A JELLYFISH-BASED AQUATIC LOCOMOTOR WITH TUNABLE GAITS

By

LILLIAN WANG

A thesis submitted to the

Graduate School- New Brunswick

Rutgers, The State University of New Brunswick

In partial fulfillment of the requirements

For the degree of

Master of Science

Graduate Program in Mechanical and Aerospace Engineering

Written under the direction of

Aaron Mazzeo

And approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October 2017

ABSTRACT OF THE THESIS

A JELLYFISH-BASED AQUATIC LOCOMOTOR WITH TUNABLE GAITS

By LILLIAN WANG

Thesis Director:

Aaron Mazzeo

The main objective of this thesis is to characterize the fluid flow around a jellyfish-inspired locomotor and evaluate its efficiency by measuring the cost of transport. This thesis describes the design and fabrication of a soft locomotor inspired by the swimming patterns seen in nature by the moon jellyfish. The locomotor used the linear actuation of a syringe to pull on its soft bell to emulate the contraction of muscles in a jellyfish. The locomotor swam with various gaits categorized into three paces of actuation: fast, moderate, and slow. By performing particle image velocimetry and collecting vertical displacement data through acquired video, it was possible to characterize the motion of the locomotor and surrounding fluid. The flow pattern for the moderate gait most closely followed the flow pattern of living, oblate jellyfish. The displacement and velocity profiles showed distinct regimes of contraction and relaxation as seen in living jellyfish. At moderate and slow gaits, there was a third regime of post-relaxation depicted by an additional acceleration phase. The calculated average costs of transport for various gaits indicate there is a dependence on swimming speed. In general,

these results suggest that jellyfish might be capable of adjusting their propulsive efficiency by varying their gait, but that they have evolved to swim at a gait for maximum feeding efficiency rather than propulsive efficiency.

ACKNOWLEDGEMENTS

I am deeply grateful to my advisor Professor Mazzeo for his encouragement, motivation, and guidance on bio-inspired jellyfish locomotion.

In addition, I would like to express my most sincere gratitude to Ke Yang. It is an honor for me to continue his work on jellyfish-inspired robotics. I would like to thank the many graduate and undergraduate students for their support and inspirations. It is my pleasure to acknowledge Saleh Tahir, Ajay Dusane, Hillel Hochsztein, and Denny Zheng for all their time spent on assisting with experiments and analyzing the collected data. I would also like to acknowledge Professor Yi and Kaiyan Yu for their help with the PID controller.

Lastly, I would like to thank the reviewers of this thesis Professor Denda and Professor Zou.

DEDICATION

This thesis is dedicated to my beloved parents and brother.

TABLE OF CONTENTS

LIST OF FIGURES

1. Introduction

The study of locomotion in living organisms has led to the development of many bio-inspired robotic technologies. There have been efforts to understand and imitate flapping locomotion in air and water.[1],[2],[3],[4] Many bio-inspired robotic mechanisms have the potential to perform various engineered tasks, such as robotic fish fins that allow for fast acceleration and maneuverability of underwater vehicles,[3],[5] inflatable actuators that take advantage of peristalsis,[6] or even bladders that affect lift and drag on fins/airfoils.[7] Roboticists and engineers are working to improve these mechanisms to be more efficient and simpler than those observed in nature.

1.1 Medusae locomotion

The widely accepted method of locomotion for medusae is the basic principle of jet propulsion.[8] The muscles in the bell of a medusa contract and push water out from under its bell to propel itself forwards.[9] This generalization comes from many studies done on the swimming patterns of multiple species of medusae.[8],[9],[10],[11] Different types of medusae have swimming patterns dependent on their techniques used for survival. For some types, swimming and feeding occur simultaneously, while for others, feeding occurs separately.[8] There are two broad categories of medusae, prolate and oblate, defined by the shape of their bell. The size of the velum relative to the bell diameter determines the shape of the bell.[8],[10] The prolate species have a large velum relative to their diameter. Their muscles contract rapidly, and minimal fluid travels to the underside of the bell. The oblate species have a small velum and long pauses between contractions compared to the prolate species. When they contract, a large vortex, also called the starting vortex, forms at the margin of the bell. A large volume of water from around and

under the bell pushes downward. When they relax, the water rushes inwards and refills the bell, forming a large vortex (the stopping vortex) that is in the opposite direction of the starting vortex. The repetition of the contraction and relaxation of the bell results in a series of closely spaced vortices as the medusa swims forwards, seen in Fig. 1. The oblate species have higher propulsion efficiencies and feeding rates due to the increase in the volume of water under the bell during the relaxation time between contractions.[8]

According to Gemmell, et al.[12], the moon jellyfish (*Aurelia aurita)* may be the most efficient swimmers in the world because they demonstrate a phenomenon called passive energy recapture (PER). The moon jellyfish, categorized as oblate medusae, are a commonly studied type of jellyfish. Other researchers have carefully studied and documented their swimming and feeding patterns.[9],[10] During their contraction, also known as the power stroke, the bell diameter reduces, and a starting vortex forms at the margin of the bell. Pressure forces the water out from under the bell, thrusts the jellyfish forwards, and accelerates it to maximum velocity. The starting vortex sheds at the beginning of relaxation, or the recovery stroke, and merges with the stopping vortex forming under the bell. The decrease in pressure leads to water refilling under the bell and causes the jellyfish to move backward (showing negative velocity). However, the negative velocity is smaller in comparison to the positive velocity attained during the contraction, so the overall net movement is forwards.[10] PER occurs during the relaxation phase since the starting vortices partially merge with the stopping vortices under the bell before the next contraction, which produces a small positive velocity profile.

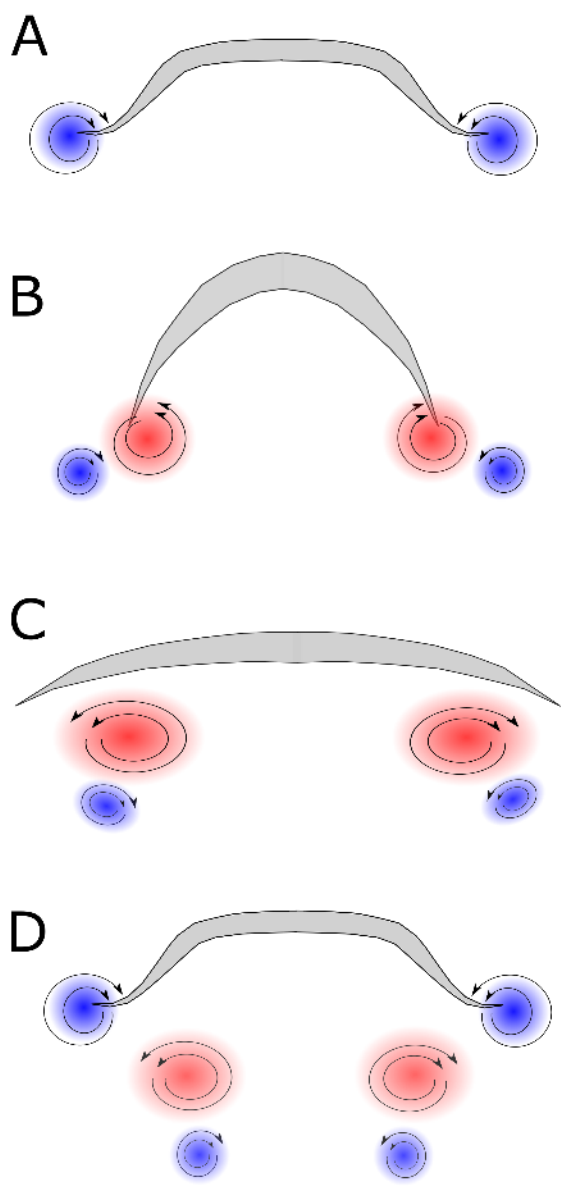


Figure 1. Schematic diagram depicting the vortices created during a swimming cycle of an oblate jellyfish. (a) During the power stroke, initial vortices form at the margin of the bell. (b) At the end of the contraction, the initial vortices shed off the bell and stopping vortices form under the margin of the bell. (c) During the recovery stroke, part of the initial vortices merge with and enhance the stopping vortices. The extra thrust demonstrates PER. (d) Initial vortices form again at the beginning of the next power stroke. A series of vortices closely follows the wake of the bell.

Based on models done on swimming and feeding, studies suggest the moon jellyfish might be capable of even higher propulsion efficiencies than those observed in living jellyfish.[10] However, they have developed to capture prey effectively for survival instead of maximizing propulsive efficiency.

1.1. Objective

We explore propulsion inspired by jellyfish locomotion. Several jellyfish-inspired robots developed by other research groups demonstrate the contraction and replicate the curvature of the bell of living jellyfish. Priya, et al.[13] and Najem,[14] built bio-inspired jellyfish vehicles that use smart actuators, such as shape memory alloy (SMA) and ionic polymer metal composites (IPMCs), to bend the bell during contraction. These vehicles imitate the geometry of the bell for the deformation cycle. However, they do not focus on the analysis of the vortices formed during a contraction cycle or demonstrate passive energy recapture.

Our main objective is to design a vehicle that replicates the propulsion and swimming pattern of an oblate jellyfish. We want to observe the phenomenon of passive energy recapture and improve the propulsive efficiency upon the moon jellyfish. In addition, we want to explore the hypothesis that the gaits and fluid flow associated with propulsion depend on the jellyfish's efforts to survive. We want to visualize the flow field produced by jellyfish-inspired flapping motion, which would further extend the understanding of vortex shedding of animals with fins, such as fish, jellyfish, eel, insects, and birds.[15] We hope to categorize the vortices associated with various lengths of contraction and determine the cost of transport.

To approach our problems, we designed and fabricated a soft locomotor inspired by the moon jellyfish. We characterized the motion by measuring vertical displacement, velocity, and pressure at various actuation rates. We also analyzed the fluid flow around the margin of the bell using particle image velocimetry (PIV). Additionally, we built an automated hydraulic control system to control the rate of actuation and the amount of water delivered to the locomotor.

2. Experimental Design and Set Up

The objective of the design for the soft locomotor was to replicate the shape of the bell and to mimic the basic movement of living oblate jellyfish. Other established soft robots inspired the design of the locomotor.[16] The locomotor consists of a soft bell structure, a syringe, nylon string, and tubing. The design of the bell structure in this thesis replicated the jellyfish-based robot presented in Ke Yang's thesis[15] with minor changes. The bell structure consisted of six fin-like sections, small parts that connected to the top of the fins, and thin membranes that connected to the sides of the fins. The syringe expanded in one direction linearly, and the string tied the end of the syringe to the underside of the bell of the locomotor. The linear actuation corresponded to the contraction and relaxation of the bell. The tubing delivered the water to the locomotor.

The locomotor had an automated hydraulic control system. Linear actuators pushed small syringes to deliver water to the large syringe attached to the underside of the underwater locomotor. A microcontroller controlled the position and speed of the linear actuators, while it also measured and collected the data using Simulink and MATLAB.

2.1. Fabrication of the Bell of the Jellyfish-Based Locomotor

We fabricated the completely soft bell structure from two types of silicone rubber, Mold Star® 30 and Ecoflex® 30. Mold Star 30 is a blue silicone rubber that is tear resistant and harder than Ecoflex 30, a white translucent silicone rubber. The material used for parts closer to the center of the bell was Mold Star 30. The center of the locomotor had a set initial curvature due to the stiff nature of the Mold Star 30. The harder, stronger rubber also prevented the string from tearing through the bell. The material used for parts at the outer edge of the bell was Ecoflex 30. The outer edge of the bell was extremely soft, which allowed the locomotor to move easily in the water and manipulate the surrounding water. We use soft lithography with 3D printed molds to make all the components of the bell structure. For both Mold Star 30 and Ecoflex 30, the liquid rubber consisted of two equally weighted parts and required degassing.

