Larvacean Robotics: A biomimetic approach to the study of enhanced fluid transport

Karla Haiat Sasson, University of Rhode Island

Mentor: Kakani Katija

Summer 2018

Keywords: Biomimetics, Giant Larvacean, Robotics, Computer vision

ABSTRACT

Giant larvaceans of genus *Bathochordaeus* are abundant zooplanktonic animals. They build complex structures out of mucus and use their tail motion to feed water into these filtering structures, which are used to concentrate particles for ingestion. Two species of larvaceans, *B. mcnutti* and *B. stygius,* were analyzed for this paper. The volume flow rate of *B. mcnutti* has been shown to be higher than *B. stygius* (K. Katija et al, 2017a). An observation that tail locomotion is different for these two species, with *B. mcnutti* appearing to have a hinge between its flexible tail and a stiffer tail end (A. Baumer and Katija, 2015), motivates an interest in exploring these kinematic differences to understand if the function of the hinge observed in *B. mcnutti* is correlated to the higher flow rate in this species. A robotic system was designed to test these differences. The system relied on passive actuation of a soft robot by a servo motor, and computer vision to analyze the kinematics of the tails. Two tail models, actuated by the motor, were created to match the two different species' motions, aiming to achieve a hinge in one of the tail models and analyze the kinematic differences between the tail structures. A first approximation of the hinge was created by adding a stiff 635 microns thick shim stock piece at the end of one modeled tail. Additionally both modeled tails, which were made of silicone rubber, included a tab at their base for actuation of the whole tail. Kinematic values were then identified from live footage of *B. mcnutti,* and compared to the two models. Hinged and unhinged model motion was visually different, which was supported by the different acquired kinematic values between the two models.

INTRODUCTION

Larvaceans are pelagic zooplankton found in the world's oceans. They are basal chordates with a body morphology composed of a tail and a trunk (or head), and have been found to be second to copepods in zooplankton abundance (C. Jaspers et al, 2009). Larvaceans build complex mucus structures (referred to as "houses" since they reside inside them), which are used to concentrate particles for larvaceans to feed on. Giant larvaceans of the genus *Bathochordaeus* are larger than other species, reaching up to 10 cm in length (W.M. Hamner et al, 1992). Their houses can even exceed 1 m in length. The houses consist of and an inner layer of finer mesh where particles can be filtered and concentrated to the appropriate size for ingestion, and an outer structure, the function of which is still subject of debate. Water is forced into the inner filter by the beating larvacean tail. After passing through the tail chamber, water and suspended particles flow through two fluted structures. From these structures, a buccal tube leads particles to the animal's mouth (K. Katija et al, 2017a). When the filter structure becomes clogged, the larvacean discards the house and builds a new one. This has important ecological implications, since the discarded houses are full of carbon and due to their size, increase the speed of sinking particles to the seafloor (B. H. Robison et al, 2005). Larvacean houses are estimated to constitute up to 1/3 of the carbon flux to the seafloor in Monterey Bay (B. H. Robison et al, 2005), which has a significant impact on the biological pump.

Two larvacean species found in Monterey Bay include *Bathochordaeus stygius* and *Bathochordaeus mcnutti*. Through analysis of Monterey Bay Aquarium Research Institute's Remotely Operated Vehicle (ROV) footage of these two species, it can be observed that the tail movement of *Bathochordaeus stygius* resembles a traveling wave (A. Baumer and Katija, 2015). However, this contrasts with *mcnutti,* whose tail is observed to have a stiffer section that bends 2/3 of the way down the tail (Figure 1). This stiff end section appears to be held at an angle, resembling a hinge. The function of this hinge is unclear.

While pumping, the tail shape of *B. mcnutti* is composed of a flexible base with a stiff end. Other flying and swimming animals have been shown to bend their propulsive appendages within a predictable range of characteristic motions (Lucas et al, 2014). These characteristic motions are defined by a rigid body or appendage followed by a flexion point, after which the rest of the structure becomes flexible (Figure 2). Interestingly, *B. mcnutti* has the opposite structure. Besides the structure of the flexion point, it has been shown that that point on the

appendage or body of the animals happens at the same ratio on the body (Figure 2), which the larvacean matches. If the bending modes for swimming or flying are optimized for efficient movement, perhaps the bending modes seen in giant larvaceans indicate an optimization for enhanced pumping efficiency.

A study on the role of giant larvaceans in oceanic carbon cycling analyzed both of these species and calculated the volume flow rate of *B. mcnutti* to be higher (K. Katija et al, 2017a). A table adapted by K. Katija (Table 1) includes normalized values of this flow rate, indicating that even when normalized to the animal's size and pumping frequency, *B. mcnutti* maintains a higher flow rate. This brings up the question of whether the hinge has any effect on this difference in flow rate.

Figure 1. Tail movement of *Bathochordaeus stygius* (L) and *Bathochordaeus mcnutti* (R). Images from MBARI ROV footage.

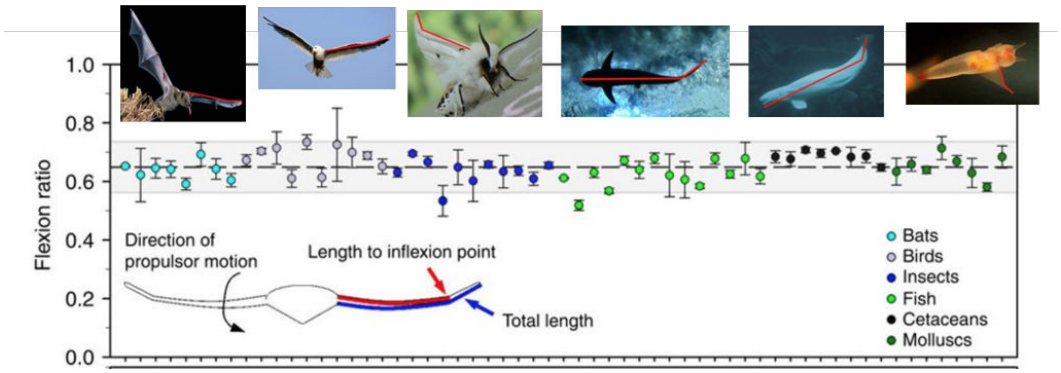

Figure 2. The position of flexion points of different taxonomic groups. (Lucas et al, 2014).

Table 1. Volume flow rate chart adapted by K. Katija from (Katija et al, 2017a). Volume flow rate was normalized using tail beat frequency and tail length, yielding a dimensionless quantity. Volume flow rate measurements were made in situ using particle image velocimetry. Swimming cycles are defined as one wave cycle and standard deviations are obtained from measurements done on all observed individuals of each species.

Species	Number of Individuals ()	Swimming Cycles ()	Tail Length (cm)	Tail Beat Frequency (1/s)	Volume Flow Rate (L/hr)	Normalized Volume Flow Rate ()
Bathochordaeus stygius	**5**	**43**	**4.9 ± 0.30**	**0.97 ± 0.02**	**27.01 ± 4.31**	**0.07 ± 0.02**
Bathochordaeus mcnutti	**3**	**10**	**6.17 ± 0.44**	**0.77 ± 0.24**	**69.39 ± 6.03**	**0.11 ± 0.02**

Figure 3 shows a microscope photo of a larvacean tail. It confirms that the end of the tail, where the hinge happens, has stiffer tissue than the rest of the tail, since cells appear to be clumped together and have thicker walls than the epithelial cells surrounding the rest of the body. Additionally, it's important to point out that the end of the tail does not contain muscle tissue. This suggests that the hinge is not due to musculature in the tail but is due to some other mechanism.

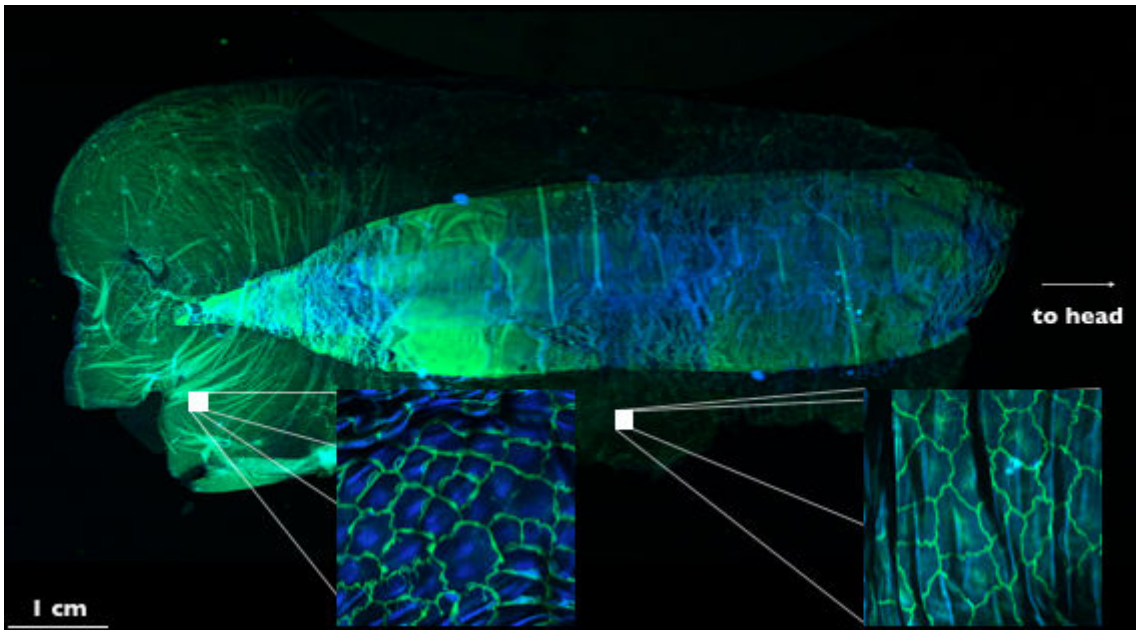

Figure 3. Microscope photo of a *B. mcnutti* tail, stained to identify internal features. Theusculature in the center of the structure, does not span all the way to the end of the tail. In the middle of the muscle tissue there is a notochord. The surrounding tissues are epithelial cells. Magnified sections show tissue at the end of the tail has thicker walls and more spatially dense cells, compared to the rest of the epithelial cells with less dense cells and thinner cell membranes. Image courtesy of Janna Nawroth (University of Hawaii).

