# AN ABSTRACT OF THE DISSERTATION OF

Nathan Justus for the degree of Doctor of Philosophy in Robotics presented on May 30, 2024.

Title: The Geometry of Passive and Constrained Locomotion

Abstract approved: _____

Ross L. Hatton

There are three major contributions discussed in this thesis:

1. A framework for understanding and optimizing the gaits of inertia-dominated kinematic locomoting systems that make use of simple passive-elastic elements. We specifically focus on mass-dominated systems with only one control input and one uncontrolled shape mode affected by linear spring forces and linear damping. For such systems, we do not have full control of the gait's profile in shape-space and must simultaneously shape the gait trajectory of the active and passive components by varying only the control input signal. We discuss two systems in particular, one that has a discrete passive joint and one that exhibits continuously flexible bending along the tailbone. We show that continuous flexibility enables more efficient locomotion when the dynamic parameters are correctly optimized. We compare these swimmers under the case where we are capable of tuning the elastic parameters of

the system and for cases where we must work with suboptimal elastic properties. We demonstrate how motions for both of these systems can be optimized over three different objective functions. In particular, we separately optimize passive-inertial gaits that maximize swimmer speed, to maximize the energetic efficiency with respect to actuator effort, and to maximize the energetic efficiency with respect to overall system energy expenditure that includes both actuator effort and a metabolic overhead cost. We discuss the use cases and strengths of each of these objective functions and demonstrate that factoring in metabolic cost produces a wide range of gaits useful for efficient locomotion.

2. Simulation and analysis of the locomotion of the sea salp. These chains of jellyfish-like zoids are not well studied and possess interesting modes of locomotion. We model these organisms as chains of links with periodically-firing thrusters that exist in a drag-dominated environment and are connected by passive joints that have some linear stiffness. We show how this simple model of collected individual agents evolves complex structural motion through the fluid media. We discuss the evolution of buckling through the salp and comment on the stability of the salp to certain joint configurations and trajectories through position-space. We also develop an origami thruster prototype constructed from a collapsible origami mechanism that produces thrust when acted on by a series of twisted-and-coiled actuators.

3. A framework for performing system identification on locomoting robots con-

strained to a experimental testbeds, along with a methodology for using the identified system model to generate and optimize motion candidates that can be used to construct a gait library, enabling locomotive control of the unconstrained mobile systems. When constrained, such locomotive systems exhibit different dynamic properties because of the way the constraint changes the system's interaction with the environment. This makes it difficult to use constrained experimental data to make predictions about a free-motion counterpart. We develop a methodology for extracting the principal coefficients that determine system locomotive properties from constrained experimental data, and use these principal coefficients to make predictions about how the unconstrained system will move. We also discuss how this method can affect experimental design by determining whether the desired locomotive coefficients are actually observable given a certain body-motion experiment. We validate this method using constrained and unconstrained experiments involving a swimming robot platform called the AmoeBot developed by our research collaborators at the University of California San Diego. We show that data from constrained AmoeBot experiments can be extracted from the physical context of the experiment and applied to the free-swimming case, allowing for locomotive optimization and control. We show two experimental cases of the AmoeBot being manually piloted using these controls to perform the locomotive tasks of waypoint navigation and trajectory following.

# The Geometry of Passive and Constrained Locomotion

by

Nathan Justus

A DISSERTATION

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Doctor of Philosophy

Presented May 30, 2024
Commencement June 2024

Doctor of Philosophy dissertation of Nathan Justus presented on May 30, 2024.

APPROVED:

---

Major Professor, representing Robotics

---

Associate Dean of Graduate Studies for the College of Engineering

---

Dean of the Graduate School

I understand that my dissertation will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my dissertation to any reader upon request.

---

Nathan Justus, Author

# ACKNOWLEDGEMENTS

Thanks to my family for supporting me and raising me to approach the world with the curiosity to find questions and the confidence and drive to seek answers. Thanks to my friends and lab-mates for their endless patience to be sounding-boards both intellectual and emotional. Thanks to my partner Sam, who's rockin' body has been my anchor in turbulent waters. Thanks to my advisor Ross Hatton for helping mold me into the competent professional that I have become.

# TABLE OF CONTENTS

# TABLE OF CONTENTS (Continued)

# LIST OF FIGURES

# LIST OF FIGURES (Continued)

# LIST OF FIGURES (Continued)

# LIST OF FIGURES (Continued)

# LIST OF FIGURES (Continued)

# LIST OF TABLES

# Chapter 1: Thesis Introduction

An appropriate way to begin discussing the concepts I have covered in my graduate work would be to explain the choice of words in the title of my thesis: "The Geometry of Passive and Constrained Locomotion." Let us begin by doing so.

## 1.1 Why Geometry?

In designing robot motion, it becomes useful to operate on a variety of different mathematical manifolds. By this, I mean that there are multiple types of coordinate systems that can define any particular robotic system, and examining the robot's motion is often a problem of analyzing trajectories and mappings between these coordinate frames [24, 25]. Such a problem will fundamentally rely on the toolset of differential geometry. For an example of such a problem, consider the problem of an inertial double pendulum that has been laid flat along a frictionless horizontal surface and pinned in place through the proximal end of the first link. There are two obvious choices of coordinates that can define the pose of the structure.

The first choice is the Cartesian coordinates for the $(x, y)$ position of each of the two mass elements on the surface. This frame is useful because it is where the system's inertial physics live, making it convenient to quickly calculate useful

properties such as system kinetic energy, and neither of the link masses change dependent on the system's current coordinate location [20]. However, these coordinates are overdetermined for a pendulum of fixed link lengths, and only a small subset of this four-dimensional coordinate space will be reachable by the fundamentally two-dimensional system. Additionally, concepts such as torque and rotational velocity require some manipulation to abstract into this space.

The system can alternatively be described by what I will refer to throughout this document as the system shape, a set of indirect coordinates minimally necessary to determine linkage positions in Cartesian space. For this double pendulum example, the system shape can be parametrized by two joint angles: one at the location where the system is pinned to the surface and one at the connection point between the two links. This choice of coordinate frame is convenient in that it provides a clean one-to-one relationship between any arbitrary pair of coordinate values and a corresponding robot pose, but is inconvenient because the system's effective inertia is not constant over the space. This type of frame is also useful for robotic systems in particular because they are often driven by electric motors, which are well-modeled through concepts such as torque and angular displacements.

We address the weaknesses of each of these coordinate choices by providing a mapping between them, allowing us to cross over and combine the utility of each. The differential of the mapping between the coordinates provides a Jacobian that allows us to project the physics native to the Cartesian space onto the more abstract angular coordinates [20]. This physical projection enables the notion of a

Reimmanian metric on the angular coordinate frame, providing us with the ability to estimate the "length" of trajectories in this space in a way that takes into account the system physics [22, 46].

The metric formulation is invaluable, because from a description of the metric across the coordinate frame, we can model the angular coordinate space as a two-dimensional manifold embedded in an abstracted three-dimensional space in which concepts from differential geometry such as geodesics and local curvature have real, intuitive, physical meaning [10,20]. For an example of such an application, imagine choosing an arbitrary starting pose for the pendulum system and providing the masses with an initial velocity. These initial conditions can be represented on the shape manifold as a single starting point and a velocity vector indicating a direction of travel. For the case in which there are no external or joint forces acting on the system, this starting configuration produces a natural time-parametrized evolution of system state that lies upon one of the shape manifold's geodesics, meaning that at every point of the path evolution on the shape manifold the trajectory is locally straight and there is no tangential or normal acceleration.

This system can be modified by introducing complexity to the model that effects the geometric behavior of the system evolution [28]. Friction on the actuators will result in tangential decelerations that slow shape evolution along the geodesic and might be compensated for by actuator control actions. Forces such as drag or gravity that act directly on the link elements might produce normal accelerations on the shape trajectory if uncompensated for by the actuators, resulting in a change in the trajectory evolution from the unforced case.

In this document, I will derive and utilize a variety of tools to examine robotic motion that are geometric in nature, and rely at their heart on manifold analysis. A core philosophy of this work is that many types of system motion can be conveniently examined through the framework of geometric curves and manifolds.

## 1.2 Why Passive?

For many types of robotic systems, we tend to visualize constructs that have electric motors operating on stiff, inflexible metal connections. In such a system, motion can be modeled as being completely controlled and fully driven by the output of the mechanical motors. However, in the analysis of robotic motion, it behooves the roboticist to extrospect and draw inspiration from the biological motions that exist around us in the natural world. Such biological motion, without fail, leverages flexibility within the system to improve behavior [50]. Having passive flexibility within your system provides resilience to impact damage, allows a wider range of construction materials and designs, and can reduce the energetic cost required to perform a particular motion by admitting "free" behavior from components that are not driven directly from a control system [14, 68]. Passive-elastic components, however, add considerable dynamics to the robot system, which can create challenges for gait modeling and analysis [14, 39].

Examples abound. Microscopic flagella develop a helical structure that propel cells from a simple oscillatory input by exploiting the passive flexible properties of the flagella structure itself [9]. Fish increase the efficacy of their swimming

motion by leveraging flexible fins that are only partially driven by muscle input [34]. Human muscle has a suite of convenient elastic properties naturally exert forces to pull the body back to a resting state during and after excitation, reducing the energy required to perform tasks like walk or manipulate objects [36].

The utility of passive elasticity becomes especially clear when considering a human swimming in a body of water. The efficacy of our swim can be improved by wearing plastic flippers, also known as swimfins, which bend and stretch in the water as we kick our legs [43]. The passive elasticity of modern swimfins has been carefully tuned to produce useful motion properties [3]. As the swimfin becomes more stiff, it requires more and more energy to perform the propulsive swimming kick, and as the swimfin becomes more flexible, the toes of the swimfin begin to bend more aggressively and follow less closely the undulatory motion of the leg. A particular swimfin stiffness will result in a particular pattern of motion from human swimming as the swimmer accommodates the cost and benefits of this artificial modification. Tuning the swimfin passive elasticity through mechanical design and material choice allows us to select for a type of leg motion that produces efficient travel through water [51].

In this work, passive motion behavior will be a recurring theme as we attempt to examine the benefits of passive dynamics on robotic systems, and decide how we might design robot mechanisms or control strategies that leverage elastic behaviors.

## 1.3 Why Constrained?

There are multiple types of constraints present in the systems that I will discuss in this work. There are geometric constraints that provide information about how mechanical linkages in the system are connected and that allow us to reduce our shape dimensionality to the degrees of freedom within the connections. There are modeling constraints that allow us to simplify our mathematical approach, such as assuming that a robot floating on the surface of water can be approximated as a robot travelling on a flat plane. Finally, there exist locomotive constraints, which is the category I will spend the most time discussing in this thesis.

Locomotive constraints exist within any mobile biological system and define much of the physics of system motion. One example is the constraint between a human foot and the ground. The friction in this interface produces a special interaction that induces little or no slip between the two surfaces, allowing us to leverage and push off from static contact points to walk. Humans will often select foot placements while walking that exploit such constraints and minimize slip [40]. If this constraint is lifted even partially, such as on a slick ice surface, walking becomes less efficient and we must adapt our gait.

It is not uncommon to apply other locomotive constraints beyond the basic ones that enable motion. For example, consider trying to push a heavy bookshelf. In addition to the foot-ground interaction, we accommodate an additional constraint between our hands and the furniture such that sufficient force transmission is required to move the bookshelf and the connection location is otherwise static.

It should come as no surprise that this additional constraint changes the way we position and move our body relative to our normal walking motion even though the objective to move forward remains the same. However, between these two differently constrained cases, the dynamic properties and capabilities of the body remain the same, and could be observed with appropriate sensing.

For mobile robotics, it is occasionally expected to apply new locomotive constraints to the systems during experimentation. Such constraints can provide additional sensing into the robot's behavior, such as through load-cell connections that measure force transmission between two constrained points [7,35,65], or serve a purpose of convenience such as an elastic tether that keeps a system within an experiment testbed [11] or a sliding rail that limits motion to one degree of freedom [55]. However, such additions do not generally change the internal dynamic characteristics of the system.

In this work, we will discuss when and how it is appropriate to apply additional experimental locomotion constraints to robotic systems, a process for identifying a robot's internal dynamic properties using constrained experimental data, and how this dynamic information transfers between the physical context of separate locomotive constraints.

## 1.4 Why Locomotion?

There is much versatility in biological locomotion. Organisms can alter gaits to gain precise control over body motion for maneuvering, can cross terrain that is

inaccessible to conventional wheeled systems, and can operate at a high level of energetic efficiency [16, 32]. Through the study of locomotive systems, we hope elevate modern robotics to be more robust and better able to handle the diverse environments that the world has to offer.

We also hope to gain through these studies a better understanding of the natural world. Through robotic locomotive studies, we begin collaborative conversations with biologists and can form better and more interesting hypotheses to test concerning biological motion [18].

In this work, we will address physical modeling and experimentation for locomoting systems. In particular, we will focus on systems that locomote through fluid, occasionally referring to this phenomenon as "swimming" when appropriate. Our concentration on fluid-immersed systems exists for a variety of reasons. First, such systems are mathematically more simple to model than other strategies such as legged locomotion, which has a suite of event-based and stability-related dynamics that can obfuscate some of the core principals that we seek to learn from our locomotion studies [26]. Second, swimming robots are well-suited for experimentation. If the electronics are correctly shielded, a locomotive misstep is unlikely to result in catastrophic failure of the robotic system. This is a significant advantage over legged or aerial robots, which can easily suffer hardware damage after a controls failure [17, 27]. In addition, unlike aerial robotic systems, swimming robots can fairly easily carry enough power through off-the-shelf electrical batteries to power themselves for appreciable periods of time [59]. Robots that locomote along the surface of a body of water are particularly convenient, because

they require only limited waterproofing and can be approximated as travelling along a flat plane, reducing the dimensionality of the locomotion problem.

Although there are a few confounding factors that complicate the process of studying swimming locomotion such as vortex generation and unmodeled surface interactions, we feel that these systems offer promising leads into the questions of what optimal biomimetic locomotion looks like.

In this thesis, I will introduce three types of system that locomote through fluid. The first two systems operate at opposite ends of the Reynolds number spectrum, with the third situated between [2]. First, we will discuss high-Reynolds number locomotion for elastic systems that pertains to systems where inertial effects dominate the locomotive physics and drag can be neglected. Then, we will discuss low-Reynolds number locomotion for systems where we can assume that drag effects dominate the inertial contribution. Finally, we will leverage the lessons learned from each and of this spectrum to study and optimize designs and controls for robotic systems that swim in the intermediate Reynolds number domain of aquatic motion.

# Chapter 2: Passive-Elastic Locomotion

## 2.1 Introduction

It is typical for animals and some biomimetic robots to locomote by interacting with the environment through cyclic shape changes, or *gaits*. The study of these motions provides insight into the organisms that perform these gaits and could lead to the development of biomimetic robots with improved maneuverability and increased mobility across different environmental terrains.

Animals often make use of passive-elastic body elements in their gaits, utilizing flexible tails or pendulum action in the limbs to increase locomotive capabilities. Some fish, for example, have passive properties such that vorticity in a current can excite passive dynamics in the body and cause the fish to 'swim' upstream even after the fish has died [5], highlighting the importance of passive mechanical properties in biological locomotion. Humans also exploit passive dynamics while walking: almost no muscle input is supplied to the knee during its swing phase [38].

The geometric mechanics community has developed a range of tools to study the properties of gaits for different locomotor physiologies. Many of these tools, however, assume that the locomotor's shape space is fully actuated, and therefore the tools cannot be directly applied to systems with passive body elements. Although previous works have developed geometric tools for the study of passive

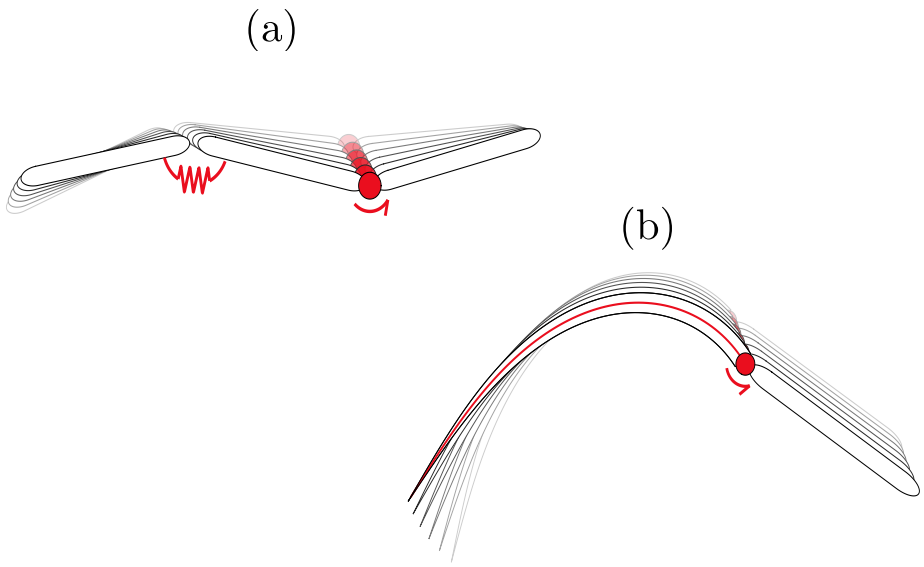


Figure 2.1: The active-passive swimmer models discussed in this chapter, shown in motion due to gaits at their passive limit-cycles. Active joints are represented by red circle motors, and passive joints by red springs. (a) Purcell's swimmer (b) 'Fish' swimmer with a linear curvature passive tail

systems operating using no-slip constraints [15] and for swimming systems operating in drag-dominated regimes [6, 49], many systems such as fish or other aquatic systems larger than a few millimeters in length are subject to a different set of dominant physical forces. Such swimmers are better modeled as inertia-dominated systems swimming in a high Reynolds number 'perfect fluid' that require a different mathematical approach [20, 29].

In this chapter, we construct a methodology for optimizing gaits in inertia-dominated systems with passive-elastic elements that translate and rotate in a

plane. We demonstrate that this methodology produces compact equations of motion that allow for simple optimization of gaits that exploit passive dynamics. In particular, we will discuss two models of simulated active-passive systems, which are assumed to be swimming in a perfect fluid and are illustrated in Fig. 2.1:

(a) The three-link swimmer, with one passive joint

(b) A fish-tail swimmer with a passively flexible tail

For both of these systems, we discuss the properties of optimal motion and the process of passive parameter selection. We show how to select a spring stiffness and damping coefficient that will produce the most favorable passive dynamics for a desired level of mean power consumption from the driving motor given a definition of body geometry. We also discuss how to compare the efficacy of different passive shape modes by observing their respective optimal motions at unit average power exertion from the input joint with properly normalized passive-dynamic coefficients. We demonstrate that the fish-tail swimmer produces more efficient locomotion than the three-link swimmer, indicating that continuous-curvature systems have more favorable locomotive properties.

We then consider the process of optimizing gaits for systems in which the passive dynamic coefficients have already been selected, potentially suboptimally, by material choice or system design. We show that optimizations on both of these swimming systems result in gaits that have similar properties. Optimizing for

speed produces a maximum-speed gait that consists primarily of a sinusoid to the active joint. In cases of suboptimal coefficient selection, the input sinusoid can be augmented with a small amount of high-order motion that provides beneficial characteristics in shape space.

Unlike a fully active swimmer, which can move arbitrarily fast when provided with sufficient actuator power, swimmers with fixed passive elements have a single highest-speed gait because of the passive-dynamic interaction. Optimizing for mechanical efficiency produces zero-motion gaits because actuator cost increases faster than displacement as the gait size increases. We demonstrate that considering an additional metabolic cost alongside actuator effort costs, however, produces a class of useful gaits that yield efficient locomotion for systems with varying overhead energy consumption. Low swimmer metabolisms produce minimal motion, as the swimmer is not incentivized to move quickly. High swimmer metabolisms drive the system to gaits similar to the maximum-speed gait. Middling metabolisms produce gaits that compromise between speed and mechanical costs. High-level optimization results for the three-link swimmer are illustrated in Fig. 2.2.

## 2.2 Mathematical Formulation

In this section, we review the assumptions and techniques that facilitate this work and provide the formulation for the optimization used in our following dynamic system examples. We then briefly discuss techniques for qualitatively understanding the properties of optimal gaits.

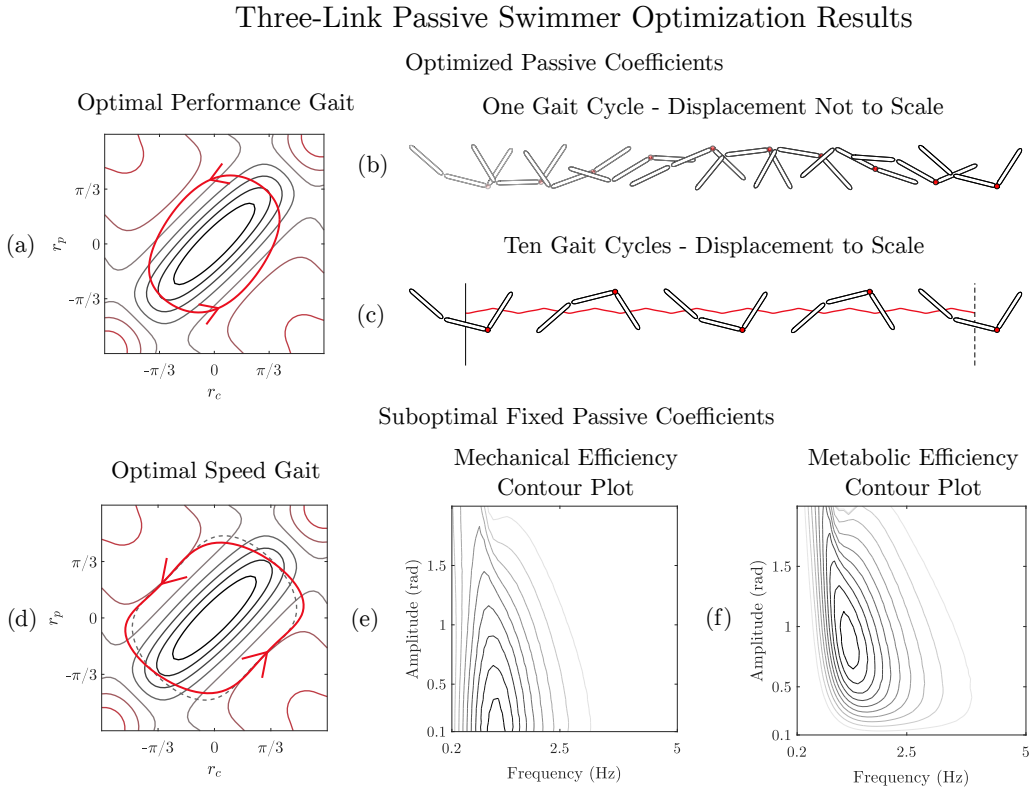


Figure 2.2: High-level optimization results for the three-link swimmer: (a) When the passive joint properties can be tuned to suit desired exertion level, the optimal gait is a simple sinusoid input. The gait formed from this input and the passive response is shown superimposed on the forward-motion CCF. (b) The three-link swimmer in the process of its optimal gait, not to scale with swimmer body size. (c) The result of executing ten gait cycles, to scale with swimmer body size. (d) When the passive joint properties are fixed suboptimally and cannot be optimized alongside the gait, the optimal gait includes high order motion on top of the input sinusoid. Gait asymmetry is highlighted with a grey dashed line. (e) A contour plot of mechanical efficiency over various sinusoid inputs to the active joint. Optimizing directly for mechanical power results in zero-amplitude gaits that produce no displacement. (f) A contour plot of metabolic efficiency for a metabolic rate of $\gamma_m = 0.05$. Including metabolic costs alongside mechanical cost of transport in the objective function produces non-trivial efficient optimal gaits.

### 2.2.1 Obtaining the Equations of Motion

The models we employ in this work are built upon the assumption that the momentum of the swimmer-fluid system remains constant throughout the gait cycle [23] - i.e. we only apply forces to the system through active joint forces, and we do not consider the effects of vortex shedding or fluid drag. This assumption allows us to relate body motion to a system's shape variables $r$ and their time rate of change $\dot{r}$ through the momentum-free *reconstruction equation*,

$$\mathring{g} = \mathbf{A}(r)\dot{r}, \tag{2.1}$$

in which $\mathring{g}$ represents swimmer velocity expressed in its local body frame and $\mathbf{A}$ is the *motility map*, which linearly maps shape velocities to the resulting body velocities produced through fluid interaction.[1]

For inertial systems, the motility map can be found by first writing the system's kinetic energy in terms of its body and shape velocity and its generalized mass matrix $M$,

$$\text{KE} = \frac{1}{2}\begin{bmatrix}\mathring{g}^T & \dot{r}^T\end{bmatrix} M \begin{bmatrix}\mathring{g} \\ \dot{r}\end{bmatrix}. \tag{2.2}$$

The total mass matrix is formulated from individual link mass matrices $\mu_i$. The individual mass matrices are the sum of mass contributions from both link masses

[1] In previous works, we have used the equation $\mathring{g} = -\mathbf{A}(r)\dot{r}$ and referred to $\mathbf{A}$ as the local connection. Here, to reduce sign confusion, we have included the negative sign in $\mathbf{A}$ and chosen to refer to it as the motility map instead as in [4].

and hydrodynamic added mass terms that take into account the fluid mass that must accelerate with swimmer motion,

$$\mu_i = (\mu_i)_{body} + (\mu_i)_{fluid}. \tag{2.3}$$

The hydrodynamic added mass plays a key role in the dynamics of swimming systems, as it introduces the anisotropy of reaction forces that allows the systems to generate net position changes through cyclic shape changes.

The individual mass matrices are pulled back into generalized coordinates via the Jacobians $J_i$ relating swimmer velocity and shape velocity to link velocity in the link's frame as in [20],

$$M(r) = \begin{bmatrix} M_{gg} & M_{gr} \\ M_{rg} & M_{rr} \end{bmatrix} = \sum_i \left( J_i^T \mu_i J_i \right). \tag{2.4}$$

This mass matrix can be used to intuitively map generalized velocity to generalized kinetic energy as in Eq. (2.2), and can also be used to map generalized velocity to generalized momentum,

$$p = M \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} M_{gg} & M_{gr} \\ M_{rg} & M_{rr} \end{bmatrix} \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}. \tag{2.5}$$

Under the assumption that no mass components are shed from the swimmer, the total momentum of the swimmer-fluid system is conserved. If the swimmer starts at rest, the system maintains zero position-space momentum for all time.

Using this Pfaffian constraint,

$$0 = M_{gg}\mathring{g} + M_{gr}\dot{r}, \tag{2.6}$$

we can solve for the motility map $\mathbf{A}$ that maps joint velocities to swimmer body velocities that are consistent with both the specified shape motion and the zero momentum constraint in Eq. (2.6),

$$\mathring{g} = -M_{gg}^{-1}M_{gr}\dot{r} = \mathbf{A}(r)\dot{r}. \tag{2.7}$$

For a closed cyclic gait path $\phi$ through shape space, total resultant body motion $g_\phi$ from one gait cycle can be found as the line integral of the motility map along the gait path mapped from the body frame into the system's local coordinates through the body configuration $g$,

$$g_\phi = \oint_\phi g\mathbf{A}(r). \tag{2.8}$$

To estimate the actuator forces required to enforce a desired gait shape, it is useful to represent the swimmer's effective mass in the shape space. This reduced mass matrix $M_r$ can be calculated from the mass matrix and the motility map [20],

$$M_r(r) = \begin{bmatrix}\mathbf{A}^T(r) & \mathrm{Id}\end{bmatrix} M(r) \begin{bmatrix}\mathbf{A}(r) \\ \mathrm{Id}\end{bmatrix}. \tag{2.9}$$

This formulation allows us to decouple the internal shape-change dynamics from the external body-motion dynamics and write the full system Lagrangian $L$

only in terms of the shape variables without direct consideration of body velocity, because the position-space motion induced by shape change motion is handled implicitly through the motility map. We also take into account passive joint behavior through a stiffness matrix $K$ that encodes the potential energy from passive joint stiffness and a Rayleigh dissipation function $G$ that encodes the damping frictional force at the passive joint through the dissipation matrix $B$. In this work, stiffness and damping act on the passive shape mode only, and so $K$ and $B$ are zero in the position space terms and can by position-space symmetry be expressed conveniently in the same reduced-dimension space as $M_r$:

$$L = \mathrm{KE} - \mathrm{PE} = \frac{1}{2}\dot{r}^T M_r(r)\dot{r} - \frac{1}{2} r K r, \tag{2.10}$$

$$G = \frac{1}{2}\dot{r}^T B \dot{r}. \tag{2.11}$$

When the system shape is composed of a controlled mode $r_c$ and a passive mode $r_p$ such that $r^T = [r_c, r_p]$, the stiffness matrix can be written using only the passive mode spring constant $k$,

$$K = \begin{bmatrix} 0 & 0 \\ 0 & k \end{bmatrix}. \tag{2.12}$$

Similarly, for an inertial fluid where the only source of dissipation is damping on the passive joint, the dissipation matrix can be written using only the damping constant $b$,

$$B = \begin{bmatrix} 0 & 0 \\ 0 & b \end{bmatrix}. \tag{2.13}$$

Passing the Lagrangian and the Rayleigh dissipation function into the Euler-Lagrange equations produces equations of motion for the system shape variables that account for the dynamics of induced body locomotion and passive joint behavior,

$$\tau = M_r(r)\ddot{r} + C(r, \dot{r}) + Kr + B\dot{r}. \tag{2.14}$$

As discussed in our previous work on inertial arts [20], the Coriolis forces can be calculated as

$$C(r, \dot{r}) = \left( \sum_{i=1}^{d} \frac{\partial M_r(r)}{\partial r_i} \dot{r}_i \right) \dot{r} - \frac{1}{2} \begin{bmatrix} \dot{r}^T \frac{\partial M_r(r)}{-\partial r_1} \dot{r} \\ \vdots \\ \dot{r}^T \frac{\partial M_r(r)}{\partial r_d} \dot{r} \end{bmatrix}. \tag{2.15}$$

Solutions to Eq. (2.14) for a periodic control joint input signal $r_c(t)$ can be lifted to position-space solutions via the reconstruction equation in Eq. (2.7) to provide gait limit cycle locomotive properties.

### 2.2.2 Limit-Cycle Estimation

Our previous work [49] optimizing passive swimming in low Reynolds number systems used Laplace transforms and frequency-space analysis to estimate the passive

joint limit cycles. However, this technique is not feasible for inertial systems.[2] In this work we estimate the limit cycle by evaluating the ODE from Eq. (2.14) in the time domain, although others have previously used techniques such as frequency-domain nonlinear harmonic balance methods [15]. Using any of these limit-cycle estimation techniques, we can find signal parameters to the active joint motor that best exploit passive dynamics to suit some objective function. Here, we discuss our time-domain methods.

By separating the shape space into the actively controlled shape mode $r_c$ and the uncontrolled passive shape mode $r_p$, the equations of motion in (2.14) can be rewritten as

$$\begin{bmatrix} \tau_c \\ -kr_p - b\dot{r}_p \end{bmatrix} = \begin{bmatrix} M_{cc}(r) & M_{cp}(r) \\ M_{cp}(r) & M_{pp}(r) \end{bmatrix} \begin{bmatrix} \ddot{r}_c \\ \ddot{r}_p \end{bmatrix} + \begin{bmatrix} C_c(\dot{r}, r) \\ C_p(\dot{r}, r) \end{bmatrix}. \tag{2.16}$$

In order to simulate these dynamics, we write the active joint control signal as an $n^{th}$-order Fourier function of time parameterized by the Fourier variables $a_{0\ldots n}$, $b_{1\ldots n}$, and $\omega$,

$$r_c(t) = a_{c,0} + \sum_{i=1}^{n} \left( a_{c,i} \cos(i\omega t) + b_{c,i} \sin(i\omega t) \right). \tag{2.17}$$

By taking time derivatives of this equation, the joint velocities $\dot{r}_c$ and joint

[2] In our previous work, we approximated that the drag matrix is constant, which allowed us to perform Laplace transforms and directly estimate the passive transfer function. For inertial systems, the mass matrix is heavily dependent on the swimmer shape, so the assumptions required for Laplace domain analysis are no longer valid. Additionally, assuming a constant mass matrix eliminates our ability to factor in centrifugal and Coriolis forces, as these are calculated from mass matrix derivatives.

accelerations $\ddot{r}_c$ can be readily calculated. These values are used to numerically solve for the configurations of the passive modes over time by solving Eq. (2.16) for $\ddot{r}_p$ and evaluating the resulting ODE,

$$\ddot{r}_p = M_{pp}^{-1} \left( -kr_p - b\dot{r}_p - M_{cp}\ddot{r}_c - C_p \right). \tag{2.18}$$

We estimate limit cycle shape motion by finding the behavior that results from multiple executions of the candidate control signal. We then estimate gait displacement by using the local connection relationship in Eq. (2.7) to calculate the net motion that results from limit cycle execution of the gait. In Section 2.3, we will use the properties of this limit cycle such as net displacement, gait cost, and gait period to evaluate the fitness of a candidate control signal and perform optimizations over control signal parameters and passive shape properties.

### 2.2.3 Mechanical Cost of Transport

Once the gait limit cycle is known, we can also find the energy consumption required to enact this gait. In this work, we use positive mechanical power as the primary source of energy consumption, representing the energy required by the control motor to enforce the desired shape motions over the course of the gait period $T$. The control joint requires no energy input when backdriven during periods of negative mechanical power [1]. The mechanical cost of transport $E_\tau$ is the total positive mechanical power usage over the gait,

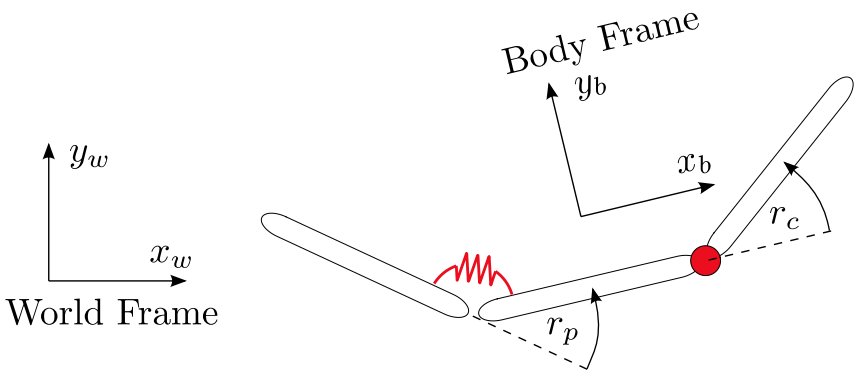


Figure 2.3: Shape parameterization of the three-link swimmer.

$$E_\tau = \int_0^T \max(\tau_c \dot{r}_c, 0) dt. \tag{2.19}$$

Here, the $c$ subscript refers to the fact that we only consider power consumed by the control joint, as passive joint mechanical power comes for free from the passive components.

We will use the notion of control motor power consumption in later sections of the work to discuss gait efficiency in terms of displacement with respect to the mechanical cost of transport.

### 2.2.4 Gait Intuition through the Constraint Curvature Function

The motility map $\mathbf{A}$ is a covector field over the shape space mapping shape space motions to body motions. Net gait displacement can be found by integrating the motility map over the path of the gait $\phi$ while mapping instantaneous motion from

the body frame into the world frame through the swimmer state $g$ as in Eq. (2.8). If the gait shape motion is a closed loop, this line integral can be approximated by an area integral over the enclosed space $\phi_a$ in a manner similar to Stokes' theorem [19],

$$g_\phi = \oint_\phi g\mathbf{A}(r) \approx \iint_{\phi_a} D\mathbf{A}, \tag{2.20}$$

where $D\mathbf{A}$ is the curvature of the constraints encoded by $\mathbf{A}$. This constraint curvature function (CCF) is generally useful for examining regions of the shape space that contribute to locomotion in the desired direction [19]. Gaits are more effective at achieving displacement if they enclose sign-definite regions of the CCF. The forward-motion CCF for the three-link swimmer is shown in Fig. 2.2(a).

This approximation is most accurate in body coordinates at a generalized center of mass. These coordinates minimize the noncommutative interactions from the intermediate motions along the gait [22].

In this work, we use the motility map to directly measure the net motion from gait limit cycles as in Eq. (2.8) and the CCF to provide high-level intuition into why optimal gaits tend to develop particular shapes. Speedy and efficient gaits will generally enclose primarily the positive black region in the center while minimizing the amount they enclose the surrounding negative red regions.

### 2.2.5 Swimmer Modeling

To demonstrate the principles described above, we apply them to two swimming locomotors.

The first is the three-link swimmer, a model that is commonly used as a minimal reference example for locomoting systems [44]. As illustrated in Fig. 2.3, the three-link swimmer is composed of a chain of three elongated links, and the swimmer shape is parameterized by the angular deflection of the two joints in the chain. By performing the geometric process detailed above, the low Reynolds number Purcell swimmer can be modified into a perfect-fluid swimmer in an idealized high Reynolds number environment. To model hydrodynamic added mass, we approximate each link as an ellipse and use added mass coefficients given for ellipses in previous works [41]. To simplify the process, we assume that each swimmer element has a constant hydrodynamic mass, although a full shape-dependent, hydrodynamically coupled model could be found using the panel method [29] at some additional computational expense.

