AN ABSTRACT OF THE THESIS OF

Jacquelin Leah Remaley for the degree of Master of Science in Robotics presented on May 31, 2018.

Title: Improving Understanding of Geometric Swimmer Locomotion

Abstract approved: ____

Ross L. Hatton

This work seeks to improve the knowledge surrounding geometric snake swimmers through both theoretical and practical means.

To begin, manufacturing techniques for a silicon-based pneumatic actuator were validated through experimentation. When inflated, the actuators exhibited an unanticipated elongation. In an attempt to confirm theoretical displacement predictions, a more accurate dynamic model was created which included curvature-coupled extension. The resulting model was simulated with varying degrees of extension and corresponding optimized gaits. Results suggest that the pneumatic actuators efficiency does improve, although serpenoid systems used as a baseline comparison decreased in efficiency when extension was added.

For the second stage of this work, it was assumed that a systems theoretical model would be either completely unknown or unreasonable to calculate, thus another motion prediction method would be required. This method, referred to as Data-

Driven or Empirical Local Connection Derivation, requires the system to execute a gait which adequately spans the desired shape space while its position and velocity are tracked through motion capture. This process is demonstrated using two different locomotors.

Finally, these methods are integrated into a continually updated MATLAB® graphical user interface (GUI) titled Geometric System Plotter. The aim of this software is to provide geometric system evaluation techniques without requiring background knowledge in geometric mechanics. In addition to the projects described here, other tools and general performance improvements have been integrated into the software for future releases.

Improving Understanding of Geometric Swimmer Locomotion

by

Jacquelin Leah Remaley

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented May 31, 2018
Commencement June 2018

ACKNOWLEDGEMENTS

I would first and foremost like to acknowledge my advisor and mentor Dr. Ross Hatton for his guidance and understanding throughout this work. He was always available for discussion and provided a sturdy backdrop from which I could continue research, and I am extremely thankful that I was able to work with him.

Collaborators I would like to acknowledge include Callie Branyan, Chloe Fleming, Ammar Kothari, and Suresh Ramasamy from the Oregon State University Robotics Institute, and Julian Whitman and Brandon Hung from the Carnegie Mellon University Biorobotics Laboratory. I am deeply appreciative of all their efforts which helped make this work a reality. Suresh also provided access to his optimizer, without which much of this work would not be possible.

Next, I would like to thank my lab-mates Suresh Ramasamy, Hossein Faraji, Luke Hill, and Andrew Otto for indulging many on- and off-topic conversations and being incredibly supportive friends during our graduate work.

I would also like to acknowledge the love and support I received from my family. My parents were always available to provide a voice of support and reason, and my siblings helped keep me entertained during long nights.

This work was funded by National Science Foundation Grant #1653220.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

LIST OF FIGURES

LIST OF FIGURES (Continued)

LIST OF FIGURES (Continued)

Chapter 1: Introduction

Efficiently driving robot locomotion has been a topic of high interest as long as there have been robots. Recently, research efforts in the geometric mechanics community have centered on visualization tools and finding efficient methods to intuitively predict the dynamic behavior of robotic systems. Many of these systems take inspiration from nature in an attempt to emulate the kinematics of a biological system, such as snakes or other swimming animals. Designs and control methods for such articulated systems have varied greatly, from the Purcell Swimmer first introduced over forty years ago [3] to modular continuum snake robots in more recent years [4,5]. Further complications arise when more versatile materials are introduced to the design, such as silicone [6–9] and other compliant materials with nonlinear behavior.

In order to broaden our understanding surrounding these swimmers, this work defines and demonstrates the capabilities of a modified geometric swimmer, describes a novel procedure for physical displacement prediction, and integrates each new method into a graphical user interface (GUI). Each of these contributions attempts to improve knowledge behind design and behavior prediction of geometric swimmers in the hope of facilitating future research.

1.1 Outline

To provide context and motivation behind much of this work, previous advances in geometric mechanics will be described briefly in the following chapter.

In the second chapter, the groundwork for geometric mechanics theory is summarized to provide a foundation for work described in later chapters.

The third chapter outlines the first contribution of this work. It examines the effects of extensibility on piecewise soft snake and serpenoid backbones. This study stemmed from investigating manufacturing techniques for a modular soft snake robot. During testing, the pneumatic silicon actuators experienced a lengthwise extension as a consequence of the actuator structure. A modified dynamic model is proposed which includes this extension. The chapter concludes with results from simulating the new model and corresponding optimal gaits.

The fourth chapter presents a method for computing the local connection empirically using motion capture data and describes the mathematics required. It then gives examples of this data-driven method in action, both on theoretical and physical systems.

The last chapter describing a contribution details how work in the previous sections was integrated with a custom Matlab GUI called Geometric Systems Plotter.

Finally, there will be a discussion considering the common themes linking all contributions and potential research areas they may lead to.

1.2 Relevant Work

Interest in swimmer locomotion rose in the mid-twentieth century with investigations into propulsion mechanisms of biological organisms, particularly those which reside in homogeneous fluids such as air or water. At the time, analysis of bodies moving through fluids considered both the propulsive force of the driving mechanism and the momentum offered by the fluid as the body moves through it. In 1951, Taylor [10] demonstrated that propulsion of swimming organisms such as fish or spermatozoa through low Reynolds number fluid relies largely on viscous forces, as opposed to inertia.

Two and a half decades later, Edward Purcell gave a talk, later reprinted in the American Journal of Physics, titled *Life at Low Reynolds Number* describing the alien fluid world many organisms inhabit [3]. Organisms that swim in these environments are typically microscopic and must rely on articulating flagella or something similar to move through a fluid which is relatively viscous to them. At viscosities such as these, inertial forces dominate and motion is theoretically impossible without more than one articulating joint. Purcell proposed a swimmer composed of three links and two articulating joints as the simplest geometric system capable of movement in a low Reynolds number fluid. This model became a standard benchmark for locomotion prediction in highly viscous fluids.

Following these observations, several works began investigating motion prediction of kinematic systems. One of the first was written by Shapere and Wilczek [11], and described a mathematical framework for predicting net translation and rotation under cyclic joint articulations, or gaits. This framework resulted in the *gauge potential field* A, later referred to as the *local connection.* This acts as a Jacobian, converting directly between body and gait velocity.

Murray [12] and Kelley [13] expanded on the framework concerning effects from the evaluation of the Lie bracket of the gait to generate optimal joint articulations, or shape changes, for wheeled planar robots.

The local connection was mentioned again in a work by Ostrowski [14], which dealt with motion planning for systems that locomote through shape undulations. A special class of such systems are principally kinematic systems, where all group symmetries are annihilated by the kinematic constraints, resulting in the *principal kinematic connection* equation.

With improved motion prediction methods, Purcell's Swimmer was revisited by Becker et. al. [15]. They determined that both the amplitude of the joint angle changes and relative link sizes affect net translation. It was also demonstrated that low Reynolds number fluids exert anisotropic drag forces on inextensible swimmers. Tam and Hosoi continued this analysis, determining an optimal gait based on expended energy [16].

Methods for gait generation continued to be modified and applied to other systems. The reconstruction equation first proposed by [14] was evaluated for a geometric pivoting robot and a 3-link kinematic snake by Shammas et. al, with the addition of *height functions*, a tool meant to assist in designing efficient gaits [16]. These functions are surface representations of the integral of a position velocity, whose geometric features such as peaks and valleys can be used to design gaits. Avron and

Raz [17] also discuss height functions under the name *curvatures*, and investigate a turning gait for the Purcell Swimmer.

Following this, Hatton and Choset produced several works which add to the repertoire of gait evaluation tools [18]. The first, similar to height functions, is a vector representation of the local connection A over the shape space. This allowed for informed gait generation without integrating equations of motion. Next, they proposed the *body velocity integral* (BVI) as a more correct interpretation of the height-function integrals. The coordinate choice affects the accuracy of the BVI as a proxy for the net displacement [19], and a comparison between the traditional and proposed 'intuitive' coordinate frame is presented.

The issue of selecting ideal coordinate systems for accurate prediction was addressed in [1]. The presented method applies Hodge-Helmholtz decomposition to the constraints on a system to find the optimal coordinate choice for that system. The result is a coordinate set which experiences 'minimum perturbation' of the position variables in response to shape variables.

Hatton and Choset then demonstrated the usefulness of this optimal coordinate set by revisiting the Purcell Swimmer in low Reynolds number fluids [20]. This work and its successor [21] demonstrated a clear reduction in prediction error between the previous arbitrary coordinate frame selection and the new optimized coordinate approximation.

Another useful tool for gait generation considers the energy cost of a system moving through certain shapes [22]. Using cartographic principles, it is shown that although an efficient gait can be drawn using the information from connection vector fields or curvature height functions, it may not be a true projection of the cost that gait will incur. The distortion generated by the energy required to move through the shape space is encoded in a *Riemannian metric*. This can be used to distort the shape space itself for a more accurate representation based on cost.

The most recent advancement which is useful to this work was completed by Ramasamy, and utilizes the concepts from [22] to define optimal gaits for kinematic systems. The optimization process can be compared to the equilibrium experienced by a soap-bubble. Gaits which produce maximum displacement encircle areas of high magnitude on the curvature surfaces [16,17]. Thus, maximizing displacement pushes the gait towards a zero-level isocline, while the length of the gait tends to shrink to minimize the shape change cost. Optimization was demonstrated on systems with two inputs in [23], extended to three inputs in [24], and to n inputs in [25] (in review).

Chapter 2: Background

In this chapter, a broad background concerning geometric mechanics topics covered in this paper will be given. Additional detail will be shown in the following chapters as necessary.

2.1 Geometric Mechanics

Robotic systems which are capable of locomotion do so through exploiting their physical relationship with their surrounding media. Defining this relationship first requires knowledge of the robot's morphology and kinematics: how the robot can move and what shapes it makes given certain input commands. Secondly, movement prediction requires defining how the robot's shape interacts with the world around it, e.g., the directional force it experiences from fluid friction.

We take a robot's shape r as an element of its shape space $r \in M$ and its resulting body velocity $\overset{\circlearrowright}{g}$ in the position space $g \in G$. The body velocity $\overset{\circlearrowright}{g}$ is the velocity with respect to a defined body frame attached to some segment of the robot, while world velocity $\dot{g}$ is defined in global coordinates. Body velocity and shape can be related through the *kinematic reconstruction equation*,

$$\mathring{g} = -A(r)\dot{r} + \Gamma(r)p. \tag{2.1}$$

where $A(r)$ is the *local connection*, $\Gamma(r)$ is the *momentum distribution equation*, and p is the *generalized nonholonomic momentum* [26–28].

In this work, we only consider systems which experience local anisotropic friction, primarily swimming in high viscosity fluids with very low Reynolds numbers. This results in the 'coasting' effects from the momentum terms dropping out [26], and equation (2.1) is simplified as

$$\mathring{g} = A(r)\dot{r}. \tag{2.2}$$

The local connection $A(r)$ acts as a Jacobian directly relating the changes in geometric shape, or shape velocity, to the resulting body velocity of a system [11,22]. Here, we assume a system which operates in a two-dimensional plane with body coordinates $g = (x, y, \theta)$ and two shape inputs $r = (\alpha_1, \alpha_2)$. Body coordinates are defined in meters and radians, and shape inputs are unitless unless otherwise specified.

The following sections detail the process of defining the local connection for a given geometry.

2.1.1 Backbone Locus

As discussed previously, there are two relationships required to constrain a system geometry and predict its movement.

First, the geometry of the system and its response to shape inputs must be defined. Because this work focuses primarily on locomotion inspired by snakes, the methods described here will concern systems with a single backbone whose shape is driven by an input gait $r(t) = [\alpha_1(t), \alpha_2(t)]$, where the shape inputs are functions of time. The backbone is assumed to be unit length, with a uniform and unchanging mass distribution.

For a general backbone, the global position and orientation of a frame tangent to the backbone of a particle on the backbone $h(s) \in SE(2)$ relative to the body frame is defined as

$$h(s) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \int_0^s \begin{bmatrix} R(\theta(s)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \kappa(\alpha, s) \end{bmatrix} d\sigma, \tag{2.3}$$

where s is the curve length along the backbone measured from the body frame, α is the input, and $\kappa(\alpha, s)$ is the curvature experienced by a discrete segment of the backbone.

For a system with two inputs, the total curvature is the sum of curvatures contributed by each input α,

$$\kappa(\alpha, s) = \begin{bmatrix} \kappa_1(s) & \kappa_2(s) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = \kappa_1\alpha_1 + \kappa_2\alpha_2. \tag{2.4}$$

Functions for κ_1 and κ_2 determine the physical shape of the backbone and its response to shape changes. As examples, this work will hereafter refer to two distinct systems; the three-link swimmer which relies on discrete bending, and the serpenoid swimmer, which has a more continuous shape.

The first system is the three-link or Purcell swimmer. It was first proposed as

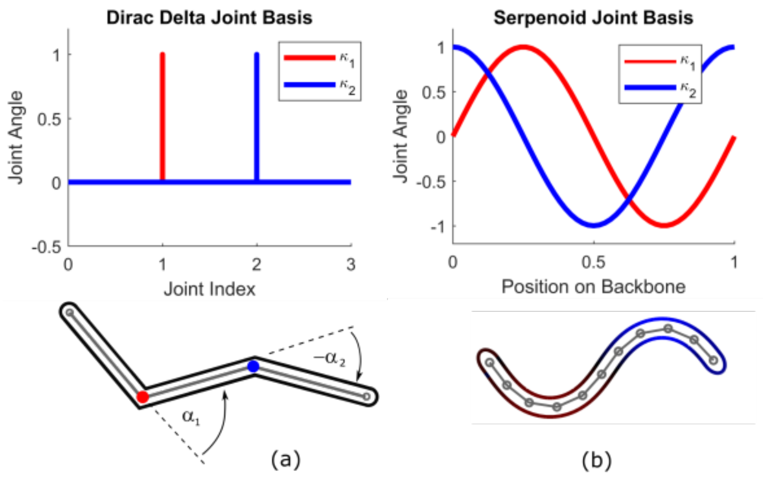


Figure 2.1: Shape bases and basic geometries for the Purcell (a) and Serpenoid (b) swimmers.

the simplest mechanism which achieves movement through low Reynolds number fluids [3, 21], and remains a useful model for locomotion comparison. Its geometry is shown in figure 2.1a. This backbone's curvature is given by a set of delta functions that define the articulating corners between three equal-length links [22]. Each link has a length of $\frac{1}{3}$, thus the curvature function utilizes half of this length to map distinct curvatures to each joint,

$$\kappa(\alpha, s) = \delta(s + \frac{1}{6})\alpha_1 + \delta(s - \frac{1}{6})\alpha_2. \tag{2.5}$$

The second system is a serpenoid backbone with sinusoidal curvature, shown in figure 2.1b. Serpenoid curvature uses both inputs as modal amplitudes to determine wave shape [29],

$$\kappa(\alpha, s) = \alpha_1 \cos(2\pi s) + \alpha_2 \sin(2\pi s). \tag{2.6}$$

With the backbone shape defined, the body velocity of an individual point on the backbone $\overset{\circ\!\rightarrow}{g}(s)$ can be found by calculating both the gradient of its position with respect to the input and the time derivative of the input shape. Multiplying these together gives the time derivative of global position.

To obtain the gradient of position with respect to shape, we differentiate equation (2.3) with respect to α and convert to body coordinates.

