# A HIERARCHY OF MODELS FOR THE CONTROL OF FISH-LIKE LOCOMOTION 

Juan B. Melli

A Dissertation<br>Presented to the Faculty of Princeton University<br>in Candidacy for the Degree<br>of Doctor of Philosophy

Recommended for Acceptance<br>by the Department of<br>Mechanical and Aerospace Engineering

Advisor: C.W. Rowley
(c) Copyright by Juan B. Melli, 2008.

All Rights Reserved

## Abstract

Inspired by the advanced capabilities of fish and other underwater swimmers, we seek a greater understanding of fish-like propulsion and control.

Our study begins by modeling two extremes of fish-like locomotion: a potential flow model, which ignores the effects of viscosity, and a Stokes flow model in which inertial forces are negligible relative to viscous forces. We emphasize the importance of the local form of a mathematical object called the connection, which appears in the equations of motion and relates internal shape changes to body velocities. We demonstrate how the process of designing large-amplitude gaits for systems characterized by Abelian connections can be facilitated by visualizing the curvature of the connection over the shape space. These results are partially extended for a class of non-Abelian connections where the group is the semidirect product of an Abelian group and a vector space.

A third model accounts for the effects of both inertia and viscosity. Although still in potential flow, the effects of viscosity are partially modeled through the shedding of vorticity from sharp trailing edges. Our focus is on the interaction of the swimmer with its own vortex wake. We take a heuristic approach and perform a series of numerical experiments to identify a strategy for producing near-optimal thrust-producing gaits. We implement a phase-locked loop controller to achieve the control objective and demonstrate its effectiveness at generating high thrust-producing gaits.

## Acknowledgements

This work would not have been possible without the support of friends, family and colleagues.

I owe my greatest thanks and gratitude to my advisor Clancy Rowley for his invaluable guidance, support, and encouragement.

I am thankful for the thoughtful feedback and constructive criticism from Naomi Leonard and Lex Smits that helped make this a stronger and more accessible document.

It was a true pleasure collaborating with Eva Kanso, Jerry Marsden, and Dzhelil Rufat on deformable swimmers in potential flow. I thank George Lauder for his insights on fish swimming and tandem foils.

Many thanks to my friends and colleagues for the numerous helpful discussions and suggestions: Melissa Green, Sunil Ahuja, Zhanhua Ma, Miloš Ilak, Peter Norgaard and Steve Brunton.

I gratefully acknowledge the funding sources for this work, including a U.S. Department of Defense NDSEG fellowship and a National Science Foundation Graduate Research Fellowship.

Loving thanks to my wonderful and infinitely patient wife, Stephanie.
This is dedicated to my parents for their constant love and support.
This dissertation carries the number T-3186 in the records of the Department of Mechanical and Aerospace Engineering.

## Contents

Abstract ..... iii
Acknowledgements ..... iv
List of Tables ..... xi
List of Figures ..... xii
1 Introduction ..... 2
1.1 Motivation ..... 2
1.2 Literature Survey ..... 4
1.2.1 Theoretical models ..... 4
1.2.2 Numerical Models ..... 7
1.2.3 Control of fish-like swimmers ..... 10
1.3 Overview of Contributions ..... 10
2 Mathematical Preliminaries ..... 13
2.1 Lie algebras and Lie groups ..... 13
2.2 Rigid motion in the plane ..... 14
2.3 Actions of Lie Groups ..... 16
2.4 Principal bundles, connections and curvature ..... 17
2.5 Semidirect product groups ..... 21
3 Motion in Potential Flow ..... 22
3.1 Potential Flow ..... 24
3.2 The Swimmer ..... 25
3.3 Kinetic Energy ..... 27
3.4 Numerical Method ..... 30
3.5 Equations of Motion ..... 34
3.6 Example Gaits ..... 36
3.6.1 Forward Gait ..... 37
3.6.2 Turning Gait ..... 39
3.7 Coordinate-free Analysis ..... 40
3.8 Summary ..... 43
4 Motion in Stokes Flow ..... 45
4.1 Transforming Stokes Equations ..... 45
4.2 Stokes Flow Algorithm ..... 46
4.3 Numerical Solution ..... 50
4.4 Code Validation ..... 55
4.4.1 Squirming Cylinder ..... 55
4.4.2 Purcell Swimmer ..... 57
4.5 Example Gaits ..... 58
4.5.1 Forward Gait ..... 60
4.5.2 Turning Gait ..... 60
4.6 Summary ..... 62
5 Control of Reversible Systems ..... 63
5.1 Reversible Swimming Systems ..... 63
5.2 Holonomy in semidirect product groups ..... 64
5.3 Curvature of the Connection ..... 67
5.4 Gait generation ..... 68
5.4.1 Gait Generation Example - Potential Flow ..... 72
5.4.2 Gait Generation Example - Stokes Flow ..... 75
5.5 Abelian vs non-Abelian ..... 77
5.6 Summary ..... 78
6 Numerical Method for Potential Flow with Point Vortices ..... 80
6.1 Motivation ..... 81
6.2 Model Overview ..... 83
6.3 Influence Coefficients ..... 87
6.4 Details of Numerical Scheme ..... 90
6.4.1 Wake Modeling ..... 90
6.4.2 Boundary Condition ..... 91
6.4.3 Boundary Velocity ..... 93
6.4.4 Kutta Condition ..... 95
6.4.5 Wake Convection ..... 100
6.4.6 Pressure Coefficient ..... 101
6.4.7 Velocity Potential ..... 105
6.4.8 Diverting Vortices ..... 107
6.5 Numerical Algorithm ..... 108
6.6 Code Validation ..... 111
7 Control Through a Vortex Wake ..... 120
7.1 Development of a Heuristic Control Objective Through Numerical Ex- periments ..... 122
7.1.1 Prior Work: Swimming Through a Vortex Wake ..... 123
7.1.2 Effective Angle of Attack ..... 124
7.1.3 Stagnation Point ..... 127
7.1.4 Numerical Experiments ..... 128
7.1.5 Results ..... 131
7.1.6 Analysis ..... 133
7.2 Literature review: Modeling and Control of Fluid Systems ..... 134
7.3 Control Methodology ..... 136
7.4 Example ..... 139
7.4.1 Results ..... 141
7.4.2 Lock-in Time ..... 144
7.5 Swimming with and without control ..... 146
7.6 Summary ..... 150
8 Conclusions and Future Work ..... 154
A Potential flow code ..... 172
A. 1 driver.m ..... 172
A. 2 rk4.m ..... 173
A. 3 shape_var.m ..... 173
A. 4 connection.m ..... 174
A. 5 getadmass.m ..... 175
A. 6 threebody.m ..... 175
A. 7 influence.m ..... 176
A. 8 admass.m ..... 177
A. 9 adjointinv.m ..... 181
A. 10 adjoint.m ..... 181
A. 11 vel_fun.m ..... 181
B Stokes flow code ..... 183
B. 1 driverStokes.m ..... 183
B. 2 shape_var.m ..... 184
B. 3 StokesConnection.m ..... 185
B. 4 surfacemodes.m ..... 185
B. 5 threelinkbody.m ..... 187
B. 6 influencematrix.m ..... 189
B. 7 stressletinfluence.m ..... 190
B. 8 vel_ful.m ..... 190
C Potential flow + vortex shedding code ..... 191
C. 1 driver.m ..... 191
C. 2 inputs.m ..... 193
C. 3 initialize.m ..... 194
C. 4 prescribedmotion.m ..... 196
C. 5 nacacoords.m ..... 197
C. 6 shape.m ..... 198
C. 7 foilarea.m ..... 198
C. 8 geometry.m ..... 198
C. 9 centroid.m ..... 200
C. 10 momentofinertia.m ..... 200
C. 11 totalinertia.m ..... 200
C. 12 nexttimestepinitialguess.m ..... 201
C. 13 swimmershape.m ..... 201
C. 14 initialguess.m ..... 201
C. 15 bodyvelocity.m ..... 201
C. 16 flowsolution.m ..... 202
C. 17 panelinfluence.m ..... 205
C. 18 wakepanel.m ..... 207
C. 19 computeforces.m ..... 208
C. 20 potential.m ..... 208
C. 21 velatpts.m ..... 211
C. 22 liftdragmoment.m ..... 212
C. 23 computemotion.m ..... 213
C. 24 convergencecheck.m ..... 214
C. 25 convergencecheck1.m ..... 214
C. 26 updatemotionvars.m ..... 214
C. 27 replacewakepanel.m ..... 214
C. 28 stagpoint.m ..... 215
C. 29 angleofattack.m ..... 215
C. 30 controller.m ..... 215
C. 31 convectvortices.m ..... 216
C. 32 ptvortinfluence.m ..... 218
C. 33 expandbody.m ..... 219
C. 34 savedata.m ..... 219
C. 35 plotandsave.m ..... 219
C. 36 adjustvortexpos.m ..... 220
C. 37 vortin.m ..... 221

## List of Tables

6.1 Summary of nomenclature ..... 85
6.2 Influence coefficients ..... 90
6.3 Propulsive efficiency comparison ..... 119
7.1 Experiment parameter values ..... 131
7.2 Results for $\omega_{1}=\omega_{2}=7, A=9^{\circ}$ experiments for a fixed swimmer . ..... 133

## List of Figures

2.1 Schematic of a principal bundle. ..... 18
3.1 Schematic of 3-link articulated swimmer. ..... 26
3.2 Coordinate frames and identification of links for 3-link articulated swimmer ..... 26
3.3 Coordinate system fixed to a panel ..... 32
3.4 Potential flow swimmer: Forward gait example ..... 38
3.5 Potential flow swimmer: Forward gait snapshots ..... 38
3.6 Potential flow swimmer: Turning gait example ..... 40
3.7 Potential flow swimmer: Turning gait snapshots ..... 41
4.1 Snapshots of the Shapere and Wilczek [105] swimmer shape during one cycle ..... 56
4.2 Comparison of code with analytic result from Shapere and Wilczek [105] 57
4.3 Comparison of code with analytic result from Becker, Koehler and Stone [12] ..... 58
4.4 The three-link Purcell swimmer with $\theta_{1}=0.8$ and $\theta_{2}=0.4$. ..... 59
4.5 Stokes flow swimmer: Forward gait example ..... 61
4.6 Stokes flow swimmer: Forward gait snapshots ..... 61
4.7 Stokes flow swimmer: Turning gait example ..... 62
4.8 Stokes flow swimmer: Turning gait snapshots ..... 62
5.1 Curvature components of potential flow swimmer ..... 69
5.2 Curvature components of Stokes flow swimmer ..... 69
$5.3 \quad F_{\omega}$ curvature component for potential and Stokes flow swimmers ..... 70
5.4 Two gaits resulting in same Abelian group holonomy ..... 71
5.5 Zero contours of $F_{\omega}$ and $F_{u}$ curvature components ..... 72
5.6 Potential flow swimmer: Lateral translation gait ..... 73
5.7 Potential flow swimmer: A path in shape space resulting in net rotation of $\frac{\pi}{4}$ radians ..... 74
5.8 Potential flow swimmer: Translation and rotation for a sample gait ..... 74
5.9 Potential flow swimmer: Snapshots of swimmer configuration during turning gait ..... 75
5.10 Stokes flow swimmer: A turning gait ..... 76
5.11 Stokes flow swimmer: Translation and rotation for a turning gait ..... 76
5.12 Stokes flow swimmer: Snapshots of swimmer configuration during turn- ing gait ..... 77
5.13 Various gait starting points ..... 79
5.14 Translation and rotation for a family of gaits ..... 79
6.1 Drucker and Lauder [26] experiment ..... 82
6.2 Schematic of two-foil swimmer system ..... 84
6.3 Flowchart of numerical algorithm for advancing one time step ..... 88
6.4 Point vortex crossing integration path ..... 106
6.5 Foil with surrounding region ..... 108
6.6 Schematic of layout and coordinate system for Tuncer and Platzer [119] validation case ..... 112
6.7 Layout and integration paths not crossing wake for two foil code vali- dation ..... 112
6.8 Integration path not crossing vortex wake: Unsteady drag coefficient vs time compared to the potential flow solver of Tuncer and Platzer [119]113
6.9 Integration path not crossing vortex wake: Unsteady lift coefficient vs time compared to the potential flow solver of Tuncer and Platzer [119] 114
6.10 Layout and integration path crossing wake for two foil code validation 114
6.11 Integration path crossing vortex wake: Unsteady drag coefficient vs time compared to the potential flow solver of Tuncer and Platzer [119] 115
6.12 Integration path crossing vortex wake: Unsteady lift coefficient vs time
compared to the potential flow solver of Tuncer and Platzer [119]
6.13 Layout, integration paths and wake vortex distribution at $t=10 \frac{c}{U_{\infty}}$ for two foil code validation against Tuncer and Platzer [119] Navier Stokes solver ..... 116
6.14 Code validation: Unsteady drag coefficient vs time ..... 117
6.15 Code validation: Unsteady lift coefficient vs time ..... 118
7.1 Schematic of two foil swimmer ..... 121
7.2 Three components of effective angle of attack ..... 126
7.3 Normalized effective angle of attack and stagnation point position along the surface of the leading foil versus time ..... 129
7.4 Normalized effective angle of attack and leading edge stagnation point position along the surface of the trailing foil versus time for a series of experiments ..... 132
7.5 Fixed velocity: Thrust coefficient versus the phase difference between $\alpha_{\text {eff }}$ and $x_{\text {stag }}$ at steady state for a series of experiments ..... 134
7.6 Free-swimming: Thrust coefficient versus the phase difference between $\alpha_{\text {eff }}$ and $x_{\text {stag }}$ at steady state for a series of experiments where the swimmer is free-swimming in the $x$ direction ..... 135
7.7 Classical phase-locked loop block diagram ..... 137
7.8 Simplified phase-locked loop block diagram
7.9 With control (leading foil frequency, $\omega_{1}=7$ ): Normalized effective angle of attack, $\alpha_{\text {eff }}$, and the stagnation point position along the surface, $x_{\text {stag }}$, of the trailing foil vs non-dimensionalized time for a selfpropelling fish-like swimmer using a phase-locked loop controller142
7.10 Non-dimensional swimmer speed relative to the freestream versus time ..... 143
7.11 Non-dimensional steady-state speed vs. $\phi_{2}$ for constant phase gaits . ..... 144
7.12 With control (leading foil frequency, $\omega_{1}=4$ ): Normalized effective angle of attack, $\alpha_{\text {eff }}$, and the stagnation point position along the surface, $x_{\text {stag }}$, of the trailing foil vs non-dimensionalized time for a selfpropelling fish-like swimmer using a phase-locked loop controller . . . 145
7.13 Non-dimensional speed vs time for controlled and uncontrolled gaits . ..... 146
7.14 Leading foil frequency, $\omega_{1}(t)$ vs non-dimensionalized time ..... 147
7.15 Swimmer performance with and without control ..... 1487.16 Normalized effective angle of attack and stagnation point positionalong the surface vs time for the cases (a) with control and (b) withoutcontrol corresponding to the results in Figure 7.15149
7.17 Snapshot of swimmer and vortex wake ..... 150
7.18 Relative speed of swimmer for uncontrolled case from Figure 7.15 (red,solid) versus non-dimensionalized time along with the low-pass filteredproduct of $x_{\text {int }}$ and a $\pi / 2$ phase-shifted version of $\alpha_{\text {eff }}$ in Figure 7.16(b)(black, dashed)1517.19 Normalized stagnation point position along the surface versus effectiveangle of attack for the cases (a) with control and (b) without control152
"I know the human being and fish can coexist peacefully."
— George W. Bush, September 29, 2000 [3]

## Chapter 1

## Introduction

### 1.1 Motivation

Fishes and other swimmers have benefitted from the guiding hand of millions of years of evolution which has increasingly optimized their morphology and physiology to survive in their natural environments. Their remarkable swimming characteristics often remain unmatched by man's most advanced engineering accomplishments.

In recent years there has been renewed interest in underwater exploration and the use of unmanned underwater vehicles (UUV) and particularly autonomous underwater vehicles (AUV) to explore marine environments. Improved AUV maneuverability may benefit the study of small-scale oceanographic processes, while improved efficiency could benefit the study of large-scale processes. Enhanced stealth capabilities could benefit military or marine wildlife exploration. Furthermore, an understanding of small-scale, highly viscous motion could play a key role in micro or nanoscale drug delivery systems.

One area where biological swimmers outperform most underwater vehicles is in maneuverability. Fish, sea lions, and cetaceans have minimum turning radii as small as $5.5 \%, 9 \%$ and $11 \%$ of their body length, respectively [33], largely due to their
more flexible bodies. Even tuna - a thunniform-type swimmer with a more rigid forebody and lunate tail - has a turning radius of $47 \%$ of its body length [16]. In contrast, submarines have minimum turning radii of $200-300 \%$ of their body length [79]. An improved understanding of fish-like locomotion may lead to more maneuverable underwater vehicles.

The exploration of large-scale oceanographic processes requires sampling vast areas and may benefit from more efficient AUV propulsion systems than the more standard propeller-driven, fixed hull designs. Bottlenose dolphins, for example, are estimated to achieve a propulsive efficiency of about $81 \%$ during steady swimming [32]. Likewise, California sea lions (80\%), harp seals and ringed seals (85\%), and rainbow trout and sockeye salmon ( $70 \%-90 \%$ ) achieve similarly impressive propulsive efficiency values [29, 31, 123].

Another challenge often faced by fixed hull propeller-driven systems is the tradeoff between thrust and noise due to cavitation [131]. Also, glider-based AUVs, which benefit from high efficiency and stealth, suffer from low maneuverability. By contrast, the manta ray's low frequency, large amplitude winged motion is stealthy, maneuverable and thought to be highly efficient [17].

It is in this context that we pursue a deeper understanding of the locomotion of swimmers. We seek to develop a series of models of swimmers that capture the critical dynamical elements of the body-fluid system yet are tractable enough to apply mathematical tools for analyzing robotic locomotion. Although we draw inspiration from biological systems, we do so with the understanding that nature imposes different constraints such as the need to reproduce, feed and survive which do not exist for engineering applications. Thus it is possible and arguably likely that man-made engineering applications could out-perform their biologically-inspired counterparts.

### 1.2 Literature Survey

Humans have been fascinated by the swimming capabilities of animals for thousands of years. Around 343 BC, the Greek naturalist Aristotle was the first scientist to study and document fish, whales, porpoises and dolphins in Historia Animalium [117]:

As a general rule the larger fishes catch the smaller ones in their mouths whilst swimming straight after them in the ordinary position; but the selachians, the dolphin, and all the cetacea must first turn over on their backs, as their mouths are placed down below; this allows a fair chance of escape to the smaller fishes, and, indeed, if it were not so, there would be very few of the little fishes left, for the speed and voracity of the dolphin is something marvellous.

Ever since, scientists have attempted to understand and explain the propulsion of fish-like swimmers. Here we provide a brief survey of relevant progress.

### 1.2.1 Theoretical models

The earliest attempts [48] at understanding fish hydrodynamics was in estimating drag by assuming that self-propelling fish experienced the same drag as that of a similar size rigid body being towed at the same speed. British zoologist James Gray [45] made the same assumption to estimate the drag on dolphins and concluded that the power density of a dolphin's muscle could not be large enough to provide the power needed to overcome the drag force. This seeming contradiction became known as 'Gray's paradox' and has been the subject of decades of debate.

Among the first attempts at modeling fish-like swimming was the work of Sir G. I. Taylor [111], who represented an anguilliform (eel-like) swimmer as a flexible, constant cross-section cylinder with uniform wavelength and amplitude waves traveling down the length of the body. Lighthill [70] considered a similar system and
applied inviscid slender body theory to model a "slender fish" and determine transverse undulatory motions which result in a high propulsive efficiency. Though this work required a perturbative analysis which limited the scope to small-amplitude motions, Lighthill [71] later extended the work to allow for arbitrarily large amplitude deformations.

Wu [128] used linearized inviscid flow theory to study the two dimensional potential flow about a spatially variable-amplitude, sinusoidally waving fixed plate and found analytical expressions for the thrust and propulsive efficiency. Wu [129] expanded upon this work to allow for the plate to travel at a prescribed, variable, unsteady forward speed.

The aerodynamic community's work on unsteady flow over moving wings has proven invaluable to the study of fish-like (and winged) locomotion. The groundwork was laid by Prandtl [97], Birnbaum [14], Wagner [122], Glauert [43], Küssner [62, 63], Theodorsen [116], Kármán [56], Sears [104] and others.

Due to the complexity of accurately modeling the rich hydrodynamics of fish-like locomotion, some efforts have been made to understand considerably simplified models of swimmers as a first approximation. While the motion of most swimmers is governed by both inertial and viscous effects, when the Reynolds number is large, viscous effects are minimized and can often be reasonably modeled through the generation of vorticity by a Kutta condition as in some of the previously noted work (however this will not give the viscous drag on the swimmer). Still, the introduction of vorticity significantly increases the complexity of the model, and it is instructive to understand the motion of a swimmer completely due to inertial forces. Indeed, even without the benefit of viscosity, it is possible for a swimmer to propel through a fluid. This was first demonstrated by Benjamin and Ellis [13] and Saffman [101]. Later, Galper and Miloh studied the motion of general deformable shapes in potential flow $[84,36,37]$.

Our study is most closely aligned with a recent branch of work from the nonlinear control and robotics community. This includes the work of Kelly [59] who used perturbation theory to determine an analytical expression for a mathematical object called the mechanical connection, which relates shape changes to body velocity, for a twodimensional, nearly-circular amoeba-like deformable swimmer in an incompressible, inviscid fluid. The result is valid for small-amplitude shape deformations.

Mason and Burdick [81] formulated the motion of a deformable swimmer in potential flow with tools from geometric mechanics. The authors provided an explicit formula for the mechanical connection as a function of the velocity potential, which they expressed as a sum of generalized Kirchhoff potentials $[60,64]$ corresponding to rigid body motions and internal shape changes. By using the Magnus [74] expansion for a Lie algebra-valued function, Mason and Burdick found an approximate expression for the displacement of the body resulting from small-amplitude periodic shape changes. The authors observed that "the curvature [of the connection] is an excellent measure of the effectiveness of the swimmer."

Radford [99] modeled a fish as three articulated rigid links in an inviscid, incompressible fluid. The elliptical links were assumed to be mechanically coupled but hydrodynamically decoupled, meaning that the added masses for each of the links are the same as that of an isolated link in an infinite fluid. In other words, under this assumption, the individual links did not "feel" any added resistance to movement due to the presence of the other links. This assumption allowed him to derive an analytical expression for the local connection as a function of the swimmer shape (defined by the joint angles) and the link inertia terms.

On the other end of the spectrum from potential flow is Stokes flow where viscous forces dominate over inertial forces. Taylor was the first to model the self-propulsion of a two-dimensional swimmer [110] - a sheet with a traveling wave - as well as a flexible, circular cross-section cylinder [112] in Stokes flow. Meanwhile, Lighthill [69]
found that the velocity of deformable, nearly-spherical bodies in Stokes flow is at most proportional to the square of the deformation amplitude.

Unlike most realistic flows in which both friction and inertia play a role, the "Scallop theorem" specifies that non-reciprocal motion is needed to swim in Stokes flow. Purcell [98] introduced a hypothetical three-link, two-hinged animal which is the simplest creature capable of achieving net motion in Stokes flow due to non-reciprocal motion in an unbounded domain. Much of the work on locomotion in Stokes flow has focused on the so-called Purcell swimmer [12, 24, 109].

Shapere and Wilczek [105] described a method for determining the connection for a deformable, two-dimensional Stokes swimmer by solving a linear boundary value problem [105]. Kelly derived this connection for a Stokesian swimmer from a dissipation function [59]. Various authors have formulated the equations for Stokes flow as boundary integral equations [49, 96, 95], and we follow a similar procedure in Chapter 4 of this thesis to model a deformable swimmer in Stokes flow.

### 1.2.2 Numerical Models

The advent and advances in digital computer technology brought about the next major wave of progress in modeling unsteady fluid effects. Hess and Smith [47] developed a panel method for computing the steady incompressible, potential flow about an arbitrary two-dimensional body. In this method, the body is discretized into panels with a distribution of source and vorticity singularities which are determined as the solution of a set of linear equations. Although the constant strength source distribution varied from panel to panel, the vorticity density distribution was the same on all panels. By solving the steady problem of Hess and Smith at each time step and prescribing a Kutta condition at the trailing edge where equal magnitude and direction of the velocity was specified, Giesing [39, 40] developed a similar panel method for unsteady potential flow about a single body with a non-linearly deforming wake.

Giesing [41, 42] extended the method to handle the arbitrary motion and unsteady flow due to the interaction between two arbitrary bodies. Basu and Hancock [10] improved upon Giesing's approach for the unsteady potential flow problem and introduced a Kutta condition specifying equal pressures along the top and bottom of the trailing edge. Djojodihardjo [25] developed a method to solve the three-dimensional unsteady potential flow past an arbitrarily-shaped wing undergoing arbitrary motion.

By employing a linear rather than constant distribution of vorticity over the panels, Vezza and Galbraith [120] modified the Basu and Hancock [10] method to require the solution of only a set of linear equations, and eliminated the need to solve a quadratic equation. Teng [115] and Pang [91] built upon the work of Basu and Hancock [10] to develop unsteady two-dimensional panel methods for one and two bodies, respectively.

Two-dimensional unsteady panel methods have been applied and validated extensively [53, 51, 119, 52]. Galls and Rediniotis [35] applied an unsteady two-dimensional panel method to a deformable hydrofoil body along with a trained neural network to autonomously navigate along a desired path. The resulting fluid-body coupling was weak as the force was computed at the end of each time step to advance the motion of the swimmer. Recently, three-dimensional unsteady panel methods have been used to model the hydrodynamics of fish-like swimming [127, 9, 20, 130] though in all these cases, the motion of the swimmer through the fluid was prescribed, not determined as part of the solution.

In these panel methods, singularities are continuously shed into the flow and need to be included in the computation at the next time step. Hence, the size of the state and computation time increases at each time step. One attempt [126] at improving the computation time adopts a precorrected fast Fourier transform (FFT) accelerated iterative integral solver as well as a Fast Multipole Tree algorithm.