The second system is the fish-tail swimmer, which has a passively flexible tail. Continuously flexible systems are common in aquatic locomotion, as biological swimmers tend to locomote using motions that continuously deform the body rather than with joint-like behavior [33,52]. Aside from the flexible tail, this model is conceptually similar to the three-link swimmer, with oscillations of a rigid 'head' driving locomotion. The fish-tail swimmer is illustrated in Fig. 2.4

In this work, we simplify the model of a continuously flexible tail by assuming

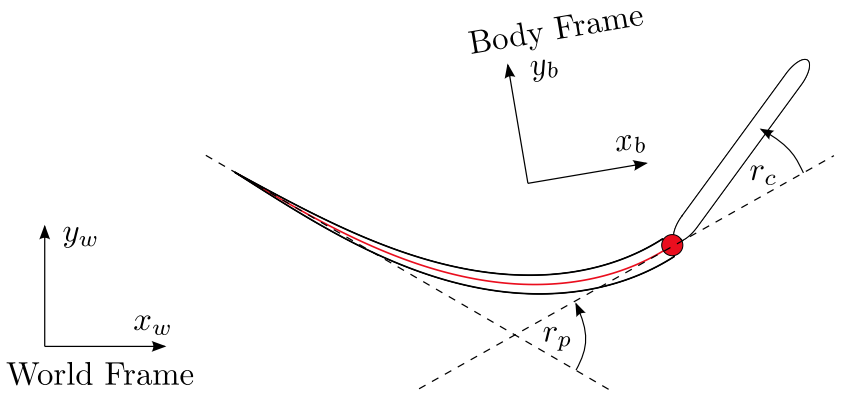


Figure 2.4: Shape parameterization of the fish-tail swimmer. The passive shape mode is tuned so that the shape magnitude is equal to the angular deflection at the tip of the tail.

that tail stiffness is high enough that the forcing frequency is below the fundamental eigenfrequency, so all motion corresponds to the first mode of an inertial cantilever beam. Curvature of the tail is at a maximum where it connects to the swimmer head, and decays to zero at the tip of the tail.

To estimate the curvature mode of the passive-flexible tail, we take the tail as an Euler-Bernoulli beam of tapering width. Using beam theory, we find the relationship between the curvature of the tail and the force profile applied along the length. Using this curvature-force relationship, we iterate until we find a curvature mode consistent with the inertial force profile that results from that modal acceleration.

We initially model the force profile on the beam as the force on a straight beam undergoing angular acceleration through fluid at the base of the tail. We model the inertial force $f(s)$ on the rigidly rotating beam as linearly decreasing along the

tail arclength $s$ from a maximum at the beam tip ($s = 0$) to zero at the base of the tail ($s = 1$) due to the acceleration of the beam components through the fluid,

$$f(s) = f_0(1 - s). \tag{2.21}$$

The particular force magnitude $f_0$ is not important at this point because we will later rescale the shape parameterization such that the mode has unit integrated curvature. We then analytically solve for the internal moment $m(s)$ at each point along the beam that gives balance to the applied inertial forces,

$$m(s) = \frac{f_0}{2}\left(s^2 - \frac{s^3}{3}\right). \tag{2.22}$$

To approximate the continuous stiffness of a biological fish, we use a tapered body profile, such that beam moment of inertia $I(s)$ is at a maximum where it connects to the 'head' and decays to zero at the tail tip,

$$I(s) = I_0 s. \tag{2.23}$$

The moment of inertia profile combined with the elastic modulus $E$ gives a local bending stiffness $k(s)$ for each point along the beam arclength that relates the internal moment at that point to the resultant local curvature $\kappa_e$ based on the beam mechanical properties,

$$k(s) = EI(s) = \frac{m(s)}{\kappa_e(s)}. \tag{2.24}$$

Using the local stiffness profile given by the tapering beam shape and the internal moment profile found from the force distribution, we can calculate the resulting curvature profile over the beam,

$$\kappa_e(s) = \frac{m(s)}{k(s)} = \frac{f_0}{2EI_0}\left(s - \frac{s^2}{3}\right). \tag{2.25}$$

To ensure convergence of the iterative algorithm, we enforce through a scaling factor $C$ that each iteration of the curvature mode has unit amplitude,

$$C\int_0^1 \kappa_e(s)ds = 1. \tag{2.26}$$

This gives a normalized shape mode curvature profile in terms of the passive shape magnitude $r_p$,

$$\kappa_p(s) = C\kappa_e(s)r_p. \tag{2.27}$$

A convenient side-effect of normalizing the curvature mode is that it leads to shape magnitudes $r_p$ that are directly comparable to those of the three-link swimmer. With this normalization, the tip orientation associated with a particular shape magnitude aligns with the orientation of the three-link swimmer tail at the same magnitude. This correspondence of tip angle to a discrete joint on the swimmer is illustrated in Fig. 2.4.

Once we have this shape mode, we replace the force profile used in Eq. (2.21) with the inertial force profile that would result from acceleration of this curvature mode,

$$f_{n+1}(s) = f_0 \int_0^s (s-\ell)\kappa_p(\ell)d\ell, \tag{2.28}$$

where $\ell$ is an intermediate variable that allows for integration along the arclength up to the specified location $s$. We then iterate these calculations, using forces to estimate curvatures and curvatures to estimate forces, until the shape mode converges to a curvature profile consistent with inertial forces that would result from that profile. We use the converged curvature profile as our mode-shape for low-dimensional analysis. This iteration process is illustrated in Fig. 2.5.

This passive shape mode approximates the deflections experienced by a tapered beam in inertial fluid where the forcing frequency serves to excite primarily the first eigenmode, which has been previously shown to be an effective way of producing reactive thrust for slender metal beams submerged in water [62]. Subsequent modes can be added to generate traveling waves and more complex passive behavior.

Once we have a curvature profile for the shape mode, we can calculate the Jacobian $J(r,s)$ continuously along the beam arclength using techniques presented in our previous works [47]. The mass metric contribution from the tail is calculated akin to Eq. (2.4) by integrating the hydrodynamic mass pullback along the beam using the beam's hydrodynamic mass density $\rho_m$,

$$M(r) = J_h(r)^T \mu_h J_h(r) + \int_0^1 J(r,s)^T \rho_m J(r,s) ds, \tag{2.29}$$

where the $h$ terms represent the mass metric contribution from the rigid head element.

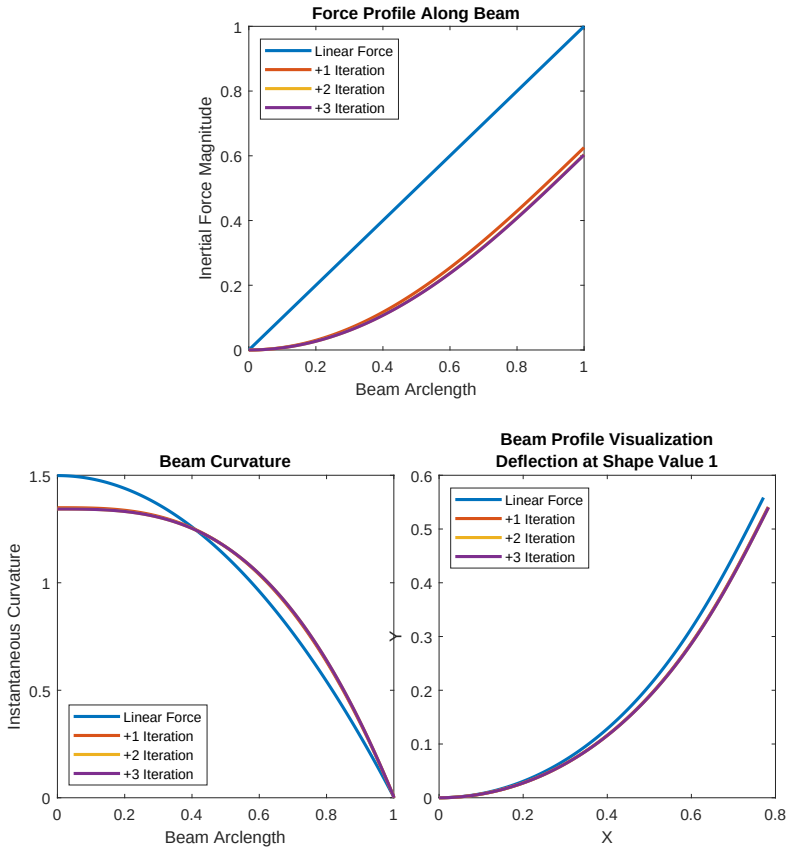


Figure 2.5: Results of iterating back and forth between beam force profile and resultant beam curvature. From the linear force mode, the beam quickly converges to a mode that has a self-consistent curvature and force loading. This iterated mode qualitatively similar to the linear force mode.

Beam stiffness can be expressed in the shape mode as the second derivative of tail potential energy with respect to the passive shape mode deformation. Integrating along the tail to find the potential energy, we find

$$PE = \int_0^1 \frac{1}{2} k(s) \kappa_p^2(s) ds. \tag{2.30}$$

From here we calculate the modal stiffness,

$$k_p = \frac{\partial^2 PE}{\partial r_p^2} = C^2 \int_0^1 k(s) \kappa_e^2(s) ds. \tag{2.31}$$

The modal stiffness $k_p$ is a spring constant that relates shape mode magnitude to the net beam passive response due to the beam mechanical properties, and can be used to build the stiffness matrix in Eq. (2.12).

Previous works have investigated limit cycle estimation for locomotors with multiple passive modes [15], allowing for the optimization techniques presented in this work to extend to systems with additional passive shape modes. If given physical motion data for points along a flexible body moving in fluid, appropriate shape modes can be extracted using the technique of eigenvector analysis [58]. This allows for a model to be built with behaviors that more closely resemble physical motion of a desired system.

| **Described Quantity** | **Symbol** | **Unit** |
|---|---|---|
| Swimmer Length | $\ell$ | $\ell$ |
| Time | $t$ | $s$ |
| Mass | $m$ | $m$ |
| Net Gait Displacement | $D$ | $\ell$ |
| Gait Period | $T$ | $s$ |
| Gait Frequency | $Hz$ | $1/T \rightarrow 1/s$ |
| Instantaneous Body Velocity | $\mathring{g}$ | $\ell/s$ |
| Gait Speed | $\eta_v$ | $D/T \rightarrow \ell/s$ |
| Shape Deflection | $r$ | rad |
| Shape Velocity | $\dot{r}$ | rad$/s$ |
| Shape Acceleration | $\ddot{r}$ | rad$/s^2$ |
| Torque | $\tau$ | $m\ell^2/s^2$ |
| Spring Stiffness Coefficient | $k$ | $\tau/r \rightarrow m\ell^2/\text{rad}s^2$ |
| Damping Coefficient | $b$ | $\tau/\dot{r} \rightarrow m\ell^2/\text{rad}s$ |
| Energy | $E$ | $m\ell^2/s^2$ |
| Power | $P$ | $E/s \rightarrow m\ell^2/s^3$ |
| Metabolic Rate | $\gamma_m$ | $E/s \rightarrow m\ell^2/s^3$ |
| Mechanical Efficiency | $\eta_\tau$ | $D/E \rightarrow s^2/m\ell$ |
| Metabolic Efficiency | $\eta_m$ | $s^2/m\ell$ |

Table 2.1: Generalized units for quantities used in this work.

### 2.2.6 Units

The numerical results we present later in this chapter use the set of units provided in Table 2.1.

## 2.3 Optimizing Performance

In general our high-level optimization process is as follows. First, we choose a swimmer model and use this model to develop passive-dynamic equations of motion. Then, we can estimate the whole-body limit-cycle that results from a particular

choice of control joint input signal $r_c(t)$ and passive joint parameters $k$ and $b$. Net gait motion is found by integrating the motility map over the gait cycle, gait period is found from the gait frequency parameter, and gait energy cost due to actuator usage is found by integrating actuator effort over the gait cycle. These metrics are combined into a fitness function that quantifies the performance of a choice of control joint input signal and passive parameters. The input signal and passive joint parameters can then be optimized for the chosen objective function using any sample-based optimization algorithm.

Depending on whether we are optimizing the gaits of a fixed physical plant or simultaneously optimizing gaits alongside physical design, there are multiple possible routes for optimizing passive-inertial systems.

If the optimization is performed before the system is fully instantiated, the spring and damping constants can be chosen alongside the gait parameters to optimize system behavior and the passive dynamics with respect to a desired average level of energy exertion. This method can be used to compare the performance of potential locomotor geometries or shape modes by comparing ideal performance of the different systems under identical power consumption.

It is also possible to optimize gaits for systems that have immutable passive coefficients that cannot be optimized alongside the gait parameters. This route is more applicable to systems where the passive coefficients are fixed in place by material choice or other design considerations.

In this section, we will describe how we construct the objective functions that we use to optimize gaits for the separate cases of mutable and immutable passive

joint behavior.

### 2.3.1 Simultaneously Optimizing Gaits and Passive Coefficients

The systems we deal with in this work are heavily nonlinear, with the system mass matrix and subsequent limit-cycle dynamic behavior being strongly coupled to shape space. However, once a limit-cycle transfer function for a given input motion is established using nonlinear methods, we can use a linear approach to find a curve through the space of joint properties and actuator frequency that produces constant transfer functions. Along these particular identical-transfer-function curves in parameter space, nonlinear effects scale cleanly because there are no variations in the gait's path and relative pacing. For these cases, scaling from linear theory is exactly correct even on the nonlinear system. These curves can be monotonically parameterized by input power to the active joint, so we can select a point along it to match the available power. The optimization problem then becomes finding the highest-speed limit cycle that can be executed at unit power. After finding this pairing of optimal unit-power input motion and passive parameters that enable this motion, the behavior can be scaled to any desired average power level using linear theory, giving simultaneously the fastest and most efficient locomotion possible at that power level.

Our identical-transfer-function curves are formed in the space of three parameters: the spring constant $k$, the damping constant $b$, and the gait frequency $\omega$. For the limit cycle transfer function to stay the same, two ratios must be preserved:

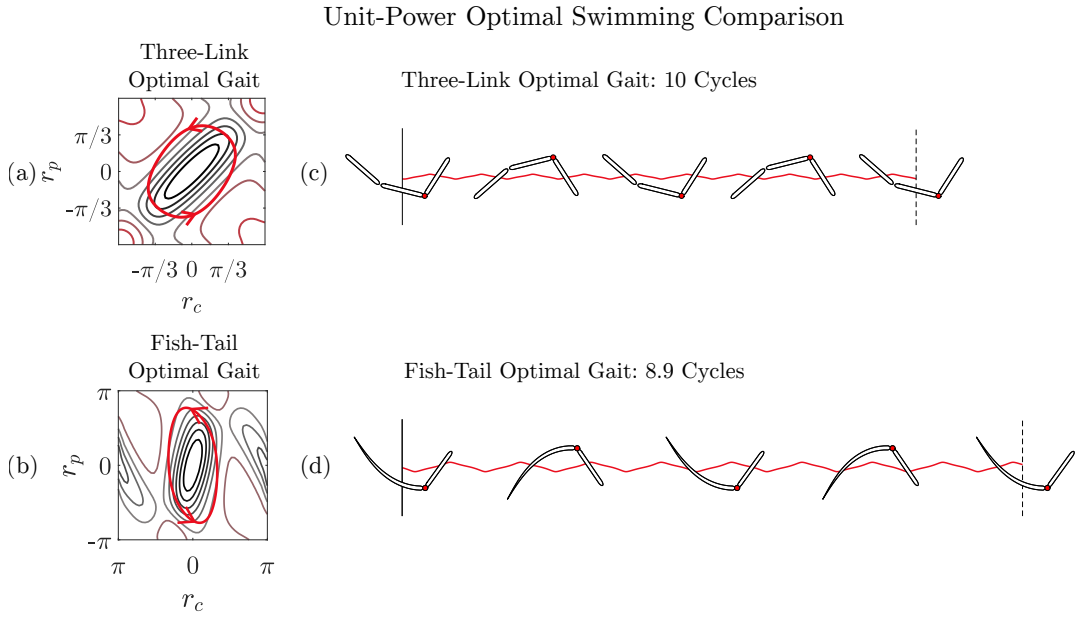


Figure 2.6: Comparison of the three-link swimmer and the fish-tail swimmer with optimized passive coefficients performing their optimal gaits at unit power consumption. Although the three-link swimmer can perform its gait 10% faster than the fish-tail swimmer at unit power, the displacement per gait cycle is substantially lower, so the fish-tail swimmer has more efficient locomotion overall. (a) The three-link optimal gait superimposed on the forward-motion CCF. Optimal motion for tuned passive parameters is a simple sinusoid input. (b) The fish-tail optimal gait superimposed on the forward-motion CCF. Optimal motion for tuned passive parameters is a simple sinusoid input. (c) Result of the three-link swimmer performing 10 gait cycles at unit power consumption, with displacement to scale with swimmer body. Body center-of-mass tracked with a red background line. (d) In the time it takes the three-link swimmer to perform 10 gait cycles at unit power, the unit-power fish-tail gait can be performed only 8.9 times. Result of the fish-tail swimmer performing 8.9 gait cycles, with a red background line tracking body center-of-mass motion.

The first is the ratio of the input frequency to the system's natural frequency,

$$R_\omega = \frac{\omega}{\omega_n} = \frac{\omega}{\sqrt{k/m}}, \tag{2.32}$$

and the second is the damping ratio,

$$\zeta = \frac{b}{2\sqrt{km}}. \tag{2.33}$$

In the space of our three parameters $k$, $b$, and $\omega$, the frequency ratio and damping ratio act as two constraints that enforce a constant limit-cycle transfer function. This leaves us with a one-dimensional curve through the parameter space on which the transfer functions will be identical for the same input motions. Along the identical-transfer-function curve, the two ratios are constant, so along the curve the spring constant is proportional to $\omega^2$ and the damping constant is proportional to $\omega$ (neglecting variations in mass because they will be identical along the curve). This means that we can parametrize the constant-transfer-function curve as a function of gait frequency,

$$TF(\omega) = (k(\omega), b(\omega), \omega) = (k_0\omega^2, b_0\omega, \omega), \tag{2.34}$$

in which $k_0$ and $b_0$ are the spring and damping constants corresponding to unit gait frequency.

For the system effort, we consider primarily the positive mechanical power,

$$P_c = \max(\tau_c \dot{r}_c, 0), \tag{2.35}$$

representing the rate of energy supply required to enforce actuator motions. An actuator without regenerative braking will consume energy during periods of positive mechanical power and will require no energy when backdriven during periods of negative mechanical power [1].

Examining time-derivatives of the Fourier parametrization expressed in Eq. (2.17), we observe that input speed across the gait $\dot{r}_c$ is proportional to $\omega$ and input acceleration $\ddot{r}_c$ is proportional to $\omega^2$. We also note from the dynamic equation of motion expressed in Eq. (2.14) that control torque $\tau_c$ is proportional to input acceleration. Combining these relationships in Eq. (2.35), we see that the average positive mechanical power supplied to the control motor is proportional to $\omega^3$ along the constant-transfer-function curve. This relationship is formed as

$$P_{avg} = P_0 \omega^3, \tag{2.36}$$

where $P_0$ is the average power required to execute the gait when performed at unit gait frequency. From this power-scaling relationship, we can solve for the gait period that would result in a unit-power execution of the given limit cycle:

$$T_1 = P_0^{1/3}. \tag{2.37}$$

The corresponding unit-power gait frequency comes from the simple inverse relationship between period and frequency,

$$\omega_1 = \frac{1}{T_1}. \tag{2.38}$$

This relationship lets us avoid the computationally difficult nonlinear constraint of directly restricting gaits to unit power. Instead, we fix the gait frequency to unit value, observe the power required to enact the limit cycle for given passive parameters and gait definition coefficients, and then find the gait frequency and passive parameters that give the same limit cycle in shape space at unit power consumption.

In summary, we can optimize the physical parameters that determine the passive dynamics alongside the control parameters that define the gait. To do this, we fix gait frequency to unit value, leaving the remaining optimization variables as the Fourier parameters, stiffness coefficient, and damping coefficient. To perform our optimization, we simulate the passive-dynamic response to the described input motion and observe the limit-cycle energy cost $E_\tau$ and the net forward displacement $D$ that result. From the unit-period energy cost, we find the gait period and passive coefficients that result in unit-power execution of the same limit cycle with the same gait parameters. Our objective function is then the locomotor speed when the gait is executed at unit power,

$$\eta = \frac{D}{T_1}. \tag{2.39}$$

The gait that maximizes speed given unit power input can be rescaled using Eqs. (2.34) and (2.36) to find the frequency and passive parameters that produce the fastest and most efficient locomotion at any desired power budget.

Results for optimizing gaits and passive parameters for the three-link swimmer

and the fish-tail swimmer are shown in Fig. 2.6. Both optimal gaits consist of first-order sinusoids. We attribute the lack of high-order motion to the fact that high-order joint activity comes with cubicly increasing power costs.

Comparing the behavior of the two systems at unit average power consumption of the driving motor, we can see that the fish-tail swimmer is approximately 20% more efficient than the three-link swimmer. Although the three-link swimmer can perform its optimal gait 10% faster than the gait of the three-link swimmer, its displacement per cycle is only 75% of that of the three-link swimmer. It is likely that other choices of body curvature could improve the performance of the fish-tail swimmer even further. Unlike the fish-tail model, migratory biological swimmers like trout and salmon produce efficient long-distance locomotion through body curvature that is heavily concentrated towards the rear of the swimmer [53]. The tapering-stiffness beam, however, develops a high curvature concentration near the center of the swimmer body, leading to potentially suboptimal behavior.

### 2.3.2 Objective Functions for Fixed Suboptimal Passive Coefficients

For a robot, it is not always the case that we have full control over the passive parameters of body stiffness and damping. These factors are likely to be influenced by material choice in the body of the locomotor or other considerations in mechanical design. For these cases, we seek to find the input joint behavior that produces best locomotion given an existing physical system.

With fixed passive behavior, we can no longer rescale the stiffness and damping

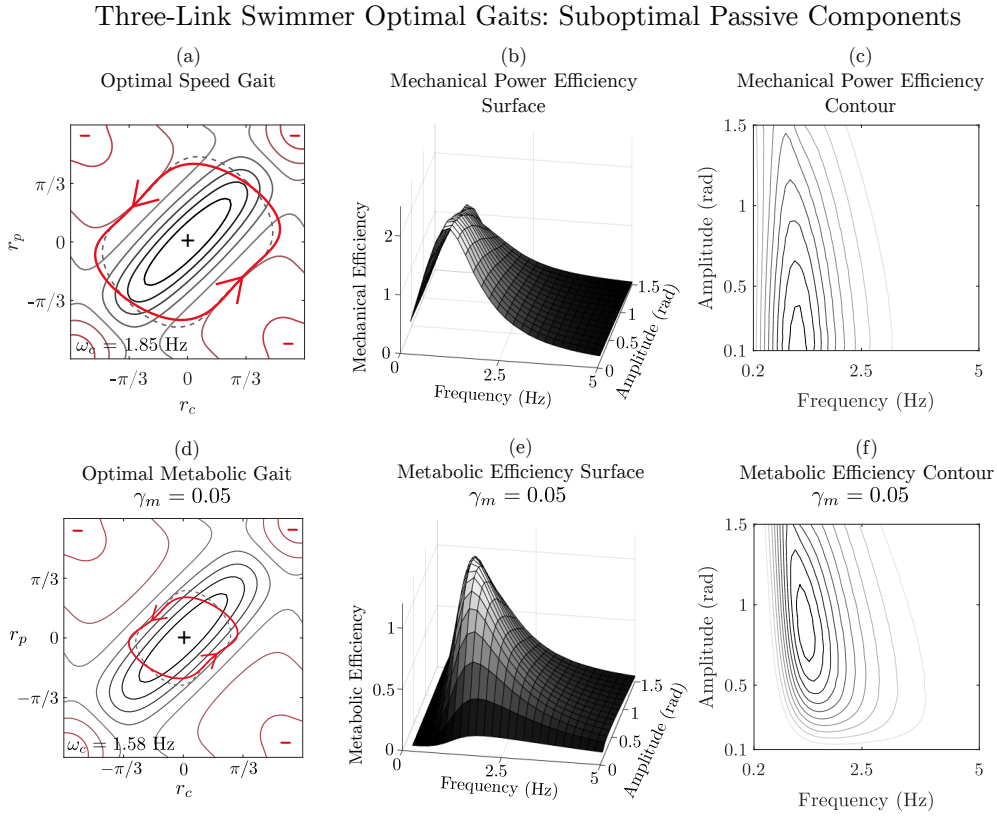


Figure 2.7: Results for the suboptimally-tuned passive three-link swimmer across our three objective functions: (a) The maximum-speed gait superimposed on the forward-motion CCF. With fixed passive coefficients, optimizing for speed results in a single best gait with frequency dependent on the passive parameters. The gait uses high-order motion at the active joint to compensate for poor passive-properties, resulting in asymmetric motion. Asymmetry highlighted using a symmetric grey dashed line. (b-c) Mechanical efficiency of the passive three-link swimmer at various input sinusoids for the mechanical power objective function. Considering only mechanical power costs without fixing a power budget results in an optimal gait with near zero input motion. (d) Results for optimizing a gait while taking into account a metabolic energy drain rate of $\gamma_m = 0.05$. Considering metabolic costs causes the optimizer to compromise between speed and energy expenditure. (e-f) Locomotive efficiency of the passive three-link swimmer when considering metabolic drain. Taking into account energy overhead for the swimmer results in a non-trivial optimal gait even without constraining the optimization to a power budget.

coefficients to shift an arbitrary passive-dynamic limit cycle to any desired input frequency. Under this condition it no longer makes sense to restrict gaits to unit power consumption, because behavior will not scale cleanly over different power budgets. For fixed-coefficient swimming systems, we must therefore define a new set of objective functions.

The first of these objective functions is the locomotor speed,

$$\eta_v = \frac{D}{T}, \tag{2.40}$$

where $D$ represents the net forward displacement of the swimmer per gait cycle and $T$ is the gait period. This quantity gives a measure of how far the swimmer will travel using a certain gait in unit time.

For our second optimization function, we choose mechanical efficiency,

$$\eta_\tau = \frac{D}{E_\tau}. \tag{2.41}$$

By dividing gait displacement by the energetic cost, we find how far the swimmer can travel per unit energy sent to the control motor.

The final objective function takes into account both actuator power consumption and other miscellaneous energy overhead via a metabolic rate $\gamma_m$ that represents the rate of time-dependent energy expenditures, such as processor power consumption:

$$\eta_m = \frac{D}{E_\tau + \gamma_m T}. \tag{2.42}$$

### 2.3.3 Results of Optimization

In the following text, we discuss the results of optimizing passive gaits on our two swimmers when passive coefficients are frozen at arbitrary suboptimal values. In particular, we used a stiffness coefficient of 0.075 and a damping coefficient of 0.01 for both systems. We first discuss the effects of optimizing each of our three objective functions on both systems. Unlike fully actuated swimmers, which can locomote arbitrarily fast, passive swimmers have a bounded maximum speed because of the passive-dynamic interaction. Moving at too high a frequency produces straight-line gaits in shape space that produce no net locomotion [49]. The maximum speed gait for both systems consists of an active joint input at a moderate frequency that is predominantly a first-order sinusoid with small high-order augmentations that provide beneficial characteristics in the shape space. Optimizing for efficiency with respect to actuator energy expenditure produces trivial zero-motion gaits that would cause swimmers to take infinite time to traverse between any two points. Adding metabolic considerations to the energy expenditure cost produces a range of non-trivial efficient gaits.

We then discuss how allocating a power budget affects optimal motion for swimmers with fixed passive coefficients. We see that increasing budgets for mechanical power expenditure allows for increased gait frequency and input motion magnitude. Beyond the budget required for the maximum-speed gait however, there are no benefits to increasing the power budget because it is not possible for the system to move faster with the additional energy.

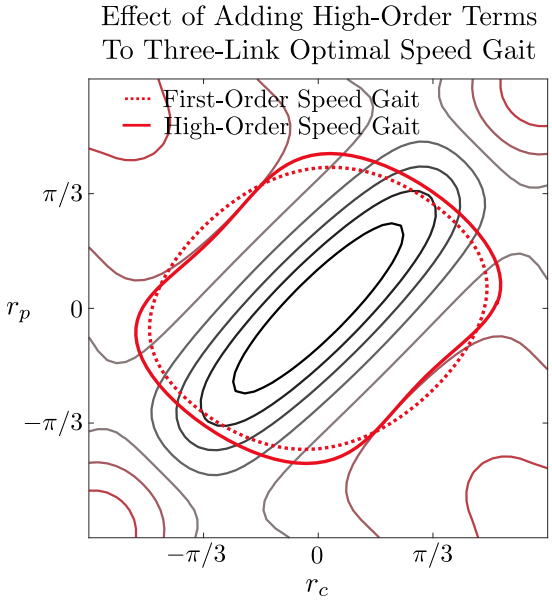


Figure 2.8: Effect of adding high-order terms to the fixed-coefficient optimal speed gait for the three-link swimmer: The optimal speed gait is shown (red line) superimposed on the forward-motion CCF (contour plot) alongside the effects of performing only the first-order terms of the same gait (red dotted line). The high-order terms aid the gait by slightly extending the gait along the positive-motion black diagonal area of the CCF while reducing the amount the gait enters the negative-motion red area.

**Speed** - The fixed-coefficient passive swimmers each have a bounded upper limit on speed, even with an unbounded power budget, because of the passive dynamics between the active and passive joints [49]. At high input frequencies, the passive response phase lag shifts so that there is very little resultant net motion.

Results for optimizing swimmer speed are illustrated in Fig. 2.7(a) for the three-link swimmer and in Fig. 2.9(a) for the fish-tail swimmer. Both have very similar characteristics. The optimizer tends to converge towards a predominantly

sinusoidal input with a small amount of high-order motion. The high-order motion compensates for the imperfect tuning of the passive parameters by 'flicking' the passive tail and achieving beneficial characteristics in shape space. The contribution of high-order motion to the optimum-speed gait for the three-link swimmer is shown in Fig. 2.8. This additional high-frequency motion is rather expensive, however, requiring a large amount of energy to be supplied to the actuator, and disappears from optimal results as the power budget is reduced.

For the arbitrary suboptimal coefficients we chose, the optimizer found the optimal gait frequency to be approximately 1.85 Hz for the three-link swimmer and 1.95 Hz for the fish-tail swimmer. These optimal frequencies depend on the particular passive coefficients that are chosen for each system alongside the shape-mode description. Because of the different mechanics of the passive-elastic components, the only way to 'fairly' compare these two shape modes is by using the equal-power optimally-tuned results given in Fig. 2.6.

For the suboptimal passive coefficients we chose, the three-link maximum-speed gait locomotes at approximately 80% of the speed of the optimal-coefficient gait when the latter is performed at the same level of power exertion. For the fish-tail swimmer, the fixed-coefficient optimal-speed gait locomotes at roughly 75% of the speed of the corresponding optimal-coefficient fish-tail gait.

**Mechanical Efficiency** - Optimizing for gaits that are efficient with respect to the mechanical cost of transport does not produce motions that would be useful for physical swimmers. The optimizer converges to asymptotically-zero amplitude, producing zero-motion gaits that would take infinite time to traverse between any

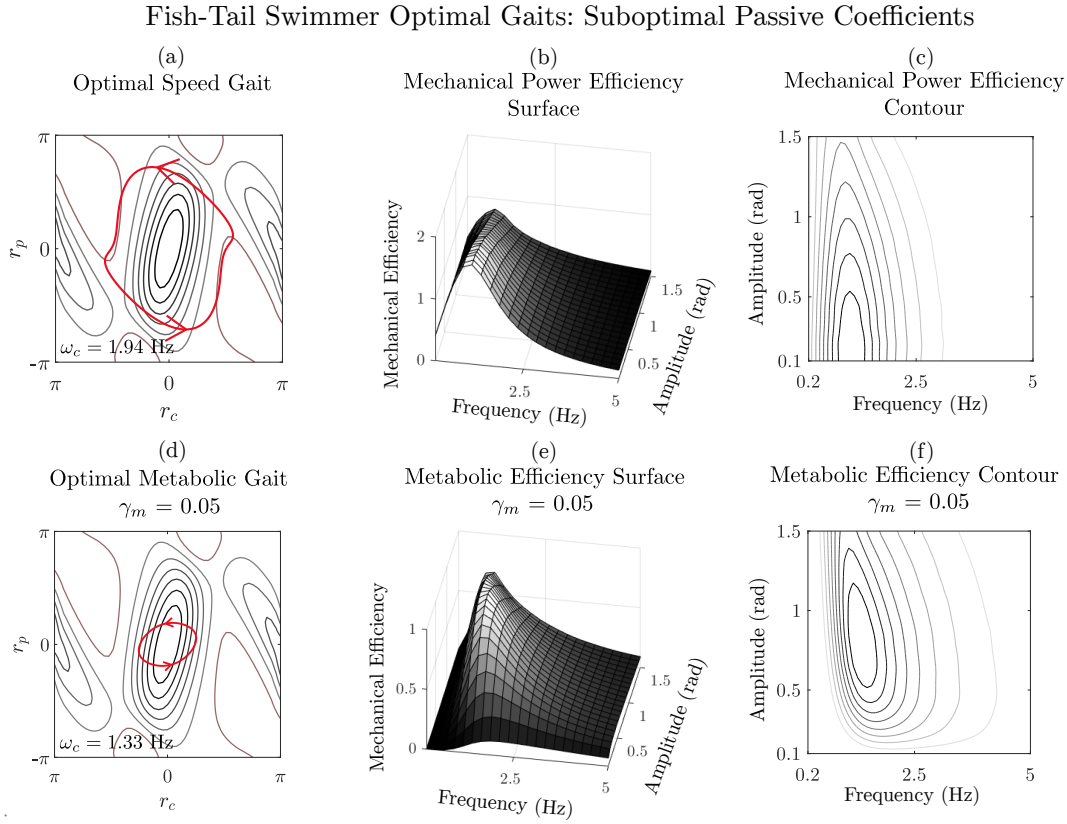


Figure 2.9: Results for the passive fish-tail swimmer across our three objective functions: (a) The maximum-speed gait (superimposed on the forward-motion CCF), which consists of a large sinusoid input augmented with a small amount of high-order motion that widens the gait in shape-space. (b-c) Locomotive efficiency of the passive fish-tail swimmer at simple input sinusoids for the mechanical power objective function. Considering only mechanical power costs results in an optimal gait with near zero input motion. (d) Results for optimizing a gait while taking into account a metabolic rate of $\gamma_m = 0.05$. (e-f) Locomotive efficiency of the passive fish-tail swimmer when considering metabolic drain. Taking into account energy overhead for the swimmer results in a non-trivial optimal gait.

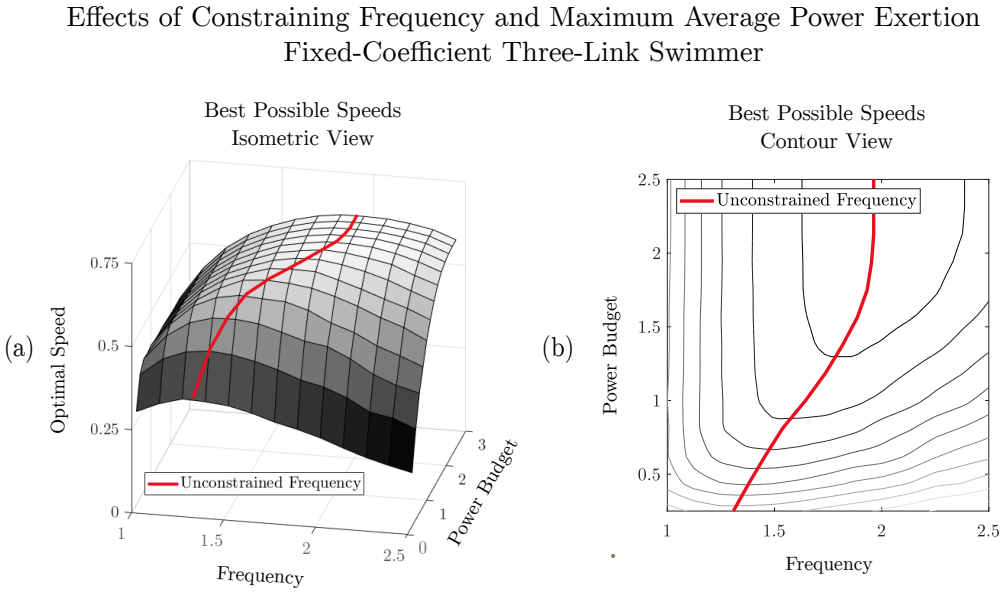


Figure 2.10: Optimal speed behavior over various power budgets and gait frequencies for the fixed-coefficient three-link swimmer. (a) Surface plot of optimal gait speeds over input power budgets and gait frequencies. Each node represents an individual gait speed optimization problem at a constrained level of maximum average power consumption and constrained input frequency. The red line represents the Pareto frontier that results from relaxing the frequency constraint in the optimizations. Over this Pareto frontier, gait speed trades off with power consumption. (b) Contour map of the same plot. Optimal frequency and gait speed increases with increasing power budget until the global maximum speed gait can be executed, after which there is no benefit or change in behavior from increasing power budget.

two points. This lack of motion is because power consumption increases more quickly than locomotive distance as active joint motion increases. Fig. 2.7(b-c) and Fig. 2.9(b-c) show mechanical efficiency across input sinusoid amplitudes and frequencies for both the three-link swimmer and fish-tail swimmer. Both efficiency surfaces indicate that minuscule gaits produce the most efficient motion when considering only actuator energy costs. For real-world swimmers, such low-motion gaits would produce negligible locomotion speeds and would not be useful for real-world applications. We can, however, indirectly optimize for mechanical efficiency by allocating a power budget and finding the highest speed possible that makes use of that budget. Performing this optimization across multiple power budgets results in a Pareto frontier of gaits across the dual objective functions of power expenditure and speed. We will discuss results of this process in §2.3.4.