The gradient of (2.3) can be evaluated by taking advantage of the Leibniz rule to move the gradient term inside the integral, then evaluating through chain rule. The following equations evaluate each component of this gradient expression, beginning with the gradient itself,

$$\nabla_\alpha h(s) = \int_0^s \left(\nabla_\alpha \begin{bmatrix} R(\theta(\sigma)) & 0 \\ 0 & 1 \end{bmatrix} \right) \begin{bmatrix} 1 \\ 0 \\ \kappa(\alpha,\sigma) \end{bmatrix} + \begin{bmatrix} R(\theta(\sigma)) & 0 \\ 0 & 1 \end{bmatrix} \left(\nabla_\alpha \begin{bmatrix} 1 \\ 0 \\ \kappa(\alpha,\sigma) \end{bmatrix} \right) d\sigma. \tag{2.7}$$

The gradient of the rotation matrix gives

$$\nabla_\alpha(R(\theta)) = \nabla_\alpha \begin{bmatrix} \cos\theta(s) & -\sin\theta(s) \\ \sin\theta(s) & \cos\theta(s) \end{bmatrix} = \nabla_\alpha \theta \cdot \begin{bmatrix} -\sin\theta(s) & -\cos\theta(s) \\ \cos\theta(s) & -\sin\theta(s) \end{bmatrix}. \tag{2.8}$$

The gradient of $\theta(s)$ can be found using the definition of $\kappa(\alpha, s)$ in (2.4), with $\theta(s)$ defined as

$$\theta(s) = \int_0^s \kappa(\alpha, \sigma) d\sigma \tag{2.9}$$

giving

$$\nabla_\alpha \theta(s) = \int_0^s \nabla_\alpha \kappa(\alpha, \sigma) d\sigma. \tag{2.10}$$

To differentiate curvature κ, recall that each κ_n term is associated with an α_n, as in equation (2.4),

$$\nabla_\alpha \kappa(\alpha, s) = \begin{bmatrix} \frac{\partial \kappa}{\partial \alpha_1} & \frac{\partial \kappa}{\partial \alpha_2} \end{bmatrix} |_{\alpha=(\alpha_1,\alpha 2)} = \begin{bmatrix} \kappa_1 & \kappa_2 \end{bmatrix}. \tag{2.11}$$

With the gradient of the position with respect to shape fully defined, we can calculate the gradient of local position with respect to time $\dot{h}(s)$ by multiplying with the derivative of shape with respect to time $\dot{\alpha} = \frac{d\alpha}{dt}$,

$$\dot{h}(s) = \nabla_\alpha h(s) \cdot \dot{\alpha}. \tag{2.12}$$

Local body velocity $\overset{\circ}{h}(s)$ is then converted from world velocity $\dot{h}$ using a left tangent transformation,

$$\overset{\circ}{h}(s) = T_{h(s)} L_{h^{-1}(s)} \dot{h}(s), \tag{2.13}$$

where the left tangent transformation $T_{g_0} L_{g_\Delta}$ is the lifted action for converting between body and world velocities and has a structure similar to a rotation matrix.

$$T_{g_0}L_{g_\Delta} = \frac{\partial(g_\Delta g_0)}{\partial g_0} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.14}$$

Finally, the general body velocity of the backbone $\overset{\circ\rightarrow}{g}(s)$ can be found by summing the transformed base velocity and the body velocity of that point on the locus, relative to the system's body frame.

$$\overset{\circ\rightarrow}{g}(s) = Ad^{-1}_{h(s)}\overset{\circ\rightarrow}{g}(0) + \mathring{h}(s). \tag{2.15}$$

2.1.2 Swimming Forces

The other relationship required for displacement prediction utilizes the shape-to-body velocity definition to determine the resultant forces experienced by an articulating system while in a low Reynolds number media.

One assumption this work makes is that the backbone is swimming in a fluid which gives locally anisotropic viscous friction. This implies that the forces on the backbone are linearly related to the body velocity $\overset{\circ\rightarrow}{g}$. Secondly, the backbone is in quasi-static equilibrium, which allows the model to be constrained to motions in which the sum of the forces equal zero.

Beginning with the viscous friction constraints, we can relate the local force $F(s)$ to the body velocity $\overset{\circ\rightarrow}{g}$ as,

$$F(s) = C\mathring{\vec{g}}(s) = \begin{bmatrix} 1 & & \\ & 2 & \\ & & 0 \end{bmatrix} \begin{bmatrix} \mathring{\vec{g}}^{x} \\ \mathring{\vec{g}}^{y} \\ \mathring{\vec{g}}^{\theta} \end{bmatrix} (s), \tag{2.16}$$

where C is a drag coefficient matrix containing coefficients for converting between body velocity of points on the backbone to local friction. We assume $2:1$ friction, with the lateral drag force having twice the magnitude of the tangent component, and no effect from rotational forces. [21].

Given the position of each body segment relative to the static body frame, local forces can be converted to body forces acting on the system using an inverse adjoint transformation,

$$F_b(s) = Ad^{-1}_{h(s)} F(s), \tag{2.17}$$

where the inverse adjoint operation performs transformations using a combination of cross products and rotations between linked frames. For a frame at $h(s) = (x, y, \theta)$, this transformation is

$$Ad^{-1}_{h(s)} = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -y \\ 0 & 1 & x \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.18}$$

Finally, the net force is equal to the integral of the forces contributed by different points along the body with the integral limits set so the backbone is of unit length with the body frame at the center,

$$F_{b,total} = \int_{-0.5}^{0.5} F_b ds. \tag{2.19}$$

2.1.3 The Local Connection, Gait Generation, and Other Estimation Tools

The result of evaluating (2.19) consists of several coordinate transformations on the body and shape velocity. These velocities can be separated and factored out, leaving a 3×5 matrix of coefficients $\omega(\alpha)$ and a 5×1 matrix of velocities,

$$F_{b,total} = \omega(\alpha) \begin{bmatrix} \overset{\circ\rightarrow}{g} \\ \dot{\alpha} \end{bmatrix}. \tag{2.20}$$

Recall that the system is assumed to be in quasi-static equilibrium, and the external force in (2.19) can be set to zero. Concurrently, the coefficient matrix $\omega(\alpha)$ can be split into two blocks which relate to either the body or shape velocity:

$$F_{b,total} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \omega_g^{3x3} & \omega_\alpha^{3x2} \end{bmatrix} \begin{bmatrix} \overset{\circ\rightarrow}{g} \\ \dot{\alpha} \end{bmatrix} \tag{2.21}$$

Multiplying through gives $\omega_g \overset{\circ\rightarrow}{g} = -\omega_\alpha \dot{\alpha}$, thus $\overset{\circ\rightarrow}{g}$ is

$$\overset{\circ\rightarrow}{g} = -\omega_g^{-1} \omega_\alpha \dot{\alpha}. \tag{2.22}$$

Once the constraint equations have been solved to match the previous format, the local connection $A(r)$ is defined as

$$A(\alpha) = -\omega_g^{-1}\omega_\alpha \tag{2.23}$$

giving the same definition for simplified kinematic reconstruction equation as in 2.2.

The resulting local connection matrix has one row for each body position variable, and one column for each input α value.

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} A_{\alpha_1,x} & A_{\alpha_2,x} \\ A_{\alpha_1,y} & A_{\alpha_2,y} \\ A_{\alpha_1,\theta} & A_{\alpha_2,\theta} \end{bmatrix} \begin{bmatrix} \dot{\alpha_1} \\ \dot{\alpha_2} \end{bmatrix} \tag{2.24}$$

Each row of A can be interpreted as the element-wise effects the shape has on position. This can be visualized by evaluating both coefficients over a grid of α values and displaying the results as a vector plot. For reference, connection vector plots for the Purcell swimmer are shown in figure 2.2.

Utilizing a generalization of Stokes' theorem [26], it can be shown that the net displacement resulting from a closed loop gait can be approximated with the area integral of the surface region of the curl of $A(\alpha)$ and Lie bracket $[A_1, A_2]$ enclosed by the gait.

$$\Delta g = \oint_\phi -A(\alpha)d\alpha = \iint_\phi -\text{curl}A + [A_1, A_2]d\alpha \tag{2.25}$$

A useful tool for visualization is the *constraint curvature function* (CCF). This func-

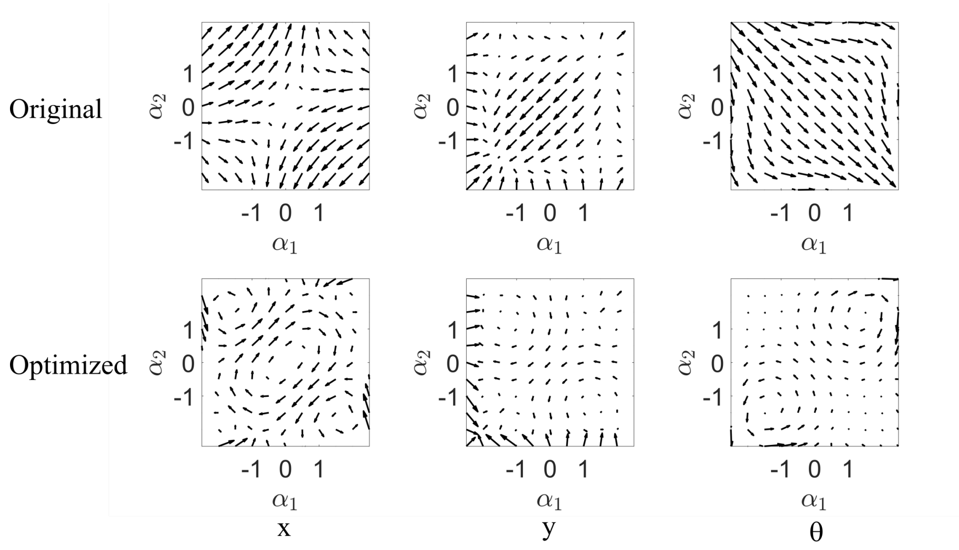


Figure 2.2: Connection vector plots for the 3-link Purcell Swimmer. Left shows the x coordinate, middle is y, and right is θ, while the top row shows the original coordinate system and the bottom shows optimized coordinates, which are detailed later this chapter [1].

tion is the augmented curl of A and includes Lie bracket effects $[A_1, A_2]$ as in (2.25). CCF plots for the Purcell Swimmer are shown in figure 2.3.

When evaluating (2.25), selecting an arbitrary orientation and position for the body frame can result in significant position estimate error for motions beyond planar rotation and infinitesimal translations. However, applying a body frame whose orientation in world coordinates experiences minimum perturbation, i.e. *optimal coordinates*, results in a much more accurate estimation [1, 26]. The bottom rows of figures 2.2 and 2.3 show the respective plots in optimal coordinates, while the top rows are in the original body frame.

In order to achieve maximum displacement per cycle, the gait should enclose the maximum amount of sign-definite area on the CCF surface, thereby maximizing the

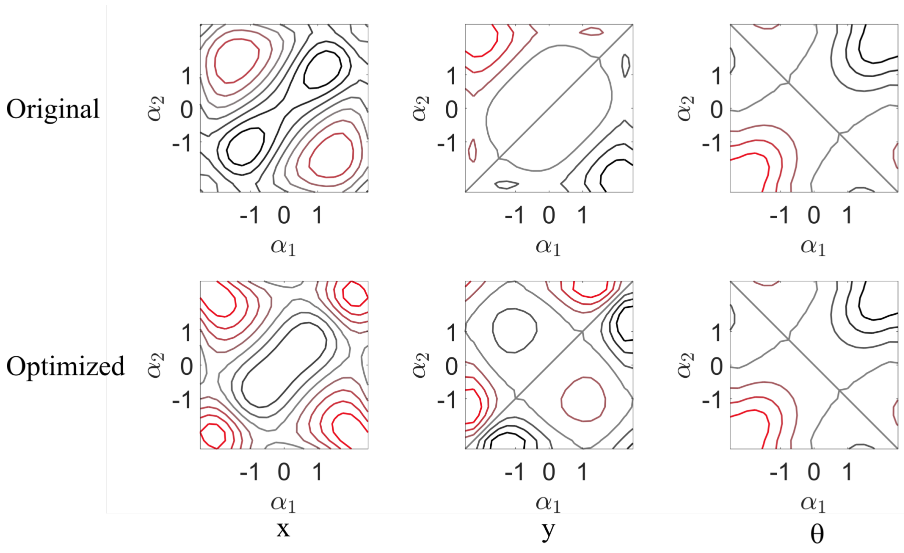


Figure 2.3: Constraint Curvature Function contour plots for the 3-link Purcell Swimmer. Left shows the x coordinate, middle is y, and right is θ, while the top row shows the original coordinate system and the bottom shows optimized coordinates, discussed later. Red contour lines represent positive regions, and black represents negative.

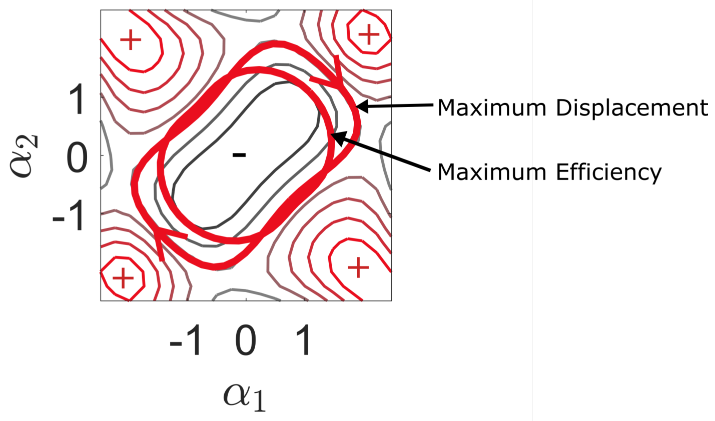


Figure 2.4: x Curvature Constraint Function plot for the Purcell Swimmer. The maximum displacement gait, which matches that given in [2], lies on the zero level of the height function DA. The maximum efficiency gait is similar, but gives up areas of high cost and low gain.

value of the integral in equation (2.25). The simplest illustration of this is a gait following the zero-contour line of the contour representation [21].

However, this is not necessarily the highest *efficiency* gait. If the cost of executing a specific shape change is considered a metric for optimization, it follows that a path with the least cost, and therefor the shortest length, while maximizing enclosed surface integral can be generated from a starting seed gait [23]. The seed gait is usually a path along the zero contour, although a simple circle or other 'best-guess' seed can also be used. As compared to the maximum-displacement gait, the maximum efficiency gait gives up areas of high cost and low displacement. An interesting example of this difference is the Purcell Swimmer, whose maximum displacement and maximum efficiency gaits are shown in figure 2.4. Maximizing efficiency causes the gait to compress inwards, trimming the extreme ends of the enclosed area.

Optimizing according to path length requires use of the *Riemannian Metric's* local *metric tensor* $\mathcal{M}$, a measure of the relative movement cost in each dimension, to relate path length to actual costs felt by the system. The full derivation is described in detail in [22]. The resulting metric for cost of movement through a system's shape space is

$$ds^2 = d\alpha^T \mathcal{M} d\alpha \tag{2.26}$$

For systems swimming in low Reynolds Number fluids, this can be rewritten as

$$ds_p^2 = pdt^2 = \begin{bmatrix} d\alpha_1 & d\alpha_2 \end{bmatrix} \mathcal{M}_p \begin{bmatrix} d\alpha_1 \\ d\alpha_2 \end{bmatrix} \tag{2.27}$$

where $\mathcal{M}_p$ relates the torque experienced by each body joint and the input body velocities by $\mathcal{T} = \mathcal{M}_p \dot{\alpha}$.

The previously mentioned prediction tools (connection vector fields, CCF plots, and metric stretch) are all intended to provide intuition for how a system will act when driven with a certain gait, as well as effective shapes for that gait. They will be revisited in later chapters as analysis methods.

Chapter 3: Curvature-Driven Backbone Extensibility

As mentioned in Chapter 1, the original project which inspired much of this work involved researching materials and methods from recent advances in soft robotics to fabricate a mobile snake robot system. Ideally, this swimmer would be articulated similarly to the Purcell Swimmer with only two input variables α_1 and α_2, yet consist of a continuous geometry, closer in form to a traditional serpenoid swimmer.

Consider a spectrum which is characterized by continuity in the backbone in one direction and simplicity in actuation in the other. At one end the Purcell Swimmer represents systems which are geometrically simple. That is, the system consists of a relatively low number of segments with spatially-localized modes. On the opposite end, a theoretical serpenoid system has a fully continuous backbone with overlapping modes. A physical version of this system would also require many more degrees of control. Both systems can be seen in figure 3.1a,c.