In addition to the relatively simple inviscid panel methods, various other numerical
models have been developed to more accurately account for the effect of the fluid viscosity. Williams et al. [125] employed a two-dimensional Navier-Stokes model to study the hydrodynamics about a lamprey by prescribing both the body geometry and motion through the fluid. Liu $[72,73]$ solved for the flow about an undulating tadpole by solving the Navier-Stokes equations at a Reynolds number of 7200 in two and three dimensions. Unlike most prior work which prescribed both the shape change and overall motion through the fluid, Carling et al. [19] solved the self-propelling motion of an anguilliform swimmer by coupling the Navier-Stokes equations with Newton's equations of motion. Similarly, Leroyer and Visonneau [67] numerically solved the Reynolds-Averaged-Navier-Stokes equations and computed the self-propelled motion of a flexible fish-like body.

Eldredge et al. [28] developed a grid-independent viscous vortex particle method which Eldredge [27] used to study the self-propelling motion of a three-link swimmer at moderate Reynolds numbers.

Inspired by the experimental work of Drucker and Lauder on bluegill sunfish [26] and their hypothesis that the presence of a dorsal fin fore of the caudal fin contributed to increased hydrodynamic performance, Akhtar et al. [2] employed an immersed boundary Navier-Stokes solver to model the hydrodynamic interaction between tandem pitching and heaving plates at a Reynolds number of about 600. They found that the presence of the second body downstream of the leading plate enhanced the thrust coefficient and propulsive efficiency of the system by as much as $107 \%$ and $52 \%$, respectively, compared to the case of a single pitching and heaving plate. They further determined that performance improvement is sensitive to the phase difference between the motion of the leading and trailing plate. The mechanism for the thrust enhancement relies on the vortices shed by the upstream plate interacting in such a way as to increase the effective angle of attack on the downstream plate and creating a leading edge stall vortex which increases thrust due to the lower pressure.

### 1.2.3 Control of fish-like swimmers

There have been various attempts to control and optimize the motion of swimmers and underwater vehicles. Radford [87, 99] proved local controllability of a three-link potential flow swimmer by showing that the curvature of the connection and its covariant derivatives span the Lie algebra corresponding to local body velocities. This was a generalization of prior work by Leonard and Krishnaprasad [66] in which the local connection form was constant. Additionally, Radford presented a gait to generate forward motion in the fish. Kanso et al. [55] accurately numerically computed the mechanical connection for a three-link swimmer that accounted for the hydrodynamically coupled links and presented gaits for forward and turning motions. By applying nonlinear optimization techniques, Kanso et al. [54] found locally optimal gaits for the three-link swimmer. Likewise, Tam et al. [109] found optimal gaits for Purcell's swimmer in terms of speed and efficiency.

The problem of motion planning for robotic vehicles has received considerable recent attention [87, 100, 81, 86, 90, 22, 80]. Underwater gliders - a class of AUVs - have been modeled for the purpose of applying feedback control [65, 44]. The RoboTuna - a flexible hull swimmer designed to mimic the motion of a tuna - has been studied to better understand how vorticity-control is used in fish-like locomotion $[118,8,114,113]$. A second-generation underwater vehicle with the shape of the yellowfin tuna known as the Draper Laboratory Vorticity Control UUV (VCUUV) is fully autonomous and has been shown to exhibit maneuverability comparable to live tuna $[6,7,5]$.

### 1.3 Overview of Contributions

The main contributions of this thesis are as follows:

- We developed hydrodynamically-accurate numerical models of two-dimensional deformable swimmers in potential and Stokes flow (Chapters 3 and 4).
- We presented numerical computations of the components of the curvature of the mechanical connection for such swimmers and a theory for developing gaits to achieve motion in certain corresponding Abelian subgroup directions (Chapter 5).
- We made several extensions to existing models for a self-propelling unsteady potential flow swimmer with vorticity shedding (§6.4.7 and 6.4.8).
- We conducted an investigation of a mechanism for improving swimming through a nearly periodic vortex wake (§7.1).
- We developed and implemented a feedback controller to nearly optimize thrust for a swimmer traveling through a nearly-periodic vortex wake ( $\S 7.3,7.4$ and 7.5).

In Chapter 2 we present some of the mathematical tools that are used throughout this thesis.

We begin by considering two considerably simplified cases of swimming where (1) viscous forces and (2) inertial forces can be ignored. Chapter 3 concerns the motion of a deformable swimmer in potential flow in an inviscid fluid. The mechanical connection which relates internal shape changes to overall group motion can be expressed simply as a function of the geometry of the swimmer. Added masses which depend only on the system geometry account for all of the effects of the fluid. The model and numerical code for the potential flow swimmer is based on collaborative work with Eva Kanso et al. [55], though the control and gait design techniques for such a swimmer are original contributions. Likewise, in Chapter 4, we present a model for a deformable swimmer in Stokes flow. Again, the connection can be expressed as a function of the geometry of the swimmer.

Both of these systems are completely time-reversible in the sense that if their internal shape change is exactly reversed, the swimmer will return to its original position and orientation. Chapter 5 concerns the control of these types of systems, and is an original contribution adapted from and expanding upon previously published work [82]. If the group of motions is the semidirect product of an Abelian group and a vector space (such as is the case for the Special Euclidean group which corresponds to rigid motions in the plane, $S E(2)$ ), the equations of motion for the Abelian subgroup decouple from the other components. In this case, we show how a plot of the components of the curvature of the connection is useful for developing gaits by inspection.

In Chapter 6 we introduce a more realistic fish-like potential flow swimmer which models one effect of viscosity by shedding vorticity at sharp edges. We also employ a very simple viscous drag model in which the drag is proportional to the square of the relative swimmer speed. Inspired by real fish which effectively use vorticity in the flow to enhance their swimming, we build upon well-known numerical models for swimming in potential flow with vorticity shedding to study a self-propelling swimmer that interacts with vorticity shed from itself.

We use the model developed in Chapter 6 to conduct a series of numerical experiments in Chapter 7. These experiments lead to a heuristic control objective which optimizes steady-state thrust for a swimmer moving through a nearly-periodic vortex wake. A phase-locked loop controller is designed and implemented to achieve this objective.

We conclude with suggestions for future work in Chapter 8.

## Chapter 2

## Mathematical Preliminaries

In this chapter we introduce some relevant mathematical concepts for completeness, particularly in the areas of differential geometry, geometric mechanics and Lie group theory. It is assumed that the reader is familiar with manifolds and multiplication on these spaces. These concepts are particularly relevant to $\S 3.7$ and Chapter 5 . The interested reader may consider various additional references [78, 1, 77, 75] for a full exposition on these and other related topics.

### 2.1 Lie algebras and Lie groups

A Lie algebra is a vector space $\mathfrak{g}$ on a manifold $M$ with a binary operation, the Lie bracket $[\cdot, \cdot]: \mathfrak{g} \times \mathfrak{g} \rightarrow \mathfrak{g}$ such that for all $x, y, z \in \mathfrak{g}$ and $a, b \in \mathbb{R}$,

- $[\cdot, \cdot]$ is bilinear: $[a x+b y, z]=a[x, z]+b[y, z]$
- $[\cdot, \cdot]$ is skew-symmetric: $[x, y]=-[y, x]$
- $[\cdot, \cdot]$ satisfies the Jacobi identity: $[x,[y, z]]+[y,[z, x]]+[z,[x, y]]=0$.

A group $(G, \cdot)$ is a set $G$ with a binary operation $\cdot$ satisfying:

- (closure) $a \cdot b \in G \quad \forall a, b \in G$
- (associativity) $a \cdot(b \cdot c)=(a \cdot b) \cdot c \quad \forall a, b \in G$
- (identity) $G$ has an identity element $e$ such that $a \cdot e=e \cdot a=a \forall a \in G$
- (inverse) for each $a \in G$, there exists $b$ such that $a \cdot b=b \cdot a=e$

A Lie group $G$ is a group that is also a smooth manifold such that group multiplication $\mu: G \times G \rightarrow G ; \mu:(g, h) \mapsto g h$ is a $C^{\infty}$ smooth operation for all $g, h \in G$. An Abelian group (also known as a commutative group) is a group whose operation is commutative such that $g h=h g$ for all $g, h \in G$. We denote the space of vectors tangent to G at a point $p$ as $T_{p} G$ and $e$ as the identity element of $G$. The Lie algebra of $G$, which we denote by $\mathfrak{g}$, is the tangent space to $G$ at the identity, $T_{e} G$, together with the commutator $[\cdot, \cdot]$.

Left and right translation by $g$ are represented by the maps $L_{g}: G \rightarrow G ; L_{g}: h \mapsto g h$ and $R_{g}: G \rightarrow G ; R_{g}: h \mapsto h g$, respectively. If $v \in T_{h} G$, we may write $g v \in T_{h g} G$ as shorthand for $T_{h} L_{g} v$.

For $\xi \in \mathfrak{g}$, the equation

$$
\begin{equation*}
\dot{g}=T_{e} L_{g} \xi, \quad g(0)=e \tag{2.1}
\end{equation*}
$$

has a unique solution $g_{\xi}(t) \in G \forall t[46,59]$. The exponential map $\exp : \mathfrak{g} \rightarrow G$ maps elements of the Lie algebra to the Lie group and is defined by $\exp \xi=g_{\xi}(1)$.

### 2.2 Rigid motion in the plane

Here we discuss the concepts introduced in the previous section in the context of a planar swimmer.

The general linear group $G L(n)$ is the set of all non-singular $n \times n$ matrices where the group operation is ordinary matrix multiplication. The orthogonal group $O(n)$ is the set of all matrices $A \in G L(n)$ such that $A A^{T}=e$. The special orthogonal group $S O(n)$, also known as the set of $n$-dimensional rotation matrices, is the set
of all matrices $A \in O(n)$ where $\operatorname{det} A=+1$. Finally, the special Euclidean group $S E(n)$ is the set of all rotations and translations in $\mathbb{R}^{n}$ which can be represented by $(n+1) \times(n+1)$ matrices of the form

$$
\left\{\left.\left(\begin{array}{cc}
R & r  \tag{2.2}\\
0 & 1
\end{array}\right) \right\rvert\, R \in S O(n) \text { and } r \in \mathbb{R}^{n}\right\}
$$

For planar motion, $n=2$ and we let $r=(x, y)^{\mathrm{T}}$ and $R=\left(\begin{array}{cc}\cos (\beta) & -\sin (\beta) \\ \sin (\beta) & \cos (\beta)\end{array}\right)$, where $\beta \in S^{1}$ physically represents the angle and $x$ and $y$ the coordinates of the swimmer relative to an inertial frame. Expressed in matrix form, the group operation is the ordinary matrix product. One can readily show that $S O(2)$, the rotation group in the plane, is commutative, while $S E(2)$, the special Euclidian group, is not. We sometimes also express elements of $S E(2)$ as triplets, $g=(x, y, \beta)$.

Let $h(t)$ be a smooth curve in $S E(2)$ parameterized by $t$, and let $\dot{h}=\frac{d h}{d t}$. Then elements of $\mathfrak{s e}(2)$, the Lie algebra of the Lie group $S E(2)$, are of the form $\xi=h^{-1} \dot{h}$. Physically, elements of the Lie algebra correspond to the velocity of the swimmer in the body frame and can be expressed in matrix form as

$$
h^{-1} \dot{h}=\left(\begin{array}{ccc}
C \beta & S \beta & -x C \beta-y S \beta  \tag{2.3}\\
-S \beta & C \beta & x S \beta-y C \beta \\
0 & 0 & 1
\end{array}\right)\left(\begin{array}{ccc}
-\dot{\beta} S \beta & -\dot{\beta} C \beta & \dot{x} \\
\dot{\beta} C \beta & -\dot{\beta} S \beta & \dot{y} \\
0 & 0 & 0
\end{array}\right)=\left(\begin{array}{ccc}
0 & \omega & u \\
-\omega & 0 & v \\
0 & 0 & 0
\end{array}\right)
$$

where $C \beta=\cos (\beta), S \beta=\sin (\beta)$ and $u, v, \omega \in \mathbb{R}$. Here, $u=\dot{x} \cos (\beta)+\dot{y} \sin (\beta)$ and $v=-\dot{x} \sin (\beta)+\dot{y} \cos (\beta)$ correspond to the swimmer's forward and lateral body velocities, respectively, while $\omega=\dot{\beta}$ is the rotational velocity. We sometimes express elements of $\mathfrak{s e}(2)$ as triplets, $\xi=(u, v, \omega)$.

For $S E(2)$, the exponential map it is given by: $\exp (u, v, \omega)=(x, y, \beta)$ where

$$
\begin{align*}
\beta & =\omega  \tag{2.4}\\
(x, y) & = \begin{cases}(u, v) & \omega=0 \\
\frac{1}{\omega}(u \sin \omega+v(1-\cos \omega), u(\cos \omega-1)+v \sin \omega) & \omega \neq 0\end{cases} \tag{2.5}
\end{align*}
$$

The inverse of the exponential map is the logarithm map and for $\mathrm{SE}(2)$ is given by $\log (x, y, \beta)=(u, v, \omega)$ where

$$
\begin{align*}
\omega & =\beta  \tag{2.6}\\
(u, v) & = \begin{cases}(x, y) & \beta=0 \\
\frac{\beta}{2}\left(-x \frac{\sin \beta}{\cos \beta-1}-y, x+y \frac{\sin \beta}{\cos \beta-1}\right) & \beta \neq 0\end{cases} \tag{2.7}
\end{align*}
$$

### 2.3 Actions of Lie Groups

The left translation induces a left action of $G$ on a manifold $Q$, which is a smooth mapping $\Phi: G \times Q \rightarrow Q$ such that:

1. $\Phi(e, q)=q$ for all $q \in Q$
2. $\Phi(g, \Phi(h, q))=\Phi\left(L_{g} h, q\right)$ for all $g, h \in G$ and $q \in Q$.

We may sometimes alternatively write $\Phi(g, q)$ as $\Phi_{g}(q)$. The conjugation map $\Phi^{C}$ : $G \times G \rightarrow G$ is defined as $\Phi_{g}^{C}(h)=g h g^{-1}$. Conjugation can be thought of as a generalization of a change of coordinates from one reference frame to another.

The adjoint action of $G$ on $\mathfrak{g}$ can be determined by differentiating the conjugation map at the identity and is defined as

$$
\begin{equation*}
\operatorname{Ad}: G \times T_{e} G \rightarrow T_{e} G ; \quad \text { Ad }:(g, \xi) \mapsto T_{g} R_{g^{-1}} T_{e} L_{g} \xi \tag{2.8}
\end{equation*}
$$

The infinitesimal generator of $\Phi$ corresponding to $\xi \in \mathfrak{g}$ is the vector field $M$ given by

$$
\begin{equation*}
\xi_{M}(x)=\left.\frac{\mathrm{d}}{\mathrm{dt}}\right|_{t=0} \Phi(\exp (t \xi), x) \tag{2.9}
\end{equation*}
$$

### 2.4 Principal bundles, connections and curvature

Let $G$ be a Lie group and $Q$ a manifold with base space $\mathcal{S}$ and structure group $G$. In our analysis we typically consider the motion of a three-link, two hinge swimmer, where $G=S E(2)$ represents the orientation and position of the swimmer in the plane and $\mathcal{S}=T^{2}$ is the shape of the swimmer parameterized by the two joint angles such that the configuration space is $Q=S E(2) \times T^{2}$. In this section we introduce the concept of fiber bundles which are useful in describing locomotion, where the base space describes the internal shape of the fish-like swimmer, and the group describes the overall position and orientation.

A (left) principal fiber bundle (refer to Figure 2.1) is a manifold $Q$ endowed with a (left) action such that

1. $\mathcal{S}=Q / G$;
2. The natural projection $\pi: Q \rightarrow \mathcal{S}$ is differentiable;
3. $Q=\mathcal{S} \times G$ locally.

Fibers are the sets $\pi^{-1}(s) \subset Q$ for $s \in \mathcal{S}$, and the point $q \in Q$ lies in the fiber over $\pi(q) \in \mathcal{S}$. The tangent space at a point $q$ on the fiber is denoted $T_{q} Q$. A vector


Figure 2.1: Schematic of a principal bundle.
$v_{q} \in T_{q} Q$ tangent to the fiber through $q$ is said to be vertical. The space of all such vertical vectors is $V_{q} Q$.

A connection on the principal bundle $Q$ is an assignment of a horizontal subspace, $H_{q} Q$, to $V_{q} Q \subset T_{q} Q$ for each point $q \in Q$ such that $H_{q} Q$ depends smoothly on $q$ and

1. $H_{h q} Q=T_{q} \Phi_{h} H_{q} Q$;
2. $T_{q} Q=V_{q} Q \oplus H_{q} Q$.

Vectors in $H_{q} Q$ are called horizontal. Given a connection on $Q$, a tangent vector $v_{q} \in T_{q} Q$ can be decomposed into horizontal and vertical components as $v_{q}=$ hor $v_{q}+$ ver $v_{q}$.

Equivalently, a connection can be described by a Lie algebra-valued one-form, the connection form. The connection form is a map $\Gamma(q): T_{q} Q \rightarrow \mathfrak{g}$, where $\mathfrak{g}$ is the Lie algebra corresponding to the Lie group $G$, with the properties:

1. $\Gamma(q)\left(\xi_{Q}\right)=\xi$ for $\xi \in \mathfrak{g}$;
2. $\Gamma(q) v_{q}$ is equivariant, ie. $\Gamma\left(\Phi_{h}(q)\right) T_{q} \Phi_{h}(q) v_{q}=\operatorname{Ad}_{h} \Gamma(q) v_{q}$.

The connection one-form thus defines the horizontal subspace of $T_{q} Q$ as the set of all tangent vectors upon which the connection form vanishes: $H_{q} Q=\left\{v_{q} \mid \Gamma(q) v_{q}=0\right\}$.

Further, it can be shown that the connection one-form can be expressed in local coordinates $q=(g, s)$ as:

$$
\begin{equation*}
\Gamma(q) v_{q}=\operatorname{Ad}_{g}\left(\mathcal{A}(s) \dot{s}+g^{-1} \dot{g}\right) \tag{2.10}
\end{equation*}
$$

where $\mathcal{A}: T \mathcal{S} \rightarrow \mathfrak{g}$ is the "local" form of the connection. Any vector that lies in the horizontal subspace must satisfy the constraint:

$$
\begin{equation*}
g^{-1} \dot{g}=-\mathcal{A}(s) \dot{s} \tag{2.11}
\end{equation*}
$$

This is the governing equation for a particular class of nonholonomic systems where the configuration space is that of a principal fiber bundle and the equations of motion are specified by the connection on the bundle [100]. Equation (2.11) specifies the relationship between internal shape changes, $\dot{s}$, and the group motion, $g$. In the general case, the solution to (2.11) can be expressed as

$$
\begin{equation*}
g(T)=g(0) \exp \xi(s) \tag{2.12}
\end{equation*}
$$

where $s:[0, T] \rightarrow \mathcal{S}$ is a closed curve and $\xi(s)$ is a Lie algebra valued function given by the expansion [74]:

$$
\begin{equation*}
\xi(s)=-\overline{\mathcal{A}}+\frac{1}{2} \overline{[\overline{\mathcal{A}}, \mathcal{A}]}-\frac{1}{3} \overline{[\overline{[\overline{\mathcal{A}}, \mathcal{A}]}, \mathcal{A}]}-\frac{1}{12}[\overline{\mathcal{A}}, \overline{[\overline{\mathcal{A}}, \mathcal{A}]}]+\ldots \tag{2.13}
\end{equation*}
$$

where $\overline{\mathcal{A}}(t) \equiv \int_{0}^{t} \mathcal{A}(s(\tau)) \dot{s}(\tau) \mathrm{d} \tau$. In the special case where $G$ is Abelian, all of the bracketed terms on the right hand side of the expression in Equation (2.13) are zero,
and the solution simplifies to

$$
\begin{align*}
g(T) & =g(0) \exp \left(-\int_{0}^{T} \mathcal{A}(s(\tau)) \dot{s}(\tau) \mathrm{d} \tau\right)  \tag{2.14}\\
& =g(0) \exp \left(-\int_{\partial \mathrm{C}} \mathcal{A}(s) \mathrm{d} s\right)  \tag{2.15}\\
& =g(0) \exp \left(-\iint_{\mathrm{C}} \mathrm{~d} \mathcal{A}\right) \tag{2.16}
\end{align*}
$$

where C is the area in shape space enclosed by the path $\partial \mathrm{C}, \mathrm{d} \mathcal{A}$ is the exterior derivative of $\mathcal{A}$ and the final equality is by Stokes' theorem. In component form, $\mathrm{d} \mathcal{A}_{i j}=\frac{\partial \mathcal{A}_{j}}{\partial s_{i}}-\frac{\partial \mathcal{A}_{i}}{\partial s_{j}}$.

Given vector fields $X$ and $Y$ on $M$ and a connection form $\Gamma: T Q \rightarrow \mathfrak{g}$, the curvature form, $\gamma(X, Y): T Q \times T Q \rightarrow \mathfrak{g}$ is given by [78]

$$
\begin{equation*}
\gamma(X, Y)=\mathrm{d} \Gamma(X, Y)-[\Gamma(X), \Gamma(Y)] \tag{2.17}
\end{equation*}
$$

where $[\cdot, \cdot]$ is the Jacobi-Lie bracket of vector fields on $M$.
The local form of the curvature of the connection has coordinates:

$$
\begin{equation*}
F_{i j}=\frac{\partial \mathcal{A}_{j}}{\partial s_{i}}-\frac{\partial \mathcal{A}_{i}}{\partial s_{j}}-\left[\mathcal{A}_{i}, \mathcal{A}_{j}\right] \tag{2.18}
\end{equation*}
$$

where $\mathcal{A}_{i}$ is the $i^{\text {th }}$ component of the local form of the connection and $s_{i}$ is the $i^{\text {th }}$ shape variable. Note that in the special case when $G$ is Abelian (and the Lie bracket is zero), $F=\mathrm{d} \mathcal{A}$. The curvature can be thought of as an infinitesimal version of holonomy or geometric phase. In general, the locomotion of a swimmer is composed of both a dynamic phase and a geometric phase. In the particular case when the system momentum is zero, as in most of the examples considered in this study, the geometric phase completely determines the net motion. In the context of a momentum-free swimmer, the geometric phase is the net displacement and rotation due to cyclic
inputs in the swimmer's internal shape. In the language of geometric mechanics, the geometric phase is the motion in the fiber group variables associated with a closed path in the shape (base) space.

### 2.5 Semidirect product groups

Let $G$ be a Lie group, $V$ a vector space, $g_{1}, g_{2} \in G$ and $v_{1}, v_{2} \in V$. One can then form the semidirect product Lie group $S=G(S) V$ where group multiplication is

$$
\begin{equation*}
\left(g_{1}, v_{1}\right) \cdot\left(g_{2}, v_{2}\right)=\left(g_{1} g_{2}, v_{1}+g_{1} v_{2}\right) . \tag{2.19}
\end{equation*}
$$

In particular, the semidirect product group $S E(2)$ can be expressed as the semidirect product of $G=S O(2)$ and $V=\mathbb{R}^{2}$.

Consider $S_{1}, S_{2} \in S E(2)$ where $S_{1}=\left(\begin{array}{cc}R_{1} & r_{1} \\ 0 & 1\end{array}\right)$ and $S_{2}=\left(\begin{array}{cc}R_{2} & r_{2} \\ 0 & 1\end{array}\right)$. Expressed in this form, the semidirect product of two elements of $S E(2)$ can be found by the ordinary matrix product:

$$
S_{1} S_{2}=\left(\begin{array}{cc}
R_{1} & r_{1}  \tag{2.20}\\
0 & 1
\end{array}\right)\left(\begin{array}{cc}
R_{2} & r_{2} \\
0 & 1
\end{array}\right)=\left(\begin{array}{cc}
R_{1} R_{2} & r_{1}+R_{1} r_{2} \\
0 & 1
\end{array}\right)
$$

Equivalently, $S_{1} S_{2}=\left(R_{1}, r_{1}\right) \cdot\left(R_{2}, r_{2}\right)=\left(R_{1} R_{2}, r_{1}+R_{1} r_{2}\right)$.

## Chapter 3

## Motion in Potential Flow

In this chapter we consider the motion of a two-dimensional deformable body through an inviscid, incompressible fluid. Specifically, we wish to understand the relationship between the internal shape changes and the overall motion of a two joint, three link swimmer which begins at rest and where the generalized notion of momentum of the body-fluid system is zero for all time.

We consider this work to be an extension of progress in the study of momentumfree swimming in potential flow. Benjamin and Ellis [13] and Saffman [101] were the first to demonstrate that net motion is possible in potential flow even in the absence of vorticity. By formulating the dynamics of a deformable swimmer in the language of geometric mechanics, Kelly [59] elucidated how the variables parameterizing the internal shape changes were related by the connection to those associated with the overall motion in the plane. The connection in turn is a function of 'unit' velocity potentials which depend only on the shape of the swimmer surface and account for all of the effects of the fluid. Perturbation theory was applied to derive analytical expressions of the connection for an "amoeba-like" swimmer modeled by small-amplitude deformations of a nominally circular cross-section cylinder, though the result applies only to small-amplitude deformations.

Rather than attempting to compute a hydrodynamically accurate form of the connection as Kelly did, Radford [99] took a slightly different approach in studying the motion of a three link swimmer in potential flow. Each link was assigned constant added masses to account for resistance due to the surrounding fluid. Although the presence of the other body links affects the resistance to motion in a non-trivial manner, Radford simply assumed that the added mass terms used were those for a hydrodynamically isolated elliptical body. However no attempt was made to compute the true added inertia terms which are a function of the swimmer's instantaneous shape. Still, for this simplified system Radford computed the connection and showed that net translation - in addition to rotation - could be achieved even in the absence of viscosity.

In this work we also consider a three linked swimmer in potential flow, and we numerically compute the various added inertia terms which are needed to accurately determine the connection. Whereas Radford assumed the various links to be hydrodynamically decoupled, our solution accounts for the coupling between links due to the fluid. We believe this to be the first accurate solution for the motion of a multiply connected body undergoing large amplitude shape changes in potential flow.