For each fin-like segment, we first made the top half of the fin in a partial mold with Mold Star 30. We cured it fully using an oven and cut small holes into the bottom edge to ensure better bonding between the two rubber segments. Then, we placed the first half of the fin into the full mold and poured Ecoflex 30 into the mold. The part made from Ecoflex 30 bonded firmly with the top half. We poured small amounts of Mold Star 30 into the grooves in the top half of the fin, clamped the fin half an inch from the tip, and hung it from a stand until it fully cured. The weight of the fin set its initial curvature. We glued the smaller parts that were closer to the center of the bell together with Sil-Poxy®, a silicone adhesive. We made the thin membrane by making sheets of Ecoflex 30 and carefully glued the membrane onto the fins using Sil-Poxy. Figure 2c, d shows the constructed soft bell structure of the locomotor.

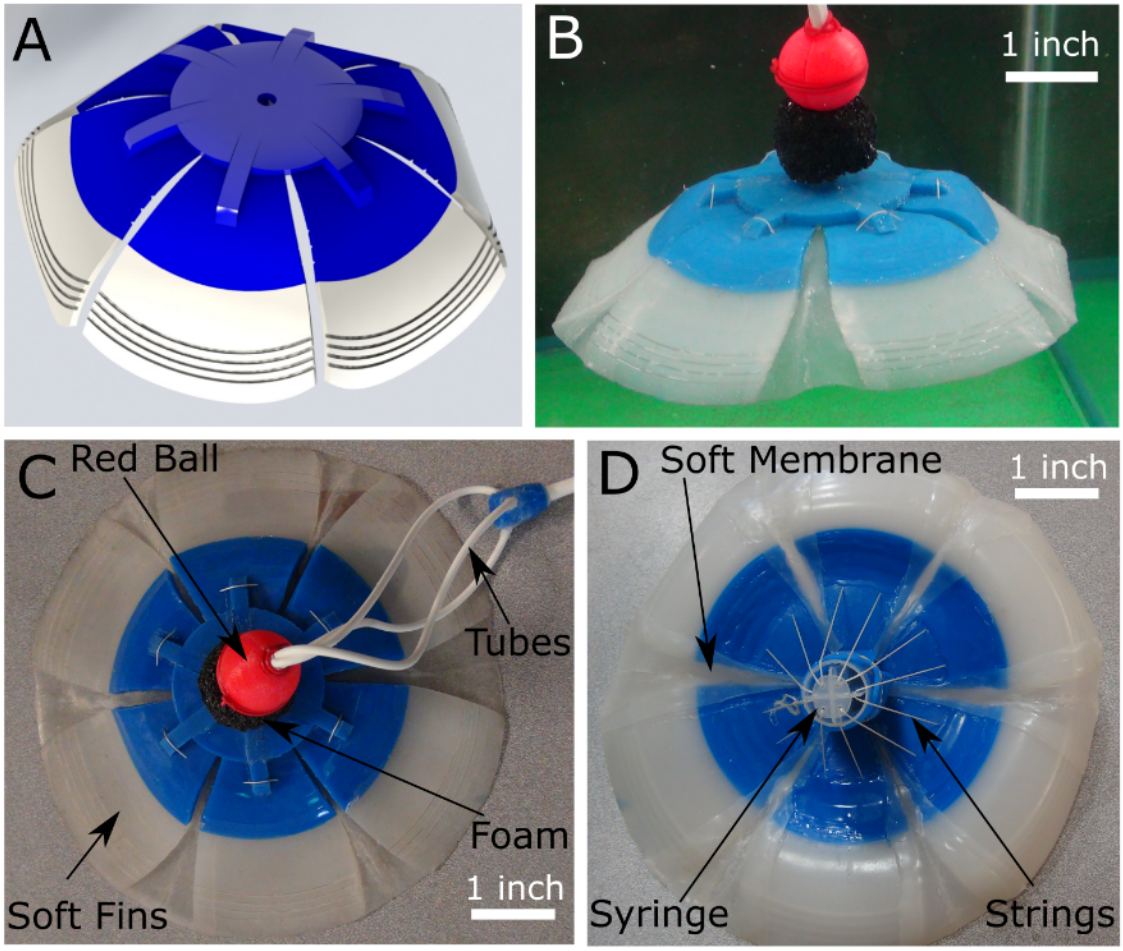


Figure 2. (a) Computer-drawn isometric view of the locomotor with ideal curvature for the fin-like segments. (b) Perspective view of the locomotor in the tank to demonstrate the curvature of the bell in water. (c) The top view of the locomotor with six fin-like segments, a foam piece that points the locomotor upwards, a red ball used for tracking, and tubes to tether the locomotor to the control system. (d) The bottom view of the locomotor showing the soft connecting membranes, the string, and part of a 30-mL syringe. The string connected the end of the syringe to the underside of the bell structure.

2.2. Actuation Mechanism for the Jellyfish-Based Locomotor

To mimic the muscle contraction in the bell of jellyfish, we designed a pulling mechanism, consisting of a syringe and nylon string. We used a saw to cut a 30-mL syringe at the 20-mL marker to decrease its length and drilled holes into the end of the syringe. We used one continuous nylon string to thread the end of the syringe through the underside of the bell for each fin-like segment. We ensured that the length of the string was the same for each section so that the bell had radial symmetry during contraction.

We used silicone tubing and lure lock fittings to connect the tubing to the syringe. To reduce pulling forces by the tubing, we decided to use three thinner tubes (1/8" OD) between the locomotor and the main (1/4" OD) tubing connected to the control system. The elasticity of the thinner tubes reduced any pulling forces that may have acted on the locomotor.[6] We glued the three thinner tubes to a slightly larger tube using Sil-Poxy to attach it to the lure lock fitting. On the other end of the three tubes, we made a small part, using Mold Star 30, with three small openings for the thinner tubes and a larger opening for the main tubing. The three-to-one part had a relatively large intersection to allow water to pass through at a sufficient flow rate.[15]

At the start of the contraction phase, the actuator started to fill, and the strings pulled the bell towards the center of the locomotor linearly (see Fig. 3a). The fins of the bell curved outwards. At the end of the contraction phase, the actuator filled entirely, and the bell curved slightly inwards (see Fig. 3b). During the relaxation phase, when the syringe was empty, the strings were fully relaxed, as they only acted in tension, and the bell shape reverted to its original form (see Fig. 3c).

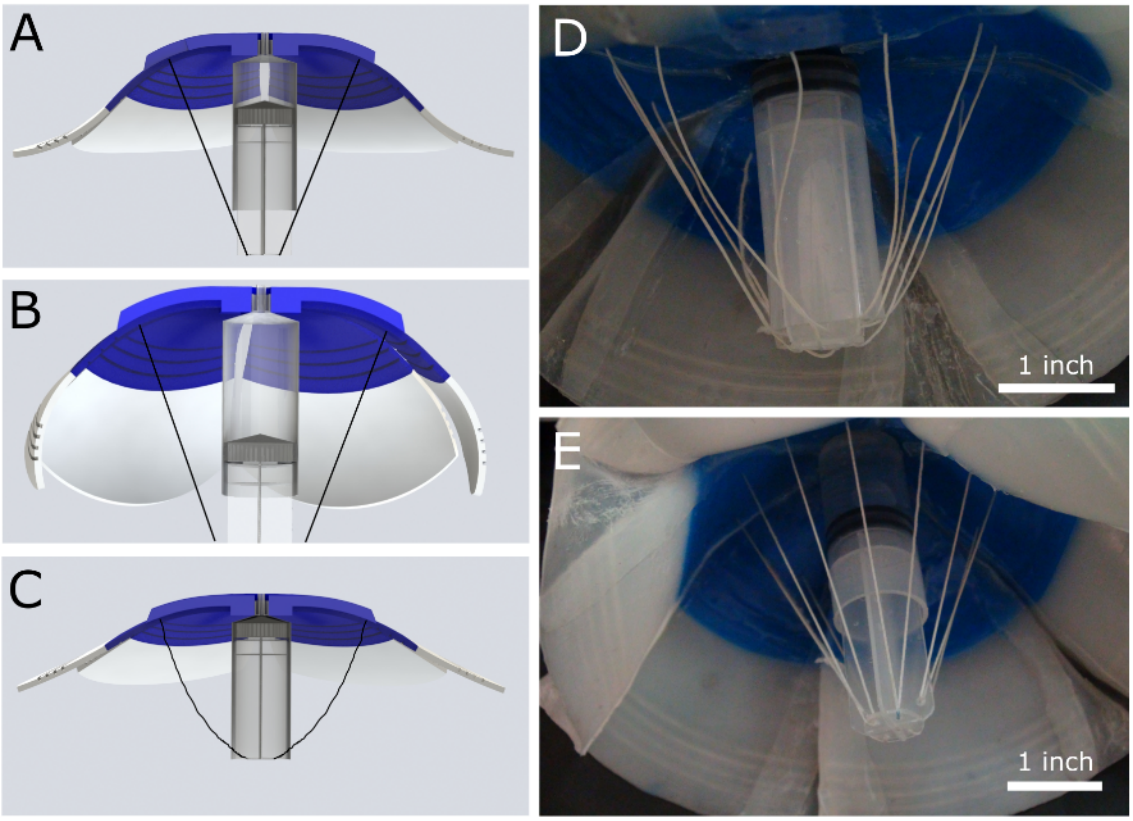


Figure 3. Computer drawings and images of our locomotor demonstrating how the use of a syringe as a linear actuator affected the curvature of the fins. (a) The locomotor started to contract and partially filled the syringe. The outer edge of the bell curved outward while the locomotor traveled upward. (b) The locomotor was in its contraction phase when the syringe was full. The outer edge of the bell pointed slightly inwards. (c) The locomotor was in its relaxation phase when the syringe was empty. (d) There was no tension from the strings acting on the bell. (e) The strings pulled the bell structure downward when the syringe was full.

2.3. Automated Control for the Jellyfish-Based Locomotor

A controlled hydraulic system delivered hydraulic power to the locomotor. We designed a control board consisting of three linear actuators, corresponding linear potentiometers and motor boards, and a microcontroller, as shown in Fig. 4. The linear potentiometer measured the position of the linear actuator. The microcontroller, an Arduino Uno, controlled and measured the positions of the linear actuators and voltage applied through MATLAB and Simulink. Figures 5 through 7 show the Simulink block diagrams used. Using a proportional-integral-derivative (PID) controller and pulse width modulation (PWM), we set and measured the exact positions of the linear actuators. Since we assigned the desired positions of the linear actuators, we were able to control the amount of water delivered to the locomotor. We also controlled the period and length of contraction and relaxation phases of the locomotor. Water traveled to the locomotor from two 12-mL syringes through silicone tubing and tube fittings. Fittings and stands for the syringes were 3D printed and placed such that the linear actuators pushed and pulled the syringes. Another microcontroller, an Arduino Uno, measured the pressure separately, using another laptop. A differential pressure sensor, by Honeywell©, measured the pressure difference between the tubes and the atmosphere.

