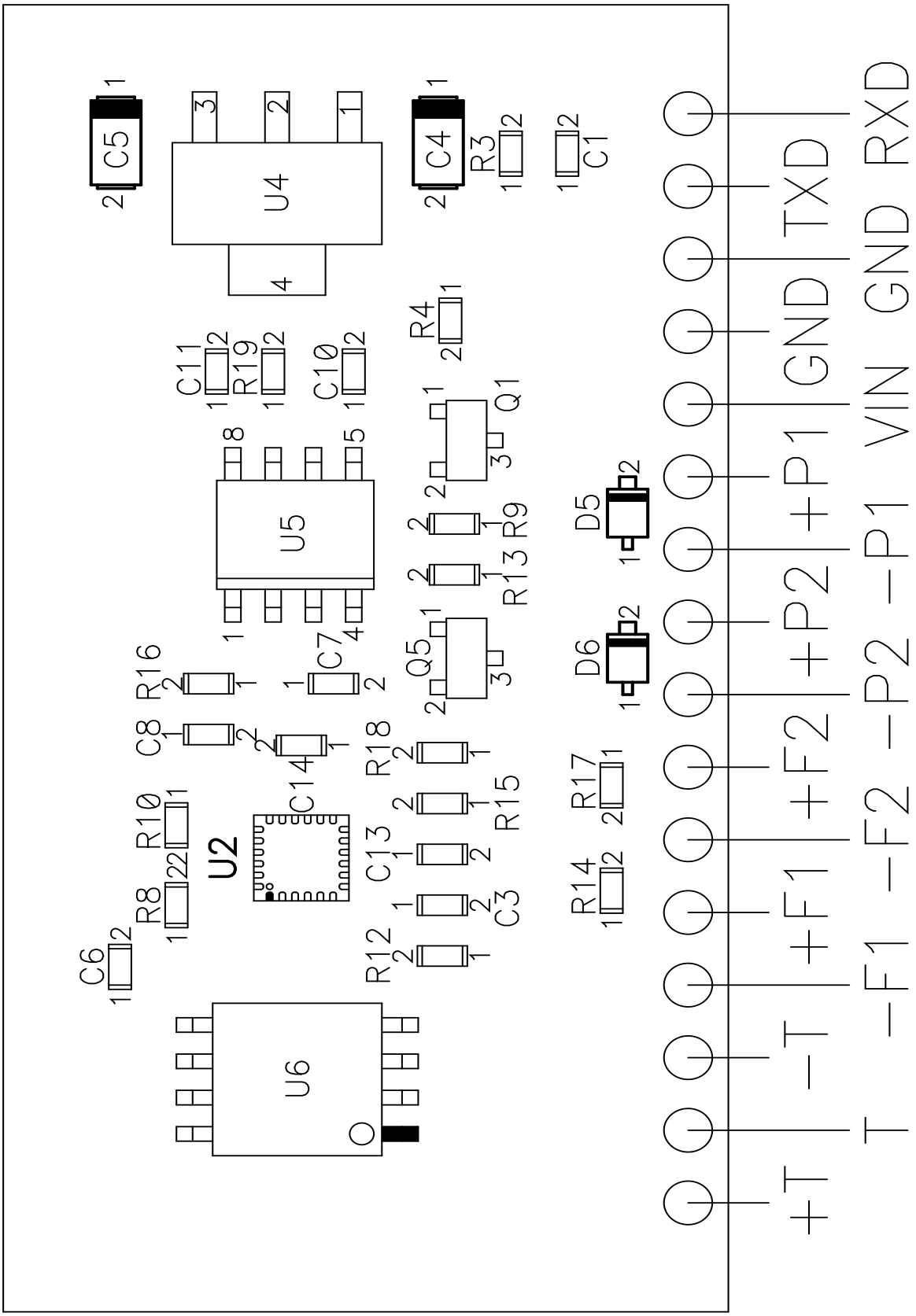

Fig. 65. Motherboard bottom assembly.

9.2 Software Package

The full software package consists of the following files:

- JenniFish_wFlash.ino
- JenniFish.h
- JenniFlash.h
- JenniFlash.cpp
- inClass.h
- inClass.cpp
- Audible.h
- Audible.cpp

To upload properly, all eight files need to be within a folder titled ‘JenniFish_wFlash’ and the TeensyDuino add-on for the Arduino IDE must have been previously downloaded. Only the main Arduino code, found in the JenniFish_wFlash.ino has been reproduced here for organizational purposes.

```
//JenniFish Software Package Revision 3 (June 2016)
#include "jenniFish.h"
#include "inClass.h"
#include "Audible.h"
#include <SPI.h>
#include "Arduino.h"
#include "JenniFlash.h"
Audible greenLED(jenni_led_g,jenni_led_g);
JenniFlash save; // 'initiator', set an instance of class JenniFish
```

```
inClass hall_1(input_hall_1);
inClass hall_3(input_hall_3);
long int cpt = 0; // counter
////////////////////////////////////////////////////////////////////////////////////////////////////
void setup() {
  Wire.begin();
  // Set accelerometers low pass filter at 5Hz
  I2CwriteByte(MPU9250_ADDRESS,29,0x06);
  // Set gyroscope low pass filter at 5Hz
  I2CwriteByte(MPU9250_ADDRESS,26,0x06);
  // Configure gyroscope range
  I2CwriteByte(MPU9250_ADDRESS,27,GYRO_FULL_SCALE_250_DPS);
  // Configure accelerometers range
  I2CwriteByte(MPU9250_ADDRESS,28,ACC_FULL_SCALE_2_G);  // was 4G
  // Set by pass mode for the magnetometers
  I2CwriteByte(MPU9250_ADDRESS,0x37,0x02);
  // Request continuous magnetometer measurements in 16 bits
  I2CwriteByte(MAG_ADDRESS,0x0A,0x16);
  // Initialize outputs
  pinMode(jenni_led_g, OUTPUT);
  pinMode(jenni_led_b, OUTPUT);
  pinMode(jenni_led_r, OUTPUT);
  pinMode(jenni_pump_1, OUTPUT);
```

```
  pinMode(jenni_pump_2, OUTPUT);
  // Initialize Flash
  Serial.begin(115200);
  delay(5000);
  if(save.begin()==1) {
    Serial.println("memory full");  }
  Serial.println("Send 'Show' to retreive recorded data");
  Serial.println("data to store:");
// Initiaze RX and TX
Serial3.begin(115200); // Defaults to 10-bit resolution, but can change to 12-bit.
Serial3.setTimeout(50);
while(!Serial3.available()) {
// wait for Serial3 to become available }
kp = Serial3.parseFloat();
kd = Serial3.parseFloat();
  hall_1.begin();  // begin method in inClass sets up the input pins
  hall_3.begin();
  tempThresh = analogRead(input_temp_sense) + 10;
  report.ready();  // report is an object created in the Audible class   }
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void loop()  {
  sTime = millis();  // lowest increment is approximately 3; could call micros()
  if (save.memoryFull) {
```

```
    greenLED.beep(5);  }
  else if (pCase != 0)  {
    greenLED.dash(2,20,1);  }
  serialSniffer();  // looks for user input
//****************** IMU code ******************
  Wire.begin();
  // Read accelerometer and gyroscope
     uint8_t Buf[14]; // unsigned integer with 8 bits (1 byte) of memory
     I2Cread(MPU9250_ADDRESS,0x3B,14,Buf);
  // Create 16 bits values from 8 bits data
     int16_t Ax=-(Buf[0]<<8 | Buf[1]); // only 2 bytes of memory instead of 4
     int16_t Ay=-(Buf[2]<<8 | Buf[3]);
     int16_t Az=-(Buf[4]<<8 | Buf[5]); // added negative sign
  // Remove zero g bias and divide by sensitivity
     int sensitivity = 16384;
     float xbmean = 0.06*sensitivity;
     float ybmean = 0.06*sensitivity;
     float zbmean = 0.08*sensitivity;
     float ax = (Ax-xbmean)/sensitivity;
     float ay = (Ay+ybmean)/sensitivity;
     float az = (Az+zbmean)/sensitivity;
  // Apply transformation to get data w.r.t. JenniFish
     const float pi = 3.14159265359;
```

```
    float theta = 15.67*(pi/90);
    axj = (-sin(theta)*ax+cos(theta)*az)*10000;
    ayj = (-ay)*10000;
    azj = (cos(theta)*ax+sin(theta)*az)*10000;
  // Gyroscope
    int16_t Gx=-(Buf[8]<<8 | Buf[9]);
    int16_t Gy=-(Buf[10]<<8 | Buf[11]);
    int16_t Gz=Buf[12]<<8 | Buf[13];
    int gyroSensitivity250 = 131; // LSB/(deg/s)
    gx = (Gx / gyroSensitivity250) - 11.0; // deg/s
    gy = (Gy / gyroSensitivity250) + 27.0; // deg/s
    gz = (Gz / gyroSensitivity250) - 0.80; // deg/s
//**************************************
  hall_1.bRead(&h1State); // method bRead updates the variable h1State
  hall_3.bRead(&h3State);
  flex1 = analogRead(input_flex_1);
  flex2 = analogRead(input_flex_2);
  f1 = mapping(flex1, flex1_max, flex1_min, 0, 100); // constrain(map,0,100);
  f2 = mapping(flex2, flex2_max, flex2_min, 0, 100); // constrain(map,0,100);
  temperature = analogRead(input_temp_sense);
  batVolt = analogRead(batVolt_pin) * (10.6 / 1024.0);
  if (batVolt > 6.0) {
    digitalWrite(jenni_led_b, HIGH);   }
```

```
else  {
  digitalWrite(jenni_led_b, LOW); }
pCase = switchBoard(); // Define JenniFish program modes (case 0 is default)
switch (pCase)
{ // NO DELAY IN SWITCH CASES
  case 0: // Idle
    pump1 = 0;
    pump2 = 0;
    stage1 = 0;
    stage2 = 0;
    jenisAWESOME = 1;
    startt = millis();
    t = 0;
    sineTime = 0;
    eLast1 = 0;
    eLast2 = 0;
    cal1_min = 0;
    cal2_min = 0;
    cal1_max = 0;
    cal2_max = 0;
    break;
  case 1: // Calibrate maximum and minimum flex sensor values
    if (millis() - startt < 80000)  {
```

```
    swim(100, 0, 100, 0, 1000, 4000);  }
   else if (!flexCalibration());
   else  {
    flex1_min = cal1_min / iC;
    flex1_max = cal1_max / iR;
    flex2_min = cal2_min / iC;
    flex2_max = cal2_max / iR;
    pCase = 0;  }
   break;
 case 2: // Swim
   if (millis() - startt < 46000)  {
    swim(100, 0, 100, 0, 900, 2300);
    saveVariables();  }
   else pCase = 0;
   break;
 case 3: // Swim
   if (millis() - startt < 60000)  {
    swim(100, 0, 100, 0, 1000, 3000);
    saveVariables();  }
   else pCase = 0;
   break;
 case 4: // Square Wave Tracking
   if (millis() - startt < 60000)  {
```

```
      squareWave(1000, 3000, 100, 0);
      thetaD1 = thetaD;
      thetaD2 = thetaD;
      controller();
      saveVariables();  }
    else pCase = 0;
    break;
  case 5: // Offset Square Wave Tracking
    if (millis() - startt < 60000)  {
      squareWave(900, 2300, 100, 45);
      thetaD1 = thetaD;
      thetaD2 = thetaD;
      controller();
      saveVariables();   }
    else pCase = 0;
    break;
  default:
    pCase = 0;
    break;  }
// Set PWM limits (255 means 100% PWM or battery voltage)
report.thread();
greenLED.thread();
pinMode(jenni_pump_1, OUTPUT);
```

```
  pinMode(jenni_pump_2, OUTPUT);
  if (pump1 <= 100 && pump2 <= 100)  {
    analogWrite(jenni_pump_1, round(pump1 * (255.0 * 6.0 / batVolt / 100.0)));
    analogWrite(jenni_pump_2, round(pump2 * (255.0 * 6.0 / batVolt / 100.0))); }
  else  {
    analogWrite(jenni_pump_1, 0);
    analogWrite(jenni_pump_2, 0);     }
  // Serial3 Print
  out();
  // LOOP MANAGER
  loopTime = millis() - sTime;
  if (loopTime < jenni_loop_period) delay(jenni_loop_period - loopTime);
  else  { // report error
    Serial.print("Over loop error  ");
    Serial.println(loopTime); } }
////////////////////////////////////////////////////////////////////////////////////////////////////////////
double mapping(double x, double in_min, double in_max, double out_min, double
out_max) {
  return (x - in_min) * (out_max - out_min) / (in_max - in_min) + out_min; }
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void swim(int c1, int r1, int c2, int r2, int cT, int rT) {
  switch (stage1)  {
    case 0: //intitialize
```

```
      actTime1 = millis();
      stage1++;
    case 1: // contract
      pump1 = c1;
      pump2 = c2;
      if ((millis() - actTime1) > cT) {
        stage1++; }
      break;
    case 2: //relax
      pump1 = r1;
      pump2 = r2;
      if ((millis() - actTime1) > rT) {
        stage1 = 0; }
      break;  }  }
////////////////////////////////////////////////////////////////////////////////////////////////////////////////
boolean flexCalibration() {
  switch (stage2)  {
    case 0: //intitialize
      actTime2 = millis();
      iC = 0.0;
      iR = 0.0;
      stage2++;
    case 1: // contract
```

```
  pump1 = 100;
  pump2 = 100;
  if ((millis() - actTime2) >= 1400 && (millis() - actTime2) <= 1600) {
    cal1_min += flex1;
    cal2_min += flex2;
    iC += 1.0;  }
  if ((millis() - actTime2) > 1600)   {
    stage2++;  }
  break;
case 2: //relax
  pump1 = 0;
  pump2 = 0;
  if ((millis() - actTime2) >= 5800 && (millis() - actTime2) <= 6000)    {
    cal1_max += flex1;
    cal2_max += flex2;
    iR += 1.0;  }
  if ((millis() - actTime2) > 6000)   {
    stage2++;  }
  break;
case 3: // Return done
  stage2 = 0;
  return true;
  break; }
```

```
  return false;  }
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void squareWave(int cT, int rT, int cLimit, int rLimit)  {
  if (millis() - t >= rT)   {
    t = millis();  }
  if (millis() - t <= cT) {
    thetaD = cLimit;  }
  if (millis() - t > cT) {
    thetaD = rLimit; } }
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void sawTooth(double delta)  {
  if (millis() - t >= 5000)   {
    t = millis();     }
  if (millis() - t <= 1000)    {
    thetaD = 100;      }
  if (millis() - t > 1000)    {
    thetaD = constrain(next, 0, 100);    }
  next = thetaD - delta;   }
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void sineWave(double dt)  {
  w = 3.1415926536 * (2.0 / 10.0); // 10 second period
  thetaD = 50 * sin(w * sineTime) + 50;
  sineTime += dt; }
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void multiStep()   {
  if (millis() - t >= 45000)   {
    t = millis();  }
  if (millis() - t <= 15000)  {
    thetaD = 30;  }
  if ((millis() - t) > 15000 && (millis() - t) <=30000)  {
    thetaD = 60;  }
  if (millis() - t > 30000)  {
    thetaD = 90;  }  }
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void controller()   {
  e1 = thetaD1 - f1;
  e2 = thetaD2 - f2;
  thetaDot1 = e1 - eLast1;
  thetaDot2 = e2 - eLast2;
  p1 = kp * e1 + kd * (thetaDot1 / jenni_loop_period);
  p2 = kp * e2 + kd * (thetaDot2 / jenni_loop_period);
  pump1 = constrain(p1, 0, 100);
  pump2 = constrain(p2, 0, 100);
  eLast1 = e1;
  eLast2 = e2;  }
////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
void out()  {
  Serial3.print("F1:");
  Serial3.print(thetaD);
  Serial3.print(",F2:");
  Serial3.print(flex1);
  Serial3.print(",F3:");
  Serial3.print(f1);
  Serial3.print(",F4:");
  Serial3.print(thetaDot1);
  Serial3.print(",F5:");
  Serial3.println(p1);   }
////////////////////////////////////////////////////////////////////////////////////////////////////////////
void saveVariables()   {
  save.setSampleTime(millis());
  save.setBatteryVoltage((short int) batVolt);
  save.setTemp((short int) temperature);
  save.setFlexSensor1((short int) flex1);
  save.setFlexSensor2((short int) flex2);
  save.setpwmPump1((short int) pump1);
  save.setpwmPump2((short int) pump2);
  save.setaccelX((short int) axj);
  save.setaccelY((short int) ayj);
  save.setaccelZ((short int) azj);
```

```
  save.setgyroX((short int) gx);
  save.setgyroY((short int) gy);
  save.setgyroZ((short int) gz);
  save.settheta((byte) thetaD);
  save.setmapF1((byte) f1);
  save.setmapF2((byte) f2);
  save.setprogramNum((byte) pCase);
  save.write(); // prints variables to screen and saves them to the flash }
////////////////////////////////////////////////////////////////////////////////
void serialSniffer()  {
  if (Serial.available())  {
    String eventInp;
    while (Serial.available()) eventInp = eventInp += (char)Serial.read();
    if (eventInp.equalsIgnoreCase("Show"))   {
      save.printAll();
      Serial.println("Send 'Delete' to clear memory");    }
    if (eventInp.equalsIgnoreCase("delete"))    {
      save.eraseFlash();    }  } }
////////////////////////////////////////////////////////////////////////////////
// This function read Nbytes bytes from I2C device at address Address.
// Put read bytes starting at register Register in the Data array.
void I2Cread(uint8_t Address, uint8_t Register, uint8_t Nbytes, uint8_t* Data)
{     // Set register address
```

```
  Wire.beginTransmission(Address);
  Wire.write(Register);
  Wire.endTransmission();
  // Read Nbytes
  Wire.requestFrom(Address, Nbytes);
  uint8_t index=0;
  while (Wire.available())
    Data[index++]=Wire.read();    }
////////////////////////////////////////////////////////////////////////////////////////////////
// Write a byte (Data) in device (Address) at register (Register)
void I2CwriteByte(uint8_t Address, uint8_t Register, uint8_t Data)   {
  // Set register address
  Wire.beginTransmission(Address);
  Wire.write(Register);
  Wire.write(Data);
  Wire.endTransmission();   }
////////////////////////////////////////////////////////////////////////////////////////////////
byte switchBoard()   // hall_1 cycles through programs and hall_3 runs programs
{     if (h1State == 1)  // button state
  {   h1State = 0;     // clear press state flag
    if (pCase != 0)  // if in a program return to idle
    {   report.beep(5); // Audible response to press
      return 0;      // jump out of program   }
```

```
  else pTemp++;   // otherwise increment the temporary program variable   }
if (h1State == 2)   {
  h1State = 0;
  if (pCase != 0)   {
    report.beep(5);
    return 0;  // jump out of program   }
  else report.dash(pTemp);  }
if (pTemp > 5) pTemp = 0;  // loop program count
if (h3State == 2)  {
  h3State = 0;
  if (pCase != 0)    {
    report.beep(5);
    return 0;  // jump out of program    }    }
if (h3State == 3)  {
  h3State = 0;
  report.beep(5);
  return 0;// return shutdown PCase  }
if (h3State == 1)  {
  h3State = 0;
  if (pCase != 0)    {
    report.beep(5);
    return 0;    // jump out of program    }
  else   {
```

```
    report.beep(3);
    return pTemp;  }  }
 else return pCase;   }
```