The observation that the movement is different in the two species of *Bathochordaeus,* and that *B. mcnutti* has a higher volume flow rate, motivates an interest in the kinematic differences between the two species. Does the hinge have an effect on the flow rate? How can we recreate this motion to study associated fluid transport? Here we describe the development of a biomimetic physical model to study the kinematics of the movement for future flow measurements.

MATERIALS AND METHODS

SETUP

A system based on motor-actuated soft robotics was used to mimic the larvacean tail. The actuation of the 8 cm tail mimic consisted of a servo motor (Hitec HS-645MG) controlled by an Arduino Uno. The servo was placed in a waterproof housing, since the tail model portion was inside a 24 in L x 12 in W x 16 in H tank (Figures 4-6). The testing portion for the mimic relied on computer vision to process kinematics via Matlab R2018a.

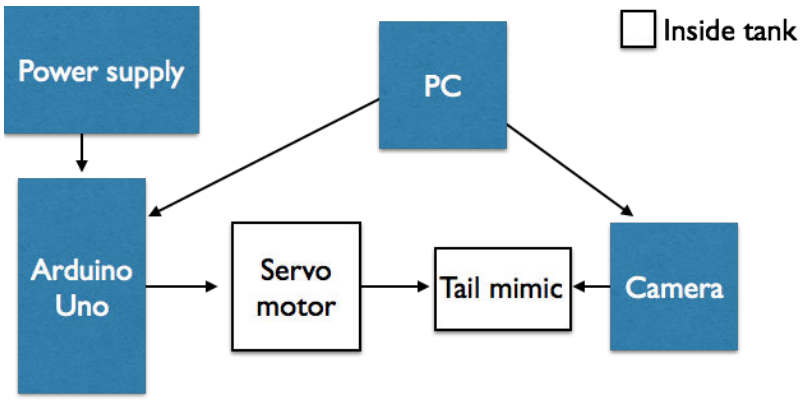

Figure 4. Diagram of system. PC provided power for Arduino and Camera. Power supply provided power to servo motor. Diagram indicates Arduino controlled servo motor, which controlled the tail mimic, which was recorded by the camera. Arduino software and camera software were used to control the Arduino and Camera respectively. Matlab was used to process the camera recordings.

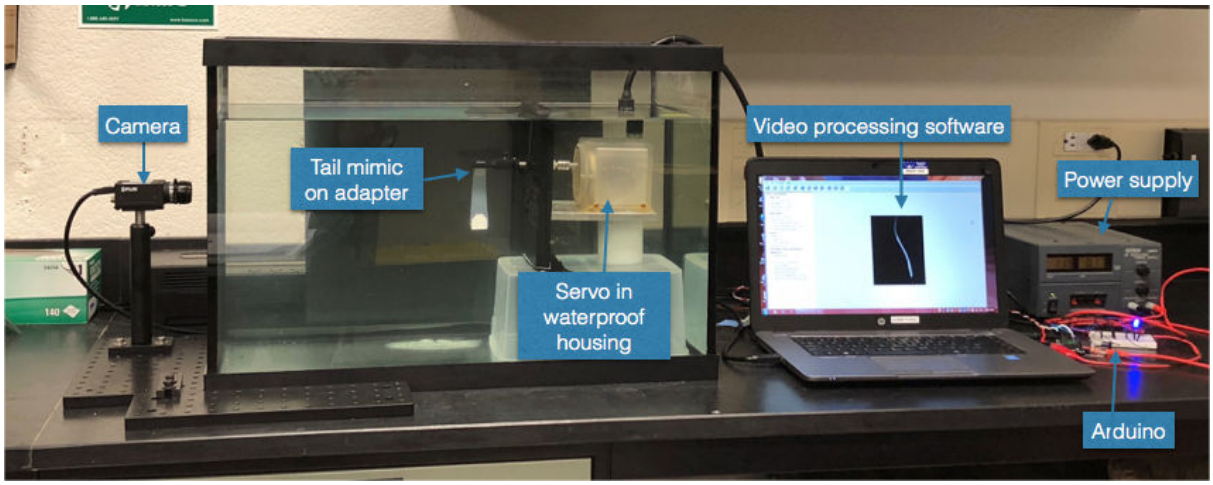

Figure 5. The system setup. Servo motor controlled by the Arduino was contained in a waterproof housing, which was attached to a shaft that actuated the tail. A CMOS camera, controlled by video processing software, was placed on the side of the tank for visual processing.

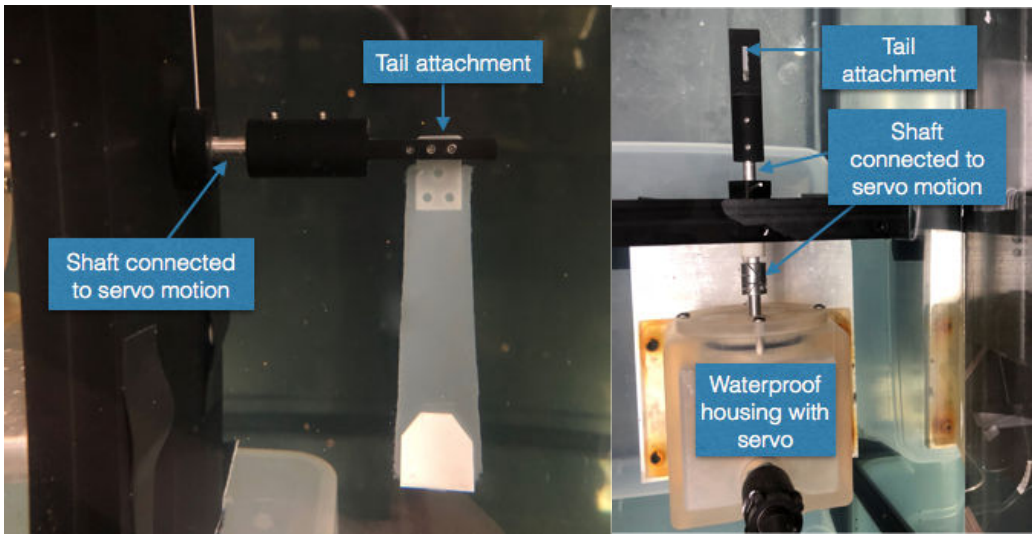

Figure 6. Front view (L) and top view (R) of tail attachment mechanism. Adapter was attached to a shaft that was moved by the servo motor. The adapter had a slot at its middle where the tail could be clamped in with set screws.

The Arduino was wired (Figure 7) and coded (Appendix 1) to incorporate a sweep mechanism with an LED indicating the forward and reverse motion to help keep track of one cycle. The sweep motion was created by specifying the extreme positions to adjust angle and incorporating delays to set the desired frequency. To calculate the delay the following simple calculations were done:

#steps = total angle * 2

delay = period in ms / # steps

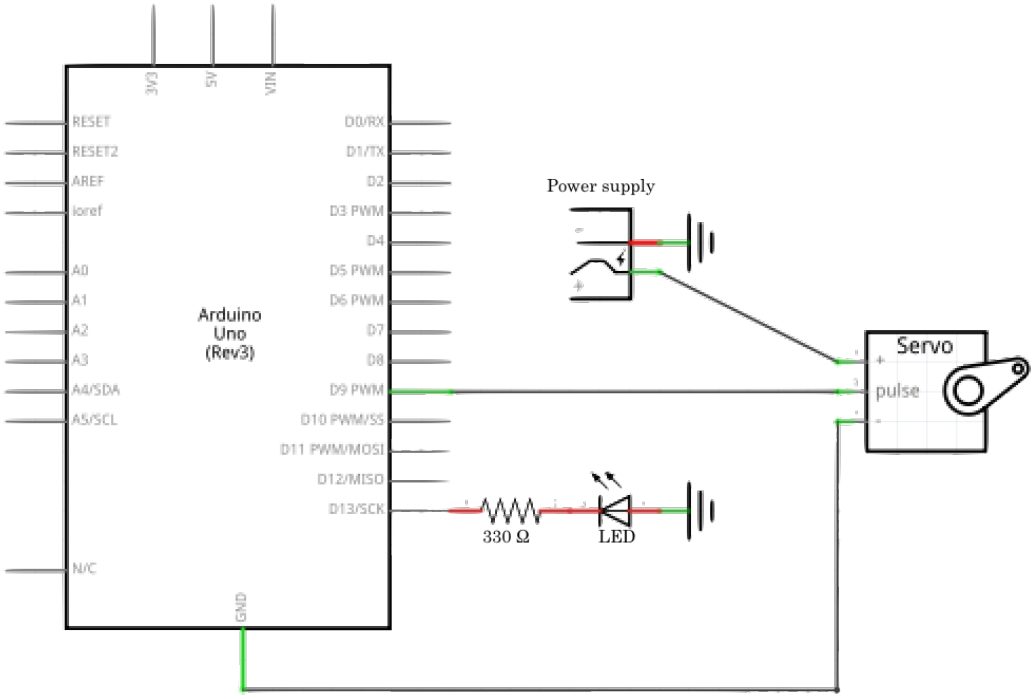

Figure 7. Circuit of the Arduino, servo motor and LED in the system.

To be able to measure the kinematics of the tail, a FLIR (PointGrey) Grasshopper3 camera controlled by FlyCap 2.0 software was placed next to the tank (Figure 5). The camera was a 5 megapixel, monochrome CMOS camera recording at up to 75 fps (set to 55 fps to avoid flicker due to fluorescent lighting). Attention was paid to maintaining the same exposure settings and framing for every recording so videos could be compared and easily processed using the same automated algorithm in Matlab. It was also important for the base of the tail to be placed roughly in the middle of the tank, with enough space from the bottom and the walls to avoid interference of these walls on the flow. The tail base was placed halfway up the tank, and at least 4 in away from the walls. Figure 8 shows interference of the flow when having the tail too close to the bottom.

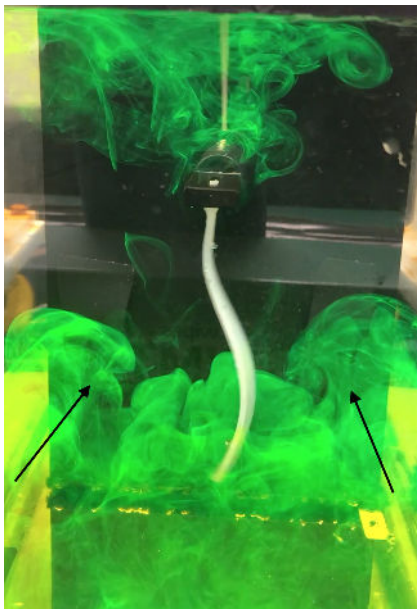

Figure 8. Interference of flow in a small tank. Green dye was injected at the base of the tail, the movement of it shows the flow bounced off the bottom of the tank and the side walls making its way back up to the tail.