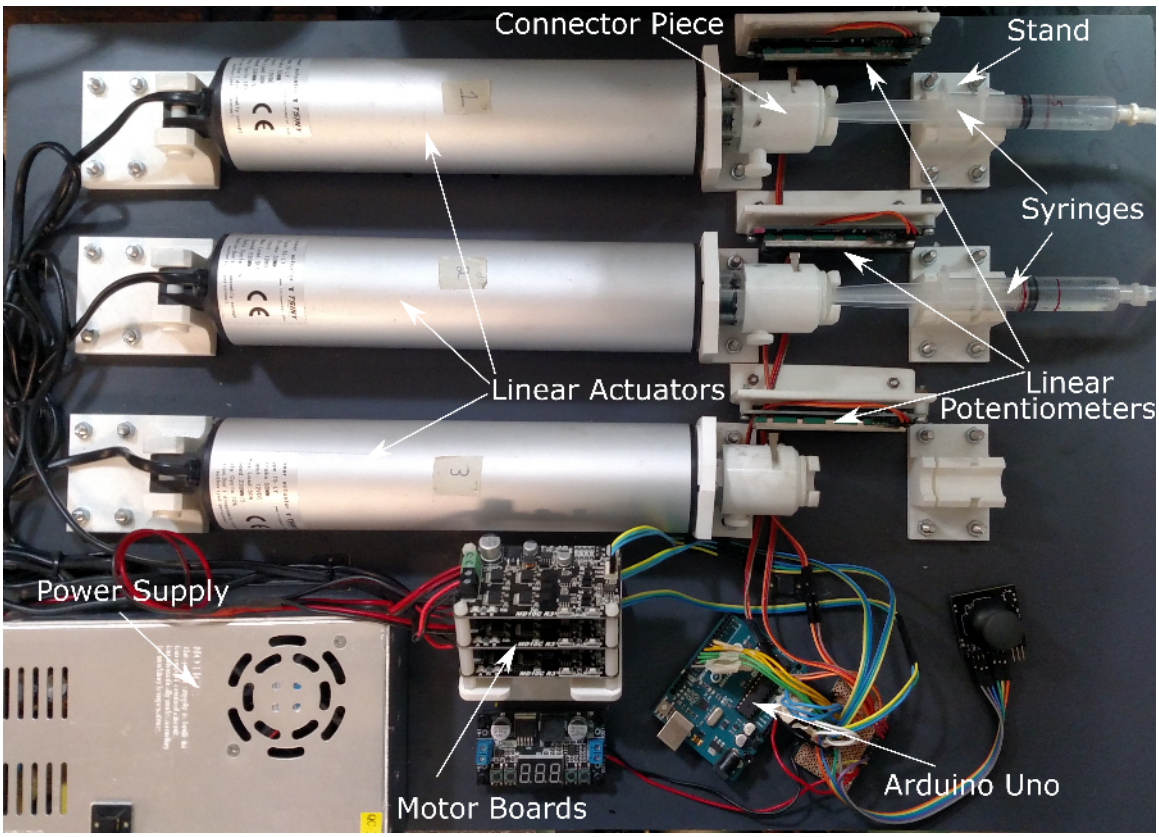


Figure 4. The automated hydraulic system used to control the locomotor. It consisted of three linear actuators, motor boards, linear potentiometers, an Arduino Uno, a power source, and various 3D printed parts. The control board delivered water to the attached locomotor through syringes. A connector piece attached the end of the syringe and the front of the actuator, which allowed the linear actuators to push and pull the syringes. A stand placed three inches in front of the actuator propped the syringe up, and zip ties held it in place. The linear potentiometers coupled to the connector pieces measured the position of the linear actuators.

(Ke Yang[15] designed and built this control board.)

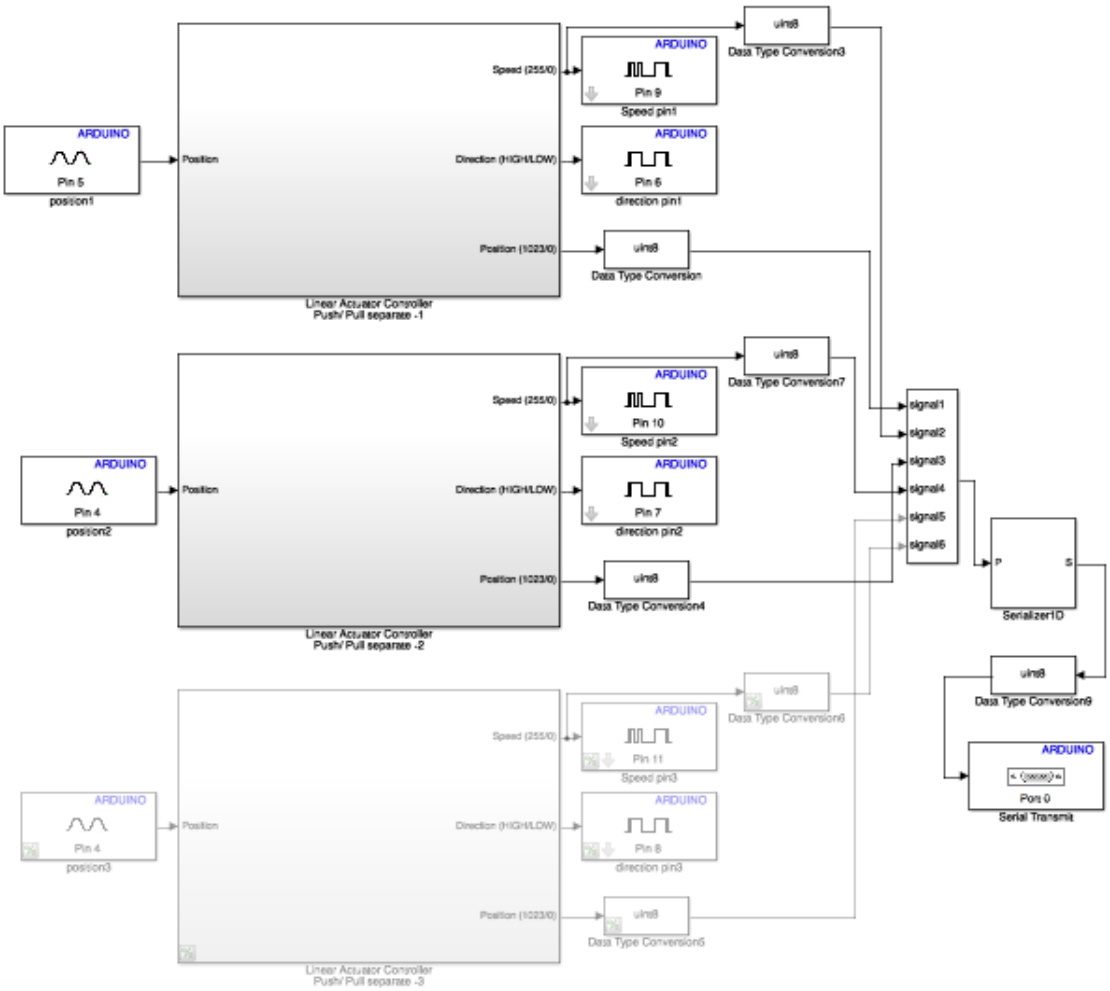


Figure 5. The main Simulink block diagram used to control the positions of the linear actuators. (The third linear actuator is disabled). The input was the current position of the actuator, and the outputs were the voltage applied to and the direction of the actuator. The sampling rate was 100 Hz. The serialization of the position and speed data allowed for data transmission between the microcontroller and laptop.

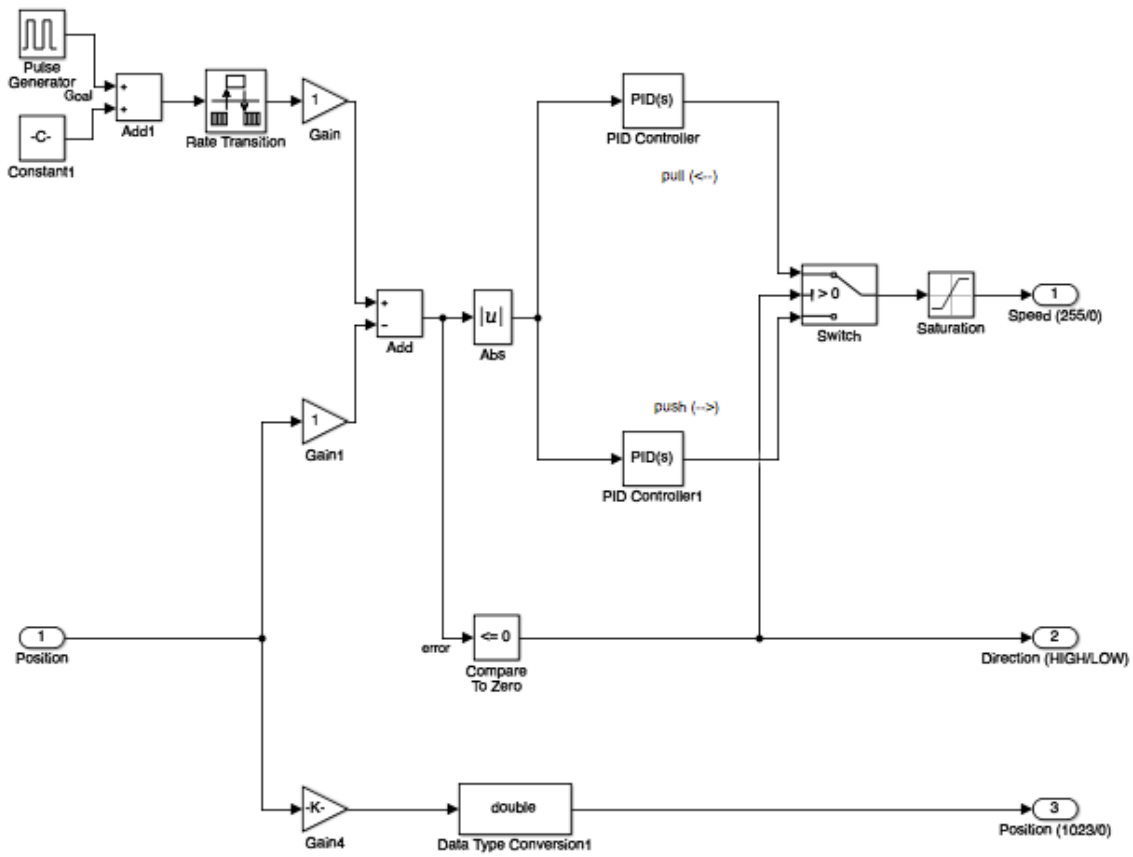


Figure 6. The Simulink block diagram inside one of the "Linear Actuator Controller" blocks. The pulse generator block was a square wave with the amplitude of the set desired position of the linear actuator. The PID controller determined the voltage applied to the linear actuator using the error, which was the difference between the desired position and the actual position. The sign of the error determined the direction of the linear actuator.

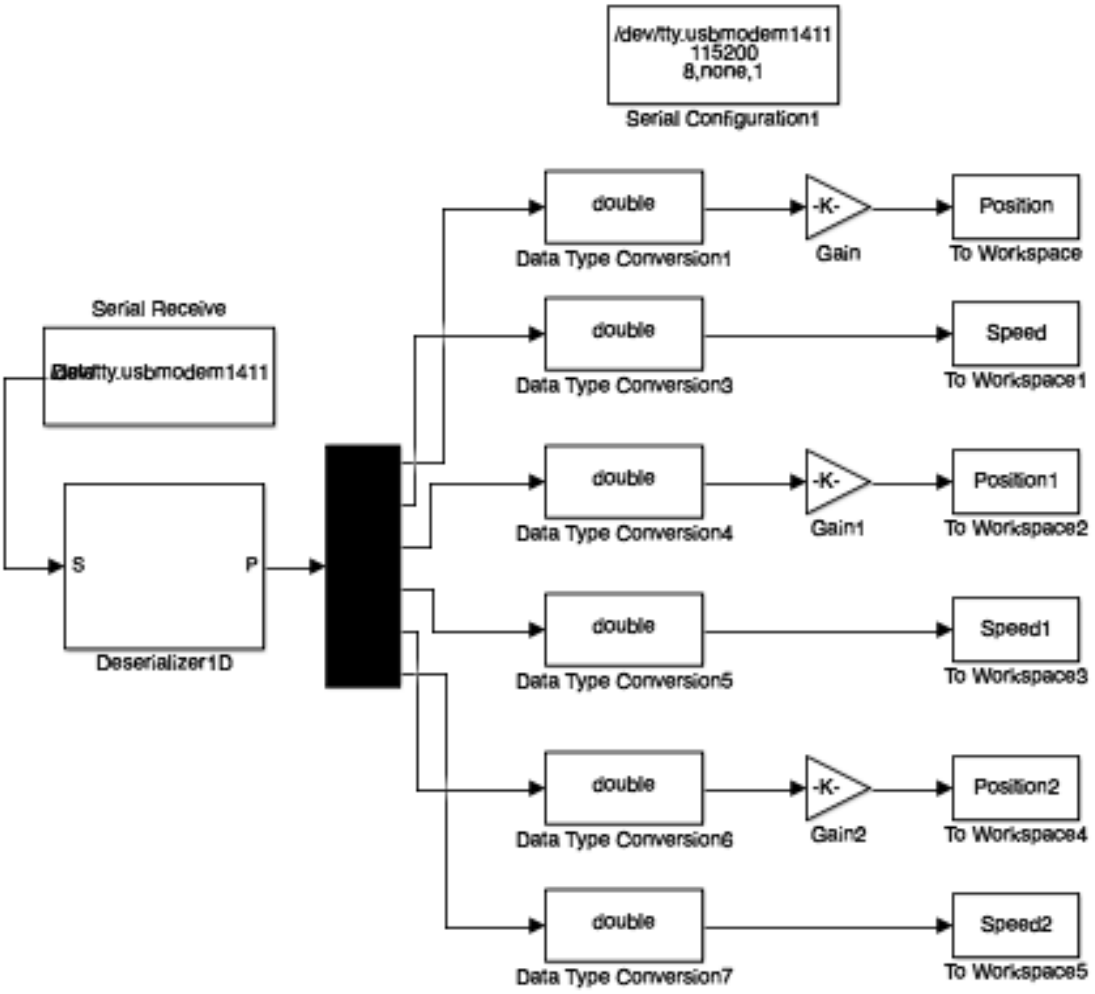


Figure 7. A Simulink block diagram that deserialized the position and speed data from the microcontroller and assigned variable names for the data sent into the MATLAB workspace.