9.3 Supplementary Material

This section contains URLs to help easily locate the YouTube videos of free-swimming tests done with both JenniFish revisions.

- Test highlights: https://www.youtube.com/watch?v=8QoGTTMCj50
- Full ocean footage: https://www.youtube.com/watch?v=Aaugq16rCAw
- Pass-through footage: https://www.youtube.com/watch?v=RwimKVM7Lok
- Turning footage: https://www.youtube.com/watch?v=9HOlvF0xW_E
- Rev. 2 tank swimming: https://www.youtube.com/watch?v=M4_2yL830vw
- Rev. 1 tank swimming: https://www.youtube.com/watch?v=gCydmQWxdSA

Also included is the FCRAR 2016 conference proceedings URL, which contains a paper that was submitted pertaining to preliminary results up to JenniFish Rev. #1 testing.

- http://www.eng.fiu.edu/mme/robotics/fcrar2016/fcrar2016proceedings.pdf

9.4 Data Analysis

MATLAB was used to do load cell calibration and data parsing, including the generation of all data plots. For this reason, the in-line calibration and bending scripts have been included in this appendix section.

```
% Calibration for load cell with default amplifier gain
clear all
close all
clc
% For on-table calibration
g = 9.81/1000; % [gm/s^2]
mass = [0; 1; 2; 5; 10; 20; 50; 100]; % [g]
for i = 1:length(mass)
```

```
    % Calibration: compression
    file = ['./bending_06_11_16/calBefore/Lup'…
                          …num2str(mass(i),'%03d'),'.mat'];
    load(file);
    volts(i,:) = x2(:,2);
end
file = './bending_06_11_16/calBefore/Lup000b.mat';
load(file);
volts((i+1),:) = x2(:,2);
t = x2(:,1)'; % Time vector [s]
freq = 1/t(2); % [Hz]
duration = (length(t)-1)/freq; % [s]
Nsamples = length(t);
% Low pass filter raw data
filter_cutoff = 30; % [Hz]
run.sample_rate=freq; % [Hz]
Wn=filter_cutoff/(run.sample_rate/2);
[b,a]=butter(2,Wn);
for j = 1:length(volts(:,1))
filt_volts(j,:) = filtfilt(b,a,volts(j,:));
av_volts(j,:) = mean(filt_volts(j,:));
end
drift = av_volts(1)- av_volts(end); % [V]
%% Curve fit
av_v = av_volts(1:(end-1));
xfit_all=linspace(av_v(1),av_v(end));
[fitdat1,gof1,out1] = fit(av_v,mass,'poly1'); %
polyfit(x,y,fitType)
yfit1=fitdat1(xfit_all);
driftN = drift*fitdat1.p1*g % [N]
%% Plot
figure(1)
plot(av_v,mass,'ok'); hold on;
plot(xfit_all,yfit1,'-m')
xlabel('Voltage [V]')
ylabel('Mass [g]')
legend('Averages','Linear Fit','Location','best')
slope = fitdat1.p1;
figure(2)
plot(t,filt_volts(1:(end-1),:),'-')
xlabel('Time [s]')
ylabel('Voltage [V]')
title('Filtered Voltage Data')
legend('0','1g','2g','5g','10g','20g','50g','100g','Locatio
n','best')
```

```
%% In-line Load Cell Testing 6/11/16
% (17V, Green to Red in, Orange to Black in)
% 9V, 0.65A when both pumps are on
% case2: swim (1on, 2off)
% case3: swim (.9on, 1.4off)
% case4: sqWave (1on, 2off)
% case5: offSqW (1on, 2off, 25)
% case6: offSqW (0.9on, 1.4off, 45)
clear;
close all;
clc;
%% Load raw data files
load('slope.mat')
g = 9.81/1000;
cases = [0; 2; 3; 4; 5; 6; 8];
for k = 1:length(cases)-2
    load(['./Set3/case'
num2str(cases(k+1),'%01d'),'.mat']);
    flex1(k,:) = f1.signals.values(1,:)';
    flex2(k,:) = f2.signals.values(1,:)';
    dTheta1(k,:) = thetaD.signals.values(1,:)';
    dotTheta1(k,:) = thetaDot1.signals.values(1,:)';
    pump1(k,:) = p1.signals.values(1,:)';
end
serialT = f1.time;
serialFreq = 1/serialT(2); % Hz
for m = 1:length(cases)
    load(['./Set3/case' num2str(cases(m),'%01d'),'.mat']);
    fish(m,:) = x2(:,2);
end
t = x2(:,1)'; % Time vector [s]
daqFreq = 1/t(2); % [Hz]
duration = (length(t)-1)/daqFreq; % [s]
Nsamples = length(t);
%% Low pass filter raw data
filter_cutoff = 15; % [Hz]
run.sample_rate=daqFreq; % [Hz]
Wn=filter_cutoff/(run.sample_rate/2);
[b,a]=butter(2,Wn);
for n = 1:length(fish(:,1))
    filt_fish(n,:) = filtfilt(b,a,fish(n,:));
end
%% Calculate Net Force and Plot
% Find force in Newtons, find average force from first peak
to last peak,
% and plot first peak to last peak in frequency groups
(then 14 cycles)
```

```
zeroAv = mean(filt_fish(1,:));
Fn = (filt_fish-zeroAv)*slope*g;
drift = mean(Fn(1,:))-mean(Fn(end,:));
c = {'b','g','r','k','c','m','g'};
co = {'bo','go','ro','ko','co','mo','go'};
for i=2:length(Fn(:,1))-1
    [pkf, pki] =findpeaks(Fn(i,:),…
              …'MinPeakDistance',1000,'MinPeakHeight',0.3);
    hold on;  % I is index
    pkF = pkf(1:19);
    pkI = pki(1:19);
    pkForce(i-1,:) = pkF;
    pkIndices(i-1,:) = pkI;
    for p=1:length(pkI)-1 % Find average force per cycle
        avCycleForce(i-1,p)= mean(Fn(i,pkI(p):pkI(p+1)));
    end
    avForce_normalized(i-1)= mean(avCycleForce(i-1,5:end));
    %avForce_normalized(i-1) = mean(avCycleForce(i-1,:));
    avForce(i-1) = mean(Fn(i,pkI(5):pkI(end)));
    %avForce(i-1) = mean(Fn(i,pkI(1):pkI(end)));
    if i == 3 || i == 6
        figure(1)
          plot(t(pkI(5):pkI(end))-t(pkI(5)),…
                …Fn(i,pkI(5):pkI(end)),c{i},'LineWidth',2);
        hold on;
        % hightlight peaks that are selected
        plot(t(pkI(5:end))-t(pkI(5)),…
                          …pkF(:,5:end),co{i},'LineWidth',2)
        if i == 6
            plot(t(pkI(5):pkI(end))-t(pkI(5)),…
                 … Fn(1,pkI(5):pkI(end)),'k','LineWidth',2)
            plot(t(pkI(5):pkI(end))-t(pkI(5)),…
               …Fn(end,pkI(5):pkI(end)),'b','LineWidth',2);
            xlabel('Time [s]','fontsize',17)
            ylabel('Force [N]','fontsize',17)
            title('Last 14 Cycle Filtered Force Data
                (0.435Hz, 39.1% duty cycle)','fontsize',17)
            legend('No Control','Thrust Peaks','45% Offset
                   Sq Wave','Thrust Peaks','0','00',
              'Location','best','orientation','horizontal')
            set(gca, 'fontsize', 17)
            hold off;
        end
    else
        figure(2)
        plot(t(pkI(5):pkI(end))-t(pkI(5)),
               Fn(i,pkI(5):pkI(end)),c{i},'LineWidth',2);
```

```
        hold on;
        % hightlight peaks that are selected
        plot(t(pkI(5:end))-t(pkI(5)),
                    pkF(:,5:end),co{i},'LineWidth',1.5);
        if i == 5
            plot(t(pkI(5):pkI(end))-t(pkI(5)),
                  Fn(1,pkI(5):pkI(end)),'m','LineWidth',2)
            plot(t(pkI(5):pkI(end))-t(pkI(5)),
                 Fn(end,pkI(5):pkI(end)),'y','LineWidth',2)
            xlabel('Time [s]', 'fontsize', 17)
            ylabel('Force [N]', 'fontsize', 17)
            title('Last 14 Cycle Filtered Force Data
              (0.333Hz, 33.3% duty cycle)', 'fontsize', 17)
            legend('No Control','Thrust Peaks','Sq Wave',
                    'Peaks','25% Offset Sq Wave','Peaks',
                    '0','00','Location','best',
                    'orientation', 'horizontal')
            set(gca, 'fontsize', 17)
            hold off;
        end
    end
end
%% anova2
% Max thrust per cycle (last 14 cycles and all 18 cycles)
anova2([pkForce(2,5:end);pkForce(5,5:end)]);
anova2([pkForce(1,5:end);pkForce(3,5:end)]);
anova2([pkForce(1,5:end);pkForce(4,5:end)]);
anova2([pkForce(3,5:end);pkForce(4,5:end)]);
   % anova2([pkForce(2,:);pkForce(5,:)]);
   % anova2([pkForce(1,:);pkForce(3,:)]);
   % anova2([pkForce(1,:);pkForce(4,:)]);
   % anova2([pkForce(3,:);pkForce(4,:)]);
% Net-thrust per cycle (last 14 cycles and all 18 cycles)
anova2([avCycleForce(2,5:end);avCycleForce(5,5:end)]);
anova2([avCycleForce(1,5:end);avCycleForce(3,5:end)]);
anova2([avCycleForce(1,5:end);avCycleForce(4,5:end)]);
anova2([avCycleForce(3,5:end);avCycleForce(4,5:end)]);
   % anova2([avCycleForce(2,:);avCycleForce(5,:)]);
   % anova2([avCycleForce(1,:);avCycleForce(3,:)]);
   % anova2([avCycleForce(1,:);avCycleForce(4,:)]);
   % anova2([avCycleForce(3,:);avCycleForce(4,:)]);
%% Plot Serial with load cell data
sI = ceil(pkIndices/5);
sIndices = sI(:,5:end);
caseDescription = {'Last 14 Cycle Flex-Force Data (No
Control 0.333Hz, 33.3% duty cycle)','Last 14 Cycle Flex-
Force Data (No Control 0.435Hz, 39.1% duty cycle)','Last 14
```

```
Cycle Flex-Force Data (Sq Wave 0.333Hz, 33.3% duty
cycle)','Last 14 Cycle Flex-Force Data (25% Offset Sq Wave
0.333Hz, 33.3% duty cycle)','Last 14 Cycle Flex-Force Data
(45% Offset Sq Wave 0.435Hz, 39.1% duty cycle)'};
% Flex Data plotted on top of load cell data
for u = 1:5
    croppedT = t(pkIndices(u,5):pkIndices(u,end))-
t(pkIndices(u,5));
    croppedFn = Fn(u+1,pkIndices(u,5):pkIndices(u,end));
    croppedST = serialT(sIndices(u,1):sIndices(u,end))-
serialT(sIndices(u,1));
    croppedFlex1 = flex1(u,sIndices(u,1):sIndices(u,end));
    figure(11)
    [hAx,hLine1,hLine2] =
plotyy(croppedT,croppedFn,croppedST,croppedFlex1);
    title(caseDescription(u),'fontsize',17)
    xlabel('Time [sec]','fontsize',17);
    ylabel(hAx(1),'Force [N]','fontsize',17); % left y-axis
    hLine1.LineWidth = 2;
    set(hLine1,'color','red');
    ylabel(hAx(2),'Flex [%]','fontsize',17); % right y-axis
    hLine2.LineWidth = 2;
    set(hLine2,'color','blue');
    set(hAx,{'ycolor'},{'r';'b'})
%     pause
%     fileTitle = ['Flex',num2str(u+1)];
%     saveas(gcf,fileTitle,'jpg')
end
%% Plot remaining serial data
caseDescription = {'Last 14 Cycle Controller Data (No
Control 0.333Hz, 33.3% duty cycle)','Last 14 Cycle
Controller Data (No Control 0.435Hz, 39.1% duty
cycle)','Last 14 Cycle Controller Data (Sq Wave 0.333Hz,
33.3% duty cycle)','Last 14 Cycle Controller Data (25%
Offset Sq Wave 0.333Hz, 33.3% duty cycle)','Last 14 Cycle
Controller Data (45% Offset Sq Wave 0.435Hz, 39.1% duty
cycle)'};
for u = 1:5
    croppedST = serialT(sIndices(u,1):sIndices(u,end))-
serialT(sIndices(u,1));
    croppedThetaD =
dTheta1(u,sIndices(u,1):sIndices(u,end));
    croppedThetaDot =
dotTheta1(u,sIndices(u,1):sIndices(u,end));
    croppedPump = pump1(u,sIndices(u,1):sIndices(u,end));
    figure(12)
    subplot(3,1,1)
```

```
    plot(croppedST,croppedThetaD,'LineWidth',2);
    title(caseDescription(u),'fontsize',17)

ylabel('\boldmath$\theta_d$','fontsize',20,'interpreter','l
atex');
    subplot(3,1,2)
    plot(croppedST,croppedThetaDot,'LineWidth',2);

ylabel('\boldmath$\dot{\theta_d}$','fontsize',20,'interpret
er','latex');
    subplot(3,1,3)
    plot(croppedST,croppedPump,'LineWidth',2);
    xlabel('Time [sec]','fontsize',17);
    ylabel('Pump %','fontsize',15);
%     pause
%     fileTitle = ['Serial',num2str(u+1)];
%     saveas(gcf,fileTitle,'jpg')
end
```