Although the body begins at rest, the fluid is inviscid and there are no external forces acting on the body, it is able to achieve net translation and rotation as a result of geometric phase. This is analogous to a cat's rotation as it falls to the ground. A cat that is held still and dropped upside-down will move its limbs in such a way that it rotates and lands on its feet. Like the momentum-free swimmer-fluid system, since no external forces are acting on the cat, this motion happens while maintaining zero total angular momentum.

The assumptions and governing equations for the system are discussed in Section 3.1. We introduce the geometry of the swimmer in Section 3.2. In Section 3.3 we compute the kinetic energy of the swimmer-fluid system, and from the conservation
of energy, we derive the equations of motion in a classical mechanics framework in Section 3.5 and in the context of geometric mechanics in Section 3.7. The numerical method used to solve Laplace's equation is described in Section 3.4. Some example gaits are presented in Section 3.6. With the equations formulated with the structure of a fiber bundle over the shape space, the hydromechanical connection provides insight into the development of useful gaits, as detailed in Chapter 5. In Appendix A we include a MATLAB version of the code used to numerically implement this model.

### 3.1 Potential Flow

Here we present the assumptions and governing equations that we will use for the rest of this chapter to study the motion of a deformable swimmer in two-dimensional potential flow. An inviscid fluid beginning at rest will remain irrotational such that the velocity $\mathbf{v}$ satisfies

$$
\begin{equation*}
\nabla \times \mathbf{v}=0 \tag{3.1}
\end{equation*}
$$

which implies that the velocity field may be expressed as the gradient of a potential:

$$
\begin{equation*}
\mathbf{v}=\nabla \Phi \tag{3.2}
\end{equation*}
$$

Further, an incompressible fluid is divergence-free:

$$
\begin{equation*}
\nabla \cdot \mathbf{v}=0 \tag{3.3}
\end{equation*}
$$

Equations (3.2) and (3.3) combined yield Laplace's equation

$$
\begin{equation*}
\nabla^{2} \Phi=0 \tag{3.4}
\end{equation*}
$$

In addition, we also assume a constant density fluid which implies incompressibility. We will solve Laplace's equation subject to the boundary conditions of the swimmer to find the velocity potential.

### 3.2 The Swimmer

According to Purcell's scallop theorem [98] a swimming body with only one degree of freedom is unable to achieve net motion through cyclic shape changes in a zero Reynolds number flow due to the time-reversibility of the system. On the opposite end of the Reynolds number range $(R e=\infty)$, the system is similarly reversible in potential flow when the system is free of vorticity [101]. Hence to study both cases, we consider a planar two degree of freedom swimmer - the simplest swimmer capable of achieving net motion through cyclic shape changes.

This swimmer considered in the potential flow study is a three-link, neutrally buoyant articulated body as shown in Figure 3.1. The links are chosen to be ellipses with semi-major and semi-minor axes of lengths $a$ and $e$, respectively, though the results easily generalize to arbitrary geometries. The three links, identified by $\mathcal{B}_{i}, i=$ $1,2,3$, are assumed to be connected by invisible hinged joints.

The joints are a distance $l=a+c$ from the center of each ellipse, and the angles between the major axis of the center ellipse and that of the outer ellipses are $\theta_{1}$ and $\theta_{2}$. The planar position and orientation of the middle link, relative to an inertial frame is parameterized by the variables $(x, y, \beta)$. The various coordinate frames fixed to the links and the inertial reference frame are shown in Figure 3.2.

We seek to understand how to achieve overall motions as a result of internal shape changes.


Figure 3.1: Schematic of 3-link articulated swimmer.


Figure 3.2: Coordinate frames and identification of links for 3-link articulated swimmer.

### 3.3 Kinetic Energy

In Section 3.5, we will use the conservation of the system impulse - a momentumlike quantity - to derive the equations of motion for the swimmer. The impulse is a function of the body velocities, the mass and moment of inertia of the swimmer, and a generalized notion of mass - the added inertia terms - which incorporates the effect of the surrounding fluid. In this section we derive the kinetic energy of the fluid to elucidate the physical significance of the added inertia terms. We show that the added inertia terms are functions of the velocity potential, which in turn depend only on the swimmer geometry. Since the added inertia terms completely account for the effect of the fluid, there is no need to directly compute the fluid properties in a numerical simulation.

For a planar body undergoing rigid motions with translational velocities $u, v$ and rotational velocity $\omega$ in a perfect fluid, Kirchhoff $[60,64]$ showed that one can express the velocity potential $\Phi$ as a sum of functions which depend only on the shape and configuration of the body:

$$
\begin{equation*}
\Phi=u \phi_{u}+v \phi_{v}+\omega \phi_{\omega} . \tag{3.5}
\end{equation*}
$$

Similarly, for a three link swimmer where each link has body velocity $\xi_{i}=\left(u_{i}, v_{i}, \omega_{i}\right)$, the velocity potential may be expressed as:

$$
\begin{equation*}
\Phi=\sum_{i=1}^{3}\left(u_{i} \phi_{u_{i}}+v_{i} \phi_{v_{i}}+\omega_{i} \phi_{\omega_{i}}\right) \tag{3.6}
\end{equation*}
$$

where the index $i$ identifies the velocity components of the $i^{\text {th }}$ body. These functions, which depend only on the variables of the solid bodies, capture the energy of the surrounding fluid. This greatly simplifies the problem, since we no longer must explicitly keep track of each fluid particle.

The kinetic energy of the unit density fluid is

$$
\begin{equation*}
K E_{\mathfrak{F}}=\frac{1}{2} \iint \mathbf{v} \cdot \mathbf{v} d \Omega . \tag{3.7}
\end{equation*}
$$

Substituting Equation (3.2) into (3.7) and applying the identity $\nabla \cdot(\Phi \nabla \Phi)=\nabla \Phi \cdot$ $\nabla \Phi+\Phi \nabla^{2} \Phi$ plus Equation (3.4), the kinetic energy becomes

$$
\begin{equation*}
K E_{\mathfrak{F}}=\frac{1}{2} \iint \nabla \cdot(\Phi \nabla \Phi) d \Omega . \tag{3.8}
\end{equation*}
$$

Finally, Green's theorem allows us to express Equation (3.8) as an integral over the surface of the bodies, rather than over the fluid domain:

$$
\begin{equation*}
K E_{\mathfrak{F}}=\frac{1}{2} \int \Phi \frac{\partial \Phi}{\partial n} d S \tag{3.9}
\end{equation*}
$$

where $\frac{\partial \Phi}{\partial n}=\nabla \Phi \cdot n$, and $n$ is the outward normal at the surface. The Neumann boundary condition at the surface of the $i^{\text {th }}$ body is given by

$$
\begin{equation*}
\frac{\partial \Phi_{i}}{\partial n}=\frac{\partial \phi_{u_{i}}}{\partial n}\left(u_{i}-\omega_{i} y_{i}\right)+\frac{\partial \phi_{v_{i}}}{\partial n}\left(v_{i}+\omega_{i} x_{i}\right) \tag{3.10}
\end{equation*}
$$

where $x_{i}$ and $y_{i}$ are distances in the $i_{i}$ and $j_{i}$ directions from the origin of $\mathcal{B}_{i}$. Due to the linearity of Laplace's equation, finding a solution to Equation (3.4) subject to the boundary condition (3.10) becomes significantly simplified. By the superposition property, the nine scalar potentials $\phi_{u_{i}}, \phi_{v_{i}}$ and $\phi_{\omega_{i}}$ may be solved individually subject to simpler boundary conditions, and then summed to "reconstruct" the scalar potential of the entire system. The simpler problem is to solve Laplace's equations
for $\phi_{u_{i}}, \phi_{v_{i}}$ and $\phi_{\omega_{i}}$ with respective boundary conditions:

$$
\begin{align*}
& \begin{cases}\frac{\partial \phi_{u_{i}}}{\partial n}=\mathbf{u}_{i} \cdot \mathbf{n}_{i}, & \text { on } \partial \mathcal{B}_{i} \\
\frac{\partial \phi_{u_{i}}}{\partial n}=0, & \text { on } \partial \mathcal{B}_{j}, j \neq i\end{cases}  \tag{3.11}\\
& \begin{cases}\frac{\partial \phi_{v_{i}}}{\partial n}=\mathbf{v}_{i} \cdot \mathbf{n}_{i}, & \text { on } \partial \mathcal{B}_{i} \\
\frac{\partial \phi_{v_{i}}}{\partial n}=0, & \text { on } \partial \mathcal{B}_{j}, j \neq i\end{cases}  \tag{3.12}\\
& \begin{cases}\frac{\partial \phi_{\omega_{i}}}{\partial n}=\left(\boldsymbol{\omega}_{i} \times X_{i}\right) \cdot \mathbf{n}_{i}, & \text { on } \partial \mathcal{B}_{i} \\
\frac{\partial \phi_{\omega_{i}}}{\partial n}=0, & \text { on } \partial \mathcal{B}_{j}, j \neq i\end{cases} \tag{3.13}
\end{align*}
$$

where $\mathbf{u}_{i}=\left(u_{i}, 0,0\right), \mathbf{v}_{i}=\left(0, v_{i}, 0\right)$ and $\boldsymbol{\omega}_{i}=\left(0,0, \omega_{i}\right)$, and where $\mathbf{n}_{i}$ is the normal vector expressed in $\mathcal{B}_{i}$ body coordinates.

Substituting (3.6) and (3.10) into (3.9), the kinetic energy of the fluid can now be expressed as
$K E_{\mathfrak{F}}=\frac{1}{2} \sum_{i=1}^{3} \sum_{j=1}^{3}\left(\mathbb{T}^{u_{i} u_{j}} u_{i} u_{j}+\mathbb{T}^{v_{i} v_{j}} v_{i} v_{j}+\mathbb{I}^{\omega_{i} \omega_{j}} \omega_{i} \omega_{j}+\mathbb{I}^{u_{i} v_{j}} u_{i} v_{j}+\mathbb{T}^{u_{i} \omega_{j}} u_{i} \omega_{j}+\mathbb{I}^{v_{i} \omega_{j}} v_{i} \omega_{j}\right)$,
where

$$
\begin{align*}
& \mathbb{I}^{u_{i} u_{j}}= \begin{cases}-\iint \phi_{u_{i}} \frac{\partial \phi_{u_{j}}}{\partial n} d S & i=j \\
-\frac{1}{2} \iint \phi_{u_{i}} \frac{\partial \phi_{u_{j}}}{\partial n}+\phi_{u_{j}} \frac{\partial \phi_{u_{i}}}{\partial n} d S & i \neq j\end{cases}  \tag{3.15}\\
& \mathbb{I}^{v_{i} v_{j}}= \begin{cases}-\iint \phi_{v_{i}} \frac{\partial \phi_{v_{j}}}{\partial n} d S & i=j \\
-\frac{1}{2} \iint \phi_{v_{i}} \frac{\partial \phi_{v_{j}}}{\partial n}+\phi_{v_{j}} \frac{\partial \phi_{v_{i}}}{\partial n} d S & i \neq j\end{cases}  \tag{3.16}\\
& \mathbb{I}^{\omega_{i} \omega_{j}}= \begin{cases}-\iint \phi_{\omega_{i}} \frac{\partial \phi_{\omega_{j}}}{\partial n} d S & i=j \\
-\frac{1}{2} \iint \phi_{\omega_{i}} \frac{\partial \phi_{\omega_{j}}}{\partial n}+\phi_{\omega_{j}} \frac{\partial \phi_{\omega_{i}}}{\partial n} d S & i \neq j\end{cases}  \tag{3.17}\\
& \mathbb{I}^{u_{i} v_{j}}= \begin{cases}-\iint \phi_{u_{i}} \frac{\partial \phi_{v_{j}}}{\partial n} d S & i=j \\
-\frac{1}{2} \iint \phi_{u_{i}} \frac{\partial \phi_{v_{j}}}{\partial n}+\phi_{v_{j}} \frac{\partial \phi_{u_{i}}}{\partial n} d S & i \neq j\end{cases}  \tag{3.18}\\
& \mathbb{I}^{u_{i} \omega_{j}}= \begin{cases}-\iint \phi_{u_{i}} \frac{\partial \phi_{\omega_{j}}}{\partial n} d S & i=j \\
-\frac{1}{2} \iint \phi_{u_{i}} \frac{\partial \phi_{\omega_{j}}}{\partial n}+\phi_{\omega_{j}} \frac{\partial \phi_{u_{i}}}{\partial n} d S & i \neq j \\
-\iint \phi_{v_{i}} \frac{\partial \phi_{\omega_{j}}}{\partial n} d S & i \neq j\end{cases} \tag{3.19}
\end{align*}
$$

The added inertia terms, $\mathbb{I}$, are functions of the body geometry and are computed numerically by a panel method described in the next section.

### 3.4 Numerical Method

To compute the added inertia terms in (3.15)-(3.20), we need to solve for the nine velocity potentials $\phi_{u_{i}}, \phi_{v_{i}}$, and $\phi_{\omega_{i}}(i=1 \ldots 3)$. Each velocity potential is a solution to Laplace's equation subject to corresponding boundary conditions (3.11)-(3.13). By Green's theorem, Laplace's equation over the fluid domain can be transformed into a boundary integral equation [47]:

$$
\begin{equation*}
-2 \pi \sigma(p)+\oint \frac{\partial}{\partial n}\left(\frac{1}{r\left(p, p^{\prime}\right)} \sigma\left(p^{\prime}\right) d A^{\prime}\right)=v_{n}(p) \tag{3.21}
\end{equation*}
$$

where $p$ is a point on the surface of the body, $\sigma(p)$ is the source (or sink) singularity strength density at $p, r\left(p, p^{\prime}\right)$ is the distance between points $p$ and $p^{\prime}$ and $v_{n}(p)$ is the normal component of the velocity at $p$. These equations can be solved by a panel method (also known as a boundary element method). This usually results in significant computational savings compared to solving the original partial differential equation over the fluid domain. We follow the approach of Hess \& Smith [47].

First, the surface of the swimmer is discretized into $N$ straight line panels, or elements. At the center of each panel we define a control point which is where the boundary condition will be imposed. A constant unit source density is assigned to each panel. This source density distribution over the panel induces a velocity everywhere in the fluid, including at each control point. Consider a unit strength point source singularity located at $\left(x_{o}, y_{o}\right)$. The velocity induced at a point $(x, y)$ due to the singularity has components:

$$
\begin{align*}
u_{s} & =\frac{1}{2 \pi} \frac{x-x_{o}}{\left(x-x_{o}\right)^{2}+\left(y-y_{o}\right)^{2}}  \tag{3.22}\\
v_{s} & =\frac{1}{2 \pi} \frac{y-y_{o}}{\left(x-x_{o}\right)^{2}+\left(y-y_{o}\right)^{2}} \tag{3.23}
\end{align*}
$$

We now assume that the $j^{\text {th }}$ panel has a constant unit source density distribution. We fix a coordinate frame to the panel such that the center of the panel is at the origin, the abscissa is aligned with the tangent direction of the panel, and the positive ordinate is in the outward normal direction (See Figure 3.3). The velocity induced at the point $(x, y)$ — which may represent the coordinates of the $i^{\text {th }}$ control point in the panel $j$ frame - by the source distribution over the $j^{\text {th }}$ panel is found by integrating the right hand side of Equations (3.22) and (3.23) over the length of the $j^{\text {th }}$ straight line panel. The formulas for these velocities expressed with respect to a frame fixed to panel $j$ are [47]:

$$
\begin{equation*}
V_{x}=\frac{1}{4 \pi} \ln \left[\frac{(x+\Delta s / 2)^{2}+y^{2}}{(x-\Delta s / 2)^{2}+y^{2}}\right] \tag{3.24}
\end{equation*}
$$



Figure 3.3: Coordinate system fixed to a panel. $\xi$ and $\eta$ are coordinates along a panel while $x$ and $y$ are used for points not on the panel. Reproduced from Hess and Smith [47].

$$
\begin{equation*}
V_{y}=\frac{1}{2 \pi} \tan ^{-1}\left[\frac{y \Delta s}{x^{2}+y^{2}-(\Delta s / 2)^{2}}\right] \tag{3.25}
\end{equation*}
$$

where $\Delta s$ is the length of panel $j$. The velocity induced at a control point by its own panel has components $V_{x}=0$ and $V_{y}=1 / 2 \pi$. Since the expressions for the velocity components are in a coordinate system based on the particular panel, they must be transformed into the reference coordinate system in which the body geometry was defined.

Let $\mathbf{t}_{j}=\left(t_{j x}, t_{j y}\right)$ and $\mathbf{n}_{j}=\left(n_{j x}, n_{j y}\right)$ be the unit normal and tangent vectors fixed to the $j^{\text {th }}$ panel. Then the inertial frame velocity components at the control point of panel $i$ due to a unit source density distribution on panel $j$ are:

$$
\begin{align*}
X_{i, j} & =V_{x} t_{j x}-V_{y} t_{j y}  \tag{3.26}\\
Y_{i, j} & =-V_{x} n_{j x}+V_{y} n_{j y} \tag{3.27}
\end{align*}
$$

Let $\mathbf{V}_{i, j}$ be the velocity induced at control point $i$ due to a unit source distribution along panel $j$, which can expressed with respect to an inertial frame as $\mathbf{V}_{i, j}=X_{i, j} \mathbf{i}+Y_{i, j} \mathbf{j}$ where $\mathbf{i}$ and $\mathbf{j}$ are unit vectors in the inertial coordinate frame.

If $\mathbf{n}_{i}$ are the unit normal vectors corresponding to the $i^{\text {th }}$ panel, we can define the following expression:

$$
\begin{equation*}
A_{i, j}^{n}=\mathbf{n}_{i} \cdot \mathbf{V}_{i, j}, \tag{3.28}
\end{equation*}
$$

which represents the normal component (with respect to a frame fixed to the $i^{\text {th }}$ panel) of the velocity induced at the control point of panel $i$ due to a unit strength source distribution on panel $j$. Finally, Equation (3.21) can be numerically approximated by a set of linear equations:

$$
\left[\begin{array}{cccc}
A_{11}^{n} & A_{12}^{n} & \cdots & A_{1 N}^{n}  \tag{3.29}\\
A_{21}^{n} & A_{22}^{n} & \cdots & A_{2 N}^{n} \\
\vdots & \vdots & \ddots & \vdots \\
A_{N 1}^{n} & A_{N 2}^{n} & \cdots & A_{N N}^{n}
\end{array}\right]\left[\begin{array}{c}
\sigma_{1} \\
\sigma_{2} \\
\vdots \\
\sigma_{N}
\end{array}\right]=\left[\begin{array}{c}
v_{n 1} \\
v_{n 2} \\
\vdots \\
v_{n N}
\end{array}\right]
$$

which we can write as $A^{n} \sigma=v_{n}$. For each of the particular boundary conditions (3.11)-(3.13), we determine the corresponding normal component of velocity at each control point as a vector, $v_{n}$. Since all the influence coefficients can be determined from the geometry of the body, the system of equations is then solved for the source density distribution by inverting the matrix of influence coefficients: $\sigma=\left(A^{n}\right)^{-1} v_{n}$. For each of the source distributions, the corresponding velocity potential is computed at the control points. The velocity potential at a point $p^{\prime}$ due to a point source singularity of strength $\sigma$ at point $p$ is $\phi\left(p^{\prime}, p\right)=\sigma(p) \log r\left(p, p^{\prime}\right) / 2 \pi$, where $r\left(p, p^{\prime}\right)$ is the distance between $p^{\prime}$ and $p$. Thus, the potential at a point $p^{\prime}$ due to a distribution of sources over panels on a boundary can be found from a discrete form of Equation (3.30).

$$
\begin{equation*}
\phi\left(p^{\prime}\right)=\int \frac{\sigma(p) \log r\left(p, p^{\prime}\right)}{2 \pi} \mathrm{~d} A \tag{3.30}
\end{equation*}
$$

These velocity potentials are assumed to be constant along each panel - a valid assumption for sufficiently small panels - and we discretize the integrals in Equa-
tions (3.15)-(3.20) to compute the added inertia terms from these velocity potentials.

### 3.5 Equations of Motion

Although the momentum of the system is unbounded in a fluid with infinite extent [64], an analogous quantity - the impulse which is the product of the body velocities and the real plus added inertias - is finite. Let $T_{i}$ be the transformation

$$
T_{i}=\left[\begin{array}{ccc}
\cos \theta_{i} & \sin \theta_{i} & \pm l \sin \theta_{i}  \tag{3.31}\\
-\sin \theta_{i} & \cos \theta_{i} & \pm l\left(1+\cos \theta_{i}\right) \\
0 & 0 & 1
\end{array}\right], i=1,2
$$

where + and - correspond to $i=1$ and 2 , respectively. Let $\mathbb{M}_{i, j}=m_{i, j}+\mathbb{I}_{i, j}$ be the matrix of real plus added inertia terms where

$$
\mathbb{I}_{i, j}=\left[\begin{array}{lll}
\mathbb{I}^{u_{i} u_{j}} & \mathbb{I}^{u_{i} v_{j}} & \mathbb{I}^{u_{i} \omega_{j}}  \tag{3.32}\\
\mathbb{I}^{v_{i} u_{j}} & \mathbb{I}^{v_{i} v_{j}} & \mathbb{I}^{v_{i} \omega_{j}} \\
\mathbb{I}^{\omega_{i} u_{j}} & \mathbb{I}^{\omega_{i} v_{j}} & \mathbb{I}^{\omega_{i} \omega_{j}}
\end{array}\right]
$$

and

$$
m_{i, j}=\left\{\begin{array}{lll}
{\left[\begin{array}{ccc}
m & 0 & 0 \\
0 & m & 0 \\
0 & 0 & J
\end{array}\right]} & i=j  \tag{3.33}\\
\mathbf{0}^{3 \times 3} & & i \neq j
\end{array}\right.
$$

where $m$ and $J$ are the mass and moment of inertia about the center of mass, respectively, of one link.

In general, the three (translational and rotational) impulse components vary in time. However, when the system begins from rest, the total impulse of the system expressed with respect to the body frame of the middle link $\left(\mathcal{B}_{3}\right)$ is conserved and thus equal to zero, and all components remain zero for all time:

$$
\begin{equation*}
h_{s}=\sum_{i=1}^{2} \sum_{j=1}^{3} T_{i}^{\mathrm{T}} \mathbb{M}_{i, j} \xi_{j}+\sum_{j=1}^{3} \mathbb{M}_{3, j} \xi_{j}=\mathbf{0} \tag{3.35}
\end{equation*}
$$

Since the motion of the outer links is constrained by the joints, their body velocities, $\xi_{i}=\left(u_{i}, v_{i}, \omega_{i}\right)^{\mathrm{T}}$ can be expressed as functions of the velocity of the middle link plus the velocity of the outer links relative to the middle link as follows,

$$
\left[\begin{array}{c}
u_{i}  \tag{3.36}\\
v_{i} \\
\omega_{i}
\end{array}\right]=\left[\begin{array}{ccc}
\cos \theta_{1} & \sin \theta_{1} & \pm l \sin \theta_{i} \\
-\sin \theta_{1} & \cos \theta_{1} & \pm l\left(1+\cos \theta_{i}\right) \\
0 & 0 & 1
\end{array}\right]\left[\begin{array}{c}
u_{3} \\
v_{3} \\
\omega_{3}
\end{array}\right]+\left[\begin{array}{c}
0 \\
\pm l \\
1
\end{array}\right] \dot{\theta}_{i},
$$

or in shorthand as $\xi_{i}=T_{i} \xi_{3}+L_{i} \dot{\theta}_{i}$, where + and - correspond to $i=1$ and 2 , respectively. Substituting (3.36) into (3.35), grouping and rearranging terms yields

$$
\begin{equation*}
\mathbb{I}_{\mathrm{loc}} \xi_{3}+\left(\sum_{i=1}^{2} \sum_{j=1}^{2} T_{i}^{\mathrm{T}} \mathbb{M}_{i, j} L_{j}+\sum_{j=1}^{2} \mathbb{M}_{3, j} L_{j}\right) \dot{\theta}_{j}=0 \tag{3.37}
\end{equation*}
$$

where

$$
\begin{equation*}
\mathbb{I}_{\text {loc }}=\sum_{i=1}^{2} \sum_{j=1}^{2} T_{i}^{\mathrm{T}} \mathbb{M}_{i, j} T_{j}+\sum_{i=1}^{2} T_{i}^{\mathrm{T}} \mathbb{M}_{i, 3}+\sum_{j=1}^{2} \mathbb{M}_{3, j} T_{j}+\mathbb{M}_{3,3} \tag{3.38}
\end{equation*}
$$

is known as the locked moment of inertia which is equivalent to the instantaneous moment of inertia of the body if its joints were to be locked in their current state. When the locked moment of inertia matrix is nonsingular, the velocity of the body, expressed with respect to the $\mathcal{B}_{3}$ body frame, may be expressed as a function of the
joint velocities, $\dot{\theta}_{1}$ and $\dot{\theta}_{2}$ :

$$
\begin{equation*}
\xi_{3}=-\mathbb{I}_{\mathrm{loc}}^{-1}\left(\sum_{i=1}^{2} \sum_{j=1}^{2} T_{i}^{\mathrm{T}} \mathbb{M}_{i, j} L_{j}+\sum_{j=1}^{2} \mathbb{M}_{3, j} L_{j}\right) \dot{\theta}_{j} . \tag{3.39}
\end{equation*}
$$

Equation (3.39) can be used directly to determine the planar motion of the swimmer resulting from the prescribed motion of the joints by numerically integrating the body velocity, $\xi_{3}$, in time. Since the added inertia terms are a function of the shape of the swimmer, they are recomputed at each time.

### 3.6 Example Gaits

Through intuition and numerical experiments, the code (see Appendix A) was used to identify cyclic gaits to achieve various types of motions. In all cases, the links consisted of ellipses discretized into 200 panels with $a=1, e=0.1$ and $c=0.2$ (see Figure 3.1). Each gait was divided into 50 equally spaced time steps. The resulting motion is non-dimensionalized by $l=a+c$.