**Metabolic Efficiency** - Physical implementations of these swimming systems will have continuous sources of energy loss other than the actuator, such as processor power consumption. These metabolic costs can be included into the energetic efficiency by introducing a component that scales with the metabolic rate $\gamma_m$ and the gait time period $T$. This produces a new metabolic energy objective function that encourages the optimizer to find gaits that balance speed and energy costs. This objective function, expressed in Eq. (2.42), allows for efficiency optimizations that produce meaningful results.

Optimizing with this new objective function produces results that change alongside the metabolic rate. When the swimmer metabolism is near zero, optimizations converge towards low displacement gaits that prioritize reduction of actuator

effort over actual locomotion, giving results similar to mechanical efficiency optimization. For swimmers with very high metabolic costs, the time-scaling term dominates the energy consumed by the active joint and the swimmer is encouraged to find speedy gaits. Such optimizations produce speed-optimal gaits like those shown in Fig. 2.7(a) and Fig. 2.9(a). Middling metabolism values, where metabolic costs rival transport costs and neither of the terms dominate, produce gaits that compromise between these two extremes. An example of such a metabolic gait is illustrated in Fig. 2.7(d) for the three-link system and in Fig. 2.9(d) for the fish-tail system. By optimizing for a metabolic rate $\gamma_m = 0.05$, we find a non-trivial optimal gait for each system that is much more energetically efficient than the max speed gait. Fig. 2.7(e-f) and Fig. 2.9(e-f) show metabolic efficiencies for the two swimmers across different sinusoidal inputs. Unlike the mechanical power objective function shown in Fig. 2.7(b-c) and Fig. 2.9(b-c), the metabolic objective function results in non-trivial optimal gaits even without mandating a minimum energy expenditure.

### 2.3.4 Effects of Allocating Power Budgets

In our previous work [20] we found efficient gaits by allocating an energy budget and finding the highest-speed gait that made use of this budget. The methodology of optimizing for speed given a power budget rather than directly for mechanical efficiency produces a single non-trivial gait that both maximizes speed-at-power and minimizes power-at-speed. For the fully active problem this most-efficient gait

gives the optimal solution at any desired power level by a simple time-scaling of the same path through shape space. As we showed in §2.3.1, this time-scaling property of the optimal gait across all power budgets can be extended to passive systems with mutable passive coefficients by rescaling the system stiffness and damping alongside the gait frequency. This solution is less simple for passive systems with predetermined immutable coefficients, however, because the elastic joint reacts differently to a high-power input than it does to a low-power input. For the fixed-coefficient passive problem, different energy budgets will result in differently-shaped gaits until the budget exceeds that of the maximum speed gait.

Although we cannot perform a simple time rescaling of a particular optimal gait for fixed-coefficient passive systems, we can examine the results of speed optimizations across different power budget allocations. In Fig. 2.10(a-b) we show optimization results for maximum possible speed across problems of constrained frequency and constrained maximum average power exertion. In these figures, each point on the greyscale surface is the result of an individual speed-optimization problem subject to the corresponding power and frequency constraints. The red curve represents the optimization problem of constrained power exertion while allowing the frequency to become an optimization variable. This line represents a Pareto frontier of optimal gaits over the two competing objective functions of gait speed and gait power consumption.

At low levels of power exertion, the motor cannot output enough acceleration to keep up with the best possible overall input frequency, so lower frequencies are more successful. As the power limit constraint is relaxed, the optimal frequency

and speed increases until the maximum-speed frequency is reached. Past this point, increasing the power budget provides no benefit, as the motor is already capable of performing the maximum speed gait. Results for the fish-tail system are qualitatively identical to those of the three-link system.

### 2.3.5 Transfer-Function Linearization of Passive Dynamics

In this section, we lay out a method for the transfer-function linearization of passive dynamic systems. We first discuss a theoretical overview of how the passive linearization works and when it should be used, and then derive the linearization formuli in their entirety.

#### 2.3.5.1 Overview

Our previous work [49] optimizing passive swimming in low Reynolds-number systems used Laplace transforms and frequency-space analysis to estimate the passive joint limit cycles. Crucial to this previous work was the approximation that the metric $M$ is constant, which allowed us to perform Laplace transforms and directly estimate the passive transfer function. Unfortunately, this technique is not suitable for inertia-dominated swimmers. The reduced mass matrix $M_r$ is heavily dependent on $r$, so the assumptions required for frequency domain analysis are no longer valid. Additionally, assuming that $M(r)$ is constant on $r$ eliminates our ability to factor in centrifugal and Coriolis forces, as these are calculated from derivatives

of the mass matrix. For inertial systems, we can no longer directly calculate the transfer function, and must instead approximate it numerically.

However, we can use the estimated limit cycle as a linearization point. This linearization allows us to estimate how the passive component of the gait will respond to changes in input motion and is useful for shaping passive responses. For simple passive systems, the limit cycle response to a sinusoidal input can be found by scaling and phase-shifting the input signal. This action of scaling and phase shifting the input signal is often referred to as the transfer function. Using the Fourier transform to break complex signals into a finite number of simple sinusoids, we can calculate the transfer function for each of the input/output Fourier sinusoid pairs. We then linearize around these Fourier parameters by enforcing that each transfer function stays accurate at nearby input points. This linearization process is illustrated in Fig. 2.11.

This process allows direct shaping and manipulation of passive motion to suit an objective function, even for slightly nonlinear systems. Given an input function and the passive response, we can reliably estimate exactly how to adjust the active input to achieve desirable passive output motion.

There are two primary assumptions required for this approach to linearizing passive dynamics. The first assumption is that the passive components are not heavily nonlinear. In effect, the passive components must have approximately symmetric spring and damping coefficients in the relevant shape-space regime. It is possible to reliably linearize more complicated systems, such as those with asymmetric or nonlinear spring functions, but the process is slightly more involved.

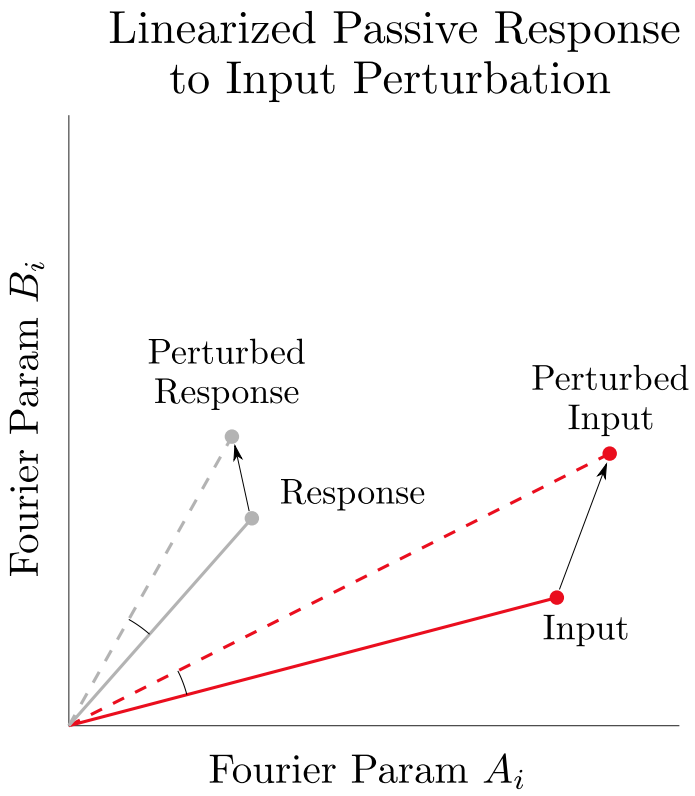


Figure 2.11: Linearized Passive Response to Input Perturbation: When a pair of control input Fourier coefficients are perturbed a small amount, the transfer-function linearized response of the corresponding passive coefficients is to maintain the same phase offset and amplitude ratio relative to the forcing input. In the Fourier plane, the perturbed input and linearized perturbed response make similar triangles with the original input and original passive response. This linear approximation can be used to generate gradients useful for gradient descent on passive dynamics.

We intend to expand this in future work.

The second assumption required for this approach is that there is sufficient shape activity to excite the passive dynamics. Gaits with very low motion will not

produce a measurable output response, making observing the transfer function impossible. To account for this, at each input-output coefficient pair, we check to see if the input magnitude exceeds some minimum threshold that is sufficient to excite measurable passive dynamics. For input-output pairs that exceed this minimum excitation, we can expect accurate results using the methods described above. For signals that do not minimally excite the passive dynamics, we instead approximate the transfer function using the magnitude-averaged transfer functions at other frequencies.

#### 2.3.5.2 Derivation

First, we note that each of the pairs of Fourier coefficients from our input control function in (2.17) are innately expressed in the Cartesian coordinate frame ($a_c$,$b_c$). We can equivalently express them using the magnitude $M_c$ and phase $\theta_c$ in polar coordinates using the amplitude-phase coordinates ($M_c$,$\theta_c$). From here on, we drop the subscript $i$ from (2.17) for readability and will only discuss one of the $n$ input-output Fourier coefficient pairs, because the derivation is the same for all pairs,

$$\begin{aligned} M_c &= \sqrt{a_c^2 + b_c^2} \\ \theta_c &= \tan^{-1}(b_c/a_c) \\ a_c &= M_c \cos\theta_c \\ b_c &= M_c \sin\theta_c. \end{aligned} \tag{2.43}$$

We write the passive response Fourier coordinates in the polar frame ($M_u$,$\theta_u$) and Cartesian frame ($a_u$,$b_u$) using the numerically observed transfer function. Specifically, we write them in terms of the magnitude output/input ratio $R$ and phase-shift $\Delta\theta$,

$$\begin{aligned} M_u &= RM_c \\ \theta_u &= \theta_c + \Delta\theta \\ a_u &= M_u \cos\theta_u = RM_c \cos(\theta_c + \Delta\theta) \\ b_u &= M_u \sin\theta_u = RM_c \sin(\theta_c + \Delta\theta). \end{aligned} \tag{2.44}$$

The bulk of this derivation comes from noting that the transfer function magnitude ratio and phase shift values act conveniently in the Fourier-plane polar coordinates, and using these simple polar transforms to perform analysis on the Cartesian coordinates for the input and output Fourier expressions of the swimmer shapes. The transfer function values $R$ and $\Delta\theta$ can be calculated from a numerical simulation of a single gait by fitting an $n^{th}$-order Fourier series to the

periodic passive response and estimating the transfer function values for each set of active-passive coefficient pairs,

$$R = \frac{M_u}{M_c}, \tag{2.45}$$

$$\Delta\theta = \theta_u - \theta_c. \tag{2.46}$$

We then assert that these transfer function values will stay approximately constant at nearby input parameters and use them to calculate gradients of the passive response parameters with respect to the control parameters

$$\begin{aligned} \frac{\partial a_p}{\partial p_c} &\approx \frac{\partial M_c}{\partial p_c} R\cos(\theta_c + \Delta\theta) - \frac{\partial \theta_c}{\partial p_c} R M_c \sin(\theta_c + \Delta\theta), \\ \frac{\partial b_p}{\partial p_c} &\approx \frac{\partial M_c}{\partial p_c} R\sin(\theta_c + \Delta\theta) + \frac{\partial \theta_c}{\partial p_c} R M_c \cos(\theta_c + \Delta\theta). \end{aligned} \tag{2.47}$$

Here, we use $p_c$ as a stand-in for either $a_c$ or $b_c$ because the equations are the same for both. The gradients of the control magnitude are found from Eq. (2.43) by

$$\begin{aligned} \frac{\partial M_c}{\partial a_c} &= \frac{a_c}{\sqrt{a_c^2 + b_c^2}} = \frac{a_c}{M_c} \\ \frac{\partial M_c}{\partial b_c} &= \frac{b_c}{\sqrt{a_c^2 + b_c^2}} = \frac{b_c}{M_c}, \end{aligned} \tag{2.48}$$

and similarly the gradients of the control phase are found as

$$\begin{aligned}\frac{\partial \theta_c}{\partial a_c} &= \frac{-b_c}{M_c^2} \\ \frac{\partial \theta_c}{\partial b_c} &= \frac{a_c}{M_c^2}.\end{aligned} \tag{2.49}$$

This method allows for the calculation of gradients of most of the passive Fourier coefficients with respect to the active Fourier coefficients. However, there are two sets of gradients that cannot be cleanly calculated using this method: gradients with respect to the controlled offset parameter $a_{c,0}$ and gradients with respect to the control frequency $\omega$. To estimate these gradients, we make two assumptions. The first is that changing the control offset parameter elicits only a small response to the uncontrolled Fourier parameters $p_u$ that can be neglected, since the passive response will stay centered around zero deflection for symmetric springs and dampers

$$\nabla_{a_{c,0}} p_u = 0. \tag{2.50}$$

The second assumption is that the passive response frequency will be the same as the control input frequency,

$$\frac{\partial \omega_u}{\partial \omega_c} = 1. \tag{2.51}$$

This assumption should be valid so long as the gait is allowed to reach the limit cycle and there are minimal external periodic forces acting through the fluid.

Because we cannot calculate derivatives of the transfer function using our single sample point, we cannot estimate how changing the control frequency will alter the response. We set these other gradient values to zero and use multi-start gradient descent to encourage full exploration of the frequency space.

Equations (2.47-2.51) produce accurate approximations of the true gradients whenever the magnitude $M_c$ is sufficiently large that it causes a meaningful passive response and the singularity at $M_c = 0$ is avoided. In order to extract meaningful transfer function values, we need to minimally excite the passive dynamics at that frequency and observe the response. However, this minimal excitation is not always guaranteed. It is not a given that optimal gaits will have significant activity across all of the Fourier components, so we need to adapt these equations to be accurate when $M_c$ is small.

We perform this adaptation by saying that the transfer function values for input parameters near zero be approximated by using the magnitude-averaged transfer functions for the rest of the Fourier components

$$\begin{aligned} R_{avg} &= \frac{\sum M_i R_i}{\sum M_i} \\ \Delta\theta_{avg} &= \frac{\sum M_i \Delta\theta_i}{\sum M_i}. \end{aligned} \tag{2.52}$$

We then choose some threshold $\epsilon$ for minimum excitation magnitude past which we can extract meaningful information from the passive response. For the simulated systems discussed in this chapter, we found success by choosing the mini-

mum magnitude to be one angular degree of excitation at that frequency, although this value will likely have to be increased for experimental systems where limitations on passive joint sensing resolution will come into play. We then produce weighted transfer function values $R_w$ and $\Delta\theta_w$ for each Fourier frequency that use the magnitude-averaged values for magnitudes of zero and smoothly scale to the numerically predicted transfer function values when the magnitude reaches the minimum excitation value. This produces equations like so for low-excitation magnitudes $M_c < \epsilon$,

$$\begin{aligned} R_w &= R\frac{M_c}{\epsilon} + R_{avg}\left(1 - \frac{M_c}{\epsilon}\right) \\ \Delta\theta_w &= \Delta\theta\frac{M_c}{\epsilon} + \Delta\theta_{avg}\left(1 - \frac{M_c}{\epsilon}\right). \end{aligned} \tag{2.53}$$

We can then substitute these weighted transfer function values into (2.44) and again calculate the gradients to get the following equations

$$
\begin{aligned}
&\frac{\partial a_u}{\partial p_c} \approx \\
&\frac{\partial M_c}{\partial p_c} R_w \cos(\theta_c + \Delta\theta_w) - \frac{\partial \theta_c}{\partial p_c} R_w M_c \sin(\theta_c + \Delta\theta_w) \\
&\quad + \frac{\partial R_w}{\partial p_c} M_c \cos(\theta_c + \Delta\theta_w) - \frac{\partial \Delta\theta_w}{\partial p_c} R_w M_c \sin(\theta_c + \Delta\theta_w), \\
&\frac{\partial b_u}{\partial p_c} \approx \\
&\frac{\partial M_c}{\partial p_c} R_w \sin(\theta_c + \Delta\theta_w) + \frac{\partial \theta_c}{\partial p_c} R_w M_c \cos(\theta_c + \Delta\theta_w) \\
&\quad + \frac{\partial R_w}{\partial p_c} M_c \sin(\theta_c + \Delta\theta_w) + \frac{\partial \Delta\theta_w}{\partial p_c} R_w M_c \cos(\theta_c + \Delta\theta_w).
\end{aligned}
\tag{2.54}
$$

Gradients of the weighted transfer function values with respect to the Fourier coefficients at this frequency are found by the chain rule as

$$
\begin{aligned}
\frac{\partial R_w}{\partial p_c} &= \frac{\partial R_w}{\partial M_c} \frac{\partial M_c}{\partial p_c} \\
\frac{\partial \Delta\theta_w}{\partial p_c} &= \frac{\partial \Delta\theta_w}{\partial M_c} \frac{\partial M_c}{\partial p_c},
\end{aligned}
\tag{2.55}
$$

The gradients of magnitude with respect to Fourier parameters can be found in (2.48), and the gradients of the weighted transfer function values with respect to the control magnitude are found as

$$\begin{aligned}\frac{\partial R_w}{\partial M_c} &= \frac{R}{\epsilon} - \frac{R_{avg}}{\epsilon} + \frac{\partial R_{avg}}{\partial M_c}\left(1 - \frac{M_c}{\epsilon}\right) \\ \frac{\partial \Delta\theta_w}{\partial M_c} &= \frac{\Delta\theta}{\epsilon} - \frac{\Delta\theta_{avg}}{\epsilon} + \frac{\partial \Delta\theta_{avg}}{\partial M_c}\left(1 - \frac{M_c}{\epsilon}\right),\end{aligned} \tag{2.56}$$

Finally, we calculate the gradients of the magnitude-averaged transfer function values with respect to the control magnitude

$$\begin{aligned}\frac{\partial R_{avg}}{\partial M_c} &= \frac{R\sum M_i - \sum(M_i R_i)}{(\sum M_i)^2} \\ \frac{\partial \Delta\theta_{avg}}{\partial M_c} &= \frac{\Delta\theta\sum M_i - \sum(M_i \Delta\theta_i)}{(\sum M_i)^2}.\end{aligned} \tag{2.57}$$

These equations are sufficient to reliably predict changes in output passive motion given changes in input motion. Equations (2.47-2.51) are used for Fourier components with sufficient excitation, and Equations (2.52-2.57) are used for Fourier components that are not minimally excited. Fig. 2.12 compares predictions from this linearization to the full simulated dynamics of the passive Purcell swimmer used in this work.

Overall, the process of linearizing passive dynamics is as follows: We first observe the steady-state passive response to the control input and fit the passive response to an $n^{th}$-order Fourier series. For each of the $n$ sets of input-output Fourier coefficients, we determine if the control input magnitude exceeds our minimum excitation threshold. Based on this determination, we either calculate the transfer function from the fit of the passive response or estimate the transfer func-

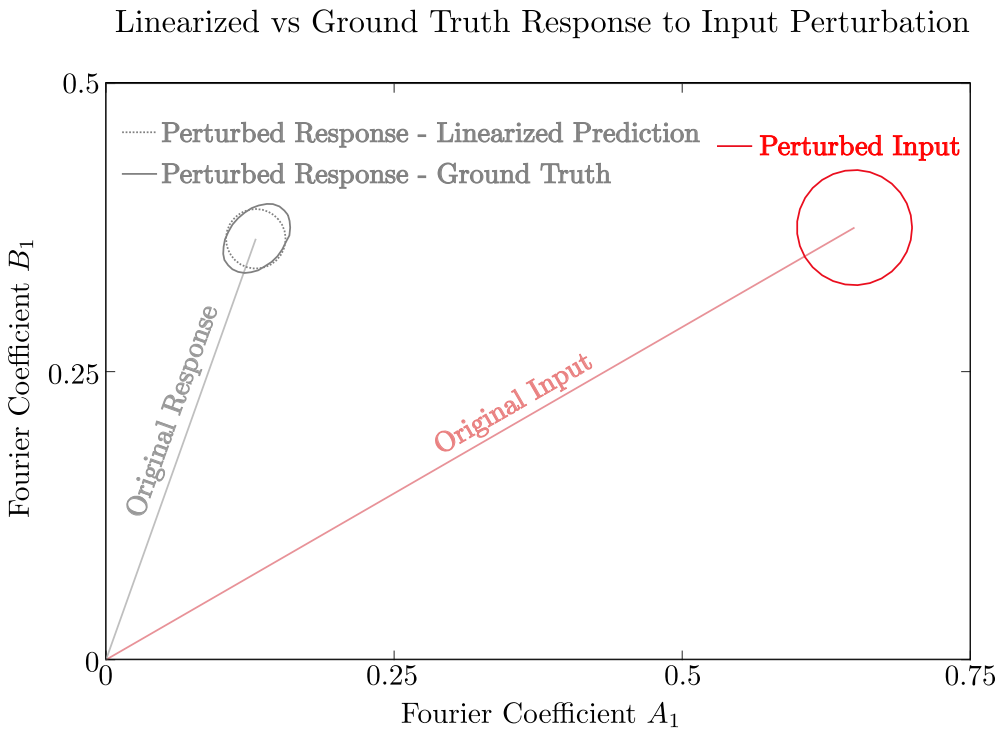


Figure 2.12: Comparison of linearized dynamics to the ground truth response at an example gait for the passive Purcell swimmer in Fourier space. For a set of input perturbations in a circle around the original input in Fourier space, the linear response predictions form a circle around the original passive response. These predictions match the elliptical ground truth responses closely enough to perform reliable gradient descent.

tion from the activated sets of coefficients. Once we have an approximate transfer function for each Fourier input-output pair, we estimate the gradients of the passive response coefficients with respect to the input signal coefficients by asserting that the transfer function is locally constant to the input parameters. In practice, this transfer-function gradient calculation looks like forming pairs of similar triangles in parameter space, as illustrated in Fig. 2.11.

For an infinitesimal sphere of perturbations about the original input signal in Fourier space, every point on the perturbation sphere maps to a point on the linear response perturbation ellipsoid. This ellipsoid provides a first-order fit of the true nonlinear perturbation response projection and is sufficiently accurate to indirectly shape the passive signal by varying the input parameters. We will use this method in later sections of the chapter to simultaneously shape the active joint and passive joint response to optimize the speed and energetic objective functions via gradient descent.

### 2.3.6 Performing Gradient Descent

In this section, we cover the process of estimating gradients of the objective functions introduced in §2.3.2. We first provide an overview of the process, and then derive specific gradients useful for the calculations used in this work.

#### 2.3.6.1 Overview

In addition to providing intuition, the CCF is also useful for estimating gradients of displacement with respect to a gait's path through shape-space. This makes it possible to perform gradient descent on our previously proposed objective functions by varying the gait parameters that control the motion of the active joint. In general, our approach is to vary these active parameters, use the linearized passive dynamics from §2.3.5 to predict variation in passive shape-mode behavior, and

combine both of these gait alterations on the CCF to examine how altering these control parameters changes the net displacement over the cycle. We can use similar techniques to estimate gradients on gait period and total power consumption to shape gaits that optimize for speed and energetic efficiency. By using a gradient descent solver that allows nonlinear constraints such as MATLAB's fmincon, we can add additional realism to the problem, such as limits on maximum actuator acceleration and bounds on allowable joint motion.

First, we explicitly choose our optimization parameters. These will be the $n$ pairs of Fourier coefficients $a_{1-n}$ and $b_{1-n}$, the Fourier offset coefficient $a_0$, and the gait frequency $w$. From an arbitrary seed gait, we perform the numerical dynamic gait simulation, calculate the gait's net displacement and cost, and fit the passive joint response to an $n^{th}$ order Fourier series. This fitting allows us to estimate gradients of the passive response Fourier parameters using the linearized passive dynamics outlined in §2.3.5.

By taking sequences of partial derivatives of the Fourier formulation (2.17), we can determine gradients of shape $\nabla_p r$, gradients of shape velocity $\nabla_p \dot{r}$, and gradients of shape acceleration $\nabla_p \ddot{r}$ with respect to all of the active and passive Fourier coefficients $p$ for all points on the gait. These gradients are derived explicitly in §2.3.6.2. We also take derivatives of the components of each of our objective functions $\eta$ with respect to the gait parameters, finding gradients of displacement $\nabla_p D$, gradients of mechanical cost $\nabla_p E_\tau$, and the gradient of gait period $\nabla_p T$, all in terms of the Fourier gradients described above. As part of this process, we evenly sample gait properties across the gait cycle so that total behavior over the

gait is taken into account. This process is performed in §2.3.6.3. We can then estimate the gradients of our chosen objective function with respect to the active and passive Fourier parameters.

Once we have gradients of the chosen objective function with respect to all of the active and passive Fourier parameters, we can use the linearized passive dynamic gradients from §2.3.5 to take into account how changing a control input optimization parameter $p_c$ will also change all of the passive Fourier parameters

$$\frac{\partial \eta}{\partial p_{c,full}} = \frac{\partial \eta}{\partial p_c} + \sum_i \frac{\partial \eta}{\partial p_{u,i}} \frac{\partial p_{u,i}}{\partial p_c}. \tag{2.58}$$

These final gradients allow us to perform gradient descent on our chosen objective function to simultaneously shape the control joint and passive joint path through shape space in a way that maximizes displacement subject to some cost. For the systems discussed in this chapter, we fix the control parameter $a_{c,0}$ at zero because it is unlikely that joint bias will produce gaits with optimal forward-direction behavior. We leave the control frequency parameter $w_c$ free-floating despite the incomplete passive dynamic gradient to leverage information about how it changes gait period, and use multi-start gradient descent for $w_c$ in order to ensure the adequate exploration of the frequency parameter. We also add adjustable joint limit constraints and joint acceleration constraints to reflect actuator limitations and use MATLAB's fmincon to perform gradient descent subject to these nonlinear constraints.

#### 2.3.6.2 Gradients of the Fourier Series

In order to perform gradient descent on the optimization coefficients that control active joint motion, we must calculate gradients of shape, shape velocity, and shape acceleration with respect to the active and passive Fourier parameters. For a $n^{th}$ order Fourier parameterization, we are concerned with gradients on the offset parameter, $a_0$, the pairs of Fourier coefficients at each Fourier frequency $a_{1-n}$, and $b_{1-n}$, and the fundamental frequency $\omega$. As these equations are the same for both of the two sets, we will show only calculations for a set of Fourier parameters on arbitrary joint $j$. We start with the Fourier parameterization of the gait shape on joint $j$,

$$r_j = a_0 + \sum_{i=1}^{n} (a_i \cos i\omega t + b_i \sin i\omega t). \tag{2.59}$$

From this, the gradients of shape with respect to our Fourier parameters $\nabla_p r_j$ can be calculated

$$\begin{aligned} \frac{\partial r_j}{\partial a_0} &= 1, \\ \frac{\partial r_j}{\partial a_i} &= \cos i\omega t, \\ \frac{\partial r_j}{\partial b_i} &= \sin i\omega t, \\ \frac{\partial r_j}{\partial \omega} &= -ita_i \sin i\omega t + itb_i \cos i\omega t. \end{aligned} \tag{2.60}$$

Next, we take a time-derivative of (2.59) to get shape velocity in terms of the

Fourier parameters over the course of the gait

$$\dot{r}_j = \sum_{i=1}^{n} (-i\omega a_i \sin i\omega t + i\omega b_i \cos i\omega t), \tag{2.61}$$

and then use this parameterization to calculate $\nabla_p \dot{r}_j$

$$\begin{aligned} \frac{\partial \dot{r}_j}{\partial a_0} &= 0, \\ \frac{\partial \dot{r}_j}{\partial a_i} &= -i\omega \sin i\omega t, \\ \frac{\partial \dot{r}_j}{\partial b_i} &= i\omega \cos i\omega t, \\ \frac{\partial \dot{r}_j}{\partial \omega} &= -i^2 \omega t a_i \cos i\omega t - i^2 \omega t b_i \sin i\omega t. \end{aligned} \tag{2.62}$$

Finally, we take a time-derivative of (2.61) to parameterize shape acceleration from the Fourier coefficients

$$\ddot{r}_j = \sum_{i=1}^{n} (-i^2\omega^2 a_i \cos i\omega t - i^2\omega^2 b_i \sin i\omega t), \tag{2.63}$$

and use this parameterization to calculate $\nabla_p \ddot{r}_j$

$$\begin{aligned}
\frac{\partial \ddot{r}_j}{\partial a_0} &= 0, \\
\frac{\partial \ddot{r}_j}{\partial a_i} &= -i^2\omega^2 \cos i\omega t, \\
\frac{\partial \ddot{r}_j}{\partial b_i} &= -i^2\omega^2 \sin i\omega t, \\
\frac{\partial \ddot{r}_j}{\partial \omega} &= i^3\omega^2 t a_i \sin i\omega t - i^3\omega^2 t b_i \cos i\omega t.
\end{aligned} \tag{2.64}$$

#### 2.3.6.3 Gradients of the Objective Functions

We propose three objective functions for which we can optimize the gaits of our underactuated swimming systems. Here, we will show calculations for gradients of these objective functions with respect to all of the active and passive Fourier parameters. The first of the objective functions is swimmer speed,

$$\eta_v = \frac{D}{T}. \tag{2.65}$$

We calculate the gradient of speed with respect to the Fourier parameters as

$$\nabla_p \eta_v = \frac{\nabla_p D}{T} - D\frac{\nabla_p T}{T^2}. \tag{2.66}$$

The calculation of $\nabla_p D$ is described in detail in our previous work [46, 48, 49], but for completeness we provide a brief outline here. We first approximate the gait as $n$ evenly-spaced points connected by line segments. Each point $r_k$ along the gait forms a triangle with its neighbors $r_{k-1}$ and $r_{k+1}$. For each point, we care to find

how the area of this triangle changes on the Fourier parameters, and multiply this with the value of the CCF at $r_k$ calculated from (2.20) giving an estimate of how the motion of this point contributes to locomotion in the desired direction.

This process is made fairly simple by recognizing that only the outward motion of $r_k$ with respect to this triangle will contribute to area change in the triangle. We can calculate the direction of outward motion $\hat{e}_\perp$ as being perpendicular to the vector along the base of the triangle $\hat{e}_\parallel$. This value is found using central differencing

$$r_{k+1} - r_{k-1} = e_\parallel = \ell \hat{e}_\parallel, \tag{2.67}$$

where $\ell$ is the base length of the triangle. This vector is rotated depending on the chirality of the gait such that $\hat{e}_\perp$ points in the outward direction. We can then find the outward component of the gradient of gait shape at this point

$$\nabla_p r_{k,\perp} = \nabla_p r_k \cdot \hat{e}_\perp \tag{2.68}$$

and use this value along with the CCF at each point to estimate the gradient of locomotion in the desired direction, taking care not to confuse net gait displacement $D$ with the value of the CCF $D\mathbf{A}_k$

$$\nabla_p D = \sum_{k=1}^{n} \left( \frac{\ell}{2} \nabla_p r_{k,\perp} D\mathbf{A}_k \right). \tag{2.69}$$

The gradient of the gait period $\nabla_p T$ is very easy to calculate from its definition $T = 2\pi/\omega$ because the only parameter that affects the gait period is the

fundamental frequency of the active joint

$$\frac{\partial T}{\partial \omega} = -\frac{2\pi}{w^2}. \tag{2.70}$$

The second objective function we consider is the energetic efficiency

$$\eta_\tau = \frac{D}{E_\tau}, \tag{2.71}$$

which has a gradient similar to that of speed

$$\nabla_p \eta_\tau = \frac{\nabla_p D}{E_\tau} - D\frac{\nabla_p E_\tau}{E_\tau^2}. \tag{2.72}$$

Here, the only missing piece is the gradient of the energetic cost. We start by phrasing this as the integral of absolute mechanical power required to enforce gait motion of the active joint over the limit cycle $\phi$ of the gait through shape space

$$E_\tau = \oint_\phi |\tau_c \dot{r}_c| dr. \tag{2.73}$$

We can calculate this gradient as

$$\nabla_p E_\tau = \oint_\phi \text{sign}(\tau_c \dot{r}_c)(\dot{r}_c \nabla_p \tau_c + \tau_c \nabla_p \dot{r}_c) dr. \tag{2.74}$$

The gradient of active joint velocity $\nabla_p \dot{r}_c$ is found by (2.62), noting that the gradients of the active joint velocity with respect to the passive Fourier parameters are all zero. We then calculate $\nabla_p \tau$ by applying the gradient $\nabla_p$ to each of the terms in (2.14). Refer to Appendix D of our previous work [20] to see the details

of this lengthy calculation.

The final objective function we consider in this chapter is the metabolic efficiency, which takes into account some metabolic rate $\gamma_m$,

$$\eta_m = \frac{D}{E_\tau + \gamma_m T}. \tag{2.75}$$

This gradient can be calculated using the tools given above,

$$\nabla_p \eta_m = \frac{\nabla_p D}{E_\tau + \gamma_m T} - D\frac{\nabla_p E_\tau + \gamma_m \nabla_p T}{(E_\tau + \gamma_m T)^2}. \tag{2.76}$$

## 2.4 Conclusion

In this work, we showed that geometric mechanics provides a computationally simple framework for finding optimal gaits in inertial systems with passive elements. We described a process for optimizing swimming system for cases where swimmer passive parameters can be optimized alongside swimmer gait motion and for cases where swimmer passive parameters are fixed in place. We believe that these tools can accelerate the design process for locomoting systems with passive-elastic components by identifying promising families of motion, and that this analysis provides fundamental intuition that can be useful for the development of such systems.

We considered two models of inertial swimming systems with passive-elastic components. We used the passive three-link swimmer as a simple model to discuss gaits that leverage passive dynamics to optimize swimmer behavior. Then, we presented a fish-tail swimmer to demonstrate how to geometrically represent continu-

ously flexible systems and showed that this flexible swimmer leverages properties of motion that are qualitatively similar those used by the three-link swimmer.

By using passive parameters optimized for each system's unit-power limit cycle, we were able to generate the fairest possible comparison between the two systems, showing that a well-tuned fish-tail swimmer using our chosen curvature mode can be 20% faster than a well-tuned three-link swimmer at identical levels of power consumption.

For both of these swimming systems, we discussed results from optimizing with respect to three objective functions. We showed that optimizing with respect to speed produces a single highest-speed gait consisting of a simple sinusoid that in cases of suboptimal passive properties is augmented with a small amount of high-order motion that has beneficial characteristics in the swimmer shape space. This maximum speed is limited by the passive dynamics of the swimmer: attempting to increase motor frequency beyond this point without change to passive coefficients elicits a response with poor behavior in shape space. We also show that efficiency objective functions that take into account only motor power consumption result in zero-motion optimal gaits that would not actually be useful for a swimmer attempting to travel between two points. This result aligns with our previous findings for passive swimmers in the drag-dominated regime [49]. Finally, for both classes of swimmer, we demonstrate that including metabolic overhead in the energy expenditure calculation results in a range of useful gaits that produce nontrivial efficient locomotion.

In future work, we intend to investigate selection of passive shape-modes that

are most efficient for locomotion. Here, we assumed linearly varying mechanical properties across the length of our fish-tail swimmer, resulting in a particular passive shape mode that was more efficient than the three-link system. By intelligently choosing the moment of inertia of the tail along the backbone, we could attempt to design and optimize passive responses that are even more useful for locomotion. We also hope to examine swimmer locomotion in non-perfect fluids that allow for fluid drag and attempt to find optimal gaits for geometric swimming systems with both hydrodynamic mass and drag. Finally, we aim to extend this methodology to more complex systems, such as those with nonlinear springs that have been shown to have beneficial swimming properties in the past [54] or to systems with multiple active and passive shape modes, allowing for the modeling of traveling waves.

# Chapter 3: Locomotive Analysis and Thruster Design for a Robotic Sea Salp

## 3.1 Introduction

In this chapter, we study the locomotive properties of sea salps, which are chains of individual jellyfish-like agents. By individually thrusting through hydrodynamic jet mechanisms, they are capable of large-scale emergent locomotion that produces fluid flow over each zooid element of the chain [60]. These organisms execute unique gait patterns that arise from collective zooid action [13]. Sea salp biology is pictured in Fig. 3.1.

Our efforts here will be collected on two fronts. The first is to build a framework for a mathematical simulator that allows us to study the properties of salp locomotion and the emergent body structures that result from individual zooid thruster activity. For our second effort, we will design a soft origami thruster mechanism driven by twisted-and-coiled actuators that can be used to drive aquatic salp-inspired robots.

We hope that by studying salp locomotion, we can enable a new class of hydrodynamic swimming systems and learn more in turn about the physics and collective properties that drive the formation of biological salp structures.