TAIL DESIGN

A major challenge of the system was the matching of an active movement (muscular tail movement) by using a passive system, in which the entire tail was driven by the base of the tail.

To create a biomimic of the tail, various characteristics of the real tail were taken into account:

1. The end of the tail has stiffer tissue.
2. Notochord connects the motion from the head to the end of the tail.
3. There is a connection between the head and notochord.

A 3D printed mold was designed to incorporate different tail designs to match these characteristics (Figure 9). This mold allowed for different thicknesses of shim stock plastic to be placed at different parts of the mold. Additionally, a 3D printed tab (Figure 9d) was created to fit into the mold. The thinner part of the tab (1.27 mm) went into the tail model, and the thicker side (2.30 mm), which was inside a gap in the mold (under Figure 9a) was used to fit into the adapter piece on the shaft. After the materials were placed where desired, Dragon Skin® FX-Pro, a silicon rubber material, was prepared, degassed to get rid of bubbles and poured into the mold. Before mixing in the dragon skin, a small amount of material thinner could be incorporated into the mix. A piece of metal was then placed on top of the

mold between the material on the level of A and level of B on Figure 9. This allowed the material to be evened out. The material was left to dry for 1 hour and then slowly removed from the mold. Any extra material was cut out from the tail.

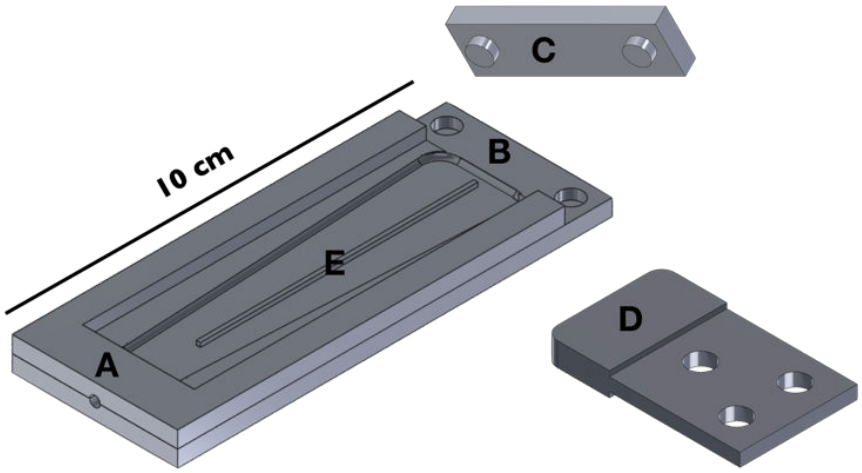

Figure 9. Tail mold design. Tab D can be placed on the gap under part A to allow the thinner part to fit into the tail design and the thicker part to be the piece for the adapter. Shim stock can be placed between B and C and secured by joining the pieces together. The standoff at part E can be used to place material in the middle of the tail.

KINEMATIC ANALYSIS

Data from a single sequence ROV video of *B. mcnutti* (Figure 10), manually traced per frame (A. Baumer and Katija, 2015), was used as comparison for the model kinematics. Simple wave measurements were sought out as values for a preliminary approximation for this comparison. These values, previously defined in the work by Baumer et al, included wavelength, amplitude, frequency and angle (Figure 11).

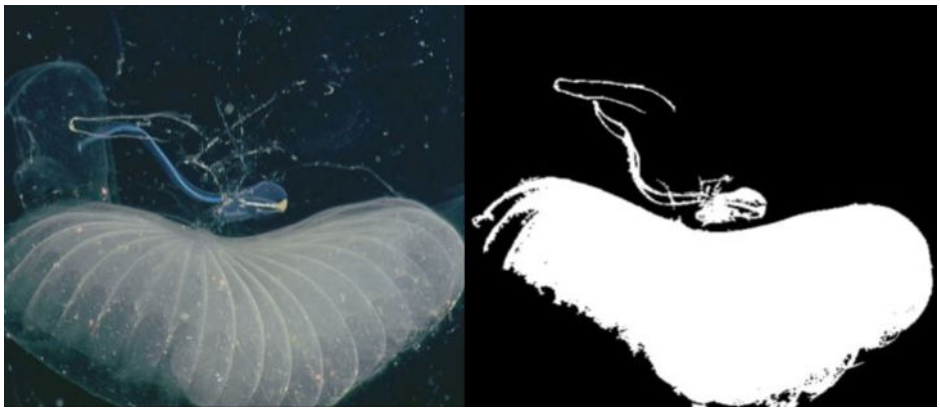

Figure 10. Example of ROV footage image processing taken from (A. Baumer and Katija, 2015). Image shows video frame of organism in house (A) and boundary detection of animal (B). Since the detection of the tail was complicated because of the house, the tail was manually traced for analysis.

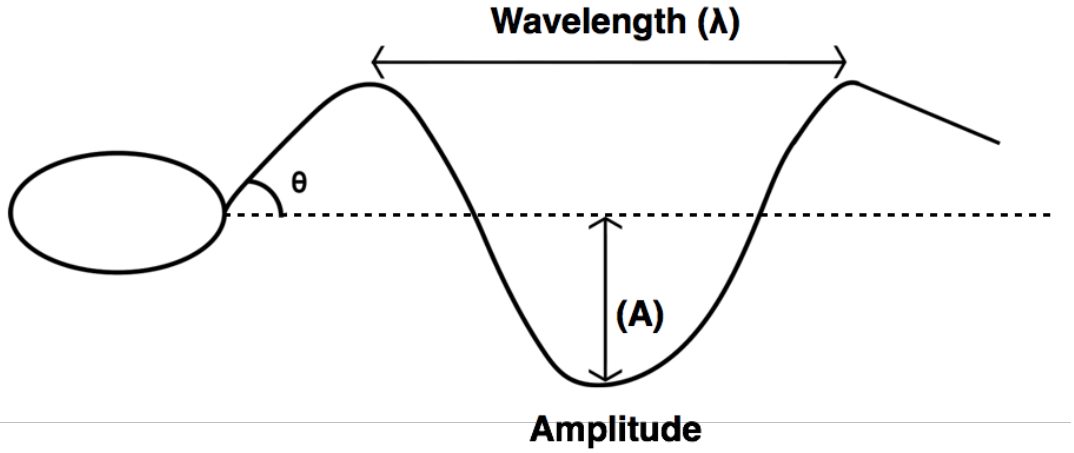

Figure 11. Diagram of simple wave metrics used for kinematic analysis. Amplitude was defined as maximum value of maximum peak of the cycle (and minimum value of minimum peak of the cycle), wavelength was defined as the distance between the minimum and maximum peak doubled, and the angle was defined as the maximum from the base that encompasses the entire tail. Diagram by Alexa Baumer.

An algorithm was written in Matlab (Appendix 2) to extract these kinematic values from the pre-processed tail traces. The algorithm shifted the curve so the base of the tail was at the origin of the coordinate system. Then, the endpoint of each frame was identified and a median vertical endpoint position was found from the values of all the frames. A rotation was then applied to the data to make the midpoint of the endpoints the center of the movement (Figure 12). To fit some parameters to the curve, the following assumptions were made:

1. The horizontal distance from minima to maxima in a frame is half a wavelength
2. Calculations are done as if the wave continues for at least a wavelength
3. X values after hinge is observed are ignored (Figure 13)

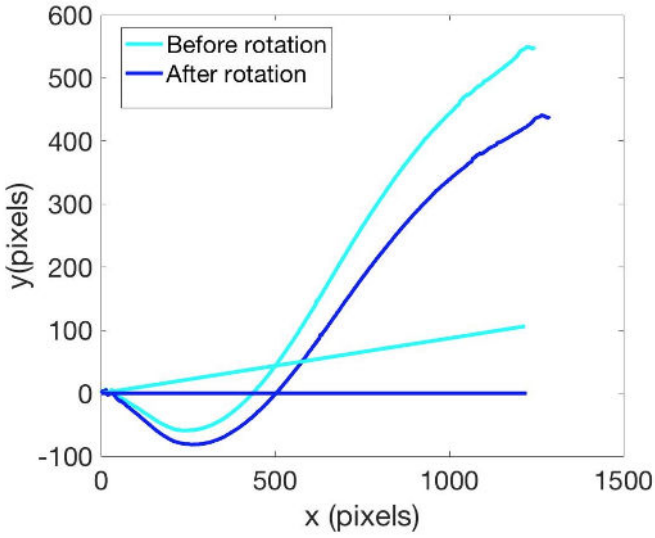

Figure 12. Data rotation. Tail curve was rotated by identifying median value of the tail endpoints and shifting the curve to have the median value be the middle of the endtail motion. This way, the tail movement is a little more symmetrical.

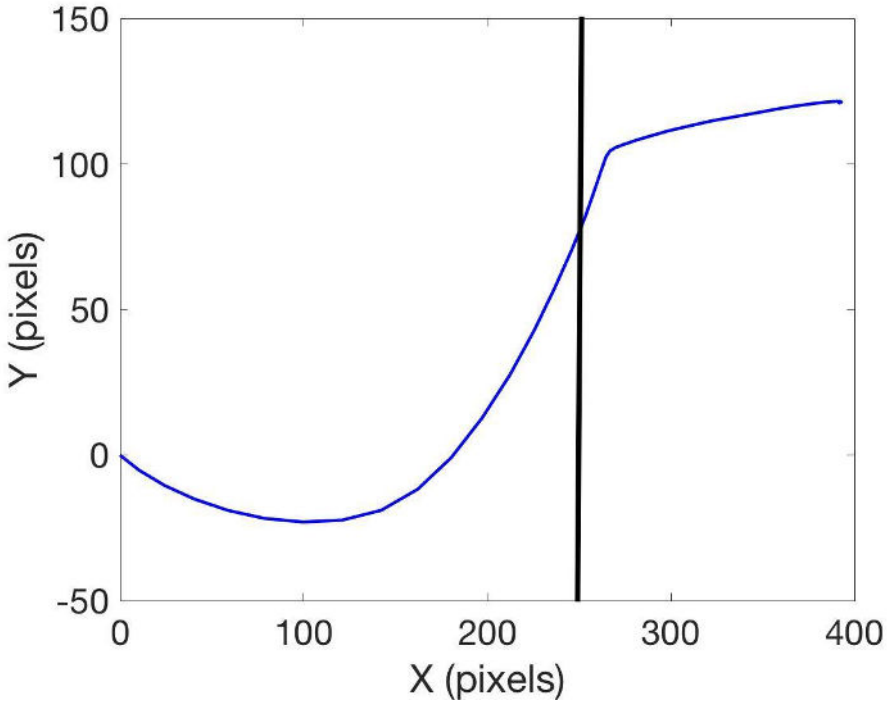

Figure 13. Processed frame at which the hinge is visible. In this case all the values higher than 250 are excluded from the kinematic analysis. Full length of curve was 485.4 pixels, while length without the hinge was 283.5 pixels.