One important and potentially counter-intuitive aspect of vorticity-free, momentumfree motion in potential flow is that the instant the body ceases to deform, its motion stops. The swimmer does not continue to "coast". Further, if the deformation is reversed, the swimmer velocity will exactly reverse. Both of these results should be apparent from Equation (3.39) or equivalently from the fact that the local form of the connection relating $\dot{\theta}_{j}$ and $\xi_{3}$ is a linear map. These curious results were observed as early as 1966 by Saffman [101]:

The situation is therefore very different from the inviscid propulsion mechanisms of Lighthill and Wu where there is a persistent transfer of momentum from body to fluid associated with the creation of vorticity. Also there is no energy dissipation in the present case and no net work is ex-
pended. There is of course a transfer of energy between body and fluid, but this is loss-free and reversible.

Despite the seemingly non-physical nature of the dynamics of this system, an understanding of motion in potential flow may lend insight to generating certain types of motions in robotic applications or enhancing our understanding of some biological swimming maneuvers. In particular, we postulate that like the falling cat, the impressive turning performance of marine animals such as sea lions [34] may be partially due to their great flexibility which allows them to explore a large range of shape space and generate a significant amount of geometric phase.

### 3.6.1 Forward Gait

By symmetry, circular paths centered about the origin of the $\theta_{1}-\theta_{2}$ shape space will generate motion with zero net rotation. This family of gaits can be expressed as:

$$
\begin{align*}
& \theta_{1}=A \cos (t-\phi)  \tag{3.40}\\
& \theta_{2}=A \sin (t-\phi) \tag{3.41}
\end{align*}
$$

As an example, we prescribe a gait specified by Equations (3.42) and (3.43) which generates a net translation of approximately $1.85 l$ and zero net rotation of the middle link as seen in Figure 3.4:

$$
\begin{align*}
& \theta_{1}=1.5 \cos \left(t-\frac{\pi}{4}\right)  \tag{3.42}\\
& \theta_{2}=1.5 \sin \left(t-\frac{\pi}{4}\right) \tag{3.43}
\end{align*}
$$

Snapshots of the swimmer motion are shown in Figure 3.5. Although the middle link does rotate during the gait, it experiences zero net rotation after one complete closed-path gait. Due to the rotation, the total length of the displacement path is


Figure 3.4: Potential flow swimmer: Forward gait example. (a) The translation of the center of the middle link non-dimensionalized by $l$. The black dot indicates the initial position. (b) the angle of the middle link versus time during the forward swimming gait specified by Equations (3.42) and (3.43). Snapshots of the swimmer motion are shown in Figure 3.5. The solid blue curves corresponds to the motion when the hydrodynamics are accurately computed while the dashed red curve corresponds to the motion under the assumption of hydrodynamically decoupled links.


Figure 3.5: Potential flow swimmer: Forward gait snapshots. The swimmer shape, position and orientation is shown for five instance of time during the gait specified by Equations (3.42) and (3.43). The blue curve is the same as in Figure 3.4(a) and shows the path traveled by the center of the middle link.
considerably longer than the net translation of 1.85l. Changing the starting position by choosing a different value for $\phi$ would not affect the net rotation, but would result in a different displacement path and net translation. This is a manifestation of the non-Abelian nature of the special Euclidean group and significantly complicates the problem of generating more arbitrary gaits to achieve a desired net translation.

To highlight the importance of accurately computing the hydrodynamic interaction between links, we also show in Figure 3.4(b) (red dashed curves) the resulting motion for this same gait when the links are assumed to be hydrodynamically decoupled. This was the assumption made by Radford [99]. For this geometry, it seems that
the failure to account for the hydrodynamic interaction between links typically results in over-estimating the resulting body velocity due to internal shape changes. Though we observe that for this particular gate the net translation for the hydrodynamically decoupled case is less than for the accurate, hydrodynamically coupled result, this result does not hold in general due to the non-Abelian nature of the system.

### 3.6.2 Turning Gait

Circular paths in shape space shifted away from the origin along the $\theta_{1}=-\theta_{2}$ diagonal tend to generate net rotations in addition to translation in the plane. This family of gaits can be expressed as:

$$
\begin{align*}
& \theta_{1}=-B+A \cos (t-\phi)  \tag{3.44}\\
& \theta_{2}=B+A \sin (t-\phi) \tag{3.45}
\end{align*}
$$

Physically, these gaits corresponds to the same motion as with the forward gaits but with a bias of $B$ in the orientation of the joints so that the swimmer nominally assumes a curved 'C' shape.

Equations (3.46) and (3.47) are a specific example gait which generates the translation and rotation of the middle link seen in Figure 3.6:

$$
\begin{align*}
& \theta_{1}=-0.8+0.8 \cos \left(t-\frac{\pi}{4}\right)  \tag{3.46}\\
& \theta_{2}=0.8+0.8 \sin \left(t-\frac{\pi}{4}\right) . \tag{3.47}
\end{align*}
$$

Snapshots of the swimmer motion during the gait are shown in Figure 3.7. In this example, the swimmer achieves a net rotation of about 0.59 radians, while translating about $0.95 l$. Further, for fixed amplitude $A$, we find that for a certain range of the bias $B$, the net rotation for a gait increases monotonically with increasing $B$. This


Figure 3.6: Potential flow swimmer: Turning gait example. (a) The translation of the center of the middle link non-dimensionalized by $l$. The black and blue dots indicate the initial and final positions, respectively; (b) the angle of the middle link versus time during the turning swimming gait specified by Equations (3.46) and (3.47). Snapshots of the swimmer motion are shown in Figure 3.7. The solid blue curves corresponds to the motion when the hydrodynamics are accurately computed while the dashed red curve corresponds to the motion under the assumption of hydrodynamically decoupled links.
may confirm the intuition that the more curved the swimmer is, the more rotation it is able to achieve. This particular specific result can be used to generate a look-up tablebased algorithm to follow a given trajectory in the plane. To track higher or lower curvature paths, the bias in the gait would be increased or decreased, respectively.

Although we have found one family of gaits to generate a desired amount of net rotation, in Chapter 5 we explain the difficulty in identifying large-amplitude gaits that achieve net rotation without translation.

### 3.7 Coordinate-free Analysis

In this section we apply a coordinate-free approach to derive the equations of motion in $\S 3.5$. This formalism often helps elucidate the system dynamics, particularly through the appearance of the connection which relates internal shape changes to the overall group motion. As we will later show, the geometric phase resulting from a


Figure 3.7: Potential flow swimmer: Turning gait snapshots. The swimmer shape, position and orientation is shown for five instance of time during the gait specified by Equations (3.46) and (3.47). The blue curve is the same as in Figure 3.6(a) and shows the path traveled by the center of the middle link.
specified internal shape change can often be expressed as an integral of the curvature of the connection over the area enclosed by the gait path in shape space. Although our $\theta_{1}-\theta_{2}$ shape space is two dimensional, this concept generalizes if we wish to extend the swimmer to have $N$ links. More generally, dynamical systems expressed in this more abstract geometric framework lend themselves to the application of wellknown tools from control theory. One such example is satellite with internal motors, where the dynamics are represented by a generalized form of the rigid-body phase formula [85, 78], and feedback control is applied to reorient or stabilize the system.

We represent the position and angle of the middle link of the swimmer as an element of the Special Euclidean Lie group, $g \in S E(2)$ (see $\S 2.5$ for details on $S E(2)$ ). Elements of the associated Lie algebra, $\xi \in \mathfrak{s e}(2)$, correspond to the body velocities of the swimmer links, $\xi_{i}=\left(u_{i}, v_{i}, \omega_{i}\right)$. The velocity of the outer links can be expressed with respect to the middle link by

$$
\begin{equation*}
\xi_{i}=\operatorname{Ad}_{\theta_{i}^{-1}} \xi_{3}+\zeta_{i}, \tag{3.48}
\end{equation*}
$$

where $\operatorname{Ad}_{\theta_{i}^{-1}}$ is the adjoint map defined as the derivative of the conjugation map at the identity, and $\zeta_{i}$ is the velocity of link $i$ relative to the middle link, expressed with respect to a frame fixed to link $i$. In the notation used in the $\S 3.5, \zeta_{i}=L_{i} \dot{\theta}_{i}$ $\left(L_{i}=\left[\begin{array}{lll}0 & \pm l & 1\end{array}\right]^{\mathrm{T}}\right)$. Using the matrix representations of the Lie group and Lie algebra elements, an explicit expression for the adjoint map's action on an element of
the Lie algebra can be found from multiplying out the right hand side of the expression $\operatorname{Ad}_{g} \xi=g \xi g^{-1}$. Equivalently, the adjoint map may be expressed in matrix form as

$$
\operatorname{Ad}_{g_{i}} \xi_{i}=T_{i}\left[\begin{array}{c}
u_{i}  \tag{3.49}\\
v_{i} \\
\omega_{i}
\end{array}\right]
$$

where $T_{i}$ is given by Equation (3.31). Here, $g_{i}$ corresponds to the rigid motion of the outer link relative to the middle link and may be parameterized by the angle of the joint, $\theta_{i}$.

We now wish to express the total impulse of the system with respect to a frame fixed to the middle link. Since the inertia terms $\mathbb{M}_{i, j}$ are functions of scalar potentials with respect to the $i^{\text {th }}$ and $j^{\text {th }}$ body frames, they must be expressed with respect to a frame fixed to the middle link, $\mathcal{B}_{3}$. In the case of a free rigid body, one would achieve this through the parallel axis theorem. Here, the adjoint map may be thought of as a generalization of the parallel axis theorem. Hence, the total impulse of the system relative to a frame fixed to the $\mathcal{B}_{3}$ frame is

$$
\begin{equation*}
h_{s}=\sum_{i=1}^{2} \sum_{j=1}^{3} \operatorname{Ad}_{\theta_{i}^{-1}}^{\mathrm{T}}\left(\mathbb{M}_{1, j} \xi_{j}\right)+\sum_{j=1}^{3} \mathbb{M}_{3, j} \xi_{j} \tag{3.50}
\end{equation*}
$$

Note that Equation (3.50) is equivalent to (3.35). Again, since the system begins from rest, $h_{s}=0$ and is conserved for all time due to the lack of external forces or moments on the body+fluid system [84]. Substituting (3.48) into (3.50), and rearranging terms yields the expression

$$
\begin{equation*}
\mathbb{I}_{\mathrm{loc}} \xi_{3}+\left(\sum_{i=1}^{2} \sum_{j=1}^{2} \operatorname{Ad}_{\theta_{i}^{-1}}^{\mathrm{T}} \mathbb{M}_{i, j} L_{j}+\sum_{j=1}^{2} \mathbb{M}_{3, j} L_{j}\right) \dot{\theta}_{j}=0 \tag{3.51}
\end{equation*}
$$

where

$$
\begin{equation*}
\mathbb{I}_{\mathrm{loc}}=\sum_{i=1}^{2} \sum_{j=1}^{2} \operatorname{Ad}_{\theta_{i}^{-1}}^{\mathrm{T}} \mathbb{M}_{i, j} \operatorname{Ad}_{\theta_{j}^{-1}}+\sum_{i=1}^{2} \operatorname{Ad}_{\theta_{i}^{-1}}^{\mathrm{T}} \mathbb{M}_{i, 3}+\sum_{j=1}^{2} \mathbb{M}_{3, j} \operatorname{Ad}_{\theta_{j}^{-1}}+\mathbb{M}_{3,3} \tag{3.52}
\end{equation*}
$$

Equations (3.51) and (3.52) are equivalent to (3.37) and (3.38), respectively. Finally, when $\mathbb{I}_{\text {loc }}$ is non-degenerate (i.e., its matrix representation is invertible), Equation (3.51) may be solved for $\xi_{3}$ and written as

$$
\begin{equation*}
\xi_{3}=-\mathcal{A}(\theta) \dot{\theta} \tag{3.53}
\end{equation*}
$$

where $\mathcal{A}(\theta)$ is the local form of the connection which is an $\mathfrak{s e}(2)$-valued one-form on the shape space. In other words, it maps instantaneous shape changes, parameterized by $\dot{\theta}=\left[\begin{array}{ll}\dot{\theta}_{1} & \dot{\theta}_{2}\end{array}\right]^{\mathrm{T}}$, into body velocities. It is a function of only the instantaneous shape, parameterized by $\theta=\left[\begin{array}{ll}\theta_{1} & \theta_{2}\end{array}\right]^{\mathrm{T}}$ and is given by the expression

$$
\mathcal{A}=\mathbb{I}_{\mathrm{loc}}^{-1}\left[\begin{array}{ll}
\mathbb{M}_{1} & \mathbb{M}_{2} \tag{3.54}
\end{array}\right],
$$

where

$$
\begin{equation*}
\mathbb{M}_{j}=\sum_{i=1}^{2} \operatorname{Ad}_{\theta_{i}^{-1}}^{\mathrm{T}} \mathbb{M}_{i, j} L_{j}+\mathbb{M}_{3, j} L_{j} \tag{3.55}
\end{equation*}
$$

Finally, the planar motion of the swimmer may be reconstructed in matrix form by recalling that $\xi=g^{-1} \dot{g}$, such that Equation (3.53) becomes

$$
\begin{equation*}
\dot{g}_{3}=-g_{3} \mathcal{A}(\theta) \dot{\theta} \tag{3.56}
\end{equation*}
$$

### 3.8 Summary

In this chapter, the equations of motion for an articulated swimmer in potential flow were derived from both classical and geometric mechanics frameworks. Whereas
prior work ignored the hydrodynamical interactions between bodies or assumed small amplitude deformations, the current work accurately models the motion of the swimmer. The hydrodynamical interactions between links are accounted for through added inertia terms that depend only on the system geometry and which are computed numerically at each time step. Finally, gaits determined through intuition and numerical experimentation were presented for forward and turning motions. For both the forward and turning gaits, the effect of accurately modeling the hydrodynamic interactions between links was demonstrated by comparing the resulting motion in both the hydrodynamically coupled and decoupled cases.

In Chapter 5, we show how to systematically develop gaits by studying properties of the hydromechanical connection. But first, in Chapter 4 we consider a swimmer in Stokes flow, which despite existing in an environment on the opposite end of the Reynolds number scale, shares many similarities to a swimmer in potential flow.

## Chapter 4

## Motion in Stokes Flow

In this chapter we consider the planar motion of a deformable body through an incompressible fluid where the Reynolds number is very small. A MATLAB implementation of the model we describe in this chapter is included in Appendix B. In the limit as the Reynolds number approaches zero, inertial forces become negligible relative to viscous forces, and the incompressible Navier-Stokes equations become the Stokes equations:

$$
\begin{equation*}
\nabla p(x)=\mu \nabla^{2} u(x), \quad \nabla \cdot u=0 \tag{4.1}
\end{equation*}
$$

where $\mu$ is the kinematic viscosity, $p$ is the pressure and $u$ is the fluid velocity. We solve these equations subject to the no-slip boundary condition, $u(\xi)=U(\xi)$ for all $\xi \in L$, where $L$ is a curve defining the surface of the swimmer and $U$ is the swimmer velocity at the surface.

### 4.1 Transforming Stokes Equations

In general, Stokes equations are expressed as in Equation (4.1). The viscosity $\mu$ remains as a parameter in the system, though it can be eliminated via a change of
coordinates. Consider the following non-dimensionalization:

$$
\begin{equation*}
\nabla=\frac{1}{L} \nabla^{*}, \quad x=x^{*} L, \quad u=u^{*} \frac{U^{2} L}{\nu}, \quad t=t^{*} \frac{L^{2}}{\nu} . \tag{4.2}
\end{equation*}
$$

Then Equation (4.1) becomes

$$
\begin{equation*}
\nabla^{*} p\left(x^{*}\right)=\nabla^{* 2} u^{*}\left(x^{*}\right), \quad \nabla^{*} \cdot u^{*}=0 . \tag{4.3}
\end{equation*}
$$

One can therefore work with the Stokes equations in a non-dimensional form knowing that we can always arrive at this form via a coordinate change.

### 4.2 Stokes Flow Algorithm

The numerical algorithm employed is based on the work of Power [95]. A solution for two-dimensional Stokes flow is proposed as a function of singularity distributions over the surface and in the interior of the body.

Let the domain $\Omega$ be the exterior of the body surface defined by the closed curve $L$ in a plane, and $x=\left(x_{1}, x_{2}\right) \in \Omega$. Consider the reduced Stokes equations:

$$
\begin{equation*}
\frac{\partial p(x)}{\partial x_{i}}=\frac{\partial^{2} u_{i}(x)}{\partial x_{j}^{2}}, \quad \frac{\partial u_{i}(x)}{\partial x_{i}}=0 \tag{4.4}
\end{equation*}
$$

where the subscript indices $i$ and $j$ specify the $x$ and $y$ components, respectively, with the no slip condition at the surface boundary $L$ :

$$
\begin{equation*}
u_{i}(\xi)=U_{i}(\xi) \text { for all } \xi \in L \tag{4.5}
\end{equation*}
$$

where $U_{i}$ is the velocity of the surface of the swimmer and at infinity

$$
\begin{equation*}
u_{i}-A_{i} \ln |x|=O(1), \quad p=o(1) \text { as }|x| \rightarrow \infty \tag{4.6}
\end{equation*}
$$

where $A$ is the given drag force on the body.
Before we proceed further, we briefly focus on the logarithmic divergence of the velocity at infinity in Equation (4.6). In two-dimensional Stokes flow, the velocity field resulting from a point source of strength $A$ acting on the fluid is represented by a Stokeslet singularity. In $(r, \theta)$ polar coordinates, this singularity has the stream function

$$
\begin{equation*}
\psi_{\text {Stokeslet }}=\frac{A}{4 \pi \mu} r \ln r \sin \theta \tag{4.7}
\end{equation*}
$$

Differentiating $\psi$ with respect to $r$, we get the radial velocity component of the fluid

$$
\begin{equation*}
u_{r}=\frac{\partial \psi_{\text {Stokeslet }}}{\partial r}=\frac{A}{4 \pi \mu}(\ln r+1) \sin \theta . \tag{4.8}
\end{equation*}
$$

From a distance, a body with a non-zero net force will appear like a Stokeslet singularity, and the velocity at infinity will have a logarithmic divergence as in Equation (4.8). Sir George Gabriel Stokes, the founder of modern hydrodynamics, was able to find the solution for uniform flow past a sphere, but failed to solve for the uniform flow past a stationary cylinder in two dimensions. The situation in which a finite external force applied to an infinite cylindrical body generates a flow with diverging speed at $r=\infty$ is known as the Stokes paradox. This seeming paradox, which was initially resolved by Oseen in 1910 [89] and more rigorously half a century later by Finn [30] and Smith [106], is due to the fact that Stokes equations are valid near a boundary, though not farther away where it is no longer valid to neglect the convective terms.

Since the swimmer experiences no external forces and cannot generate net forces or moments on itself through its self-deformation [105], we have $A=0$ and we avoid this issue completely.

Returning again to Equations (4.4)-(4.6), Hsiao and Kress [49] found a solution to this problem by representing the velocity field as a combination of single and double layer potentials over the boundary of the body. These single and double layer
potentials are surface integrals with corresponding density distributions where the kernels are the Stokeslet and stresslet (the symmetric component of a Stokes doublet) - fundamental singular solutions to the Stokes equation. The force and torque on the body are functions of only the single layer potential, however numerical issues arise due to the logarithmic nature of its kernel. The double layer potential is the integral in Equation (4.10) and the stresslet kernel is given by Equation (4.11). See [49] for further details.

Simply eliminating the single layer potential is not an option in the general case since a double-layer potential solution can represent only those systems with zero torque and force on the body. Further, a double-layer potential can only represent solutions where the surface velocity $U(\xi)$ satisfies $\int_{L} U_{i}(y) a_{i}^{k}(y) \mathrm{d} \sigma_{y}=0$ for $k=1,2,3$ (refer to [95] for details). Instead, Power found an equivalent solution by replacing the single layer potential with a pair of singularities inside the body - a Stokeslet and a rotlet - to generate a force and torque on the body, respectively. A Stokeslet and rotlet are the singularities corresponding to a unit point force and torque in Stokes flow, respectively. The stream function for a Stokeslet was given in Equation (4.7), while the stream function for a line rotlet of strength $w$ is

$$
\begin{equation*}
\psi_{\text {rotlet }}=\frac{w}{4 \pi \mu} \ln (r) \tag{4.9}
\end{equation*}
$$

where $r$ is the distance from the rotlet.
Power presents a solution for the velocity and pressure field as a function of a double layer potential of density $\phi$, a two-dimensional Stokeslet at the origin of strength equal to $A$, a two-dimensional rotlet at the origin with strength $w$, and an unknown constant vector $\alpha$ :

$$
\begin{equation*}
u_{i}=A_{j} u_{i}^{j}(x)+\int_{L} K_{i j}(x, y) \phi_{j}(y) d \sigma_{y}+\frac{\epsilon_{i j k} w_{j} x_{k}}{R^{2}}-\alpha_{i} \tag{4.10}
\end{equation*}
$$

where $R=|x|$. The two-dimensional vector $x=\left(x_{1}, x_{2}\right)$ is identified with the threedimensional vector $x=\left(x_{1}, x_{2}, 0\right)$, and the term $\frac{\epsilon_{i j k} w_{j} x_{k}}{R^{2}}$ represents the velocity due to the rotlet with zero constant pressure. $K_{i j}(x, y)$ is the $i^{\text {th }}$ component of the influence coefficient at the point $x$ of the symmetric component of a Stokes doublet located at the point $y$ and oriented in the $j^{\text {th }}$ direction and is given by the expression

$$
\begin{equation*}
K_{i j}(x, y)=-\frac{1}{\pi} \frac{\left(x_{i}-y_{i}\right)\left(x_{j}-y_{j}\right)\left(x_{k}-y_{k}\right) n_{k}(y)}{r^{4}} \tag{4.11}
\end{equation*}
$$

where $n_{k}(y)$ is the outward normal vector at the point $y \in L$. The values for $\alpha=\left(\alpha_{1}, \alpha_{2}, \alpha_{3}\right)$ and $w=\left(w_{1}, w_{2}, w_{3}\right)$ are taken to depend linearly on the density distribution $\phi$ :

$$
\begin{gather*}
\alpha_{i}=\frac{1}{2 \pi} \int_{L} \phi_{j}(x) \varphi_{j}^{i}(x) d \sigma_{x} \quad \text { for } i=1,2 \quad \text { and } \quad \alpha_{3}=0,  \tag{4.12}\\
w_{1}=w_{2}=0 \quad \text { and } \quad w_{3}=\frac{1}{2 \pi} \int_{L} \phi_{j}(x) \varphi_{j}^{3}(x) d \sigma_{x}, \tag{4.13}
\end{gather*}
$$

where $\varphi^{i}$ represents the rigid body motion of the fluid and is given by

$$
\begin{equation*}
\varphi^{i}=\left(\delta_{1 i}, \delta_{2 i}, 0\right) \quad \text { for } \quad i=1,2 \quad \text { and } \quad \varphi^{3}=\left(x_{2},-x_{1}, 0\right) \tag{4.14}
\end{equation*}
$$

and $\delta_{i j}$ is the Kronecker delta: $\delta_{i i}=1, \delta_{i j}=0(i \neq j)$.
Although a Stokelet produces zero net torque, the total force exerted on the body is equal to the strength of the Stokeslet. On the other hand, a rotlet enclosed by a curve produces a torque equal to its strength on that body, but zero net force. As noted previously, a double-layer distribution produces zero force or torque on the body. Thus the force and torque acting on the curve $L$ are given by:

$$
\begin{equation*}
F_{i}=A_{i}, \quad T=w_{3}=\frac{1}{2 \pi} \int_{L} \phi_{j}(x) \varphi_{j}^{3}(x) d \sigma_{x} \tag{4.15}
\end{equation*}
$$

By applying the boundary condition (4.5) to Equations (4.10)-(4.13) Power found the linear system of equations for the unknown vector density $\left(\phi_{1}, \phi_{2}, 0\right)$ on $L$ :

$$
\begin{equation*}
U_{i}(\xi)-A_{j} u_{i}^{j}(\xi)=-\frac{1}{2} \phi_{i}(\xi)+\int_{L} K_{i j}(\xi, y) \phi_{j}(y) d \sigma_{y}+\frac{\epsilon_{i j k} w_{3} \delta_{3 j} \xi_{k}}{R_{\xi}^{2}}-\alpha_{i}, \quad \xi \in L \tag{4.16}
\end{equation*}
$$

where $R_{\xi}$ is the distance from the origin to $\xi . A$ is the given total drag force on the body which we will take equal to zero since there are no external forces present and a self-deforming body in Stokes flow is unable to generate net forces on itself.

### 4.3 Numerical Solution

The linear system of Equation (4.16) is solved by discretizing the body surface $L$ into $N$ straight-line panels each with a constant double layer distribution. We denote the integral of the $i^{\text {th }}$ component of $\phi$ along the $k^{t h}$ panel as $\phi_{i}^{k}$, and the length of the $k^{t h}$ panel is $a^{k}$. Subscript indices correspond to spatial dimensions while superscript indices correspond to panel numbers.

Equation (4.16) can be expressed as a sum over the panels as:

$$
\begin{equation*}
U_{i}^{k}=-\frac{\phi_{i}^{k}}{a^{k}}+\mathbb{K}_{i j}^{k l} \phi_{j}^{l}+\frac{\epsilon_{i j n} w_{j} x_{n}^{k}}{R^{k^{2}}}-\alpha_{i} \tag{4.17}
\end{equation*}
$$

where $\mathbb{K}_{i j}^{k l}$ is the influence coefficient in the $i^{t h}$ direction at the control point (usually specified to be the geometric center) of panel $k$ due to the $j^{\text {th }}$ direction component of a constant double layer distribution over panel $l . U_{i}^{k}$ is the $i^{t h}$ component of velocity at the control point of the $k^{t h}$ panel, $x_{n}^{k}$ is the $n^{t h}$ coordinate of the control point at panel $k$.