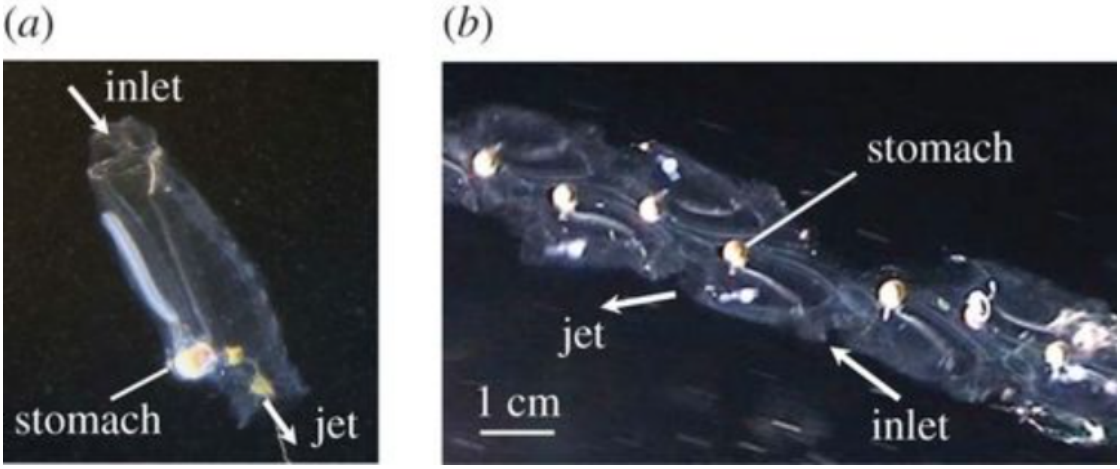


Figure 3.1: The biology of the sea salp. **a)** An individual zooid jellyfish-like agent that can produce period thrust by forcing water through its body. **b)** A chain of zooids that form the salp. Through individual agent action and the flexible connection between zooids, the salp takes on a larger structure that has favorable locomotive properties. Figure borrowed from the work of Kelly Sutherland [61], who we collaborated with on this project.

## 3.2 Locomotive Analysis

We model the sea salp as a modular series of individual zooid agents, each with a periodically firing thruster. We model each element as existing in a drag-dominated environment where fluid drag forces dominate inertial effects. This modeling effort will be similar in concept to recent works done analyzing kinematic snake robots actuated by wheel side-thrusts [64].

To allow for a variety of mechanical designs, we choose two ways of enabling flexibility within the system. In the first method, we assume that the individual zooid elements are rigid but are linked by passive, flexibly elastic joints at the discrete connection points. This means that for an $n$-link chain of zooid agents,

there are $n-1$ degrees of freedom that can be parameterized by the angles between the rigid links. In the alternative representation of the salp system, we attempt to address the continuum flexibility of biological salps by representing each agent as elastic rather than rigid. For simplicity, we model each zooid element as being capable of developing constant curvare over the arclength of the element. This means that for an $n$-segment chain of zooid agents there are $n$ degrees of freedom, each parameterized by the curvature of that salp segment. We leverage our previous works to develop models for both of these systems [48], which are illustrated in Fig. 3.2. For this work, we assume that the systems are travelling in the plane and can be modelled as SE(2) locomotors, and that we can ignore the influence of gravity on the links because gravity acts normal to the plane of translation.

### 3.2.1 N-Link Chain Salp Model

To begin constructing our model for the N-Link Chain salp, we first construct a coordinate frame for the robot. From previous work [21], we have found that an appropriate choice of body coordinates for chain systems is located at the average of the link centers and oriented along the average of the link orientations.

From this frame, we can build a mobile Jacobian for each arclength point $s$ along the spine of the salp that maps planar translations and rotations of the body frame $\mathring{g}$ and joint velocities $\dot{r}$ to motion of that point through the instantaneous optimal coordinate frame,

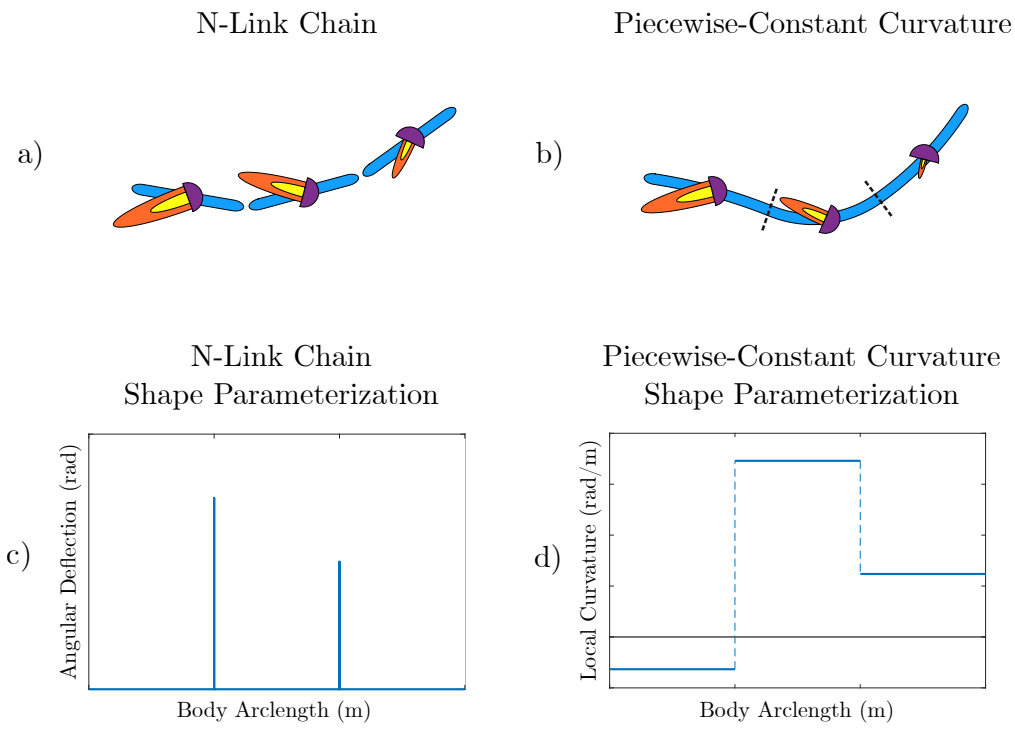


Figure 3.2: Illustrations of the shape deformations of the two salp systems we model in this work along with their shape parameterizations: (a) The N-Link chain model of the salp, with rigid links and passive flexibility between the zooids. Each link is acted on by a thruster that exerts a time-dependent force on the link. (b) The Piecewise-Constant Curvature (PCC) model of the salp, with flexible segments that are each capable of developing constant curvature across the link arclength. (c) The shape parameterization of the N-Link chain model. Curvature is zero across the arclength except for at the discrete connection points, where the curvature is a dirac function that integrates to give values for joint angle deflection. The salp shape is parameterized by these angular deflections. (d) The shape parameterization of the PCC salp. Curvature is constant across each of the independent zooid arclengths, but can instantaneously change between them. Unlike the N-Link chain model, the tangent vectors along the backbone are first-order differentiable.

$$\begin{bmatrix} \dot{x}_s \\ \dot{y}_s \\ \dot{\theta}_s \end{bmatrix} = J_s(s,r) \begin{bmatrix} \overset{\circ}{g} \\ \dot{r} \end{bmatrix}. \tag{3.1}$$

We define three types of force that act on the salp structure and must be at equilibrium. The first is the drag force that results from the viscous interaction between the fluid and the salp links as they move through the medium. The second are the thrust forces that result from the zooid actuators. The third type of forces acting on the salp are the restorative forces from the passive connection points.

For the first type of force, we define the physical interaction with each link on the fluid using a low Reynolds number resistive force model, where drag forces at each location are linearly proportional to that point's relative velocity in the fluid. We are particularly interested in the Jacobians at arclengths that correspond to link centers, as these points offer convenient physical properties. At the link centers, integrated drag force across the link can be expressed using a single force vector that is a linear function of the velocity of that link center. For the $i$th link, the drag force $F^d$ can be expressed as

$$F_i^d = -d \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \\ \dot{\theta}_i \end{bmatrix}, \tag{3.2}$$

where the drag matrix $d$ is a diagonal matrix constructed from the drag coefficient $c_d$, the drag ratio $\lambda$, and the link length $\ell$,

$$d = \begin{bmatrix} c_d\ell & 0 & 0 \\ 0 & \lambda c_d \ell & 0 \\ 0 & 0 & \frac{\lambda c_d \ell^3}{12} \end{bmatrix}. \tag{3.3}$$

Because $d$ is dependent only upon the link's geometric design and the properties of the surrounding fluid, we assume that each of the links in the chain has the same drag matrix.

Using the link center Jacobians $J_i$, we can combine and pull back the individual drag matrix contributions to retrive the drag metric, which relates salp body motions and joint velocities to generalized forces $F^d$ that acts each of these degrees of freedom as a result of the viscous interaction,

$$F^d = -D \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}, \tag{3.4}$$

where

$$D(r) = \sum J_i^T(r) d J_i(r). \tag{3.5}$$

For the second type of force acting on the salp, we develop a model for the thruster actuators. Each actuator is capable of producing a time-dependent thrust vector that acts on the zooid. For our simulation framework, the actuators can be placed anywhere along the links, and there is no restriction on maximum or minimum actuators per link. However, for the simulation results we will discuss later in this section, we will predominantly examine salp designs where there is one

thruster per zooid and the thrusters are coincident with the link centers. For the $j$th thruster that is oriented to fire $\phi_j$ radians away from the link-frame backwards direction, this results in an actuator force vector $F^a$ acting on the link that is a function of the time-dependent actuator thrust magnitude $f_j(t)$,

$$F_j^a(t) = \begin{bmatrix} f_j(t)\cos\phi_j \\ f_j(t)\sin\phi_j \\ 0 \end{bmatrix}. \tag{3.6}$$

For our simulator, the thrust profile $f_j(t)$ can be an arbitrary function that produces scalar value of thrust over time. For the results we present later, this function generally falls into one of two categories. In the first category, each actuator produces sinusoidal thrust of the same amplitude $A$ and period $T$, and adjacent actuators experience thrust magnitudes that are separated by a phase offset $\Delta t$. The thrust magnitude for the $j$th actuator is then

$$f_j(t) = \frac{A}{2} - \frac{A}{2}\cos\left(\frac{2\pi}{T}t + j\Delta t\right). \tag{3.7}$$

This produces a thrust magnitude that starts at zero and sinusoidally rises to magnitude $A$ before falling back to zero.

Alternatively, rather than synchronizing the thrusters to a cohesive sinusoidal signal, we can have each thruster fire stochastically and independently without requiring knowledge of what the neighbor is doing. For this thruster model, we still use a sinusoid to model how the thruster produces thrust while activated that is parameterized by maximum thrust $A$ and period $T$. However, rather than use

a phase offset, we add a stochastic delay time before the thruster fires again that is parameterized by the thruster uptime ratio $\eta$. When $\eta$ is 1, the thruster fires continuously with no break, when $\eta$ is zero the thruster does not fire at all, and when $\eta$ is 0.5 the thruster will begin firing again on average T seconds after the thruster deactivates. We produce the desired thruster uptime by enforcing a delay period after the thruster fires choosing a random uniformly distributed number between 0 and $2T(\frac{1}{\eta} - 1)$, which gives on average a total period producing the desired thruster uptime. For the most recent thruster ignition time $t_0$, this profile can be written as

$$f_j(t) = \left\{ \begin{array}{ccc} \frac{A}{2} - \frac{A}{2}\cos\left(\frac{2\pi}{T}(t-t_0)\right) & \text{when} & (t-t_0) \leq T \\ 0 & \text{when} & T < (t-t_0) \leq \text{rand}(T, \frac{2T}{\eta} - T) \end{array} \right\}. \tag{3.8}$$

The thrust vectors that result from these actuator force profiles can be pulled back into the salp's body frame and summed to find the resultant generalized force produced on each of the salp's degrees of freedom $F^a(t)$ that results from actuator activity,

$$F^a(t) = \sum J_j^T F_j^a(t). \tag{3.9}$$

For the final type of force that acts on the salp structure, we examine the elastic connection between the links. For this we define a spring stiffness $k$ between each joint that exerts a restorative force taking the links back to a straight line when

undriven by the actuators. We can construct a stiffness matrix $K$ that relates the joint angles to the generalized forces on the salp degrees of freedom $F^k$ that arise due to the passive components. Because the only springs are on the links between the zooid agents, all forces are internal to the salp and there are no force terms acting on the salp's SE(2) pose that are directly due to passive components. For the other degrees of freedom, the stiffness matrix is a diagonal matrix turning joint angle displacement into a restorative torque by action of the spring constant. This force function can be expressed as

$$F^k = -Kr \tag{3.10}$$

where for an $n$-link chain of zooid agents the stiffness matrix is an $n+2$ by $n-1$ matrix can be expressed as

$$K = \begin{bmatrix} 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 \\ k & 0 & \dots & 0 & 0 \\ 0 & k & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & k & 0 \\ 0 & 0 & \dots & 0 & k \end{bmatrix}. \tag{3.11}$$

We assume that these three forces are the only forces that will act on the salp while it locomotes. Under the low Reynolds number assumption, this means that

the forces must be at equilibrium. This produces a model whereby actuator input and joint stiffnesses result in generalized forces on the degrees of freedom that are immediately compensated for by a velocity on each mode that brings balance to the forces,

$$0 = F^d + F^a(t) + F^k. \tag{3.12}$$

We can combine the expressions we derived from Eq. (3.4), Eq. (3.9), and Eq. (3.10) to express this force balance in terms of the system shape and velocities,

$$0 = -D \begin{bmatrix} \overset{\circ}{g} \\ \dot{r} \end{bmatrix} + \sum J_j^T F_j^a(t) - Kr. \tag{3.13}$$

From this, we can easily solve for the body-frame and shape velocities that give balance to the actuator forces and passive joint torques,

$$\begin{bmatrix} \overset{\circ}{g} \\ \dot{r} \end{bmatrix} = D^{-1} \left( \sum J_j^T F_j^a(t) - Kr \right). \tag{3.14}$$

By numerically integrating this time-dependent ODE using a tool such as Euler's method or MatLab's `ode45`, we can investigate the shape and body motion evolution of the N-Link salp structure.

### 3.2.2 Piecewise-Constant Curvature Salp Model

Derivation of the Piecewise-Constant Curvature (PCC) salp model is virtually identical to the derivation of the N-Link Chain model, save with a different description of the shape mode parameterization on the salp which results in a different method to pull back drag forces into the body frame.

Here, we describe how we leverage our previous works on continuously flexible systems [28, 48] to develop a model for the salp that deforms smoothly on each zooid element rather than at discrete joints. For the shape parameterization of this salp, we use the curvature values of each of the zooid agents, meaning that for an $n$-segment salp there will be $n$ curvature values that together fully define the salp shape.

We use a similar methodology for constructing the optimal coordinate frame. The frame lies at the geometric center of the salp and is oriented along the average orientation of the infinitesimal arclength elements.

From this frame description, we can build a Jacobian which relates the body frame and shape motion of the salp to the resultant body-frame velocity of any point along the arclength as a function of arclength and system shape. At each point along the salp backbone, the lateral and longitudinal drag densities relate the vector of fluid flow relative to the local backbone tangent vector to an infinitesimal force that can be pulled back into the body frame. Unlike the n-link chain model where the swimmer backbone tangent vectors are undefined at the joints, the tangent vectors along the backbone are first-order differentiable at all points. This

results from the connection of sequential sections of backbone by the alignment of the corresponding tangent vectors.

Analagous to the individual drag matrix that we used to describe the physical behavior of each link in Eq. (3.3), we can write a matrix that describes the drag force response as a function of local velocity for each infinitesimal segment of the salp arclength that has length $\delta s$,

$$d = \begin{bmatrix} c_d \delta s & 0 & 0 \\ 0 & \lambda c_d \delta s & 0 \\ 0 & 0 & \frac{\lambda c_d \delta s^3}{12} \end{bmatrix}. \tag{3.15}$$

Because the $\delta s^3$ term is extra teensy-tiny, we can drop it from our integration process without harm. This lets us factor out the infinitesimal drag matrix for an infinitesimal section of salp arclength,

$$d = \begin{bmatrix} c_d & 0 & 0 \\ 0 & \lambda c_d & 0 \\ 0 & 0 & 0 \end{bmatrix} \delta s = d_{PCC} \delta s. \tag{3.16}$$

Viscous resistance to rotational velocity will arise from the pullback of individual element translational resistances. This pullback is done by integrating along the total arclength $L$ of the salp to find the collective resistances of the salp elements expressed in the degrees of freedom on the body frame,

$$D_{PCC}(r) = \int_0^L J(r,s)^T d_{PCC} J(r,s) \delta s. \tag{3.17}$$

Viscous drag forces that act on the salp degrees of freedom can then be expressed similar to Eq. (3.4) as

$$F^d = -D_{PCC} \begin{bmatrix} \overset{\circ}{g} \\ \dot{r} \end{bmatrix}. \tag{3.18}$$

As we had already expressed the thruster actuation pullback as a function of arclength along the salp, there is no need to recalculate for the continuously flexible system, and we can use Eq. (3.9) along with our previously discussed thruster activation profiles to model the effect of actuation on the salp.

The pullback of the elastic stiffness from shape values into the salp's degrees of freedom is similarly straightforward, resulting in an $n + 3$ by $n$ matrix for a salp composed of $n$ individual zooid segments. This resolves to be identical to Eq. (3.11) save for one more row and column in the matrix representing the fact that the continuously flexible salp has one more degree of freedom than its rigid-chain counterpart.

While the stiffness for the discrete-joint shape modes was analogous to a torsoinal spring, the stiffness for this salp formulation is more similar to that of an Euler-Bernoulli beam. Here, we assume that the mechanical properties of the zooid element resist torsional deformation and when displaced result in a restorative force to a straight beam segment. At each point in the beam there is an internal moment $m(s)$ that relates the instantaneous mechanical stiffness $k_m(s)$ to the instantaneous curvature $\kappa(s)$. Together, across the beam, all of these points act like a series of springs in parallel acting on the instantaneous local curvatures

that serve to pull the beam back to true. The potential energy of the $i$th individual zooid segment can be expressed as

$$\mathrm{PE}_i = \int_{s_{c,i}-\ell/2}^{s_{c,i}+\ell/2} \frac{1}{2} k_m(s)\kappa^2(s)\delta s, \tag{3.19}$$

where $s_{c,i}$ refers to the arclength representing the center of the zooid and $\ell$ represents the total zooid arclength.

By assuming that the zooid mechanical properties are constant throughout, we can remove the dependence on $s$, and by defining our shape mode such that the local curvature $\kappa$ is also constant on an individual zooid agent and corresponds to the $i$th shape mode $r_i$, we can rewrite this relationship as

$$\mathrm{PE}_i = \int_{s_{c,i}-\ell/2}^{s_{c,i}+\ell/2} \frac{1}{2} k_m r_i^2 \delta s = \frac{1}{2} k_m r_i^2 \ell. \tag{3.20}$$

We can express this continous-flexibility model in the shape-mode by taking the second derivative of the potential energy with respect to the shape mode,

$$k_{PCC} = \frac{\delta^2 \mathrm{PE}}{\delta r^2} = k_m \ell. \tag{3.21}$$

With these shape-mode stiffnesses for the PCC system, we can flesh out the stiffness matrix $K$ as in Eq. (3.11) and solve for the body-frame and shape-mode velocities that through the drag metric give balance to the actuator and stiffness forces the same way we did for the N-Link Chain system,

$$\begin{bmatrix} \overset{\circ}{g} \\ \dot{r} \end{bmatrix} = D_{PCC}^{-1} \left( \sum J_j^T F_j^a(t) - Kr \right). \tag{3.22}$$

In the same way as the N-Link Chain system, we can numerically integrate this relationship over time for any desired number of salp segments or thruster actuation profile to examine the shape and pose evolution of the salp as the thrusters execute their actions.

## 3.3 Simulator Design

To actually implement a simulator for this, I wrote a MatLab GUI through `appdesigner` called `salpPlotter` to load and execute simulations for arbitrary salp designs including configurable thruster placements, thruster actuation profiles, number of zooid units, zooid flexibility, initial conditions, and simulation times. A snapshot of the GUI with a loaded minimal 3-link rigid-zooid salp structure and symmetric thruster placement with a sinusoidal activation profile is shown in Fig. 3.3.

This simulator offers multiple different ways of visualizing and analyzing the evolution of salp behavior over time. In Fig. 3.4 we illustrate GUI plots that result from execution of the simulation described above. We see a circular evolution through the ambient SE(2) space, which is representative of the three-dimensional helical salp structure projected onto two dimensions. Along this circular arc we see trajectory portions where the salp turns quickly that are connected by straightaways. These turning motions correspond to high amplitudes of thruster firing

activity, which cause structure buckling on the passive joints and emergent maneuvering. The periodic buckling of the passive joints can be seen in the lower plot, and correspond roughly to the actuation intensity shown in the bottom section of Fig. 3.3. After simulation, this pose and shape evolution data can be exported and saved to a `.mat` file for later processing and analysis.

We can also gather large-scale intuition about salp behavior by examining video animations of the salp behavior, which can highlight trends that might be obfuscated in the raw data. In Fig. 3.5 we show a snapshot in time from the video output produced by `SalpPlotter` of the simulation parameters described above.

## 3.4 Results Discussion

Although much of the technical work done to analyze the salp system stability to trajectories and shape deformations will be done by future collaborators, here we present discussion for the preliminary results that we have seen while testing the simulation capabilities.

### 3.4.1 Effect of Thruster Angle on Curvature Development

As illustrated in Fig. 3.4, utilizing an alternating thruster placement along the zooid centers results in a circular motion of the salp through the ambient space. This is true regardless of the choice between the previously modeled shape modes and regardless of the thruster activation profile, so long as the thrusters recieve

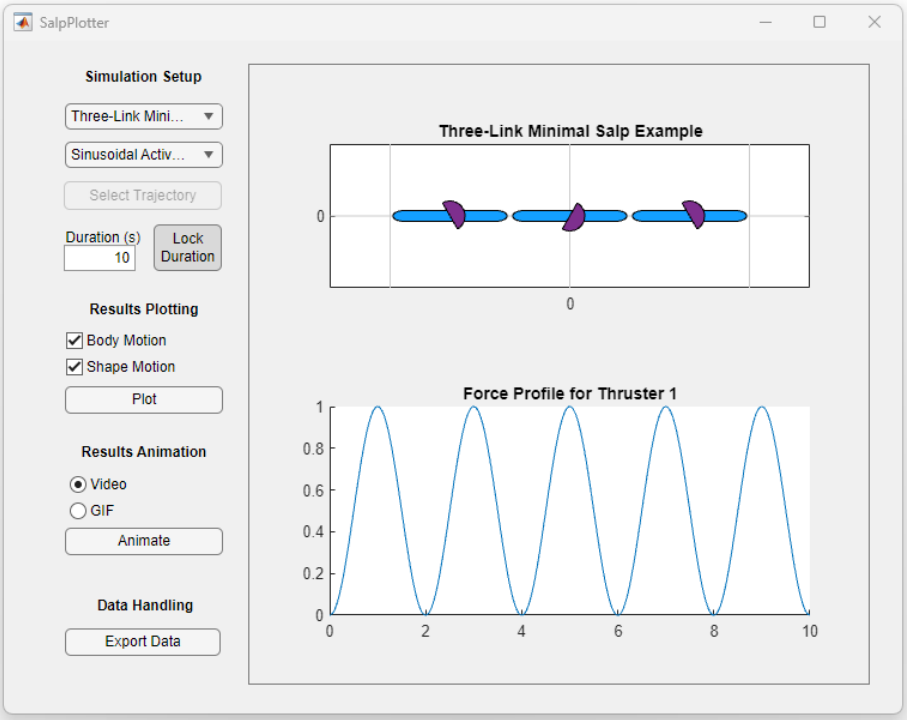


Figure 3.3: Illustration of the system-loading section of SalpPlotter, shown with a loaded 3-Link rigid-zooid salp structure and sinusoidal thruster actuation profile. On the left are dropdown menus to load customizable salp and force definition files, a text input defining the simulation time, checkboxes for different simulation results plotting options and a button that runs the simulation, options for video output of the salp's trajectory through space over time, and an option to export and save the shape and pose evolution of the salp over time.

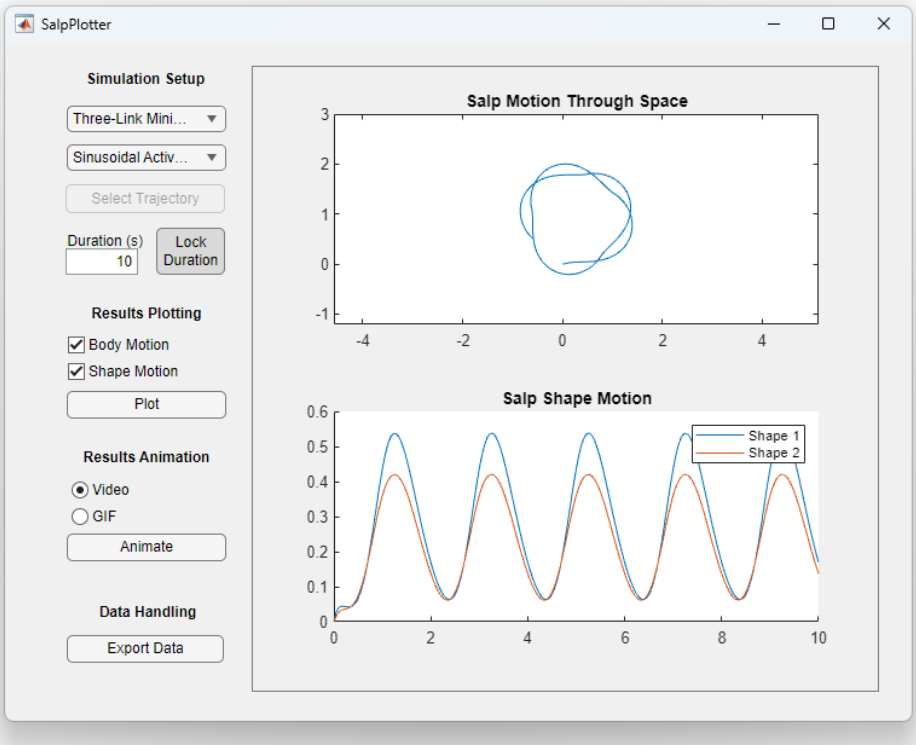


Figure 3.4: Illustration of the static results for the simulation input shown in Fig. 3.3. The 3-Link salp with alternating thruster placement results in a circular trajectory, which is equivalent to the three-dimensional helical salp motion projected onto SE(2). In the top plot, we can see points along the SE(2) evolution of salp motion where the salp executes rapid turning motions connected by straightaways. These turning motions correspond to high amplitudes of thruster firing activity, which cause structure buckling on the passive joints and emergent maneuvering. The periodic buckling of the passive joints can be seen in the lower plot.

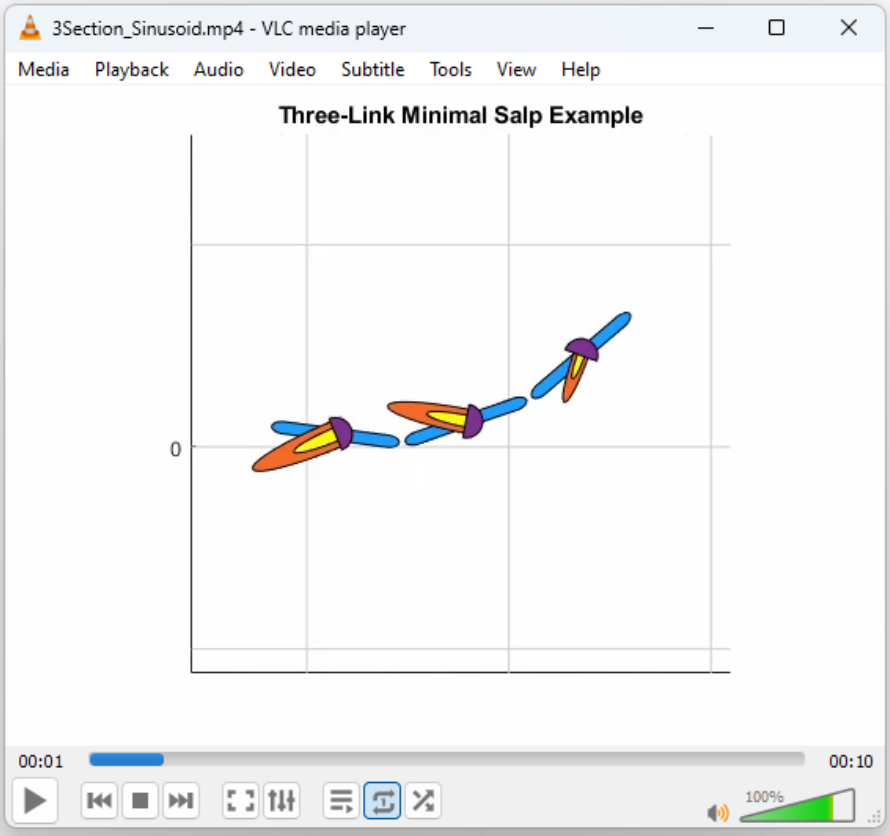


Figure 3.5: Video screenshot of the 3-Link rigid-body salp system discussed in the previous figures executing its emergent trajectory through space.

qualitatively similar actuation signals. The chirality of the circular motion is dependent upon the orientation of the frontmost thruster, with salp chirality roughly flowing along that thrust vector.

It is possible, however, to achieve non-circular net displacement. By altering way the thrusters are positioned along the salp backbone so that they are all aligned to produce the same relative unit thrust vector, we can produce net motion that travels laterally rather than in a circle. Along this lateral trajectory there is a small amount of sinusoidal activity that seems to be a function of the number of salp links and the stiffness of the elastic connections. This lateral motion is likely a different projection of a three-dimensional helical trajectory onto SE(2). These results are similar for the case of the PCC salp.

### 3.4.2 Effect of Number of Salp Segments on Curvature Development

As we increase the number of salp segments while maintaining the same joint stiffnesses, thruster profiles, and drag behavior, more intricate curvature development across the salp spine. Rather than being simple sinusoidal functions, we see shape begin to oscillate as a function of position along the spine. Joint positions along the middle stretch of the spine experience higher-magnitude oscillations than joint positions at the head or tail of the spine. This is likely due to the fact that these joints experience approximately half of the relative constraint forces felt by the middle-segment counterparts. As the front thrusters exert force, they do not need to 'push' against forward segments to exert motion. Similarly, the segments in

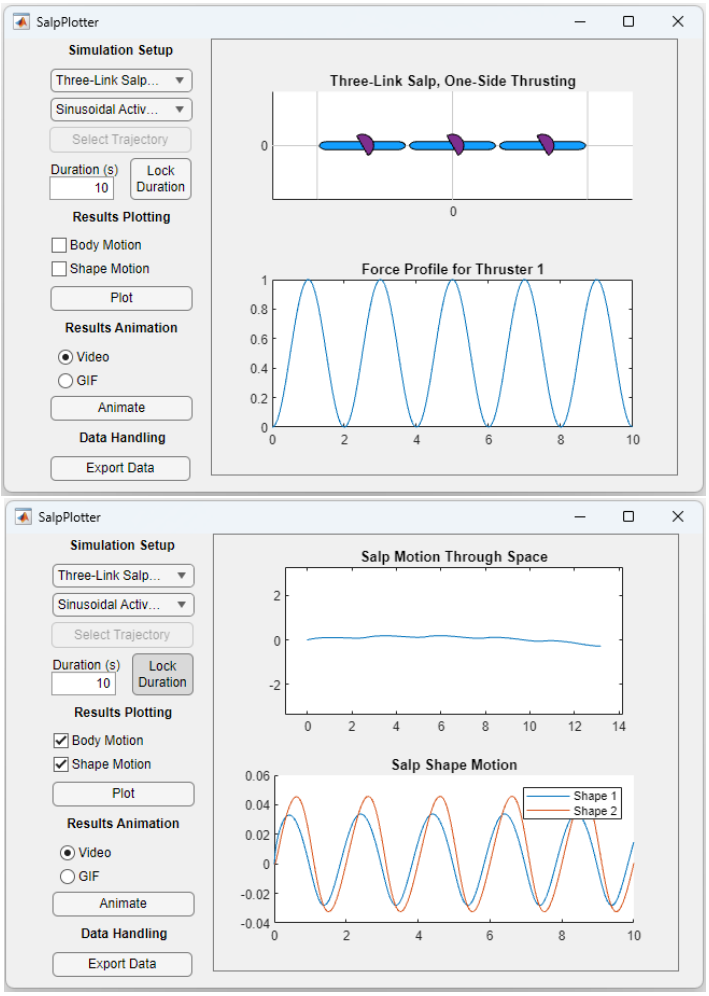


Figure 3.6: Results of salp evolution from a simulation case where each thruster along the body produces the same normal thrust vector. This alignment of thrust produces forward motion rather than circular motion, and is likely a different projection of helical three-dimensional motion onto SE(2).

the back experience no force exerted on them by previous segments. Meanwhile, the salp zooids in the middle both push against forward segments and are pushed by posterior segments, resulting in larger constraint torques and correspondingly larger shape magnitudes.

Qualitatively, this shape evolution produces a buckling wave that travels along the salp backbone and propogates relative to the thruster activation phase. This buckling behavior is captured in Fig. 3.8. As the number of links increases, the circular trajectory of the salp through the ambient space becomes more smooth. This is predominantly because the geometric-center coordinate choice for the salp more and more accurately expresses salp body-frame motion.

Curiously, salps with stochastic thrusting profiles where there is no meaningful thruster activation phase across the backbone of the salp also produce periodic buckling. This is likely because thruster forces occasionally concentrate near the tightly-constrained center links of the salp, producing buckling behavior, which is then 'pulled' through to the back of the salp by whole-body salp motion. An example of this asynchronous buckling is visualized in Fig. 3.9.

### 3.4.3 Effect of Salp Stiffness and Thruster Phase Delay

As noted in the previous paragraphs, longer salps tend to produce buckling waves that travel along the backbone and roughly correspond to regions where the thrusters are firing at full magnitude. The phase delay between adjacent thrusters determines the arclength distance between regions where the thrusters are firing at max-

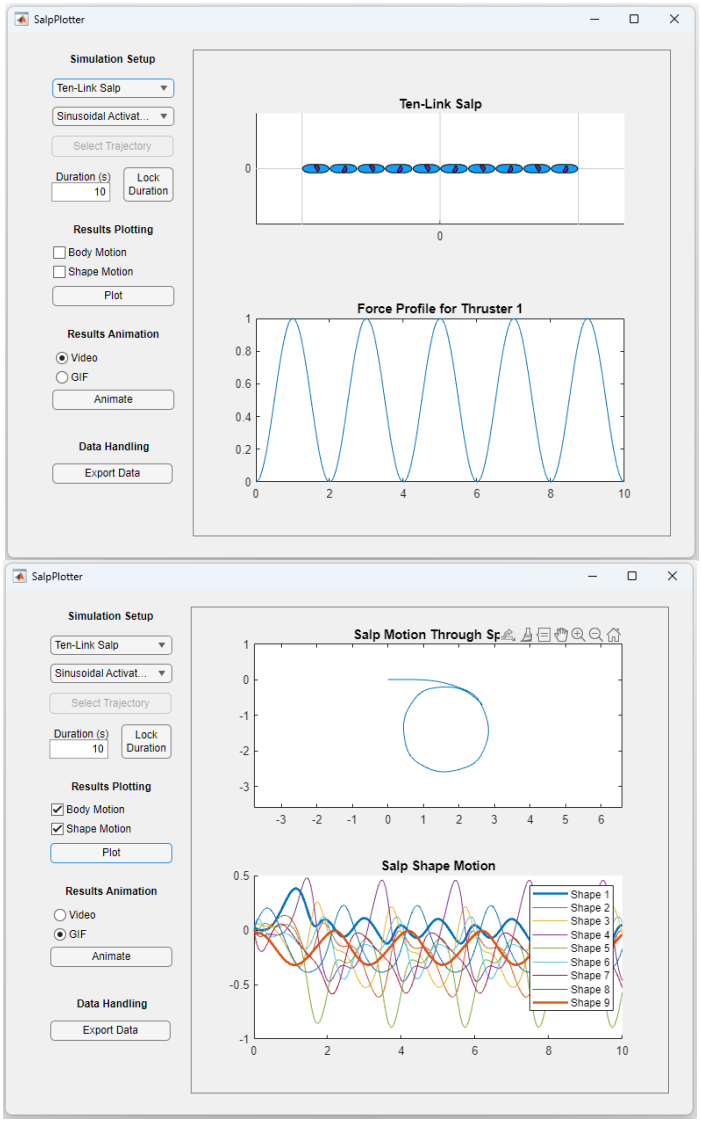


Figure 3.7: Results of salp evolution from a simulation case where thrusters alternate sides with a sinusoidal activation profile for a 10-Link salp. Aside from the number of segments, this simulation is identical to the 3-Link salp simulation across thruster behaviors, inter-link stiffnesses, and salp drag behavior. The total salp length was increased by a factor of 10/3 to maintain the same relative stiffness. Bolded on the lower plot of shape evolutions are the first and last salp joint values, showing that they experience smaller deviation from their average value over the course of the gait.