For the amplitude calculations, the x location of the hinge was visually identified (Figure 13) and all values higher than that x value were excluded from analysis. The maximum and minimum were then computed for every frame and then plotted together as a function of

video frame (Figure 14). The shape of this maximum Y values vs video frame plot gave the locations of the maxima and minima of a swimming cycle. Before identifying the peaks, the curve was smoothed by identifying the moving average of the line to reduce periodic signals. By indexing the frame at which the peaks happened, the data from the original curve is used to find the maximum and minimum values of that frame.

The maximum amplitudes in both directions (below and above the x axis) were obtained per cycle. For the final data comparison, an average was taken between the positive and negative direction amplitudes, and then an average of the swimming cycles was taken to obtain a final number.

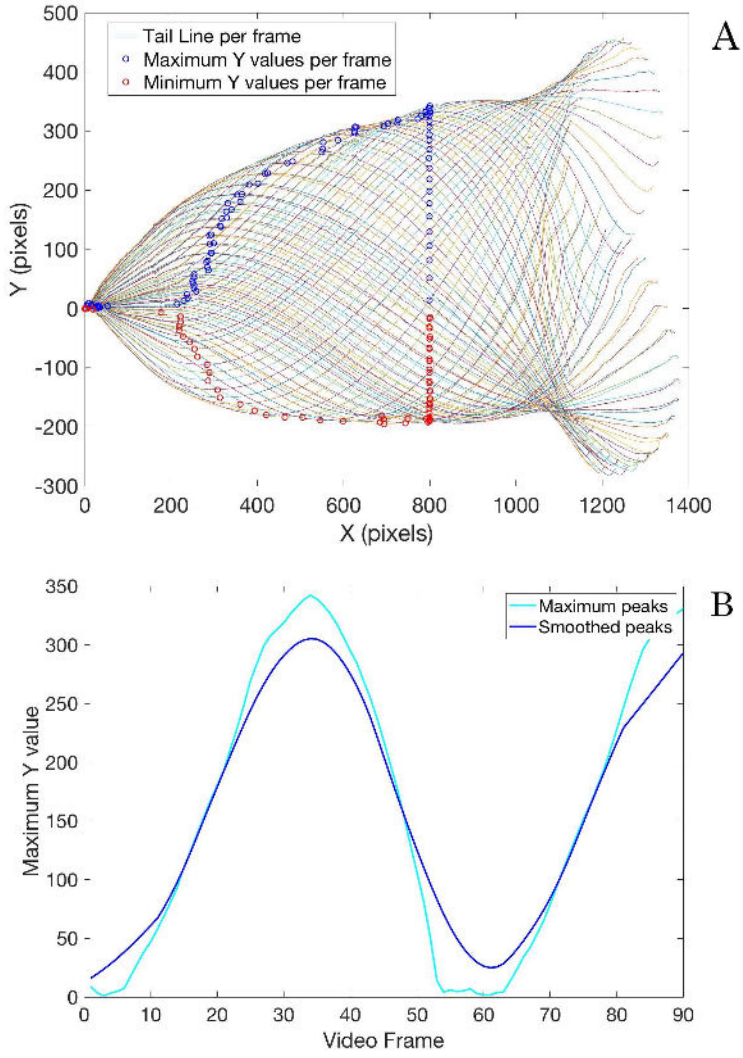

Figure 14. Minimum and maximum data processing. Tail from a tail model video was traced at every frame and a maximum and minimum was identified (A). The maximum and minimum values were then plotted separately, yielding a curve with Y values per frame (B), which was later smoothed with a simple smooth curve matlab function to find the frames at which the peaks happened. This curve allows identification of the maximum Y value per swimming cycle. In this case only a cycle and a half was observed.

The maximum angle from the base that encompasses the entire tail was calculated per frame as well. This was done by using the starting point and every point on the curve to get Δy and Δx values and finding the $\tan^{-1}(\Delta y/\Delta x)$ to get the angle at every point. The maximum angle was then identified.

The wavelength was a little more complicated since the wave is not a simple sine wave. As an approximation, the maxima and minima were determined for the frames at which the maximum amplitudes were found. The x distance between the minimum point and maximum point were then calculated. In the case of a sinusoidal wave, the distance between the minima to the maxima spans ½ of a wavelength, so the wavelength was multiplied by 2.

To make the values comparable to other videos, the data was non-dimensionalized. This was done by finding the total curved length of the curve (in pixels) per frame, and averaging this length; and then dividing the kinematic values by this length; eliminating the factor of length in potential calculation differences.

For the data from the model, the videos from the CMOS camera were input into Matlab and an image processing algorithm (Appendix 3) is applied (Figure 15). This algorithm rotated the video, monochromized it, converted it to binary and eliminated any extra objects that may have been in the video besides the tail. After the video was converted to binary data, the median of the y values at each x position was found and a tail line was traced every frame. The resulting larvacean model curve was treated as the equivalent of the manually traced ROV video, and analyzed with the same algorithm described above.

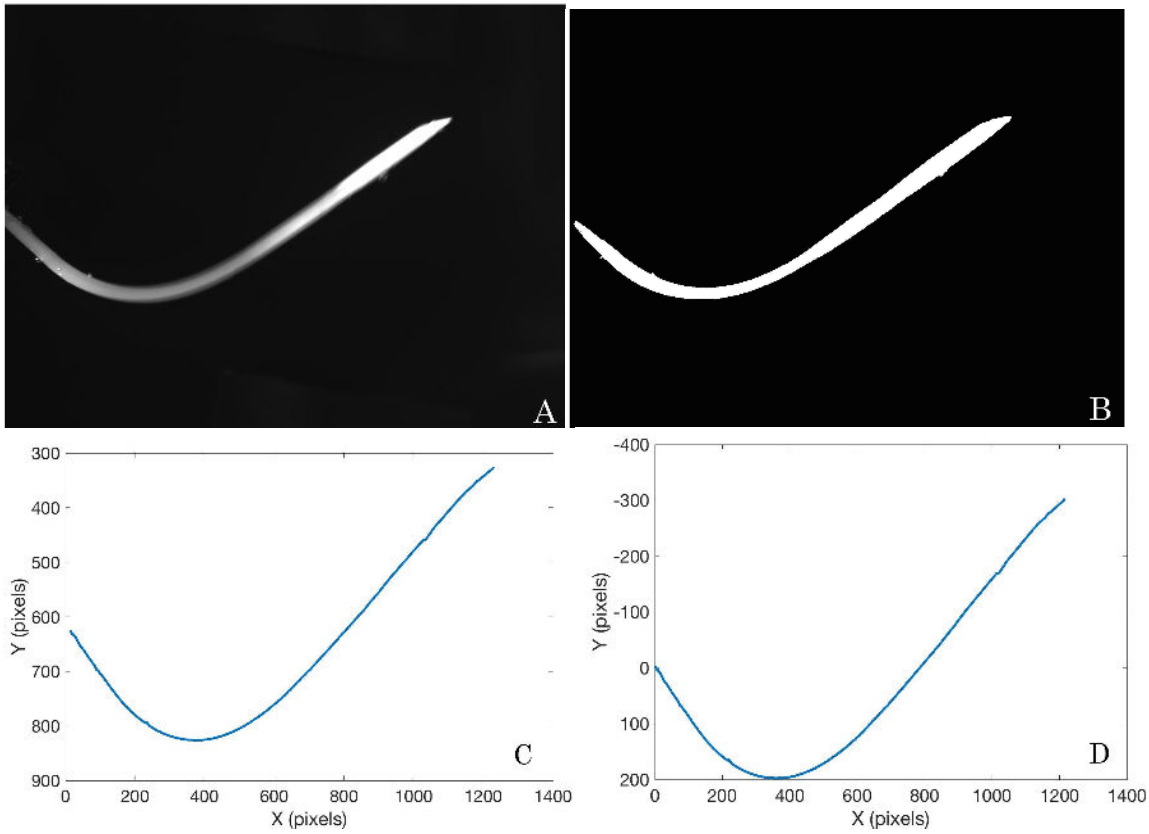

Figure 15. Image processing algorithm applied to model videos. Grayscale image (A), Binary image (B), Tail line traced (C), Curve is shifted to origin (D).

RESULTS

Over 20 different tail designs were tested on the setup (Figure 16). The designs ranged from different thicknesses of tails, different amounts of material thinner, different shapes and thicknesses of plastic shim stock embedded and different placements of the shim stock in the tail. Some other design choices were designed to fit the different characteristics of the real tail. These designs include the incorporation of a "notochord" made out of shim stock in the middle of the silicone material, which resulted in making the tail too stiff to allow movement, or the shim stock being to thin to have any effect on the movement. Including a notochord along with stiff material at the end, which resulted in a delay of the tip of the tail, but not a good curve for the same reasons as the previously mentioned design. A notochord with no material at the end of the tail (a flexible end), which was unsuccessful, since the larvacean tail has an opposite mechanism (flexible tail base, stiff end). And reducing the amount of silicon rubber material at the end of the tail right before the stiff end, which created a hinge, but the method also hinged on a tail with no shim stock at the end and therefore was not an accurate model to compare *stygius* and *mcnutti*.