Located between the extremes of this spectrum is the concept for a pneumatically controlled piecewise swimmer, shown in figure 3.1b. This swimmer consists of two bi-directional actuators linked serially, giving the system two control variables, similar to the Purcell Swimmer. Driving each input with pneumatics means each half of the snake experiences a continuous bending force and will have a continuous curvature. Additionally, the nature of such an actuator requires it to be fabricated from a compliant durable material which is ideal for a variety of scenarios, such as

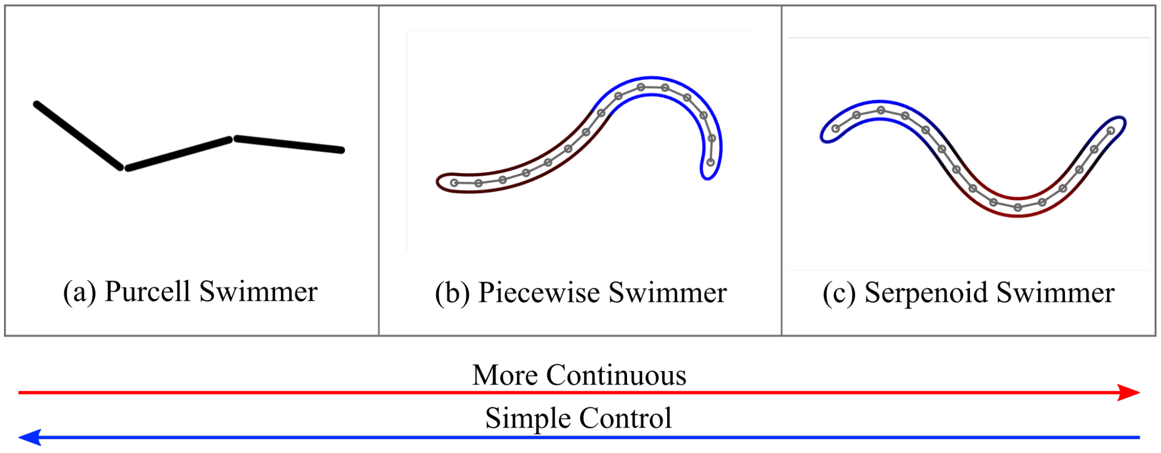


Figure 3.1: A spectrum concerning each swimmer's backbone shape and controllability. (a) shows the Purcell Swimmer, a common benchmark for locomotion analysis. The concept for the piecewise swimmer is shown in (b), while a traditional serpenoid swimmer model is in (c).

safe personal use or as a robust exploration snake for search and rescue operations.

3.1 Piecewise Swimmer Manufacturing and Testing

The manufacturing process leading to the final design for a physical soft snake robot and the subsequent testing are outlined in detail in [6], but will be repeated here for continuity.

3.1.1 Manufacturing

The base requirements for the proposed soft actuators were as follows:

1. The actuators should be pneumatically-driven and contain no hard materials.

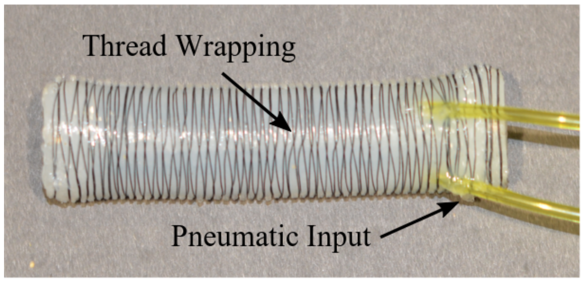


Figure 3.2: A single pneumatically-driven silicone actuator.

2. Each actuator should bend only in the plane of operation; i.e., no twisting or out-of-plane motion.

3. Each actuator must be able to achieve bidirectional bending to at least 90°.

4. The design should be modular, so a series of links can consist of as many actuators as needed.

The first three points are readily achieved using pre-existing actuators [7, 9, 30]. These all use some form of reinforcement or wrapping to achieve bend in a particular direction. However, the actuators in these works either did not exhibit planar bi-directional bending or were impractical for our use.

To achieve all requirements inspiration was taken from [7–9] to create a molded dual-chambered wrapped silicon actuator. The final design is shown in figure 3.2.

Each actuator was molded from EcoFlex®00-30 in three pieces; two caps, each with an inset neodymium magnet, and a central cylinder, which housed the two air chambers. The three individual sections were then glued together with SilPoxy®. A completed actuator was roughly 110mm in length, had an elliptical cross-section whose semi-major and semi-minor axes were 30mm and 20mm respectively, and a

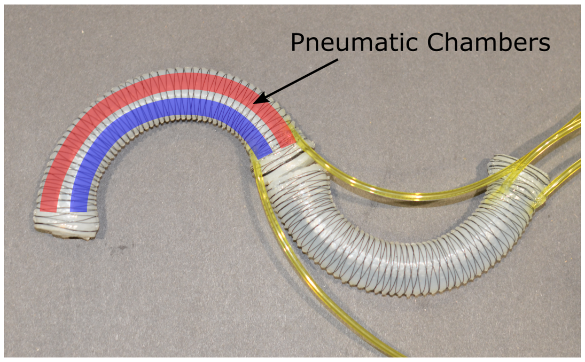


Figure 3.3: The complete soft snake. One actuator has its air chambers outlined to show their relative locations.

wall thickness of 3mm.

The actuators required wrapping to restrict inflation and encourage planar bending as opposed to ballooning behavior. An ideal solution was to wrap along the length counterclockwise and clockwise in a helical pattern with a strong sewing thread. The entire actuator was sealed with another thin layer of EcoFlex®00-30.

Because each actuator corresponds to a single input, a swimmer consisting of two actuators could now be controlled similarly to the Purcell Swimmer. Driving the pressure appropriately generated a continuous curvature along each half of the backbone. Due to the discrete nature of the swimmer's actuation, it was termed and is here referred to as a Piecewise Continuous Swimmer. The final full snake is shown in figure 3.3, which uses two actuators connected in series via magnets.

3.1.2 Geometric Model and Experiment Setup

To determine the piecewise swimmer's viability, gait tests were performed by placing the full swimmer in a bed of millet and observing it with an Optitrack Prime 13 motion capture system. The millet had a low density ratio of 0.78 to 1, and a lateral-to-longitudinal drag ratio of 1.8 to 1. Three 4mm reflective markers were placed on each actuator for tracking. This allowed a circle to be fit to the locations reported by the motion capture system, from which a curvature could be defined for comparison.

To aid in determining the swimmer's displacement, a geometric model was created using the principles described in chapter 2 and simulated using a Matlab interface built around these principles [31,32]. This software, named Geometric System Plotter and further described in Chapter 5, allows the user to input the desired geometric configuration, apply a theoretical gait, and simulate the resulting displacement in addition to other helpful tools based in geometric mechanics.

The model used here assumed a low Reynolds number fluid with a local tangential-to-normal drag ratio of 1.8:1. It also uses an ideal representation of the piecewise snake, assuming the backbone is split exactly in half and each section creates a perfect circular arc with the driven curvature.

Recall that the general equation for the position of a point along a snake backbone with a particular curvature definition is

$$h(s) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}(s) = \int_0^s \begin{bmatrix} R(\theta(\sigma)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \kappa(\alpha,\sigma) \end{bmatrix} d\sigma. \tag{2.3 revisited}$$

For a piecewise continuous curvature actuator, κ is defined by the input α and the position as a function of the associated actuator,

$$\kappa(\alpha, s) = \alpha_1 H(-s) + \alpha_2 H(s). \tag{3.1}$$

The use of the Heaviside function $H(s)$ here allows a single expression instead of a case-dependent set of equations. When the curvature of a particular point along the backbone is being evaluated, this classic step function will negate the alpha input on the opposite half.

The resulting constraint curvature function for the piecewise swimmer is shown in figure 3.4. It has similar characteristics to the Purcell swimmer's CCF plot (in 2.4). Both have a negatively-oriented well around the center and are symmetric about $\alpha_1 = \alpha_2$ and $\alpha_1 = -\alpha_2$

3.1.3 Testing and Motion Capture Results

To gain a wider understanding of the resultant displacement, the snake was run through five different gaits. Four were ellipses in the shape space, whose major axes were oriented towards either $\pi/4$ or $3\pi/4$ and whose major-to-minor axis ratio was either 2 or 4. The last gait was a circle with radius 2π. These gaits are shown in figure 3.5.

As shown in figure 3.6, all gaits run on the extensible snake in granular media performed better than the inextensible theoretical model predicted. Additionally, when curvature-based gaits were extracted from the motion capture data, the measured

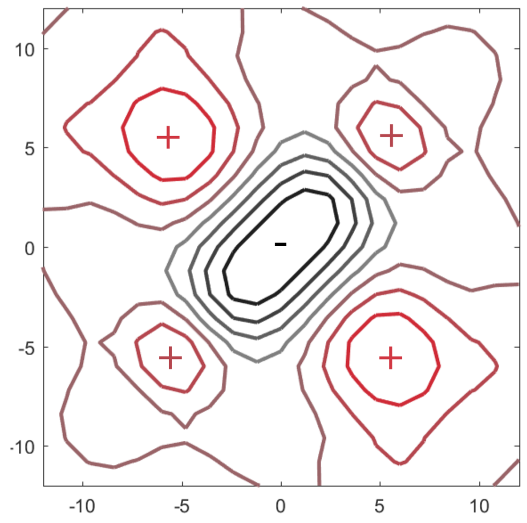


Figure 3.4: Constraint Curvature Function contour plot for the piecewise swimmer dynamic model. Sign definite regions are labeled as positive or negative, with red showing above-zero values, black showing below-zero values, and gray representing the zero contour.

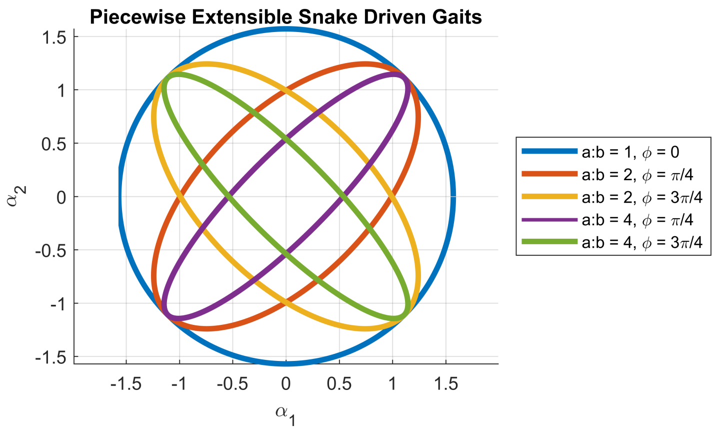


Figure 3.5: Piecewise Swimmer driven gaits.

displacement was again higher than predicted when normalizing for body length and cycle time. Based on these results, we concluded the inextensible-viscous theoretical model was not sufficient to provide accurate predictions for the physical-granular soft snake. Although similar trends are visible between paths, such as the period of the apparent motion, the displacement prediction was not consistent or accurate.

3.2 Dynamic Model Modification

Discrepancies between the piecewise swimmer's expected motion and the motion seen in experiments prompted an additional investigation. In particular, while the swimmer did achieve a net displacement, it did so while stretching tangentially by a non-negligible amount. This highlighted an inaccuracy in the predictive model, though it was still adequate for rough estimates. An example of the observed stretch is shown in figure 3.7, where the original actuator is compared to an actuator manufactured later with a rigid backbone embedded during the molding process.

The length change experienced by any segment of the backbone appeared to be directly coupled to the input pressure, and therefor curvature. The predictive model assumed a static length, thus was not accurate to the behavior of the actuators. To accommodate this, we first consider the portion of the original backbone definition given in (2.3) which considers local rotations,

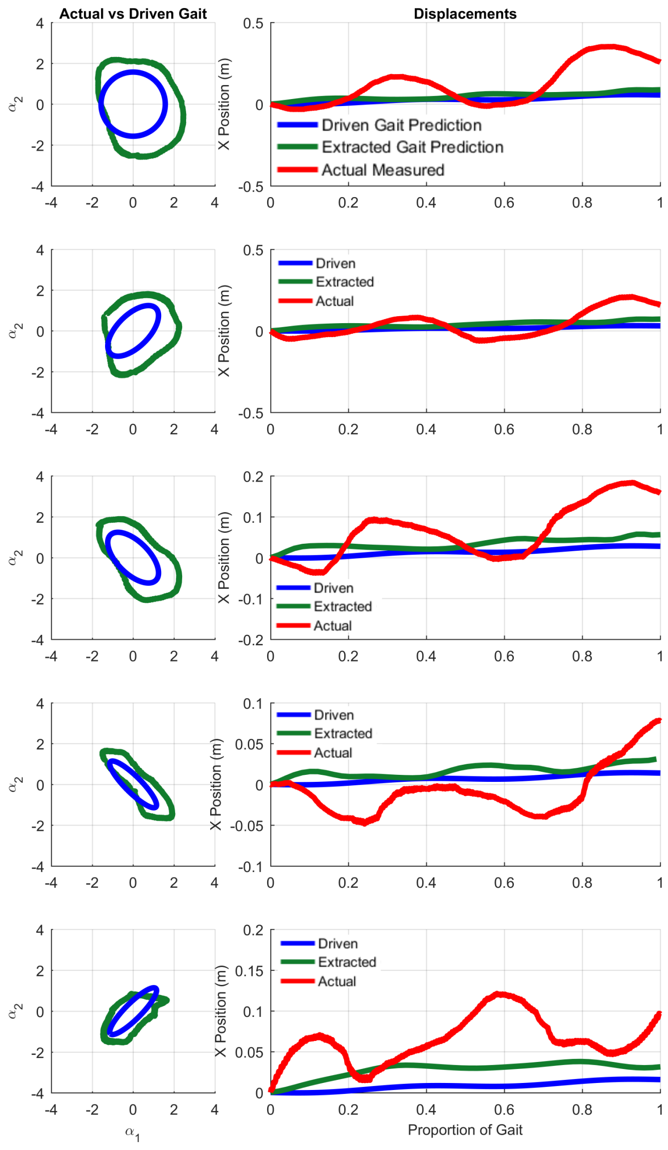


Figure 3.6: Gaits and displacement predictions for each test. Displacements have been normalized for body length, and are displayed corresponding to the proportion of a full gait cycle performed.

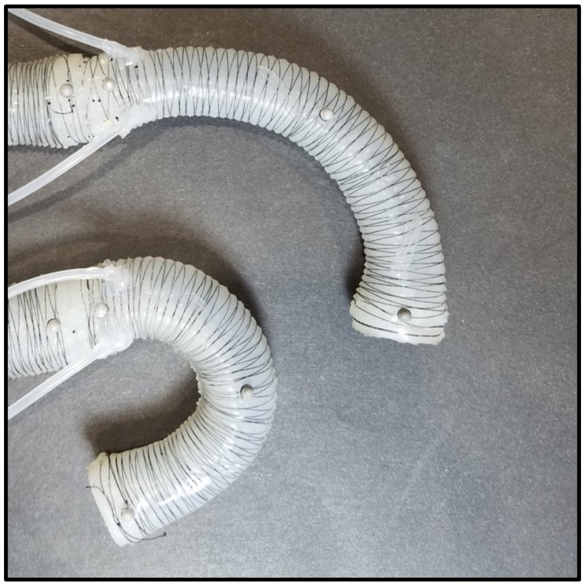


Figure 3.7: Extension in the original actuator (top), and an actuator manufactured later whose extension is restricted with an embedded rigid backbone. Both actuators are inflated with the same pressure.

$$\begin{bmatrix} R(\theta(s)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ \kappa(\alpha, s) \end{bmatrix}. \tag{3.2}$$

The right-hand matrix is modified by the left according to some rotation matrix $R(\theta)$, assuming a fixed unit integration speed and orientation $\kappa(\alpha, s)$. This unit ds is valid in systems with unchanging lengths. However, this is no longer the case for the extensible piecewise swimmer.

Instead of this backbone model assuming unit integration, we define a proportional elongation term, $\lambda(\alpha, s)$ to replace it,

$$\lambda(\alpha, s) = f(\alpha, s). \tag{3.3}$$

Elongation here is a proportional term relating the change in length of each segment to the original unit length using the curvature at that segment. The relationship is defined through a selected function which depends on the nature of the materials chosen.