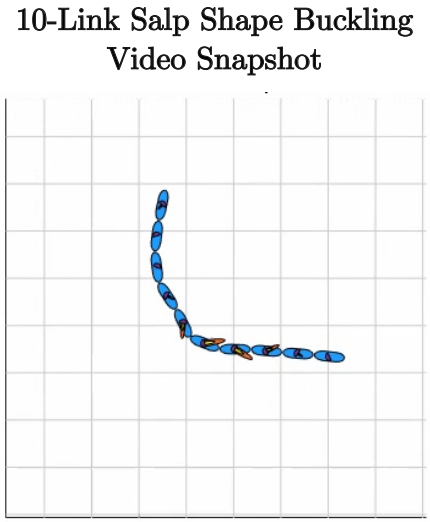


Figure 3.8: Snapshot of the 10-Link salp experiment described above experiencing local buckling of the backbone in response to thruster activation phase.

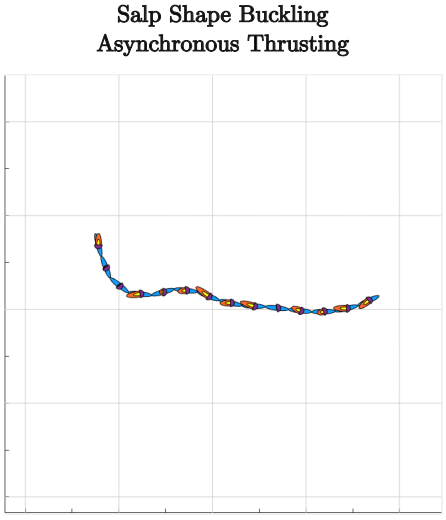


Figure 3.9: Snapshot of a 14-Link salp experiment experiencing periodic backbone buckling despite the lack of a meaningful thruster activation phase.

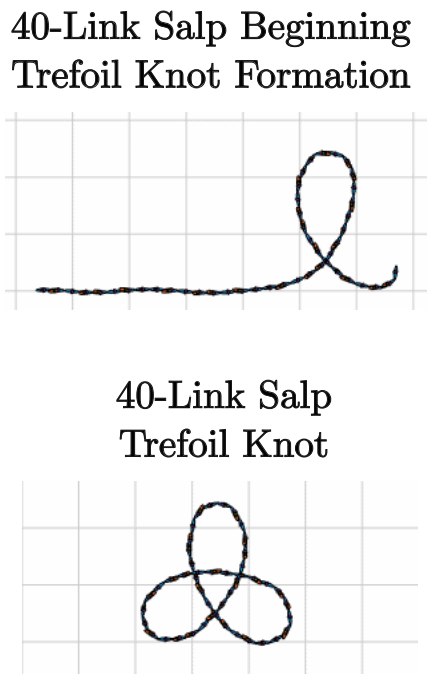


Figure 3.10: Snapshots of a 40-Link salp experiment that has been tuned by modulating thruster phase delay and link stiffness to form a trefoil knot oscillating structure.

imum amplitude. At the buckling point, the local stiffness of the bending joints determines the magnitude of the curvature that develops, with lower stiffnesses corresponding to more aggressive buckling and higher stiffnesses corresponding to lower buckling magnitudes. By tuning the thruster phase delay and the salp backbone stiffness, we can produce interesting emergent geometric behaviors of the salp structure. In Fig. 3.10 we show an example of a longer 40-Link salp that has been tuned via stiffness and thruster phase modulation to produce a stationary trefoil knot structure.

### 3.4.4 Effect of Salp Shape Mode

The primary effect of changing the shape mode of the salp from that of a rigid-body N-Link structure to a continuous curvature mode is that the simulations quickly become much more computationally intensive. While a 3-Link rigid body simulation might be completed in a fraction of a second, the corresponding 3-Section Piecewise-Continuous Curvature simulation might take upwards of 10 minutes to complete. This is predominantly because of the nested integration tasks required in the simulator. Generating the backbone Jacobian along the arclength for the PCC system requires integrating the infinitesimal adjoint backbone actions related to the instantaneous curvature at each point. These Jacobians are then themselves part of an integration problem used to pull back and formulate the drag metric on the salp degrees of freedom. Finally, the results of this nested integration are then used in the time-domain integration of the salp ODE in Eq. (3.22).

There are definitely ways to streamline this problem and reduce computational complexity, perhaps through clever use of integration coupling or drag-metric function fitting across the shape modes, but this will for now remain an area of future work.

Other than this difference in computational expense, the behavior of the N-Link model and the PCC model are very similar. In fact, as the number of segments are increased, it can be argued that the N-Link chain model produces a more accurate representation of organic flexible behavior as the emergent joint dynamics are not subject to the simplifying constant-curvature approximation. Behavior of the 3-

Link PCC model is illustrated in Fig. 3.11. Similar to the 3-Link rigid-body model, the PCC structure generates periodic curvature predominantly near the middle of the salp.

### 3.4.5 Conclusion and Future Work

In this section we introduced potential models to represent an SE(2) salp with passive elasticity that locomotes through the emergent behavior resulting from thruster activation on the salp body. We compared emergent behavior across different modeling parameters such as curvature shape mode, backbone stiffness, thruster phase, number of segments, and actuation strategies.

We found different behaviors that likely represent projections of three-dimensional helical motion onto the SE(2) plane. We also found periodic buckling of the passive shape modes that can by tuning backbone stiffness and thruster activity be used to produce predictable geometric deformations of the salp body.

In future work, we intend to investigate futher how thrusters for a three-dimensional salp structure can be projected onto the two-dimensional plane for these models and used to predict resultant helices motion. We also intend to use these simulation results to drive experimentation for physical instantiations of a robotic salp swimmer.

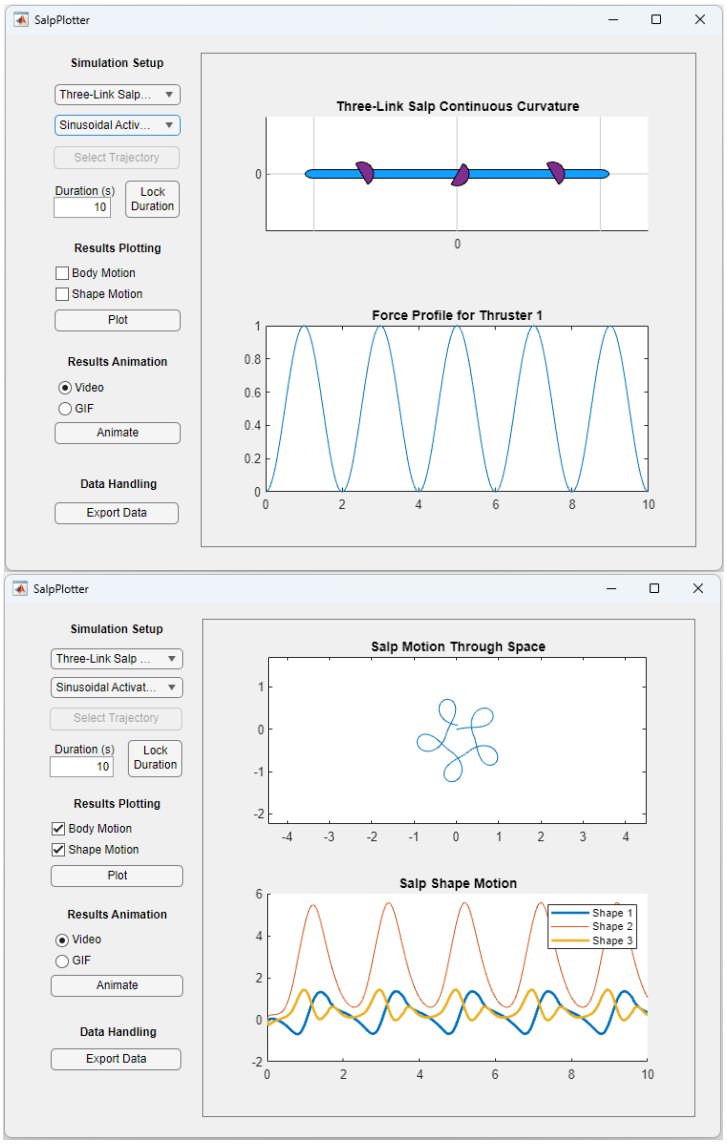


Figure 3.11: Simulation results for the 3-Segment PCC model of a salp locomotor. Similar to the 3-Link rigid-body model, the structure generates periodic curvature predominantly near the middle of the salp. This simulation of 10 seconds of actuation took approximately 20 minutes of computation time.

## 3.5 Thruster Mechanism Design

In the previous section, we developed simulation models to investigate the performance of an SE(2) salp structure consisting of individual zooid segments that produces emergent buckling and locomotion. In this section, we will attempt to produce a thruster capable of periodically supplying thrust comparable to that of a biological salp zooid through actuation of an origami mechanism that is driven by twisted-and-coiled polymer actuators.

Twisted-and-coiled polymer actuators (TCAs) consist of a polymer strand that has been twisted and coiled to form a helical structure that contracts when heated [63]. These coiled strands are often coated in conductive material so that Joule heating can be used to actuate the mechanism. They have previously been used to generate tendon-like actuation for mechanisms like robotic fingers [12] or other soft robots [42], and are well-suited for applying tension on the order of a few Newtons [67].

Here, we seek to combine these capabilities with collapsible origami functionality similar to that which has been explored in previous works on underwater jet propulsion. While previous works have used a tessellated waterbomb crease-pattern as the base of the origami mechanism and actuated the origami using electrical motors [70], here we attempt to design a mechanism that is well-suited to being driven by TCAs by using a sequence of hexagonal Kresling layers [31]. These individually collapsing layers are convenient, because they can be stacked in layered pairs of opposite chirality to produce a spring-like cylindrical mechanism

that collapses with no net rotation.

As biological salps have the capability to produce sinusoidal thrust in the range of 0.6-2 milliNewtons [8], we seek to design a lightweight mechanism capable of similar thrust output. In this section we will discuss the design process and preliminary results of this thruster design. My primary contribution to this project was in designing the origami mechanism and 3D-printed nozzle endcaps, while my collaborator Ali Jones ran the TCA deployment, production, and modeling and coordinated the hydrodynamic experiments on the thruster mechanisms. A prototype of this thruster is pictured in Fig. 3.12.

### 3.5.1 Origami Design

To begin designing the origami mechanism, we first wrote a MatLab script that generates a `.dxf` file used to laser cut the crease patterns, outline edges, and bolt mounts for an arbitrary Kresling stack of $N$ sides, $M$ layers of two antichiral Kresling layers, and side length $L$. Illustrated in Fig. 3.13 is an example of a crease pattern output by this program for $N = 6$ sides and $M = 3$ double-chirality Kresling layers.

After the desired origami geometry is determined and the print file is produced, we cut the file from the desired sheet material using a 50W lasercutter. Choosing the exact lasercutting parameters to cut and etch the sheet was a rather finicky process, consisting of multiple trial runs. Particularly tricky are the mountain and valley fold etch lines, where we need a strong enough cut to cause the material

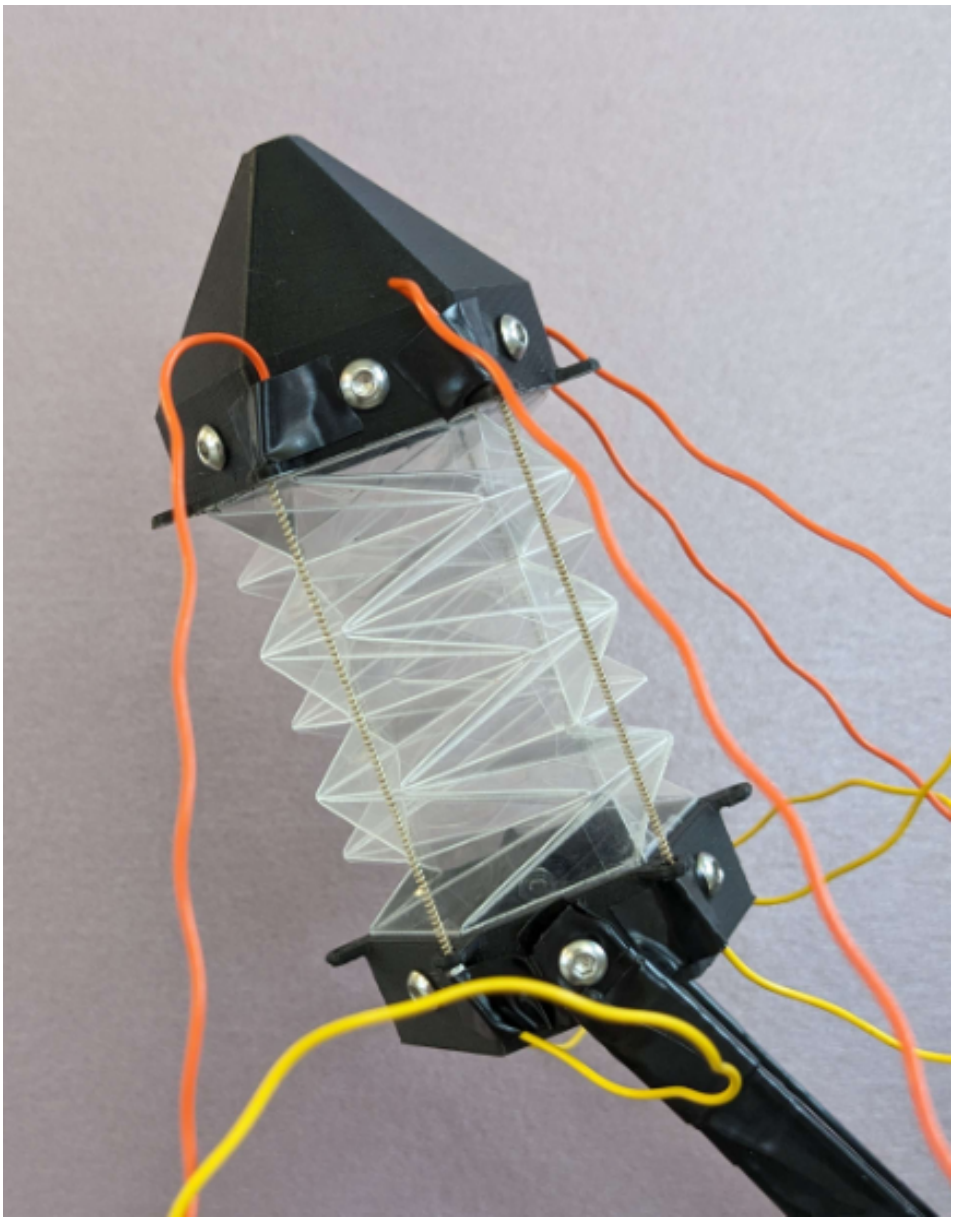

Figure 3.12: Origami thruster made of staked hexagonal Kresling units that can be actuated by TCA activation. This model is attached to a platform that we used to test the water propulsion capabilities. The wires supply an electrical potential difference that is used to drive the TCAs through Joule heating.

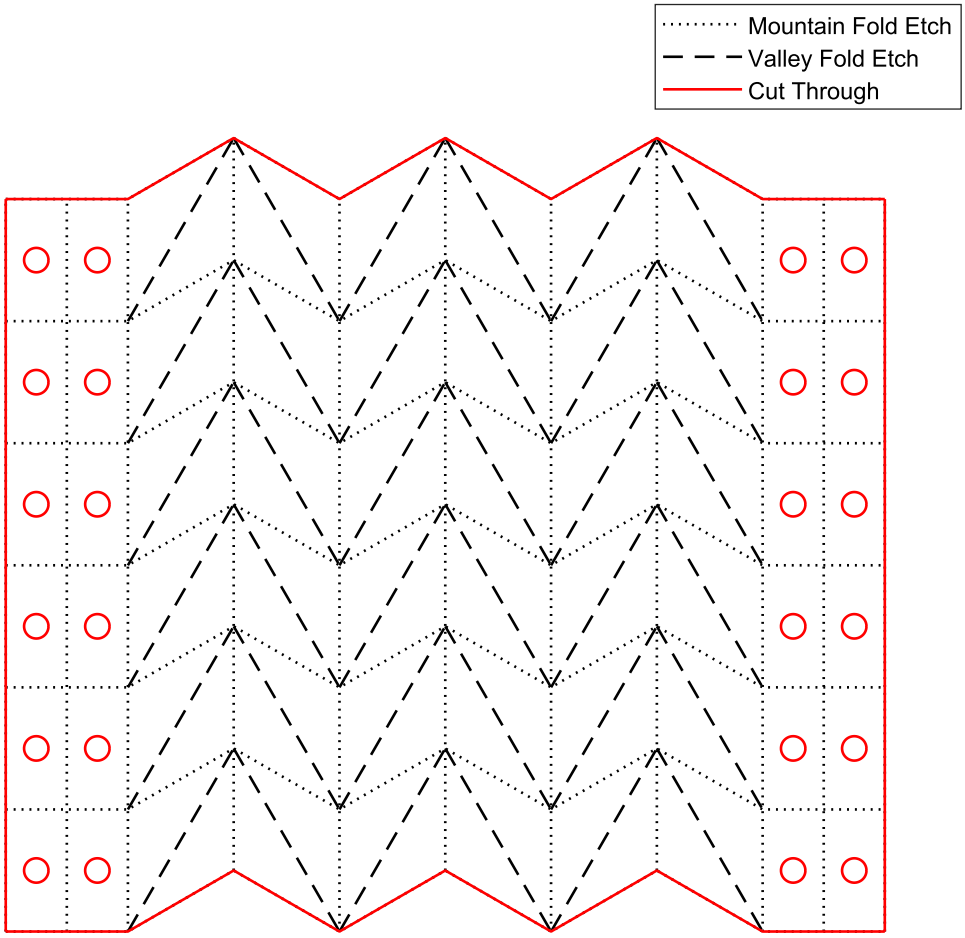


Figure 3.13: Example of an origami precrease pattern generated by the Kresling mechanism MatLab generator for $N = 6$ sides and $M = 3$ antichiral pair Kresling layers. This file can be fed to a lasercutter to programatically etch and cut the template out for rapid manufacturing of the origami mechanism. Mountain and valley folds are etched using identical laser cutter settings but are illustrated differently to highlight the fold geometry.

to naturally collapse into the origami structure but a light enough cut so that the material retains the strength to provide the mechanism functionality. If the etch cut is too light, folding the structure proves tedious, time-consuming, and inaccurate, and if the etch cut is too heavy you risk cutting through the material and breaking the water-tight seal that we need to produce thrust force by providing the water with a method of escape other than the nozzle exit. Careful management of laser speed and power is necessary, as the laser has a minimum power that will produce a usable etched precrease but also a maximum speed that provides consistency of this precrease depth across all cut directions. For the thin material that we ended up using in our actuator design, we operated at the very bottom of this laser's operational range.

After the origami precrease template is cut from the laser cutter, the rectangular strips at the ends are folded inwards to reinforce the mount points to the 3D printed endcaps, the mechanism is rolled into a cylinder along the jagged edge, and commercial 3M Scotch tape is used to seal the open edge and form the cylindrical section. Then, the precrease etches on the origami template are massaged into fold formation and pinched to retain plastic memory of the collapsible deformation. This process takes approximately 15 minutes if the laser cutter settings have been appropriately tuned. An example of a folded and taped origami module is pictured in Fig. 3.14.

After the origami module is prepared, we attach the 3D printed endcaps to each end of the mechanism using M3 bolts. For each end, there is an interior ring that slides into the hexagon to stabilize the hexagonal structure and provide a press-fit

Figure 3.14: Photo of a folded and taped origami collapsible cylinder made from 3 collapsible antichiral Kresling-layer pairs. This is what the origami module looks like before it is bolted to the 3D printed endcaps using M3 bolts.

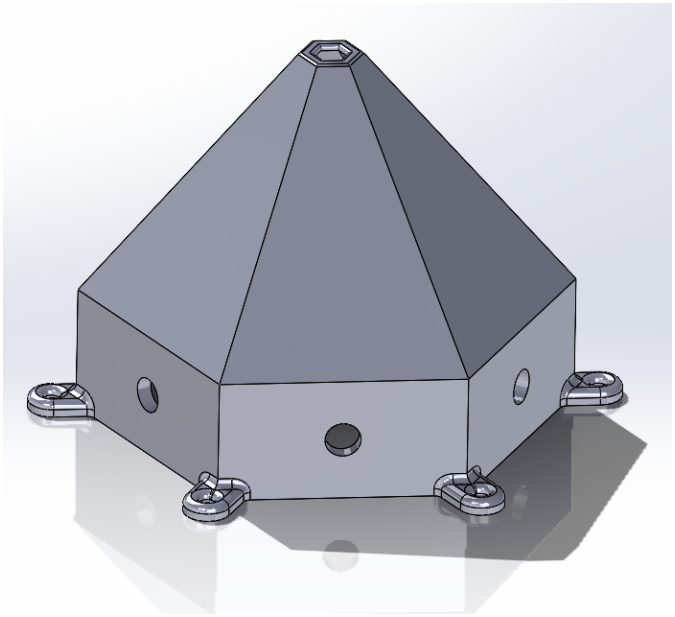

Figure 3.15: CAD mockup of the 3D printed nozzle for the origami mechanism. The interior ring is inserted into one end of the origami structure and the nozzle is slid on over the top, sandwiching the origami material between the two layers. The origami is then fixed in place through the cut hole mounting points by small M3 bolts. The cantilever mounting points are used to secure the TCAs to the corners of the actuator.

method of holding the nuts in place for assembly. Onto one end we attach the nozzle and onto the other end we attach the base plate endcap. These 3D printed components are visualized in Fig. 3.15, Fig. 3.16, and Fig. 3.17.

Originally, we arbitrarily chose $L = 1$" to produce a hexagonal mechanism where each side of the hexagon measured 1 inch across. This origami mechanism measured approximately 6 inches in length after precreasing and assembly. Because of restrictions on the TCA length that Ali could reliably fabricate, we later scaled this down to $L = 7/8$", which made the device short enough to reliably produce

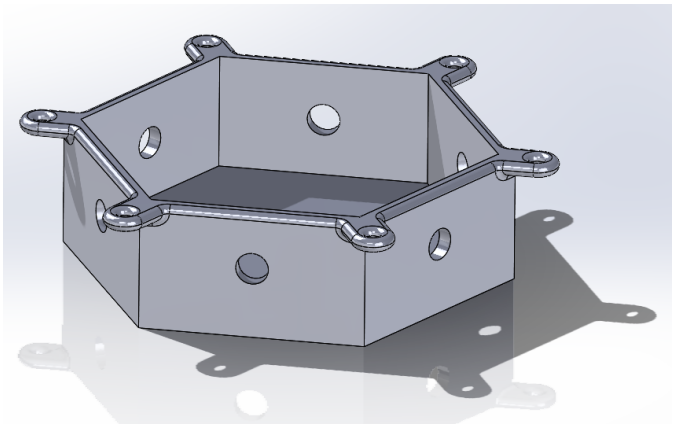

Figure 3.16: CAD mockup of the 3D printed endcap baseplate for the origami mechanism. The interior ring is inserted into one end of the origami structure and the base endcap is slid on over the top, sandwiching the origami material between the two layers. The origami is then fixed in place through the cut hole mounting points by small M3 bolts.

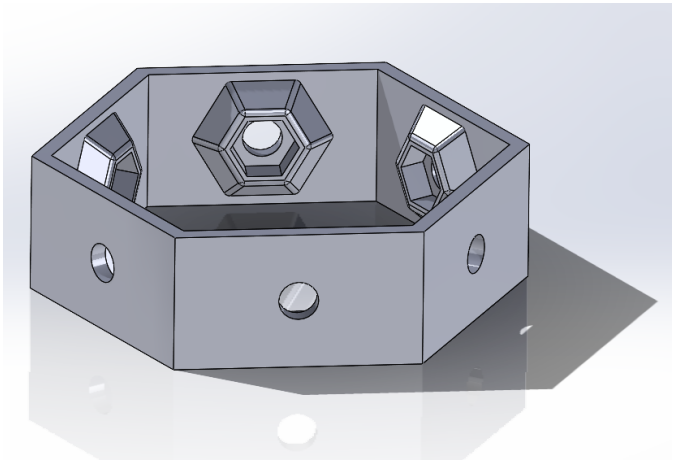

Figure 3.17: CAD mockup of the interior mounting ring used to stabilize the ends of the hexagonal structure of the origami mechanism and hold the press-fit M3 nuts for ease of assembly.

TCAs of the appropriate length.

Choice of material for the origami structure also ended up being an important decision. First, it was important for the material to be waterproof for it to serve its purpose. This narrowed our decision down to stone paper and plastic sheeting, both of which are lightweight and relatively waterproof. After some trial prototypes of each of these materials, we determined that 0.002" PET plastic sheeting resulted in the thinnest achievable origami structure possible using our laser cutter. This material thickness served to be an important parameter in the mechanism design, as we needed the origami mechanism to be very flexible so that it could be realistically actuated using the TCAs.

### 3.5.2 Experimentation

Much of this work will likely end up in Ali Jones' masters thesis, but I will include some preliminary results of their experimentation here to complete the storyline. To take advantage of the fact that TCAs cool and expand much more rapidly than they heat and contract in water, Ali altered the 3D printed endcaps to fit inside a linear railing and attached an elastic spring that served to compress the actuator. This allowed Ali to activate TCAs that expanded the origami mechanism rather than contracting it. This actuator setup is pictured in Fig. 3.18. On the recovery stroke, the TCA's activate to pull and reset the origami mechanism and to refill it with water. On the thrusting stroke, the actuators relax and the passive spring contracts to provide thrust.

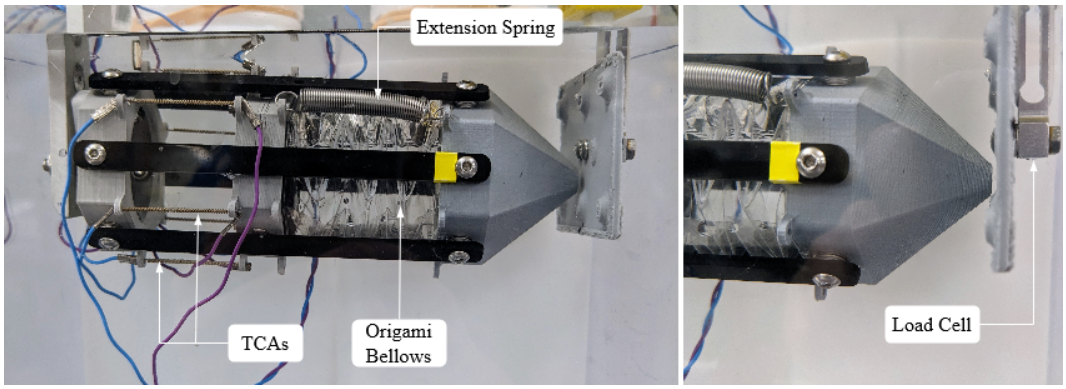


Figure 3.18: Submerged origami mechanism with mounting and actuation modified by Ali Jones. Using the passive spring for the contraction stroke takes advantage of the fact that the TCAs cool down in water much faster than they heat up. This assembly was mounted to a static frame and placed adjacent to a load cell that was used to measure the force of the water exiting the mechanism.

By attaching this frame to a static mounting point and aiming the thruster directly at a load cell with a flat plate, Ali was able to capture an estimate for the thrust values that can be expected from this actuation setup. This thrust data can be seen in Fig. 3.19. From this, we see that the thrusters produce about 0.5 mN of thrust on the thrust stroke, which is slightly below what a biological salp produces.

## 3.6 Conclusion

In this chapter, we discussed modeling and experimentation for analysis of the locomotion of the sea salp. First, we proposed a mathematical model for low Reynolds number drag-dominated sea salp comprised of a chain of individual zooids, each capable of producing thrust through fluid expulsion and possessing passive flexibil-

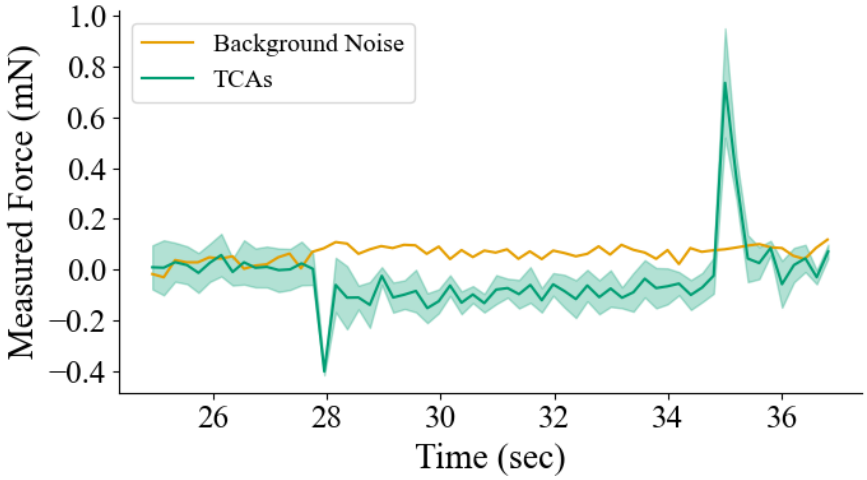


Figure 3.19: Preliminary results for the thrust from this actuator captured by Ali Jones' experimentation. The the peaks in the green TCA data are estimates of the actuator thrust as the origami mechanism is excited by TCA action. From this data, we see that the mechanism produces about 0.8 mN of thrust on the thrust stroke. This is at the lower end of what a biological salp is capable of. We expect this number to improve though experimentation revisions, salp mount redesigns, nozzle tuning, and changes to the endcap that add a passive checkvalve and allow the chamber to refill from the bottom.

ity. We then built and described a simulator that we can use to examine emergent locomotive and buckling behaviors of various salp models and used the results of these simulations to form some preliminary observations about the effects of salp design factors on emergent behavior.

Following this, we described the mechanical design of a soft origami mechanism that we hope to use as a thruster for future robotic salp experimentation. We show how this origami mechanism is fabricated and assembled with 3D components, and discuss some preliminary experimentation data found by Ali Jones, where we see that the thruster is capable of producing thrust comparable to a biological salp zooid.

Future work for the locomotive model includes the investigation of how these SE(2) salp model systems can be used in the analysis of their three-dimensional SE(3) counterparts. We intend to use these simulation results and modeling tools to drive gait design in future experiments of a robotic salp swimmer. Future work for the origami thruster involves improving the thrust performance on both the thrust and recovery stroke and integration into a full robotic salp analog.

# Chapter 4: System Identification and Gait Library Construction for the AmoeBot

## 4.1 Introduction

Bioinspired locomotion is an active area of research in the robotics community. Robots that move through the world using natural body motions rather than typical robotic wheels or propellers are generally more capable of traversing a wider variety of environmental terrain, often have higher locomotive efficiencies than their wheeled counterparts, and help provide insights into the biological mechanisms of the natural world [16, 32].

Bioinspired locomotors make use of a complex relationship between body shape changes and resulting body motion to locomote through the world. This complexity necessitates a large number of laboratory experiments throughout the process of locomotor design to test variations in things like gait parameters, body geometry, and passive component selection.

It is often difficult to perform these tests in a laboratory setting, especially for swimming systems that locomote through water. It is possible to perform these free-swimming tests in the lab [69], but large aquatic testbeds in which robot swimmers can freely move are expensive and space-filling. To record the results of free-swimming, it is typically necessary to perform a field deployment of the

aquatic robotic system in a large body of water, which is time-consuming and inconvenient.

Because of the difficulty inherent in deploying freely swimming robotic locomotors for laboratory tests, it is common to perform minimal experiments where the robotic system is held constrained in a relatively small water tank while connected to a force sensor. The robot can then execute shape changes while the force sensor records the constraint forces that hold the system in place. The nature of these constraints is highly variable. Some experiments hold the robot completely stationary in water so that there is no allowed body displacement, while some experiments allow for displacement in a select few degrees of freedom, and yet others place the robot in a flow tank so that the locomotor can have a controlled velocity of one degree of freedom with respect to the fluid.

Figure 4.1 shows examples of such experiments.

- In Fig. 4.1(a), robot 'frog' legs are attached to a force sensor that reads constraint forces and holds the legs in place while they enact a 'kicking' swimming gait [65].

- In Fig. 4.1(b), a robot 'breaststroke' swimmer is constrained to a rail so that it can move only in one direction while it executes shape changes. A force sensor records forward and lateral thrusts during the experiment [55].

- In Fig. 4.1(c), a robot 'fish' with a passive tail is held stationary in a flow tank through an attachment to a force sensor that records constraint forces. The robot executes shape changes while the flow tank is running, such that

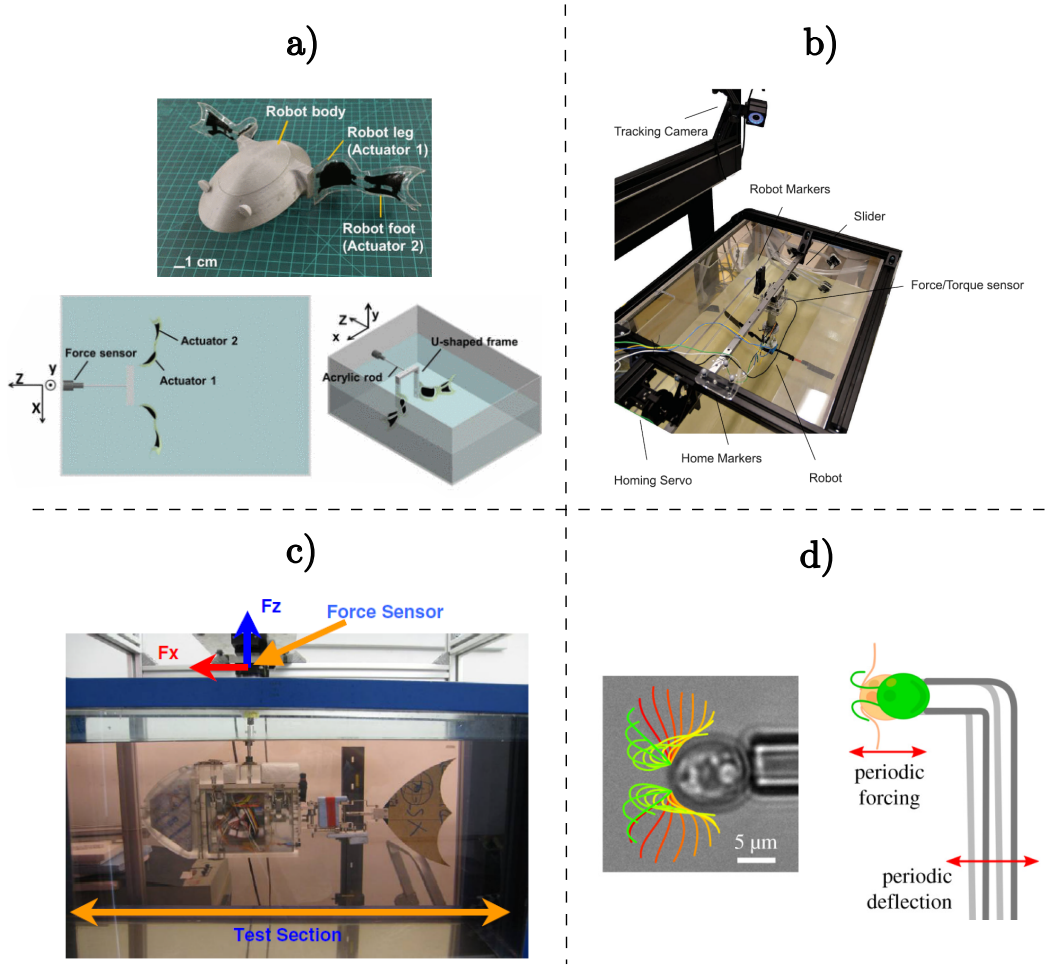


Figure 4.1: Examples of constrained locomotion experiments. **a)** A 'frog' swimmer. For experiments, the frog legs are attached to a stationary force sensor [65]. **b)** A 'breaststroke' swimmer. For experiments, the robot is tethered to a force-sensing rail system that allows only forward displacement [55]. **c)** A 'fish' swimmer. For experiments, the robotic system is executes shape changes in a running flow tank while attached to a force sensor [35]. **d)** An algae cell *Chlamydomonas reinhardtii*. For experiments, the cell is constrained at the tip of a micropipette force sensor [7].

the system has a constant forward velocity with respect to the fluid [35].

- In Fig. 4.1(d), a multiflagellated algae cell is partially aspirated using hydrostatic pressure differences while constrained by a micropipette force sensor. As the cell is aspirated, the two flagella actuate, causing a forward thrust that deflects the tip of the micropipette. The deflections of the pipette are measured using optical flow and cantilever beam theory is used to estimate the thrust profile from the flagella [7].

While such experiments are useful for testing forward thrust, they are currently not typically used to develop models that relate shape changes to body motion. This is because constraining a swimming robot changes the fluid profile that the locomotor would see from the corresponding fluid profile of a free-swimming system. For example, in the robot frog research pictured in Fig. 4.1(a), the authors do not include the frog's 'body' in the force observation experiment. Because of the force-sensor constraint, there is no motion of the legs that would cause the body to experience motion through the fluid, so the body would have no impact on the thrusting force if included in this experiment. However, for the unconstrained system, the robot body is subject to fluid drag and an extra hydrodynamic mass that affect body motion. Because the experiment provides no measure of body drag or body mass, it is impossible to use the thrust data alone to predict how the swimmer will move through the fluid. But the lack of a complete hydrodynamic profile for the robot does not make this a poor experiment! On the contrary, it is easy to intuitively why this experiment was performed, as we understand on a

gut level that such motion response data will capture some information about the behavior of this system in water. The experimental problem for model formation then becomes identifying the dynamic parameters that drive motion and force for the aquatic system from the context of the constrained experiment.