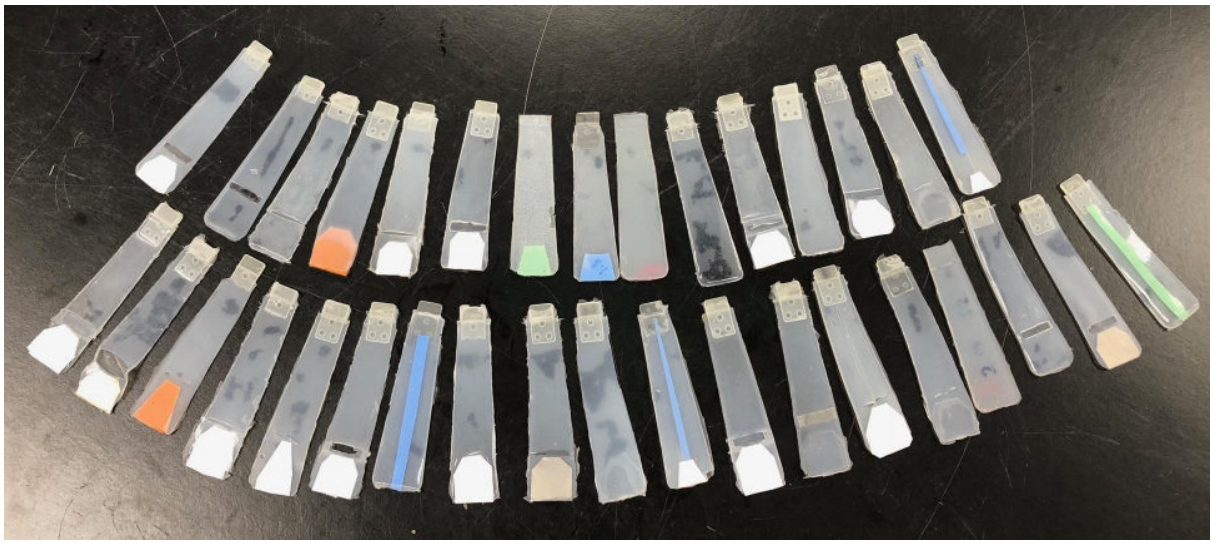

Figure 16. Various tail designs that were made and tested, including incorporation of a notochord design, different shapes and thicknesses of material at the end of the tail, and reduction of silicon rubber before the stiff material at the end of the tail.

Ultimately, a design for the two different motions was more successful than the rest. Two tails of similar design but one with the incorporation of a hinge (Figure 17) had some important characteristics. The design allowed for the mimics to move flexibly like a tail, and allowed one of them to have a hinge while the other didn't. The tails were 2 mm in thickness, had a 3D printed tab placed in between the layers of material, and both had thinner (a non-reactive silicone fluid that will lower the mixed viscosity of the silicone rubber) incorporated into the dragon skin. It was important for the weight of the thinner added to be 10% of the total weight of the two mixed parts of dragon skin to make the silicone rubber flexible enough to actuate like the flexible larvacean tail. The tab was particularly important to successfully propagate the movement of the driven end to the end of the tail. A notochord design was ultimately not utilized since it was not a good method to mimic the movement. This is because the animal has muscle tissue around the notochord, eliminating the possible stiffening of the tail from the thickness in the area, which the model does not have. Finally, a 635 micron thick shim stock piece was placed at the end of one of the tails. It was important that the shape of the material was closer to the rim of the tail at the bottom of the tail and thinner as it goes up the tail to allow the tail to be heavier at the end and also allow for a more smooth transition from the flexible material to the stiff material. Additionally, the material needed to be evenly placed in the tail, since any tilt would create a slanted motion instead of a hinge. The placement of the material had to be at the desired vertical place for the hinge. The combination of the tab, the material shape and thickness, and the thinner in the material allowed the tail to be flexible like the larvacean tail and also allowed a hinge to be created for one of the tails and not the other.

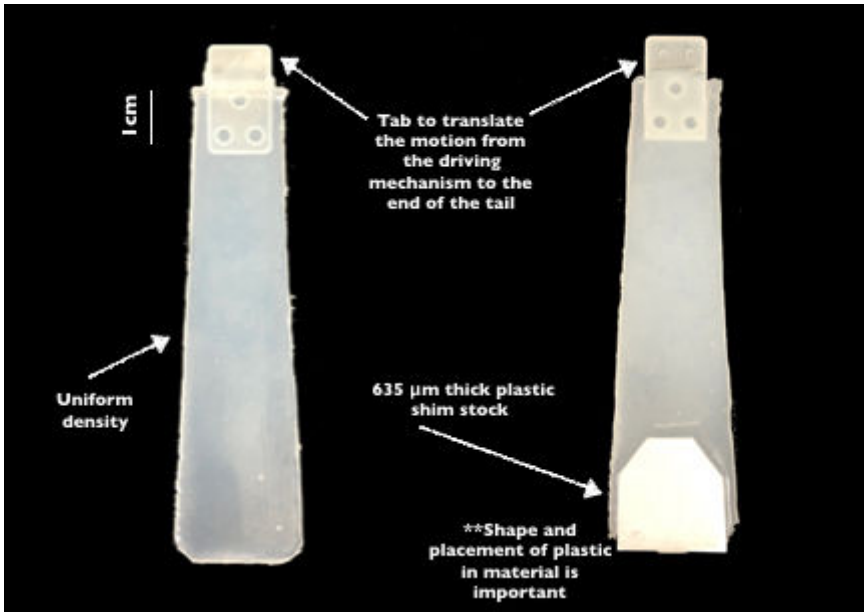

Figure 17. The final tail design for *Bathochordaeus stygius* (L) and *Bathochordaeus mcnutti* (R).

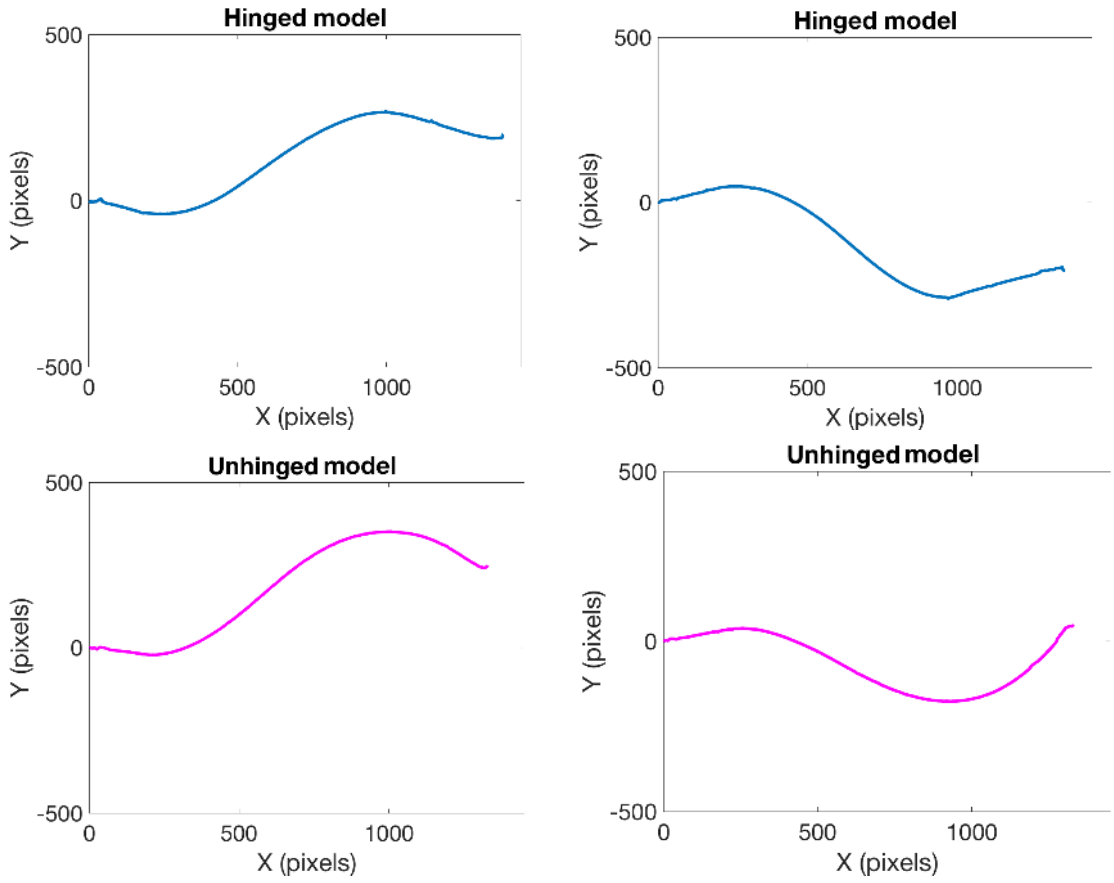

Figure 18. Digital detection of hinged model (Top) and unhinged model (Bottom).

Table 2. Table comparing kinematic values of videos of the models. Same colored rows indicate the data comes from the same video (but different swimming cycles). Three videos were analyzed for the hinged model (actuated at different frequencies but same angle) and two were analyzed for the unhinged model. Error values come from the standard deviation of averaged values.

	Parameters set to Arduino (average length of cycle and angle)	Half angle (positive and negative directions)	Amplitude (positive)	Amplitude (negative)	Wavelength (positive)	Wavelength (negative)
Hinged Cycle 1	810 ms 82.5	29.2676 ± 1.7788 28.2622 ± 1.0959	0.2654 ± 1.236e-4	0.3312 ± 0.003	0.6374 ± 0.009	0.7225 ± 0.025
Hinged Cycle 2	810 ms 82.5	29.2676 ± 1.7788 28.2622 ± 1.0959	0.2652± 1.236e-4	0.3252 ± 0.003	0.6561± 0.009	0.6731± 0.025
Hinged Cycle 1	1030 ms 82.5	29.2791 ± 1.1228 28.0791 ± 1.7204	0.2309 ± 0.003	0.3762 ± 0.006	0.5220± 0.002	0.6040± 0.007
Hinged Cycle 2	1030 ms 82.5	29.2791 ± 1.1228 28.0791 ± 1.7204	0.2259 ± 0.003	0.3634± 0.006	0.5186± 0.002	0.5780± 0.007
Hinged Cycle 1	1250 ms 82.5	28.4873 ± 0.9941 28.9102 ± 1.6742	0.2340 ± 0.002	0.3466 ± 8.037e-04	0.5978 ± 0.019	0.5825 ± 0.008
Hinged Cycle 2	1250 ms 82.5	28.4873 ± 0.9941 28.9102 ± 1.6742	0.2302 ± 0.002	0.3450 ± 8.037e-04	0.5978 ± 0.019	0.5825 ± 0.008
Unhinged Cycle 1	1030 ms 82.5	38.2210 ± 1.7745	0.1458 ± 5.862e-04	0.2120 ± 0.002	0.3885± 0.005	0.5025 ± 0.009

		28.0297 ± 1.5372				
Unhinged Cycle 2	1030 ms 82.5	38.2210 ± 1.7745 28.0297 ± 1.5372	0.1470± 5.862e-04	0.2152 ± 0.002	0.3779 ± 0.005	0.4840 ± 0.009
Unhinged Cycle 1	1250 ms 82.5	33.6373 ± 3.1963 27.5336 ± 1.1315	0.1872 ± 8.733e-04	0.1715 ± 0.002	0.5322 ± 0.003	0.5234 ± 0.003
Unhinged Cycle 2	1250 ms 82.5	33.6373 ± 3.1963 27.5336 ± 1.1315	0.1889 ± 8.733e-04	0.1747 ± 0.002	0.5272 ± 0.003	0.5284 ± 0.003

Table 3. Table of comparison between live organism video, and hinged and unhinged model videos. Values were normalized using tail length in pixels, eliminating effect of field of view differences in the video. To simplify comparison between the footage, only videos with length of cycle of 1.030 seconds are displayed below. The values presented below are averaged values of data with length of cycle of 1.030 seconds, displayed in Table 2. The values are averaged across swimming cycles and from positive and negative positions. Error values come from the standard deviation of averaged values from Table 2.