2.4. Characterization and Measurement Methods

We tethered the jellyfish-based locomotor to the controlled hydraulic system by silicone tubing and placed it into a large vertical tank filled with water, as shown in Fig. 8a. We built the vertical tank using five half-inch thick sheets of clear tempered glass and silicone sealant. The vertical tank had an 18-inch square base and was 36 inches tall. We made a long cardboard clamp that lies across the top of the tank and a moveable fixture to hold the tubing to the locomotor in place at the top of the tank (see Supp. Fig. 1). We filled two 12-mL syringes with 8 mL of water each and slotted them into the connector pieces attached to the linear actuators. Since we were only interested in the vertical displacement of the locomotor, we attached a black Styrofoam piece to the top of the bell structure, which pointed it upwards. We also attached a red ball on top of the black Styrofoam for easier tracking in MATLAB. Overall, the locomotor had slightly negative buoyancy. We placed a camcorder on the table across from the tank and recorded the vertical swimming motion of the locomotor at 60 frames per second. Before each recording, we made sure the water was steady before we began actuating the locomotor. We also placed markers on the tank for relative scale lengths to convert pixels to distance traveled in inches. We used image processing and g-input in MATLAB to find the coordinates of the center of the red ball over time. We smoothed the displacement data using a robust version of local regression using weighted linear least squares (zero weight to data outside six mean absolute deviations) and a 2nd-degree polynomial model. We calculated the velocity by taking the derivative of displacement with respect to time. We repeated this procedure for various speeds of actuation.

We used particle image velocimetry (PIV) to study the vortices produced by the jellyfish inspired “flapping” motion of the locomotor. We introduced ten grams of 300-355 micrometer fluorescent green polyethylene microspheres into the vertical tank. We placed a projector on the right side of the tank to project a thin strip of white light and to illuminate a plane in the tank, as shown in Fig 8b. We also placed a camcorder on the table across from the tank and recorded the vertical swimming motion of the locomotor at 60 frames per second. Using the video footage, we extracted the frames and used the PIVlab 1.4 toolbox[19] in MATLAB to process the frames to be able to visualize the flow around the locomotor. With the same frames, we used image processing and g-input in MATLAB to measure and calculate the associated displacement and velocity data. We repeated this procedure for various speeds of actuation.

The cost of transport (COT), defined as work per mass per distance traveled, determined the efficiency of the locomotor. We calculated it using data collected for the displacement of the locomotor, position of the linear actuators, and pressure measurements. We used the position data acquired from the linear potentiometers from the control board to find the volume of water injected into the system with respect to time. We also measured the differential pressure between inside the tubes and the atmosphere. For each trial, we started the actuation, camcorder, and pressure readings at the same time. We repeated this procedure for various speeds of actuation multiple times to obtain an estimated average for the calculated cost of transport values.

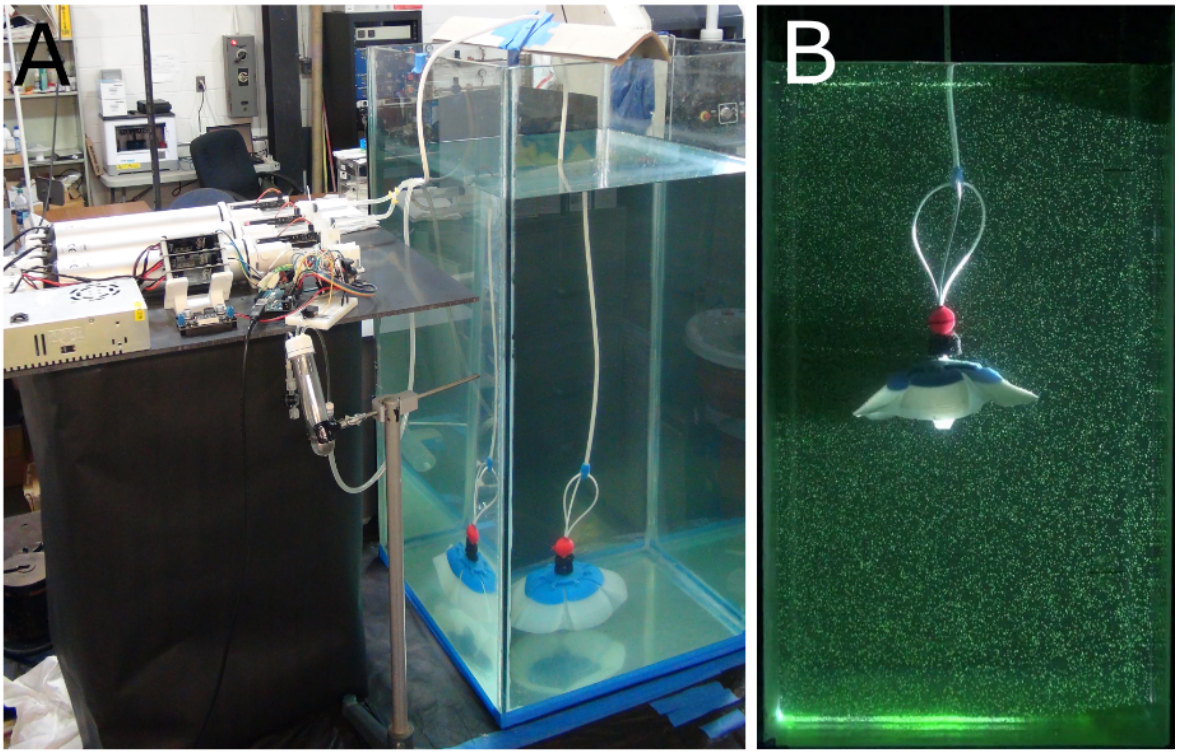


Figure 8. (a) The experimental set-up with the control system connected to the soft locomotor in the custom-built vertical tank used to observe the vertical movement of the locomotor. (b) One frame from a video footage recorded, at 60 frames per second, of the locomotor in the particle-filled tank. The room had to be completely dark. The projector projected a thin strip of white light from the right side of the tank, and the camcorder recorded perpendicularly to the plane from the front of the tank.

3. Results and Discussion

Initially, we selected three actuation rates based on observations during preliminary testing. Each actuation cycle consisted of a contraction phase and relaxation phase (see Supp. Fig. 3). We considered the fast gait actuation rate to be 0.75 seconds per cycle, the moderate gait actuation rate to be 1.2 seconds per cycle, and the slow gait actuation rate to be 2.0 seconds per cycle. The duration of contraction and relaxation phases was the same. Our automated hydraulic control system allowed us to change the actuation rate and length of contraction of the jellyfish-based locomotor accurately and easily. Based on these three initial actuation speeds and observations, we determined a range of actuation rates and actuated the locomotor at various rates, ranging from 0.75 seconds per cycle to 3.2 seconds per cycle. We chose to analyze the collected data further for actuation rates of 1.0, 1.6, 2.2, and 2.5 seconds per cycle. The lengths of contraction and relaxation phases remained the same for each rate. We recorded video footage of the locomotor swimming in a vertical direction in a particle- filled tank of water. Using these videos, we obtained and calculated displacement and velocity measurements. We also analyzed the associated fluid flow field around the locomotor through PIVlab 1.4 in MATLAB. Based on the observation and collected data, we considered the actuation rates between 0.75 seconds per cycle and 1.2 seconds per cycle to be fast gaits, the actuation rates between 1.4 seconds per cycle and 2.0 seconds per cycle to be moderate gaits, and the actuation rates between 2.2 and 3.2 seconds per cycle to be slow gaits. The following table categorizes the actuation rates we tested.

Table 1. The categorized actuation rates as fast, moderate, and slow gaits.			
Gait	Actuation Rate (seconds per cycle)		
Fast	0.75	1.0	1.2
Moderate	1.4	1.6	2.0
Slow	2.2	2.5	3.2

3.1 Position of the Linear Actuators

We controlled the position of the linear actuators by using a PID controller and a feedback loop through Simulink and MATLAB. We used pulse width modulation to change the voltage applied the linear actuator based on the error between the desired position and current position. We tuned the first order system to have a fast response time with no overshoot or oscillations. We used the full stroke of the linear actuators because of the required volume of water delivered to the locomotor.

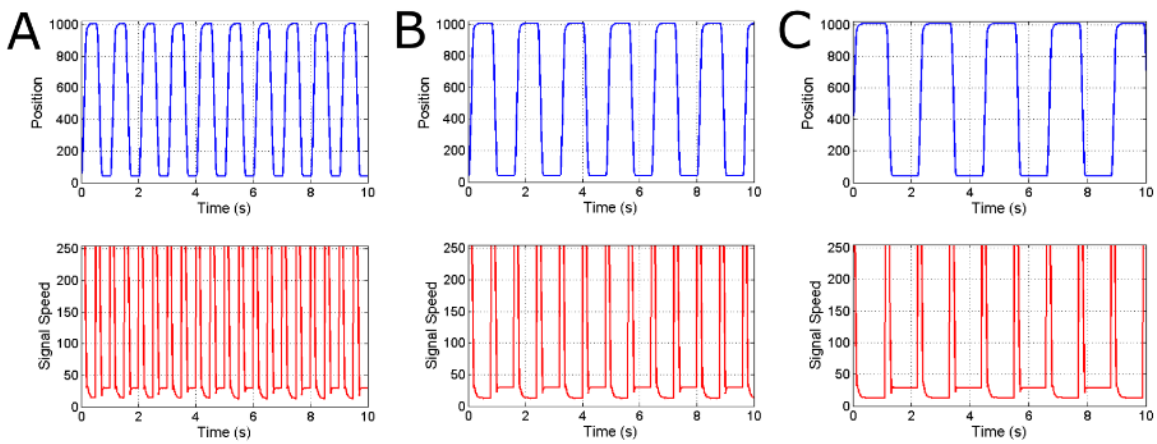


Figure 9. Position and voltage applied (V/255) over time for the linear actuators on the control board. Pushing the two syringes attached to the linear actuators corresponded to the contraction of the bell and pulling the syringes corresponded to the relaxation of the bell. The actuators moved backward (pulling the syringes) when the readings of the linear potentiometer are from zero to 1000, and the actuators moved forward (pushing the syringes) when the readings of the linear potentiometer are from 1000 to zero. The locomotor started contracted for each of the video footages.

3.2. Particle Image Velocimetry Analysis

We used PIVlab 1.4[19] in MATLAB to visualize the vortices associated with the fluid flow around the jellyfish-based locomotor at various stages in an actuation cycle at three rates, 1.0, 1.6, and 2.2 cycles per second, shown in Figures 10 through 12. The frames processed were when the locomotor had already reached steady state. There were vortices left over from the previous cycle, and each cycle started with the contraction phase and ended with relaxation.

The vertical propulsion of the locomotor exhibited different flow patterns for each actuation rate. For the fast gait, an actuation rate of 1.0 second per cycle, the stopping vortices traveled outward and away from the bell. The starting vortices shed off the bell and dissipated quickly with the surrounding water. For the moderate gait, an actuation rate of 1.6 seconds per cycle, the stopping vortices traveled downward and under the bell. The starting vortices partially merged with the stopping vortices during the relaxation phase. The previous starting and stopping vortices remained present in the wake of the bell during the next cycle but gradually became weaker. For the slow gait, an actuation rate of 2.2 seconds per cycle, the starting and stopping vortices traveled downward and under the bell without merging. The vortices remained present in the wake of the bell during the next few cycles.