9.5 Extended Results

This appendix section contains additional results from JenniFish Revision #2 in-line load cell testing which were left out of the thesis' main body to help streamline its content for readers. Hopefully this separation keeps the paper organized and focused on hypotheses while retaining scientific integrity.

9.5.1 Data Set 1

The following plots and tables pertain to the full 18 cycles of JenniFish Revision #2 in-line load cell data set 1. Fig. 66 contains the first set of load cell measurements at a frequency of 0.435Hz with a 39.1% duty cycle. Load cell drift for set 1 was 3.50mN

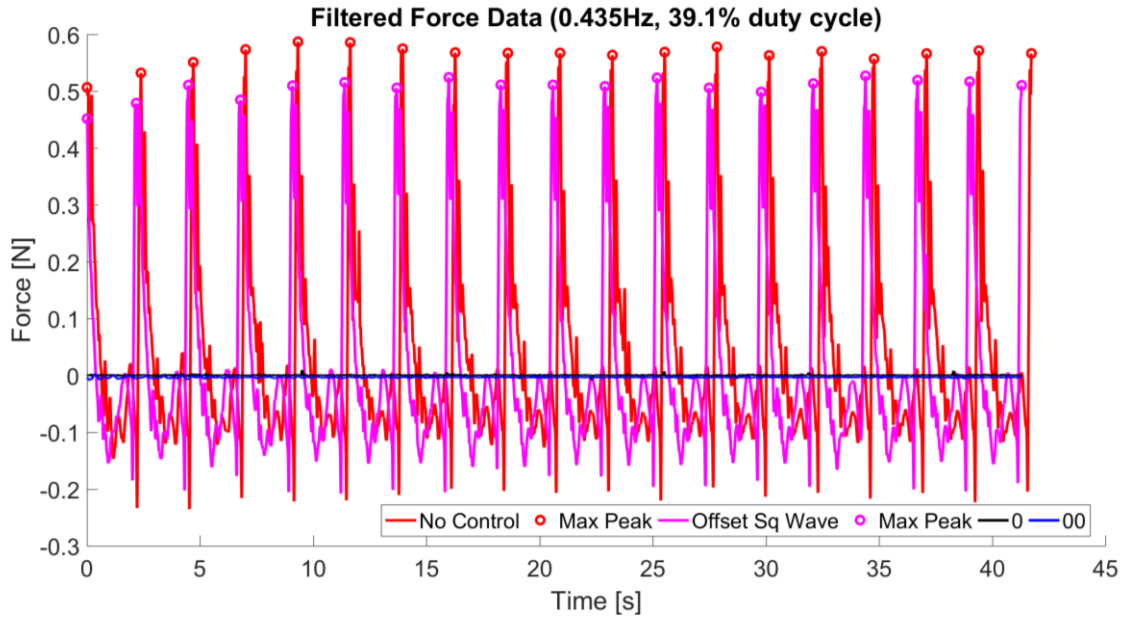

Fig. 66. Set 1 filtered force data (0.435Hz, 39.1% duty cycle).

Fig. 67 contains the first set of load cell measurements at a frequency of 0.435Hz with a 39.1% duty cycle.

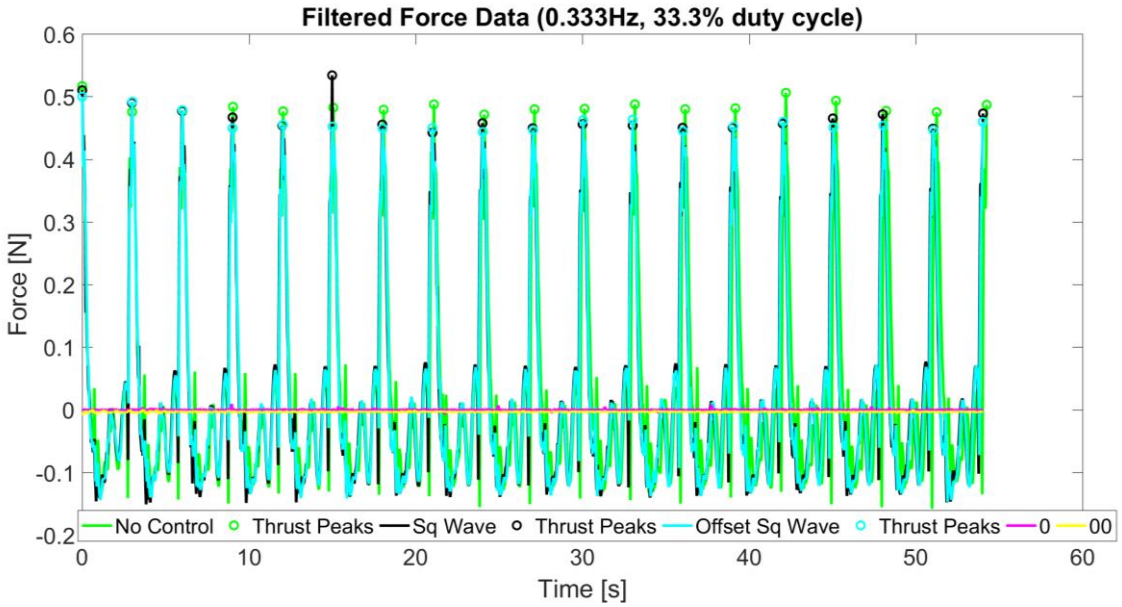

Fig. 67. Set 1 filtered force data (0.333Hz, 33.3% duty cycle).

Table 22 and Table 23 summarize the ANOVA2 results from MATLAB done for set 1 max thrust per cycle and net thrust per cycle respectively.

Table 22: Set 1 max thrust per cycle ANOVA2 results.

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.0002
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0.0033	0.1759
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.0317
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0929	0.0131

Table 23: Set 1 net thrust per cycle ANOVA2 results.

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0.0274	0.9998
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.0125
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.8431
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0776	0.9757

Fig. 68 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 1 no control 0.333Hz, 33.3% duty cycle.

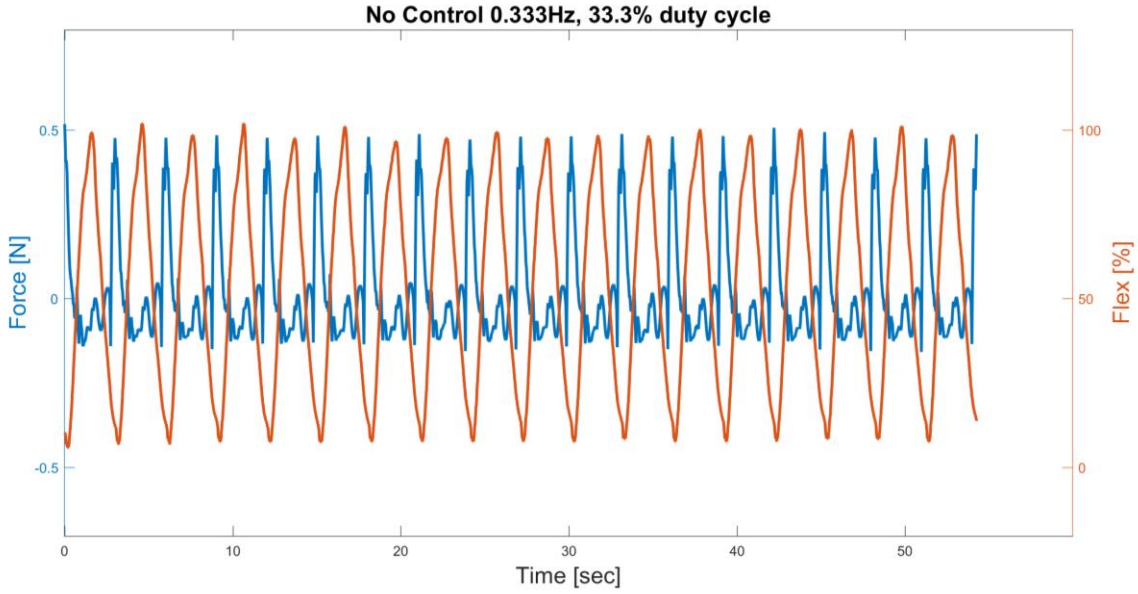

Fig. 68. Set 1 flex and force data for no control 0.333Hz, 33.3% duty cycle.

Fig. 69 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 1 no control 0.435Hz, 39.1% duty cycle.

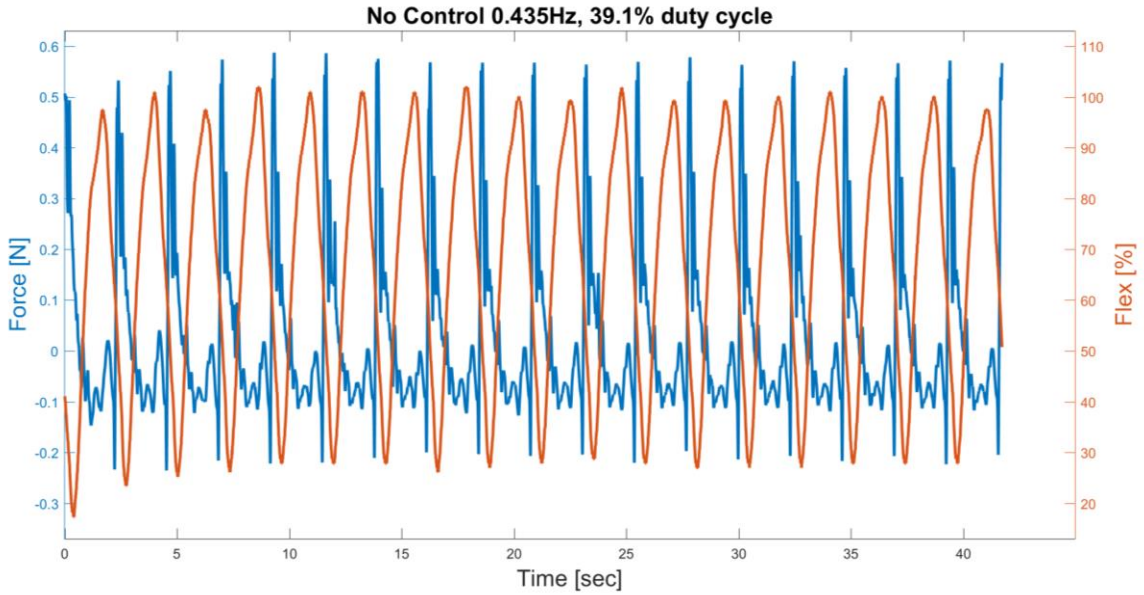

Fig. 69. Set 1 flex and force data for no control 0.435Hz, 39.1% duty cycle.