Video	Maximum Angle (°)	Maximum Amplitude, A	Wavelength, λ	Average length of cycle (sec)	Swimming cycles
B. mcnutti	36.63 ± 2.990	0.246± 0.016	0.801 ± 0.105	1.030 ± 0.220	2
Hinged model	28.68 ± 0.600	0.299 ± 0.041	0.556 ± 0.021	1.030	2
Unhinged model	33.13 ± 5.090	0.180 ± 0.019	0.438 ± 0.032	1.030	2

Two tail models (Figure 17) were created to match the differing motions in two species of Giant larvacean. Figure 18, showing frames of the digital detection of the two models

created, displays two different motions achieved with the models. It can be observed that the bottom (unhinged) model has a rounded shape to the movement, while the top (hinged) seems to have a straight end, creating a hinge. The method of passive actuation was successful in recreating this movement in this first iteration of the design.

To analyze the kinematics of these models, 6 different videos were recorded. For the hinged model, 3 videos were recorded with different frequencies set to the Arduino. The servo frequencies applied to these videos were taken from the average length of cycle previously defined in the work done on ROV footage of *B. mcnutti* (A. Baumer and Katija, 2015). It is important to note that the ROV data processed was a single sequence of one animal with only 2 swimming cycles, therefore, more data will give more accurate results in the future. Three lengths of cycle were applied to the model: 810 ms, 1030 ms and 1250 ms. These values were taken from the average length of cycle and the error that had been defined on the ROV processed data. As seen in Table 2, the length of cycle had almost no effect on the amplitude or angle calculations of the hinged model, however, there seems to be a variation in the wavelength values, which we can expect since frequency and wavelength are interrelated. Amplitude values were defined by finding the maximum of the maximum peak and the minimum of the minimum peak, since that curve inverted would be the maximum too. Because of this, the "negative" amplitude is recorded with the "positive" amplitude. The wavelength calculations were done using these amplitude values, which is why we also have a positive and negative wavelength value. For the unhinged model, 1030 ms and 1250 ms were used. The 810 ms data was not used because of problems with the video detection, which failed to identify parts of the tail at some of the frames. Additionally, the length of cycle seemed to have very little effect on the kinematic values of interest on the hinged model, and the 4 cycles analyzed seemed to not vary too much in output values. The defined method was found to be sensitive to the quality of the video, requiring for the tail to be significantly lighter than the background in the footage for the kinematic detection.

Table 3, which serves to quickly depict differences between the models and the real footage, shows that all kinematic values between the different footage seem to be close to each other and at least in the same order of magnitude. The unhinged and hinged models also show clear differences in values, displaying the unhinged videos as closer in angle values to the live footage than the hinged model, but the same difference from the live footage in amplitude.

Discussion

Since the system was passively actuated, exact kinematics of the tail were difficult to achieve with this method. However, kinematic values of interest in the models were within the same order of magnitude as the live footage (Table 3). The unhinged model seems to be closer in angle to the live footage than the hinged model, but there is variation in the wavelengths for both models and the live footage.

In the end, unhinged videos had more variation in the range of kinematic values obtained from the videos, having higher standard deviations for the angle and wavelength than the hinged model (Tables 3) and a bigger difference in values obtained per cycle (Table 2). The wavelength also seems to be the kinematic parameter with the most variation in the models and even the live animal footage, an expected result since wavelength seems to be affected by the frequency of the tail, as seen on Table 2. Part of this variation is due to the frequency differences. However, this shows that the wavelength method may not be accurate enough to describe the curve. A more accurate approach for non-sinusoidal curve fitting, such as spline interpolation, may be desirable in the future for better amplitude and wavelength values.

It is important to mention that part of the reason the wavelength method is not completely accurate, besides frequency variation, is the shape of the movement of the tail (Figure 19) as well as automated identification of peaks, resulting in identification of local maxima peaks, which don't give information about the period maximums (Figure 19). This can be manually modified in this case due to the small number of swimming cycles, but it interferes with the automated calculation of wavelength. Additionally, as seen in Figure 19, the frames at which the maximum peaks happen do not always have a sine wave shape. The curve from Figure 19b, which comes from the third video frame, seems to identify a minimum and maximum that do not necessarily show the shape of a half wavelength. Therefore, this method of finding a wavelength may not be accurately describing the shape of the larvacean tail.

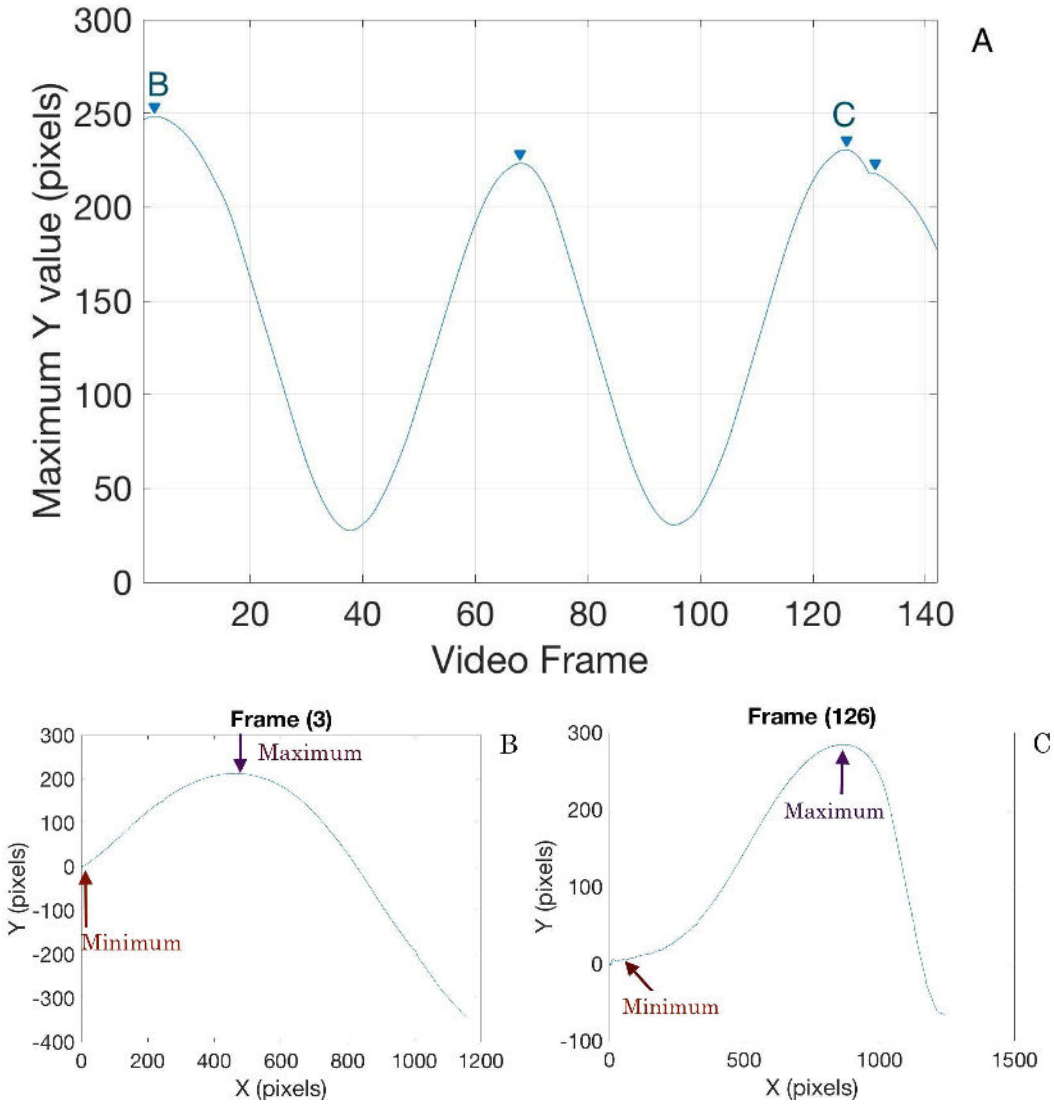

Figure 19. Issues with wavelength characterization method. (A) shows that the findpeaks function will identify all local maxima, even if our interest was in the periodic maximum peaks observed in the first two identified peaks. B and C show the difference of what the maximum amplitude values can look like in different frames. These frames were identified from two peaks of A, and show an example of how B is not necessarily giving a measurement that would work as a wavelength.

Other limitations in the analysis include the angle calculations. The average angle calculations from the models ranged from 28.7-31.9°, however, the set angle on arduino was 80°, meaning the angle calculations should have yielded 40°. This means that the calculated angle on the live footage may not be expressing the real angle of movement with the current algorithm. This is most likely due to the fact that the calculation depends a lot on the visual input. In Table 2, we can see that the angle in the negative and positive directions of the x-axis are not centered, which means the motion of the tail may be titled in the analysis and may reduce accuracy of kinematic values. Last but not least, other limitations of the system

include the servo actuation. The servo can only sweep back and forth, and ideally the motion would have a smoother transition like the live organism. Additionally, the movement of the real tail seems to move more prominently one way and then meets resistance on the way down, as if flow is getting in the way. These challenges should be explored in future iterations of the models.