While the vortices in the slow and moderate gaits both moved downward and under the bell, in the slow gait, the vortices moved downward faster and further away from the bell while remaining in its wake. In the moderate gait, the vortices remained closer to the bell. We observed that the flow pattern for the moderate gait most closely followed the flow pattern of an oblate jellyfish.

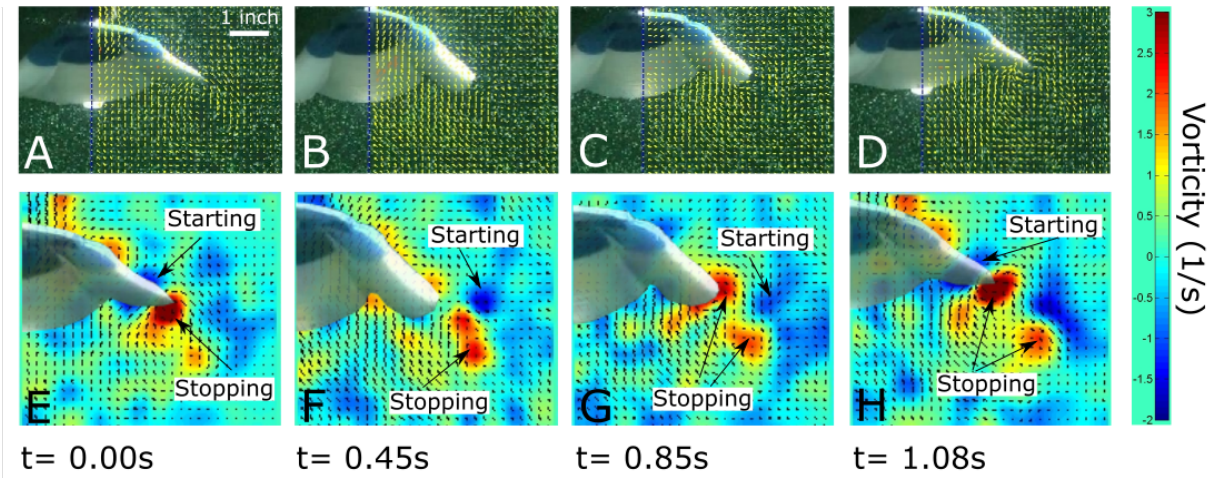


Figure 10. The visualization of the fluid flow field around the jellyfish-based locomotor at various stages in an actuation cycle obtained at a fast actuation rate of 1.0 second per cycle. (a), (e) The locomotor began contraction and a starting vortex formed at the soft edge of the bell. There was a stopping vortex from the previous cycle. (b), (f) During the contraction phase, the starting vortex shed off the bell, and the stopping vortices dissipated and moved downward, away from the bell. (c), (g) During the relaxation phase, a new stopping vortex drew water under the bell as the previous starting and stopping vortices continued to dissipate further away from the bell. (d), (h) At the start of the next contraction phase, another starting vortex formed at the edge of the bell. The previous stopping vortices traveled outward and away from the locomotor.

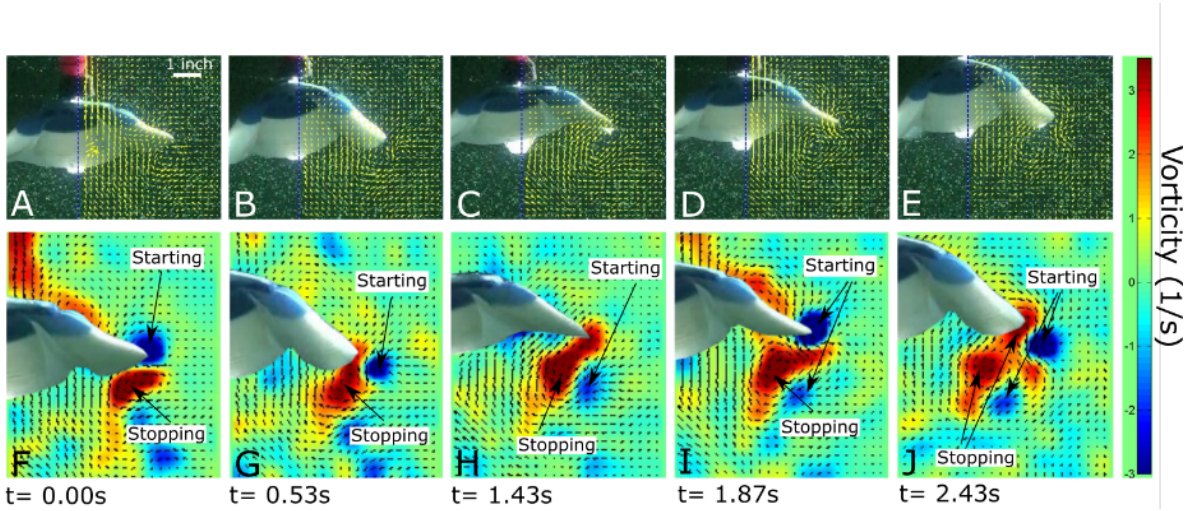


Figure 11. The visualization of the fluid flow field around the jellyfish-based locomotor at various stages in an actuation cycle obtained at a moderate actuation rate of 1.6 seconds per cycle. (a), (f) The locomotor began with a power stroke and a strong starting vortex formed at the edge of the bell. The strong stopping vortex present was from the previous cycle. (b), (g) During the contraction phase, the locomotor shed off the starting vortex while drawing water under the bell. The stopping vortex formed at the edge of the bell and merged with the previous stopping vortex. (c), (h) During the relaxation phase, the starting vortex partially merged with the strong stopping vertex. The enhanced stopping vortex and starting vortex moved downwards and under the bell. (d), (i) At the start of a new cycle, the locomotor began to contract and a new starting vortex formed at the edge of the bell. The previous stopping vortex started to dissipate under the bell. (e), (j) The starting vortex shed off the bell. The previous starting and stopping vortices moved downward and under the bell.

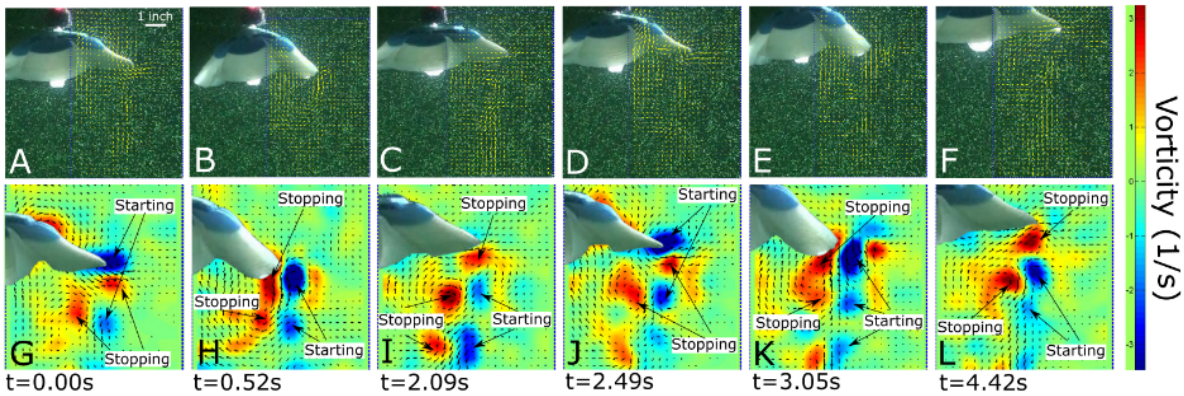


Figure 12. The visualization of the fluid flow field around the jellyfish-based locomotor at various stages in an actuation cycle obtained at a slow actuation rate of 2.2 seconds per cycle. (a), (e) The locomotor began contraction and a strong starting vortex formed at the edge of the bell. There were two stopping and one starting vortices present from the previous cycle. (b), (h) At the end of the contraction, the strong starting vortex shed off the bell and a stopping vortex formed at the edge of the bell. The previous stopping and starting vortices moved downwards. (c), (i) During relaxation, another stopping vortex formed under the bell. The previous starting and stopping vortices continued to move under and toward the center of the bell. (d), (j) The locomotor began contraction again and a strong starting vortex formed. (e), (k) As the locomotor ended contraction, a stopping vortex formed again, and the previous vortices continued to move downward. (f), (l) As the locomotor relaxed, a stopping vortex formed and the previous vortices began to weaken and moved under the bell.

3.3. Displacement and Velocity Analysis

As stated in Section 2.4, we recorded the vertical movement of the locomotor in the tank using a camcorder. We obtained the displacement of the locomotor over time by tracking the attached red ball through image processing of the frames extracted from the videos using MATLAB. We showed several periods at each speed of actuation in Fig 13. From the displacement data, we calculated the actual period of each cycle of the locomotor to be 1.10, 1.78, and 2.45 seconds per cycle. The actual period of the locomotor was higher because of the time it took for the water to travel through a significant length of tubing. The displacement and velocity versus time plots showed two or three stages in an actuation cycle. In the displacement versus time plots, during the contraction phase, the locomotor moved upward, and there was an increase in velocity. We observed a difference in the relaxation phases for various actuation rates. The fast gait showed two regions: a contraction phase and a relaxation phase. The moderate and slow gaits showed three regions, a contraction phase, a relaxation phase, and a post-relaxation (PER) phase. We also observed that the locomotor required more actuation cycles to travel the same distance at faster gaits. In the velocity versus time plots, the moderate and slow gaits showed an extra acceleration period after the relaxation phase. The post relaxation phases occurred in moderate and slow gaits because the starting and stopping vortices moved downwards and under the bell to push the locomotor further upward. The additional phases seen in the Fig. 13b, c were indicators of the passive energy recapture phenomenon found in living moon jellyfish. The distance traveled during contraction of the shown cycle is the largest for the slow gait while the largest maximum velocity attained corresponds to the moderate gait.

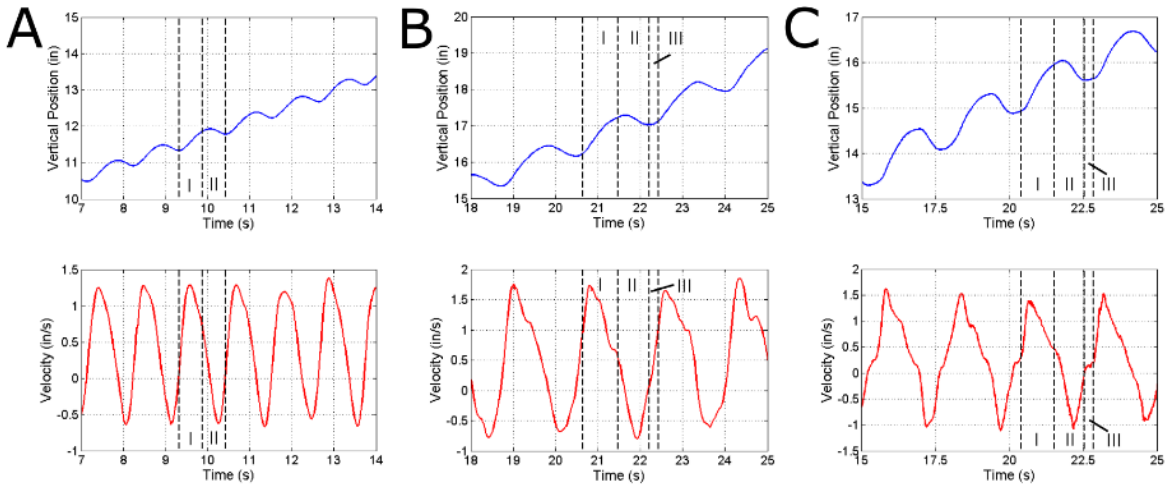


Figure 13. These are the plots for displacement and velocity with respect to time of the jellyfish-based locomotor for actuation rates of 1.0, 1.6, and 2.2 seconds per cycle. The displacement data was smoothed with a robust version of local regression using weighted linear least squares and a span of 1.5% to 2.5% of the total number of data points. (a) For an actuation rate of 1.0 second per cycle, there were two distinct regions. During the second region, the relaxation phase, the locomotor initially continued to move upward but then decreased in altitude due to its weight, and the vortices shed outwards from the bell. There was no presence of PER. The vertical distance during contraction was 0.59 inches and the maximum velocity attained was 1.28 inches per second. (b) For an actuation rate of 1.6 seconds per cycle, there were three regions. In the third region, the post relaxation phase, as the vortices traveled under the bell, the locomotor moved upwards with a small increase in velocity. The vertical distance traveled during contraction was 1.04 inches and the maximum velocity attained was 1.73 inches per second. (c) For an actuation rate of 2.2 seconds per cycle, there were also three regions. The vertical distance traveled during contraction was 1.102 inches and the maximum velocity attained was 1.40 inches per second.