Generally speaking, performing a thrust-force measurement imposes a constraint on the body of a locomotor in the fluid. These constraints fundamentally change the way hydrodynamic elements interact with the fluid over the course of an arbitrary shape change. This means that thrust profiles will be different than when the locomotor is unconstrained. Because of these complications, attempts to perform model prediction for biomimetic robots based on thrust-force data are rarely successful.

In this chapter, we will derive a framework that relates the body geometry and hydrodynamic properties of an arbitrary locomotor to generalized body constraint forces. We will demonstrate how this framework can be used to perform system identification by using constraint force data to predict hydrodynamic properties such as mass and drag coefficients for all elements in the locomotor. We will discuss how to tell if desired hydrodynamic properties are observable given a certain experiment.

After we formulate this framework for system identification from constrained experimentation, we will apply the methodology to extract dynamic information from constrained experimentation performed on the AmoeBot, a swimming robotic platform developed by research collaborators Nick Gravish and Curtis Sparks at the University of California San Diego [57]. The AmoeBot moves atop the surface

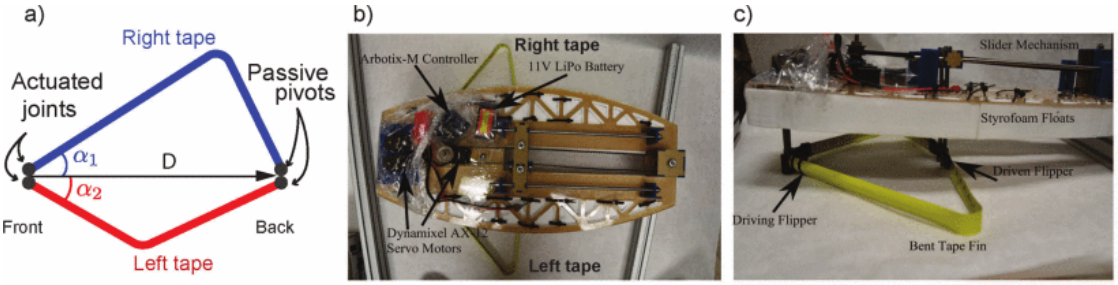


Figure 4.2: The AmoeBot locomotor on which we validate the geometric techniques discussed in this work. Two semi-independent thrust-element tape appendages are actuated by motors that set the angle of the tape with respect to the front of the robot, and by a linear actuator that sets the distance between the front and back connection points. Using force sensing, we hope to identify the linear mass and drag densities of the tape-spring paddles. We then hope to use these hydrodynamic properties to produce accurate predictions of robot motion in unconstrained scenarios. Figure borrowed from [57].

of water through the motion of two tape-spring appendages that act as propulsive fins. We perform system identification of the tape-spring appendage hydrodynamic properties using constrained force sensing, and use these hydrodynamic properties to develop a locomotive model of the swimming system. We then use the model to optimize the actuator shape and pacing of swimming gaits and develop a library of navigational maneuvers for the system. Once this library is in place, we use a joystick controller to select gaits from the library alongside a gait attractor control field that enables seamless transitions between gait executions. This controller is used to manually pilot the AmoeBot across two navigational experiments, both of which took place in an aquatic test tank. In the first, the AmoeBot navigates to a series of waypoints, matching the pose and orientation at each. In the second, the AmoeBot follows a figure-eight trajectory around obstacles inside the tank.

## 4.2 Mathematical Formulation

In this section, we will derive the equations of motion for a locomoting system, and then show how this formulation can be used to perform system identification for systems that are subject to experimental constraints. We will also discuss how the connection between system geometry and hydrodynamic constraints can be used to determine from an experiment description if the experiment is sufficient to develop a full system model.

### 4.2.1 Derivation of Momentum-Aware Body-Frame Dynamics

Here, we will derive the full momentum-aware equation of motion for body-frame dynamics using Lagrangian analysis.

We will make use of two ways to express locomotor velocity. The first is $\dot{g}$, the velocity of the locomotor's body with respect to the inertial world frame. The second is $\overset{\circ}{g}$, the velocity of the locomotor's body with respect to the instantaneously stationary body frame. These velocities are typically expressed as column vectors of individual velocity components. For locomotors operating in the planar space of rotations and translations $SE(2)$,

$$\dot{g} = \begin{bmatrix} \dot{x}_w \\ \dot{y}_w \\ \dot{\theta} \end{bmatrix}, \text{and } \overset{\circ}{g} = \begin{bmatrix} \dot{x}_b \\ \dot{y}_b \\ \dot{\theta} \end{bmatrix}. \tag{4.1}$$

The body velocity can be mapped to the world velocity using a left action $g$

that rotates the body frame to be aligned with the world frame,

$$\dot{g} = g\mathring{g}. \tag{4.2}$$

The action $g$ is a simple rotation constructed from the locomotor's body orientation $\theta$ with respect to the world frame. In $SE(2)$, $g$ can be expressed as,

$$g = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.3}$$

The inverse of this rotation, $g^{-1}$, moves a world frame velocity to a body frame velocity. Because $g$ is an orthonormal rotation matrix,

$$g^{-1} = g^T. \tag{4.4}$$

As discussed in §2.2.1, we can form a locomotor mass matrix $M$ by pulling back individual body mass element matrices $\mu_i$ using the Jacobians $J_i$ linking body and shape velocity to individual element velocities in the world frame,

$$M(r) = \sum_i J_i^T(r)\mu_i J_i(r) \tag{4.5}$$

where $M$ and $J_i$ are both functions of the locomotor's shape $r$. The individual mass elements composing the body can be expressed in $SE(2)$,

$$\mu_i = \begin{bmatrix} m_{x,i} & 0 & 0 \\ 0 & m_{y,i} & 0 \\ 0 & 0 & m_{\theta,i} \end{bmatrix}. \tag{4.6}$$

Because rotational inertia term is a result of the displacement of mass from a mass element's frame origin, we can neglect $m_\theta$ if the mass elements are sufficiently small. This disappearance of angular rotation terms for the individual metrics on small elements is similar to the treatment that we gave to the Piecewise-Constant Curvature salp model in the previous chapter in Eq. (3.16), and similar simplifications have been done before for hydrodynamic systems [69],

$$m_\theta = 0. \tag{4.7}$$

In these cases where there are many small mass elements as opposed to a few large ones, body rotational inertia will arise from the pullback of the many mass elements into the central body frame in Eq. (4.5). For cases where we are instead composing the body of a few larger mass elements, we can estimate $m_\theta$ using standard calculations of rotational inertia from mass density and body geometry as in §2.2.1.

To find the full dynamics of our locomotor in terms of our generalized degrees of freedom $q$ and generalized forces $f$ acting on those degrees of freedom, we will use the Euler-Lagrange equation,

$$f = \frac{\delta}{\delta t}\frac{\delta L}{\delta \dot{q}} - \frac{\delta L}{\delta q}, \tag{4.8}$$

where the Lagrangian $L$ is the difference between the kinetic energy and the potential energy

$$L = KE - PE. \tag{4.9}$$

We can consider passive elastic elements by adding a stiffness potential energy on the shape variables $r$ or, rarely, the locomotor displacement $\tilde{g}$. For linear springs, it is convenient to use a stiffness matrix $K$ to define the elastic connections,

$$PE = \frac{1}{2}\begin{bmatrix}\tilde{g}^T & r^T\end{bmatrix} K \begin{bmatrix}\tilde{g} \\ r\end{bmatrix}, \tag{4.10}$$

where $\tilde{g}$ is a vector of locomotor translations and rotations in the world frame. Because we deal with SE(2) systems in this work that are modeled as being in a horizontal plane, we can neglect the potential energy contribution due to gravity, as all inertial elements remain in the same gravitational plane. For three-dimensional SE(3) systems or for SE(2) cases where there remains some gravitational influence, we must model the potential energy due to gravity.

Here we can also add a Rayleigh dissipation function to consider velocity-proportional friction effects. However, because we will be considering both linear and quadratic drag terms, this function is rather complex. It is easier to add generalized drag forcing terms after Lagrangian analysis is performed, because there

are established relationships describing how linear and quadratic drags act on generalized shape modes [69]. Rayleigh dissipation functions are typically constructed purposefully to recreate specific dissipation relationships, so there is little lost by adding the drag-force relationship models after dealing with the Lagrangian energy mechanics.

We can use the mass matrix $M$ to map body velocity $\mathring{g}$ and shape velocities $\dot{r}$ to total locomotor kinetic energy as in Eq. (2.2),

$$KE = \frac{1}{2} \begin{bmatrix} \mathring{g}^T & \dot{r}^T \end{bmatrix} M \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}. \tag{4.11}$$

We now have enough information to evaluate the Euler-Lagrange equation. We will perform the Euler-Lagrange calculation in the inertial world frame, so we will end up with equations of motion in terms of generalized world-frame forces acting on the locomotor $f_w$ and generalized shape mode forces $\tau$. We can rotate the generalized world-frame forces to body-frame forces using our previously established rotation map $g$,

$$\begin{bmatrix} f_b \\ \tau \end{bmatrix} = \begin{bmatrix} g^T & 0 \\ 0 & \mathrm{Id} \end{bmatrix} \begin{bmatrix} f_w \\ \tau \end{bmatrix}. \tag{4.12}$$

We will be adding this $g^T$ rotation term as we complete the Euler-Lagrange calculations to convert the equations of motion back into the body frame.

To perform the derivatives in Eq. (4.8), we move the kinetic energy equation to an inertial frame using Eq. (4.2),

$$KE = \frac{1}{2}\begin{bmatrix}\dot{g}^T g & \dot{r}^T\end{bmatrix} M \begin{bmatrix} g^{-1}\dot{g} \\ \dot{r}\end{bmatrix}. \tag{4.13}$$

We first perform the derivative with respect to the generalized shape, differentiating with respect to the body displacements and with respect to each of the $n$ shape modes,

$$-\frac{\delta L}{\delta q} = -\frac{\delta KE}{\delta q} + \frac{\delta PE}{\delta q} = -\begin{bmatrix} 0 \\ 0 \\ \begin{bmatrix}\dot{g}^T \frac{\delta g}{\delta\theta} & 0\end{bmatrix} M \begin{bmatrix}\mathring{g} \\ \dot{r}\end{bmatrix} \\ \frac{1}{2}\begin{bmatrix}\mathring{g}^T \dot{r}^T\end{bmatrix} \frac{\delta M}{\delta r_1} \begin{bmatrix}\mathring{g} \\ \dot{r}\end{bmatrix} \\ \vdots \\ \frac{1}{2}\begin{bmatrix}\mathring{g}^T & \dot{r}^T\end{bmatrix} \frac{\delta M}{\delta r_n} \begin{bmatrix}\mathring{g} \\ \dot{r}\end{bmatrix} \end{bmatrix} + \begin{bmatrix} g^T & 0 \\ 0 & \mathrm{Id}\end{bmatrix} K \begin{bmatrix}\tilde{g} \\ r\end{bmatrix}, \tag{4.14}$$

where $\frac{\delta g}{\delta\theta}$ is readily calculated from the definition of $g$,

$$\frac{\delta g}{\delta\theta} = \begin{bmatrix} -\sin\theta & -\cos\theta & 0 \\ \cos\theta & -\sin\theta & 0 \\ 0 & 0 & 0\end{bmatrix}. \tag{4.15}$$

The term that comes from the kinetic energy derivative is the first of the Coriolis

forces. For $SE(2)$ locomotors, because the top rows of these Coriolis forces are zero, we do not need to apply the rotation matrix to convert them to the body frame, so they are found as,

$$C_1 = -\begin{bmatrix} 0 \\ 0 \\ \begin{bmatrix} \dot{g}^T \frac{\delta g}{\delta \theta} & 0 \end{bmatrix} M \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} \\ \frac{1}{2}\begin{bmatrix} \mathring{g}^T \dot{r}^T \end{bmatrix} \frac{\delta M}{\delta r_1} \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} \\ \vdots \\ \frac{1}{2}\begin{bmatrix} \mathring{g}^T & \dot{r}^T \end{bmatrix} \frac{\delta M}{\delta r_n} \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} \end{bmatrix}. \tag{4.16}$$

For $SE(3)$ locomotors there are complexities in the roll-pitch-yaw rotations that do make it necessary to factor in the $g^T$ rotation from Eq. (4.12), but in this work we are primarily concerned with $SE(2)$ systems so we will not account for three-dimensional rotations.

The term in Eq. (4.14) that comes from the potential energy derivative is the force contribution due to passive stiffnesses in the system,

$$F_k = \begin{bmatrix} g^T & 0 \\ 0 & \mathrm{Id} \end{bmatrix} K \begin{bmatrix} \tilde{g} \\ r \end{bmatrix}. \tag{4.17}$$

This can generalize to nonlinear elastic elements such as asymmetric springs

by constructing stiffness $K$ as a function of world position and shape rather than as a linear matrix, and then moving results to the body frame:

$$F_k = \begin{bmatrix} g^T & 0 \\ 0 & \mathrm{Id} \end{bmatrix} K\left(\tilde{g}, r\right). \tag{4.18}$$

Continuing with the Euler-Lagrange operations, we perform the derivative with respect to generalized speeds. Because the potential energy terms do not depend on velocity, we are left with only results from the kinetic energy,

$$\frac{\delta}{\delta t}\frac{\delta L}{\delta \dot{q}} = \frac{\delta}{\delta t}\left(\begin{bmatrix} g & 0 \\ 0 & \mathrm{Id} \end{bmatrix} M \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}\right). \tag{4.19}$$

Then we perform the time derivative, factoring in the force rotation from Eq. (4.12),

$$\begin{aligned} \begin{bmatrix} g^T & 0 \\ 0 & \mathrm{Id} \end{bmatrix} \frac{\delta}{\delta t}\left(\begin{bmatrix} g & 0 \\ 0 & \mathrm{Id} \end{bmatrix} M \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}\right) = & \begin{bmatrix} g^T \frac{\delta g}{\delta \theta}\dot{\theta} & 0 \\ 0 & 0 \end{bmatrix} M \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} \\ & + \left(\sum_{i=1}^{n} \frac{\delta M}{\delta r_i}\dot{r}_i\right) \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} \\ & + M\frac{\delta}{\delta t}\left(\begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}\right). \end{aligned} \tag{4.20}$$

The first of these three terms is an adjoint term reflecting how momentum

changes due to body-frame rotations [30],

$$Ad^* M \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} g^T \frac{\delta g}{\delta \theta} \dot{\theta} & 0 \\ 0 & 0 \end{bmatrix} M \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}. \tag{4.21}$$

The second term is the second contribution to the Coriolis forces,

$$C_2 = \left( \sum_{i=1}^{n} \frac{\delta M}{\delta r_i} \dot{r}_i \right) \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}. \tag{4.22}$$

Finally, the third term is the body-frame equivalent of mass times acceleration,

$$F_a = M \frac{\delta}{\delta t} \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}. \tag{4.23}$$

Note that for the second two terms, the two force-rotation matrices are inverses of each other and cancel,

$$\begin{bmatrix} g^T & 0 \\ 0 & \mathrm{Id} \end{bmatrix} \begin{bmatrix} g & 0 \\ 0 & \mathrm{Id} \end{bmatrix} = \mathrm{Id}. \tag{4.24}$$

This completes the force contributions due to inertial effects. We can also add linear drag forces by constructing a drag matrix similar to Eq. (4.5) using body-Jacobian pullbacks and a matrix of drag coefficients $d_L$ to get drag forces that linearly scale with velocity,

$$D_L = \sum_i J_i^T d_{L,i} J_i \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}. \tag{4.25}$$

Similarly, we can use the body Jacobians to pull back drag forces that scale quadratically with velocity rather than linearly using a matrix of quadratic drag coefficients $d_Q$,

$$D_Q = \sum_i J_i^T d_{Q,i} \left( J_i \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} \odot J_i \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} \right), \tag{4.26}$$

where the $\odot$ operator refers to the element-wise Hadamard product.

We can now assemble the complete equations of motion for a freely locomoting inertial system that is subject to both linearly proportionate and quadratically proportionate drag forces:

$$\begin{bmatrix} f_b \\ \tau \end{bmatrix} = Ad^* M \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix} + F_K + C_1 + C_2 + D_L + D_Q + M \frac{\delta}{\delta t} \begin{bmatrix} \mathring{g} \\ \dot{r} \end{bmatrix}. \tag{4.27}$$

These equations are useful for both freely locomoting systems and for constrained systems. For freely locomoting systems, there are no generalized forces acting on the body of the robot, so $f_b = 0$. Shape forces $\tau$ are generalized actuator forces for the actively controlled shapes, and are zero for passive shapes because elastic forces are accounted for in the $F_K$ term. The equations can then be solved for the body-velocity rate of change and implemented through an ODE solver such

as `ode45` or Euler's method to study the evolution of body position and passive shape modes over time given input shape commands.

$$\frac{\delta}{\delta t}\begin{bmatrix}\overset{\circ}{g}\\ \dot{r}\end{bmatrix} = M^{-1}\left(\begin{bmatrix}f_b\\ \tau\end{bmatrix} - Ad^{*}M\begin{bmatrix}\overset{\circ}{g}\\ \dot{r}\end{bmatrix} - F_K - C_1 - C_2 - D_L - D_Q\right). \tag{4.28}$$

For constrained systems, body constraints are enforced by constraining the body acceleration term. For example, a swimming locomotor facing upstream in a flow tank and performing shape changes while rigidly attached to load cells would have a constant forward velocity relative to the water, zero side and rotational velocities, and zero body accelerations. The generalized body forces $f_b$ are then the reaction forces experienced through the load cells that enforce these constraints on the swimmer.

Thinking of load cell readings as a generalized body force $f_b$ is equivalent in nature as modeling the load cell as a very stiff spring with a very precisely known spring constant and including the terms in $F_K$. However, because load cells typically come with internal circuitry that performs the spring model to force conversion for you and because we can often neglect the small motions of a system relative to the base of the force cell, it is generally simpler to model load cell readings as generalized body forces. For cases where the force cell is less stiff or the system to be modeled is more massive and relative motion of the distal and proximal regions of the load cell cannot be neglected, the spring constant model may be used instead.

We will use this framework of constraining accelerations and observing load cell reaction forces in the next section to perform system identification on hypothetical locomotors in constrained experiments.

### 4.2.2 System Identification

Here we will lay out a framework for the geometric system identification of locomoting systems using the dynamic equations of motion derived in the previous section. The end goal of these formulae is to identify mass and drag coefficient values for the locomotor that best fit observations from experimental results. These coefficients can then be used to perform various useful analyses on the locomotor, such as gait optimization, gait analysis for path planning purposes, or locomotor body geometry optimization. This approach will be very similar in nature to previous works that have described regression of dynamic parameters [37, 56], but with additional focus on identifying indirect parameters such as element density rather than mass and with some additional discussion on the role of constraint frames.

When breaking a locomotor model into large discrete subcomponents, we need six coefficients for each element. First, we need the three mass values that make up the element's principal inertia matrix $\mu$: the translational masses $m_x$ and $m_y$, and the rotational inertia $m_\theta$. Then, there are three additional components that represent the drag model. In the case of linear drag we search to identify the translational drag coefficients $d_{L,x}$ and $d_{L,y}$, and the rotational drag $d_{L,\theta}$. As an alternative (or in addition at some risk of overfitting) to linear drag coefficients,

we can use the quadratic drag coefficients $d_{Q,x}$, $d_{Q,y}$, and $d_{Q,\theta}$.

When breaking up components into elements that are sufficiently small, we only need six coefficients for each element, as the rotational coefficients arise from pulling back the translational coefficients into the locomotor frame. For this case, the coefficients can be formulated as mass and drag densities rather than discrete values. Considering coefficients as mass and drag densities can reduce the number of coefficients needed to fully define the system, preventing overfitting during the extraction operation. We use this strategy in our AmoeBod system identification in Section 4.3 to break up long swimmer components into a large number of infinitesimal hydrodynamic components and use six coefficients (mass, linear drag, and quadratic drag densities for the lateral and longitudinal directions) to define the inertial system.

We start by noting that generalized reaction forces that would be observed by a constrained robot are linear on all of the coefficients. We will first demonstrate this for the mass coefficients. We constructed the total mass matrix $M$ in Eq. (4.5) by pulling back the individual mass matrices into the locomotor frame. Expressed in terms of the system coefficients, this reads as

$$M(r) = \sum_i \left( J_i^T(r) \begin{bmatrix} m_{x,i} & 0 & 0 \\ 0 & m_{y,i} & 0 \\ 0 & 0 & m_{\theta,i} \end{bmatrix} J_i(r) \right) \tag{4.29}$$

for systems with large discrete elements, and as

$$M(r) = \sum_i \left( J_i^T(r) \begin{bmatrix} \ell_i \rho_{m,x} & 0 & 0 \\ 0 & \ell_i \rho_{m,y} & 0 \\ 0 & 0 & 0 \end{bmatrix} J_i(r) \right) \tag{4.30}$$

for systems being formulated in terms of a large number of small mass components with identical hydrodynamic properties. Here, $\rho_m$ refers to the mass density in a particular direction of a hydrodynamic element $i$ in terms of the length $\ell_i$ of the small mass element component. This is an accurate approximation so long as $\ell_i$ is sufficiently small that the rotational term can be neglected. From here we will continue with the derivation for the hydrodynamic density formulation, as the large discrete element formulation is identical but with a larger number of coefficients.

We can split the density terms into two separate components and extract them from the pullback sum:

$$M(r) = \rho_{m,x} \sum_i \left( J_i^T(r) \begin{bmatrix} \ell_i & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} J_i(r) \right) + \rho_{m_y} \sum_i \left( J_i^T(r) \begin{bmatrix} 0 & 0 & 0 \\ 0 & \ell_i & 0 \\ 0 & 0 & 0 \end{bmatrix} J_i(r) \right). \tag{4.31}$$

This can be alternatively expressed in terms of mass matrix derivatives with respect to the density coefficients, as

$$M(r) = \rho_{m,x} \frac{\delta M(r)}{\delta \rho_{m,x}} + \rho_{m,y} \frac{\delta M(r)}{\delta \rho_{m,y}}. \tag{4.32}$$

In order to calculate forces due to Coriolis effects, we need derivatives of the mass matrix with respect to each of the $j$ shape variables $r_j$. Fortunately, because neither of the $\frac{\delta M}{\delta \rho}$ terms are functions of the density coefficients, and because the mass densities are also not functions of shape, this is a simple calculation that leaves the Coriolis forces linear on the density coefficients:

$$\frac{\delta M(r)}{\delta r_j} = \rho_{m,x} \frac{\delta M(r)}{\delta \rho_{m,x} \delta r_j} + \rho_{m,y} \frac{\delta M(r)}{\delta \rho_{m,y} \delta r_j} \tag{4.33}$$

For the first term,

$$\frac{\delta M(r)}{\delta \rho_{m,x} \delta r_j} = 2 \sum_i \left( J_i^T(r) \begin{bmatrix} \ell_i & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\delta J_i(r)}{\delta r_j} \right), \tag{4.34}$$

and for the second term,

$$\frac{\delta M(r)}{\delta \rho_{m,y} \delta r_j} = 2 \sum_i \left( J_i^T(r) \begin{bmatrix} 0 & 0 & 0 \\ 0 & \ell_i & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\delta J_i(r)}{\delta r_j} \right). \tag{4.35}$$

After calculating the mass matrix and its derivatives as linear combinations of the density coefficients, we can collect all of the terms from Eq. (4.27) that depend on the mass matrix and its derivatives. These are the adjoint, Coriolis,

and acceleration force terms. We then split each of those terms by the mass density coefficients and combine them into two terms that are linear on the density coefficients. We can then plug these terms back into the full equation of motion:

$$\begin{bmatrix} f_b \\ \tau \end{bmatrix} = \rho_{m,x} R_{m,x}\left(\mathring{g}, \frac{\delta \mathring{g}}{\delta t}, r, \dot{r}, \ddot{r}\right) + \rho_{m,y} R_{m,y}\left(\mathring{g}, \frac{\delta \mathring{g}}{\delta t}, r, \dot{r}, \ddot{r}\right) + D_L + D_Q + F_K. \quad (4.36)$$

Here, $R$ represents the regressor function [37], which relates the magnitude of generalized forces to the respective coefficient and is a function of shape and its derivatives, body velocity, and body-frame acceleration.

Because the linear and quadratic drag terms, calculated in Eq. (4.25) and Eq. (4.26) respectively, are also found by pulling back coefficients through the Jacobians, we can use an identical process to write the drag forces in terms of the drag density coefficients. This produces an equation for generalized body and shape forces that is completely linear on all of the desired density coefficients,

$$\begin{bmatrix} f_b \\ \tau \end{bmatrix} = R\left(\mathring{g}, \frac{\delta \mathring{g}}{\delta t}, r, \dot{r}, \ddot{r}\right) \rho + F_K, \quad (4.37)$$

where $\rho$ is a column vector of the density coefficients and $R$ is the regression matrix, which relates the dynamic coefficients to generalized forces. If the spring is linear and the spring constant is unknown, the spring constant can be experimentally identified alongside the density coefficients by taking the derivative of Eq. (4.17) with respect to the spring constant and adding an additional column to

$R$ representing the results:

$$\begin{bmatrix} f_b \\ \tau \end{bmatrix} = \begin{bmatrix} R_\rho \left( \mathring{g}, \frac{\delta \mathring{g}}{\delta t}, r, \dot{r}, \ddot{r} \right) & R_k \left( \tilde{g}, r \right) \end{bmatrix} \begin{bmatrix} \rho \\ k \end{bmatrix}. \tag{4.38}$$

If, instead, the spring forcing function is precisely known in advance, the spring forces can be factored into the generalized force observation to perform extraction on the density coefficients alone, giving

$$\begin{bmatrix} f_b \\ \tau \end{bmatrix} - F_K \left( \tilde{g}, r \right) = S \left( \mathring{g}, \frac{\delta \mathring{g}}{\delta t}, r, \dot{r}, \ddot{r} \right) \rho. \tag{4.39}$$

This is useful for extraction on systems that have a mixture of rigid and elastic constraints, such as a swimming system consisting of actuated 'fins' pinned to a sliding rail that is elastically connected to one point of a water tank, or for systems with passive-elastic shape components that possess well-known properties.

From Eq. (4.39) we can selectively choose equation rows where we have either reliable constraint force observations or knowledge of an elastic connection alongside reliable displacement observations, and discard the other rows on which we are not observing experimental data. For example, in the rail-swimming example described in the previous paragraph, we might choose to keep rows from Eq. (4.39) corresponding to forward displacement along the rail and to observed torques output by a motor executing the shape changes, ending with something akin to

$$\begin{bmatrix} -k_{rail}\tilde{g}_x \\ \tau_{motor} \end{bmatrix} = \begin{bmatrix} R_x\left(\mathring{g}, \frac{\delta\mathring{g}}{\delta t}, r, \dot{r}, \ddot{r}\right) \\ R_r\left(\mathring{g}, \frac{\delta\mathring{g}}{\delta t}, r, \dot{r}, \ddot{r}\right) \end{bmatrix} \rho. \tag{4.40}$$

More generally, we can selectively choose rows from the coefficient-linear equations of motion where we have a vector of useful experimental observations $O$. At an instantaneous time instance $i$, this forms an observed relationship between the system coefficients, system state, state derivatives, and system forces,

$$O_i = R\left(\mathring{g}_i, \frac{\delta\mathring{g}_i}{\delta t}, r_i, \dot{r}_i, \ddot{r}_i\right) \rho. \tag{4.41}$$

We perform multiple of these observations across the duration of the experiment. These observations allow us to stack information from all of the $n$ time instances that comprise the experiment, giving a compiled list of coefficient-to-force relationships,

$$\begin{bmatrix} O_1 \\ \vdots \\ O_n \end{bmatrix} = \begin{bmatrix} R\left(\mathring{g}_1, \frac{\delta\mathring{g}_1}{\delta t}, r_1, \dot{r}_1, \ddot{r}_1\right) \\ \vdots \\ R\left(\mathring{g}_n, \frac{\delta\mathring{g}_n}{\delta t}, r_n, \dot{r}_n, \ddot{r}_n\right) \end{bmatrix} \rho. \tag{4.42}$$

These stacked matrices contain all of the information concerning the relationship of system coefficients to experimental forces. We use tilde notation to denote data compiled from an entire experiment,

$$\tilde{O} = \tilde{R}\rho. \tag{4.43}$$

The experimental regression matrix $\tilde{R}$ tells us whether the proposed experiment is sufficient to perform regression on the proposed coefficients. The coefficients are observable only if $\tilde{R}$ has full column rank. If any of the columns are zero, the corresponding coefficients do not impact the experimental observations and thus cannot be estimated from the recorded data. If any of the columns are linearly dependent, there will not be sufficient information to distinguish the impact of the codependent coefficients [56].

Conveniently, $\tilde{R}$ is just a geometric statement on the possible states of the locomotor. This means that the rank of $\tilde{R}$ can be checked before performing the experiment by generating an estimated set of states that might be achieved during the experiment. For a proposed experiment, one could generate a list of a subset of possible states that might be achieved, taking into account experimental constraints that limit the set of reachable state velocities and accelerations. As long as this test regression matrix contains a sufficient sampling of experimental states, it can be used in lieu of the actual experimental sensitivity matrix to determine whether the desired coefficients will be observable given the experimental constraints.

If $\tilde{R}$ has full column rank, the coefficients are observable from the experiment and the coefficients can be extracted using the experimental observations by taking the pseudoinverse of the regression matrix,

$$\rho = \tilde{R}^{+}\tilde{O}. \tag{4.44}$$

Using the pseudoinverse of the sensitivity matrix in this manner produces a least-squares estimate of the system coefficients. This is a convenient method of coefficient extraction for when there is a relatively modest amount of experimental data and the dynamic forces present in the experiment are well-modelled.

However, this technique does not provide guarantees on satisfying some basic physical principles in the system. Nowhere in this formulation was the constraint imposed that the coefficients must be positive. In experiments with high noise thresholds, or in experiments with large unmodelled forces, it is very possible that this method will produce unphysical coefficient estimations such as negative mass or negative drag. In such cases, this problem turns from a simple pseudoinverse matrix multiplication to a constrained least-squares optimization problem, solvable using a function like MatLab's `lsqlin`. In Section 4.3 we will use this constrained optimization to regress coefficients that satisfy physical constraints on the dynamic coefficients.

The regression matrix can also be used to perform tasks such as collision detection or detection of other unmodeled events via online parameter estimation [37]. If a system is associated with a set of dynamic parameters which can be re-estimated on the fly, times where the best-fit dynamic parameters leave an acceptable threshold correspond to instances where system behavior is not well-explained by the dynamic model. Such a case likely corresponds to a collision event or unmodeled physics. To enable such capability, we also here we describe an iterative method of coefficient extraction using the regression matrix that can be tailored to produce physical coefficient values that best fit experimental results over time.

We start by choosing an arbitrary set of coefficient values $\rho_k$ that fit the basic physical constraints. We then examine one of the experimental force observations and the corresponding row of the regression matrix,

$$f = R\rho = \begin{bmatrix} R_1 & \dots & R_n \end{bmatrix} \begin{bmatrix} \rho_1 \\ \vdots \\ \rho_n \end{bmatrix}. \tag{4.45}$$

This equation defines an affine hyperplane of dimension $n-1$ in the space of $n$ coefficients. We can refine our guess for $\rho_k$ by projecting it onto this hyperplane:

$$\rho_{k+1} = \rho_k - \frac{R\rho_k - f}{\|r\|}\hat{R}^T. \tag{4.46}$$

Here, $\hat{r}$ represents the normalized row of the sensitivity matrix,

$$\hat{r} = \frac{R}{\|R\|}. \tag{4.47}$$

This update step might result in unphysical negative coefficient values. To correct this, we constrain any negative coefficient values to be zero and reform the observation equation using only the unconstrained coefficients,

$$f = R_{\rho>0}\rho_{\rho>0}. \tag{4.48}$$

We can then repeat the hyperplane projection in Eq. (4.46) using only values corresponding to above-zero coefficients to find the closest set of coefficients to our original estimate that both lie on the observational hyperplane and satisfy the

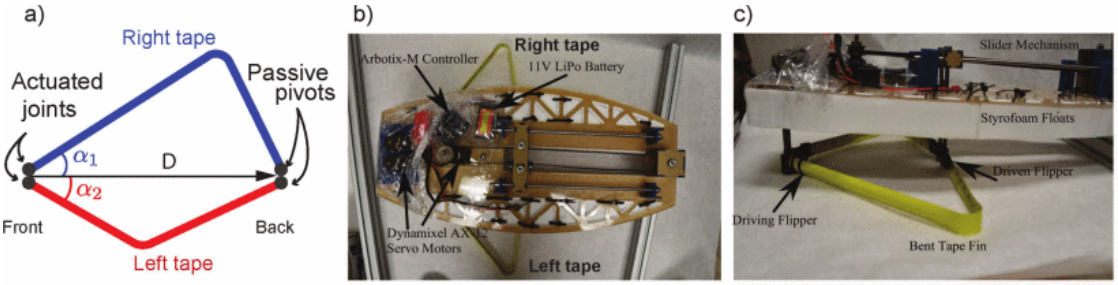


Figure 4.3: The AmoeBot swimming system. Two semi-independent tape appendages are actuated by individual motors that set the angle of the tape with respect to the front of the robot, and by a linear actuator that sets the distance between the front and back connection points. Figure borrowed from [57]. Arbitrarily, we refer to the side of the AmoeBot with the motors as the "front."

constraint of non-negative values.

We can use this iterative update method alongside a notion of sensor variance to implement a Kalman filter coefficient estimator that uses these hyperplane projections as coefficient measurements. Unscented Kalman filters have been previously used to estimate dynamic coefficients in noisy experimental systems [66,69]. Alongside the advantages of being resilient to large experimental datasets and being robust to realistic coefficient constraints, this method has the capability of being implemented in an online fashion on freely-swimming locomotors. In these cases, this form of system identification would provide an online estimate of swimmer model dynamic properties and could update model predictions for sudden changes in dynamic properties, such as mechanism breakage.

## 4.3 Experimental Validation

In this section, we will discuss the process of validating this process of constrained system identification on the AmoeBot [57], a swimming robot with a novel tape-spring swimming appendage developed by our collaborators Curtis Sparks and Nick Gravish from UCSD. This robot consists of a floating platform holding the system electronics while a pair of tape-spring 'fins' with passive buckling move to push the robot through the water. The AmoeBot can be seen in Fig. 4.3. Here, we discuss:

- How we formed our geometric model for the AmoeBot
- How we performed dynamic coefficient regression on this model to fit the experimental data
- How we used the model to optimize a gait library for the AmoeBot
- How we developed a control field for each of the gaits in the library
- Results for two manually-piloted AmoeBot navigation experiments

### 4.3.1 AmoeBot Geometric Model

The geometric model we used to represent each of the tape fins is derived using findings from the original AmoeBot design paper [57]. From this work, we saw that the tape fin typically buckles at only one point when bent and that this buckle location has a fairly constant bend radius regardless of buckle location.

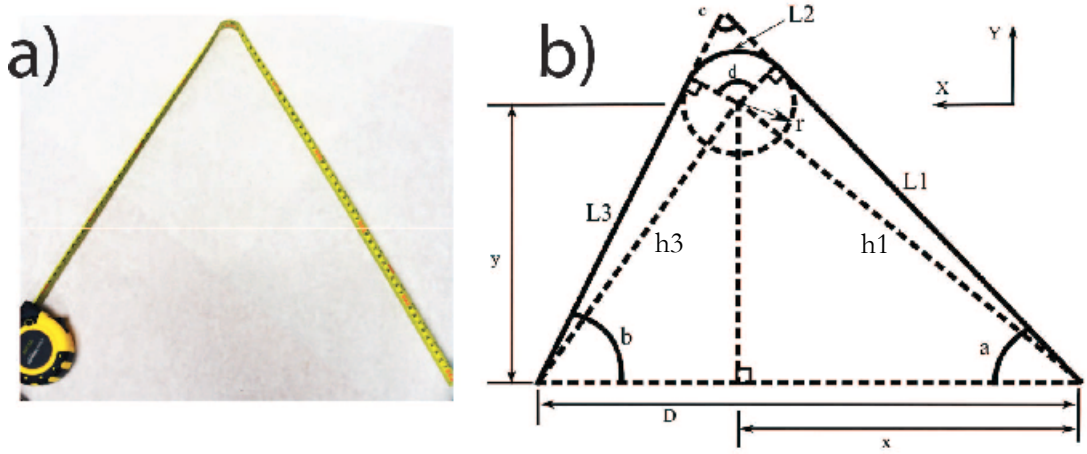


Figure 4.4: (a) An example of a tape-measure buckling point on which we base the AmoeBot geometry. This buckle point has approximately constant bending radius regardless of the buckle location. (b) The geometric model that we use to solve for the fin structure as a function of the angle setting $a$ and the distance setting $D$. Using constraints within this geometry, we can solve for the three tape section lengths $L_1$, $L_2$, and $L_3$, and the angle $b$ of the passive rear connection point. Figure borrowed from [57].