CONCLUSIONS

The volume flow rate of *B. mcnutti* has been identified to be higher than *B. stygius* (K. Katija et al, 2017a), therefore a goal of the study was to explore the kinematic differences by creating biomimics of the two species. The study resulted in the creation of two models (Figure 17) that resulted in clear visual differences in motion, and the creation of a hinge in one of the model's motion. Therefore, the method of passive actuation was found to be good first approximation of the movement recreation. Results from the kinematic analysis showed amplitude and angle values of the models to be in the same order of magnitude and close proximity to real values, but most importantly showed kinematic differences between the two models. These findings are important for future investigation into the efficiency of the pumping mechanism of the giant larvacean, since *B. mcnutti* (hinged motion) has approximately a 50% higher flow pumping rate than *B. stygius* (unhinged motion). If this flow rate difference can be understood through the visual and kinematic differences between the two species, a new framework can be built from which to understand future pumping mechanism designs, as well as reaching a better understanding of the hinge function and role in the larvacean's biology.

Future considerations for the continuation of this project may include the incorporation of artificial muscle actuation if a closer to real life movement is desired from the models, a more detailed kinematic description of the tail movement for more accurate comparisons, or exploring horizontal actuation, as it is possible the material creating the hinge may be doing so because it meets resistance with the flow because of its heavier composition, and a horizontal actuation may show if the hinge in the model was due to gravity.

REFERENCES

1. A. Baumer, K. Katija, Larvacean locomotion: a kinematic investigation, MBARI Intern Report, 2015.

2. B. H. Robison, K. R. Reisenbichler, R. E. Sherlock, Giant larvacean houses: Rapid carbon transport to the deep sea floor. Science 308, 1609–1611 (2005).

3 C. Jaspers, T. G. Nielsen, J. Carstensen, R. R. Hopcroft, E. F. Møller, Metazooplankton distribution across the Southern Indian Ocean with emphasis on the role of Larvaceans. J. Plankton Res. 31, 525–540 (2009).

4. K. Katija, R. E. Sherlock, A. D. Sherman, B. H. Robison, New technology reveals the role of giant larvaceans in oceanic carbon cycling. Sci. Adv. 3, e1602374 (2017).

5. K. N. Lucas, N Johnson, W. T. Beaulieu, E. Cathcart, G. Tirrell, S.P. Colin, Bending rules for animal propulsion. Nat. Commun. 5:3293 doi: 10.1038/ncomms4293 (2014).

6. W. M. Hamner, B. H. Robison, Insitu observations of giant appendicularians in Monterey Bay. Deep Sea Res. A 39, 1299–1313 (1992).

ACKNOWLEDGMENTS

I want to thank my mentor Kakani Katija, Joost Daniels and the rest of the members of the Bioinspiration lab for the support and guidance through this project. I would not have been successful without their knowledge and resources they made available to me.

I also want to thank MBARI staff that helped me with details of my project, such as Jim Scholfield, Frank Flores, Larry Bird, Denis Klimov, Ray Thompson, Dale Graves and Brent Jones. Thank you to Brian Kieft and Rob Sherlock for letting me tag along on their vessel work on the Paragon and the RV Rachel Carson. And thank you to George Matsumoto and Linda Kuhnz for directing the intern program, particularly for making sure we had fun this summer and the great support through the whole program.

I lastly want to thank MBARI and The Packard Foundation for the funding for the research and the experience doing research with MBARI scientists and engineers.

APPENDIX 1. Arduino Code

```
 #include <Servo.h>
Servo myservo;
// Variables that are constant:
const int ledPin = 13;      //port number for LED2
const int minimum =59;      //min servo position in command degrees
const int maximum =137.5;     //max servo position in command degrees
//const int minimum = 1000;
//const int maximaum = 2000;
const int microseconds =6.3694; //3 delay counts in milliseconds
const int inc = 1;      //servo position increment, command degrees
const int inc2 = 1;      //servo position increment reverse

// Variables that change:
//int pos = 1500;             //servo position in microseconds
int pos = 0;          //servo position in command degrees

void setup() {
  // initialize the LED as an output:
  pinMode(ledPin, OUTPUT); // sets the digital pin 13 as output
  // initialize serial communication:
  Serial.begin(9600);
  // initialize servo motor baudrate
  myservo.attach(9,544,2400);
  //attach (pin,min,max)
  // myservo.writeMicroseconds(minimum);
  myservo.write(70); //80
  delayMicroseconds(100000); //10000
 }

void loop() {
  //Servo sweep:
digitalWrite(ledPin, HIGH);   //LED ON positive increment loop
  for(pos = minimum; pos <= maximum; pos=pos+inc) {
//    myservo.write(166);//165
      myservo.write(pos);
//    myservo.writeMicroseconds(1000);
    delay(microseconds);
}
//
// myservo.write(1);
// delay(100);
// myservo.write(166);
// delay(1000);
//
 digitalWrite(ledPin, LOW);  //LED OFF on negative increment loop
   for(pos = maximum; pos >= minimum; pos=pos-inc2) {
     myservo.write(pos);
   //     myservo.write(pos);
   //  myservo.writeMicroseconds(2000);
      delay(microseconds);
   }
```

}

APPENDIX 2. Matlab analysis of ROV video footage

```
close all
clear all
%% Find endpoint
load('batho1_inhouse_tail_frontedge_rotated.mat')
%load('batho5_freeswimming_tail_frontedge.mat')
for n=1:length(tail_boundary)
   midline=tail_boundary(n).rot_boundary_data;
   %midline=tail_boundary(n).boundary_data;
   endpoint(n,1:2)=midline(end,:);
end

med=median(endpoint);
endptx=med(1,1);
endpty=med(1,2);

%% Rotation
for n=1:length(tail_boundary)
   midline=tail_boundary(n).rot_boundary_data;
   %midline=tail_boundary(n).boundary_data; %free swimming data
   ys=midline(1,2);
   xs=midline(1,1);
   midlinex=midline(:,1)-xs;
   midliney=midline(:,2)-ys;
   midline1=[midlinex,midliney];
   startptx=midline1(1,1);
   startpty=midline1(1,2);
   x =linspace(startptx,endptx,300);
   y =linspace(startpty,endpty,300);
   m=[x;y];
   medianline=transpose(m);
   %% Amplitude
   %Rotation of curve to make median endpoint the median of the curve
   endx=x(1,end);
   endy=y(1,end);
   deltay=(endy-startpty);
   deltax=(endx-startptx);
   A=atan(deltay/deltax);
   Rotated=[cos(A),-sin(A);sin(A),cos(A)];
   r2=midline1* Rotated;
   axisline=medianline*Rotated;
   endpointrotated(n,1:2)=r2(end,:);
%    plot(midline1(:,1),midline1(:,2),'m')
%    xlabel('x (pixels)')
%    ylabel('y(pixels)')
%    hold on
%    plot(medianline(:,1),medianline(:,2),'m')
%    hold on
%    plot(axisline(:,1),axisline(:,2),'b')
%    hold on
   plot(r2(:,1),r2(:,2),'b')
   pause(.1)
```

```
    hold on
    legend('Before rotation','Before rotation','After rotation')

    %% Maxima and Minima
    rx=r2(:,1);
    ry=r2(:,2);
    [arclen,~] = arclength(ry,rx);
    arc(n)=arclen;
    idy=find(rx<=250);
    ry2=ry(idy);
    mint=min(ry2); %minima
    maxt=max(ry2); %maxima
    idxmin=find(ry2==mint);
    idxmax=find(ry2==maxt);
    rxmax=rx(idxmax);
    rxmin=rx(idxmin);
    rmin(n,1:2)=[rxmin,mint];
    rmax(n,1:2)=[rxmax,maxt];
    plot(rmin(:,1),rmin(:,2),'bo')
    xlim([0 450])
    ylim([-250 150])
    hold on
    plot(rmax(:,1),rmax(:,2),'ro')

    %% %Intersection of rotated curve with x-axis
    % ty=transpose(axisline);
    % tyx=axisline(1,:);
    % tyy=axisline(2,:);
    % rr=transpose(r2);
    % rrx=rr(1,:);
    % rry=rr(2,:);
    % P= InterX(ty,rr);
    % figure(2)
    % plot(tyx,tyy,rrx,rry,P(1,:),P(2,:),'co') %rotated line intersected with axis
    % hold on
    % title(['Frame (' num2str(n) ')'])
    % set(gca,'Ydir','reverse')
    % ylim([-250 250])

    %% Angles
  rx=r2(:,1);
  p=length(rx)/3;
  for i=1:p
  y2=r2(i,2);
  x2=r2(i,1);
  y1=r2(1,2);
  x1=r2(1,1);
  deltay=-(y1-y2);
  deltax=-(x1-x2);
  A=atan(deltay/deltax);
  angleInDegrees(i) = rad2deg(A);
  max2(n)=nanmax(angleInDegrees);
  max3(n)=nanmin(angleInDegrees);
end
  % Wavelength
%Maximum
```

```
ry=rmax(:,2);
rx=rmax(:,1);
Amplitudemax=max(ry);
idy=find(ry==Amplitudemax);
maximum=rmax(idy);
minimum=rmin(idy);
altdist(n)=maximum;
distance(n)=abs((maximum-minimum)*2);
%Minimum
ry1=rmin(:,2);
rx1=rmin(:,1);
Amplitudemax=min(ry1);
idy=find(ry1==Amplitudemax);
maximum=rmax(idy);
minimum=rmin(idy);
altdist2(n)=minimum;
distance2(n)=abs((minimum-maximum)*2);
end

%mean arclength
arcln=nanmean(arc);

%Median endpoints before and after rotation
med1=median(endpoint);
med2=median(endpointrotated);

%% Angles
%Angle for negative values
max5=abs(max3);
mean_anglemin=nanmean(max5)
stdv=std(max5);
n=length(max5);
t=sqrt(n);
erroranglemin=stdv/t

%Angle for positive values
max4=abs(max2);
mean_anglemax=nanmean(max4)
stdv=std(max4);
n=length(max4);
t=sqrt(n);
erroranglemax=stdv/t

%% Amplitude and wavelength
ry=rmax(:,2);
rx=rmax(:,1);
ry1=rmin(:,2);
rx1=rmin(:,1);

%Amplitude positive values
r1=smoothdata(rmax(:,2));
[pks1,locs1]=findpeaks(r1);
findpeaks(r1)
idx=locs1;
xmaxpeaks=rx(idx);
ymaxpeaks=ry(idx);
```

```
Ampmax=abs(ymaxpeaks/arcln)
stdv2=std(Ampmax);
n=length(Ampmax);
t=sqrt(n);
errorampmax=stdv2/t

%Wavelength positive values
xminpeaksformax=rx1(idx);
yminpeaksformax=ry1(idx);
Ampmax_min=yminpeaksformax/arclen;
wavelengthmax=((xmaxpeaks-xminpeaksformax)/arclen)*2
stdv=std(wavelengthmax);
n=length(wavelengthmax);
t=sqrt(n);
errorwavemax=stdv/t

% Amplitude negative values
inv=-(rmin(:,2));
r2=smoothdata(inv);
[pks2,locs]=findpeaks(r2);
findpeaks(r2)
idx1=locs;
xminpeaks=rx1(idx1);
yminpeaks=ry1(idx1);
Ampmin=abs(yminpeaks/arcln)
stdv2=std(Ampmin);
n=length(Ampmin);
t=sqrt(n);
errorampmin=stdv2/t

%Wavelength negative values
xmaxpeaksformin=rx(idx1);
ymaxpeaksformin=ry(idx1);
Ampmin_max=ymaxpeaksformin/arclen;
wavelengthmin=((xminpeaks-xmaxpeaksformin)/arclen)*2
stdv2=std(wavelengthmin);
n=length(wavelengthmin);
t=sqrt(n);
errorwavemin=stdv2/t

% %% Average values
% Avgwavelength=(wavelengthmin + wavelengthmax)/2
% %Max
% Avgamplitudes= (Ampmax+Ampmax_min)/2
% Avgamplitudenegative= (Ampmin+Ampmin_max)/2
```