Fig. 70 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 1 square wave 0.333Hz, 33.3% duty cycle. Fig. 71 contains the controller data (associated with pump 1) that was collected via serial during testing.

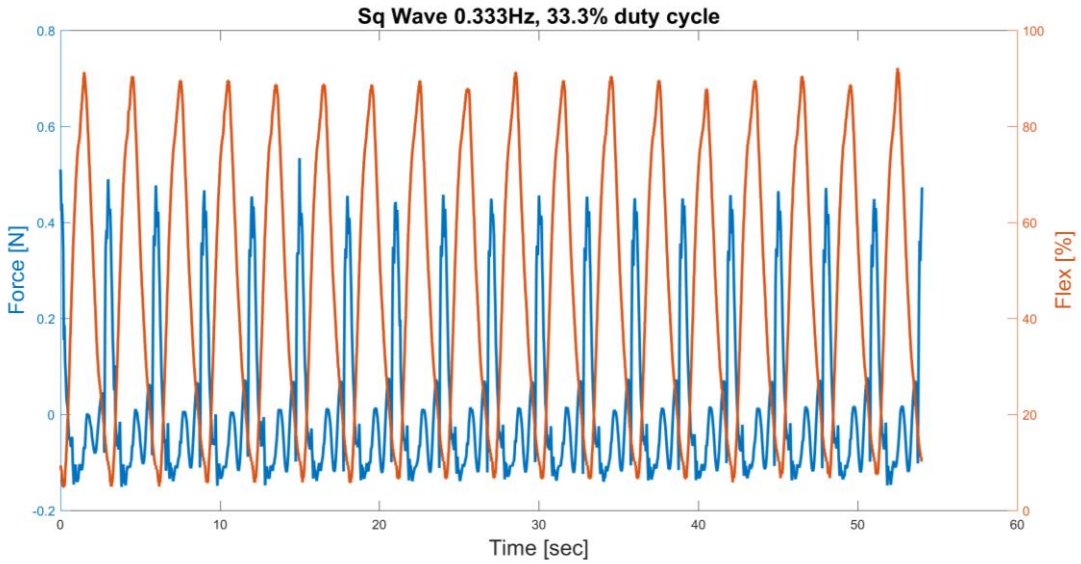

Fig. 70. Set 1 flex and force data for square wave 0.333Hz, 33.3% duty cycle.

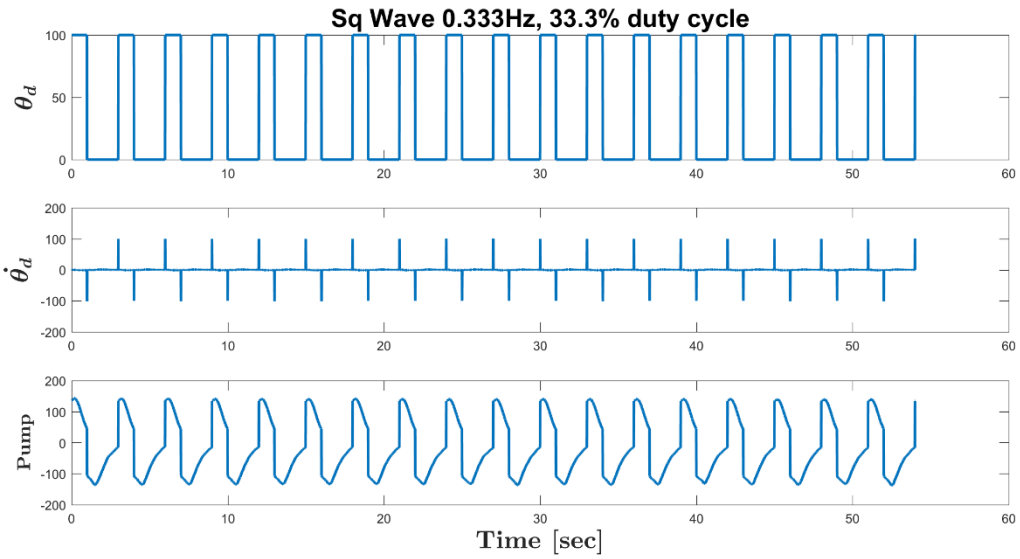

Fig. 71. Set 1 controller data for square wave 0.333Hz, 33.3% duty cycle.

Fig. 72 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 1 offset square wave 0.333Hz, 33.3% duty cycle. Fig. 73 contains the controller data (associated with pump 1) that was collected via serial during testing.

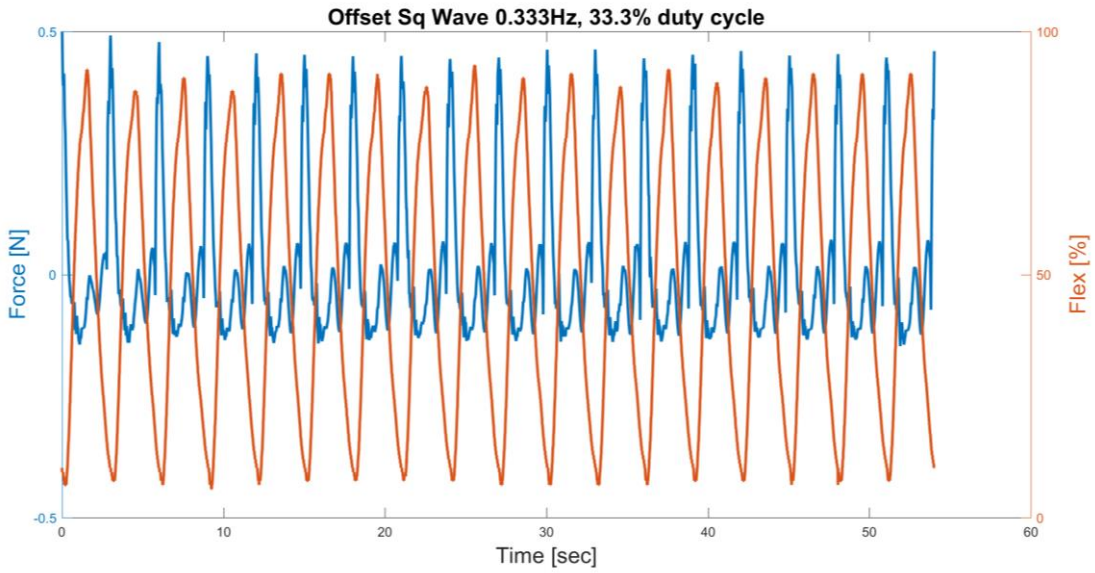

Fig. 72. Set 1 flex and force data for offset square wave 0.333Hz, 33.3% duty cycle.

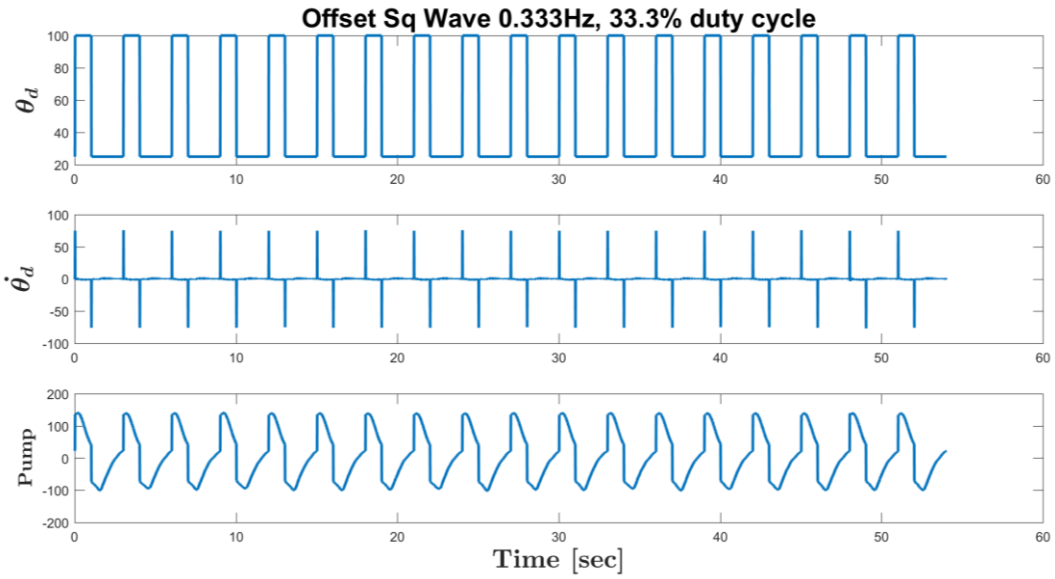

Fig. 73. Set 1 controller data for offset square wave 0.333Hz, 33.3% duty cycle.

Fig. 74 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 1 offset square wave 0.435Hz, 39.1% duty cycle. Fig. 75 contains the controller data (associated with pump 1) that was collected via serial during testing.

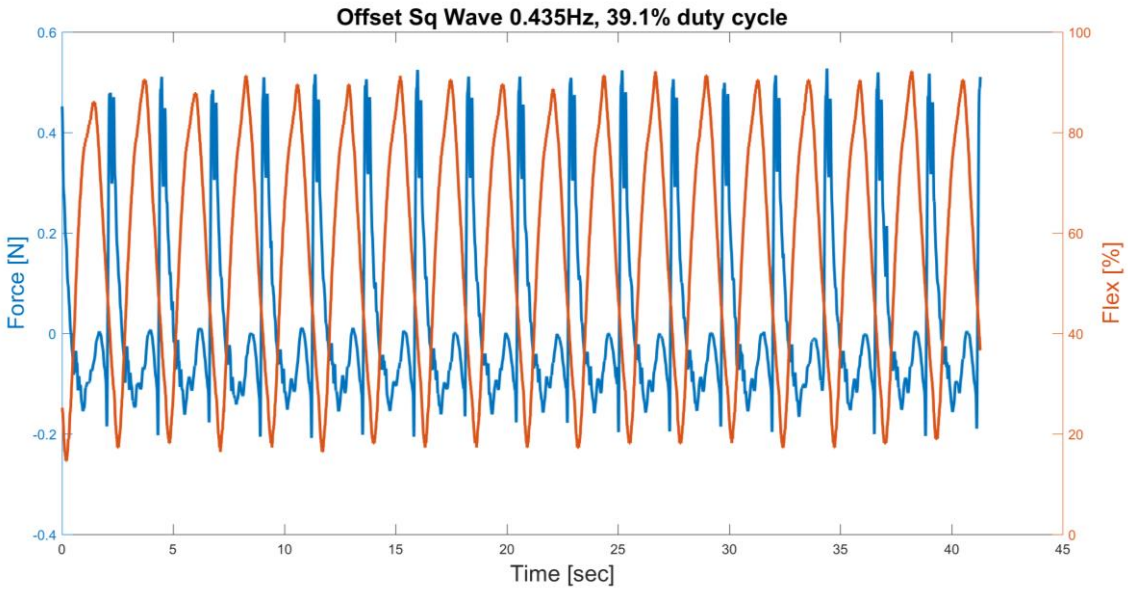

Fig. 74. Set 1 flex and force data for offset square wave 0.435Hz, 39.1% duty cycle.

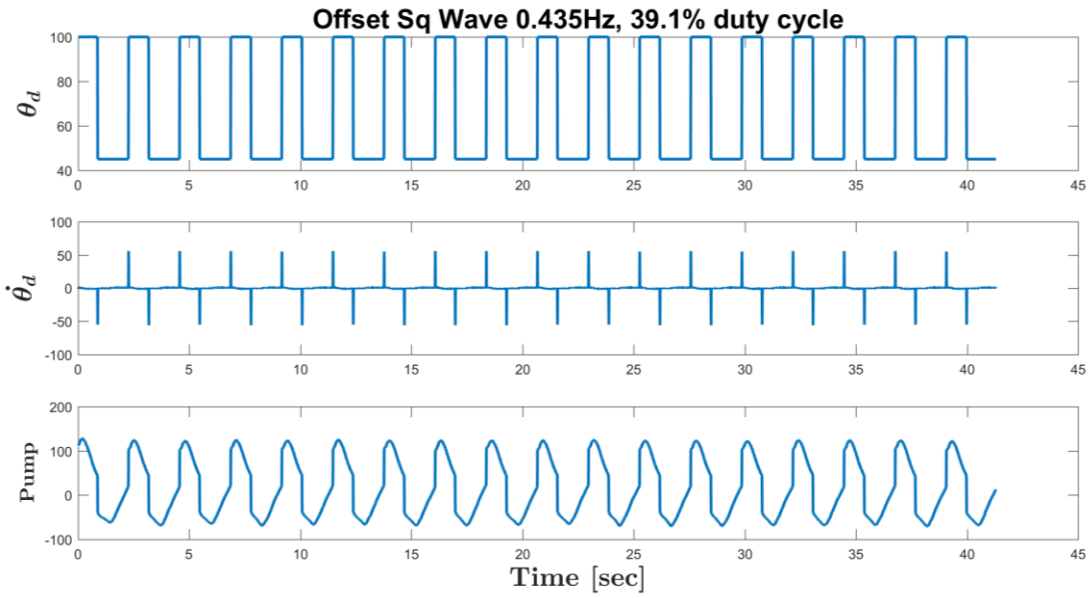

Fig. 75. Set 1 controller data for offset square wave 0.435Hz, 39.1% duty cycle.