An illustration of this buckling geometry can be seen in Fig. 4.4 and will be used to derive the constraints that determine the fin shape as a function of the input parameters.

The triangular tape structure is a function of the tape angle setting $a$ which is set by the forward servos, the distance setting $D$ which is set by a linear actuator, and the total tape length $C$ and tape bend radius $d$ which are both constant values. For the AmoeBot used in our experimentation, each tape fin at a tape length of $C = 1$ foot, and the tape bend radius was found to be approximately $d = 0.5$ inches. From these driving values, we can estimate the four parameters that completely determine the shape of the tape fin. These parameters are the

length of the forwardmost straight section of the fin $L_1$, the arclength of the tape along the curved buckle point $L_2$, the length of the rearmost straight section of the fin $L_3$, and the angle of the passive rear connection point $b$.

It also useful to use intermediate variables $h_1$ and $h_3$ which are the angle distances from the front and back connection points respectively to the center of the tape buckling 'circle'

$$h_1 = \sqrt{L_1^2 + d^2}, \tag{4.49}$$

$$h_3 = \sqrt{L_3^2 + d^2}. \tag{4.50}$$

Additionally, we will utilize in this process the angle from the swimmer spine to the buckle 'center' for the front $a_c$ and back $b_c$ of the fin,

$$a_c = a - \tan^{-1}\frac{d}{L_1}, \tag{4.51}$$

$$b_c = b - \tan^{-1}\frac{d}{L_3}. \tag{4.52}$$

We determine the driven parameters $(L_1, L_2, L_3, b)$ by applying four geometric constraints on the fin shape. First, we constrain that the three individual tape lengths must collectively be equal to the total tape length,

$$C = L_1 + L_2 + L_3. \tag{4.53}$$

Then, we apply the fact that the arclength of the curved buckling region can be found from the front and back connection angles,

$$L_2 = d(a + b). \tag{4.54}$$

Next, we reconcile the horizontal distances to the buckle center with the distance setting,

$$D = h_1 \cos a_c + h_3 \cos b_c. \tag{4.55}$$

Finally, we reconcile the vertical distances to the buckling center from each end of the fin,

$$h_1 \sin a_c = h_3 \sin b_c. \tag{4.56}$$

We can then apply an iterative method using `fsolve` in MatLab to find the values of the four driven parameters $(L_1, L_2, L_3, b)$ that satisfy these four constraints. This gives the function that we use to estimate the tape shape as a function of the joint angle and spine distance setting. We use numerical methods to estimate derivatives of these parameters with respect to the driving shape variables when performing system identification in the next section.

### 4.3.2 Experimental Coefficient Regression

To perform coefficient regression on the dynamic parameters of the AmoeBot fins, we attached a test platform consisting of one fin and associated actuators to a 6DOF load cell. Motion commands were sent to the driving motors and constraint forces and torques were recorded. The experimental setup can be seen in Fig. 4.5.

The experimental trials were repeated 20 times in each direction for each of the motion primitives used in the test. For data extraction, only one motion primitive was used, which consisted of a sweep in joint angle from 90 degrees down to 30 degrees while the base length stayed stationary. We find that this motion and the corresponding reaction forces contains enough information to identify the tape hydrodynamic parameters. This experimental data can be seen in Fig. 4.6.

There is some sinusoidal oscillation in the force data that we expect is from the cantilever action of the fin in the water away from the load cell. As the fin actuates, this cantilever structure is excited into resonance, producing the sinusoidal experimental noise. As the regression process provides a best fit despite noise, we decided to leave the sinusoidal activity in the signal rather than attempting to filter it out due to the risk of erasure of signal features.

For the force data, we chose to keep only the thrust and lateral force readings from the experiment, as these measurements proved the cleanest. To generate the actuator signal profiles, we took numerical derivatives of the provided shape data and used a low-pass butterworth filter to clean the noise, resulting in time-functions for actuator shape, velocity, and acceleration. This data was then used

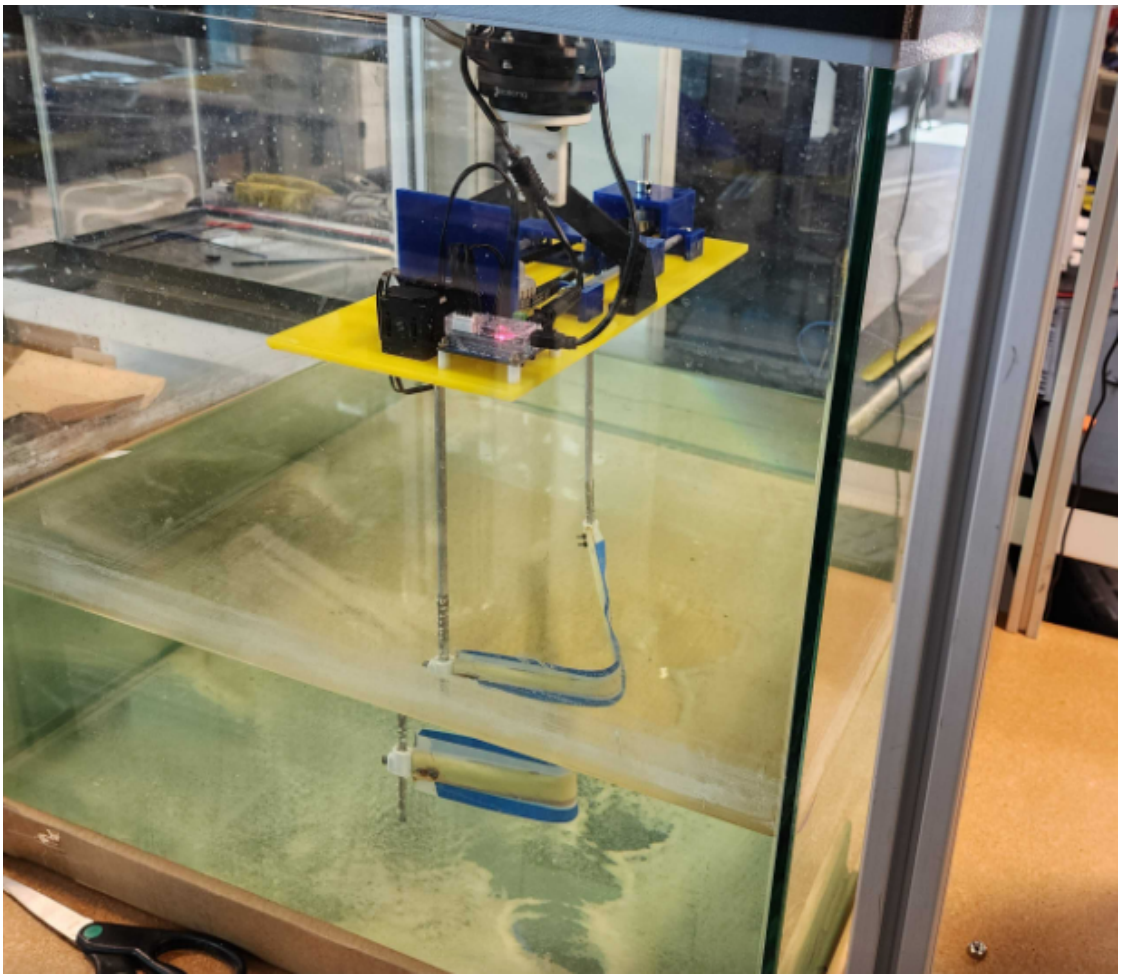

Figure 4.5: Visualization of the AmoeBot system identification experiment. One tape-spring swimming appendage is immersed in water while undergoing shape changes that it would experience while attached to the AmoeBot. Forward and lateral constraint forces that hold the system stationary in the tank were recorded and used to perform system identification on the hydrodynamic coefficients.

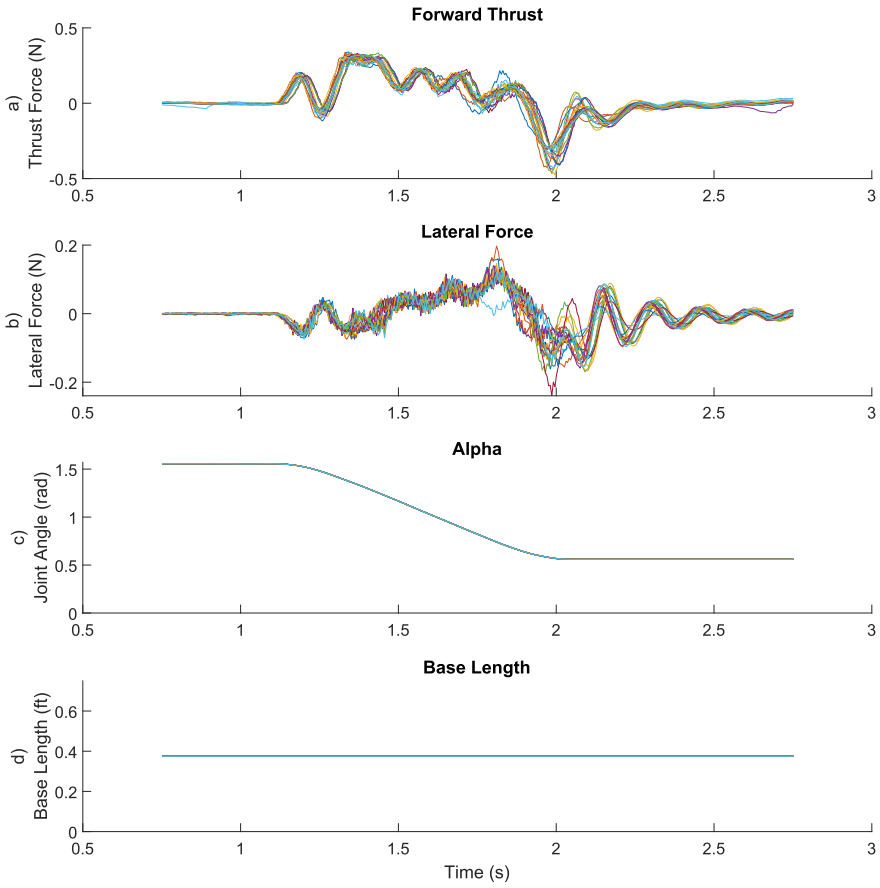


Figure 4.6: Constrained experiment data used to regress the AmoeBot fin dynamic coefficients. Data from 20 trials of the same motion primitive was used in the regression. (a) The forward thrust measured by the load cell as the fin shifts backwards. (b) The lateral thrust measured by the load cell (c) The measured joint angle profile over the experiment (d) The base length setting, which was held constant for this motion primitive.

to estimate the fin structure using the fin geometric model outlined in §4.3.1. From this geometric model, we estimated the regression matrix for every experimental data point using the methodology presented in §4.2.2 and collected them. We then solved the constrained least-squares problem to fit the dynamic coefficients while maintaining that they have positive definite values. These coefficients were the lateral and longitudinal hydrodynamic mass density, the lateral and longitudinal linear drag coefficient, and the lateral and longitudinal quadratic drag coefficient for the tape fin.

Although mathematically involved, this process has a simple qualitative explanation. For a given shape motion in the experiment, we ask ourselves "What would we expect the force response to this motion to be if this component had unit mass density in the lateral direction and all other coefficients were zero? What about in the longitudinal direction? What about for the drag coefficients?" From this isolation of the coefficients, we end up with trendlines over the experimental motion for each coefficient that collectively attempt to explain measured forces. The problem of coefficient regression then becomes finding the linear combination of these isolated coefficient trendlines that best explains the experimental forces. This trendline explanation of the regression process for the lateral mass and quadratic drag densities on the AmoeBot tape fin is shown in Fig. 4.7.

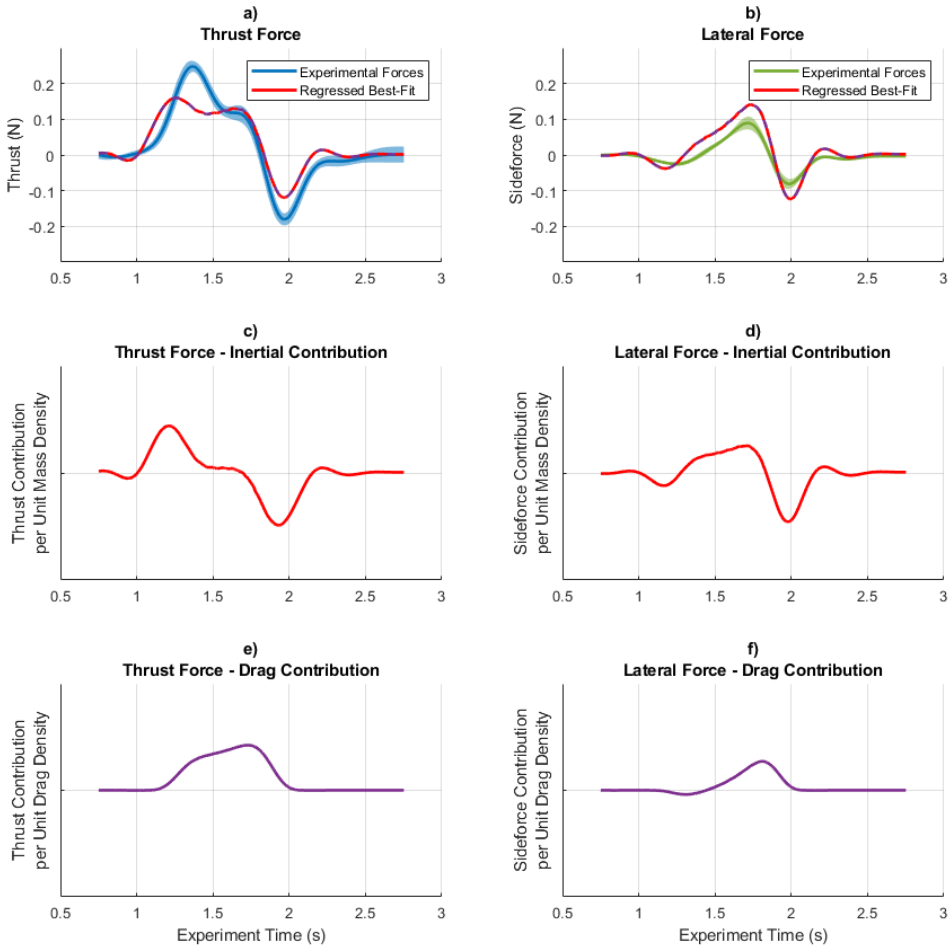


Figure 4.7: Fitting lateral mass and quadratic drag densities to the experimental AmoeBot forces. (a) In blue, the mean and standard deviation window for thrust force across the experiment. In red and purple, the linear combination of the mass and drag contributions that best fits the thrust and lateral force data. (b) In green, the mean and standard deviation window for lateral force across the experiment. In red and purple, the linear best-fit force combination of the coefficients. The thrust and lateral force data are fit simultaneously. (c) The expected inertial contribution profile to the thrust force if the tape had unit mass density. (d) The expected inertial contribution to the lateral force if the tape had unit mass density. (e) The expected drag contribution to the thrust force if the tape had unit drag density. (f) The expected drag contribution to the lateral force if the tape had unit drag density.

### 4.3.3 Gait Library Optimization

In this section, we will use the regressed coefficients to optimize locomotive gaits for the AmoeBot. For this optimization process, we use two different dynamic models. In the first, we use a model where we fit only the linear drag densities so that we may perform a low Reynolds number locomotive approximation. Using this model and the process described in §4.3.2, we found that the experimental forces are best described by a zero longitudinal drag coefficient and a lateral drag coefficient of $\rho_{l,y} = 0.4451\frac{lbfs}{ft^2}$. In the second dynamic model, we fit the mass and quadratic drag densities to generate more accurate predictions of the AmoeBot as a locomotor that is capable of admitting non-zero generalized momentum. For this model, we find the mass densities as $\rho_{m,x} = 0.0206\frac{slug}{ft}$ and $\rho_{m,y} = 0.0822\frac{slug}{ft}$, and the quadratic drag coefficients as $\rho_{q,x} = 0\frac{lbfs^2}{ft^3}$ and $\rho_{q,y} = 1.0952\frac{lbfs^2}{ft^3}$.

In our experimental coefficient regression, we determined coefficients only for the tape fins on the AmoeBot. However, there are other hydrodynamic elements in the system. Most notably, there are two styrofoam floats, one on each side of the robot, that stabilize the AmoeBot atop the water. To roughly model these floats, we used the same dynamic densities that were determined for the tape fins to model the edges of the styrofoam. This allows us to approximate the effect of the AmoeBot body on locomotive efficacy. This rough model is illustrated in Fig. 4.8.

Initial gait generation was done using the low Reynolds number physics approximation and regressed linear drag coefficients. We used these in `sysplotter`,

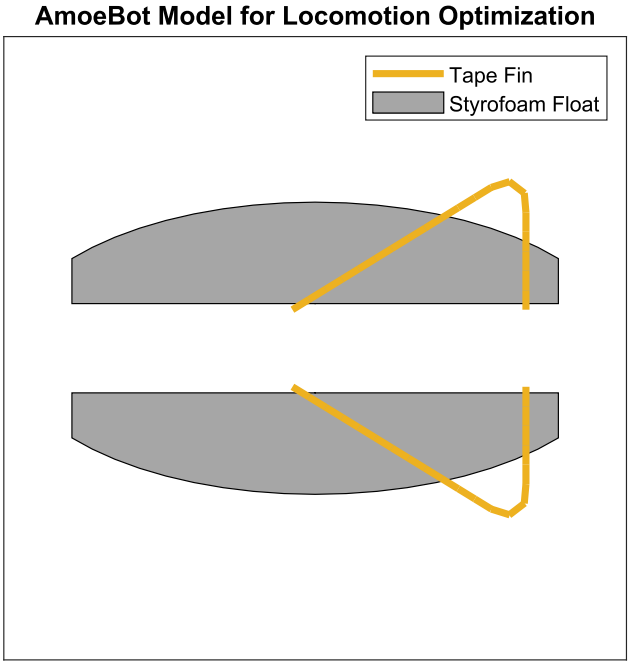


Figure 4.8: Simple model for the AmoeBot used in gait optimization. The yellow lines represent the tape fin, with dynamic properties determined by experimental coefficient regression. For the styrofoam float, we used the same experimental coefficients as regressed for the tape around the perimeter of the two independent styrofoam sections to provide a rough estimate of the effect of the styrofoam on AmoeBot locomotion.

one of the tools in our lab that has been previously developed to optimize motion for kinematic locomoting systems [45, 46, 48]. By implementing the geometric and physics model for the AmoeBot in `sysplotter`, we leveraged previous work to quickly determine cyclic shape changes parameterized by fourier coefficients on each of the shape variables that produce useful motion.

To begin gait family optimization, we decided on the different types of control actions that we need to fill out the gait library. For the AmoeBot, we ended up with a few different types of action. First, we generated a forward motion gait. This was found by optimizing for the forward gait efficiency $\eta_X$,

$$\eta_X = \frac{g_x}{D}, \tag{4.57}$$

where $D$ represents the metric-weighted arclength of the gait. This optimization produces a candidate gait that propels the AmoeBot forward by a value of $\tilde{g}_x$. This forward motion plan also comes with a backward motion counterpart. By time-reversing the gait signal, we generate motion that propels the AmoeBot backwards. Although this motion will likely not be the exact inverse of the forward motion due to the unmodeled pitching effect of the AmoeBot in water, it still serves as a useful motion plan that effectively produces reverse motion.

We also generated half-step gaits for forward-backward motion that we felt would be useful for tasks like station keeping or precise trajectory following over time. To do this, we use the same objective function as above for the small-forward gait efficiency $\eta_x$, but additionally apply the constraint that the gait must travel

half the locomotive distance,

$$\begin{aligned} \eta_x =& \frac{g_x}{D}, \\ \text{such that} \quad & g_x = \frac{\tilde{g}_x}{2}. \end{aligned} \tag{4.58}$$

In the same way as for the larger forward gait, this optimization produces a cyclic shape curve parameterized by Fourier variables that can be time-reversed to produce an analagous backwards gait.

Next, similar to the forward motion, we used sysplotter to optimize turning gaits by formulating objective functions for turning motion $\eta_\Theta$ and for half-step turning motion $\eta_\theta$. The objective function for the larger gait,

$$\eta_\Theta = \frac{g_\theta}{D}, \tag{4.59}$$

produces an optimized turning displacement per cycle $\tilde{g}_\theta$. Using this, we can constrain the half-step gait,

$$\begin{aligned} \eta_\theta =& \frac{g_\theta}{D}, \\ \text{such that} \quad & g_\theta = \frac{\tilde{g}_\theta}{2}. \end{aligned} \tag{4.60}$$

Similar to the forward motion gaits, these turning motions can be reversed by time-inverting the gait. Unlike the asymmetry of the forward-backward motion pair due to three-dimensional effects, this time reversal produces equivalent gaits

due to the bilateral symmetry of the AmoeBot.

For the final gait motion, we attempt to combine net forward and turning displacement. We call this gait the steering gait, with objective function $\eta_s$. The idea is to normalize resulting gait displacements by the maximum values and find a gait that produces both forward and turning motion using the objective function

$$\eta_s = \frac{\frac{g_x}{\tilde{g}_x} + \frac{g_\theta}{\tilde{g}_\theta}}{D}. \tag{4.61}$$

This gait is very useful and results in four types of motion. The default gait produces positive locomotion in $g_x$ and $g_\theta$, moving the AmoeBot forward and counterclockwise. By invoking the bilateral symmetry of the AmoeBot, the joint commands sent to each of the servos can be swapped, resulting in a gait that moves the AmoeBot forward and clockwise. Finally, by performing time-reversals of these two gaits, we produce motions that can move the AmoeBot backwards while simultaneously rotating either clockwise or counterclockwise. Qualitatively, these gaits look as if the AmoeBot were performing the forward or backward gait with only one fin while leaving the other stationary.

The final 'gait' used to flesh out the AmoeBot control library is not really a gait at all: it is the zero-movement case. For this gait, the AmoeBot holds still at whatever the last actuator signal was. This is useful as a default case for the AmoeBot, where the AmoeBot ceases motion whenever no high-level control command is supplied.

Collectively, these gaits can be mapped onto the space of possible inputs to a

joystick. This mapping is shown in Fig. 4.9. In the center is the black no-motion gait. Each gait is associated with a control point on the joystick range, with gait execution decided by which gait the joystick output is closest to. This results in a division of the joystick space into voronoi cells, with each cell associated to a gait.

Initially, we deployed these gaits expressed as shape-change cycles onto the AmoeBot. Our control loop executed the predetermined shape changes at the maximum possible actuator speed. However, this resulted in poor performance. The AmoeBot would generate a significant amount of beneficial momentum on the power stroke of the gait, but would perform the return stroke equally quickly and the momentum would reverse, resulting in poor locomotive performance. Although the robot still experienced some net locomotion as expected by the optimizer, we realized that we could improve performance by taking into account the momentum dynamics of the robot.

To optimize the momentum dynamics, we performed an optimization of the pacing along the gait using the mass and quadratic drag parameters regressed from the experimental data. This optimization was performed over a normalized set of shape coordinates. We found that expressing servo angles in radians and linear actuator distance in feet produced a large disparity in the relative arclength of the curve as it travels in each dimension. This was a problem, as the servomotors experienced large gait arclengths but can turn quickly, and the linear actuator corresponded to a small fraction of the uncorrected gait arclength but moved very slowly in practice. To account for this and better identify features on the gait that need accounted for by pacing, we independently normalized each of the shape

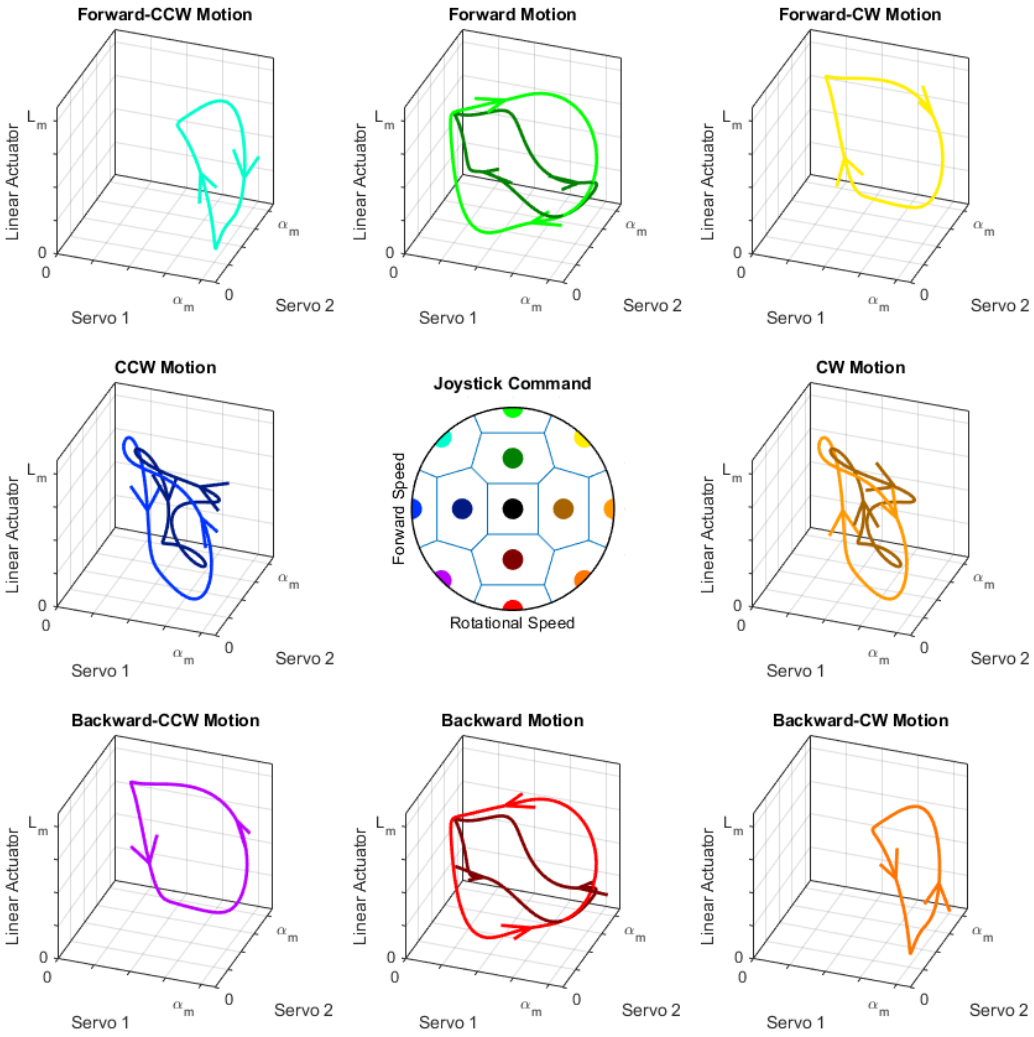


Figure 4.9: Division of the control joystick input into gaits. Each gait is associated with a control point on the joystick range, with gait execution decided by which gait the joystick output is closest to. This results in a division of the joystick space into voronoi cells, with each cell associated to a gait. Diagonally-opposed gaits are the same motion that has been time-reversed. Positive forward speed refers to forward motion and positive rotational speed refers to clockwise motion. This inversion of the right-hand rule makes the AmoeBot more intuitive to pilot.

modes $r$ by performing

$$\hat{r}(r) = \frac{r - r_{min}}{r_{max} - r_{min}}. \tag{4.62}$$

On this shape space that more equitably distributes the arclength $\hat{s}$ between the dimensions, we find that gait pacing optimizations converge faster and more reliably, as gait features are more easily associated with particular arclength ranges.

We scale pacing $v(\hat{s})$ relative to the actuator limitations, with the maximum pacing $v_{max} = 1$ performing the shape changes at the upper bound of speeds we place on the servos and the linear actuator. These speeds are quite different between the actuators due to mechanical limitations, with the servos traveling back and forth across their full range of actuation in about a second while the linear actuator takes eight seconds to perform a similar sweep. If pacing is commanded above the maximum values allowed by the actuation, the gait will follow a different path through shape space than intended as the motors struggle to keep up and locomotion will be less efficient. In addition to the pacing upper bound of $v_{max} = 1$, we also implement a pacing lower bound of $v_{min} = 0.2$, corresponding to the actuators moving at 20% speed. We implement this, because if the pacing value hits zero actuation ceases. We found that without the protection of $v(\hat{s}) \geq 0.2$, gait optimization freezes as motion slows to a halt and the full gait motion estimation cannot be completed.

With these limits on pacing values in place, we implement a similar Fourier parameterization of the pacing as a function of the normalized arclength,

$$\begin{aligned} v(\hat{s}) =& a_0 + a_1 \cos \omega_s \hat{s} + b_1 \sin \omega_s \hat{s} \\ &+ a_2 \cos 2\omega_s \hat{s} + b_2 \sin 2\omega_s \hat{s} \\ &+ a_3 \cos 3\omega_s \hat{s} + b_3 \sin 3\omega_s \hat{s} \\ &+ a_4 \cos 4\omega_s \hat{s} + b_4 \sin 4\omega_s \hat{s}. \end{aligned} \tag{4.63}$$

Here, the gait frequency $\omega_s$ is a function of the total gait normalized arclength $\hat{S}$,

$$\omega_s = \frac{2\pi}{\hat{S}}. \tag{4.64}$$

Rather than constrain the function $v(\hat{s})$ to be between $v_{min}$ and $v_{max}$ through `fmincon`, we instead allow the function to be unconstrained but apply the pacing limits as they are applied in the controls loop. This speeds up optimization by reducing the need to follow many constraints.

The Fourier parameters are then optimized using MatLab's `fmincon`. To perform this optimization, we implement a locomotion simulator as in Eq. (4.27) using the hydrodynamic mass and quadratic drag densities regressed from the AmoeBot experimental data. We optimize similar to Eqs. (4.57) to (4.61), but using total gait execution time $T$ in place of weighted arclength cost $D$. This produces a velocity pacing as a function of phase along the gait for each of the gaits in our library. This optimization need only be performed five times, once for each of the forward gait, the half-step forward gait, the turning gait, the half-step turning gait, and the steering gait. As long as the pacing is associated with a particular

shape, the time-reversed and angle-reversed gaits can effectively utilize the same speed profile.

Generally, the primary effect of this pacing scalar after optimization is to slow down the pace of the gait on the return stroke. The majority of the gait is performed at the maximum allowable speed, with a sudden drop in gait pace as the return stroke is executed. Performing most of the gait at maximum allowable speed decreases the gait period, allowing a higher gait frequency and faster speed. Dropping the pace during the return stroke reduces the buildup of detrimental momentum, increasing the net displacement per gait cycle. An example of a gait pace profile for the forward-displacement gait is shown in Fig. 4.10.

### 4.3.4 Shape Control and Gait Transitions

The dual-layer optimization process described in the previous section results in thirteen gait shapes and gait pacing evolutions that our model predicts will result in useful maneuvering locomotion. We show in Fig. 4.9 how these gaits are mapped to a controller. Before we run an experiment, however, we must establish rules for how to transition between gaits and how to converge to a gait from an arbitrary shape position. These two questions end up being identical in nature. Much of this work was developed by my labmate and research collaborator Jinwoo Choi, and more detail on this topic can be found in his bibliography.

The problem statement is to find a rule for what control action to execute from any given starting point that cause simultaneous execution of the gait and

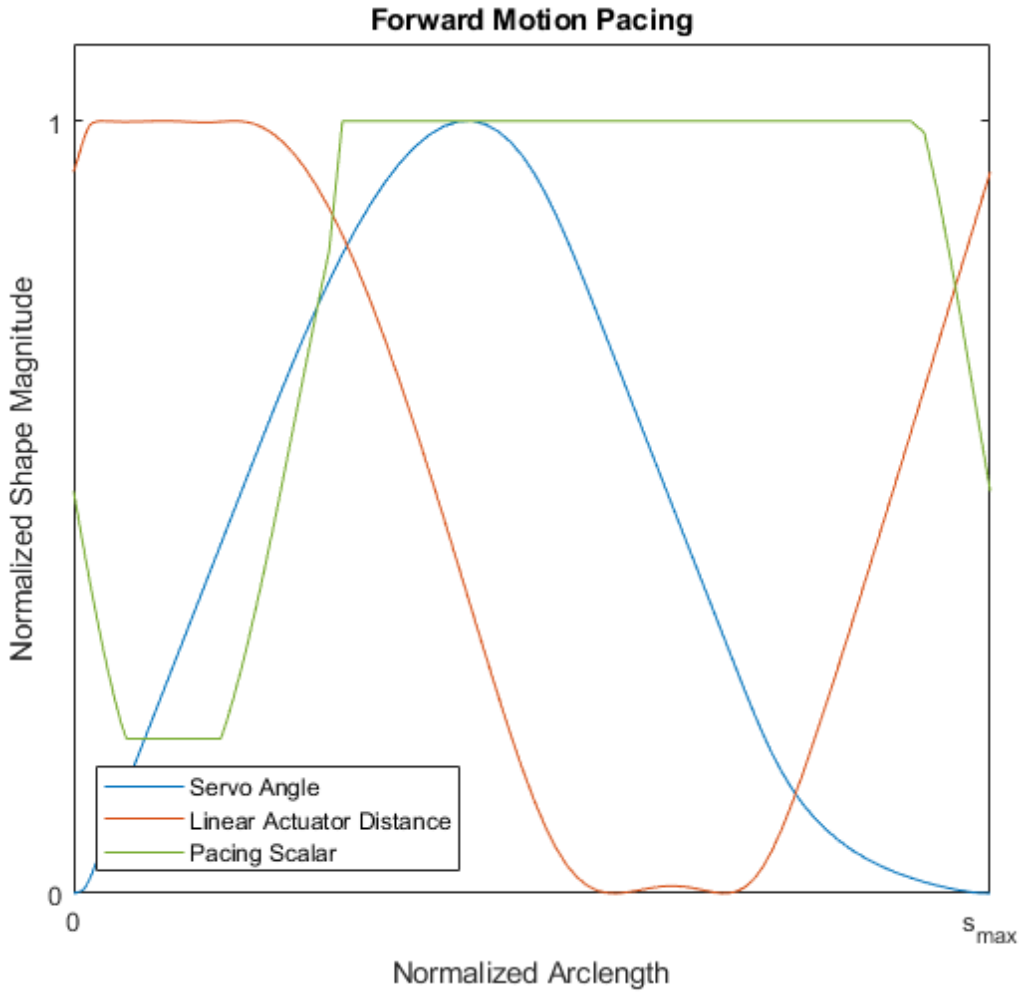


Figure 4.10: Visualization of the pacing profile for the forward-motion gait as a function of the normalized arclength. For most of the gait, the pacing scalar is 1 and the gait is executed at the maximum speed allowed by the actuators. However, for the return stroke while the servo angle 'resets,' the pacing scalar drops to the minimum value, reducing the buildup of detrimental momentum.

rejection of error to the desired limit cycle. To do this, we divide the gait $\phi$ up into $n$ points, equally spaced along the gait arclength. For an arbitrary shape value $\alpha$, we find the point on the gait $\phi_{md}$ of these $n$ points that has the shortest distance to $\alpha$,

$$\phi_{md}(\alpha) = \underset{\phi_1 \ldots \phi_n}{\arg\min} \|\phi_i - \alpha\|. \tag{4.65}$$

We then impose two control vectors based on this relationship. The first is the convergence vector $B(\alpha)$ which similar to the idea of vector field divergence points directly to the nearest location of the gait,

$$B(\alpha) = \phi_{md} - \alpha. \tag{4.66}$$

Second, we impose an analog to vector field curl that we call the circulation vector. This vector points parallel to the direction that the shape would flow if it were converged to the gait limit cycle, and is equivalent to the unit tangent vector $\hat{T}_\phi$ at the closest point on the gait. We find this unit tangent vector using central differencing on the neighboring points relative to the nearest gait location $\phi_{md}$,

$$\hat{T}_\phi(\alpha) = \frac{\phi_{md+1}(\alpha) - \phi_{md-1}(\alpha)}{\|\phi_{md+1}(\alpha) - \phi_{md-1}(\alpha)\|}. \tag{4.67}$$

We normalize the circulation vector but not the convergence vector because the convergence vector magnitude gives proportional control that rejects error from an arbitrary shape to the gait limit cycle. Meanwhile, the circulation vector should maintain constant magnitude so that we develop shape curves that execute the

gait limit cycle.

We combine these two vectors and normalize to produce the control velocity unit vector $\hat{C}(\alpha)$ using a weighting factor $w$ that determines how much convergence is prioritized over circulation,

$$\hat{C}(\alpha) = \frac{wB(\alpha) + \hat{T}_{\phi}(\alpha)}{\|wB(\alpha) + \hat{T}_{\phi}(\alpha)\|}. \tag{4.68}$$

For our control formulation, we used a weighting factor of $w = 2$.