This gives a new backbone equation,

$$h(s) = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix}(s) = \int_0^s \begin{bmatrix} R(\theta(\sigma)) & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \lambda(\alpha, \sigma) \\ 0 \\ \kappa(\alpha, \sigma) \end{bmatrix} d\sigma. \tag{3.4}$$

Additionally, the forces experienced by the simulated backbone in a low Reynolds number fluid, previously given in (2.19), must also be modified by the proportional length term,

$$F_{b,total} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \int_{-0.5}^{0.5} F_b \lambda(s) ds. \tag{3.5}$$

With these modifications to the basic dynamic model, the constraint equations defining this system's local connection can be re-evaluated for simulation.

3.3 Simulation

To validate the new model, several systems were created and simulated with variations to the functions defining curvature κ, elongation λ, and the magnitude of the function within λ, here defined as the elongation constant b.

Recall the general equation for the curvature which defines a system's backbone shape, as in (2.4),

$$\kappa(\alpha, s) = \kappa_1\alpha_1 + \kappa_2\alpha_2. \tag{3.6}$$

As stated in (3.1), the values for κ_1 and κ_2 are each a Heaviside function $H(s)$, allowing each input variable α_1 and α_2 to be mapped to one half of the backbone.

For comparison, a general serpenoid system was also simulated with included extensibility. The curvature values κ_1 and κ_2 for this system are $\cos(2\pi s)$ and $\sin(2\pi s)$ respectively, giving

$$\kappa(\alpha, s) = \alpha_1 \cos(2\pi s) + \alpha_2 \sin(2\pi s). \tag{3.7}$$

Both of these curvature functions are scale-normalized with $\kappa = \frac{d\theta}{ds}$, which allows the backbone's length to extend at the same time as the curvature increases. This results in the backbone length increasing proportionally at the same time as the curvature, so the scale of the shape increases rather than the backbone curling tighter on itself.

Extracting length over time for the piecewise snake motion capture tests consisted of multiplying the radius of a circle fit to the three evenly spaced markers on each

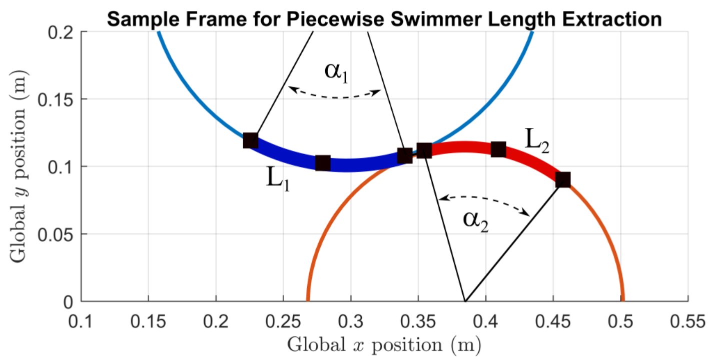


Figure 3.8: Snapshot of piecewise soft snake while executing a gait. The black squares show the locations of the reflective markers tracked by the motion capture system. Length of each actuator is calculated by extracting the radii of circles fit to three markers and multiplying by the angle α between the outermost markers.

actuator with the angle between the outermost markers. The curvature for each actuator was calculated as the inverse of the radius, $1/r$. An example configuration with fitted circles can be seen in figure 3.9. Comparing total arc length with curvature suggested a quadratic relationship between actuator length and driven curvature, shown in figure 3.9 [6]. Based on this behavior, the elongation function in (3.3) should always be positive, continuous, and have a y-intercept of 1. Overlaid on the length response is a manually selected quadratic curve which appears to be the best-fit for both systems. This was chosen as the first function for testing,

$$\lambda(\alpha, s) = b(\alpha(s)^2) + 1. \tag{3.8}$$

However, it is important to note that this relationship depends heavily on the material used and the input variable considered during comparison. Theoretically, the

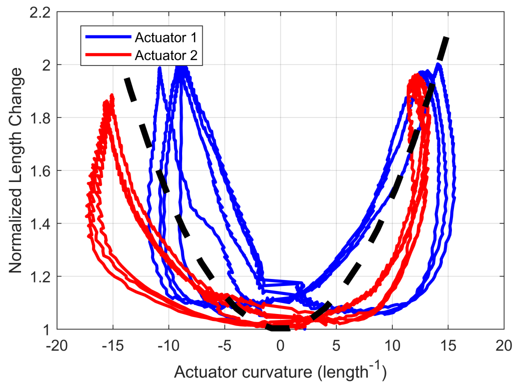


Figure 3.9: Length versus input curvature for the two actuators which comprise a full soft snake executing a circle gait. Each appears to have slightly different elongation responses. This is expected, as the actuators were manufactured by hand and likely had variations in wall thickness. Overlaid in dashed black is a manually selected best-fit line with the equation $\lambda = 0.005(\alpha) + 1$.

elongation function could have any form, provided the actuator still meets the requirements set previously. In the interest of examining multiple functions, a second elongation was selected which follows an absolute value relationship,

$$\lambda(\alpha, s) = b|\alpha(s)| + 1. \tag{3.9}$$

Both selections for elongation have an undefined elongation constant b. This was meant to provide a simple means to increase the magnitude of the elongation effect, equivalent to comparing stiff and soft silicone. The value of b was varied over a range such that at its minimum value of zero the system would revert to inextensible, and at its maximum, the elongation function would return a value of 2. For the quadratic relationship, this set of values is between 0 and 0.101, and for the absolute value relationship, the values are between 0 and 0.318. Twenty iterations of elongation constant were tested for each, resulting in 80 individual systems.

Each system was simulated with an associated optimal gait. A summary of this optimization process is described briefly in §2 and in its entirety in [23].

All systems were simulated in batches with one set of elongation and curvature definitions and the elongation constant ranging from minimum to maximum. Within one batch of varying extensibility constants, only the first system's optimization used a defined seed gait, while each subsequent system used the previous system's optimal gait as a seed. This was an effort to save computation time, as each system's optimal gait should be similar to a system whose extensibility only varied slightly.

For comparison, the total backbone length over time was extracted for each system and an average length was obtained by taking the root-mean-square (RMS) of this array. The average length was assigned to an inextensible system with the same combination of curvature and elongation functions. Each new system then underwent the same optimization procedure to extract an optimal gait.

Once all systems' simulations were completed, the resulting displacement and cost for a single gait cycle were extracted and plotted against the extensibility constant.

3.4 Results

Completing all simulations yielded data for eight distinct system types; one for each combination of serpenoid vs. piecewise curvature, absolute λ versus quadratic λ elongation, and unit- versus RMS-length.

Illustrating the CCF plots as the extensibility constant slowly increases reveals a

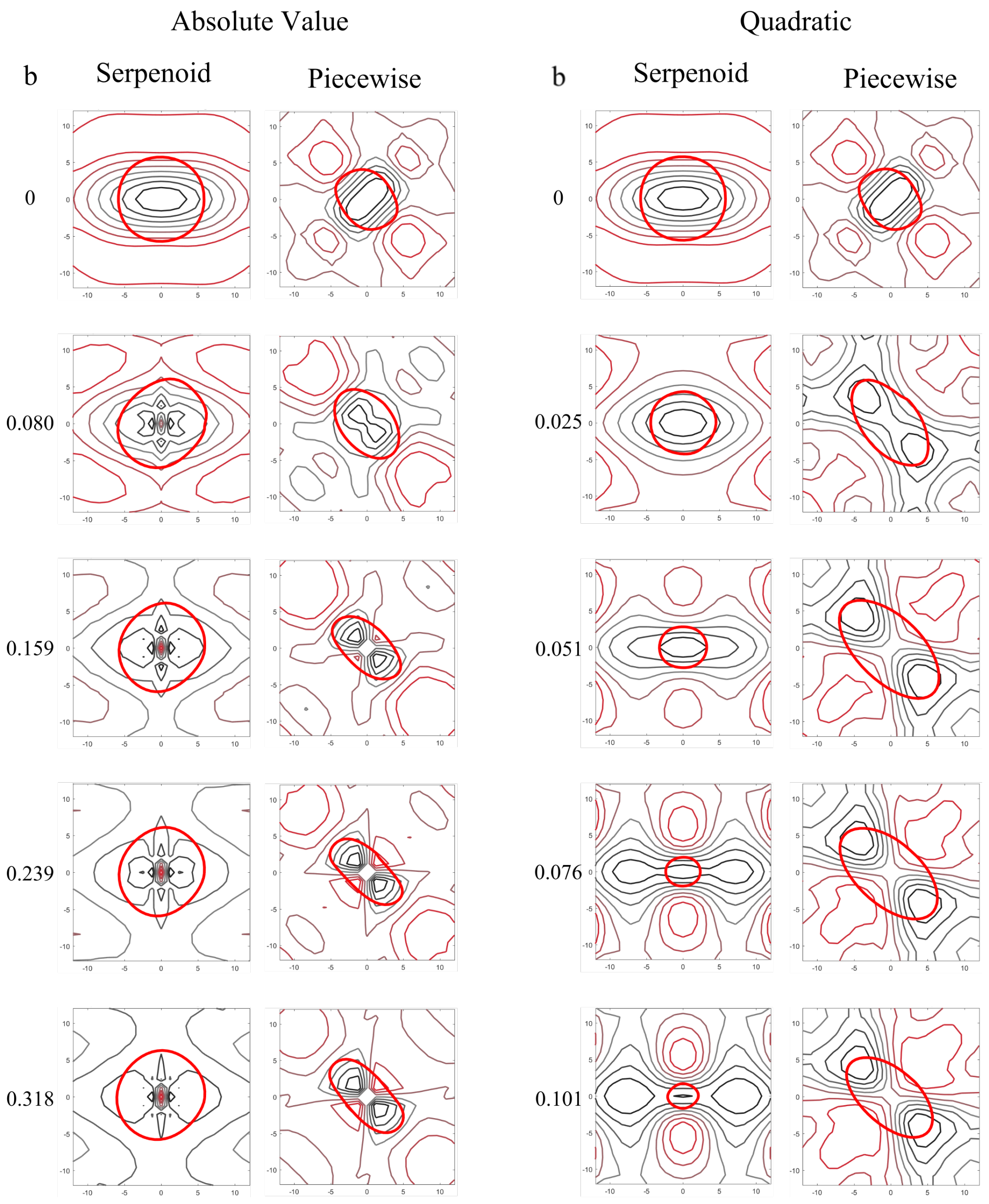


Figure 3.10: Constraint Curvature Function plots with the corresponding optimal gait overlaid. Five samples are displayed for each combination of curvature and elongation functions.

pattern which suggests a higher displacement is possible with optimal gaits and only a small increase in extensibility.

Beginning with small b values, the maximum and minimum values of the CCF plots increase in magnitude, and the CCF plot contours tend to grow more circular, increasing the amount of sign-definite area encompassed by the gait. The optimal gait calculated for each system follows these peaks as they shift within the shape space. These effects can be seen in figure 3.10.

Figure 3.11 shows the extracted displacement and path length cost for each system and its maximum-efficiency gait for all b values. The relative efficiency of each individual system can be found by calculating the slope between the origin and the corresponding data point. Collectively, these slopes allow comparison across all systems of the magnitude of displacement and input energy, shown in figure 3.12.

One interesting property shown in these plots is the immediate decrease in efficiency for the serpenoid systems, regardless of which function is used for extensibility. Conversely, although the piecewise backbone initially had almost half the efficiency of the serpenoid, allowing for extensibility caused the efficiency to rise continuously for the absolute value λ function and rapidly for the quadratic.

The piecewise systems peaked efficiency at $b = 0.03$ and $b = 0.318$ for the quadratic and absolute value curvatures respectively, while the equivalent RMS length systems dropped similarly in efficiency and did not rise beyond the inextensible system.

The serpenoid systems experienced the opposite effect. While increasing the overall length of the system to the RMS length equivalent did have a positive effect on efficiency, the extensible systems performed poorly in comparison.

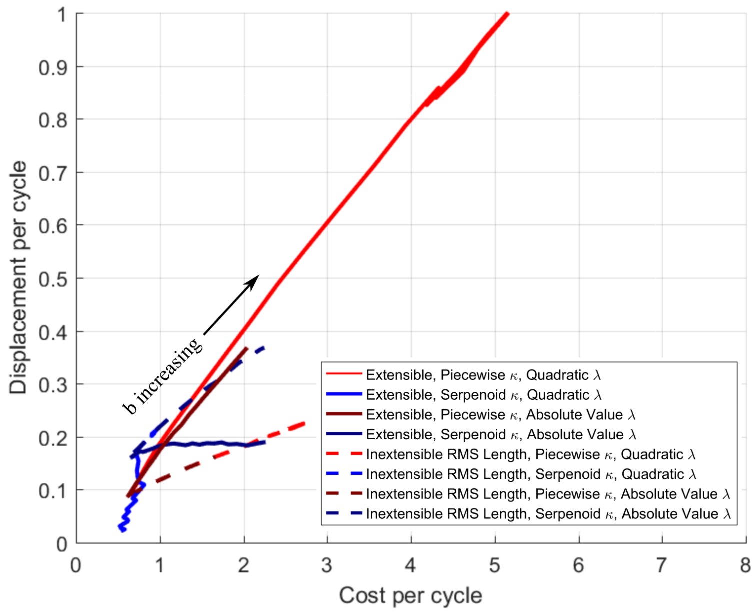


Figure 3.11: Displacement per Cycle versus Cost per Cycle for each system as the elongation constant b increases.

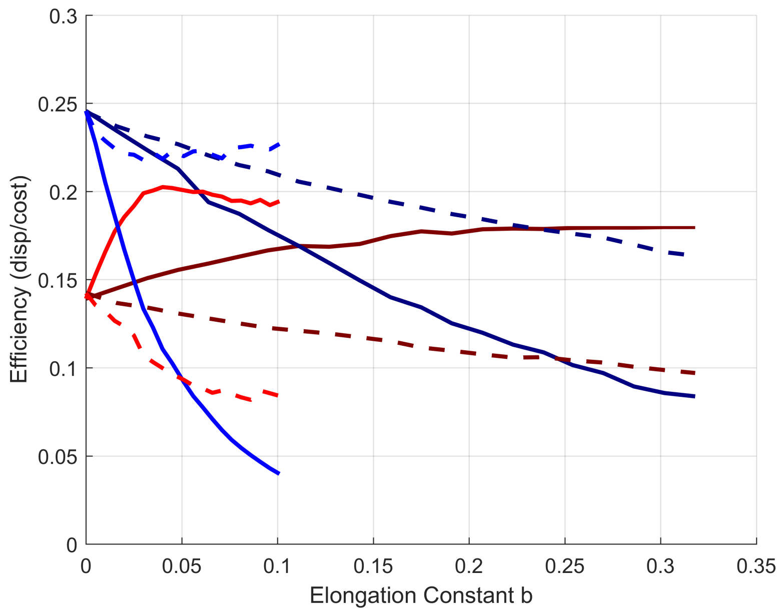


Figure 3.12: Efficiency of each system at associated elongation constant b.

Chapter 4: Data-Driven Local Connection

When generating a geometric or dynamic model for a complex system, many assumptions must be made about the system or its interactions with the world. While some error associated with model inaccuracies can be accepted or even somewhat accounted for, there are often discrepancies whose causes can only be guessed at.

Some evidence to this is the prior chapter concerning motion prediction of a soft snake robot. It is often difficult to accurately predict motion for rigidly assembled robots; the piecewise swimmer, manufactured entirely from compliant materials, poses an even greater challenge. Even with the best estimates concerning the effects input pressure has on output motion to generate a theoretically ideal model, the resulting displacement still varied from what was expected.

These difficulties stem from the amount of detail that can be accounted for when creating geometric models. Simple systems, such as the Purcell swimmer, have relatively simple physics models with minimum assumptions. Unfortunately, most interesting models are not simple. For example, although the dynamics for a 3-link kinematic snake, which is constrained by a set of passive wheels at the center of each link, forbid its wheels from experiencing any local y displacement, a physical incarnation of this snake would likely skid or rotate in some unpredictable manner. Complex dynamics such as surface friction or soft compliant materials can introduce substantial error between predicted and measured data.