9.5.2 Data Set 2

The following plots and tables pertain to the full 18 cycles of JenniFish Revision #2 in-line load cell data set 2. Fig. 76 contains the second set of load cell measurements at a frequency of 0.435Hz with a 39.1% duty cycle. Load cell drift for set 2 was 2.70mN.

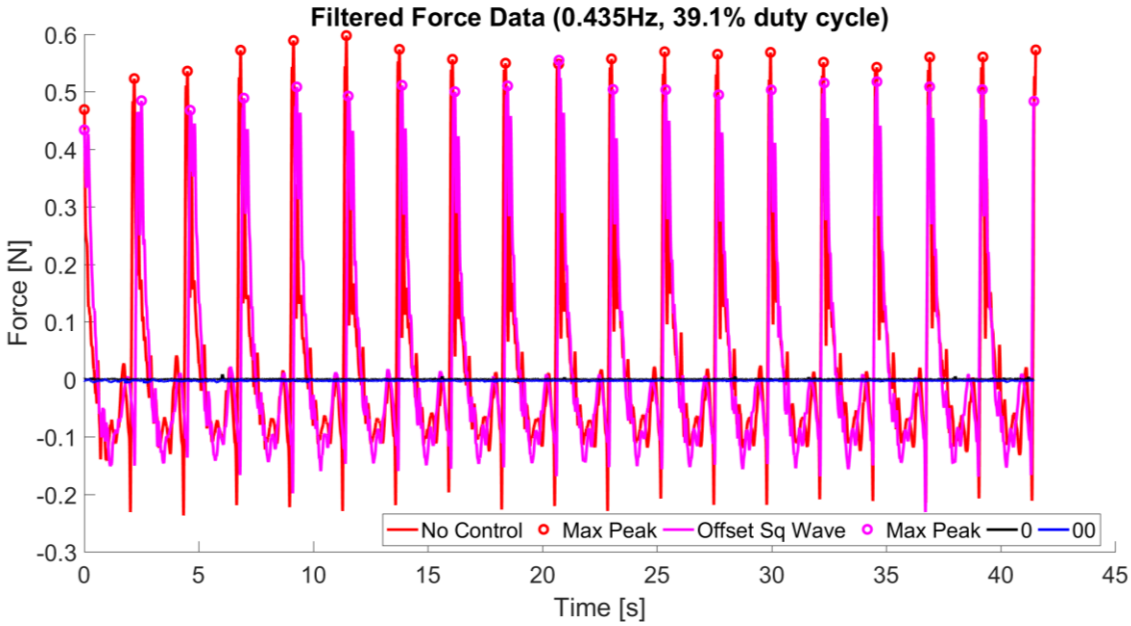

Fig. 76. Set 2 filtered force data (0.435Hz, 39.1% duty cycle).

Fig. 77 contains the second set of load cell measurements at a frequency of 0.333Hz with a 33.3% duty cycle.

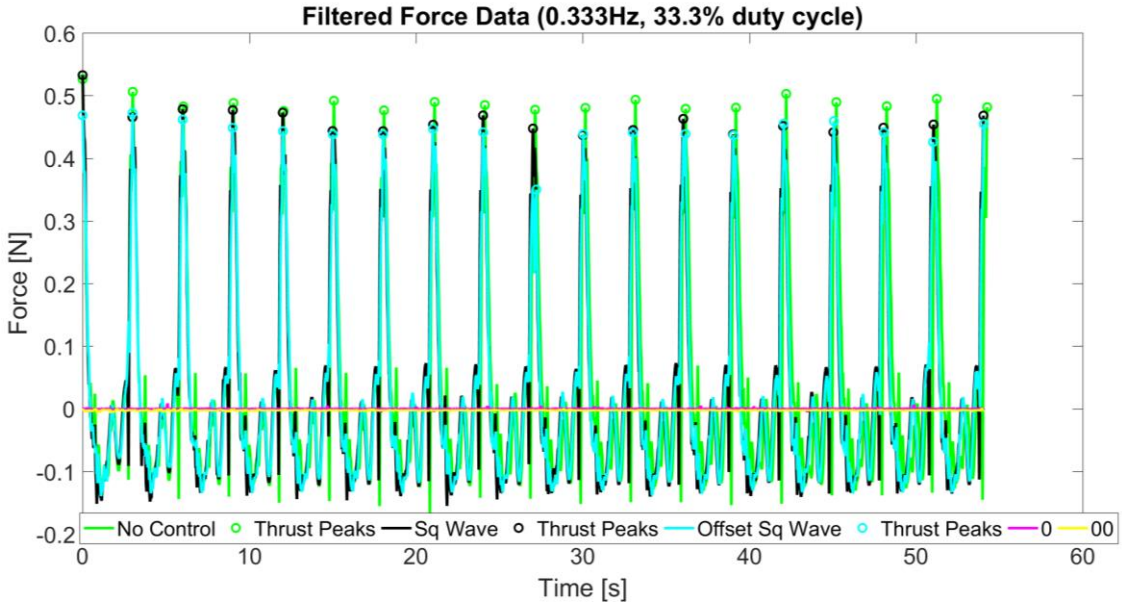

Fig. 77. Set 2 filtered force data (0.333Hz, 33.3% duty cycle).

Table 24 and Table 25 summarize the ANOVA2 results from MATLAB done for set 2 max thrust per cycle and net thrust per cycle respectively.

Table 24: Set 2 max thrust per cycle ANOVA2 results.

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.0123
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.0125
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.0598
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.011	0.0525

Table 25: Set 2 net thrust per cycle ANOVA2 results.

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0.9644	0.9677
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.2181
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0316	0.5678
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.2713	0.3952

Fig. 78 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 2 no control 0.333Hz, 33.3% duty cycle.

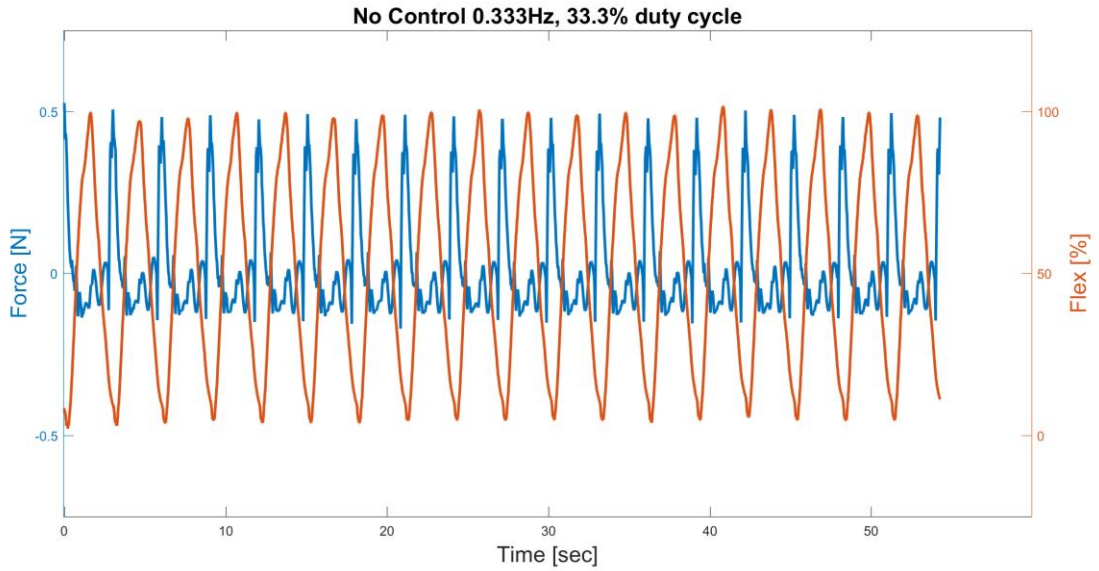

Fig. 78. Set 2 flex and force data for no control 0.333Hz, 33.3% duty cycle.

Fig. 79 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 2 no control 0.435Hz, 39.1% duty cycle.

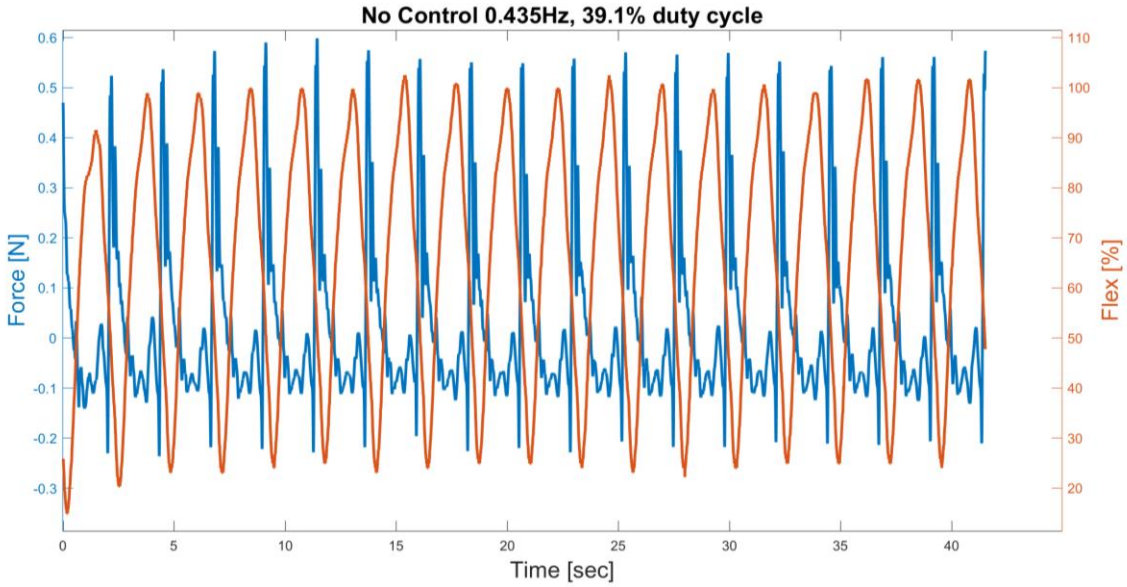

Fig. 79. Set 2 force and flex data no control 0.435Hz, 39.1% duty cycle.

Fig. 80 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 2 square wave 0.333Hz, 33.3% duty cycle. Fig. 81 contains the controller data (associated with pump 1) that was collected via serial during testing.

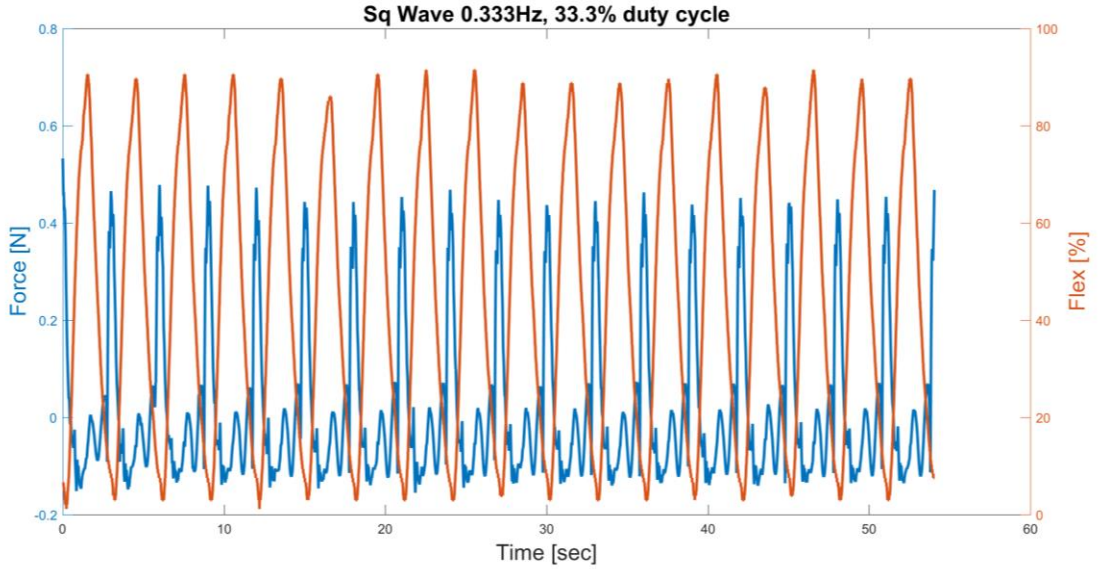

Fig. 80. Set 2 force and flex data for square wave 0.333Hz, 33.3% duty cycle.

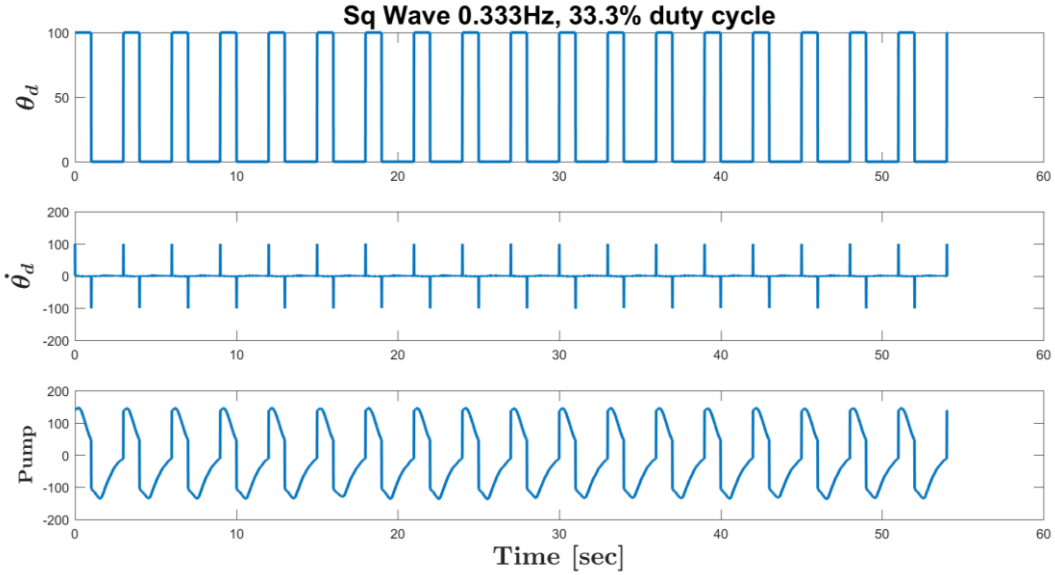

Fig. 81. Set 2 controller data for square wave 0.333Hz, 33.3% duty cycle.