To determine $\mathbb{K}_{i j}^{k l}$, we first consider a panel of length $\Delta^{k}$, centered at $\left(y_{1}, y_{2}\right)=$ $(0,0)$, and oriented along the abscissa with constant Stokes doublet (also known as
stresslet) distribution. Since $n_{1}(y)=0$ and $n_{2}(y)=1$, from Equation (4.11) we have:

$$
\begin{align*}
K_{11}(x, y) & =-\frac{1}{\pi} \frac{\left(x_{1}-y_{1}\right)^{2}\left(x_{2}-y_{2}\right)}{r^{4}}  \tag{4.18}\\
K_{12}(x, y)=K_{21}(x, y) & =-\frac{1}{\pi} \frac{\left(x_{1}-y_{1}\right)\left(x_{2}-y_{2}\right)^{2}}{r^{4}}  \tag{4.19}\\
K_{22}(x, y) & =-\frac{1}{\pi} \frac{\left(x_{2}-y_{2}\right)^{3}}{r^{4}} \tag{4.20}
\end{align*}
$$

where $r^{4}=\left(\left(x_{1}-y_{1}\right)^{2}+\left(x_{2}-y_{2}\right)^{2}\right)^{2}$. To find the influence at $\left(x_{1}, x_{2}\right)$ due to a constant strength stresslet distribution over panel $k$, we integrate (4.18)-(4.20) over the length of the panel:

$$
\begin{align*}
I_{11}(x) & =\int_{-\frac{\Delta^{k}}{2}}^{\frac{\Delta^{k}}{2}} K_{11}(x, \eta) d \eta=\int_{-\frac{\Delta^{k}}{2}}^{\frac{\Delta^{k}}{2}}-\frac{1}{\pi} \frac{\left(x_{1}-\eta_{1}\right)^{2} x_{2}}{r^{4}} d \eta_{1} \\
& =-\frac{1}{2 \pi}\left(-x_{2}\left(C_{1}-C_{2}\right)+\left(A_{1}-A_{2}\right)\right)  \tag{4.21}\\
I_{12}(x)=I_{21}(x) & =\int_{-\frac{\Delta^{k}}{2}}^{\frac{\Delta^{k}}{2}} K_{12}(x, \eta) d \eta=\int_{-\frac{\Delta^{k}}{2}}^{\frac{\Delta^{k}}{2}}-\frac{1}{\pi} \frac{\left(x_{1}-\eta_{1}\right) x_{2}^{2}}{r^{4}} d \eta_{1} \\
& =-\frac{1}{2 \pi}\left(-4 x_{2}^{2}\left(B_{1}-B_{2}\right)\right)  \tag{4.22}\\
I_{22}(x) & =\int_{-\frac{\Delta^{k}}{2}}^{\frac{\Delta^{k}}{2}} K_{22}(x, \eta) d \eta=\int_{-\frac{\Delta^{k}}{2}}^{\frac{\Delta^{k}}{2}}-\frac{1}{\pi} \frac{x_{2}^{3}}{r^{4}} d \eta_{1} \\
& =-\frac{1}{2 \pi}\left(x_{2}\left(C_{1}-C_{2}\right)+\left(A_{1}-A_{2}\right)\right) \tag{4.23}
\end{align*}
$$

where

$$
\begin{align*}
& A_{1,2}=\arctan \left(\frac{x_{1} \pm\left(\Delta^{k} / 2\right)}{x_{2}}\right)  \tag{4.24}\\
& B_{1,2}=\left(\left(\Delta^{k} \pm 2 x_{1}\right)^{2}+\left(2 x_{2}\right)^{2}\right)^{-1}  \tag{4.25}\\
& C_{1,2}=2\left(2 x_{1} \pm \Delta^{k}\right) B_{1,2} . \tag{4.26}
\end{align*}
$$

In the prior three expressions, indices 1 and 2 correspond to + and - , respectively. These coefficients are expressed in a frame local to the panel. Since the panel actually
lies at an angle $\theta$ relative to the horizontal, we express the coefficients in a global coordinate frame through the transformation

$$
\left[\begin{array}{cc}
\mathbb{K}_{11} & \mathbb{K}_{12}  \tag{4.27}\\
\mathbb{K}_{21} & \mathbb{K}_{22}
\end{array}\right]=\left[\begin{array}{cc}
\cos (\theta) & -\sin (\theta) \\
\sin (\theta) & \cos (\theta)
\end{array}\right]\left[\begin{array}{ll}
I_{11} & I_{12} \\
I_{21} & I_{22}
\end{array}\right]\left[\begin{array}{cc}
\cos (\theta) & \sin (\theta) \\
-\sin (\theta) & \cos (\theta)
\end{array}\right] .
$$

Expressing this result in terms of the outward normal and tangent unit vector components associated with the panel, $\left(t_{1}, t_{2}\right)=(\cos \theta, \sin \theta)$ and $\left(n_{1}, n_{2}\right)=(-\sin \theta, \cos \theta)$, one gets

$$
\begin{align*}
\mathbb{K}_{11} & =I_{11} t_{1}^{2}+I_{22} n_{1}^{2}+2 I_{12} n_{1} t_{1}  \tag{4.28}\\
\mathbb{K}_{12}=\mathbb{K}_{21} & =I_{11} t_{1} t_{2}+I_{22} n_{1} n_{2}+I_{12}\left(t_{1} n_{2}-n_{1} t_{2}\right)  \tag{4.29}\\
\mathbb{K}_{22} & =I_{11} t_{2}^{2}+I_{22} n_{2}^{2}+2 I_{12} n_{2} t_{2} \tag{4.30}
\end{align*}
$$

Finally, Equation (4.17) can be expressed in block matrix form as

$$
\left[\begin{array}{l}
v_{1}  \tag{4.31}\\
v_{2}
\end{array}\right]=\mathbb{Q}\left[\begin{array}{l}
\phi_{1} \\
\phi_{2}
\end{array}\right]
$$

where $v_{i}=\left[\begin{array}{llll}v_{i}^{1} & v_{i}^{2} & \ldots & v_{i}^{N}\end{array}\right]^{\mathrm{T}}$ is a vector of the $i^{\text {th }}$ component of velocity at the control points, $\phi_{i}=\left[\begin{array}{llll}\phi_{i}^{1} & \phi_{i}^{2} \ldots & \phi_{i}^{N}\end{array}\right]^{\mathrm{T}}$ and $\mathbb{Q}$ is the square block matrix given by:

$$
\mathbb{Q}=\left[\begin{array}{ll}
\mathbb{K}_{11} & \mathbb{K}_{12}  \tag{4.32}\\
\mathbb{K}_{21} & \mathbb{K}_{22}
\end{array}\right]+\left[\begin{array}{c}
-y / R^{2} \\
x / R^{2}
\end{array}\right] \cdot\left[\begin{array}{ll}
-y^{T} & x^{T}
\end{array}\right]+\left[\begin{array}{cc}
\mathbf{1} \cdot \Delta^{T} & 0 \\
0 & \mathbf{1} \cdot \Delta^{T}
\end{array}\right],
$$

where 1 is a column vector of $N$ 1's, $\Delta=\left[\Delta^{1} \Delta^{2} \ldots \Delta^{N}\right]^{\mathrm{T}}, x$ and $y$ are the coordinates of the control points as column vectors, and $R=\sqrt{x^{2}+y^{2}}$ is a column vector of the distances from the origin to the point corresponding to the index number of the
coordinate.
Since all the values inside the braces of Equation (4.31) are known, the strength of the singularity distribution $\phi=\left[\phi_{1}, \phi_{2}\right]^{\mathrm{T}}$ can be solved for by imposing the no-slip boundary condition which specifies $v_{1}$ and $v_{2}$ and then inverting $\mathbb{Q}$. At this point, we have a solution for the singularity distribution on the body, which once computed allows for reconstructing the velocity field around the body. But we are interested in computing the self-propulsion of the body due to its deforming shape.

As noted by both Blake [15] and Kelly and Murray [58], the force acting on a two-dimensional, infinite cylinder in Stokes flow is proportional to the fluid velocity at infinity. Thus, there is a one-to-one relationship between the generalized force vector $\left(F_{x}, F_{y}, M\right)$ acting on the body and the unknown values $\left(\alpha_{1}, \alpha_{2}, w_{3}\right)$ in Equation (4.16). At each time step, we compute the value of ( $\alpha_{1}, \alpha_{2}, w_{3}$ ) resulting from the prescribed shape changes as well as that resulting from a rigid body motion. Since it is well known that a deforming body in a Stokes flow can not experience any net external forces or moments [105], the forces due to the two types of motions must exactly balance. This is how we determine the net motion of the body.

Let the shape of the body be specified via shape variables $s_{1}, s_{2}, \ldots, s_{r}$ with prescribed shape changes in time $\dot{s}_{1}, \dot{s}_{2}, \ldots, \dot{s}_{r}$. The double layer distributions, $\phi^{s_{1}}, \phi^{s_{2}}, \ldots, \phi^{s_{r}}$, due to each of these individual shape changes are computed by imposing the appropriate boundary condition resulting from the corresponding unit shape change. The vector $\mathcal{F}_{s}=\left(\alpha_{1, s}, \alpha_{2, s}, w_{3, s}\right)^{\mathrm{T}}$, due to the shape changes can be expressed as the product of a matrix $\mathbb{A}$ and a vector of the shape change velocities $\dot{s}$ :

$$
\mathcal{F}_{s}=\left[\begin{array}{c}
\alpha_{1, s}  \tag{4.33}\\
\alpha_{2, s} \\
w_{3, s}
\end{array}\right]=\mathbb{A}\left[\begin{array}{c}
\dot{s}_{1} \\
\dot{s}_{2} \\
\vdots \\
\dot{s}_{r}
\end{array}\right]
$$

where

$$
\mathbb{A}=\left[\begin{array}{cccc}
\phi_{1 k}^{s_{1}} \Delta^{k} & \phi_{1 k}^{s_{2}} \Delta^{k} & \cdots & \phi_{1 k}^{s_{r}} \Delta^{k}  \tag{4.34}\\
\phi_{2 k}^{s_{1}} \Delta^{k} & \phi_{2 k}^{s_{2}} \Delta^{k} & \cdots & \phi_{2 k}^{s_{r}} \Delta^{k} \\
\left(\phi_{2 k}^{s_{1}} x_{k}-\phi_{1 k}^{s_{1}} y_{k}\right) \Delta^{k} & \left(\phi_{2 k}^{s_{2}} x_{k}-\phi_{1 k}^{s_{2}} y_{k}\right) \Delta^{k} & \cdots & \left(\phi_{2 k}^{s_{r}} x_{k}-\phi_{1 k}^{s_{r}} y_{k}\right) \Delta^{k}
\end{array}\right]
$$

We can also compute the values of $\left(\alpha_{1}, \alpha_{2}, w_{3}\right)$ for the motion of the purely rigid body with fixed shape. In this case the solution is particularly simple: $\mathcal{F}_{g}=\mathbb{B} \xi$. More explicitly,

$$
\mathcal{F}_{g}=\left[\begin{array}{c}
\alpha_{1, g}  \tag{4.35}\\
\alpha_{2, g} \\
w_{3, g}
\end{array}\right]=\left[\begin{array}{lll}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & \tau
\end{array}\right]\left[\begin{array}{l}
u \\
v \\
\omega
\end{array}\right]
$$

where $\tau$ is the value of $w_{3}$ resulting from a unit rotational velocity on the body, $\mathbb{B}=\operatorname{diag}(1,1, \tau)$, and $\xi=(u, v, \omega)^{\mathrm{T}}$ is the vector of translational and rotational velocities expressed with respect to a body fixed frame. If $g$ is an element of the group representing the orientation and position of the body, the body velocity can be expressed as $\xi=g^{-1} \dot{g}$ and parameterized with $(u, v, \omega)$. Since a self-deforming body in Stokes flow cannot exert any net forces or moments upon itself, the forces due to a purely rigid body motion must exactly balance those due to the shape change, hence $\mathcal{F}_{g}+\mathcal{F}_{s}=0$. Thus the expression for the motion of the body resulting from internal shape changes is

$$
\begin{equation*}
\xi=g^{-1} \dot{g}=-\mathcal{A} \dot{s}, \tag{4.36}
\end{equation*}
$$

where $\mathcal{A}=\mathbb{B}^{-1} \mathbb{A}$. The matrices $\mathbb{A}$ and $\mathbb{B}$ are functions of the shape, so they must be computed at each time step in order to determine the body velocity resulting from the shape change. The net motion is found by integrating this velocity in time.

Note that Equation (4.36) has the same form as the result in potential flow,

Equation (3.53). As in the potential flow case, it is apparent from (4.36) that when the body stops changing shape $\left(\dot{s}_{i}=0\right.$ for all $i$ ), the motion of the body will also instantly cease.

### 4.4 Code Validation

As validation of the numerical implementation of Power's algorithm, we consider two cases: a self-propelling deforming cylinder presented by Shapere and Wilczek [105] and Purcell's swimmer studied by Becker et al. [12].

### 4.4.1 Squirming Cylinder

We first consider the case of a self-propelling deforming body studied by Shapere and Wilczek [105].

The cylinder surface $S$ with finite size deformations is parameterized as:

$$
\begin{equation*}
S(\sigma, t)=s_{0}(t) \sigma+s_{2}(t) \sigma^{-1}+s_{3} \sigma^{-2} \tag{4.37}
\end{equation*}
$$

where $\sigma=\mathrm{e}^{\mathrm{i} \theta}$ are complex coordinates on the unit circle where $\theta$ is the angle from the horizontal, and $s_{i}$ correspond to the shape coefficients. Shapere and Wilczek prescribe the shape variables as follows:

$$
\begin{align*}
& s_{0}=1  \tag{4.38}\\
& s_{2}=0.3 \cos (t)+i 0.0015 \sin (t)  \tag{4.39}\\
& s_{3}=-0.3 \sin (t)+i 0.0015 \cos (t) \tag{4.40}
\end{align*}
$$

Hence the $(x, y)$ coordinates of the swimmer geometry may be expressed explicitly as
a function of time and angle:

$$
\begin{align*}
& \left(S_{x}(\theta, t), S_{y}(\theta, t)\right)= \\
& (\cos \theta+\cos (t)(0.3 \cos \theta+0.0015 \sin 2 \theta)+\sin (t)(0.0015 \sin \theta-0.3 \cos 2 \theta  \tag{4.41}\\
& \sin \theta+\cos (t)(0.0015 \cos 2 \theta-0.3 \sin \theta)+\sin (t)(0.0015 \cos \theta+0.3 \sin 2 \theta)
\end{align*}
$$

By conformally mapping the deformations to the unit circle, Shapere and Wilczek found analytical expressions for the velocity components of the swimmer in terms of the shape coefficients. For the shape deformations specified by (4.41), the forward and lateral translational $(u, v)$ and rotational $\omega$ body velocity components as a function of time are:

$$
\begin{align*}
& u(t)=a^{2} \sin (t)+b^{2} \cos (t)  \tag{4.42}\\
& v(t)=0  \tag{4.43}\\
& \omega(t)=-\frac{3 a b}{1+2 a^{2}+b^{2}+\left(-a^{2}+b^{2}\right) \cos ^{2}(t)} . \tag{4.44}
\end{align*}
$$

Snapshots of the swimmer's shape during one cycle are shown in Figure 4.1. Figure 4.2 shows the three velocity components computed with the numerical code for a body discretized into 200 panels compared to the exact solution found by Shapere and Wilzcek.


Figure 4.1: Snapshots of the Shapere and Wilczek [105] swimmer shape during one cycle. The analytical expressions for the shape of the swimmer are given by Equation (4.41).


Figure 4.2: Comparison of code with analytic result from Shapere and Wilczek [105]. The three body velocity components $u, v$ and $\omega$ are plotted versus time as lines for the Shapere and Wilczek exact result and as markers for the numerically computed values.

### 4.4.2 Purcell Swimmer

Since we wish to study how a three-link swimmer in Stokes flow compares to one in potential flow, we also validate our code against a solution for Purcell's swimmer (see Figure 4.4) - a simple three-link, two joint body proposed by Edward M. Purcell [98] as the simplest swimmer able to achieve motion in Stokes flow. Becker et al. [12] applied slender body hydrodynamics and symmetry arguments to determine the motion of such a swimmer. They consider the case where the ratio of the middle to outer link length is $\eta=2$, and the maximum angle of each link is $\frac{\pi}{3}$. The gait is as follows: Initially, $\theta_{1}=\frac{\pi}{3}$ while $\theta_{2}$ is swept from $-\frac{\pi}{3}$ to $\frac{\pi}{3}$. Next, $\theta_{2}$ is held fixed while $\theta_{1}$ is varied from $\frac{\pi}{3}$ to $-\frac{\pi}{3}$. Next, while $\theta_{1}$ is fixed, $\theta_{2}$ is swept from $\frac{\pi}{3}$ to $-\frac{\pi}{3}$. Finally, holding $\theta_{2}$ fixed, $\theta_{1}$ is returned back to $\frac{\pi}{3}$.

The motion of our exact numerical simulation, where the aspect ratio of the elliptic outer links is $40: 1$ is compared to the analytical result found by Becker et al. [12] in Figure 4.3. The two results match up reasonably well. Recall that Becker et al. assumed slender body hydrodynamics and the effects of one link component on the


Figure 4.3: Comparison of code with analytic result from Becker, Koehler and Stone [12]. The numerically computed $x$ and $y$ displacement of the center of the middle link, normalized by the length of the outer links, $a$ are plotted as blue and red open circles, respectively. The displacement of the middle of the center link found by Becker et al. appear as dash-dotted lines on the plot. The solid lines correspond to the displacement of one of the link hinges determined by Becker et al. The original analytic result is reproduced from Becker et al.
other links is not accounted for in their model. Still, the results are qualitatively similar. We note that for different link aspect ratios than $40: 1$, the results are qualitatively similar to the results in Figure 4.3, but the displacement is quantitatively different.

### 4.5 Example Gaits

We consider the motion of Purcell's swimmer [98] - a planar, deformable, symmetric, three link swimmer - as shown in Figure 4.4. The middle segment is twice the
length $(b)$ of the two outer links $(a)$. The length:width aspect ratio of the outer links is $10: 1$ and the ends of the links are defined by ellipses with the same aspect ratio. The middle link is the same width as the outer links. The shape of the body is specified by the two joint angles relative to the extended straight configuration. The sign of the angles is defined such that a "C" configuration of the body corresponds to one positive and one negative angle while an " S " configuration corresponds to two like-signed angles. From the numerical model, we are able to determine the motion of


Figure 4.4: The three-link Purcell swimmer with $\theta_{1}=0.8$ and $\theta_{2}=0.4$.
the body as a result of prescribed cyclic shape changes. Here we present gaits found by a combination of intuition and heuristics to achieve forward and turning motions. To ensure sufficient convergence in these examples, the body is discretized into 678 panels (due to the nature of the discretization method of the code) and each cyclic gait is broken up into 50 time steps.

The ratio between the length of the middle and outer links is $\eta=2$ and the aspect ratio of the elliptical link ends is 10:1.

### 4.5.1 Forward Gait

Although the geometry is slightly different, for the sake of comparison, we show the motion resulting from the same gaits found for the potential flow swimmer. One family of gaits which achieves forward motion corresponds to those paths in $\theta_{1}-\theta_{2}$ shape space which are circles centered about the origin as defined by Equations (3.40) and (3.41). As an example, we choose the same gait as specified by Equations (3.42) and (3.43), repeated here for convenience:

$$
\begin{align*}
& \theta_{1}=1.5 \cos \left(t-\frac{\pi}{4}\right)  \tag{4.45}\\
& \theta_{2}=1.5 \sin \left(t-\frac{\pi}{4}\right) \tag{4.46}
\end{align*}
$$

The resulting translation and rotation is shown in Figures 4.5(a) and 4.5(b). Snapshots of the swimmer shape during the gait are shown in Figure 4.6. Although the amplitude of rotation during the gait is comparable to that in Figure 3.4(b), the net translation is roughly an order of magnitude less than that in Figure 3.4(a). The net rotation is 0 in both cases.

### 4.5.2 Turning Gait

As in potential flow, one family of gaits which results in net rotation are those specified by Equations (3.44) and (3.41). They define circular paths in shape space centered along the $\theta_{1}=-\theta_{2}$ diagonal. Again, for the sake of comparison, we prescribe the same example turning gait to the Stokes swimmer as demonstrated for the potential flow swimmer. The angles of the joints are specified by Equations (4.47) and (4.48).

$$
\begin{align*}
& \theta_{1}=-0.8+0.8 \cos \left(t-\frac{\pi}{4}\right)  \tag{4.47}\\
& \theta_{2}=0.8+0.8 \sin \left(t-\frac{\pi}{4}\right) \tag{4.48}
\end{align*}
$$



Figure 4.5: Stokes flow swimmer: Forward gait example. (a) The translation of the center of the middle link non-dimensionalized by middle link length, $a$. The black and blue dots indicate the initial and final positions, respectively; (b) the angle of the middle link versus time during the forward swimming gait specified by Equations (4.45) and (4.46). Snapshots of the swimmer motion are shown in Figure 4.6.


Figure 4.6: Stokes flow swimmer: Forward gait snapshots. The swimmer shape, position and orientation is shown for five instance of time during the gait specified by Equations (4.45) and (4.46). The blue curve is the same as in Figure 4.5(a) and shows the path traveled by the center of the middle link.

The translation and rotation resulting from this gait is shown in Figures 4.7(a) and 4.7(b), respectively while sapshots of the swimmer shape during the gait are shown in Figure 4.8. As in the previous example, the motion is qualitatively similar to that of the potential flow swimmer shown in Figures 3.6(a) and 3.6(b). Quantitatively however, the net translation of $0.045 a$ and net rotation of 0.075 radians are both an order of magnitude less than the corresponding translation of $0.95 l$ and rotation of 0.59 radians in the potential flow case.


Figure 4.7: Stokes flow swimmer: Turning gait example. (a) The translation of the center of the middle link non-dimensionalized by middle link length, $a$. The black and blue dots indicate the initial and final positions, respectively; (b) the angle of the middle link versus time during the turning swimming gait specified by Equations (4.47) and (4.48). Snapshots of the swimmer motion are shown in Figure 4.8.


Figure 4.8: Stokes flow swimmer: Turning gait snapshots. The swimmer shape, position and orientation is shown for five instance of time during the gait specified by Equations (4.47) and (4.48). The blue curve is the same as in Figure 4.7(a) and shows the path traveled by the center of the middle link.

### 4.6 Summary

We have implemented and validated a numerical boundary element code for solving the motion of a deformable body in Stokes flow. Initial examples suggest - perhaps confirming intuition - that for a given input, there is less net motion through a highly viscous fluid than through an inviscid fluid. In Chapter 5, we present a more systematic approach for developing gaits for swimmers in both Stokes and potential flows. The plots of the curvature of the connection will provide numerical confirmation that motion through a highly viscous fluid is, as intuition would suggest, more difficult than through an inviscid fluid.

## Chapter 5

## Control of Reversible Systems

### 5.1 Reversible Swimming Systems

For deformable swimmers in both potential and Stokes flow, the net motion due to internal shape changes is independent of time. In other words, regardless of how quickly or slowly the shape changes, the resulting motion is the same. Equivalently, if the prescribed shape changes are reversed, the swimmer will return to its original position and orientation.

We caution that these results are not directly representative of the wide range of swimming that occurs between the extremes of inviscid and highly viscous flow regimes. Neither of these models accounts for the effects of vorticity which any 'real' swimmer would shed from its surface as it moves relative to its surrounding fluid. Indeed, a fish that simply flaps its tail back and forth is able to propel itself forward in a real fluid, but not in the idealized cases of potential and Stokes flow we present here.

Still, these results are instructive for understanding certain types of swimmers. On one extreme, the Stokes flow model could improve understanding of how microor nano-robots might propel through the bloodstream to deliver targeted drugs for
a patient. On the other hand, the potential flow model is valid when inertial forces dominate over viscous forces and may provide insights into how fish or other swimming animals achieve turning motions under high Reynolds number conditions. Such high Reynolds number examples include the sea lion, which is known to achieve unpowered turns with a turning radius as small as 0.09 body lengths [34]. Turning radii as small as 0.08 body lengths have similarly been observed for bottlenose dolphins [88]. Likewise, minimum turning radii for rainbow trout and smallmouth bass are 0.18 and 0.11 body lengths, respectively [124]. By demonstrating that such kinds of turning motions are possible in the absence of vorticity, we may shed light on the mechanisms that dominate in the turning motions of biological swimmers, which may and in turn suggest control strategies for flexible hull vehicles.

### 5.2 Holonomy in semidirect product groups

We wish to understand how our models of potential and Stokes flow swimmers achieve locomotion due to internal shape changes. In both cases, the internal shapes changes parameterized by angles $\theta_{i}$ are related to the body velocity of the swimmer through the connection, $\mathcal{A}$. Note that the equations of motion for the potential and Stokes flow swimmers, (3.56) and (4.36), respectively, can both be expressed in the form:

$$
\begin{equation*}
\dot{s}=-s \mathcal{A}(\theta) \dot{\theta} \tag{5.1}
\end{equation*}
$$

where $s$ is an element of a Lie Group $S$. It is well known (see $\S 2.4$ ) that if $s$ is an element of an Abelian Lie group $S$, then for a closed path in shape space given by $\theta(t), t \in[0, T]$, the solution to (5.1) is:

$$
\begin{equation*}
s(T)=s(0) \exp \left(-\int_{0}^{T} \mathcal{A}(\theta(\tau)) \dot{\theta}(\tau) \mathrm{d} \tau\right)=s(0) \exp \left(-\iint_{\mathrm{C}} \mathrm{~d} \mathcal{A}\right) \tag{5.2}
\end{equation*}
$$

where C is the area in shape space enclosed by the path $\theta(t)$. Physically, $s(T)$ could correspond to the final position or orientation of the swimmer. The net translation or rotation would be the difference between $s(T)$ and $s(0)$. This is a powerful result since it greatly simplifies the problem of generating gaits to achieve a desired motion. To use this result one must numerically determine in advance the value of the exterior derivative of the connection (which in the Abelian case is equivalent to the curvature) over the shape space over a sufficiently fine grid. The process of gait generation is reduced to finding paths in shape space that enclose a volume of curvature such that $\exp \left(-\iint_{\mathrm{C}} \mathrm{d} \mathcal{A}\right)$ is equal to the desired holonomy.