APPENDIX 3. Matlab analysis of test model

```
clear all
%% Load video
v = VideoReader('3_6.avi');
numframes = round(v.Duration * v.FrameRate);
inc=2;
thresh=90; %Threshold for binary conversion
```

```
%% Find median of endpoint for rotation
for fff=1:numframes
%% Load frame
imm = read(v,fff);
imm=imrotate(imm,-90);
imm=imrotate(imm,180);

% Monochromize
imm = rgb2gray(imm);
%imshow(imm)

% Cropping
% xl = [113 1432];
% yl = [199 1047];
% imm = imm(yl(1):yl(end),xl(1):xl(end),:);
%immorig = imm;
%imshow(immorig)

% Convert to Binary
immthresh = imm > thresh;
%imshow(immthresh)

% Remove extra objects
immthresh = bwareaopen(immthresh,111);
%imshow(immthresh)

%Find Midline
[xpts,ypts]=find(immthresh==1);
[xpts2,ind,~]=unique(xpts);
ypts=unique(ypts);
counter=1;
midline=1;
for n=ypts(1,1):1:ypts(end,1)
  q=find(immthresh(:,n)==1);
  midline(counter,1:2)=[n,median(q)];
  counter=counter+1;
end

%Move curve to origin
ys=midline(1,2);
xs=midline(1,1);
midlinex=midline(:,1)-xs;
midliney=midline(:,2)-ys;
midline1=[midlinex,midliney];
%Find endpoint
endpoint(fff,1:2)=midline1(end,:);
end

%Find mid-endpoint
med=median(endpoint);
endptx=med(1,1);
endpty=med(1,2);

%% Analysis
for fff=1:numframes
%% Image modifications
```

```
imm = read(v,fff);
imm=imrotate(imm,-90);
imm=imrotate(imm,180);
% Monochromize
imm = rgb2gray(imm);

% % Cropping
% xl = [113 1432];
% yl = [199 1047];
% imm = imm(yl(1):yl(end),xl(1):xl(end),:);
%immorig = imm;

% Convert to Binary
immthresh = imm > thresh;
% Remove extra objects
immthresh = bwareaopen(immthresh,111);

%%Midline
[xpts,ypts]=find(immthresh==1);
[xpts2,ind,~]=unique(xpts);
ypts=unique(ypts);
counter=1;
midline=1;
for n=ypts(1,1):1:ypts(end,1)
  q=find(immthresh(:,n)==1);
  midline(counter,1:2)=[n,median(q)];
  counter=counter+1;
end

%Move curve to origin
ys=midline(1,2);
xs=midline(1,1);
midlinex=midline(:,1)-xs;
midliney=midline(:,2)-ys;
midline1=[midlinex,midliney];
startptx=midline1(1,1);
startpty=midline1(1,2);

%Line from startpoint to median-endpoint
x =linspace(startptx,endptx,300);
y =linspace(startpty,endpty,300);
m=[x;y];
medianline=transpose(m);

%% Amplitude
%Rotation of curve to make median-endpoint the median of the curve
endx=x(1,end);
endy=y(1,end);
deltay=(endy-startpty);
deltax=(endx-startptx);
A=atan(deltay/deltax);
Rotated=[cos(A),-sin(A);sin(A),cos(A)];
r2=midline1* Rotated;
axisline=medianline*Rotated;
endpointrotated(n,1:2)=r2(end,:);
```

```
% %Rotation plot
% plot(midline1(:,1),midline1(:,2),'c','Linewidth',3);
% xlabel('x (pixels)')
% ylabel('y(pixels)')
% set(gca, 'FontSize', 22)
% hold on
% plot(r2(:,1),r2(:,2),'b','Linewidth',3);
% legend('Before rotation','After rotation')
% hold on
% plot(medianline(:,1),medianline(:,2),'c','Linewidth',3)
% hold on
% plot(axisline(:,1),axisline(:,2),'b','Linewidth',3)
% hold on
%% Maxima and Minima
rx=r2(:,1);
ry=r2(:,2);

%Find arclength
[arclen,seglen] = arclength(ry,rx);
arc(fff)=arclen;

%X-axis cutoff to ignore hinge
idy=find(rx<=800);
ry2=ry(idy);
rx2=rx(idy);
mint=min(ry2); %minima
maxt=max(ry2); %maxima
idxmin=find(ry2==mint);
idxmax=find(ry2==maxt);
rxmax=rx(idxmax);
rxmin=rx(idxmin);
rmin(fff,1:2)=[rxmin,mint];
rmax(fff,1:2)=[rxmax,maxt];

% %Min and max points
% plot(r2(:,1),r2(:,2))
% title(['Frame (' num2str(fff) ')'])
% xlabel('X (pixels)')
% ylabel('Y (pixels)')
% set(gca, 'FontSize', 22)
%hold on
% plot(rmax(:,1),rmax(:,2),'bo')
% hold on
% plot(rmin(:,1),rmin(:,2),'ro')
% legend('Tail Line per frame','Maximum Y values per frame','Minimum Y values per frame')

%% %Intersection of rotated curve with x-axis
% ty=transpose(medianline);
% tyx=medianline(1,:);
% tyy=medianline(2,:);
% rr=transpose(r2);
% rrx=rr(1,:);
% rry=rr(2,:);
% P= InterX(ty,rr);
% figure(2)
% %plot(tyx,tyy,rrx,rry,P(1,:),P(2,:),'co') %rotated line intersected with axis
```

```
% % hold on
% title(['Frame (' num2str(fff) ')'])
% set(gca,'Ydir','reverse')
% % ylim([-250 250])

%% Angles
p=length(rx)/3;
for i=1:p
y2=r2(i,2);
x2=r2(i,1);
y1=r2(1,2);
x1=r2(1,1);
deltay=(y2-y1);
deltax=(x2-x1);
A=atan(deltay/deltax);
angleInDegrees(i) = rad2deg(A);
angleInDegrees(angleInDegrees == 0)= NaN;
max2(fff)=nanmax(angleInDegrees);%angle for positive values
max3(fff)=nanmin(angleInDegrees);%angle for negative values
end
end

%mean arclength
arcln=nanmean(arc);

%Median endpoints before and after rotation
med1=median(endpoint);
med2=median(endpointrotated);

%% Angles
%Angle for negative values
max5=abs(max3);
mean_anglemin=nanmean(max5)
stdv=std(max5);
n=length(max5);
t=sqrt(n);
erroranglemin=stdv/t

%Angle for positive values
max4=abs(max2);
mean_anglemax=nanmean(max4)
stdv=std(max4);
n=length(max4);
t=sqrt(n);
erroranglemax=stdv/t

%% Amplitude and wavelength
ry=rmax(:,2);
rx=rmax(:,1);
ry1=rmin(:,2);
rx1=rmin(:,1);

%Amplitude positive values
r1=smoothdata(rmax(:,2));
[pks1,locs1]=findpeaks(r1);
findpeaks(r1)
```

```
idx=locs1
xmaxpeaks=rx(idx);
ymaxpeaks=ry(idx);
Ampmax=abs(ymaxpeaks/arcln)
stdv2=std(Ampmax);
n=length(Ampmax);
t=sqrt(n);
errorampmax=stdv2/t

%Wavelength positive values
xminpeaksformax=rx1(idx)
yminpeaksformax=ry1(idx)
Ampmax_min=yminpeaksformax/arclen
wavelengthmax=((xmaxpeaks-xminpeaksformax)/arclen)*2
stdv=std(wavelengthmax);
n=length(wavelengthmax);
t=sqrt(n);
errorwavemax=stdv/t

% Amplitude negative values
inv=-(rmin(:,2));
r2=smoothdata(inv);
[pks2,locs]=findpeaks(r2);
findpeaks(r2)
idx1=locs
xminpeaks=rx1(idx1);
yminpeaks=ry1(idx1);
Ampmin=abs(yminpeaks/arcln)
stdv2=std(Ampmin);
n=length(Ampmin);
t=sqrt(n);
errorampmin=stdv2/t

%Wavelength negative values
xmaxpeaksformin=rx(idx1);
ymaxpeaksformin=ry(idx1);
Ampmin_max=ymaxpeaksformin/arclen
wavelengthmin=((xminpeaks-xmaxpeaksformin)/arclen)*2
stdv2=std(wavelengthmin);
n=length(wavelengthmin);
t=sqrt(n);
errorwavemin=stdv2/t

% %% Average values
% Avgwavelength=(wavelengthmin + wavelengthmax)/2
% %Max
% Avgamplitudes= (Ampmax+Ampmax_min)/2
% Avgamplitudenegative= (Ampmin+Ampmin_max)/2
```