3.4. Cost of Transport Analysis

Using the position data from the linear potentiometers, we calculated and plotted the volume of water delivered to the locomotor with respect to time. We obtained pressure data from a microcontroller and laptop separate from the one used for the linear actuators. We collected the pressure data at the same time as we recorded the videos for the displacement data. We calculated the work done by the locomotor for one contraction phase by finding the area under the pressure-volume curve using a midpoint Riemann sum approximation. By watching the video footage, we counted the number of contractions it took for the locomotor to travel a certain distance after it reached steady state and multiplied it by the area under one pressure-volume curve to obtain the total work done by the locomotor during its contraction phases. We took the exact distance the locomotor traveled from our displacement data. We calculated the cost of transport using the equation $\mathrm{COT} = \frac{Energy}{mass \cdot distance} = \frac{\sum P \cdot \Delta V}{m \cdot d}$. We repeated the process for each actuation rate three to six times to find an average cost of transport. The calculated average cost of transport values are shown below.

Table 2. The average cost of transport values for each actuation rate.

Actuation Rate (seconds per cycle)	0.75	1.0	1.6	2.0	2.2	2.5
Mean COT (J/kg/m)	75.50	33.15	27.53	20.22	15.04	39.37

The calculated cost of transport decreased with the increase in the length of each actuation cycle until it reached a “critical” actuation rate. At a slow enough actuation rate, the locomotor still traveled vertically, but the cost of transport increased. However, when the actuation rate was too slow, the locomotor did not move at all.

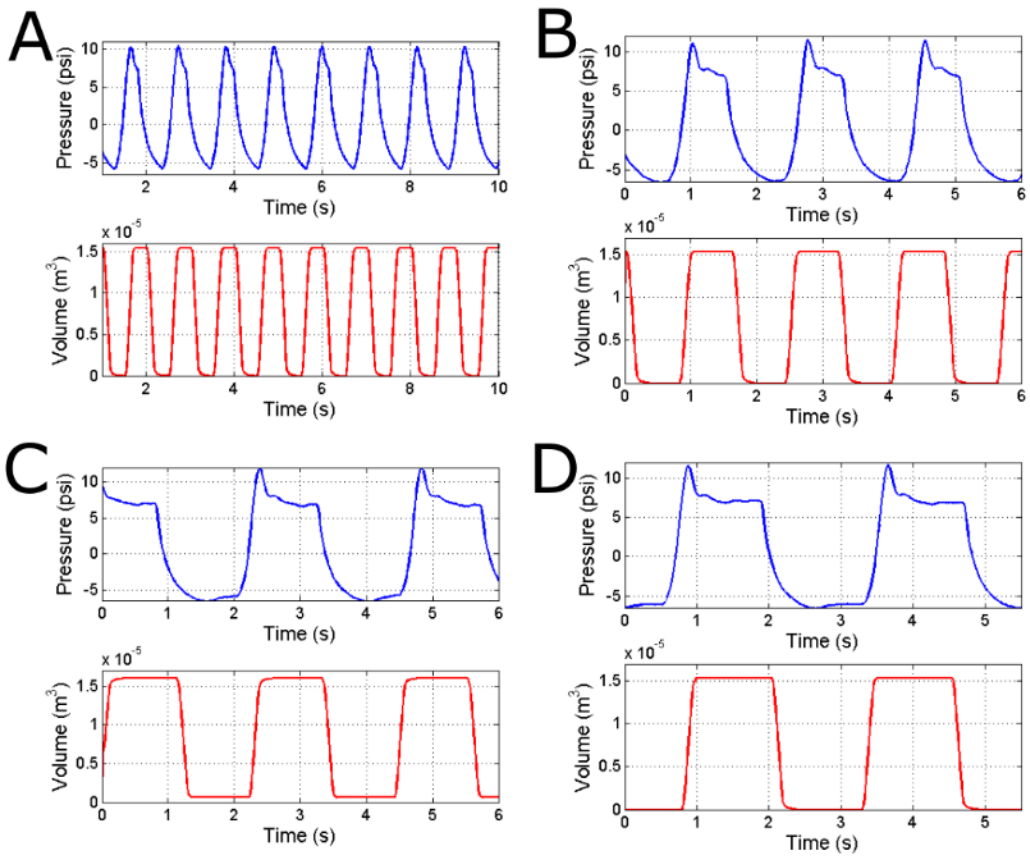


Figure 14. These are the plots for pressure and volume with respect to time of the jellyfish-based locomotor for actuation rates of 1.0, 1.6, 2.2, and 2.5 seconds per cycle. The work calculated for COT is the area under the pressure volume curve. (a) For an actuation rate of 1.0 second per cycle, the pressure versus time plot produced a sharp increase in pressure with almost no pauses. The maximum pressure obtained was 10.35 psi. (b) For an actuation rate of 1.6 seconds per cycle, the pause between cycles increased. The maximum pressure obtained was 11.64 psi. (c) For an actuation rate of 2.2 seconds per cycle, the maximum pressure obtained was 12.11 psi. (d) The maximum pressure obtained was 11.87 psi, which was lower than the maximum pressure from the previous actuation rate.

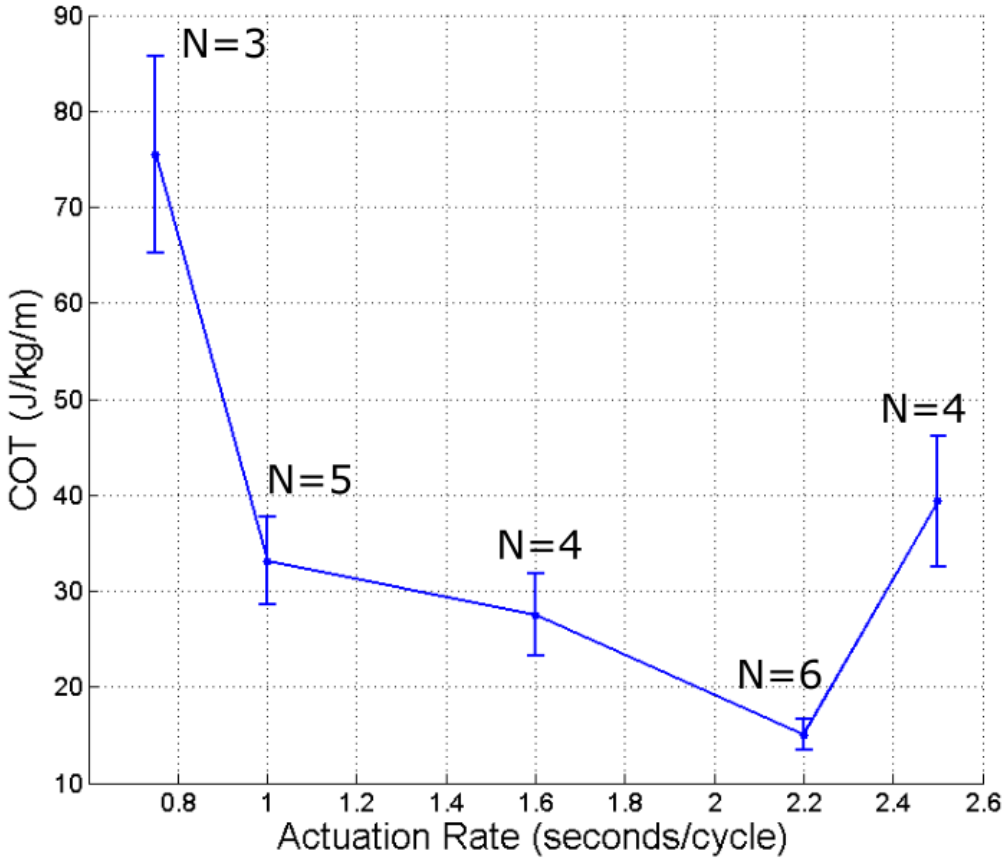


Figure 15. The average cost of transport decreased with increase in actuation rate. The "N" represents the number of trials done for that actuation rate. We calculated the error bars using a 95% confidence interval. The efficiency of the locomotor was the lowest with the slowest actuation rate of 0.75 seconds per cycle and highest with the actuation rate of 2.2 seconds per cycle. The efficiency decreased with the actuation rate of 2.5 seconds per cycles.

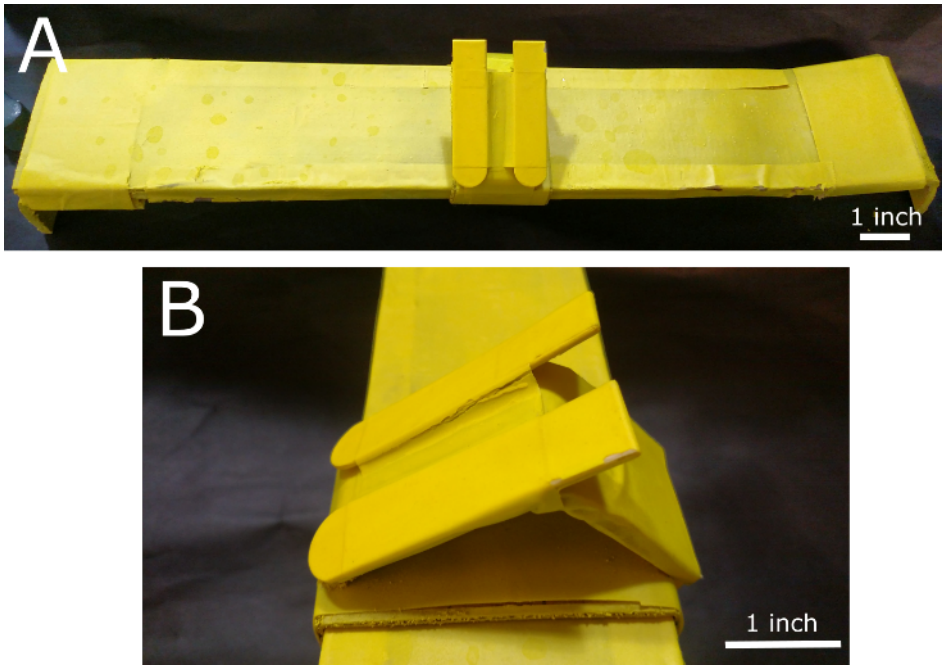
4. Conclusion

Our soft jellyfish-based hydraulic locomotor mimics the flow patterns observed in nature for the oblate jellyfish species. At certain gaits, the locomotor also demonstrates the passive energy recapture phenomena observed in living moon jellyfish. In moderate and slow actuation rates, we observed a post relaxation phase, shown by another period of acceleration before the next contraction. From our particle velocimetry analysis, we conclude that the post relaxation phase occurs because of the starting and stopping vortices that travel under the bell of the locomotor. From observation, a slower gait requires fewer contractions of the locomotion to travel the same distance as a faster gait. However, at a slow enough actuation rate, the locomotor is incapable of vertical motion. Overall, the efficiency of the locomotor reaches a local maximum at a slow gait with a period of 2.2 seconds.