Unfortunately, the case when $S$ is non-Abelian is less satisfying since the solution to (5.1) is considerably more complicated (see [105] for details). Since Stokes' theorem cannot be applied, the result cannot be expressed explicitly as an integral over an area as in (5.2). This is the case for motion in the plane, since $S=S E(2)$ is non-Abelian. However, since $S E(2)$ can be expressed as the semidirect product $S O(2) \subseteq \mathbb{R}^{2}$ (see $\S 2.5$ for details), it is possible to recover the result in equation (5.2) for the $S O(2)$ subgroup (but not for the $\mathbb{R}^{2}$ group).

Recall that if $G$ is a Lie group that acts on a vector space $V$, one can define the semidirect product $G(S) V$ as the usual product, with the group operation $\left(g_{1}, v_{1}\right)\left(g_{2}, v_{2}\right)=\left(g_{1} g_{2}, v_{1}+g_{1} v_{2}\right)$, where $g_{1}, g_{2} \in G$ and $v_{1}, v_{2} \in V$. We use $e$ and 0 to denote the identity elements of $G$ and $V$, respectively. The Lie algebras corresponding to the various groups are $\mathfrak{g}=T_{e} G, V=T_{e} V$ and $\mathfrak{s}=T_{e} S$.

Theorem 5.2.1. Let $S=G$ S $V$, where $G$ is Abelian. Consider a closed curve $\theta(t) \in Q$ for $t \in[0, T]$, and let $\mathcal{A}: T Q \rightarrow \mathfrak{s}$ be a principal connection with components $\mathcal{A}_{\mathfrak{g}}: T Q \rightarrow \mathfrak{g}$ and $\mathcal{A}_{V}: T Q \rightarrow V$. Then if $s(t)=(g(t), v(t)) \in S$ satisfies

$$
\begin{equation*}
\dot{s}=-s \mathcal{A}(\theta) \dot{\theta} \tag{5.3}
\end{equation*}
$$

with $s(0)=(e, 0)$, then

$$
\begin{equation*}
\text { holonomy }_{G}:=g(T)=\exp \left(-\int_{\partial C} \mathcal{A}_{\mathfrak{g}} d \theta\right)=\exp \left(-\iint_{C} \mathrm{~d} \mathcal{A}_{\mathfrak{g}}\right) . \tag{5.4}
\end{equation*}
$$

Proof. Since $\theta(t)$ is given, we may rewrite (5.3) as $\dot{s}=-s \xi(t)$ where $\xi(t)=\mathcal{A}(\theta) \dot{\theta}(t)$ is given. Writing $\xi=\left(\xi_{g}, \xi_{v}\right)$, where $\xi_{g} \in \mathfrak{g}$ and $\xi_{v} \in V$, (5.3) becomes (see [76])

$$
\begin{align*}
& \dot{g}=-T_{e} L_{g} \xi_{g}  \tag{5.5}\\
& \dot{v}=-\rho(g) \xi_{v}, \tag{5.6}
\end{align*}
$$

where $\rho$ is a left representation of $G$ on $V$. Clearly, equation (5.5) is decoupled from (5.6), and since $G$ is Abelian, the solution to (5.5) with $g(0)=e$ is (see $\S 2.4$ )

$$
\begin{equation*}
g(T)=\exp \left(-\int_{0}^{T} \xi_{g}(\tau) d \tau\right) \tag{5.7}
\end{equation*}
$$

The holonomy of $S$ in the $G$ component is then

$$
\text { holonomy }_{G}:=g(T)=\exp \left(-\int_{\partial \mathrm{C}} \mathcal{A}_{\mathfrak{g}} d \theta\right)=\exp \left(-\iint_{\mathrm{C}} \mathrm{~d} \mathcal{A}_{\mathfrak{g}}\right)
$$

where C is the area in shape space enclosed by the curve $\partial \mathrm{C}$ which is defined by $\theta(t)$ and the last equality is by Stokes' theorem.

Although no similarly insightful result can be found for equation (5.6) and the motion corresponding to the vector space component of the group, this result is used to generate gaits for the Abelian component of the semidirect product group, which in the case of rigid planar motions correspond to turning maneuvers. Later in this chapter, Figures 5.13 and 5.14 illustrate why it is not possible to find an analogous result for the motion in the vector space component of a semidirect product group.

### 5.3 Curvature of the Connection

For the fish-like bodies depicted in Figures 3.1 and 4.4, the shape space $Q$ is parameterized by $\left(\theta_{1}, \theta_{2}\right)$, so the local connection $\mathcal{A}\left(\theta_{1}, \theta_{2}\right): T_{\left(\theta_{1}, \theta_{2}\right)} Q \rightarrow \mathfrak{s e}(2)$ is a Lie-algebra-valued one-form on $Q$, which may be written as

$$
\begin{equation*}
\mathcal{A}\left(\theta_{1}, \theta_{2}\right)=f\left(\theta_{1}, \theta_{2}\right) \mathrm{d} \theta_{1}+g\left(\theta_{1}, \theta_{2}\right) \mathrm{d} \theta_{2} \tag{5.8}
\end{equation*}
$$

where $f, g: T Q \rightarrow \mathfrak{s e}(2)$ depend on the shape of the body, and may be computed numerically. The curvature $F: T Q \times T Q \rightarrow \mathfrak{s e}(2)$ is then a Lie-algebra valued two-form on Q, (see (2.18)):

$$
\begin{equation*}
F=\left[\left(\frac{\partial g}{\partial \theta_{1}}-\frac{\partial f}{\partial \theta_{2}}\right)-[f, g]\right] \mathrm{d} \theta_{1} \wedge \mathrm{~d} \theta_{2} \tag{5.9}
\end{equation*}
$$

The curvature maps two tangent vectors into a vector of velocities. Physically, at every swimmer shape parameterized by the joint angles $\left(\theta_{1}, \theta_{2}\right)$, when given a pair of joint angle velocities ( $\dot{\theta}_{1}, \dot{\theta}_{2}$ ), the curvature returns the corresponding resulting body velocity $(\omega, u, v)$. The translational velocity components, denoted $u$ and $v$, correspond to motion parallel and perpendicular to the middle link or segment, respectively, while $\omega$ corresponds to the rotational velocity. More explicitly, if we let $\xi=(\omega, u, v)=$ $\left(\xi_{1}, \xi_{2}, \xi_{3}\right)$, then we may write $\xi_{i}=\sum_{j=1}^{2} F_{i j} \dot{\theta}_{j}$, where $F_{i j}$ are functions of $\left(\theta_{1}, \theta_{2}\right)$ and are given in (2.18). We introduce the notation $\left(F_{1}, F_{2}, F_{3}\right)=\left(F_{\omega}, F_{u}, F_{v}\right)$ where $F_{i}=\sum_{j=1}^{2} F_{i j}$. We refer to $F_{i}$ as the three components of the curvature. As an example, $F_{\omega}\left(\theta_{1}, \theta_{2}\right)$ physically corresponds to the instantaneous rotational velocity that the swimmer with shape parameterized by $\left(\theta_{1}, \theta_{2}\right)$ would experience if both joints were moved at unit speed.

The three curvature components, $F_{\omega}, F_{u}, F_{v}$, are computed numerically on a grid in the $\left(\theta_{1}, \theta_{2}\right)$ plane and plotted in Figure 5.1 for the potential flow swimmer and
in Figure 5.2 for the Stokes flow swimmer. For the potential flow case, the plots correspond to a fish-like body as depicted in Figure 3.1 where $a=10, c=2$, the aspect ratio of the ellipses is 10 and each link is discretized into 50 panels. The geometry of the Stokes flow swimmer is depicted in Figure 4.4 where $\eta=2$ and the aspect ratio of the outer links is 10 .

### 5.4 Gait generation

Since $S O(2)$ is Abelian and $S E(2)=S O(2) \subseteq \mathbb{R}^{2}$, one can use the $\omega$ component of the curvature plot, $F_{\omega}$, to develop finite-amplitude turning gaits, by applying Theorem 5.2.1. The $\omega$-component plot is shown enlarged in Figure 5.3. Note that for both potential and Stokes flow the regions of largest curvature occur in two opposite corners of the shape space. These regions correspond to the fish configuration where the joints are bent in a ' C '-shape. The most "efficient" turning gaits will enclose these regions of high curvature. Intuitively, this makes sense as one would expect a turning fish to coil its body into a ' C '-shape to minimize the inertial resistance of the fluid as it turns. Likewise, a fish in an extended or 'S'-shape configuration trying to rotate would encounter larger inertial forces and would not be expected to turn easily. Furthermore, the scale of the curvature map plots indicate that turning in potential flow is an order of magnitude easier than in Stokes flow.

The procedure for gait-generation for the Abelian subgroup component is straightforward:

1. Choose the desired Abelian Lie subgroup element of the semidirect product group (here, the desired net rotation), $g(T)$.
2. Determine the corresponding Lie algebra element by applying the logarithm map.


Figure 5.1: Curvature components of potential flow swimmer. (a) $F_{\omega}$, (b) $F_{u}$, and (c) $F_{v}$ curvature components as a function of the shape variables $\left(\theta_{1}, \theta_{2}\right)$ for the three-link swimmer in potential flow with geometry depicted in Figure 3.1. The units are $\frac{\mathrm{rad}}{\mathrm{rad}^{2}}$ for (a) and $\frac{l}{\mathrm{rad}^{2}}$ for (b) and (c). See $\S 5.3$ for details on the curvature components. Here, $a=10, e=1, c=2$ and (b) and (c) are non-dimensionalized by $l=a+e$. Note that for clarity, the view in (c) is from the opposite direction in shape space.


Figure 5.2: Curvature components of Stokes flow swimmer. (a) $F_{\omega}$, (b) $F_{u}$, and (c) $F_{v}$ curvature components as a function of the shape variables $\left(\theta_{1}, \theta_{2}\right)$ for the threelink swimmer in Stokes flow with geometry depicted in Figure 4.4. The units are $\frac{\mathrm{rad}}{\mathrm{rad}^{2}}$ for (a) and $\frac{b}{\mathrm{rad}^{2}}$ for (b) and (c). See $\S 5.3$ for details on the curvature components. Here, $\eta=2$, the aspect ratio of the outer links is 10 and (b) and (c) are non-dimensionalized by outer link length $b$.


Figure 5.3: $F_{\omega}$ curvature component for potential and Stokes flow swimmers. Closeup view of $F_{\omega}$ curvature component for (a) potential flow and (b) Stokes flow cases described in Figures 5.1 and 5.2. The units for (a) and (b) are $\frac{\mathrm{rad}}{\mathrm{rad}^{2}}$.
3. Find a path in shape space that encloses a volume of the curvature map equal to the negative of the value of the Lie algebra element found in the previous step (due to the minus sign in the right hand side of (5.4)).

Note that, since the net rotation depends only on the enclosed area, the initial shape configuration need not be a point on the path, as shown in Figure 5.6.

We will only require the rotation component of the logarithm map given by equation (2.6). For this component of the motion, the exponential map and its inverse, the logarithm map, are simply the identity map.

Finally, we briefly note that the $F_{u}$ and $F_{v}$ curvature component plots are still useful for developing small-amplitude gaits even for non-Abelian connections. Recall from Equation (2.12) that the solution to (5.1) for a general non-Abelian group is $s(T)=s(0) \exp \xi(\theta)$. It is well known [66, 80] that for small closed paths in shape space $\xi(\theta)$ is proportional to the curvature of the connection. Thus the $F_{u}$ and $F_{v}$ curvature plots suggest areas in shape space where small-amplitude shape deformations will achieve motion in the $u$ and $v$ directions, respectively. One can then apply


Figure 5.4: Two gaits resulting in same Abelian group holonomy. Though the start and end points differ, the two paths will result in the same holonomy in the Abelian subgroup component. The holonomy in the vector space components, however, may differ.
the results of Leonard and Krishnaprasad [66] or Radford and Burdick [100] to design small-amplitude gaits.

As an example, we demonstrate how to generate a gait to achieve net sideways motion without net forward translation or rotation. Consider Figure 5.5 which shows the zero-value contours of the $F_{\omega}$ and $F_{u}$ curvature components. We expect that the points where the two contours intersect - such as $\left(\theta_{1}, \theta_{2}\right)=(0.591,0.591)$ - are locations in shape space corresponding to infinitesimally small-amplitude gaits that generate purely lateral motion. In practice, finite amplitude gaits may result in some non-zero forward translation and rotation. We specify one such gait by the equations

$$
\begin{align*}
& \theta_{1}=0.591+0.1 \cos \left(t-\frac{\pi}{4}\right)  \tag{5.10}\\
& \theta_{2}=0.591+0.1 \sin \left(t-\frac{\pi}{4}\right)
\end{align*}
$$

and indicate the path in shape space by a black circle in Figure 5.5. A smallamplitude gait such as this expectedly generates very little net motion, however we


Figure 5.5: Zero contours of $F_{\omega}$ and $F_{u}$ curvature components. The $F_{\omega}$ and $F_{u}$ zero contours are in blue and red, respectively. One of the intersections occurs near $\left(\theta_{1}, \theta_{2}\right)=(0.591,0.591)$. A small-amplitude gait centered about this point in shape space is denoted by black circle.
can repeat the same gait to achieve larger motions. As an example, the gait was prescribed 50 times and the resulting translation of the middle link is shown in Figure 5.6(a). Snapshots of the initial and final configurations of the swimmer are shown in Figure 5.6(b). Indeed, the swimmer achieves almost a purely net lateral translation.

### 5.4.1 Gait Generation Example - Potential Flow

As an example of gait-generation, we choose a desired net rotation of $\beta=\pi / 4$ radians. Recall that the logarithm map for $S E(2)$ is $\log (x, y, \beta)=(u, v, \omega)$ where

$$
\begin{align*}
\omega & =\beta  \tag{5.11}\\
(u, v) & = \begin{cases}(x, y) & \beta=0 \\
\frac{\beta}{2}\left(-x \frac{\sin \beta}{\cos \beta-1}-y, x+y \frac{\sin \beta}{\cos \beta-1}\right) & \beta \neq 0 .\end{cases} \tag{5.12}
\end{align*}
$$

Applying the logarithm map, equation (5.11), we see that $\omega=\pi / 4$ and we seek a closed path in $\left(\theta_{1}, \theta_{2}\right)$ space that encloses a volume equal to $-\pi / 4$ (again, this is due to the minus sign in (5.4)). Figure 5.7 illustrates one such path for the potential flow


Figure 5.6: Potential flow swimmer: Lateral translation gait. A close-up (a) of the translation of the center of the middle link as a result of 50 repetitions of the gait given by Equations (5.10). The start and end points are indicated by black and red dots, respectively. For scale, the initial and final configuration of the swimmer are shown in (b) in dotted and solid outlines, respectively, along with the same information from Figure 5.6(a). Note that the swimmer achieves almost purely lateral net translation as a result of the repeated gait.
swimmer, given by

$$
\begin{align*}
& \theta_{1}(t)=1.5-.45 \cos (t)  \tag{5.13}\\
& \theta_{2}(t)=-1.5-.45 \sin (t)
\end{align*}
$$

where $t \in[0,2 \pi]$. This path was determined by first visually identifying an area of shape space with large curvature. A circular path was chosen and the radius of the path was increased or decreased until the volume of curvature within that path was equal to $-\frac{\pi}{4}$. The resulting translation and rotation of the center of the middle link is shown in Figures 5.8(a) and 5.8(b), respectively. Several frames the swimmer during the gait are shown in Figure 5.9. Note that the holonomy is independent of the speed of travel along this path. Additionally, the rotation component of holonomy is also independent of the starting point on the path.


Figure 5.7: Potential flow swimmer: A path in shape space resulting in net rotation of $\frac{\pi}{4}$ radians. The path equation is given by $\theta_{1}(t)=1.5-.45 \cos (t) ; \theta_{2}(t)=-1.5-$ $.45 \sin (t)$. The blue dot indicates the starting and end point of the particular gait.


Figure 5.8: Potential flow swimmer: Translation and rotation for a sample gait. (a) Translation non-dimensionalized by $l=a+c$ and (b) rotation of the center of the middle link resulting from the gait shown in Figure 5.7. In (a), the black and blue dots indicates the initial and final positions, respectively. Figure 5.9 shows several snapshots of the swimmer during the gait.


Figure 5.9: Potential flow swimmer: Snapshots of swimmer configuration during turning gait. The gait is given by Equations (5.13). The blue curve is the path traced by the center of the middle link (same as in Figure 5.8(a)). The middle link rotates by $\frac{\pi}{4}$ radians and returns to the original shape after one complete gait.

### 5.4.2 Gait Generation Example - Stokes Flow

For completeness, we show how to develop a gait to achieve a desired rotation for the swimmer in Stokes flow. The $\omega$ curvature component is an order of magnitude smaller than in the potential flow case, so it will not be possible to achieve such large rotations. Instead, we choose a net rotation of $-10^{\circ}$, or about -0.1745 radians. Thus we seek an area in shape space enclosing a volume of the $\omega$ curvature component equal to -0.1745 . One such gait, displayed on the $\omega$ component of curvature plot in Figure 5.10, is:

$$
\begin{align*}
& \theta_{1}(t)=1.04+.96 \cos (t)  \tag{5.14}\\
& \theta_{2}(t)=-1.04+.96 \sin (t)
\end{align*}
$$

for $t \in[0,2 \pi]$. The gait begins at the configuration indicated by the blue dot and proceeds counter-clockwise about the circle, returning to the initial configuration. The resulting translation and rotation of the center of the body are shown in Figures 5.11 (a) and $5.11(\mathrm{~b})$, respectively. Snapshots of the swimmer during the gait are shown in Figure 5.12.


Figure 5.10: Stokes flow swimmer: A turning gait. A path in shape space resulting in net rotation of $-10^{\circ}(-0.1745$ radians $): \theta_{1}(t)=1.04+.96 \cos (t) ; \theta_{2}(t)=-1.04+$ $.96 \sin (t)$. The blue dot indicates the starting and end point of the gait.


Figure 5.11: Stokes flow swimmer: Translation and rotation for a turning gait. (a) Translation non-dimensionalized by outer link length, $b$ and (b) rotation of the center of the middle link resulting from the $-10^{\circ}$ turning gait shown in Figure 5.10. In (a), the black and blue dots indicate the initial and final positions, respectively. Snapshots of the swimmer during the gait are shown in Figure 5.12.


Figure 5.12: Stokes flow swimmer: Snapshots of swimmer configuration during turning gait. The gait is shown in Figure 5.10. The blue curve is the path traced by the center of the middle link (same as in Figure 5.11(a)). The middle link rotates by 10 degrees and returns to the original shape after one complete gait.

### 5.5 Abelian vs non-Abelian

We emphasize the fact that the gait design technique presented applies only to the Abelian group component of the semidirect product group. To demonstrate this, we return to the turning gait example of $\S 5.4 .1$ and modify the gait so that the starting point of the path in shape space may vary along the circle as a function of $\phi$ :

$$
\begin{align*}
& \theta_{1}(t)=1.5-.45 \cos (t-\phi)  \tag{5.15}\\
& \theta_{2}(t)=-1.5-.45 \sin (t-\phi) .
\end{align*}
$$

The starting points for 60 gaits are labeled by light grey dots in Figure 5.13 while six of those gaits are highlighted by colored dots. All gaits encircle the same area in shape space, however they begin and end at different points along the curve.

The net rotation resulting from the various gaits should be the same regardless of the starting point of the path as implied by equation (5.4). This result is confirmed in Figure $5.14(\mathrm{~b})$ where the rotation angle of the middle link is plotted versus time for the 60 cases, with the six cases highlighted by colored dots in Figure 5.13 plotted in their corresponding color. The net holonomy in the Abelian rotation component depends only on the area enclosed and not the order in which the path is followed.

By contrast, the translation component of the holonomy is not simply a function of the area enclosed by the path, but due to the non-Abelian nature of the group, also the order in which the path is followed. Figure 5.14(a) plots the non-dimensionalized
translation of the center of the middle link for the various gaits whose starting configurations are indicated in Figure 5.13 and with $g(0)=(0,0,0)$. Again, the six colored curves correspond to the translation resulting from the gaits identified by the same color dots.

Because the net holonomy in these directions depends on the starting point of the path, it is clear that for the translational component in $S E(2)$ - as is the case for general non-Abelian groups - a formula analogous to equation (5.4) is not possible, as the holonomy cannot depend only on the area enclosed by the path. However, as we demonstrated in $\S 5.4$, it is still possible to use the curvature component plots as an aide to designing small-amplitude gaits, even for non-Abelian connections.

### 5.6 Summary

We have shown that for systems that can be expressed in the form of equation (3.56), a nice result exists when the fiber group is a semidirect product group of an Abelian group with a vector space, even if that semidirect product group is non-Abelian. In Theorem 5.2.1, it was shown that since the equations of motion decouple, the component of motion corresponding to the Abelian group can be solved for by equation (5.2). The task of developing gaits for this component is reduced to finding areas in shape space enclosing a volume of curvature equal to the logarithm of the desired net holonomy element. As examples, gaits were found to achieve a net rotation of $-\pi / 4$ for the potential flow swimmer and $10^{\circ}$ for the Stokes flow swimmer. In the potential flow case, by prescribing the same path but changing the starting point along the path in shape space, it was shown that no analogous result exists for the vector space component of motion, since the net translation is dependent not only on the area enclosed by the curve, but also on the order in which the path is followed. The same holds for Stokes flow.


Figure 5.13: Various gait starting points. (a) Various starting points for gaits indicated by colored dots on the plot of the $\omega$ curvature of the potential swimmer (b) Colored and light grey dots indicating the start and end points of a family of gaits specified by $\theta_{1}(t)=1.5-.45 \cos (t-\phi) ; \theta_{2}(t)=-1.5-.45 \sin (t-\phi)$


Figure 5.14: Translation and rotation for a family of gaits. (a) Translation nondimensionalized by $l=a+c$ and (b) rotation of the center of the middle link resulting from the family of gaits $\theta_{1}(t)=1.5-.45 \cos (t-\phi) ; \theta_{2}(t)=-1.5-.45 \sin (t-\phi)$ where the initial and final positions are varied by changing $\phi$. The colored curves correspond to the gait with starting and end position specified in the same color in Figure 5.13(b). The start and end point of the closed path have no effect on the net rotation, but do affect the net holonomy in the translational component. Because the translation depends on the starting point of the path, a formula analogous to (5.4) is not possible, as the holonomy cannot depend only on the area enclosed by the path.

## Chapter 6

## Numerical Method for Potential Flow with Point Vortices

So far we have considered swimming at two extremes of Reynolds numbers: zero and infinity, corresponding to Stokes and potential flow, respectively. In realistic swimming, both inertial and viscous effects play a role in the fluid dynamics. To more accurately model these effects while still retaining a relatively low-order representation, we extend the potential flow model - which accounts only for the inertial effects - by allowing the swimmer to generate vorticity and shed it into the fluid. This effect of viscosity is modeled by introducing a numerical unsteady Kutta condition at the sharp trailing edges of the swimmer. Although this still does not account for viscous drag, we additionally introduce a simple drag model.

In Section 6.1, we provide experimental and numerical motivation for the specific geometry we consider for the swimmer. Section 6.2 gives a general overview of the geometry and variables used in the numerical model. Critical elements of the numerical model are the influence coefficients, which are described in Section 6.3. Various details of the numerical scheme include wake modeling (§6.4.1), the boundary condition (§6.4.2), determining the velocity on the boundary of the body (§6.4.3), the

Kutta condition at the trailing edge (§6.4.4), advancement of the position of the wake vortices (§6.4.5) and the computation of the pressure coefficient on the surface of the swimmer (§6.4.6). Much of the algorithm is based on well-known numerical panel methods $[47,10,115,91,93,53,57]$. We have extended these methods to accurately compute the velocity potential despite the complication of integrating through a multi-valued field (§6.4.7). We also developed a method for avoiding the numerical difficulties that occur when wake vortices (singularities) approach too close to the swimmer (§6.4.8).

All of these details are placed in context in Section 6.5 which describes the algorithm for the numerical model. Whereas most prior applications of panel methods required the motion of the swimmer to be completely prescribed, we prescribe only the shape of the swimmer and compute the overall translation as part of the solution. This is another contribution we have made to extend the traditional panel method.

Finally, in Section 6.6, we validate our code, which we include in Appendix C, against potential flow and Navier Stokes models of tandem foils.

### 6.1 Motivation

The mechanisms that fish employ to swim are as diverse as their morphology: from the snake-like motion of anguilliforms to the more rigid, lunate tail flapping of thunniforms, with a broad spectrum of motion in between. A complete understanding of this rich field is beyond the scope of this work. Instead, we seek to better understand one particular mechanism employed by a certain class of swimmers - specifically, more rigid swimmers such as carangiforms and thunniforms - as described next.

Our study is motivated by the experimental work of Drucker and Lauder [26] on sunfish and the related numerical work of Akhtar, Mittal, Lauder and Drucker [2]. Drucker and Lauder [26] performed flow visualizations of a sunfish swimming in a


Figure 6.1: Drucker and Lauder [26] experiment. The flow around the dorsal and caudal (tail) fins of a sunfish is visualized in the dorsal view with a laser sheet shining at various planes, indicated in Figure D. Figure B is a lateral view of the sunfish with the laser sheet at position 1. The laser sheet at position 2 visualizes the interaction between the dorsal fin-generated vorticity and the caudal fin [26]. We model this dorsal view cross-section of the interaction of the dorsal and caudal fin as a pair of foils. Reprinted with permission.
flow channel by shining a laser sheet that cut across the dorsal and caudal fins (See Figure 6.1). They hypothesized that the presence of the dorsal fin enhances the thrust and efficiency of the swimmer by generating a vortex wake which the caudal fin moves through and from which energy is extracted. Akhtar et al. [2] implemented a NavierStokes solver and modeled the two fins as two-dimensional rounded plates that were prescribed nearly the same motion as the fins of the sunfish. They found that both propulsive efficiency and thrust were enhanced, as compared to the case where just the "tail" body moved in the same way without the presence of a leading body and its wake.