Fig. 82 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 2 offset square wave 0.333Hz,

33.3% duty cycle. Fig. 83 contains the controller data (associated with pump 1) that was collected via serial during testing.

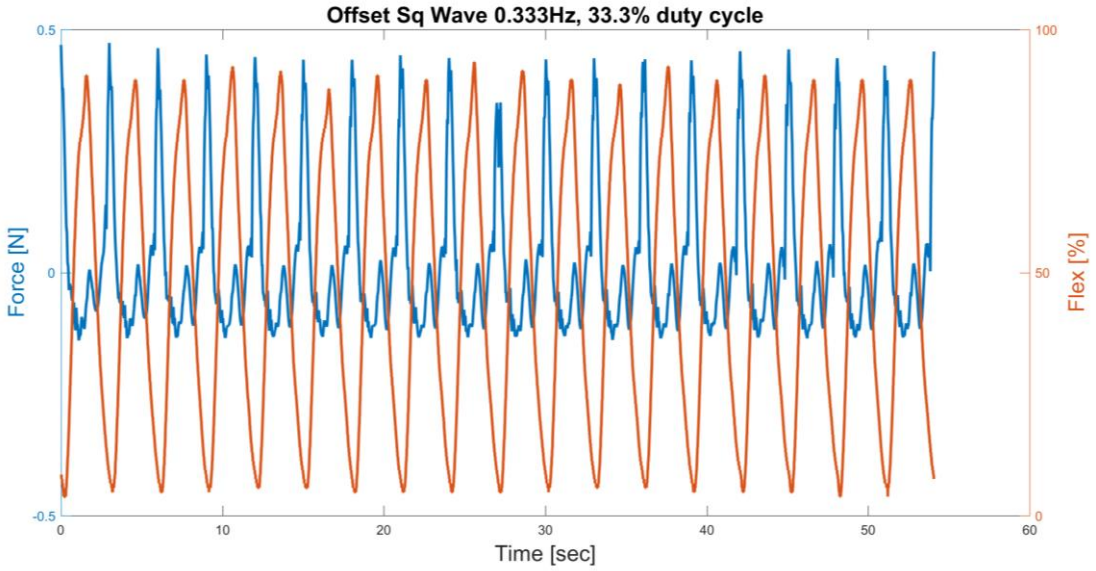

Fig. 82. Set 2 force and flex data for offset square wave 0.333Hz, 33.3% duty cycle.

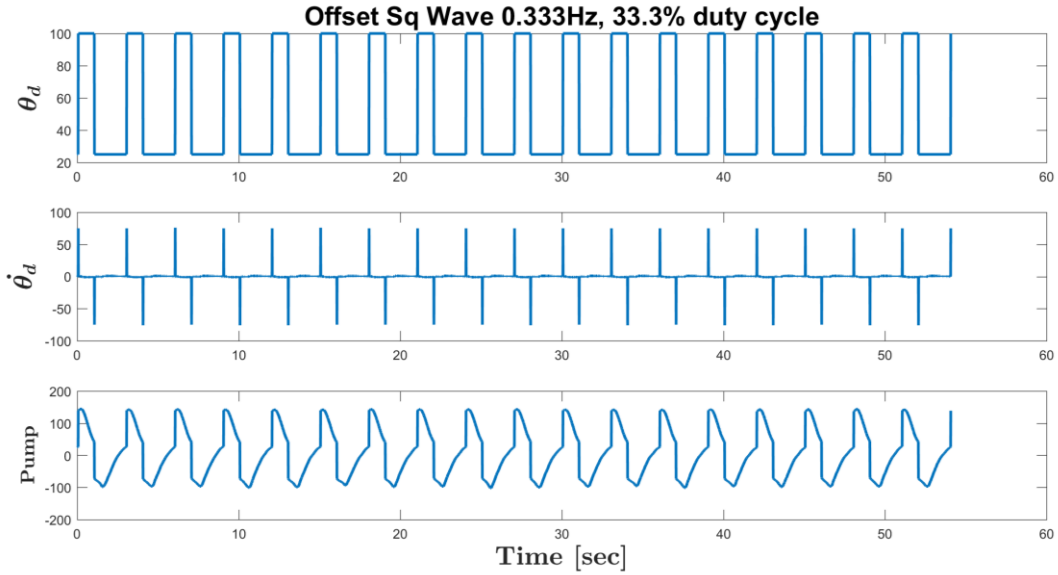

Fig. 83. Set 2 controller data for offset square wave 0.333Hz, 33.3% duty cycle.

Fig. 84 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 2 offset square wave 0.435Hz, 39.1% duty cycle. Fig. 85 contains the controller data (associated with pump 1) that was collected via serial during testing.

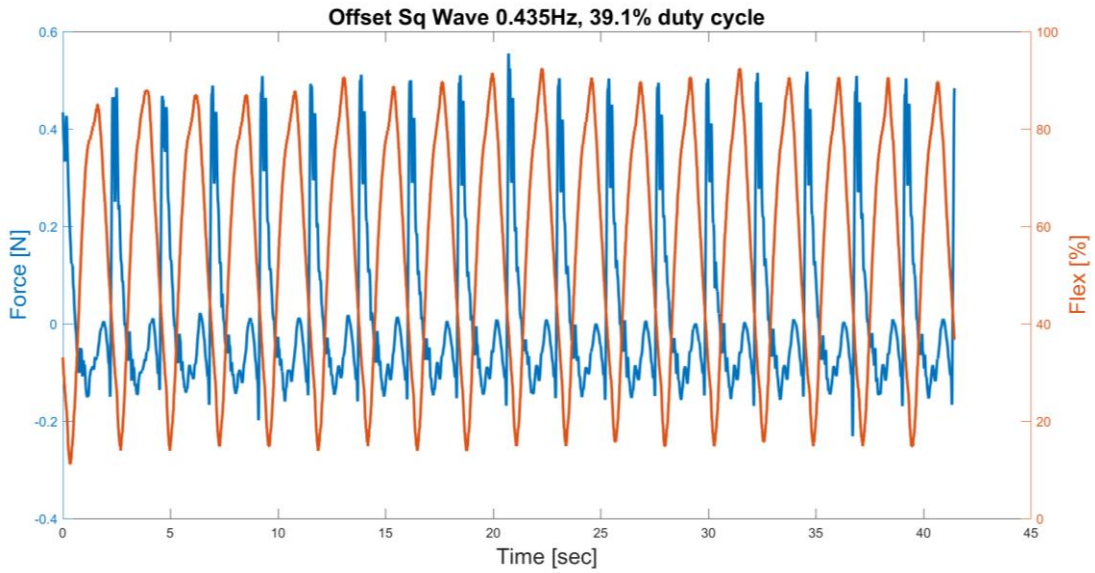

Fig. 84. Set 2 force and flex data for offset square wave 0.435Hz, 39.1% duty cycle.

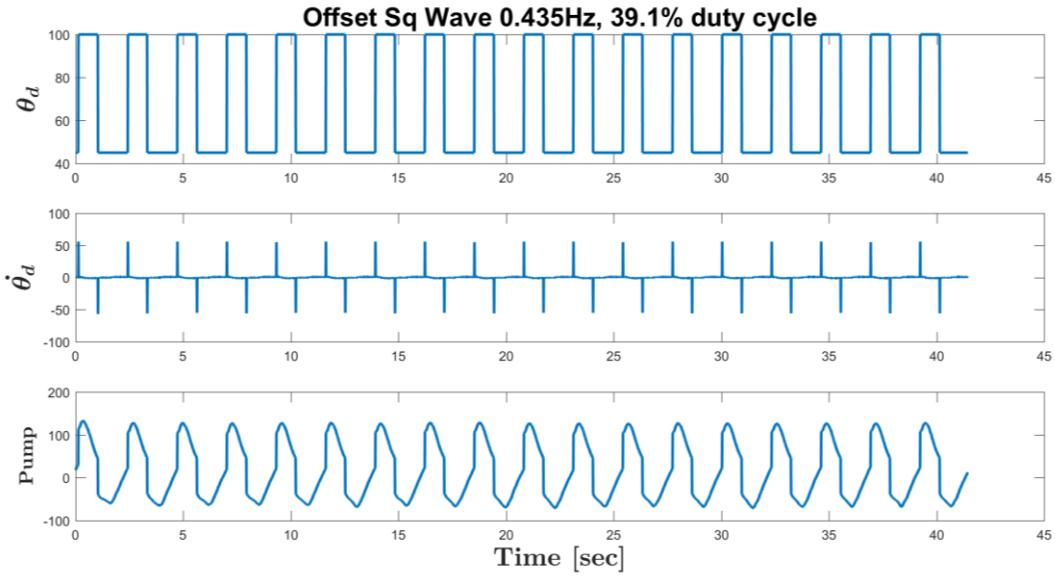

Fig. 85. Set 2 controller data for offset square wave 0.435Hz, 39.1% duty cycle.

9.5.3 Data Set 3

The following plots and tables pertain to the full 18 cycles of JenniFish Revision #2 in-line load cell data set 3. Fig. 86 contains the third and final set of load cell measurements at a frequency of 0.435Hz with a 39.1% duty cycle. Load cell drift for set 3 was 1.40mN.

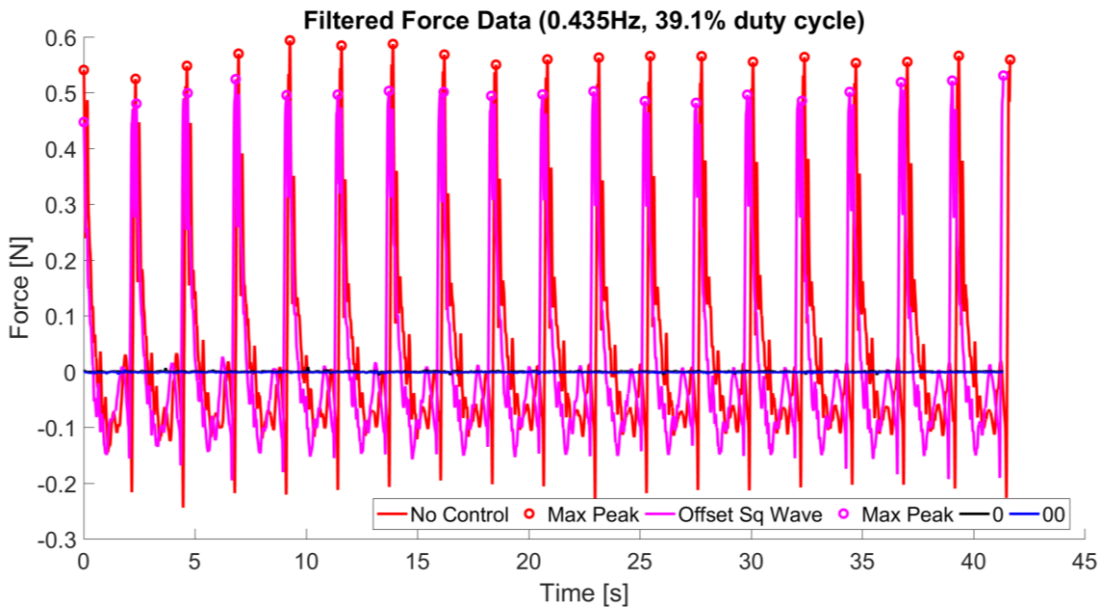

Fig. 86. Set 3 filtered force data (0.435Hz with a 39.1% duty cycle).

Fig. 87 contains the third set of load cell measurements at a frequency of 0.333Hz with a 33.3% duty cycle.

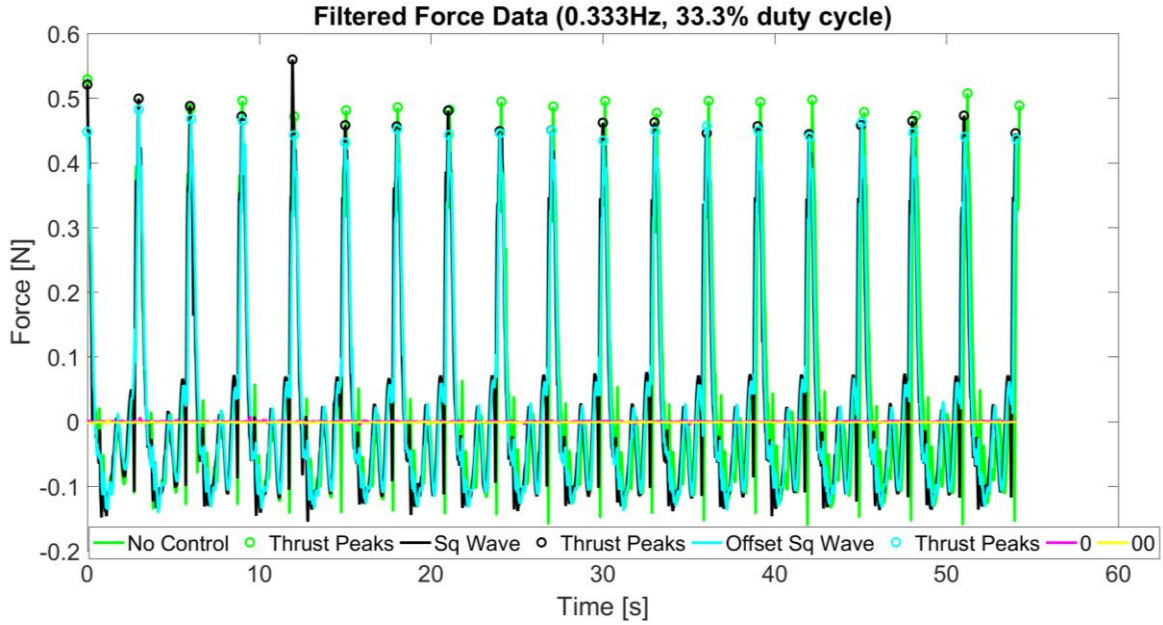

Fig. 87. Set 3 filtered force data (0.333Hz with a 33.3% duty cycle).