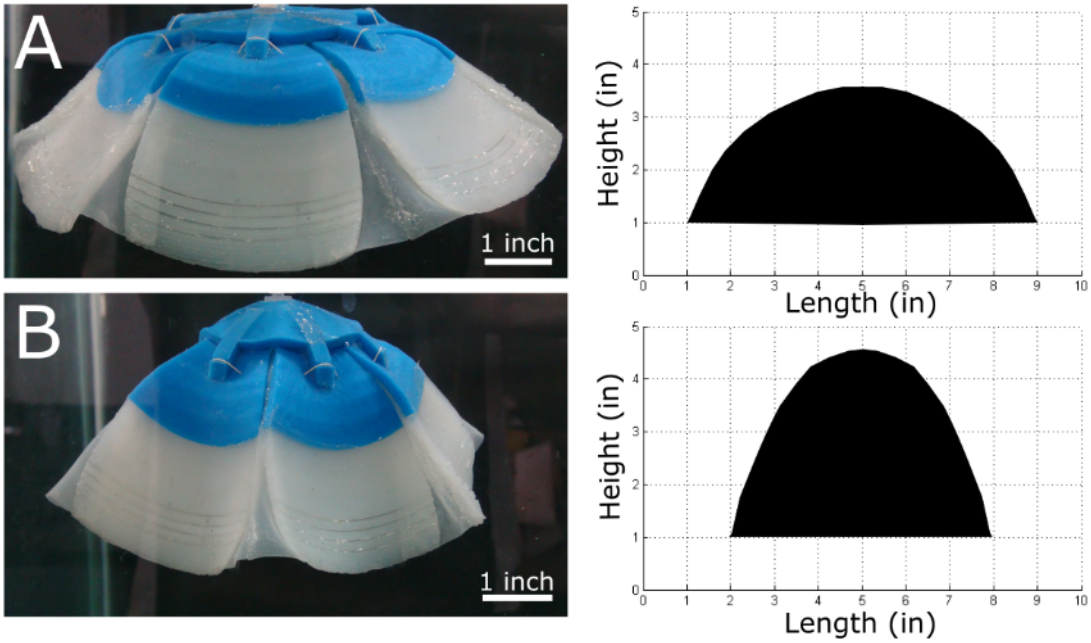
Our results suggest that the jellyfish could swim at a gait that improves their propulsive efficiencies. At a moderate gait, a large volume of water moves closer under the bell, which increases feeding efficiency but at the cost of propulsive efficiency. Our data suggest that the jellyfish would be more efficient if it swam at a slower gait.

In the future, we look to analyze the fluid flow and calculate the cost of transport for more actuation rates and to compare to the propulsive efficiency of living jellyfish. We can also look at different curvatures of the bell during contraction and relaxation to see how they affect the fluid flow and the associated cost of transport.

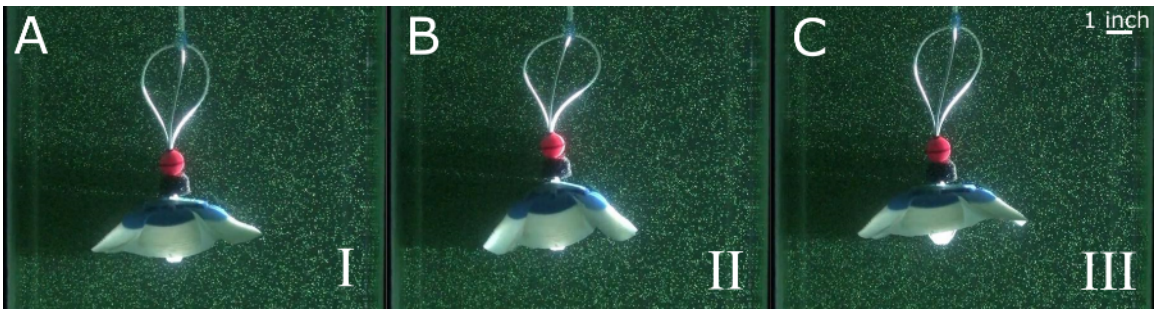
Appendix A: Supplementary Figures



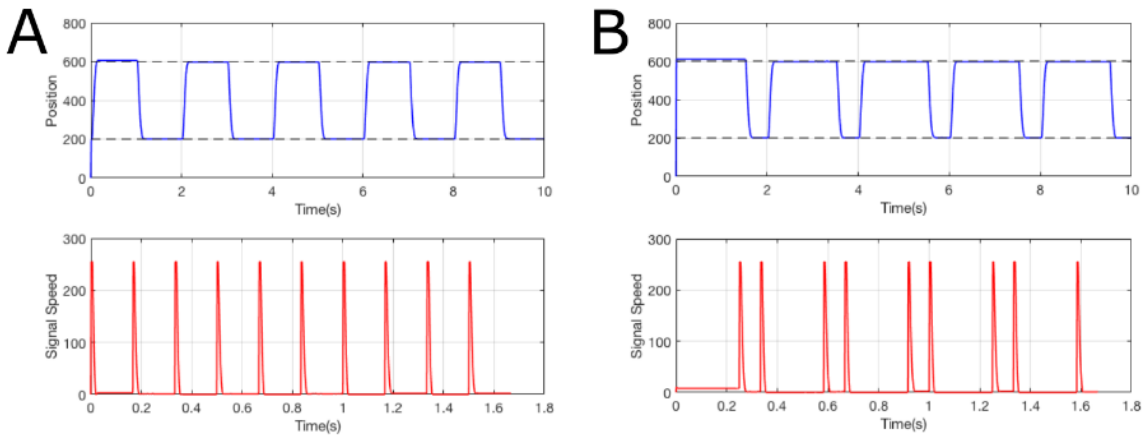
Supplementary Figure 1. (a) We made a cardboard structure to fit around the top of the tank. The location of it remained the same for all actuation rates of the locomotor. (b) The fixture was free to move across the cardboard, and the tubing lied between the wooden pieces.



Supplementary Figure 2. The curvature of the bell at static phases in relaxation and contraction when the locomotor was in water changed its diameter and height. (a) When the locomotor relaxed, it had a bell diameter of 8 inches and a height of 2.5 inches. (b) When the locomotor contracted, it had a bell diameter of 6 inches and a height of 3.5 inches.



Supplementary Figure 3. The vertical movement of the locomotor in the particle- filled tank for one actuation cycle. (a) During the initial power stroke, the fin-like segment began to curve. (b) At the end of the contraction, fin-like segment pointed downwards (c) During relaxation, the fin-like segment pointed outward.



Supplementary Figure 4. Plots showing the control of position and duty cycle of the linear actuators. (a) The desired position of the linear actuators set was between 200 and 600. The plots show the linear potentiometer readings for the position of the linear actuators over time. This allows for control over the amount of water delivered to the locomotor. (b) Changing the duty cycle allows for control over the lengths of the contraction and relaxation phases.

Appendix B: MATLAB Code

1. Extract frames from a video footage

```
%% Get Frames from a Video

obj = VideoReader('pd3k2_p35.mov'); % [Change video name]

video = obj.read();


for k = 1 : length(video(1,:,:,:))

  this_frame = read(obj, k);

  filename = sprintf('img_%d.jpg', k);

  imwrite(this_frame, filename, 'jpg')

end
```

2. Renames the files to have the same file name length in the folder

```
%% Rename Files to Same Length

files = dir('pd3k2_p35\*.jpg'); % [Change folder name]

d = 'L:\Lillian\Documents\MATLAB\PIV_July26\pd3k2_p35\'; % [Change path name]

names = dir(d);

names = {names(~[names.isdir]).name};

len  = cellfun('length',names);

mLen = max(len);

idx   = len < mLen;

len   = len(idx);

names = names(idx);
```

```
for n = 1:numel(names)
   oldname = [d names{n}];
   newname = sprintf('%s%0*s',d,mLen, names{n});
   movefile(oldname, newname)
end
```

3. Image tracking code for the position of the locomotor in the tank. It finds the center of the red ball and uses g-input for the user to click on the center of the ball, if it cannot be found.

```
%% Image Tracking
% Gets (x,y) of the red ball
frames = 2101; % [Change # of frames]
video_t = 35; % [Change length of video]
t = linspace(0,video_t,frames);
myFolder = ' L:\Lillian\Documents\MATLAB\Code For Thesis; % [Change path name]
filePattern = fullfile(myFolder, '*.jpg');
jpegg = dir(filePattern);

S = imread('00000img_1.jpg'); % [Change image name]
S = imrotate(S,180);
cropscale= [0 0 608 1080];
imshow(imcrop(S, cropscale));
[x1,y1] = ginput(4); % X clicks first, then Y clicks
close
```

```
x = x1(2,1) - x1(1,1);
y = y1(4,1) - y1(3,1);
xscale = 1/x;
yscale = 1/y;

for k = 1:length(jpegg)
   jpegFiles = strcat(jpegg(k).name);
   I = imread(jpegFiles);
   C = imrotate(I,180);
   T = imcrop(C, cropscale); % [xmin ymin width height]
   R = imsubtract(T(:,:,1), rgb2gray(T));
%     thresh = graythresh(R);
%     bw = (R >= thresh * 255);
%     bw2 = bwareaopen(bw, 20, 4);
   [c, r] = imfindcircles(R,[10,20]);
   fprintf(1, 'Now plotting %s\n', jpegFiles);
   temp = 0;
% If no center is found, manually click the center of the red ball
   if size(c) == 0
      imshow(C)
      [c(k,1), c(k,2)] = ginput(1);
      close
      c(k,2)=c(k,2)-0.5;
   else
   r_x(k,1) = 16.5-( c(1,1).*xscale); % x coordinate of red ball
```

```
    r_y(k,1) = c(1,2).*yscale; % y coordinate of red ball
    disp(r_x(k,1))
    disp(r_y(k,1))
    end
end
```

4. Smooths displacement data and calculates velocity. Plots displacement and velocity with respect to time

```
%% Smooth Displacement
close all
r_smooth = smooth(t,r_y,0.02, 'rloess'); %lower = closer to actual data
dt = video_t/frames;

% Calculate Velocity
for i = 1:length(r_y)-1
    %v(i) = (r_y(i+1,1) - r_y(i,1))/dt;
    v(i) = (r_smooth(i+1,1) - r_smooth(i,1))/dt;
end

% Plot smoothed displacement and velocity
subplot(2,1,1)
plot(t,r_smooth,'b-','LineWidth',2)
xlabel('Time (s)', 'FontSize', 16);
ylabel('Vertical Position (in)', 'FontSize',16);
grid on
```

```
subplot(2,1,2)

plot(t(1,1:length(v)), v,'r-','LineWidth',2)

xlabel('Time (s)', 'FontSize', 16);

ylabel('Velocity (in/s)', 'FontSize',16);

grid on
```

5. Measures pressure

```
%% Read Pressure
% Run on another Arduino

tic
i = 1;
j = 1;
Pmin = -30;
Pmax = 30;
Vs = 5;
figure

while i > 0
   tp(j) = toc;
   x(j) = readVoltage(a, 'A0');
   disp(x)
   j = j + 1;
   pause(0.01)
end
```

6. Calculates volume of water going into the locomotor using the position of the actuators from the control board. (Example position data used for period of 2.5 seconds per cycle shown; position data used for other periods are similar.) Plots pressure with respect to time.

```
%% Plots for Pressure vs Time

e = (x)*(Pmax-Pmin)/(Vs) + Pmin; % convert Voltage into psi

plot(tp(1,1:size(e')), e, 'b.-','MarkerSize',5)
```

```
xlabel('Time (s)', 'FontSize', 16)
ylabel('Pressure (psi)', 'FontSize', 16)

grid on
%% Plots for Pressure vs Position
% Get Position1.Data- choose when it first starts increasing to decreasing
% 2.5s
pos_data =
[48.1411764705882;52.1529411764706;60.1764705882353;112.329411764706;152.4470588235
29;212.623529411765;280.823529411765;349.023529411765;417.223529411765;489.43529411
7647;557.635294117647;633.858823529412;702.058823529412;774.270588235294;838.458823
529412;898.635294117647;946.776470588235;962.823529411765;970.847058823529;978.8705
88235294;982.882352941177;986.894117647059;990.905882352941;994.917647058824;994.91
7647058824;998.929411764706;998.929411764706;998.929411764706;1002.94117647059;100
2.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1
002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.9411764705
9;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647
059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.941176
47059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.9411
7647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94
117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.
94117647059;1002.94117647059;1002.94117647059;1006.95294117647;1002.94117647059;10
02.94117647059;1002.94117647059;1002.94117647059;1006.95294117647;1002.94117647059;
1006.95294117647;1002.94117647059;1002.94117647059;1002.94117647059;1006.952941176
47;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.9411764
7059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117
```

```
647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.941
17647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.9
4117647059;1006.95294117647;1002.94117647059;1002.94117647059;1002.94117647059;100
2.94117647059;1002.94117647059;1002.94117647059;1006.95294117647;1002.94117647059;1
002.94117647059;1002.94117647059;1006.95294117647;1006.95294117647;1002.9411764705
9;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647
059;1002.94117647059;1002.94117647059;1002.94117647059;1002.94117647059;1002.941176
47059;1002.94117647059;1002.94117647059;1002.94117647059;1006.95294117647];

Vol = 16e-6; % cubic meters % 16 mL
Pos = pos_data*(.0762/(max(pos_data)-min(pos_data))); % convert to meters (3 in = 0.0762
meters)
volume = Vol*(Pos/.0762);
Pres = e * 6894.76; % convert psi to Pa

le = length(e);
lp = length(Pos);
rep = ceil(le/lp);

%% Pressure vs Volume Plot
% Get Pressure data- choose when it first starts increasing to decreasing
newPres = Pres(1,479:504); %2.5

newVol = volume(1:length(newPres));

plot(newVol, newPres', 'b.-', 'MarkerSize', 5)
```

```
xlabel('Volume (m^3)', 'FontSize', 16)

ylabel('Pressure (Pa)', 'FontSize', 16)
```