This hydrodynamic interaction between a leading and trailing body is the propulsive mechanism that we wish to better understand. Hence, we consider a simplified swimmer composed of two connected foils thought to represent the dorsal (leading foil) and caudal (trailing foil) fins. While both bodies shed vorticity into the flow, the trailing foil also interacts with the vorticity shed from the leading foil.

### 6.2 Model Overview

The swimmer is made up of two two-dimensional foils with NACA0012 cross-sections. The leading and trailing foils are identified as bodies 1 and 2 , respectively. We imagine that they are connected by (invisible) joints and actuators which specify their positions and orientation relative to each other. There is a uniform freestream velocity of $\mathbf{U}_{\infty}$. Refer to Figure 6.2 for a schematic of the system. The swimmer is placed in a fluid assumed to be incompressible, inviscid and irrotational everywhere except at the source and vorticity singularities which are determined as part of the solution. Hence, the equations for potential flow govern the fluid motion.

We wish to determine the overall motion of the swimmer through the fluid as a result of the prescribed relative shape changes. The algorithm for our model is largely based on the work of Pang [91], and in this chapter we present the algorithm for completeness, along with our contributions.

Each body is discretized into $N$ straight line panels. We use $i$ and $j$ to index the panels and denote the bottom trailing edge panel of the leading body as $i=1$. The panels are numbered clockwise around the body all the way to $i=N$ for the trailing edge panel at the top of the leading body. Likewise, the panels of the trailing body are numbered from $i=N+1$ to $i=2 N$. The geometric center of each panel is the control point - the points at which the boundary conditions are satisfied - which is identified by the same index number as its corresponding panel.

Table 6.1 summarizes the nomenclature for the variables used throughout this chapter. At each panel, we define an outward normal vector $\mathbf{n}_{i}=\left(n_{i, x}, n_{i, y}\right)$ and tangent vector $\mathbf{t}_{i}=\left(t_{i, x}, t_{i, y}\right)$, defined in the clockwise direction around the body. The source density strength on the $i^{\text {th }}$ panel is denoted $q_{i}$ and the strength of the trailing vortices are denoted $\Gamma_{m}$. Note that $\Gamma_{b 1}=\gamma_{1} L_{1}$ and $\Gamma_{b 2}=\gamma_{2} L_{2}$ are used to denote the total circulation about the first and second foils, respectively, where $L_{1}$ and $L_{2}$ are the perimeters and $\gamma_{1}$ and $\gamma_{2}$ are the constant vorticity density strength values


Figure 6.2: Schematic of two-foil swimmer system. There is a freestream velocity $\mathbf{U}_{\infty}$. The trailing edge panels $(1, N, N+1$ and $2 N)$ at which the Kutta condition is imposed are labeled. Along each body panel, there is a constant source distribution density $q_{i}$ which varies from panel to panel and a constant vorticity distribution density which is constant for each body (either $\gamma_{1}$ or $\gamma_{2}$ ). The wake panels shed from the trailing edge of each body are denoted by a slightly bolder line. Each wake panel has length $\Delta_{w}$, orientation relative to the horizontal $\theta$ and vorticity distribution density $\gamma_{w}$. At the end of each time step - denoted by the index $k$ - the circulation along each wake panel is concentrated into a point vortex and convected into the wake.
of the respective foils. The freestream velocity is $\mathbf{U}_{\infty}$ and the kinematic velocity at the $i^{\text {th }}$ control point is $\mathbf{V}_{i}$. The equations of motion from time $t_{0}$ to $t_{f}$ are solved along equally-spaced time increments, $\Delta t$. Time steps are denoted by the index $k$. At each time step, the no-penetration Neumann boundary condition is satisfied at the control points of both bodies. The normal component of the fluid velocity at each control point due to the contributions from the distributed singularities on the bodies and wake plus the freestream velocity is equal to the normal velocity of the body at that point.

In order to more realistically model a fish-like swimmer and to allow vorticity shedding, a Kutta condition is imposed at both trailing edges. There are various possible implementations of the Kutta condition [23], and we choose to specify that the pressure difference across the trailing edge must be zero. In practice, the pressure at the control points of the top and bottom trailing edge panels (slightly upstream of the trailing edge) is specified to be equal.

To satisfy the Kutta condition (and the Kelvin circulation theorem) vorticity is shed from the trailing edge of each foil in the form of a wake panel. At each time step $k$, a straight panel containing a constant vorticity distribution is introduced at the trailing edge as shown in Figure 6.2. The length $\left(\Delta_{w 1}\right)_{k},\left(\Delta_{w 2}\right)_{k}$, orientation $\left(\theta_{1}\right)_{k},\left(\theta_{2}\right)_{k}$ and strength $\left(\gamma_{w 1}\right)_{k},\left(\gamma_{w 2}\right)_{k}$ of the vorticity distribution along the panels are computed as part of the solution.

Following the work of Basu and Hancock [10], an unknown strength constant source density $\left(q_{i}\right)_{k}$ is distributed along each panel. The source strength varies from panel to panel. Each panel also has a constant distribution of vorticity density of unknown strength, $\left(\gamma_{1}\right)_{k}$ and $\left(\gamma_{2}\right)_{k}$ on the leading and trailing foil, respectively. Unlike the source distributions, the vorticity density distribution is constant for all the panels in a given body.

After the solution is found, the vorticity along the wake panels is concentrated into a point vortex at the center of the panel, and the vortices in the wake are advanced in time due to their local velocity. Because two new vortices are shed, the dimension of the system increases by two at each time step.

A flowchart of the overall algorithm is included in Figure 6.3. We will discuss the various components and refer back to the figure in the following sections.

## Table 6.1: Summary of nomenclature.

| variable | description |
| :--- | :--- |
| $\mathbf{V}_{i}$ | swimmer velocity in inertial frame at $i^{\text {th }}$ panel control point |
| $V_{i}^{\{x, y\}}$ | $\{\mathrm{x}, \mathrm{y}\}$ swimmer velocity components in inertial frame at $i^{\text {th }}$ panel |
| $\mathbf{u}_{i}$ | fluid velocity at $i^{\text {th }}$ panel control point |
| $u_{i}^{\{n, t\}}$ | \{normal,tangent $\}$ fluid velocity at $i^{\text {th }}$ panel control point |
| $\mathbf{U}_{\infty}$ | freestream velocity |
| Continued on next page |  |

Table 6.1 - continued from previous page

| variable | description |
| :---: | :---: |
| $\begin{aligned} & U_{\infty}^{\{x, y\}} \\ & U_{h}^{\{x, y\}} \\ & \mathbf{U}_{w i} \\ & U_{w i}^{\{x, y\}} \\ & v^{\{x, y\}} \\ & \bar{v}^{\{x, y\}} \\ & \left(v^{\{x, y\}}\right)^{p} \\ & \omega \\ & \bar{\omega} \\ & \omega^{p} \\ & \left(x_{o}, y_{o}\right) \\ & \left(x_{h}, y_{h}\right) \\ & \left(x_{i}, y_{i}\right) \\ & \beta \\ & p_{i} \\ & \phi_{i} \\ & \Gamma_{b k} \\ & \gamma_{k} \\ & \gamma_{w k} \\ & L_{k} \\ & q_{i} \\ & F^{\{x, y\}} \\ & \bar{F}^{\{x, y\}} \\ & \mathrm{M} \end{aligned}$ | $\{\mathrm{x}, \mathrm{y}\}$ freestream velocity components <br> $\{\mathrm{x}, \mathrm{y}\}$ velocity components of $h^{\text {th }}$ point vortex velocity at the midpoint of the $i^{\text {th }}$ wake panel $\{\mathrm{x}, \mathrm{y}\}$ velocity components at the midpoint of the $i^{\text {th }}$ wake panel $\{\mathrm{x}, \mathrm{y}\}$ velocity components of body center of mass in inertial frame weighted $\{\mathrm{x}, \mathrm{y}\}$ velocity components of body center of mass in inertial frame adjusted $\{\mathrm{x}, \mathrm{y}\}$ velocity components of body center of mass in inertial frame angular velocity of body weighted angular velocity of body adjusted angular velocity of body swimmer center of mass coordinates in inertial frame coordinates of $h^{\text {th }}$ point vortex coordinates of $i^{\text {th }}$ panel control point in inertial frame orientation of body relative to inertial frame static pressure at $i^{\text {th }}$ panel control point velocity potential at $i^{\text {th }}$ panel control point circulation about $k^{\text {th }}$ body vorticity density strength about $k^{\text {th }}$ body vorticity density strength of $k^{\text {th }}$ wake panel perimeter length of $k^{\text {th }}$ body source density strength over $i^{\text {th }}$ panel $\{x, y\}$ force components acting on swimmer $\{x, y\}$ weighted force components acting on swimmer moment acting on swimmer center of mass |
|  | Continued on next page |

Table 6.1 - continued from previous page

| variable | description |
| :--- | :--- |
| $\bar{M}$ | weighted moment acting on swimmer center of mass |
| I | swimmer's instantaneous moment of inertia |
| $\xi$ | force weighting parameter |
| $\gamma_{\{x, y, \omega\}}$ | velocity weighting parameters |
| $\Delta_{w k}$ | length of $k^{\text {th }}$ wake panel |
| $\theta_{w k}$ | angle relative to horizontal of $k^{\text {th }}$ wake panel |
| $T$ | thrust |
| $W$ | work |
| $P$ | power |
| $\eta_{P}$ | propulsive efficiency |
| $\mathbf{n}_{i}$ | normal vector at $i^{\text {th }}$ panel |
| $\mathbf{t}_{i}$ | tangent vector at $i^{\text {th }}$ panel |
| $A, B, C$ | influence coefficients (see Table 6.2) |

### 6.3 Influence Coefficients

Influence coefficients are the scalar velocity components induced at a point due to a unit strength singularity or distribution of singularities. They are determined solely based on the geometry of the system.

We adopt the convention of Pang [91] where coefficients due to source and vorticity distributions over panels are denoted by $A$ and $B$ respectively, while those corresponding to point vortices are denoted by $C$.

The indices $i, j$ range from $1,2, \ldots, N, N+1, N+2, \ldots, 2 N$ and refer to panels on the


Figure 6.3: Flowchart of numerical algorithm for advancing one time step. Equation numbers are noted next to the corresponding step.
two bodies. The indices $h$ and $m$ refer to point vortices in the wake and $w 1$ and $w 2$ correspond to the wake panels currently being shed from bodies 1 and 2 , respectively. The letters $n, t, x$ and $y$ appear as superscripts to indicate the direction of the influence coefficients. When $n$ and $t$ are used, they indicate the outward normal and tangent direction in the frame of the panel corresponding to the first index. Table 6.2 summarizes the notation for the various influence coefficients.

As an example, consider a unit strength point source singularity located at ( $x_{o}, y_{o}$ ). The velocity induced at a point $(x, y)$ due to the singularity has components $\left(u_{s}, v_{s}\right)$ given by Equations (3.22) and (3.23). (If instead the singularity were a unit strength point vortex, the induced velocities would be: $u_{v}=v_{s}$ and $v_{v}=-u_{s}$.) Following the same change of coordinates described in detail in $\S 3.4$, if $\mathbf{n}_{i}$ and $\mathbf{t}_{i}$ are the unit normal and tangent vectors, respectively, corresponding to the $i^{\text {th }}$ panel, we can define the following expressions:

$$
\begin{equation*}
A_{i, j}^{n}=\mathbf{n}_{i} \cdot \mathbf{V}_{i, j} \text { and } A_{i, j}^{t}=\mathbf{t}_{i} \cdot \mathbf{V}_{i, j} \tag{6.1}
\end{equation*}
$$

where $\mathbf{V}_{i, j}$ is the velocity induced at the control point of panel $i$ due to a unit strength source distribution on panel $j$. Thus, we can write

$$
\begin{equation*}
\mathbf{V}_{i, j}=A_{i, j}^{n} \mathbf{n}_{i}+A_{i, j}^{t} \mathbf{t}_{i} \tag{6.2}
\end{equation*}
$$

where $A_{i, j}^{n}$ and $A_{i, j}^{t}$ are the normal and tangent components, respectively, of the induced velocity at the $i^{\text {th }}$ panel's control point in the coordinate frame fixed to the $i^{\text {th }}$ panel due to a unit source distribution on the $j^{\text {th }}$ panel. These are the first two coefficients described in Table 6.2. The other coefficients are determined in an analogous matter.

Table 6.2: Influence coefficients

| variable | influence coefficients |
| :--- | :--- |
| $A_{i, j}^{\{a, t\}}$ | \{normal,tangent $\}$ velocity component induced at the control point of <br> panel $i$ due to a unit strength source distribution on panel $j$ <br> \{normal, tangent $\}$ velocity component induced at the control point of <br> panel $i$ due to a unit strength vorticity distribution on panel $j$ |
| $B_{i, w h}^{\{n, t\}}$ | \{normal, tangent $\}$ velocity component induced at the control point of <br> panel $i$ due to a unit strength vorticity distribution on the wake panel <br> of the $k^{\text {th }}$ body |
| $A_{w,, j}^{\{x, y\}}$ | $\{x, y\}$ velocity component induced at the control point of the wake panel <br> of the $k^{\text {th }}$ body due to a unit strength source distribution on panel j |
| $B_{w k, j}^{\{x, y\}}$ | $\{x, y\}$ velocity component induced at the control point of the wake panel <br> of the $k^{\text {th }}$ body due to a unit strength vorticity distribution on panel j |
| $A_{h, j}^{\{x, y\}}$ | $\{x, y\}$ velocity component induced at the $h^{\text {th }}$ wake vortex due to a unit <br> strength source distribution on panel j |
| $B_{h, j}^{\{x, y\}}$ | $\{x, y\}$ velocity component induced at the $h^{\text {th }}$ wake vortex due to a unit <br> strength vorticity distribution on panel j |
| $B_{h, w k}^{\{x, y\}}$ | $\{x, y\}$ velocity component induced at the $h^{\text {th }}$ wake vortex due to a unit <br> strength vorticity distribution on the wake panel of the $k^{\text {th }}$ body |
| $C_{i, m}^{\{n, t\}}$ | $\{$ normal,tangent $\}$ velocity component induced at the control point of the <br> $i^{\text {th }}$ panel due to the $m^{\text {th }}$ wake vortex, assuming unit strength <br> $\{x, y\}$ velocity component induced at the control point of the wake panel <br> of the $k^{\text {th }}$ body due to the $m^{\text {th }}$ wake vortex, assuming unit strength |
| $C_{w k, m}^{\{x, y\}}$ |  |
| $C_{h, m}^{\{x, y\}}$ | \{x,y\} velocity component induced at the $h^{\text {th }}$ wake vortex due to the $m^{\text {th }}$ <br> wake vortex, assuming unit strength |

### 6.4 Details of Numerical Scheme

### 6.4.1 Wake Modeling

At each time step, the circulation about the two bodies changes to satisfy the various boundary conditions. To satisfy Kelvin's circulation theorem, which states that the circulation around a closed curve moving with the fluid must remain constant, vorticity is shed from each body in the form of a wake panel at every time step. The vorticity strength $\gamma_{w 1}$ along the wake panel with length $\Delta_{w 1}$ shedding from the first body with perimeter $L_{1}$ and circulation equal to $\gamma_{1} L_{1}$ is found by equating the circulation of the wake panel to the change in circulation since the prior time step
about the body:

$$
\begin{equation*}
\left(\gamma_{w 1}\right)_{k} \cdot\left(\Delta_{w 1}\right)_{k}=L_{1}\left\{\left(\gamma_{1}\right)_{k-1}-\left(\gamma_{1}\right)_{k}\right\} . \tag{6.3}
\end{equation*}
$$

Likewise, the vorticity strength of the wake panel shedding from body 2 is found by a similar expression:

$$
\begin{equation*}
\left(\gamma_{w 2}\right)_{k} \cdot\left(\Delta_{w 2}\right)_{k}=L_{2}\left\{\left(\gamma_{2}\right)_{k-1}-\left(\gamma_{2}\right)_{k}\right\} . \tag{6.4}
\end{equation*}
$$

The length and orientation of the trailing edge wake panels are unknown and are determined by the fluid velocity near the trailing edge. Since the wake panel represents a continuous distribution of vorticity shed from the trailing edge during the time between $t_{k-1}$ and $t_{k}$, the length of the wake panel is proportional to the local fluid velocity at its center:

$$
\begin{equation*}
\left(\Delta_{w i}\right)_{k}=\Delta t\left\|\left(\mathbf{U}_{w i}\right)_{k}\right\| \tag{6.5}
\end{equation*}
$$

and the orientation is such that the panel is tangential to the local velocity:

$$
\begin{equation*}
\left(\theta_{i}\right)_{k}=\tan ^{-1} \frac{\left(U_{w i}^{y}\right)_{k}}{\left(U_{w i}^{x}\right)_{k}} \tag{6.6}
\end{equation*}
$$

where $U_{w i}^{x}$ and $U_{w i}^{y}$ are the inertial frame $x$ and $y$ components, respectively, of the fluid velocity at the center of the $i^{\text {th }}$ wake panel. In the flowchart in Figure 6.3, this section of the algorithm is designated by the box titled "Recompute wake panel angles and length".

### 6.4.2 Boundary Condition

The velocity at the surface of the body should be tangential to the body, with zero relative normal velocity component. In other words, the fluid may slip along the body surface, but not penetrate the body. In practice, the boundary condition is imposed only at a finite number of control points and fluid does indeed "leak" into and out
of the body at other points (although with a sufficiently large number of panels, this method produces the correct forces on the body). For this reason, it is possible for point vortices in the wake to enter inside the boundaries of the body. We present a method to avoid this complication in Section 6.4.8.

We wish to solve for the unknown vorticity density strengths, $\gamma_{1}$ and $\gamma_{2}$ and the source density strength, $q_{j}$ at the $k^{\text {th }}$ time step. At each time step, the normal component of velocity induced at every control point due to the various flow components must equal the normal velocity of the body at that point (in the case of a stationary body, the normal component is zero). This condition provides one of the equations needed to solve for the unknown quantities:

$$
\begin{align*}
& \sum_{j=1}^{2 N}\left(A_{i, j}^{n}\right)_{k}\left(q_{j}\right)_{k}+\left(\gamma_{1}\right)_{k} \sum_{j=1}^{N}\left(B_{i, j}^{n}\right)_{k}+\left(\gamma_{2}\right)_{k} \sum_{j=N+1}^{2 N}\left(B_{i, j}^{n}\right)_{k}+\left(\gamma_{w 1}\right)_{k}\left(B_{i, w 1}^{n}\right)_{k} \\
& \quad+\left(\gamma_{w 2}\right)_{k}\left(B_{i, w 2}^{n}\right)_{k}+\sum_{m=1}^{2(k-1)}\left(C_{i, m}^{n}\right)_{k}\left(\Gamma_{m}\right)_{k}+\mathbf{U}_{\infty} \cdot \mathbf{n}_{i}=\left(\mathbf{V}_{i}\right)_{k} \cdot\left(\mathbf{n}_{i}\right)_{k} \tag{6.7}
\end{align*}
$$

where $i=1,2 \ldots 2 N$ and $A_{i, j}^{n}, B_{i, j}^{n}, B_{i, w 1}^{n}, B_{i, w 2}^{n}$, and $C_{i, m}^{n}$ are the influence coefficients described in Table 6.2.

Rearranging Equation (6.7) and applying Equations (6.3) and (6.4), we can express the source distribution as a function of the two vorticity densities on the bodies plus a constant:

$$
\begin{align*}
\sum_{j=1}^{2 N}\left(A_{i, j}^{n}\right)_{k}\left(q_{j}\right)_{k} & =\left\{\frac{L_{1}}{\left(\Delta_{w 1}\right)_{k}}\left(B_{i, w 1}^{n}\right)_{k}-\sum_{j=1}^{N}\left(B_{i, j}^{n}\right)_{k}\right\}\left(\gamma_{1}\right)_{k} \\
& +\left\{\frac{L_{2}}{\left(\Delta_{w 2}\right)_{k}}\left(B_{i, w 2}^{n}\right)_{k}-\sum_{j=N+1}^{2 N}\left(B_{i, j}^{n}\right)_{k}\right\}\left(\gamma_{2}\right)_{k}  \tag{6.8}\\
& -\frac{L_{1}}{\left(\Delta_{w 1}\right)_{k}}\left(\gamma_{1}\right)_{k-1}\left(B_{i, w 1}^{n}\right)_{k}-\frac{L_{2}}{\left(\Delta_{w 2}\right)_{k}}\left(\gamma_{2}\right)_{k-1}\left(B_{i, w 2}^{n}\right)_{k} \\
& -\sum_{m=1}^{2(k-1)}\left(C_{i, m}^{n}\right)_{k}\left(\Gamma_{m}\right)_{k}-\mathbf{U}_{\infty} \cdot\left(\mathbf{n}_{i}\right)_{k}+\left(\mathbf{V}_{i}\right)_{k} \cdot\left(\mathbf{n}_{i}\right)_{k}
\end{align*}
$$

The $2 N$ equations (6.8) can be expressed as a matrix equation:

$$
\begin{equation*}
A_{k}(q)_{k}=a_{k}^{1} \cdot\left(\gamma_{1}\right)_{k}+a_{k}^{2} \cdot\left(\gamma_{2}\right)_{k}+a_{k}^{3} \tag{6.9}
\end{equation*}
$$

Where $A_{k}$ is a $2 N \times 2 N$ matrix and $a_{k}^{1}$ and $a_{k}^{2}$ are $2 N \times 1$ column vectors equal to the terms within the curly braces and $a_{k}^{3}$ is the sum of the remaining terms in Equation (6.8). We can then solve for the source distribution as follows:

$$
\begin{aligned}
(q)_{k} & =\left(A_{k}\right)^{-1} a_{k}^{1} \cdot\left(\gamma_{1}\right)_{k}+\left(A_{k}\right)^{-1} a_{k}^{2} \cdot\left(\gamma_{2}\right)_{k}+\left(A_{k}\right)^{-1} a_{k}^{3} \\
& =b_{k}^{1} \cdot\left(\gamma_{1}\right)_{k}+b_{k}^{2} \cdot\left(\gamma_{2}\right)_{k}+b_{k}^{3} .
\end{aligned}
$$

Following Pang [91], we express the $j^{\text {th }}$ panel source density as:

$$
\begin{equation*}
\left(q_{j}\right)_{k}=\left(b_{j}^{1}\right)_{k} \cdot\left(\gamma_{1}\right)_{k}+\left(b_{j}^{2}\right)_{k} \cdot\left(\gamma_{2}\right)_{k}+\left(b_{j}^{3}\right)_{k} . \tag{6.10}
\end{equation*}
$$

This portion of the algorithm corresponds to the box labeled "Solve for source distribution" in Figure 6.3.

We note that in Pang's formulation, the right hand side of Equation (6.7) is simply a function of the imposed shape change. However, since we will allow the swimmer to self-propel, the boundary velocity $\mathbf{V}_{i}$ will also depend on the overall motion of the swimmer. In the next section, we explain how we determine the boundary velocity.

### 6.4.3 Boundary Velocity

The term on the right hand side of Equation (6.7), $\left(\mathbf{V}_{i}\right)_{k} \cdot\left(n_{i}\right)_{k}$ is the kinematic velocity of the $i^{\text {th }}$ control point in the direction normal to the $i^{\text {th }}$ panel at the $k^{\text {th }}$ time step. The value of the vector $\left(n_{i}\right)_{k}$ is unknown at the beginning of the time step since it depends on the overall orientation of the swimmer, which is unknown. Also, $\left(\mathbf{V}_{i}\right)_{k}$ is unknown initially, since it depends on the motion of the body, which we seek
to compute as part of the solution. We determine $\left(\mathbf{V}_{i}\right)_{k}$ by an iterative process that repeats until the computed forces acting on the body converge to within a desired tolerance. We describe this procedure in more detail in §6.5.

The velocity of the boundary of the body depends on two components. One is the deformation in the shape of the swimmer, which is prescribed in advance or determined by a controller. For example, the foils may be prescribed to pitch or plunge relative to each other. On the other hand, the swimmer is interacting with the fluid and experiencing forces which propel it through the fluid. Those forces generate a "rigid" motion of the body, which contributes a net overall translation and rotation of the center of mass $\left(x_{0}, y_{0}\right)$.

The velocity components of the body's center of mass and rotational velocity at the $k^{\text {th }}$ time step are:

$$
\begin{align*}
\left(v^{x}\right)_{k} & =\frac{\left(x_{0}\right)_{k}-\left(x_{0}\right)_{k-1}}{\Delta t}  \tag{6.11}\\
\left(v^{y}\right)_{k} & =\frac{\left(y_{0}\right)_{k}-\left(y_{0}\right)_{k-1}}{\Delta t}  \tag{6.12}\\
\omega_{k} & =\frac{\left(\beta_{0}\right)_{k}-\left(\beta_{0}\right)_{k-1}}{\Delta t} . \tag{6.13}
\end{align*}
$$

The previous three equations partially correspond to the box labeled "Advance vortices and swimmer position and orientation" on the flowchart in Figure 6.3.

The position and orientation at the previous time step are known, however the values at the $k^{\text {th }}$ time step are initially unknown and need to be determined through an iterative process. The velocity of the center of mass from the previous time step may be used to determine an initial guess for the new center of mass position and orientation. Combined with the prescribed shape of the swimmer at the previous and current time step, we can compute the coordinates of the boundary of the swimmer in an inertial frame of reference at the previous $\left(\left(x_{i}\right)_{k-1},\left(y_{i}\right)_{k-1}\right)$ and current time steps $\left(\left(x_{i}\right)_{k},\left(y_{i}\right)_{k}\right)$. Using those coordinates, we compute the vector quantity
$\left(\mathbf{V}_{i}\right)_{k}=\left(V_{i}^{x}, V_{i}^{y}\right)$ at each control point as follows:

$$
\begin{align*}
\left(V_{i}^{x}\right)_{k} & =\frac{\left(x_{i}\right)_{k}-\left(x_{i}\right)_{k-1}}{\Delta t}  \tag{6.14}\\
\left(V_{i}^{y}\right)_{k} & =\frac{\left(y_{i}\right)_{k}-\left(y_{i}\right)_{k-1}}{\Delta t} \tag{6.15}
\end{align*}
$$

Finally, the normal vector at each panel is determined from the geometry at the $k^{\text {th }}$ time step.