Table 26 and Table 27 summarize the ANOVA2 results from MATLAB done for set 3 max thrust per cycle and net thrust per cycle respectively.

Table 26. Set 3 max thrust per cycle ANOVA2 results.

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0	0.0883
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0.0136	0.4008
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0	0.4008
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0063	0.3076

Table 27. Set 3 net thrust per cycle ANOVA2 results.

Case Comparison	Prob>F Rows	Prob>F Columns
No Control vs. Offset Sq Wave (0.435Hz, 39.1% duty cycle)	0.2859	0.5578
No Control vs. Sq Wave (0.333Hz, 33.3% duty cycle)	0.0959	0.7013
No Control vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.0038	0.9567
Sq Wave vs. Offset Sq Wave (0.333Hz, 33.3% duty cycle)	0.521	0.7678

Fig. 88 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 3 no control 0.333Hz, 33.3% duty cycle.

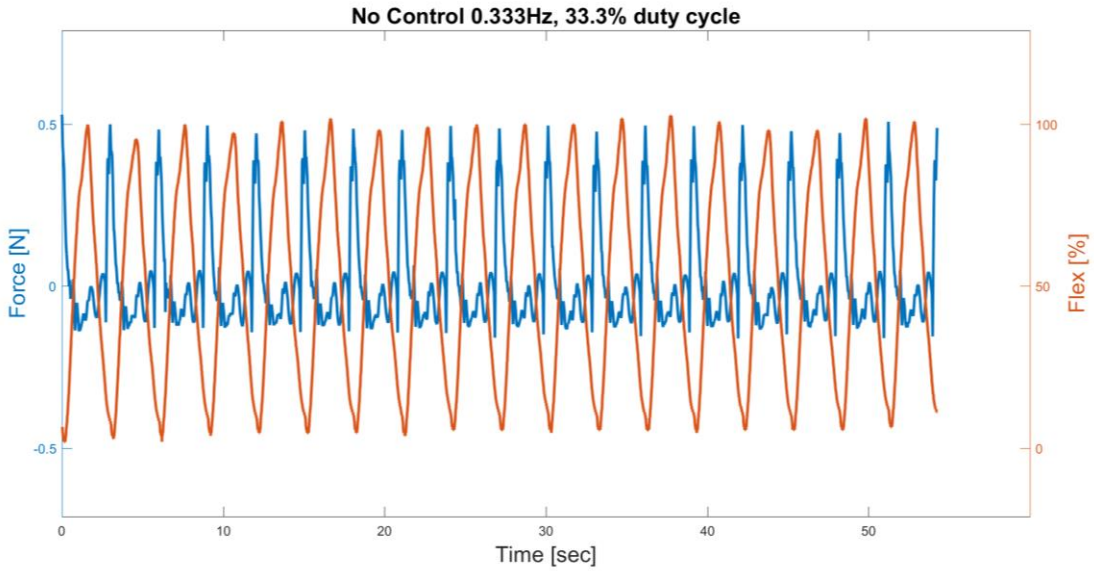

Fig. 88. Set 3 force and flex data for no control 0.333Hz, 33.3% duty cycle.

Fig. 89 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 3 no control 0.435Hz, 39.1% duty cycle.

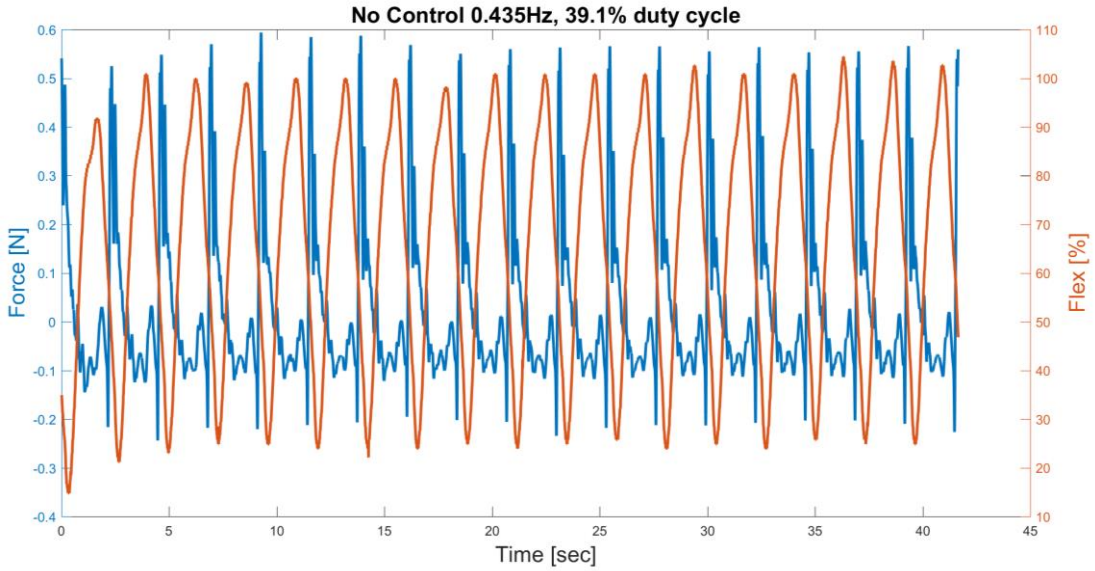

Fig. 89. Set 3 force and flex data for no control 0.435Hz, 39.1% duty cycle.

Fig. 90 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 3 square wave 0.333Hz, 33.3% duty cycle. Fig. 91 contains the controller data (associated with pump 1) that was collected via serial during testing.

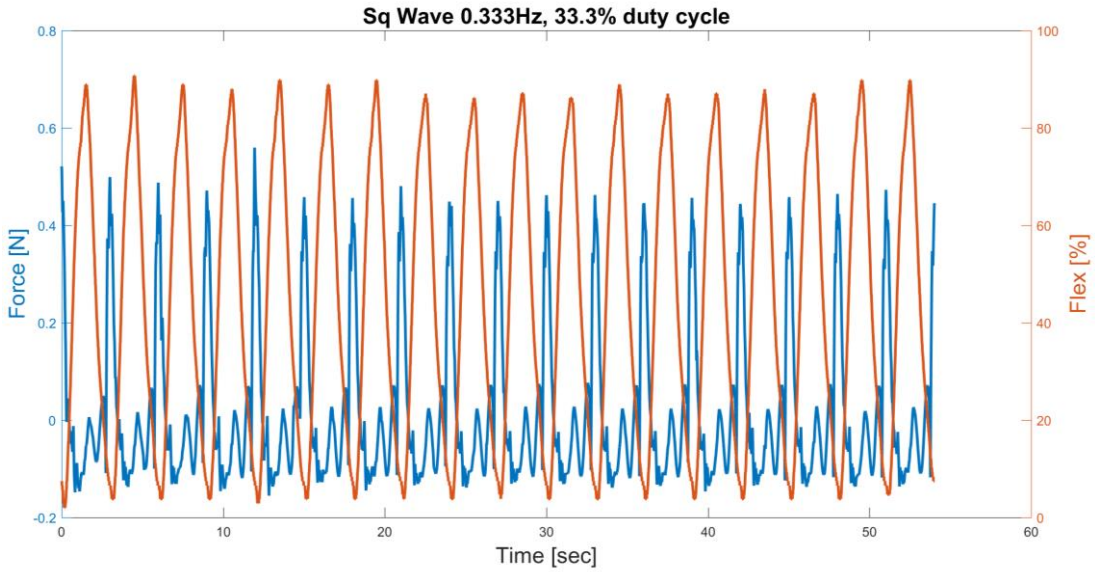

Fig. 90. Set 3 force and flex data for square wave 0.333Hz, 33.3% duty cycle.

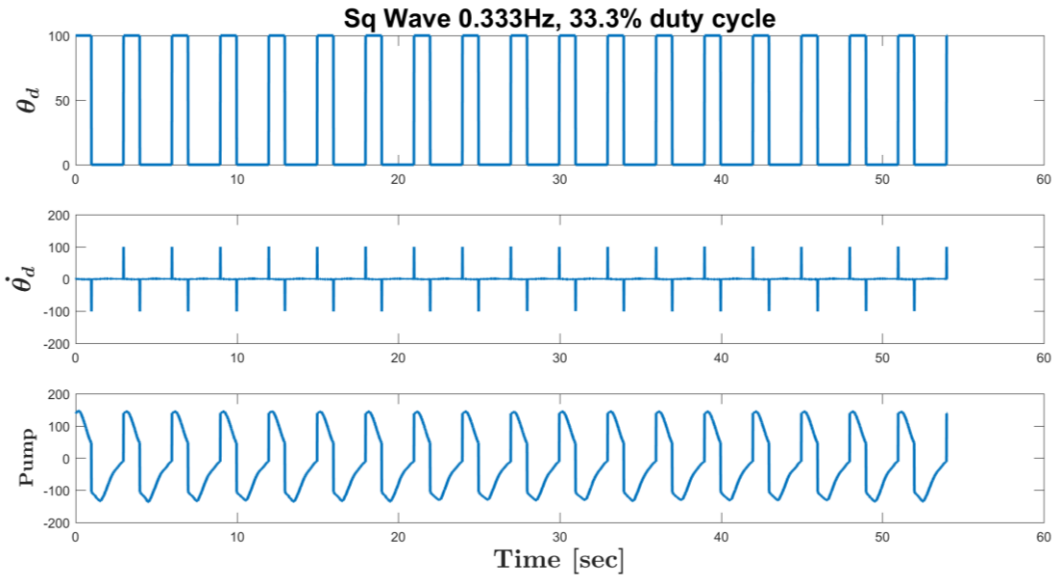

Fig. 91. Set 3 controller data for square wave 0.333Hz, 33.3% duty cycle.

Fig. 92 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 3 offset square wave 0.333Hz, 33.3% duty cycle. Fig. 93 contains the controller data (associated with pump 1) that was collected via serial during testing.

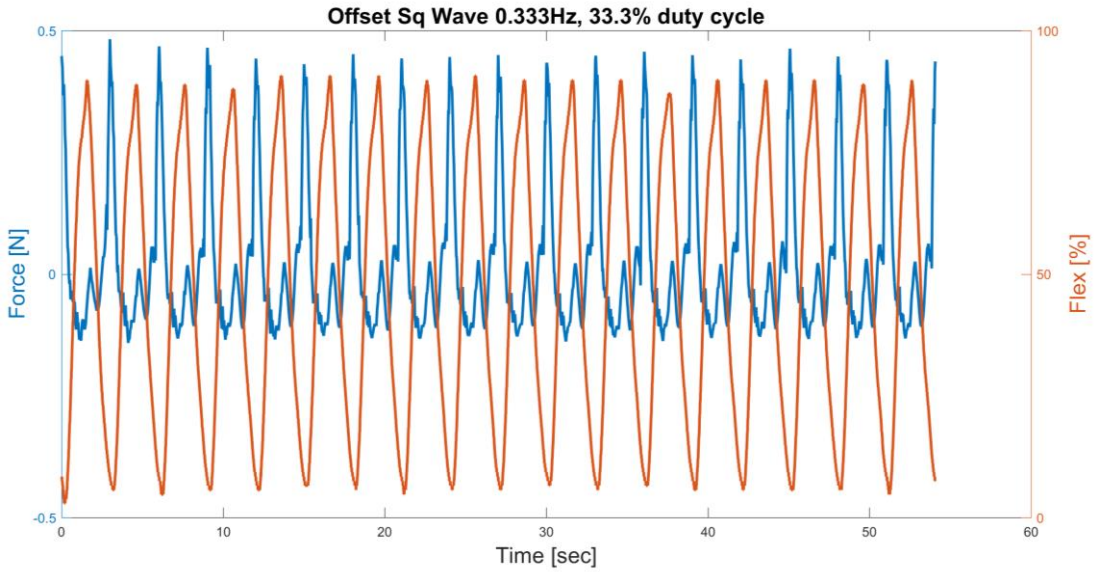

Fig. 92: Set 3 force and flex data for offset square wave 0.333Hz, 33.3% duty cycle.

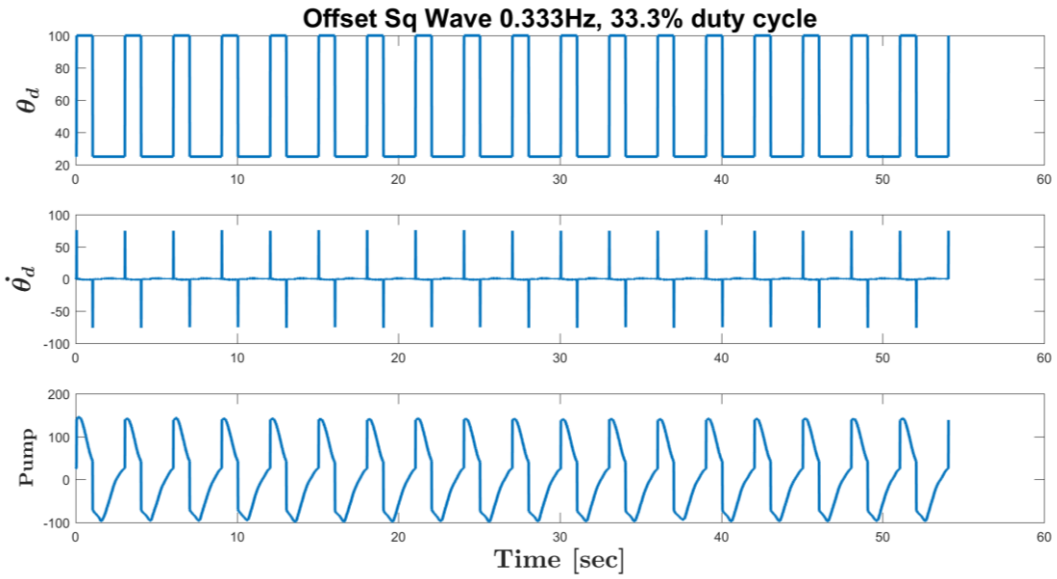

Fig. 93. Set 3 controller data for offset square wave 0.333Hz, 33.3% duty cycle.