In this chapter, we outline an empirical method for generating the local connection of a 2-degree-of-freedom system which bypasses the need for an exact dynamic model.

4.1 Empirical Derivation

Generating the local connection for any given system begins with the equation first shown in (2.2),

$$\overset{\circlearrowright}{g} = A(\alpha)\dot{\alpha}. \tag{4.1}$$

In this work, we are concerned with systems whose shape variables operate in a flat plane and have output positions reported using Cartesian coordinates x, y, and θ. Expanding to show these dimensions, equation 4.1 becomes

$$\begin{bmatrix} \dot{x}^b \\ \dot{y}^b \\ \dot{\theta}^b \end{bmatrix} = \begin{bmatrix} A_{x,\alpha 1} & A_{x,\alpha 2} \\ A_{y,\alpha 1} & A_{y,\alpha 2} \\ A_{\theta,\alpha 1} & A_{\theta,\alpha 2} \end{bmatrix} \begin{bmatrix} \dot{\alpha_1} \\ \dot{\alpha_2} \end{bmatrix}. \tag{4.2}$$

where the coefficients in $A(\alpha)$ each correspond to the effect of one input shape velocity on one output body velocity component.

Traditionally, the coefficients for the local connection are derived from a set of constraint equations which require knowledge of an appropriately linear physics model, as in (2.19).

An alternative route for obtaining the local connection begins with assuming the robot's body velocity $\overset{\circlearrowright}{g}$ and shape inputs α and $\dot{\alpha}$ are known for a set of test motions, allowing for the connection matrix to be directly solved from the relationship between these values.

Consider a system with an unknown local connection which executes a path through the shape space. We can define a discretized grid of query points over the space, where every point $[\alpha_1, \alpha_2]$ will have associated local connection coefficients A_1 and A_2.

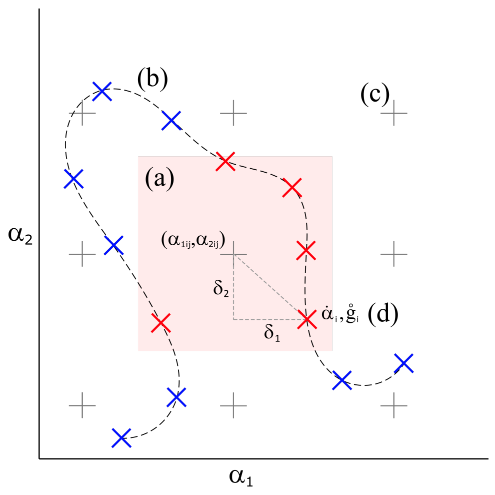
At each coordinate $[\alpha_1, \alpha_2]$, the executed gait is searched for nearest neighbors $p_1 : p_n$ which have an associated body velocity $\dot{x}^b(t)$ and shape velocity $\dot{\alpha}(t)$. Ideally, each point's velocities follow the relationship in the rows of 4.2. For the x-coordinate,

$$\dot{x}^b = \begin{bmatrix} A_{x,\alpha 1} & A_{x,\alpha 2} \end{bmatrix} \begin{bmatrix} \dot{\alpha_1} \\ \dot{\alpha_2} \end{bmatrix}. \tag{4.3}$$

Ideally, every gait point near the current query point follows this relationship with the same A_1, A_2 coefficients. Rearranging slightly, for the nearest points the above becomes

$$\begin{bmatrix} \dot{x}_1^b \\ \dot{x}_2^b \\ \vdots \\ \dot{x}_n^b \end{bmatrix} = \begin{bmatrix} \dot{\alpha}_{1,1} & \dot{\alpha}_{1,2} \\ \dot{\alpha}_{2,1} & \dot{\alpha}_{2,2} \\ \vdots & \\ \dot{\alpha}_{n,1} & \dot{\alpha}_{n,2} \end{bmatrix} \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}. \tag{4.4}$$

With a sufficient number of points, a pseudo-inverse operation can be performed to find the best fit regression of a plane passing through coordinates $(\dot{\alpha}_{n,1}, \dot{\alpha}_{n,2}, \dot{x}_n^b)$. The resulting local connection coefficients for that query point can be stored, and the



(b)
(c)
(a)
α_2
$(\alpha_{1ij}, \alpha_{2ij})$
δ_2
δ_1
$\dot{\alpha}_i, \mathring{g}_i$ (d)
α_1

Figure 4.1: Visual representation of neighborhood point search. (a) Search boundary around current grid point. (b) The dashed line shows an example system gait. Ideally, the entire gait spans shape space with adequate density. Sample points (red and blue crosses) are taken to downsample or discretize data. (c) Grid of query coordinates in shape space. (d) Points within boundary (red crosses) have an associated body velocity, shape velocity, and distance $[\delta_1, \delta_2]$ that are stored for calculation.

same process performed on the next grid location.

Our proposed method expands (4.4) to include higher order terms which define the partial derivative of $A1, A2$ with respect to the change in shape position relative to the grid point of interest, $[\delta_1, \delta_2]$.

For this modified method, the relative positions $[\delta_{p_n 1}, \delta_{p_n 2}]$ of the nearest sample points are extracted in addition to the corresponding body and shape velocity. An illustration of this is shown in figure 4.1. All components including partial differential effects are shown in the following equation, which is an expansion of (4.4).

$$\begin{bmatrix} \dot{x}^b_1 \\ \dot{x}^b_2 \\ \vdots \\ \dot{x}^b_n \end{bmatrix} = \begin{bmatrix} \dot{\alpha}_{1,1} & \dot{\alpha}_{1,2} & \alpha_{1,1}\delta_{1,1} & \alpha_{1,1}\delta_{1,2} & \alpha_{1,2}\delta_{1,1} & \alpha_{1,2}\delta_{1,2} \\ \dot{\alpha}_{2,1} & \dot{\alpha}_{2,2} & \alpha_{2,1}\delta_{2,1} & \alpha_{2,1}\delta_{2,2} & \alpha_{2,2}\delta_{2,1} & \alpha_{2,2}\delta_{2,2} \\ & & & \vdots & & \\ \dot{\alpha}_{n,1} & \dot{\alpha}_{n,2} & \alpha_{n,1}\delta_{n,1} & \alpha_{n,1}\delta_{n,2} & \alpha_{n,2}\delta_{n,1} & \alpha_{n,2}\delta_{n,2} \end{bmatrix} \begin{bmatrix} A_1 \\ A_2 \\ \frac{\partial A_1}{\partial \alpha_1} \\ \frac{\partial A_1}{\partial \alpha_2} \\ \frac{\partial A_2}{\partial \alpha_1} \\ \frac{\partial A_2}{\partial \alpha_2} \end{bmatrix} \tag{4.5}$$

Taking the pseudo-inverse of the $n \times 6$ matrix and multiplying with the x body velocity matrix on the left side leaves a local connection component matrix on the right which is a best fit regression of the selected body and shape velocity data. The first two elements A_1 and A_2 are extracted and saved for the current grid point, to be referenced when performing gait calculations.

We also implement a k-nearest neighbor search which considers the *reciprocal condition number*, a measure from 0 to 1 of how sensitive the solution to a set of equations is to small perturbations in input arguments, when populating (4.5). The number of points considered is defined such that the the reciprocal condition number of the α matrix, $rcond(\alpha \cdot \alpha^T)$, is greater than some threshold, while the maximum distance to the neighboring points is bound below the grid size. The threshold here is set to 0.7, and the number of points k must be at minimum 6, to ensure (4.5) is full rank. Once both conditions are met, the point set is finalized and calculations can continue. A more detailed explanation of this process from a coding perspective is given in §5.2.2.

To obtain an ideal distribution of sample points in the shape-space, samples should

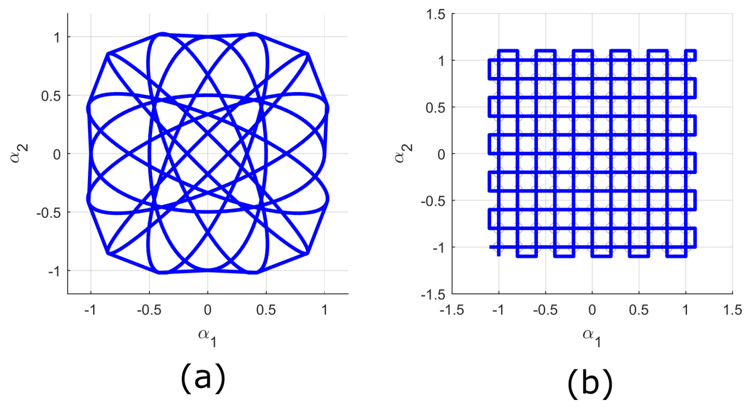


Figure 4.2: Two examples of gaits which adequately span the shape space. (a) An ellipse is rotated while augmenting the major and minor axes to fill the center and corners. (b) Boustrophedon or Lawnmower path.

come from a trajectory which adequately spans the shape-space and its tangent spaces.

One sufficient gait is here referred to as an ellipse sweep, shown in figure 4.2a. It involves taking a single ellipse and rotating it through 2π, while increasing the major axis and decreasing the minor axis when the major is oriented along $\alpha_1 = \alpha_2$ and $\alpha_1 = -\alpha_2$. Another gait, also in figure 4.2b, is a "lawnmower" path which spans the shape space with alternating vertical and horizontal segments. Both these paths are ideal as the resulting grid should provide a well distributed set of sample points whose shape velocities are in orthogonal directions.

4.2 Theoretical Validation

Before applying empirical local connection derivation to a physical system, the methodology was first tested on one of the simulated extensible piecewise systems from the previous chapter.

A lawnmower, or boustrophedon, trajectory was executed for a piecewise quadratic system with extensibility constant $b = 0.05$. Gaussian noise was added to the calculated body velocity and shape velocity to simulate raw motion capture data, and the existing local connection was treated as unknown. The above derivation process was run over the gait and velocities, and the resulting local connection compared to the original as seen in figure 4.3.

This result supported the validity of the process, as well as highlighted some aspects which would be important when evaluating a physical system. One was that the most significant error occurred on the edges. To remedy this, the query grid should be compressed such that the neighborhood search does not extend into areas of the shape space that are devoid of sample points.

4.3 Experimental Setup and Derived Local Connection Results

After theoretical validation using close to perfect data, the expanded methodology for data-driven local connection generation was applied to two distinct physical geometric systems. Both can be compared to existing theoretical models, though variations between these and the experimental results are expected as the theoretical models are idealized and make many assumptions about the system.

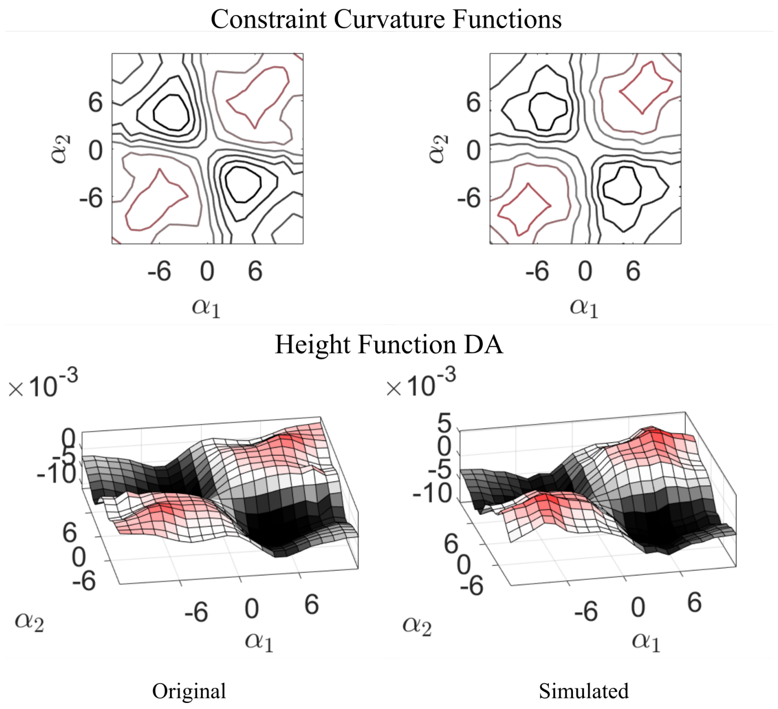


Figure 4.3: Method validation using a previously simulated piecewise system with quadratic elongation and $b = 0.05$. The left hand images are the original theoretical system, and the right shows the result of running empirical local connection derivation on data with added Gaussian noise. All plots are the optimized x solution.

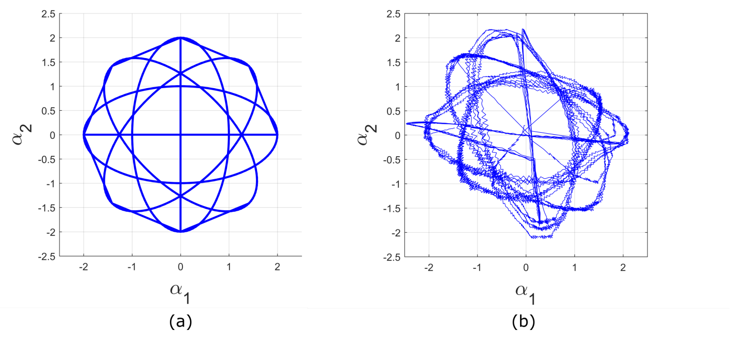


Figure 4.4: (a) Accuracy of the pneumatic solenoids prevented executing a full ellipse sweep, so the driven gait is a combination of four ellipses and one cross. (b) Gait executed by the soft snake in millet.

4.3.1 Piecewise Constant Curvature Swimmer

The soft snake robot built and tested in Chapter 3 was driven under a motion capture setup in the same millet as the original tests with a series of weighted ellipse gaits, shown in figure 4.4a. Gait coordinate values were extracted by fitting planar circles to position data from three markers evenly distributed on each actuator [33].

Running a full ellipse sweep gait proved to be challenging due to the nature of the pneumatic solenoids which drive the actuator inflation. When an actuator is instructed to hold a certain position, the corresponding solenoid remains open, continuously inflating the actuator rather than holding it still. An alternative to the single sweep gait was to repeat the ellipse runs in the original experiments and link them together. The soft snake shape response is shown in 4.4b.

Figure 4.5 shows a snapshot of the fitting process performed on the piecewise snake. Important to note here are the gaps between the gait data. Although the gait does span the shape space, it is not evenly distributed, and the shape velocities do not

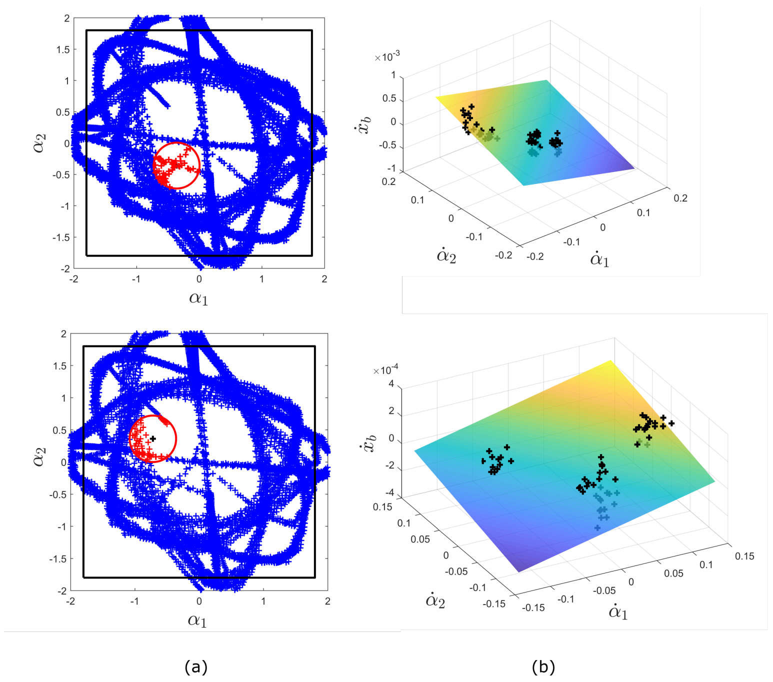


Figure 4.5: Neighborhood search performed on two gait sample points. (a) Points which fall within the k-nearest neighbor search are shown in red. (b) Calculating the local connection coefficients is equivalent to fitting a plane to the body and shape velocity data for these points plotted in 3D.

vary sufficiently over the grid. Sample iterations of the best-fit method can be seen in figure 4.5b, where for two time steps, points within the nearest neighbor search have their shape and body velocities plotted, and a plane is fit to that data whose equation has coefficients A_1 and A_2. The resulting local connection vector fields and CCF plots for x, y, θ in unoptimized and optimized coordinates can be seen in figures 4.6 and 4.7.