From here, we scale the magnitude of the control velocity using two factors. The first scaling factor $\gamma$ sets the control velocity to the maximum amount that is achievable by the actuators. We find this scaling factor using the actuator speeds $\hat{C}_i(\alpha)$ and their corresponding maximum velocities $u_{i,max}$ for each of the $m$ actuators,

$$\gamma(\alpha) = \max_{C_1 \ldots C_m} |\frac{\hat{C}_i}{u_{i,max}}|. \tag{4.69}$$

This produces $\gamma(\alpha)$ as the fractional utilization of the most heavily utilized actuator under the unit control velocity. With this scalar taking control velocity to the maximum possible speed that the actuators are capable of producing, we then apply the pacing function $v(\hat{s})$ optimized in the previous section, noting that the normalized arclength along the gait is a function of the shape of the closest gait point $\phi_{md}$. This provides us with the control velocity actions for each of the shape values $\alpha_i$ that result in execution and error correction of the desired gait limit cycles,

$$C(\alpha) = \begin{bmatrix} \frac{\delta \alpha_1}{\delta t}(\alpha) \\ \vdots \\ \frac{\delta \alpha_m}{\delta t}(\alpha) \end{bmatrix} = \frac{v(\hat{s}(\phi_{md}))}{\gamma(\alpha)} \hat{C}(\alpha). \tag{4.70}$$

In practice, we precompute these control velocity vectors for a grid across the allowable shape space and interpolate them between the points to come up with our real-time control law. This produces a control vector field that, for every point in the shape space, flows the shape to and along the desired optimized limit cycle. A simplified example of this type of control field is shown for the low Reynolds number three-link swimmer forward motion gait in Fig. 4.11. A more complex three-dimensional example of this gait control technique is shown in Fig. 4.12.

Implementing this control law formation for each of the thirteen gaits lets us seamlessly switch between them. When a gait is commanded, the controller performs a lookup through the table of control laws to see which control field it should flow along. Swapping between gaits is simply a matter of changing the control field. For the zero motion gait, we use the control law $C(\alpha) = 0$.

### 4.3.5 Experimental Results

Here, we discuss the results of two experiments we performed on the AmoeBot using the control laws optimized from the experimentally regressed dynamic coefficients. We performed these experiments in a test tank filled with water with physical markers indicating desired trajectories and waypoints. We see that this

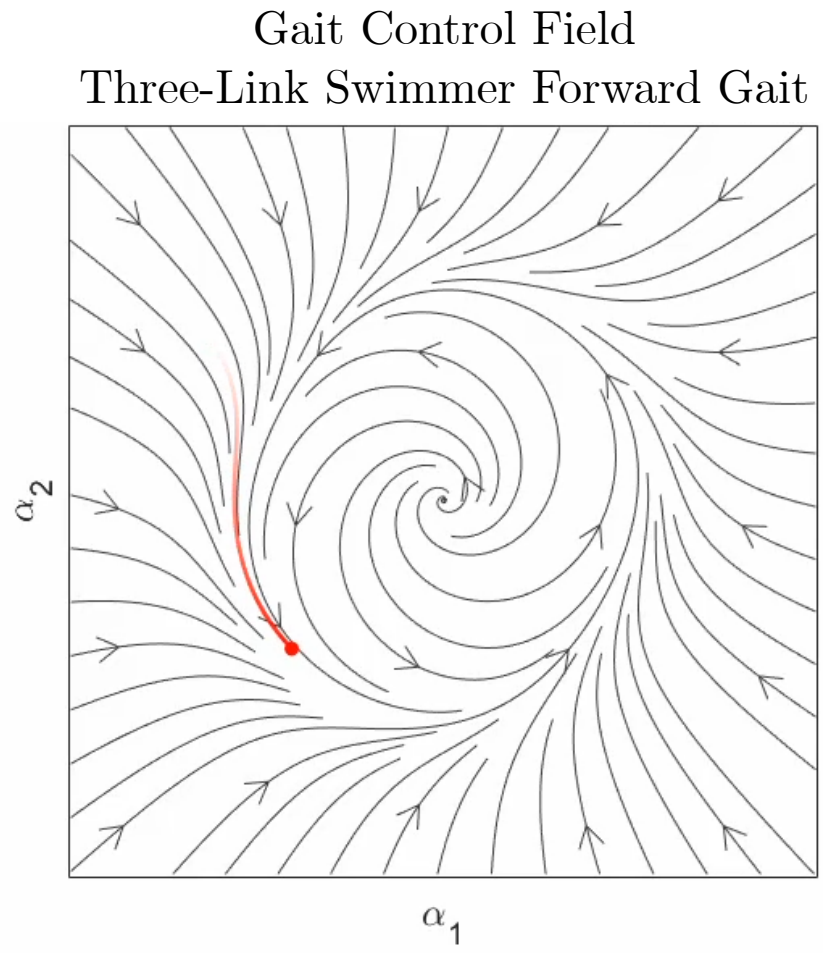


Figure 4.11: Visualization of a gait control field generated for the low Reynolds number three link swimmer forward gait using the methodology presented here. Executing the defined control actions from any initial shape point results in flows that converge to the execution of the desired gait. This figure was developed by my research collaborator Jinwoo Choi.

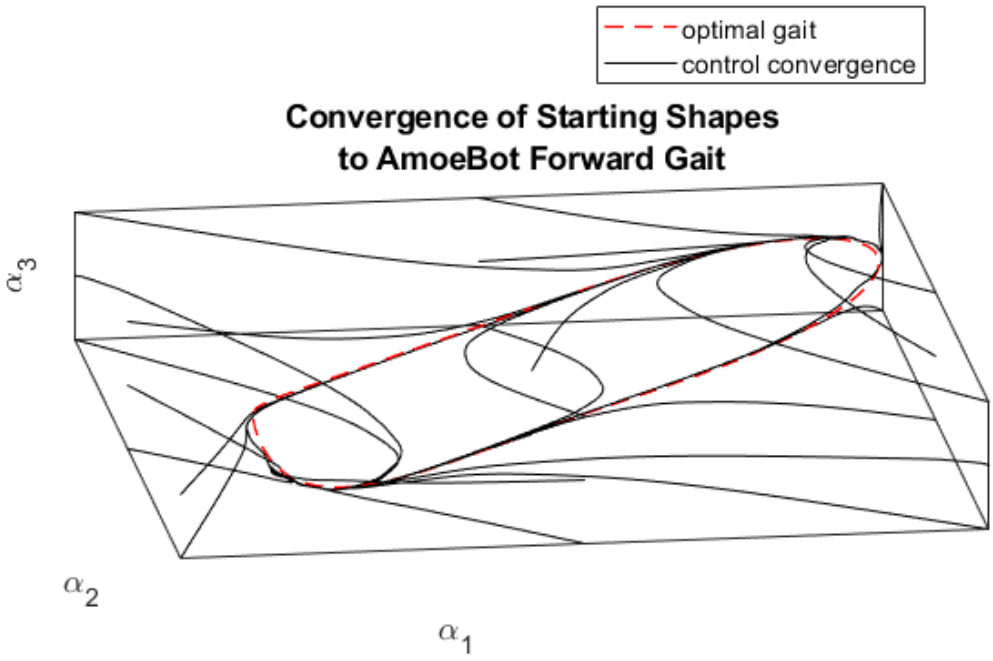


Figure 4.12: Visualization of the gait control field generated for the AmoeBot forward motion gait. The dotted red line represents the desired limit cycle and the thin black lines represent gaits from arbitrary starting points converging to the limit cycle.

method of control formulation from experimental data results in finite-time controllability to a variety of poses and orientations in the workspace and leads to intuitive control of the system through joystick input.

These experiments required coordination between three separate systems. On the AmoeBot itself, computation is performed by an Arduino MKR WiFi 1010. This microcrontroller generates and distributes a local WiFi network that it uses to receive shape motion commands and processes these commands into actuator signals that it sends to the motors. The communications handling and the actuation control are done asynchronously through an Arduino real-time operating system.

The second system consists of a laptop running MatLab near the test tank. Connected to this laptop is a USB handheld controller that recieves joystick and button input from the AmoeBot pilot. On this computer runs the software that processes joystick input from the handheld controller, chooses the corresponding gait control vector field, sequences the next 100ms of control actions for the AmoeBot from the control flow, and sends this command data at 10Hz to the AmoeBot via the local WiFi network. This computer also recorded the local command and shape data for later experimental visualization.

The third system is a Nikon D7000 DSLR camera. This camera was positioned above the test tank to capture a birds-eye view of the AmoeBot maneuverability. This video was synchronized with the MatLab data by hand through the use of visual flags and handheld controller input. After the experiments, the AmoeBot position and orientation were estimated using visual filtering and the resulting

data was overlaid on the videos.

Two experiments were run to demonstrate the maneuverability of the AmoeBot. In the first, two lasercut posterboard arrows were clamped to the side of the test tank to provide the AmoeBot with position orientation waypoints to navigate between. The experimental setup for this experiment is shown in Fig. 4.13, and the experimental results are shown in Fig. 4.14. Overall, the AmoeBot succesfully navigates between the desired waypoints. It moved rather slowly, with the total navigation time taking approximately 6 minutes and 24 seconds, but was able to visit the waypoints to within acceptable tolerances. At the first waypoint, a small reorientation maneuver was necessary to turn the AmoeBot to point along the desired orientation. This was the only maneuver throughout the experiments that used the turning-optimized gaits. For the rest of both of the experiments, the AmoeBot maneuvered using the forward, reverse, and steering gaits, which was found to be an intuitive way to pilot the vehicle.

In the second experiment, an aluminum frame was inserted into the test tank, holding two lasercut visual markers. The AmoeBot was then driven in a Figure-Eight pattern around the two markers, demonstrating its turning and maneuvering capabilities. This experiment took 9 minutes and 51 seconds to complete. Along the way, the AmoeBot occasionally bumped into the aluminum frame and the side wall of the test tank, but was able to recover and continue navigation after every incident. A side-view of the experimental setup can be seen in Fig. 4.15, and the experimental results can be seen in Fig. 4.16. Notable buildup of angular momentum was seen in this experiment, with the AmoeBot continuing to experi-


Figure 4.13: A side-view of the experimental test-tank for the waypoint navigation experiment. Two white posterboard arrows were clamped to the edges of the test tank, providing position and orientation waypoints to navigate between.

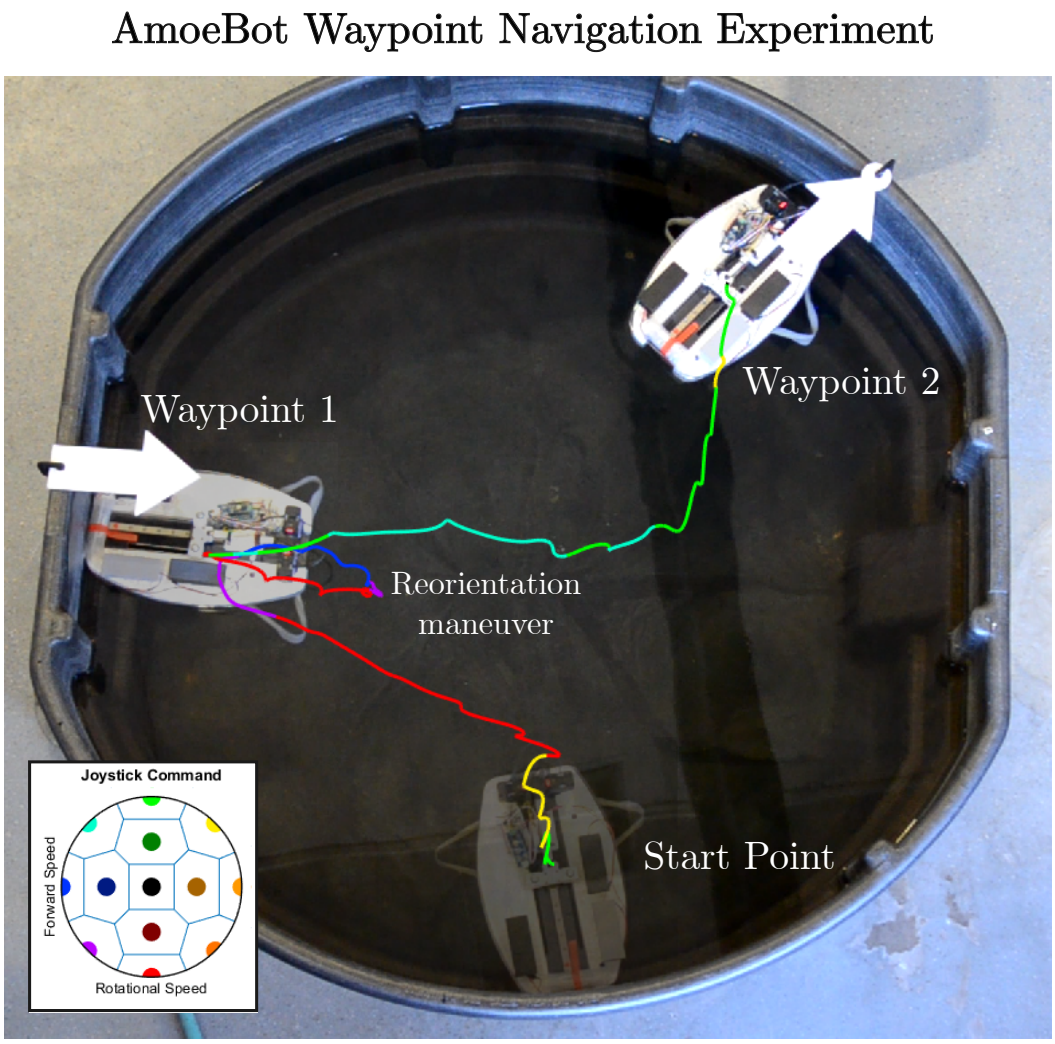


Figure 4.14: View of the AmoeBot navigating between its starting position and the two waypoints set by the lasercut posterboard arrows. The total experiment time from start to finish was 6 minutes and 24 seconds. The color-coded trajectory is an estimate of the AmoeBot center over time, with the color associated to the gait command at that point along the route. At the first waypoint, a small reorientation maneuver was necessary to turn the AmoeBot to point along the desired orientation.

Figure 4.15: A side-view of the experimental test-tank for the figure-eight experiment. An aluminum frame was inserted into the test tank and topped with white visual markers to indicate the structure in the video.

ence orientation drift multiple gait cycles after the execution of the latest turning motion. This highlights the importance of accounting for locomotor momentum during the control process.

These experiments demonstrate that the gait control methodology algorithmically built from constrained experimental data is sufficient to provide navigational control authority for the AmoeBot .

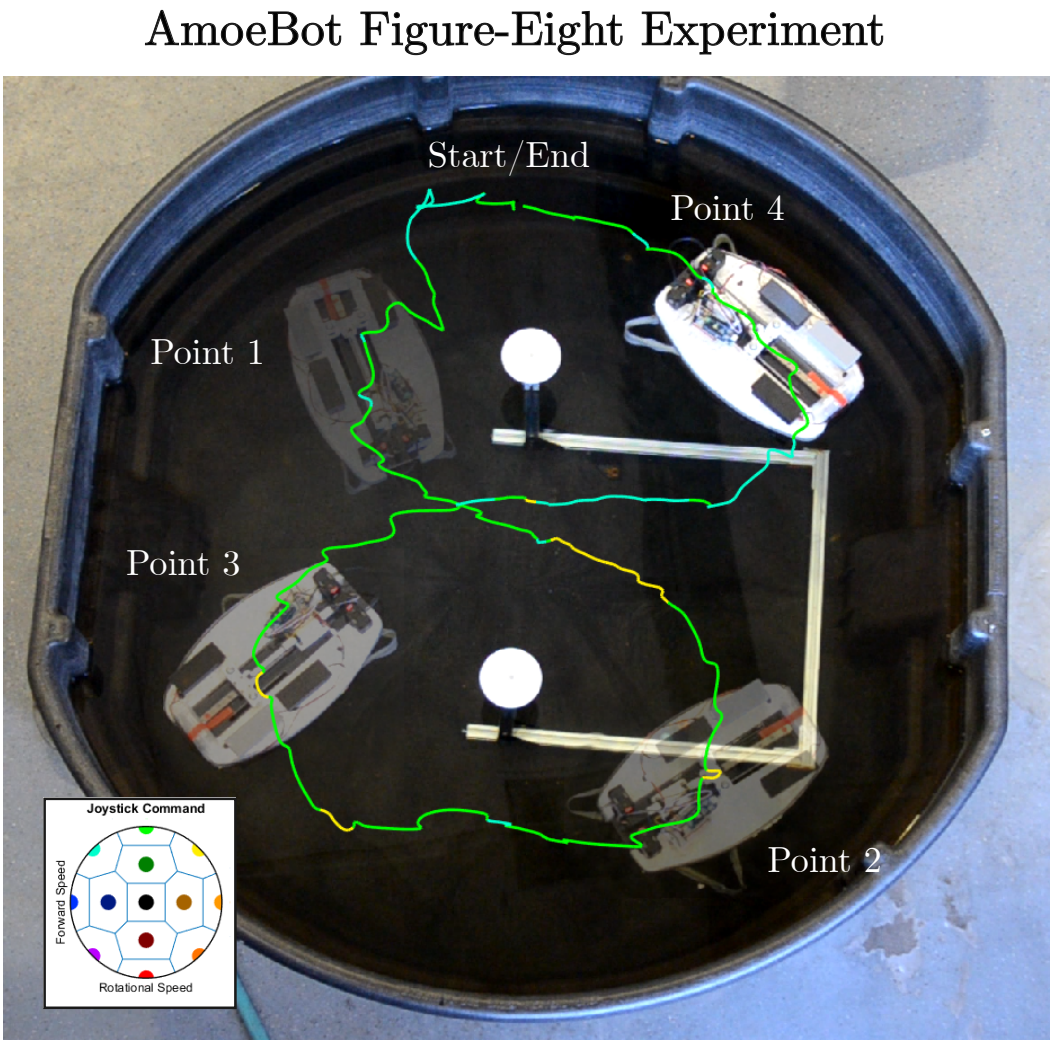


Figure 4.16: View of the AmoeBot performing figure-eight navigation around the aluminum truss. The total experiment time from start to finish was 9 minutes and 51 seconds. The color-coded trajectory is an estimate of the AmoeBot center over time, with the color associated to the gait command at that point along the route. In this experiment, the AmoeBot was controlled using only the forward and steering gaits.

## 4.4 Conclusion

In this chapter, we proposed a methodology for performing system identification on constrained locomotion experiments for aquatic systems and validate this methodology by performing experiments on a floating aquatic robotic platform called the AmoeBot.

We discuss how to use a system's hydrodynamic equations of motion to generate the regression function for experimental data, and demonstrate this process of regression for the dynamic coefficients of the AmoeBot tape fin. Then, we use this identified model to optimize for a suite of gaits useful for AmoeBot locomotion. For desired gait motion, we first optimize kinematically to find a shape profile that produces beneficial net motion, then we optimize dynamically to find a speed profile at which to execute this gait that takes into accound the buildup of swimmer momentum. We then map the input of a joystick controller to this gait library and generate a control law for each gait that converges arbitrary initial shape conditions to the desired optimized limit cycle.

We then demonstrate that this algorithmic process results in gaits useful for AmoeBot locomotion by performing two navigational experiments in a test tank filled with water. In the first experiment, the AmoeBot maneuvers between preset position and orientation waypoints. In the second, the AmoeBot performs a figure-eight maneuver around an aluminum frame submerged in the water. These experiments demonstrate the controllability of the AmoeBot using locomotion optimizations from constrained experimental data.

There is much room for future contributions using this platform and methodology. In this work, we arbitrarily chose one side of the AmoeBot as the 'front' and assumed that time-reversal of gaits would result in opposite motion. A more cohesive investigation of the stability properties of the locomotor would help determine which directions of travel actually produce the most efficient locomotion. There is also room for future controls work on many fronts. Future work aims to generate optimal gaits continuously from controller input rather than by optimizing for discrete locations on the input space. We also believe that it would be possible to use a Model Predictive Control framework to automate AmoeBot navigation, eliminating the need for a pilot.

# Chapter 5: Conclusion

In this thesis, I discussed geometric techniques and provided experimental validation for analysis on a large variety of locomoting systems. In particular, I focused on swimming systems that benefit from passive elasticity in their construction. I feel that this area of research is important, as I strongly believe that we have much to gain from understanding the role of passive elasticity in biological locomotion. If we can exploit passive behavior to improve the locomotive qualities of mobile robots, I feel that we can enable new classes of robotic systems that can provide utility and improve quality of life in areas from scientific exploration to delivery services to social robotics to home maintenance.

My work was broken down into three primary contributions, which correspond roughly to different regions on the Reynolds number spectrum.

In the first contribution, I proposed a model for passive-elastic locomotion in high Reynolds number environments where the physics are driven by inertial effects. I demonstrated a framework for gait optimization that leverages passive shape behavior and showed that continuous flexibility offers some efficiency and speed gains over discrete joints under comparable design conditions. There, I showed that it is possible to optimize gaits and passive-elastic design parameters simultaneously during the design process to enable the most efficient possible passive-elastic locomotion, and also that it is possible to optimize for gaits with

preset sub-optimal passive coefficients in the case that design restrictions prevent passive-elastic tuning. Future work includes expansion of these techniques into a higher number of active and passive shape modes and the inclusion of nonlinear stiffness and damping effects for more realistic simulations.

In the second contribution, I developed a model and simulator for a low Reynolds number analog of the sea salp in which the physics are driven by fluid drag and the robot consists of an array of independent zoid agents. In this work, I showed that excitation of fluid thrust on the zooid agents produces predictable behavior and evolution of the superstructure into trajectories that are stable to certain workspace trajectories and shape-space orbits. To enable future investigations into these findings, I developed a GUI platform to test new salp design directions and to examine robotic salp control policies. I also helped to develop a novel origami thruster driven by twisted-and-coiled actuation that produces thrust on the scale of a biological salp zooid. Future work includes experimental validation of these salp experiments and expansion of the salp model into three-dimensional SE(3) geometric space.

In my final contribution, I proposed a methodology for performing system identification on constrained locomotor experiments. I showed that although adding experimental constraints changes the dynamic interaction of the locomotor with its surrounding environments, it often remains possible to perform regression on the locomotor's dynamic coefficients using experimental data. This regression enables the formulation of a dynamic model that can be used to generate a library of motion behaviors. I validated these results on the AmoeBot locomotion plat-

form, an aquatic robot that swims using novel tape-fin actuators. Because of the peculiar hydrodynamic properties of water, this system lies solidly between the low and high Reynolds number regimes, and neither inertial nor drag effects could be safely ignored. Using constrained experimental data for the AmoeBot, we performed regression on the system's locomotive elements, and used the resulting hydrodynamic model to algorithmically generate a motion library that can be used to pilot the AmoeBot. We validated this technique through deployed navigation experiments, where we demonstrated that our model identification and gait library optimization techniques are sufficient to enable controllability of the AmoeBot in laboratory conditions. Future work on this project includes stability studies on the AmoeBot to determine most efficient directions of locomotion and improvements on this control technique that allow for the generation of gaits that are continuous on the control input space rather than formed by discrete components.

I feel that this work has helped push the boundary on our knowledge of how locomoting systems use passive dynamics to enable beneficial motion. I hope in the future to see similar techniques used to develop and drive robotic systems that improve quality of life, and I hope that this line of inquiry will enable more insight into motion in the biological world. I am left with a sense of optimism for the field, and I believe that one day I will see the natural bounce and spring in my own step mirrored in that of the robots around me. Thank you for reading.

# Bibliography

[1] Andy Abate, Jonathan W. Hurst, and Ross L. Hatton. Mechanical Antagonism in Legged Robots. In *Robotics: Science and Systems XII*. Robotics: Science and Systems Foundation, 2016.

[2] Silas Alben, Charles Witt, T. Vernon Baker, Erik Anderson, and George V. Lauder. Dynamics of freely swimming flexible foils. *Physics of Fluids*, 24(5):051901, 2012.

[3] Panagiotis Bardis and Demetri S. Mathioulakis. Performance evaluation of swim fins under zero translation speed. *International Journal of Sports Science and Coaching*, 6(2):253–268, June 2011.

[4] Capprin Bass, Suresh Ramasamy, and Ross L. Hatton. Characterizing error in noncommutative geometric gait analysis. In *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, May 2022.

[5] D. N. Beal, F. S. Hover, M. S. Triantafyllou, J. C. Liao, and G. V. Lauder. Passive propulsion in vortex wakes. *Journal of Fluid Mechanics*, 549(-1):385, February 2006.

[6] Brian Bittner, Ross L Hatton, and Shai Revzen. Data-driven geometric system identification for shape-underactuated dissipative systems. *Bioinspiration and Biomimetics*, 17(2):026004, January 2022.

[7] Thomas J. Böddeker, Stefan Karpitschka, Christian T. Kreis, Quentin Magdelaine, and Oliver Bäumchen. Dynamic force measurements on swimming chlamydomonas cells using micropipette force sensors. *Journal of The Royal Society Interface*, 17(162):20190580, January 2020.

[8] Q. Bone and E. R. Trueman. Jet propulsion in salps (tunicata: Thaliacea). *Journal of Zoology*, 201(4):481–506, December 1983.

[9] D.R. Brumley, R. Rusconi, K. Son, and R. Stocker. Flagella, flexibility and flow: Physical processes in microbial ecology. *The European Physical Journal Special Topics*, 224(17–18):3119–3140, December 2015.

[10] Alejandro Cabrera and Ross L. Hatton. Optimal control of robotic systems and biased riemannian splines. *ESAIM: Control, Optimisation and Calculus of Variations*, 30:36, 2024.

[11] Joshua M. Caputo and Steven H. Collins. An experimental robotic testbed for accelerated development of ankle prostheses. In *2013 IEEE International Conference on Robotics and Automation*, pages 2645–2650, 2013.

[12] Kyeong Ho Cho, Min Geun Song, Hosang Jung, Jungwoo Park, Hyungpil Moon, Ja Choon Koo, Jae-Do Nam, and Hyouk Ryeol Choi. A robotic finger driven by twisted and coiled polymer actuator. In Yoseph Bar-Cohen and Frédéric Vidal, editors, *Electroactive Polymer Actuators and Devices (EAPAD) 2016*. SPIE, April 2016.

[13] Alejandro Damian-Serrano and Kelly R. Sutherland. A developmental ontology for the colonial architecture of salps. *The Biological Bulletin*, page 000–000, April 2024.

[14] A. De Luca and P. Tomei. *Elastic joints*, page 179–217. Springer London, 1996.

[15] Tony Dear, Blake Buchanan, Rodrigo Abrajan-Guerrero, Scott David Kelly, Matthew Travers, and Howie Choset. Locomotion of a multi-link nonholonomic snake robot with passive joints. *The International Journal of Robotics Research*, 39(5):598–616, January 2020.

[16] Jared Diamond. Transport mechanisms: The biology of the wheel. *Nature*, 302(5909):572–573, April 1983.

[17] BJDOM Franco and Luiz Carlos Sandoval Góes. Failure analysis methods in unmanned aerial vehicle (uav) applications. In *Proceedings of COBEM 2007 19th International Congress of Mechanical Engineering*, page 11, 2007.

[18] Helena Hashemi Farzaneh. Bio-inspired design: the impact of collaboration between engineers and biologists on analogical transfer and ideation. *Research in Engineering Design*, 31(3):299–322, March 2020.

[19] R.L. Hatton and H. Choset. Nonconservativity and noncommutativity in locomotion. *The European Physical Journal Special Topics*, 224(17-18):3141–3174, December 2015.

[20] Ross L. Hatton, Zachary Brock, Shuoqi Chen, Howie Choset, Hossein Faraji, Ruijie Fu, Nathan Justus, and Suresh Ramasamy. The geometry of optimal gaits for inertia-dominated kinematic systems. *IEEE Transactions on Robotics*, 2021.

[21] Ross L Hatton and Howie Choset. Optimizing coordinate choice for locomoting systems. In *2010 IEEE International Conference on Robotics and Automation*, pages 4493–4498. IEEE, 2010.

[22] Ross L Hatton and Howie Choset. Geometric motion planning: The local connection, stokes' theorem, and the importance of coordinate choice. *The International Journal of Robotics Research*, 30(8):988–1014, 2011.

[23] Ross L Hatton and Howie Choset. Geometric swimming at low and high reynolds numbers. *IEEE Transactions on Robotics*, 29(3):615–624, 2013.

[24] Ross L. Hatton, Tony Dear, and Howie Choset. Kinematic cartography and the efficiency of viscous swimming. *IEEE Transactions on Robotics*, 33(3):523–535, June 2017.

[25] Ross L Hatton, Yang Ding, Howie Choset, and Daniel I Goldman. Geometric visualization of self-propulsion in a complex medium. *Physical Review Letters*, 110(7):078101, 2013.

[26] Philip Holmes, Robert J. Full, Dan Koditschek, and John Guckenheimer. The dynamics of legged locomotion: Models, analyses, and challenges. *SIAM Review*, 48(2):207–304, January 2006.

[27] Aaron M. Johnson, G. Clark Haynes, and D. E. Koditschek. Disturbance detection, identification, and recovery by gait transition in legged robots. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5347–5353, 2010.

[28] Nathan Justus and Ross Hatton. Optimal gaits for inertia-dominated swimmers with passive elastic joints. *Phys. Rev. E*, 109:034602, Mar 2024.

[29] E. Kanso, J. E. Marsden, C. W. Rowley, and J. B. Melli-Huber. Locomotion of articulated bodies in a perfect fluid. *Journal of Nonlinear Science*, 15(4):255–289, 2005.

[30] Scott David Kelly, Parthesh Pujari, and Hailong Xiong. Geometric mechanics, dynamics, and control of fishlike swimming in a planar ideal fluid. *IMA Volumes in Mathematics and its Applications*, pages 101–116, 2012.

[31] N. Kidambi and K. W. Wang. Dynamics of kresling origami deployment. *Physical Review E*, 101(6), June 2020.

[32] Michael LaBarbera. Why the wheels won't go. *The American Naturalist*, 121(3):395–408, March 1983.

[33] G. V. Lauder, B. Flammang, and S. Alben. Passive robotic models of propulsion by the bodies and caudal fins of fish. *Integrative and Comparative Biology*, 52(5):576–587, June 2012.

[34] George V. Lauder and Peter G. A. Madden. Fish locomotion: kinematics and hydrodynamics of flexible foil-like fins. *Experiments in Fluids*, 43(5):641–653, July 2007.

[35] K H Low and C W Chong. Parametric study of the swimming performance of a fish robot propelled by a flexible caudal fin. *Bioinspiration and Biomimetics*, 5(4):046002, November 2010.

[36] S. P. Magnusson. Passive properties of human skeletal muscle during stretch maneuvers. *Scandinavian Journal of Medicine and Science in Sports*, 8(2):65–77, April 1998.

[37] Shamil Mamedov and Stanislav Mikhel. Practical aspects of model-based collision detection. *Frontiers in Robotics and AI*, 7, November 2020.

[38] Tad McGeer. Passive dynamic walking. *Int. J. Robotics Res.*, 9(2):62–82, 1990.

[39] Rajat Mittal, Haibo Dong, Meliha Bozkurttas, GeorgeV Lauder, and Peter Madden. Locomotion with flexible propulsors: Ii. computational modeling of pectoral fin swimming in sunfish. *Bioinspiration and Biomimetics*, 1(4):S35–S41, December 2006.

[40] Renato Moraes, M.Anthony Lewis, and AftabE. Patla. Strategies and determinants for selection of alternate foot placement during human locomotion: influence of spatial and temporal constraints. *Experimental Brain Research*, September 2004.

[41] J N Newman. *Marine Hydrodynamics*. The MIT Press. MIT Press, London, England, January 2018.

[42] Benjamin Pawlowski, Jiefeng Sun, Jing Xu, Yingxiang Liu, and Jianguo Zhao. Modeling of soft robots actuated by twisted-and-coiled actuators. *IEEE/ASME Transactions on Mechatronics*, 24(1):5–15, February 2019.

[43] J. Picken and C.T. Crowe. Performance efficiency of swim fins. *Ocean Engineering*, 2(6):251–258, June 1974.

[44] Edward M Purcell. Life at low Reynolds number. *American journal of physics*, 45(1):3–11, 1977.

[45] S. Ramasamy and R. L. Hatton. Geometric gait optimization beyond two dimensions. In *2017 American Control Conference (ACC)*, pages 642–648, 2017.

[46] Suresh Ramasamy and Ross L Hatton. Soap-bubble optimization of gaits. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 1056–1062. IEEE, 2016.

[47] Suresh Ramasamy and Ross L. Hatton. The geometry of optimal gaits for drag-dominated kinematic systems. *IEEE Transactions on Robotics*, 35(4):1014–1033, August 2019.

[48] Suresh Ramasamy and Ross L Hatton. The geometry of optimal gaits for drag-dominated kinematic systems. *IEEE Transactions on Robotics*, 35(4):1014–1033, 2019.

[49] Suresh Ramasamy and Ross L Hatton. Optimal gaits for drag-dominated swimmers with passive elastic joints. *Physical Review E*, 103(3):032605, 2021.

[50] Thomas J. Roberts and Emanuel Azizi. Flexible mechanisms: the diverse roles of biological springs in vertebrate movement. *Journal of Experimental Biology*, 214(3):353–361, February 2011.

[51] S. Samimy, J. C. Mollendorf, and D. R. Pendergast. A theoretical and experimental analysis of diver technique in underwater fin swimming. *Sports Engineering*, 8(1):27–38, March 2005.

[52] J. Sánchez-Rodríguez, F. Celestini, C. Raufaste, and M. Argentina. Proprioceptive mechanism for bioinspired fish swimming. *Physical Review Letters*, 126(23), June 2021.

[53] M. Sfakiotakis, D.M. Lane, and J.B.C. Davies. Review of fish swimming modes for aquatic locomotion. *IEEE Journal of Oceanic Engineering*, 24(2):237–252, April 1999.

[54] Mohammad Sharifzadeh and Daniel M. Aukes. Curvature-induced buckling for flapping-wing vehicles. *IEEE/ASME Transactions on Mechatronics*, 26(1):503–514, February 2021.

[55] Mohammad Sharifzadeh, Yuhao Jiang, and Daniel M. Aukes. Reconfigurable curved beams for selectable swimming gaits in an underwater robot. *IEEE Robotics and Automation Letters*, 6(2):3437–3444, April 2021.

[56] B Sciavicco Siciliano, L Villani, and L Oriollo. G.(2009) robotics: Modeling, planning and control.

[57] Curtis Sparks, Nathan Justus, Ross Hatton, and Nick Gravish. Amoeba-inspired swimming through isoperimetric modulation of body shape. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, October 2022.

[58] Greg J. Stephens, Bethany Johnson-Kerner, William Bialek, and William S. Ryu. Dimensionality and dynamics in the behavior of c. elegans. *PLoS Computational Biology*, 4(4):e1000028, April 2008.

[59] Konstantin Struebig, Behzad Bayat, Peter Eckert, Anouk Looijestijn, Tim C Lueth, and Auke J Ijspeert. Design and development of the efficient anguilliform swimming robot— mar. *Bioinspiration and Biomimetics*, 15(3):035001, March 2020.

[60] Kelly R. Sutherland and Laurence P. Madin. Comparative jet wake structure and swimming performance of salps. *Journal of Experimental Biology*, 213(17):2967–2975, September 2010.

[61] Kelly R. Sutherland and Daniel Weihs. Hydrodynamic advantages of swimming by salp chains. *Journal of The Royal Society Interface*, 14(133):20170298, August 2017.

[62] Skriptyan N.H. Syuhri, David Pickles, Hossein Zare-Behtash, and Andrea Cammarano. Influence of travelling waves on the fluid dynamics of a beam submerged in water. *Journal of Fluids and Structures*, 121:103947, August 2023.

[63] Kentaro Takagi, Takeshi Arakawa, Jun Takeda, Ken Masuya, Kenji Tahara, and Kinji Asaka. Position control of twisted and coiled polymer actuator using a controlled fan for cooling. In Yoseph Bar-Cohen, editor, *Electroactive Polymer Actuators and Devices (EAPAD) 2017*. SPIE, April 2017.

[64] Yuki Takagi, Masato Ishikawa, and Koichi Osuka. A kinematic-dual snake robot: Undulatory mobile robot driven by controllable side-thrust links. *Mechatronics*, 90:102944, April 2023.

[65] Yucheng Tang, Lei Qin, Xiaoning Li, Chee-Meng Chew, and Jian Zhu. A frog-inspired swimming robot based on dielectric elastomer actuators. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, September 2017.

[66] Mark Wielitzka, Alexander Busch, Matthias Dagen, and Tobias Ortmaier. Unscented kalman filter for state and parameter estimation in vehicle dynamics. In *Kalman Filters - Theory for Advanced Applications*. InTech, February 2018.

[67] Chunbing Wu and Wen Zheng. Position and force control of a twisted and coiled polymeric actuator. *IEEE Access*, 8:137226–137234, 2020.

[68] QiDi Wu, ChengJu Liu, JiaQi Zhang, and QiJun Chen. Survey of locomotion control of legged robots inspired by biological concept. *Science in China Series F: Information Sciences*, 52(10):1715–1729, October 2009.

[69] A. Yamano, K. Shimizu, M. Chiba, and H. Ijima. Fluid force identification acting on snake-like robots swimming in viscous fluids. *Journal of Fluids and Structures*, 106:103351, October 2021.

[70] Zhiyuan Yang, Dongsheng Chen, David J. Levine, and Cynthia Sung. Origami-inspired robot that swims via jet propulsion. *IEEE Robotics and Automation Letters*, 6(4):7145–7152, October 2021.