Fig. 94 contains load cell data plotted with the mapped flex sensor data (associated with pump 1) that was collected via serial during testing for set 3 offset square wave 0.435Hz, 39.1% duty cycle. Fig. 95 contains the controller data (associated with pump 1) that was collected via serial during testing. Table 28 contains the normalized average force and sensor drift per case for all three in-line load cell test sets.

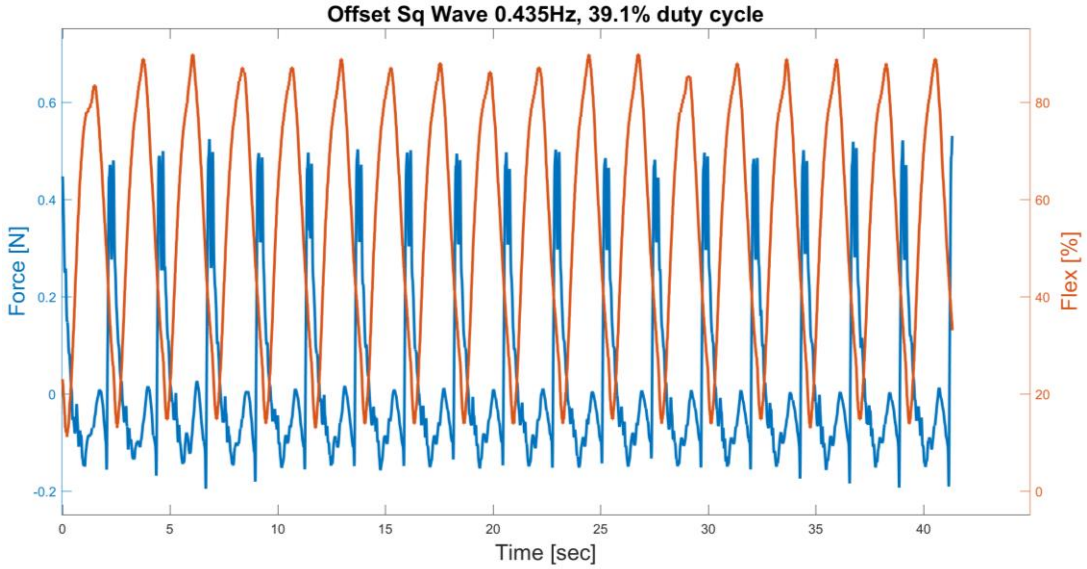

Fig. 94. Set 3 force and flex data for offset square wave 0.435Hz, 39.1% duty cycle.

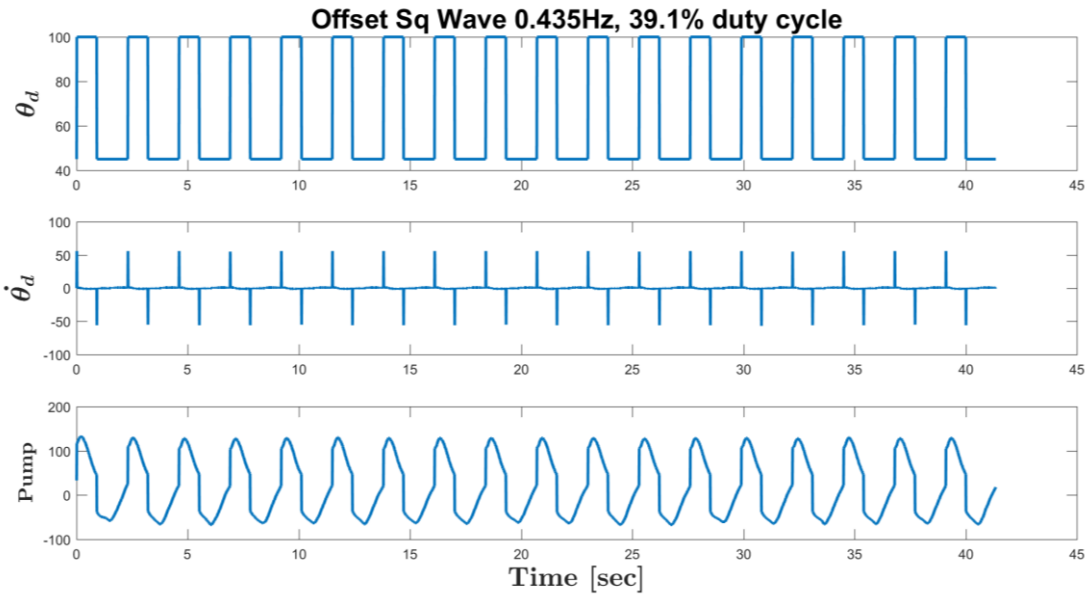

Fig. 95: Set 3 controller data for offset square wave 0.435Hz, 39.1% duty cycle.

Table 28: Summary of normalized average force per test case.

Data Set	No Control 0.333Hz, 33.1%	No Control 0.435Hz, 39.1%	Square Wave 0.333Hz, 33.3%	Offset Sq Wave 0.333Hz, 33.3%	Offset Sq Wave 0.435Hz, 39.1%	Drift
1	0.0218	0.0255	0.0194	0.0179	0.0205	0.0035
2	0.0220	0.0214	0.0191	0.0167	0.0216	0.0027
3	0.0224	0.0257	0.0205	0.0197	0.0222	0.0014

9.6 FCRAR Photo Permission

Jen Frame <jframe1@fau.edu>

FCRAR 2016 Paper - Frame

Sabri Tosunoglu <tosun@fiu.edu> Thu, Jun 23, 2016 at 7:45 PM
To: "<jframe1@my.fau.edu>" <jframe1@my.fau.edu>

Hello Jen,

You can include photos and any other content provided that you include the reference paper in your reference list and refer to the paper in your manuscript or figure titles. This will enhance your thesis since it shows acceptance to the conference and presentation to general public.

Regards,
Sabri Tosunoglu

Sent from my iPhone

On Jun 22, 2016, at 3:54 AM, Jennifer Frame <jframe1@my.fau.edu> wrote:

> Hi Sabri,
> I am submitting my thesis next month and might include one or two of the photos that were in my FCRAR submission. Do these proceedings have any copyright restrictions or are they public domain? If there are copyright restrictions, will you please provide me with a permission to reproduce letter?
> Thank you,
> Jen

10 REFERENCES

[1] C. Majidi, "Soft Robotics: A Perspective--Current Trends and Prospects for the Future," *Soft Robotics,* vol. 1, no. 1, pp. 5-11, 2014.

[2] P. Polygerinos, Z. Wang, K. C. Galloway, R. J. Wood and C. J. Walsh, "Soft robotic glove for combined assistance and at-home rehabilitation," *Robotics and Autonomous Systems,* vol. 73, no. November, pp. 135-143, 2015.

[3] P. Panagiotis, S. Lyne, Z. Wang, L. F. Nicolini, B. Mosadegh, G. M. Whitesides and C. J. Walsh, "Towards a Soft Pneumatic Glove for Hand Rehabilitation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* , Tokyo, 2013.

[4] S. Kim, C. Laschi and B. Trimmer, "Soft robotics: a bioinspired evolution in robotics," *Trends in Biotechnology,* vol. 31, no. 5, pp. 287-294, 2013.

[5] R. F. Shepherd, F. Ilievski, W. Choi, S. A. Morin, A. A. Stokes, A. D. Mazzeo, X. Chen, M. Wang and G. M. Whitesides, "Multigait soft robot," *Proceedings of the National Academy of Sciences of the United States of America,* vol. 108, no. 51, p. 20400–20403, 2011.

[6] F. Ilievski, A. D. Mazzeo, R. F. Shepherd, X. Chen and G. M. Whitesides, "Soft Robotics for Chemists," *Angewandte Chemie,* vol. 50, no. 8, pp. 1890-1895, 2011.

[7] C.-P. Chou and B. Hannaford, "Measurement and Modeling of McKibben Pneumatic Artificial Muscles," *IEEE Transactions on Robotics and Automation,* pp. 90-102, 1996.

[8] M. T. Tolley and D. Rus, "Design, fabrication and control of soft robots," *Nature,* pp. 467-475, 2015.

[9] M. Seckin, N. Y. Turan and A. C. Seckin, "Comparison of Production Methods in Soft Robotics," in *2015 International Conference on Advances in Software, Control, and Mechanical Engineering,* Antalya, 2015.

[10] B. Mosadegh, P. Polygerinos, C. Keplinger, S. Wennstedt, R. F. Shepherd, U. Gupta, J. Shim, K. Bertoldi, C. J. Walsh and G. M. Whitesides, "Pneumatic Networks for Soft Robotics that Actuate Rapidly," *Advanced Functional Materials,* vol. 24, no. 15, pp. 2163-2170, 2014.

[11] K. Suzumori, S. Endo, T. Kanda, N. Kato and H. Suzuki, "A Bending Pneumatic Rubber Actuator Realizing Soft-bodied Manta Swimming Robot," in *IEEE International Conference on Robotics and Automation,* Roma, Italy, 2007.

[12] M. Calisti, M. Giorelli, G. Levy, B. Mazzolai, B. Hochner, C. Laschi and P. Dario, "An octopus-bioinspired solution to movement and manipulation for soft robots," *Bioinspiration & Biomimetics,* p. 10, 2011.

[13] H.-J. Kim, S.-H. Song and S.-H. Ahn, "A turtle-like swimming robot using a smart soft composite (SSC) structure," *Smart Materials and Structures,* p. 11, 2013.

[14] A. D. Marchese, C. D. Onal and D. Rus, "Autonomous Soft Robotic Fish Capable of Escape Maneuvers Using Fluidic Elastomer Actuators," *Soft Robotics,* pp. 75-87, 2014.

[15] M. J. McHenry and J. Jed, "The ontogenetic scaling of hydrodynamics and swimming performance in jellyfish (Aurelia aurita)," *The Journal of Experimental Biology,* vol. 12, no. 206, pp. 4125-4137, 2003.

[16] Colin, S. P. Colin and J. H. Costello, "Morphology, swimming performance and propulsive mode of six co-occurring hydromedusae," *The Journal of Experimental Biology,* pp. 427-437, 2002.

[17] B. J. Gemmell, J. H. Costello, S. P. Colin, C. J. Stewart, J. O. Dabiri, D. Tafti and S. Priya, "Passive energy recapture in jellyfish contributes to propulsive advantage over other metazoans," *Proceedings of the National Academy of Sciences of the United States of America,* p. 6, 2013.

[18] R. Bale, M. Hao, A. p. S. Bhalla and N. A. Patankar, "Energy effiency and allometry of movement of swimming and flying animals," *Proceedings of the National Academy of Sciences,* vol. 111, no. 21, pp. 7517-7521, 2014.

[19] B. J. Gemmell, S. P. Colin, J. H. Costello and J. O. Dabiri, "Suction-based propulsion as a basis for efficient animal swimming," *Nature Communications,* vol. 6, no. November, pp. 1-8, 2015.

[20] A. Villanueva, C. Smith and S. Priya, "A biomimetic robotic jellyfish (Robojelly) actuated by shape memory alloy composite actuators," *Bioinspiration & Biomimetics,* vol. 6, no. 3, pp. 1-16, 2011.

[21] A. Villanueva, S. Bresser, S. Chung, Y. Tadesse and S. Priya, "Jellyfish inspired unmanned underwater vehicle," in *Proc. SPIE 7287*, San Diego, 2009.

[22] S. Guo, X. Ye and Y. Yang, "A New Type of Jellyfish-Like Microrobot," in *International Conference on Integration Technology*, Shenzhen, 2007.

[23] Y. Tadesse, A. Villanueva, C. Haines, D. Novitski, R. Baughman and S. Priya, "Hydrogen-fuel-powered bell segments of biomimetic jellyfish," *Smart Materials and Structures*, vol. 21, no. 4, pp. 1-17, 2012.

[24] J. Najem, S. A. Sarles, B. Akle and D. J. Leo, "Biomimetic jellyfish-inspired underwater vehicle actuated by ionic polymer metal composite actuators," *Smart Materials and Structures,* vol. 21, no. 9, pp. 1-11, 2012.

[25] S.-W. Yeom and I.-K. Oh, "A biomimetic jellyfish robot based on ionic polymer metal composite actuators," *Smart Materials and Structures,* p. 10, 2009.

[26] Y. Tadesse, J. Brennan, C. Smith, T. E. Long and S. Priya, "Synthesis and Characterization of Polypyrrole Composite Actuator for Jellyfish Unmanned Undersea Vehicle," in *Proc. SPIE 7642*, San Deigo, 2010.

[27] J. C. Nawroth, H. Lee, A. W. Feinberg, C. M. Ripplinger, M. L. McCain, A. Grosberg, J. O. Dabiri and K. K. Parker, "A tissue-engineered jellyfish with biomimetic propulsion," *Nature Biotechnology,* vol. 30, no. 8, pp. 792-800, 2012.

[28] Y. Ko, S. Na, Y. Lee, K. Cha, S. Young Ko, J. Park and S. Park, "A jellyfish-like swimming mini-robot actuated by an electromagnetic actuation system," *Smart Materials and Structures,* vol. 21, no. 5, pp. 1-9, 2012.

[29] S. Nir, S. Shoval, B. B. Moshe, I. Ruchaevski and T. Shteinberg, "A Jellyfish-like Robot for mimicking jet propulsion," in *IEEE 27th Convention of Electrical and Electronics Engineers*, Israel, 2012.

[30] K. Marut, C. Stewart, T. Michael, A. Villanueva and S. Priya, "A jellyfish-inspired jet propulsion robot actuated by an iris mechanism," *Smart Materials and Structures*, p. 12, 2013.

[31] A. Villaneuva, S. Priya, C. Anna and C. Smith, "Robojelly bell kinematics and resistance feedback control," in *IEEE Internation Conference on Robotics and Biomimetics (ROBIO)*, Tianjin, 2010.

[32] J. Frame, O. Curet and E. Engeberg, "Free-Swimming Soft Robotic Jellyfish," in *Florida Conference on Recent Advances in Robotics*, Miami, 2016.

[33] G. Hamilton, "The Secret Lives of Jellyfish," *Nature*, vol. 531, no. March, pp. 432-434, 2016.