It is important to note here that although the scales given for the plots vary due to the units of the variable being considered, there are definite similarities in the basic features of paired plots. These are most evident when viewing the vector plots in unoptimized coordinates. As an example, there are curled portions of the first and third quadrants of the x coordinate unoptimized vector plots. Additionally, the y and θ plot arrows trend towards similar directions.

When evaluated, the body velocity from the original motion capture data and the body velocity predicted by simulation (figure 4.8) do line up nicely. However, as the ellipse evaluated was also used for the connection calculation, the accuracy for this particular path is expected.

4.3.2 Serpenoid Snake Robot

The second system tested using this method is a modular snake robot first created at the Carnegie Mellon University (CMU) Biorobotics Laboratory [4]. This robot consists of several individually actuated servos in series and is capable of a wide array of motions, including climbing, crawling, and sidewinding [5].

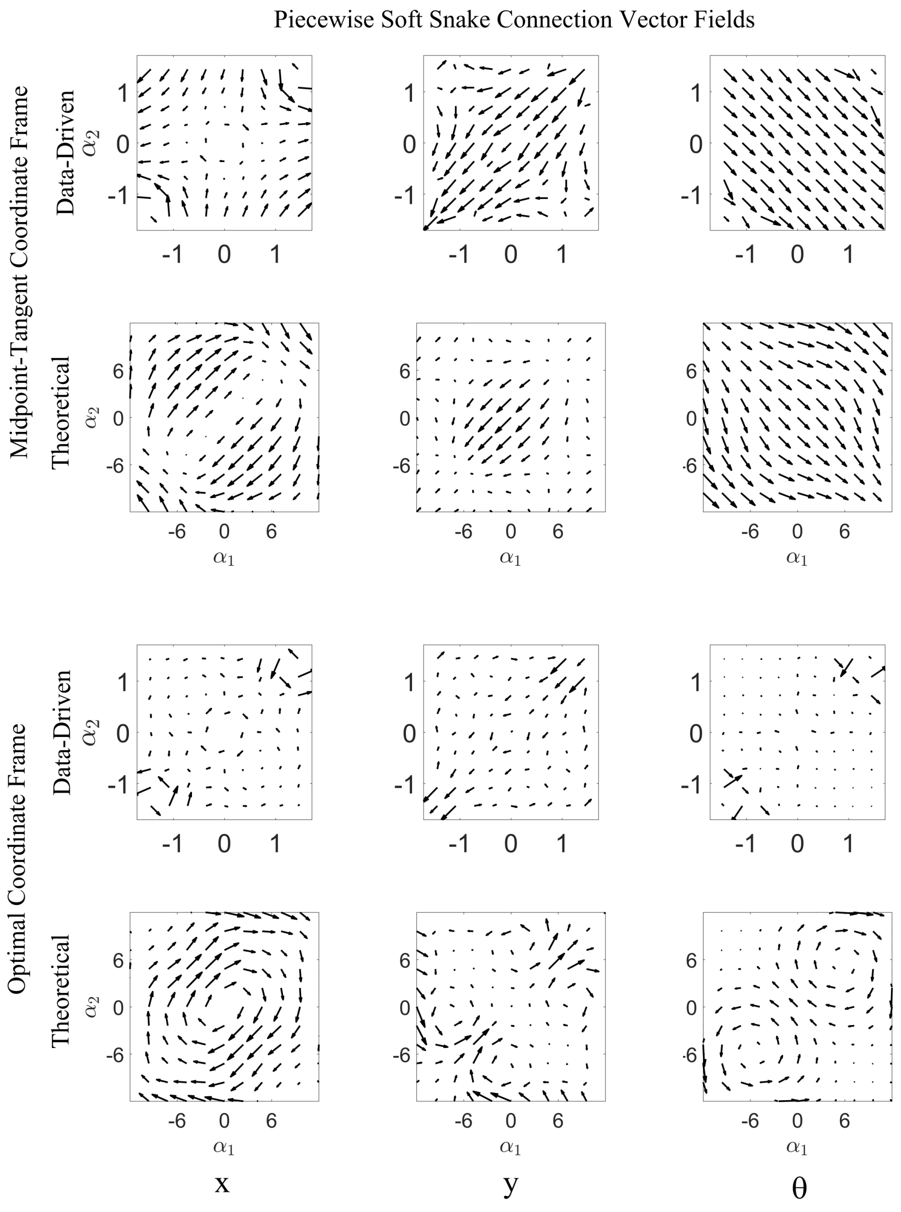


Figure 4.6: Resulting connection vector plots for analysis on the piecewise soft snake compared with its closest theoretical equivalent, a simulated piecewise system with a quadratic extension function and $b = 0.005$. The leftmost plots show data for the x coordinate, the middle is y, and right is θ. The top two rows are displayed in unoptimized coordinates, and the bottom are instead in optimized coordinates. This layout is repeated for later figures and is only listed here.

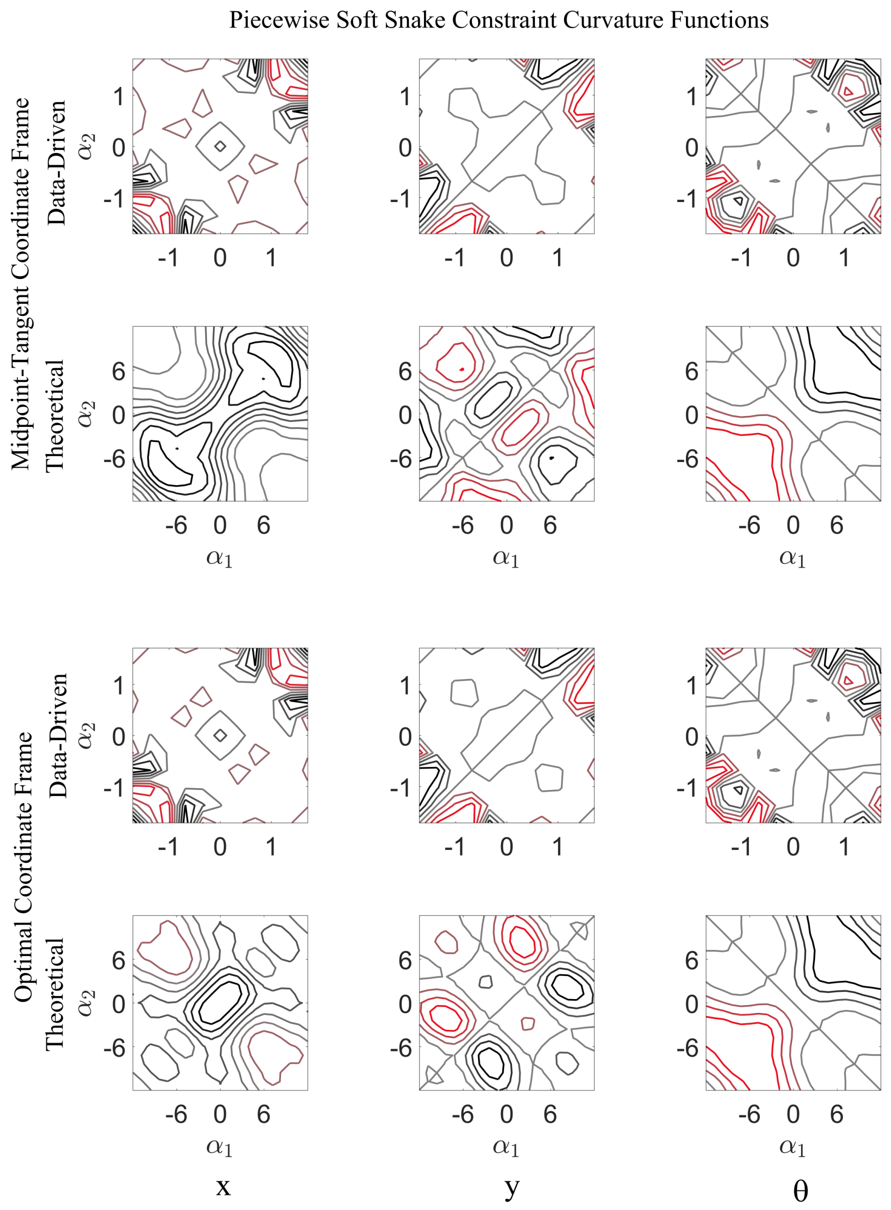


Figure 4.7: Resulting constraint curvature function plots for analysis on the piecewise soft snake compared with its closest theoretical equivalent, the simulated piecewise system with a quadratic extension definition and $b = 0.005$.

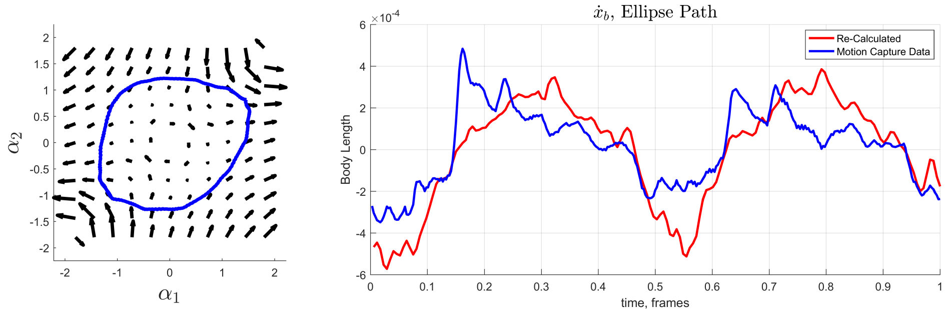


Figure 4.8: Ellipse gait (left) and associated body velocity comparison for the piecewise soft snake. Velocities here match closely because the ellipse was used for the connection calculation.

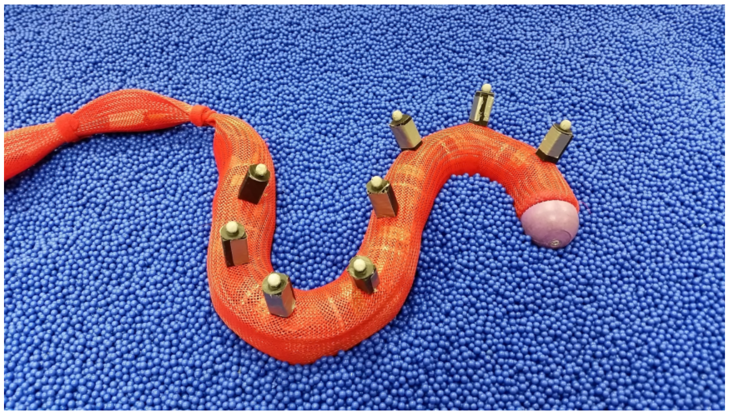


Figure 4.9: CMU modular snake on the testing bed. Each link is tracked via markers attached to small stilts to keep them visible during movement.

For this application, the robot was restricted to planar motion and driven with serpenoid curvature on a bed of uniform spheres roughly 6 mm in diameter. Actuation response was accurate enough to allow for the use of the boustrophedon gait. The driven and extracted gaits are shown in figure 4.10.

The boustrophedon gait was run five separate times. Data from each were appended to the last, so information from all runs could be taken into account and

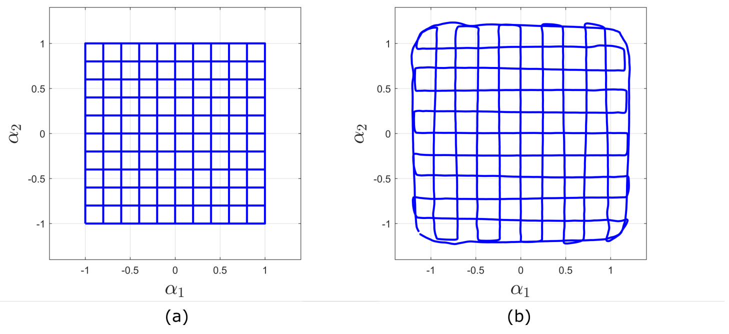


Figure 4.10: (a) Idealized boustrophedon gait and (b) the modular snake's resulting shape path.

increase the number of query points available to the nearest-neighbor search. A snapshot of the search can be seen in figure 4.11, with the resulting local connection displayed as vector plots and CCFs in figures 4.12 and 4.13. The distribution of qualifying points is more widespread for this gait, allowing for a better regression and coefficient calculation. Similarly to the piecewise results, there are equivalent features in the unoptimized x, y, and θ body coordinate plots. The most obvious of these include a rotational trend in the x plot, and a top-to-bottom trend in the θ plot.

To evaluate the accuracy of the empirically derived local connection for the modular snake, single circle and ellipse gaits were also run. Ideally, because the same snake was used for all trials without modifications, the body-to-shape velocity relationship for these gaits is the same as the boustrophedon gait. For one ellipse and circle, the compared velocities are shown in figure 4.14.

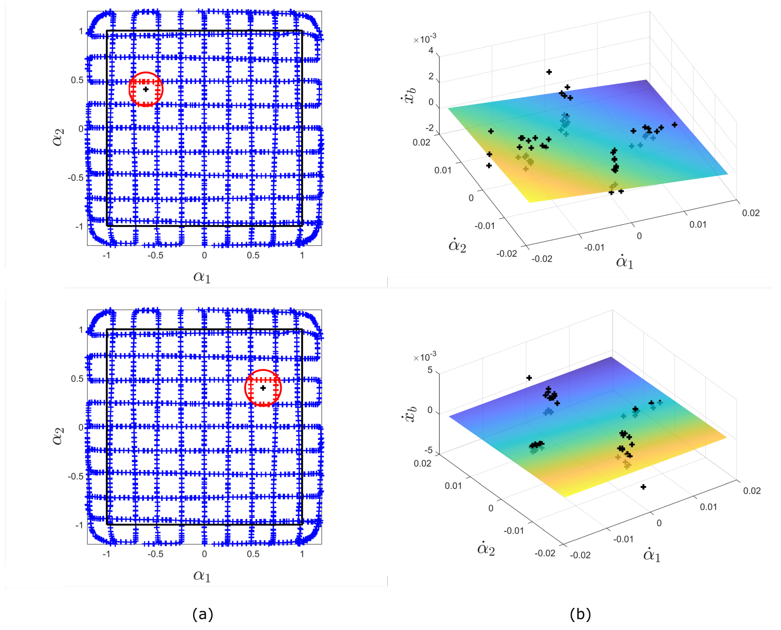


Figure 4.11: (a) Sample neighborhood search over the test gait. (b) Best-fit plane for data within neighborhood search. These points are well distributed in the shape velocity space, so the best-fit process is more accurate.