7. Calculate cost of transport (COT)

```
%% Cost of Transport Calculations

m = .264; % kilograms

dis = (r_y(length(r_y)) - r_y(1))*0.0254;  % distance traveled in meters

num = 24; % number of contractions [Change for each Video]


for qq = 1:length(newPres)-1 % Calculate the Area using Midpoint Riemann Sum

   dV(qq) = abs(newVol(qq+1))-abs(newVol(qq));

   Ps(qq) = abs((newPres(qq+1))+abs(newPres(qq)))/2;

   s(qq) = dV(qq)*Ps(qq);

end


P_V = sum(s);

COT = P_V*num/(m*dis);


disp('COT =')

disp(COT)
```

8. Plots COT with error bars.

```
%% Plot COT with Error bars

cot0k75 = [83.95337072

76.66539192
```

65.8857384];

cot1k0 = [40.58

28.25133965

28.17876348

33.18841619

35.5338937];

cot1k6 = [26.07

22.65598306

24.32102895

34.70649024

29.92773755];

cot2k0 = [16.77

20.4916933

20.12840095

22.79663703

20.8916284];

cot2k2 = [17.67

12.96270868

14.15402665

12.92957477

16.19224616

16.30710637];

```
cot2k5 = [47.2759
34.47749175
32.70286361
43.02883481];

m_0k75 = mean(cot0k75);
s_0k75 = std(cot0k75);
m_1k0 = mean(cot1k0);
s_1k0 = std(cot1k0);
m_1k6 = mean(cot1k6);
s_1k6 = std(cot1k6);
m_2k0 = mean(cot2k0);
s_2k0 = std(cot2k0);
m_2k2 = mean(cot2k2);
s_2k2 = std(cot2k2);
m_2k5 = mean(cot2k5);
s_2k5 = std(cot2k5);

% 95% confidence intervals
r_0k75 = s_0k75/(sqrt(length(cot0k75)));
r_1k0 = s_1k0/(sqrt(length(cot1k0)));
r_1k6 = s_1k6/(sqrt(length(cot1k6)));
r_2k0 = s_2k0/(sqrt(length(cot2k0)));
r_2k2 = s_2k2/(sqrt(length(cot2k2)));
r_2k5 = s_2k5/(sqrt(length(cot2k5)));
```

```
z = 1.960;

period = [0.75 1.0  1.6  2.2 2.5];
cot = [m_0k75 m_1k0  m_1k6  m_2k2 m_2k5];
hold on
%err = [s_0k75 s_1k0 s_1k6 s_2k0 s_2k2 s_2k5];
err = z*[r_0k75 r_1k0  r_1k6  r_2k2 r_2k5];
errorbar(period, cot, err, 'b.-','LineWidth',1.5,'MarkerSize',15)
xlabel('Actuation Rate (seconds/cycle)','FontSize',20)
ylabel('COT (J/kg/m)','FontSize',20)
grid on
set(gca,'FontSize',14)
```

9. Calculate distance between initial and final images for repeated tests.

```
myFolder = 'C:\Users\SoE\Documents\MATLAB'; % [Change path name]
filePattern = fullfile(myFolder, '*.png');
jpegg = dir(filePattern);

S = imread('2k5_1_End.png'); % [Change image name]
S = imrotate(S,180);
imshow(S)
[x,y] = ginput(2); % click taped distance (16.5 inches)
close

yscale = 16.5/abs(y(2)-y(1));
```

```
for k = 1:length(jpegg)
   jpegFiles = strcat(jpegg(k).name);
   I = imread(jpegFiles);
   C = imrotate(I,180);
   %T = imcrop(C, cropscale); % [xmin ymin width height]
   R = imsubtract(C(:,:,1), rgb2gray(C));
   thresh = graythresh(R);
   bw = (R >= thresh * 255);
   bw2 = bwareaopen(bw, 20, 8);
   [c, r] = imfindcircles(bw2,[10,20]);

   fprintf(1, 'Now plotting %s\n', jpegFiles);

   temp = 0;
   if size(c) == 0
      imshow(C)
      [c(k,1), c(k,1)] = ginput(1);
      close
   else
   r_y(k) = c(2).*yscale; % y coordinate of red ball
   end
end

l=r_y(1:2:length(r_y))'
s=r_y(2:2:length(r_y))'
```

10. Code for Simulink for Arduino (Control Board) in MATLAB. Also plots the linear potentiometer readings for the linear actuators.

```
clear, close all, clc
ST=0.01;
period = 2.2;
Vs = 5;
Pmax = 30;
Pmin = -30;

pulse = 50; % pulse <50 makes the push longer than the pull
GoalTop=1023;
GoalBot=0;
GoalTop1=1023;
GoalBot1=0;
GoalTop2=1023;
GoalBot2=0;

kp = 0.9;
ki=0;
kd = 0.004;
kp1 = 0.9;
ki1=0;
kd1 = 0.004;

rtwbuild('position_based_controller_withequation');
```

```
sim('arduino_test_controller_model');

Position.Time=Position.Time./6;
Position1.Time=Position1.Time./6;
Position2.Time=Position2.Time./6;
Speed.Time=Speed.Time./6;
Speed1.Time=Speed1.Time./6;
Speed1.Time=Speed1.Time./6;
Speed2.Time=Speed2.Time./6;

figure;
subplot(2,1,1)
hold on
plot(Position,'b-')
ylabel('Position','FontSize',20)
xlabel('Time (s)','FontSize',20)
hold on
fplot(@(x) GoalBot, [0,10],'k--')
fplot(@(x) GoalTop, [0,10],'k--')
subplot(2,1,2)
hold on
plot(Speed,'r-')
ylabel('Speed Signal (V)','FontSize',20)
xlabel('Time (s)','FontSize',20)

figure;
```

```
subplot(2,1,1)
hold on
plot(Position1,'b-')
ylabel('Position')
xlabel('Time (s)')
fplot(@(x) GoalBot1,[0,10],'k--')
fplot(@(x) GoalTop1,[0,10],'k--')
subplot(2,1,2)
plot(Speed1,'r-')
ylabel('Speed Signal')
xlabel('Time (s)')

figure;
subplot(2,1,1)
hold on
plot(Position2,'b.')
fplot(@(x) GoalBot2,[0,10],'k--')
fplot(@(x) GoalTop2,[0,10],'k--')
subplot(2,1,2)
plot(Speed2,'r-')

A= Position.Data;
B= Speed.Data;
fileID = fopen('position.txt','w');
printf(fileID,'%6.2f\n', A);
fclose(fileID);
```

```
fileID = fopen('speed.txt','w');
fprintf(fileID,'%6.2f\n', B);
fclose(fileID);

A1= Position1.Data;
B1= Speed1.Data;
fileID = fopen('position2.txt','w');
fprintf(fileID,'%6.2f\n', A1);
fclose(fileID);
fileID = fopen('speed2.txt','w');
fprintf(fileID,'%6.2f\n', B1);
fclose(fileID);

A2= Position2.Data;
B2= Speed2.Data;
fileID = fopen('position3.txt','w');
fprintf(fileID,'%6.2f\n', A2);
fclose(fileID);
fileID = fopen('speed3.txt','w');
fprintf(fileID,'%6.2f\n', B2);
fclose(fileID);
```

References

[1] J. M. Birch and M. H. Dickinson, "The influence of wing-wake interactions on the production of aerodynamic forces in flapping flight," *Journal of Experimental Biology* **206**, 2003.

[2] 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* **22**(1), 2012.

[3] Y. Cai, S. Bi, L. Zhang, and J. Gao, "Design of a robotic fish propelled by oscillating flexible pectoral foils," *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[4] G. V. Lauder, E. J. Anderson, J. Tangorra, and P. G. Madden, "Fish biorobotics: kinematics and hydrodynamics of self-propulsion," *Journal of Experimental Biology* **210**, 2007.

[5] C. Zhou and K. H. Low, "Design and Locomotion Control of a Biomimetic Underwater Vehicle With Fin Propulsion," *IEEE/ASME Transactions on Mechatronics* **17**, 2012.

[6] X. Gong, K. Yang, J. Xie, Y. Wang, P. Kulkarni, A. S. Hobbs, and A. D. Mazzeo, "Rotary Actuators Based on Pneumatically Driven Elastomeric Structures," *Advanced Materials* **28**(34), 2016.

[7] J. Xie, J. B. McGovern, R. Patel, W. Kim, S. Dutt, and A. D. Mazzeo, "Elastomeric Actuators on Airfoils for Aerodynamic Control of Lift and Drag," *Advanced Engineering Materials* **17**(7), 2015.

[8] S. P. Colin and J. H. Costello, "Morphology, swimming performance and propulsive mode of six co-occurring hydromedusae," *Journal of Experimental Biology* **205**, 2002.

[9] J. O. Dabiri, S. Colin, J. H. Costello, and M. Gharib, "Flow patterns Generated by Oblate Medusan Jellyfish: Field Measurements and Laboratory Analyses," *Journal of Experimental Biology* **208**, 2005.

[10] S. P. Colin and J. H. Costello, "Morphology, fluid motion and predation by the scyphomedusa *Aurelia aurita*," *Journal of Experimental Biology* **121,** 1994.

[11] S. P. Colin, J. H. Costello, and E. Klos, "*In situ* swimming and feeding behavior of eight co-occurring hydromedusae," *Journal of Experimental Biology* **253**, 2003.

[12] B. J. Gemmell, J. H. Costello, S. P. Colin, C. J. Stward, J. O. Dabiri, D. Tafti, and S. Priya, "Passive energy recapture in jellyfish contributes to propulsive advantage over other metazoans," *PNAS* **110**(44), 2013.

[13] A. Villanueva, C. Smith, and S. Priya, "A biomimetic robotic jellyfish (Robojelly) actuated by shape memory alloy composite actuators," *Bioinspiration & Biomimetics* **6**(3), 2011.

[14] J. S. Najem, *Design and Development of a Bio-inspired Robotic Jellyfish that Features Ionic Polymer Metal Composites Actuators.* MS Thesis. Virginia Polytechnic Institute and State University, 2012. Web. 5 Sept. 2017.

[15] K. Yang, *A Hydraulic, Jellyfish-Based Soft Robot.* MS Thesis. Rutgers University. 2015. Web. 4 Sept. 2017.

[16] 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," *PNAS* **108**(51), 2011.

[17] 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* **6**(8790), 2015.

[18] J. H. Costello, S. P. Colin, and J. O. Dabiri, "Medusan morphospace: phylogenetic constraints, biomechanical solutions, and ecological consequences," *Invertebrate Biology* **127**(3), 2008.

[19] W. Thielicke and E. J. Stamhuis, "PIVlab – Towards User-friendly, Affordable and Accurate Digital Particle Image Velocimetry in MATLAB," *Journal of Open Research Software* **2**(1), 2014.