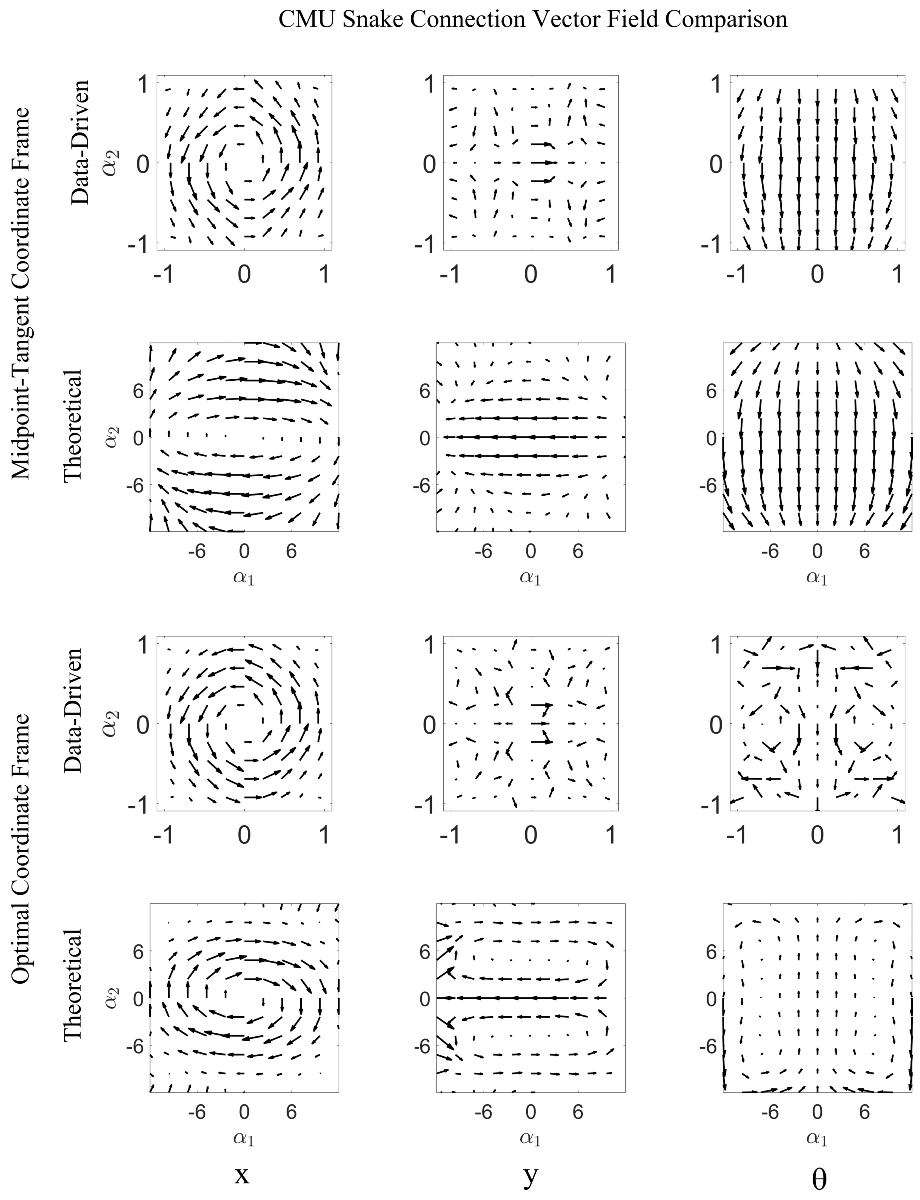


Figure 4.12: Resulting connection vector plots for analysis on the CMU modular snake compared with its closest theoretical equivalent, the serpenoid viscous swimmer.

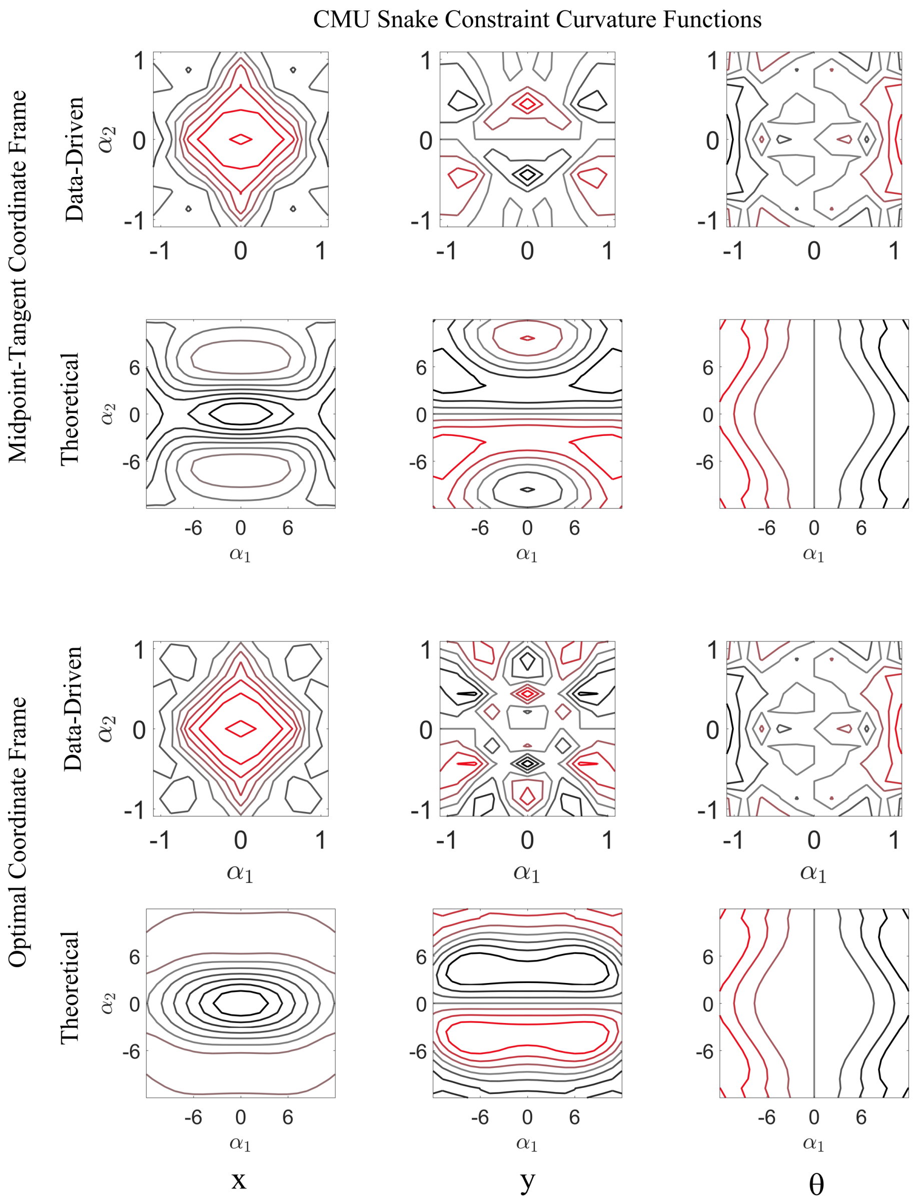


Figure 4.13: Resulting constraint curvature function plots for analysis on the CMU modular snake compared with its closest theoretical equivalent, the serpenoid viscous swimmer.

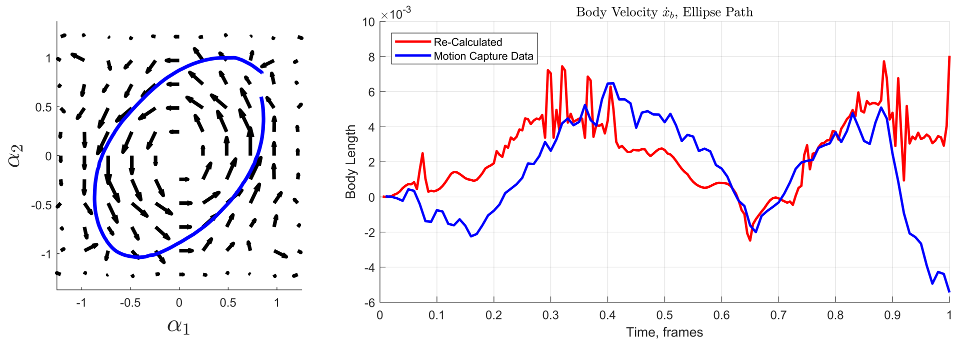


Figure 4.14: Ellipse and circle gait executed by the CMU snake. The paths, shown in blue in the left images, are overlaid on a connection vector plot (top) and a constraint curvature contour plot (bottom) in unoptimized coordinates. In the rightmost plots, the red line shows the predicted body velocity based on the empirically derived local connection, while the blue line shows the original motion capture body velocity.

Chapter 5: Geometric System Plotter

Much of this work depended on a MATLAB® graphical user interface (GUI) called Geometric System Plotter. This GUI, colloquially referred to as Sysplotter, is a collection of functions which perform many of the calculations described in §1.2. The goal of Sysplotter is to present common geometric mechanics tools in such a way that little background knowledge is required for efficient use, though more complicated analysis is available for those who are familiar with the relevant theory. It was originally conceived by Ross L. Hatton, and has been continually added to as new capabilities are explored [32].

Here we will describe the basic structure behind the current interface and how the contributions outlined in this thesis were integrated for use.

5.1 Software Package

Geometric System Plotter is a collection of functions accessible through an interface which perform calculations required for many system and gait evaluation tools made possible through geometric mechanics. These tools simulate geometric systems with or without shape changes and store the data for future use. Figure 5.1 shows the interface at startup, before any options are selected.

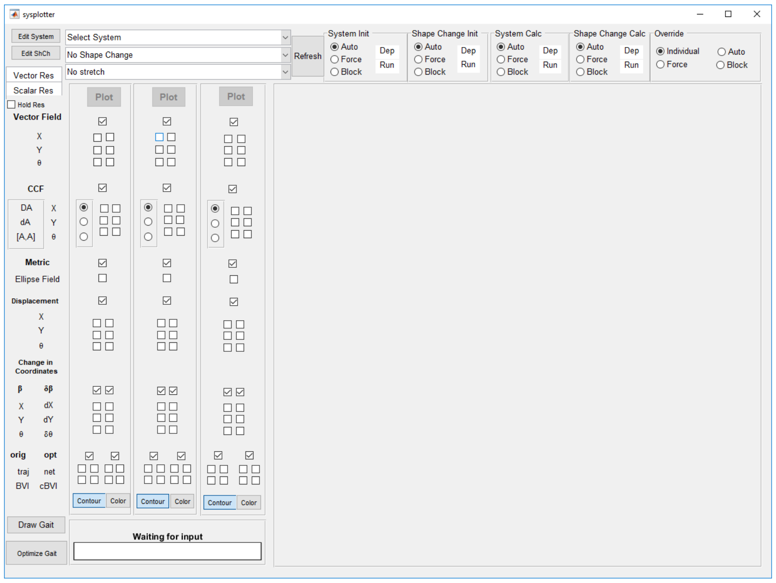


Figure 5.1: Geometric System Plotter as it appears in Windows 10 when first opened. It was created in the Graphical User Interface Development Environment (GUIDE), a MATLAB® drag-and-drop interface builder.

5.1.1 System and Shape Definitions

System and Shape files are what drive most processes in Sysplotter. Both have several input cases which can be called by other files and reference many of the existing functions.

System files contain fields for the backbone locus and force constraint equations (see §2.1.1 and §2.1.2), as well as various simulation parameters such as calculation density and dependent variables.

All system files have the same general components:

- *Display Name*: Text which will be displayed in the system drop-down menu. This is different from the file name.

- *Dependencies*: Files which are monitored for modifications. If any in this list have a 'date modified' time stamp after the stored reference date, the reference is updated and the associated file is re-initialized.

- *Initialize*: Define and calculate all system parameters. These include:

  - *Local Connection*: Functional representation which defines the Local Connection as a function of the shape variables for the given system. Forces and dynamic constraints are also encoded here.
  - *Metric*: Functional representation of cost of movement for the given system. This may also encode force and dynamic constraints if needed.
  - *Processing Details*: Parameters for display and data calculation, such as the grid range, calculation density, and plot axis options.

Shape change files are constructed in a similar manner. However, Instead of encoding the dynamic constraints of the system, they define the gait which the system will execute as a function of time and the same input variables encoded in the system's local connection.

The parameters defined in these files are as follows:

- *Display Name*: Text which will be displayed in the system drop-down menu. This is different from the file name.

- *Dependencies*: Files which are monitored for modifications.

- *Path Parameters*: Variables concerning the shape variables as a function of time:

- *Path Function*: Time-dependent function which outputs the shape values for t.
- *Run Time*: Total time to run gait.
- *Display Options*: These include directional arrow locations, number of arrows to plot, and gait path resolution.

Data for the system and shape change are stored under a combined name for later reference. If the same system-shape change combination is selected in the interface and none of the dependencies referenced by the files are activated, Sysplotter recognizes that the data already exists and forgoes further calculations to save time and resources.

5.1.2 Display Tools

There are several display tools available in Sysplotter for calculated data. The background for each can be found in §1.2, but will be briefly restated here. All of the following example figures reference the Purcell Swimmer and its maximum efficiency gaits in optimal coordinates.

The first tool is a vector field representation of the local connection. This considers a grid of shape values whose parameters are defined in the system file and calculates the local connection matrix at each coordinate. For a system with two input variables operating in $SE(2)$, the local connection is a 3×2 matrix, with each column corresponding to a shape variable and each row corresponding to a position variable. Depending on the position variable selected, the vector plot displayed will reflect the effect each shape variable has on that position coordinate.

Sysplotter can also display the Constraint Curvature Function (CCF). This calculates the height function DA over the shape space grid as the summation of the curl of the local connection dA and the lie bracket effect $[A_1, A_2]$ The definition for this is shown in equation (2.25). Each element is available for plotting as an isocline contour or surface representation. Examples for this and the connection vector plots were shown in chapter 2, figures 2.2 and 2.3.

When a gait is selected, the displacement in x, y, and/or θ can also be plotted as a function of time. The gait is also overlaid on any selected vector or CCF plots. The trajectory of the body coordinate system, net displacement, body velocity integral estimate, and corrected body velocity integral estimate can also be displayed in world coordinates.

It has been shown that the use of world coordinates in displacement estimation can involve significant error [26]. Sysplotter offers alternative display coordinates, referred to as the *optimal coordinate frame*. This reorients the body frame such that it experiences minimum perturbation from changes in the system shape. Vector plots, CCFs, displacement estimates, and trajectories can all be displayed in optimal coordinates.

Finally, Sysplotter includes a power metric calculation for each system. This metric, fully explained in [28], modifies the shape space to display the selected data such that movement in any direction requires the same cost.

5.2 Contributed Files

This section will outline additional files created solely for work done in this thesis, as well as modifications to existing features which improved Sysplotter's functionality.

5.2.1 Extensibility

To aid simulation of extensible systems in chapter 3, several files were created and integrated into Sysplotter's default package.

Although Sysplotter already included a force model for geometric swimmers submerged in a low Reynolds fluid, it assumed the system only articulates at defined joints and is fully inextensible. In order to perform accurate calculations, the existing code outlining (2.19) was modified to instead follow equation (3.5).

Next, system files were created which included extensibility as a curvature-dependent variable. This way, whenever the force model calls the position of a particular location along the backbone, it defines both the local curvature and extension. The systems considered in this section were a traditional serpenoid model (3.1) and a piecewise constant curvature model (3.7).

5.2.2 Data Driven Local Connection

As opposed to including extensibility terms in existing files, calculating a local connection based on motion capture data required creation of several tools from scratch.

First, any data from a motion capture setup must be converted to an appropriate form for remaining calculations. This must be started on the user's end, as every data collection setup varies slightly. The resulting data must be arranged in $m \times n$ matrices, one for each dimensions measured, where m is the number of time steps and n is the number of markers tracked by the camera system. At this point, the order of the axes and markers does not matter.

After the data is appropriately arranged, it can be run through a data cleaner. This file first plots all marker points at a sample time stamp, and asks the user if the axes are aligned properly. If not, the correct order is entered and the data is rearranged. Next, the same sample marker points are plotted with the current marker order as labels, and the user is asked again if the order is as expected. This is an important step, as many later calculations require calling particular markers; i.e., our piecewise swimmer model assumes a minimum of three markers on each half of the snake, and will call the first half of indices during calculations.

For the work discussed here, the only motion of interest occurs in the $x-y$ plane, while motion capture cameras record all three dimensions. The data cleaner described above is written with versatility in mind and should accommodate future applications which require three-dimensional analysis.

Next, the shape variables for the driven motion must be extracted. While it would be ideal to have the system perfectly perform its driven gait, the reality is this is rarely the case. Therefor, the clean motion capture data is analyzed under the assumption of a given backbone and the shape variables stored for later use. At the same time, the body coordinates $g(t) = [x, y, \theta](t)$ are stored. These coordinates are rotated with respect to the original orientation in the motion capture frame.

Body velocity $\overset{\circ}{\vec{g}}$ for each coordinate in g and shape velocity $\dot{\alpha}$ are obtained by taking a time-independent gradient of the respective data.

For each body coordinate, the shape values at each time frame are linked to their respective shape velocity and body velocity. A grid point in the shape space is selected and a k-nearest neighbor search is performed over the space. The number of points required for an accurate calculation must be greater than the number of output variables in the regression solution matrix. In this case, because we are calculating the local connection coefficients as well as gradient terms for each with respect to both inputs, that number is six. The nearest neighbor search selects the k nearest neighbors, while monitoring distance to the furthest point. If this range falls beneath the width of the grid window, k is increased by 1 and the search performed again. Otherwise, the process ends.

All points which lie within the search range are then used to populate equation (4.5) and perform regression calculations. This process is repeated for each body coordinate over the entire queried grid.

The system files in Geometric System Plotter are structured to allow multiple input styles for the local connection. While a typical model defines its local connection via force and dynamics functions, it is also feasible to obtain the local connection directly from a stored data file, forgoing resource-intense calculations. This is where the data-driven local connection is called for simulation calculations.

5.2.3 Other tools

Some processes not directly related to geometric swimmer locomotion but useful nonetheless were also included in Sysplotter.

In order to efficiently perform simulations across dozens of slightly varying systems, it was necessary to create systems and shape changes from a template and run them programatically instead of manually through the Sysplotter GUI. Interfaces created through GUIDE in Matlab can also be manipulated through the command window. A ‘GUI manipulator’ was created which creates all necessary files, adds them to the Sysplotter repository, runs the optimizer described in [23] to obtain an optimal gait, and simulates the system-shape-change combination.

To confirm each new system performed as expected, a quick animation file was created which takes a system and shape change and displays the backbone behavior through time. The backbone can be displayed in various color schemes, including a curvature-dependent heat map to differentiate between highly positive and highly negative curvatures.

The power metric described in §2 can be illustrated similarly to the local connection over a grid in the shape space. Each grid point is assigned a uniform circle which is then stretched to an ellipse according to the metric at the same coordinate, shown in figure 5.2a. This adds a useful tool for viewing relative cost of movement throughout the shape space. If the metric stretch option is selected at the same time as the metric ellipses, the ellipses almost return to unit circles (figure 5.2b).

Lastly, a GUI property editor was integrated into Sysplotter which considers the

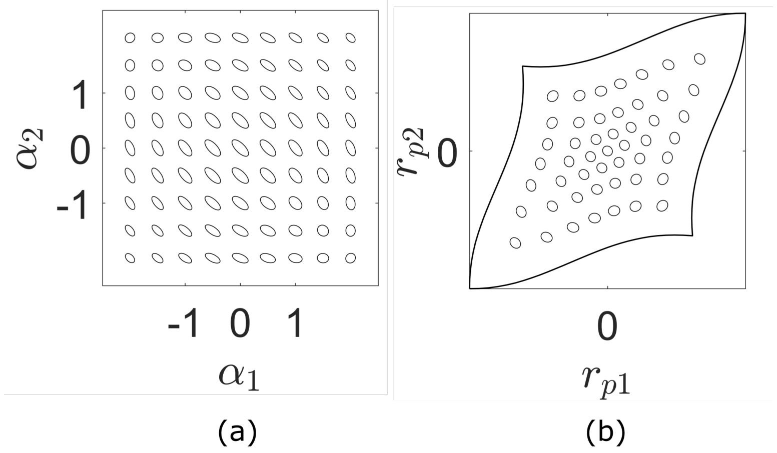


Figure 5.2: Metric ellipses (a) and stretched metric ellipses (b) for the Purcell Swimmer.

operating system on which Matlab is installed and modifies the visual properties automatically upon startup. This is meant to increase ease of use, as many people use different operating systems on their personal computers. Prior to this, several visual characteristics varied drastically between operating systems, including default text font and size, object box sizes and positions, and default window size. While not strictly necessary for performance, slight modifications improved the interface aesthetics. The process for modifying visual properties is also laid out such that a user could manipulate the interface to their liking and save a custom property file as the default. Screen captures of the interface in different operating systems can be seen in figure 5.3.

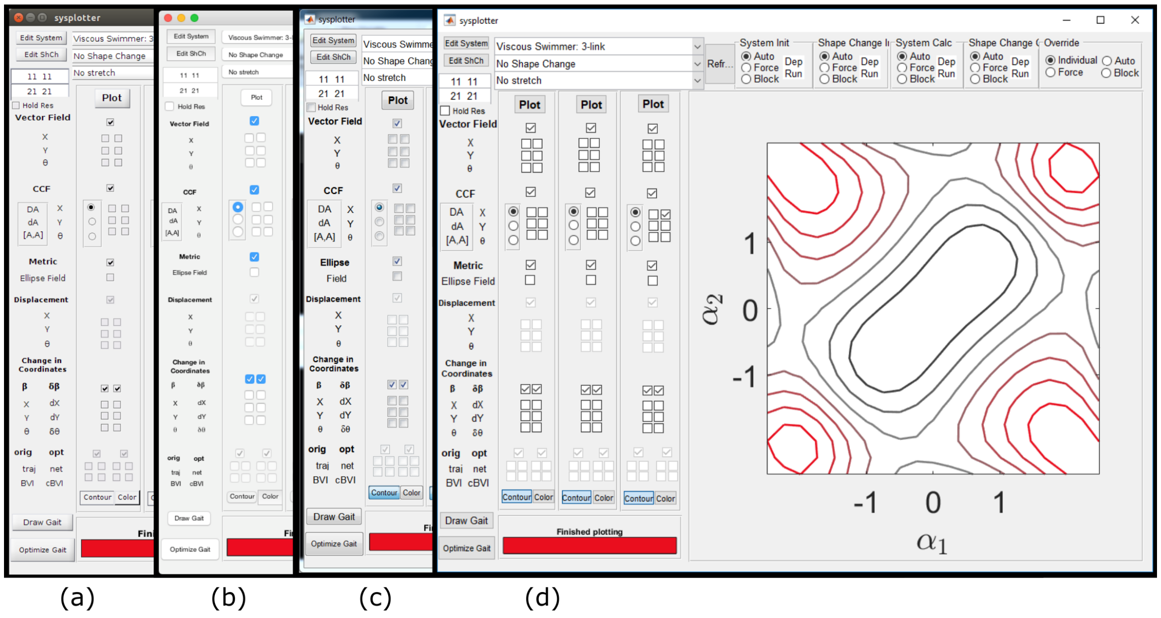


Figure 5.3: Examples of the Sysplotter interface in different operating systems. (a) is Linux, (b) MacOS, (c) Windows 7, and (d) Windows 10. Each has slightly different default texts and button styles which required hand tuning to create a similar experience across platforms. (d) also shows the optimal coordinate x CCF plot for the Purcell swimmer.

Chapter 6: Conclusion

The projects outlined in this work all aimed to improve intuition behind geometric swimmer analysis. Research in this area began with characterizing movement for simple systems, and has gradually expanded to focus on novel methods for gait generation. This resulted in both an increase in the number of tools available for analysis and the types of kinematic models which can be analyzed. We believe this work has contributed knowledge to this area in both respects. First, we described the theory behind a novel piecewise geometric swimmer while employing recent advancements in optimal gait generation. We then investigated an alternative route for model generation in a much broader sense, and proved its versatility with two distinct physical systems. Each provided its own insights and highlighted areas for future work to continue.

To the best of our knowledge, previous work has not considered swimming backbones with extensible geometry. While the inspiration behind the addition of extensibility stemmed from a physical robot, the methodology can be applied to backbones with various curvature models.

Our results suggest that some systems, like the piecewise swimmer, do benefit from the addition of extension. Provided the simulation parameters closely match physical application, this could provide a meaningful starting point for the design of pneumatically actuated soft robots. Conversely, including extensibility in a serpenoid model seemed to cause efficiency to drop rapidly. This implies that for curvature-driven extension, the most efficient model depends on the function driving curvature. The most efficient extensibility for both systems was either zero or close to zero. This could imply curvature-driven extension adds significant cost to movement. If extension was considered its own independent variable and the model considered mass distribution, this method could be adapted for peristaltic or similar motion.

Comparison between actual and predicted displacements suggested discrepancies between the assumed model and the physical snake. The most likely culprit lies in the force distribution. While the simulation assumes a long, slender backbone, the physical system has substantial width and area on its head which interact with the surrounding media. Additionally, although the millet used for experiments has a similar drag ratio to a low Reynolds number fluid, they are fundamentally different media. The soft snake has a relatively low density compared to the millet, and swam only half rather than fully submerged.

Soft robots are becoming more common in applications which require compliance; in particular, where a gentle touch is needed. Although many soft robots are manufactured with flexible yet inextensible components, this makes the process more complex and restricts some of the built-in capabilities of the soft materials. Rectifying discrepancies between the extensible model and manufactured actuator could allow for the model to be used in more practical applications. For example, the displacement of a soft gripper's fingertips could be accurately predicted and compared to the actual displacement for closed loop control applications.

Local connection generation has significant potential as a motion prediction tool.

The method described here is theoretically applicable to any planar system with two inputs. Future work could include expanding this beyond two dimensions and even to several shape inputs.

To realize this, however, improvements on this methodology are required. Most importantly, the system must be capable of executing a gait which adequately fills its shape space. Even the term 'adequately' here is loosely defined. The reciprocal condition number integrated into the k-nearest neighbor search was an initial attempt to ensure enough points were gathered to provide meaningful information. Though this does seem to improve the results, modifying the threshold required for the neighborhood search highlights how varying any of these parameters can drastically change the outcome. The body velocity comparison proved to be useful in this regard, as it provided a true measure of the similarity between models. For a true measure of accuracy, the parameters set for the regression, such as query grid limits, desired point density, and response to outliers, should be investigated on a reliable system.

If these improvements can be realized, this method could ideally be applied to any planar system with two control inputs. Additionally, the procedure for the development of this method may be expanded to include systems which operate in higher dimensions and require more inputs. This could allow for theoretically accurate prediction and control of any robotic system, provided the system can perform a sufficient test gait. This is particularly useful for systems whose basic dynamics are hard to characterize, such as soft robots.

The final portion of this work follows its overall goal; to improve on and provide useful tools for geometric system evaluation. Geometric System Plotter is structured in such a way that it is relatively straightforward to customize models and integrate brand new concepts for a variety of needs. It is our hope that the geometric mechanics research community finds practical use for the tools and methods presented here.

Bibliography

[1] R. L. Hatton and H. Choset, “Optimizing coordinate choice for locomoting systems,” in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010.

[2] D. Tam and A. E. Hosoi, “Optimal Stroke patterns for Purcell’s three-link swimmer,” *Physical Review Letters*, 2007.

[3] E. Purcell, “Life at low Reynolds number,” 1977.

[4] C. Wright, A. Johnson, A. Peck, Z. McCord, A. Naaktgeboren, P. Gianfortoni, M. Gonzalez-Rivero, R. Hatton, and H. Choset, “Design of a modular snake robot,” in *IEEE International Conference on Intelligent Robots and Systems*, 2007.

[5] M. Tesch, K. Lipkin, I. Brown, R. Hatton, A. Peck, J. Rembisz, and H. Choset, “Parameterized and scripted gaits for modular snake robots,” *Advanced Robotics*, 2009.

[6] C. Branyan, C. Fleming, J. Remaley, A. Kothari, K. Tumer, R. L. Hatton, and Y. Git Mengüç, “Soft Snake Robots: Mechanical Design and Geometric Gait Implementation,”

[7] D. Rus and M. T. Tolley, “Design, fabrication and control of soft robots,” 2015.

[8] C. D. Onal and D. Rus, “A modular approach to soft robots,” in *Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics*, 2012.

[9] M. Luo, W. Tao, F. Chen, T. K. Khuu, S. Ozel, and C. D. Onal, “Design improvements and dynamic characterization on fluidic elastomer actuators for a soft robotic snake,” in *IEEE Conference on Technologies for Practical Robot Applications, TePRA*, 2014.

[10] G. Taylor, “Analysis of the Swimming of Microscopic Organisms,” *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 1951.

[11] A. Shapere and F. Wilczek, "Geometry of self-propulsion at low reynolds number," *Journal of Fluid Mechanics*, 1989.

[12] R. M. Murray and S. S. Sastry, "Nonholonomic motion planning. Steering using sinusoids," *IEEE Transactions on Automatic Control*, 1993.

[13] S. D. Kelly and R. M. Murray, "Geometric phases and robotic locomotion," *Journal of Robotic Systems*, 1995.

[14] J. Ostrowski and J. Burdick, "The geometric mechanics of undulatory robotic locomotion," *International Journal of Robotics Research*, 1998.

[15] L. E. Becker, S. A. Koehler, and H. A. Stone, "On self-propulsion of micromachines at low Reynolds number: Purcell's three-link swimmer," *Journal of Fluid Mechanics*, 2003.

[16] E. a. Shammas, H. Choset, and a. a. Rizzi, "Geometric Motion Planning Analysis for Two Classes of Underactuated Mechanical Systems," *The International Journal of Robotics Research*, 2007.

[17] J. E. Avron and O. Raz, "A geometric theory of swimming: Purcell's swimmer and its symmetrized cousin," *New Journal of Physics*, 2008.

[18] R. L. Hatton and H. Choset, "Connection vector fields for underactuated systems," in *Proceedings of the 2nd Biennial IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, BioRob 2008*, 2008.

[19] R. Hatton and H. Choset, "Approximating displacement with the body velocity integral," *Proceedings of Robotics: Science and Systems, Seattle, USA*, 2009.

[20] R. L. Hatton and H. Choset, "Connection Vector Fields and Optimized Coordinates for Swimming Systems at Low and High Reynolds Numbers," in *ASME 2010 Dynamic Systems and Control Conference, Volume 1*, 2010.

[21] R. L. Hatton and H. Choset, "Geometric swimming at low and high Reynolds numbers," *IEEE Transactions on Robotics*, 2013.

[22] R. L. Hatton, T. Dear, and H. Choset, "Kinematic cartography and the efficiency of viscous swimming," *IEEE Transactions on Robotics*, 2017.

[23] S. Ramasamy and R. L. Hatton, "Soap-bubble optimization of gaits," in *2016 IEEE 55th Conference on Decision and Control, CDC 2016*, 2016.

[24] S. Ramasamy and R. L. Hatton, “Geometric gait optimization beyond two dimensions,” in *Proceedings of the American Control Conference*, 2017.

[25] S. Ramasamy and R. L. Hatton, “The geometry of optimal giats for drag-dominated kinematic systems,” *IEEE Transactions on Robotics, conditionally accepted*, 2018.

[26] R. L. Hatton and H. Choset, “Geometric motion planning: The local connection, Stokes’ theorem, and the importance of coordinate choice,” *International Journal of Robotics Research*, 2011.

[27] T. Lipták, M. Kelemen, A. Gmiterko, I. Virgala, and . Miková, “Base Space of Nonholonomic System,” *Journal of Automation and Control*, vol. 4, no. 2, pp. 10–14, 2016.

[28] R. L. Hatton and H. Choset, “Kinematic Cartography for Locomotion at Low Reynolds Numbers,” *Robotics: Science and Systems*, 2011.

[29] S. Hirose, “Biologically Inspired Robots: Snake-Like Locomotors and Manipulators,” *Applied Mechanics Reviews*, 1995.

[30] D. Trivedi, C. D. Rahn, W. M. Kier, and I. D. Walker, “Soft robotics: Biological inspiration, state of the art, and future research,” *Applied Bionics and Biomechanics*, 2008.

[31] MATLAB, *version 9.2.0.538062 (R2017a).* Natick, Massachusetts: The MathWorks Inc., 2017.

[32] R. L. Hatton, “Geometric system plotter.”

[33] D. Malyuta, “fit circle through 3 points.” Available at `https://www.mathworks.com/matlabcentral/fileexchange/57668-fit-circle-through-3-points`, version 1.0.