### 6.4.4 Kutta Condition

We apply the unsteady form of Bernoulli's equation to satisfy the chosen Kutta condition: that the pressure must be continuous at the trailing edge. In an incompressible, inviscid, irrotational fluid, Bernoulli's equation states that throughout the (unit density) fluid,

$$
\begin{equation*}
p+\frac{1}{2}\|\mathbf{u}\|^{2}+\frac{\partial \phi}{\partial t}=\text { constant. } \tag{6.16}
\end{equation*}
$$

For the first foil, this means that at panels 1 and $N$ :

$$
\begin{equation*}
p_{1}+\frac{1}{2}\left\|\mathbf{u}_{1}\right\|^{2}+\frac{\partial \phi_{1}}{\partial t}=p_{N}+\frac{1}{2}\left\|\mathbf{u}_{N}\right\|^{2}+\frac{\partial \phi_{N}}{\partial t} . \tag{6.17}
\end{equation*}
$$

Since the Kutta condition prescribes that $p_{1}=p_{N}$, we get the following expression:

$$
\begin{equation*}
p_{1}-p_{N}=0=\frac{1}{2}\left(\left\|\mathbf{u}_{N}\right\|^{2}-\left\|\mathbf{u}_{1}\right\|^{2}\right)+\frac{\partial \phi_{N}}{\partial t}-\frac{\partial \phi_{1}}{\partial t} . \tag{6.18}
\end{equation*}
$$

The change in potential between two points $x_{1}$ and $x_{2}$, connected by the path $s$ is given by

$$
\begin{equation*}
\Delta \phi=\int_{x_{1}}^{x_{2}} \mathbf{u} \cdot \mathrm{~d} s \tag{6.19}
\end{equation*}
$$

where $\mathbf{u}$ is the fluid velocity along the path $s$. The integral evaluated along the surface of the airfoil from panel 1 to panel $N$ in the clockwise direction is also by definition
equal to the circulation enclosed by the path:

$$
\begin{equation*}
\int_{1}^{N} \mathbf{u} \cdot \mathrm{~d} s=\Gamma_{b 1} \tag{6.20}
\end{equation*}
$$

Hence, $\phi_{N}=\phi_{1}+\Gamma$ and the Kutta condition becomes:

$$
\begin{equation*}
\left\|\mathbf{u}_{1}\right\|^{2}-\left\|\mathbf{u}_{N}\right\|^{2}=2 \frac{\partial \Gamma_{b 1}}{\partial t} \tag{6.21}
\end{equation*}
$$

In discrete form, the Kutta condition on the first body may be written:

$$
\begin{equation*}
\left\|\mathbf{u}_{1}\right\|_{k}^{2}-\left\|\mathbf{u}_{N}\right\|_{k}^{2}=2 L_{1} \frac{\left(\gamma_{1}\right)_{k}-\left(\gamma_{1}\right)_{k-1}}{\Delta t} \tag{6.22}
\end{equation*}
$$

Likewise, on the second body the expression is

$$
\begin{equation*}
\left\|\mathbf{u}_{N+1}\right\|_{k}^{2}-\left\|\mathbf{u}_{2 N}\right\|_{k}^{2}=2 L_{2} \frac{\left(\gamma_{2}\right)_{k}-\left(\gamma_{2}\right)_{k-1}}{\Delta t} \tag{6.23}
\end{equation*}
$$

The square of the fluid velocity at any control point on the body is equal to the square of the tangent velocity at that point plus the square of the normal component of the body's kinematic velocity:

$$
\begin{equation*}
\left\|\mathbf{u}_{i}\right\|_{k}^{2}=\left(u_{i}^{t}\right)_{k}^{2}+\left(\mathbf{V}_{i} \cdot \mathbf{n}_{i}\right)_{k}^{2} \tag{6.24}
\end{equation*}
$$

The tangent velocity at panel 1 is given by:

$$
\begin{align*}
\left(u_{1}^{t}\right)_{k}=\sum_{j=1}^{2 N}\left(A_{1, j}^{t}\right)_{k}\left(q_{j}\right)_{k} & +\left\{\sum_{j=1}^{N}\left(B_{1, j}^{t}\right)_{k}-\frac{L_{1}}{\left(\Delta_{w 1}\right)_{k}}\left(B_{1, w 1}^{t}\right)_{k}\right\}\left(\gamma_{1}\right)_{k} \\
& +\left\{\sum_{j=N+1}^{2 N}\left(B_{1, j}^{t}\right)_{k}-\frac{L_{2}}{\left(\Delta_{w 2}\right)_{k}}\left(B_{1, w 2}^{t}\right)_{k}\right\}\left(\gamma_{2}\right)_{k}  \tag{6.25}\\
& +\frac{L_{1}}{\left(\Delta_{w 1}\right)_{k}}\left(\gamma_{1}\right)_{k-1}+\frac{L_{2}}{\left(\Delta_{w 2}\right)_{k}}\left(\gamma_{2}\right)_{k-1} \\
& +\sum_{m=1}^{2(k-1)}\left(C_{1, m}^{t}\right)_{k} \Gamma_{m}+\mathbf{U}_{\infty} \cdot\left(\mathbf{t}_{1}\right)_{k}
\end{align*}
$$

By substituting the source density expression in Equation (6.10) into (6.25) we get the following expression for $\left(u_{1}^{t}\right)_{k}$ :

$$
\begin{equation*}
\left(u_{1}^{t}\right)_{k}=\left(D_{1}^{1}\right)_{k}\left(\gamma_{1}\right)_{k}+\left(D_{1}^{2}\right)_{k}\left(\gamma_{2}\right)_{k}+\left(D_{1}^{3}\right)_{k} \tag{6.26}
\end{equation*}
$$

where

$$
\begin{align*}
\left(D_{1}^{1}\right)_{k} & =\sum_{j=1}^{2 N}\left(A_{1, j}^{t}\right)_{k}\left(b_{1}^{1}\right)_{k}+\sum_{j=1}^{N}\left(B_{1, j}^{t}\right)_{k}-\frac{L_{1}}{\left(\Delta_{w 1}\right)_{k}}\left(B_{1, w 1}^{t}\right)_{k} \\
\left(D_{1}^{2}\right)_{k} & =\sum_{j=1}^{2 N}\left(A_{1, j}^{t}\right)_{k}\left(b_{1}^{2}\right)_{k}+\sum_{j=N+1}^{2 N}\left(B_{1, j}^{t}\right)_{k}-\frac{L_{2}}{\left(\Delta_{w 2}\right)_{k}}\left(B_{1, w 2}^{t}\right)_{k}  \tag{6.27}\\
\left(D_{1}^{3}\right)_{k} & =\sum_{j=1}^{2 N}\left(A_{1, j}^{t}\right)_{k}\left(b_{1}^{3}\right)_{k} \frac{L_{1}}{\left(\Delta_{w 1}\right)_{k}}\left(\gamma_{1}\right)_{k-1}+\frac{L_{2}}{\left(\Delta_{w 2}\right)_{k}}\left(\gamma_{2}\right)_{k-1} \\
& +\sum_{m=1}^{2(k-1)}\left(C_{1, m}^{t}\right)_{k} \Gamma_{m}+\mathbf{U}_{\infty} \cdot\left(\mathbf{t}_{1}\right)_{k}
\end{align*}
$$

Likewise, expressions for the tangent velocities on the other panels that form the trailing edge can be found in a similar fashion using Equations (6.27) to determine
the coefficients:

$$
\begin{align*}
\left(u_{N}^{t}\right)_{k} & =\left(D_{N}^{1}\right)_{k}\left(\gamma_{1}\right)_{k}+\left(D_{N}^{2}\right)_{k}\left(\gamma_{2}\right)_{k}+\left(D_{N}^{3}\right)_{k}  \tag{6.28}\\
\left(u_{N+1}^{t}\right)_{k} & =\left(D_{N+1}^{1}\right)_{k}\left(\gamma_{1}\right)_{k}+\left(D_{N+1}^{2}\right)_{k}\left(\gamma_{2}\right)_{k}+\left(D_{N+1}^{3}\right)_{k}  \tag{6.29}\\
\left(u_{2 N}^{t}\right)_{k} & =\left(D_{2 N}^{1}\right)_{k}\left(\gamma_{1}\right)_{k}+\left(D_{2 N}^{2}\right)_{k}\left(\gamma_{2}\right)_{k}+\left(D_{2 N}^{3}\right)_{k} . \tag{6.30}
\end{align*}
$$

The Kutta condition for body 1 in Equation (6.22) requires the square of the velocities on the top and bottom trailing panels. Temporarily dropping the time index for clarity, we find $\left\|\mathbf{u}_{1}\right\|^{2}$ and $\left\|\mathbf{u}_{N}\right\|^{2}$ as follows:

$$
\begin{gather*}
\left\|\mathbf{u}_{1}\right\|^{2}=\left(D_{1}^{1} \gamma_{1}+D_{1}^{2} \gamma_{2}+D_{1}^{3}\right)^{2}+\left(\mathbf{V}_{1} \cdot \mathbf{n}_{1}\right)^{2} \\
=\left(D_{1}^{1}\right)^{2} \gamma_{1}^{2}+\left(D_{1}^{2}\right)^{2} \gamma_{2}^{2}+\left(D_{1}^{3}\right)^{2}  \tag{6.31}\\
+2 D_{1}^{1} D_{1}^{2} \gamma_{1} \gamma_{2}+2 D_{1}^{1} D_{1}^{3} \gamma_{1}+2 D_{1}^{2} D_{1}^{3} \gamma_{2}+\left(\mathbf{V}_{1} \cdot n_{1}\right)^{2} \\
\left\|\mathbf{u}_{N}\right\|^{2}=\left(D_{N}^{1} \gamma_{1}+D_{N}^{2} \gamma_{2}+D_{N}^{3}\right)^{2}+\left(\mathbf{V}_{N} \cdot \mathbf{n}_{N}\right)^{2} \\
=\left(D_{N}^{1}\right)^{2} \gamma_{1}^{2}+\left(D_{N}^{2}\right)^{2} \gamma_{2}^{2}+\left(D_{N}^{3}\right)^{2}  \tag{6.32}\\
+ \\
2 D_{N}^{1} D_{N}^{2} \gamma_{1} \gamma_{2}+2 D_{N}^{1} D_{N}^{3} \gamma_{1}+2 D_{N}^{2} D_{N}^{3} \gamma_{2}+\left(\mathbf{V}_{N} \cdot n_{N}\right)^{2}
\end{gather*}
$$

By substituting (6.31) and (6.32) into (6.22), the Kutta condition for the first body may be expressed as

$$
\begin{equation*}
A A_{1}\left(\gamma_{1}\right)_{k}^{2}+B B_{1}\left(\gamma_{2}\right)_{k}^{2}+C C_{1}\left(\gamma_{1}\right)_{k}\left(\gamma_{2}\right)_{k}+D D_{1}\left(\gamma_{1}\right)_{k}+E E_{1}\left(\gamma_{2}\right)_{k}+F F_{1}=0 \tag{6.33}
\end{equation*}
$$

where

$$
\begin{align*}
& A A_{1}=\left(D_{1}^{1}\right)^{2}-\left(D_{N}^{1}\right)^{2} \\
& B B_{1}=\left(D_{1}^{2}\right)^{2}-\left(D_{N}^{2}\right)^{2} \\
& C C_{1}=2\left(D_{1}^{1} D_{1}^{2}-D_{N}^{1} D_{N}^{2}\right) \\
& D D_{1}=2\left(D_{1}^{1} D_{1}^{3}-D_{N}^{1} D_{N}^{3}-\frac{L_{1}}{\Delta t}\right)  \tag{6.34}\\
& E E_{1}=2\left(D_{1}^{2} D_{1}^{3}-D_{N}^{2} D_{N}^{3}\right) \\
& F F_{1}=\left(D_{1}^{3}\right)^{2}-\left(D_{N}^{3}\right)^{2}+2 \frac{L_{1}}{\Delta t}\left(\gamma_{1}\right)_{k-1}+\left(\mathbf{V}_{1} \cdot \mathbf{n}_{1}\right)^{2}-\left(\mathbf{V}_{N} \cdot \mathbf{n}_{N}\right)^{2}
\end{align*}
$$

There is a similar expression for the Kutta condition for the second body:

$$
\begin{equation*}
A A_{2}\left(\gamma_{1}\right)_{k}^{2}+B B_{2}\left(\gamma_{2}\right)_{k}^{2}+C C_{2}\left(\gamma_{1}\right)_{k}\left(\gamma_{2}\right)_{k}+D D_{2}\left(\gamma_{1}\right)_{k}+E E_{2}\left(\gamma_{2}\right)_{k}+F F_{2}=0 \tag{6.35}
\end{equation*}
$$

where

$$
\begin{align*}
& A A_{2}=\left(D_{N+1}^{1}\right)^{2}-\left(D_{2 N}^{1}\right)^{2} \\
& B B_{2}=\left(D_{N+1}^{2}\right)^{2}-\left(D_{2 N}^{2}\right)^{2} \\
& C C_{2}=2\left(D_{N+1}^{1} D_{N+1}^{2}-D_{2 N}^{1} D_{2 N}^{2}\right) \\
& D D_{2}=2\left(D_{N+1}^{1} D_{N+1}^{3}-D_{2 N}^{1} D_{2 N}^{3}-\frac{L_{2}}{\Delta t}\right)  \tag{6.36}\\
& E E_{2}=2\left(D_{N+1}^{2} D_{N+1}^{3}-D_{2 N}^{2} D_{2 N}^{3}\right) \\
& F F_{2}=\left(D_{N+1}^{3}\right)^{2}-\left(D_{2 N}^{3}\right)^{2}+2 \frac{L_{2}}{\Delta t}\left(\gamma_{2}\right)_{k-1}+\left(\mathbf{V}_{N+1} \cdot \mathbf{n}_{N+1}\right)^{2}-\left(\mathbf{V}_{2 N} \cdot \mathbf{n}_{2 N}\right)^{2}
\end{align*}
$$

Since Equations (6.33) and (6.35) are coupled and nonlinear, Pang proposed linearizing the equations about the circulation values from the previous time step using:

$$
\begin{align*}
& \left(\gamma_{1}\right)_{k}=\left(\gamma_{1}\right)_{k-1}+\left(\delta \gamma_{1}\right)_{k}  \tag{6.37}\\
& \left(\gamma_{2}\right)_{k}=\left(\gamma_{2}\right)_{k-1}+\left(\delta \gamma_{2}\right)_{k}
\end{align*}
$$

where $\left(\delta \gamma_{1}\right)_{k}=0$ and $\left(\delta \gamma_{2}\right)_{k}=0$ in the first iteration. The linearized Kutta condition for the first body is:

$$
\begin{array}{r}
\left\{2 A A_{1}\left(\gamma_{1}\right)_{k-1}+C C_{1}\left(\gamma_{2}\right)_{k-1}+D D_{1}\right\}\left(\delta \gamma_{1}\right)_{k} \\
+\left\{2 B B_{1}\left(\gamma_{2}\right)_{k-1}+C C_{1}\left(\gamma_{1}\right)_{k-1}+E E_{1}\right\}\left(\delta \gamma_{2}\right)_{k}  \tag{6.38}\\
+A A_{1}\left(\gamma_{1}\right)_{k-1}^{2}+B B_{1}\left(\gamma_{2}\right)_{k-1}^{2}+C C_{1}\left(\gamma_{1}\right)_{k-1}\left(\gamma_{2}\right)_{k-1} \\
+D D_{1}\left(\gamma_{1}\right)_{k-1}+E E_{1}\left(\gamma_{2}\right)_{k-1}+F F_{1}=0 .
\end{array}
$$

Likewise, the linearized Kutta condition for the second body is:

$$
\begin{array}{r}
\left\{2 A A_{2}\left(\gamma_{1}\right)_{k-1}+C C_{2}\left(\gamma_{2}\right)_{k-1}+D D_{2}\right\}\left(\delta \gamma_{1}\right)_{k} \\
+\left\{2 B B_{2}\left(\gamma_{2}\right)_{k-1}+C C_{2}\left(\gamma_{1}\right)_{k-1}+E E_{2}\right\}\left(\delta \gamma_{2}\right)_{k}  \tag{6.39}\\
+A A_{2}\left(\gamma_{1}\right)_{k-1}^{2}+B B_{2}\left(\gamma_{2}\right)_{k-1}^{2}+C C_{2}\left(\gamma_{1}\right)_{k-1}\left(\gamma_{2}\right)_{k-1} \\
+D D_{2}\left(\gamma_{1}\right)_{k-1}+E E_{2}\left(\gamma_{2}\right)_{k-1}+F F_{2}=0 .
\end{array}
$$

Equations (6.38) and (6.39) are two equations in two unknowns, and they are solved in the box labeled "Impose Kutta condition to solve for circulation about bodies" on the flowchart in Figure 6.3. The values of $\left(\delta \gamma_{1}\right)_{k}$ and $\left(\delta \gamma_{2}\right)_{k}$ are used to update the guess by Equation (6.37). The process is repeated until $\left(\delta \gamma_{1}\right)_{k}$ and $\left(\delta \gamma_{2}\right)_{k}$ converge to within a desired tolerance. Then the circulation values at the current time step are updated using Equation (6.37).

### 6.4.5 Wake Convection

At each time step, the point vortices in the wake are convected at the local fluid velocity. This step partially corresponds to the box labeled "Advance vortices and swimmer position and orientation" on flowchart in Figure 6.3. The velocity compo-
nents $\left(U_{h}\right)$ at the $h^{\text {th }}$ vortex are:

$$
\begin{align*}
\left(U_{h}^{x}\right)_{k} & =\sum_{j=1}^{2 N}\left(A_{h, j}^{x}\right)_{k}\left(q_{j}\right) k+\left(\gamma_{1}\right)_{k} \sum_{j=1}^{N}\left(B_{h, j}^{x}\right)_{k}+\left(\gamma_{2}\right)_{k} \sum_{j=N+1}^{2 N}\left(B_{h, j}^{x}\right)_{k}  \tag{6.40}\\
& +\left(\gamma_{w 1}\right)_{k}\left(B_{h, w 1}^{x}\right)_{k}+\left(\gamma_{w 2}\right)_{k}\left(B_{h, w 2}^{x}\right)_{k}+\sum_{m=1}^{2(k-1)}\left(C_{h, m}^{x}\right)_{k}\left(\Gamma_{m}\right)_{k}+U_{\infty}^{x}
\end{align*}
$$

and

$$
\begin{align*}
\left(U_{h}^{y}\right)_{k} & =\sum_{j=1}^{2 N}\left(A_{h, j}^{y}\right)_{k}\left(q_{j}\right) k+\left(\gamma_{1}\right)_{k} \sum_{j=1}^{N}\left(B_{h, j}^{y}\right)_{k}+\left(\gamma_{2}\right)_{k} \sum_{j=N+1}^{2 N}\left(B_{h, j}^{y}\right)_{k}  \tag{6.41}\\
& +\left(\gamma_{w 1}\right)_{k}\left(B_{h, w 1}^{y}\right)_{k}+\left(\gamma_{w 2}\right)_{k}\left(B_{h, w 2}^{y}\right)_{k}+\sum_{m=1}^{2(k-1)}\left(C_{h, m}^{y}\right)_{k}\left(\Gamma_{m}\right)_{k}+U_{\infty}^{y}
\end{align*}
$$

For the $h^{\text {th }}$ vortex the coordinates are $\left(x_{h}, y_{h}\right)$, and we use the explicit Euler method to advance their position in time. At the $k^{\text {th }}$ time step, the position is $\left(\left(x_{h}\right)_{k},\left(y_{h}\right)_{k}\right)$. We assume that between time step $k$ and $k+1$, the velocity $\left(\left(U_{h}^{x}\right)_{k},\left(U_{h}^{y}\right)_{k}\right)$ of the vortex is constant. Hence, to determine the vortex position at time step $k+1$ we use

$$
\begin{align*}
& \frac{\left(x_{h}\right)_{k+1}-\left(x_{h}\right)_{k}}{\Delta t}=\left(U_{h}^{x}\right)_{k}  \tag{6.42}\\
& \frac{\left(y_{h}\right)_{k+1}-\left(y_{h}\right)_{k}}{\Delta t}=\left(U_{h}^{y}\right)_{k} . \tag{6.43}
\end{align*}
$$

### 6.4.6 Pressure Coefficient

Assume a body-fixed coordinate system $(x, y)$ and an inertial frame of reference $(X, Y)$ such that the two coincide at time $t=0$. Let the location of the origin of the body-fixed system be specified by $R(t)=\left(x_{0}, y_{0}\right)$ and the orientation relative to the horizontal by $\beta(t)$.

In the inertial frame of reference, the pressure coefficient on the surface of the
body can be derived directly from Equation (6.16) and is given by (see [57]):

$$
\begin{equation*}
C_{p}:=\frac{p-p_{\infty}}{\frac{1}{2} \rho\left\|\mathbf{U}_{\infty}\right\|^{2}}=1-\frac{\left\|\nabla_{X, Y} \phi\right\|^{2}}{\left\|\mathbf{U}_{\infty}\right\|^{2}}-\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}} \frac{\partial \phi}{\partial t} \tag{6.44}
\end{equation*}
$$

The difficulty in using this form of the equation arises in computing the time derivative of the velocity potential. Using the chain rule, the time derivative in the inertial frame is related to the time derivative in the body frame by:

$$
\begin{align*}
\frac{\partial}{\partial t}_{\text {inertial }} & =\frac{\partial}{\partial t}_{\text {body }}+\frac{\partial X}{\partial t} \frac{\partial}{\partial x}+\frac{\partial Y}{\partial t} \frac{\partial}{\partial y}  \tag{6.45}\\
& =\frac{\partial}{\partial t}_{\text {body }}+V \cdot \nabla_{x, y} \tag{6.46}
\end{align*}
$$

where $V=-\left[\mathbf{V}_{0}+\omega \times r\right]$ is the kinematic velocity of the freestream flow due to the motion of the body relative to a body-fixed reference frame. The velocity of the origin of the body frame is $\mathbf{V}_{0}=\left(\dot{x}_{0}, \dot{y}_{0}\right), \omega=\dot{\beta}$ is the rotation rate of the body frame and $r=(x, y)$ is the position vector in the body frame. Here, the gradient derivatives are with respect to the body coordinates.

Equation (6.44) can be expressed in the body reference frame by applying Equation (6.46) to yield:

$$
\begin{equation*}
C_{p}=\frac{p-p_{\infty}}{\frac{1}{2} \rho\left\|\mathbf{U}_{\infty}\right\|^{2}}=1-\frac{\left\|\nabla_{x, y} \phi\right\|^{2}}{\left\|\mathbf{U}_{\infty}\right\|^{2}}-\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}}\left[\mathbf{V}_{0}+\omega \times r\right] \cdot \nabla_{x, y} \phi-\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}} \frac{\partial \phi}{\partial t}, \tag{6.47}
\end{equation*}
$$

where now, $\nabla_{x, y} \phi=\left(\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}\right)$ is with respect to the body-fixed frame. Note that a change of coordinates does not affect the magnitude of $\nabla \phi$. The expressions for $\nabla \phi$
in the inertial and body frames, respectively, are:

$$
\begin{aligned}
\nabla_{X, Y} \phi & =\left(\frac{\partial \phi}{\partial X}, \frac{\partial \phi}{\partial Y}\right) \\
\nabla_{x, y} \phi & =\left(\frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}\right) \\
& =\left(\frac{\partial \phi}{\partial X} \cos \beta+\frac{\partial \phi}{\partial Y} \sin \beta,-\frac{\partial \phi}{\partial X} \sin \beta+\frac{\partial \phi}{\partial Y} \cos \beta\right) \\
& =\left(u^{x} \cos \beta+u^{y} \sin \beta,-u^{x} \sin \beta+u^{y} \cos \beta\right)
\end{aligned}
$$

Then the term $-\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}}\left[\mathbf{V}_{0}+\omega \times r\right] \cdot \nabla_{x, y} \phi$ is expressed as:

$$
\begin{aligned}
-\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}} & {\left[\mathbf{V}_{0}+\omega \times r\right] \cdot \nabla_{x, y} \phi=} \\
& -\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}}\left[\mathbf{V}_{0}+\omega \times r\right] \quad \cdot\left(u^{x} \cos \beta+u^{y} \sin \beta,-u^{x} \sin \beta+u^{y} \cos \beta\right) .
\end{aligned}
$$

The normal velocity component is equal to the normal velocity of the body at the point and the tangential velocity component is determined by an equation analogous to (6.25). These two components are projected onto the inertial axis frames to yield $u^{x}$ and $u^{y}$. On the other hand, $\left[\mathbf{V}_{0}+\omega \times r\right]$ is the velocity of the body at a point expressed in the inertial axis frame. Since the no-slip condition does not apply, these two terms are not equal.

Finally, $C_{p}$ has the form:

$$
\begin{align*}
C_{p}=1 & -\frac{\left(u^{x}\right)^{2}+\left(u^{y}\right)^{2}}{\left\|\mathbf{U}_{\infty}\right\|^{2}}-\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}} \frac{\partial \phi}{\partial t}  \tag{6.48}\\
& -\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}}\left[\mathbf{V}_{0}+\omega \times r\right] \cdot\left(u^{x} \cos \beta+u^{y} \sin \beta,-u^{x} \sin \beta+u^{y} \cos \beta\right) .
\end{align*}
$$

This expression, which corresponds to the section of the flowchart in Figure 6.3 labeled
"Compute pressure, \& forces on body", can be expressed in discrete form as:

$$
\begin{align*}
\left(C_{p, i}\right)_{k} & =1-\frac{\left\|\mathbf{u}_{i}\right\|_{k}^{2}}{\left\|\mathbf{U}_{\infty}\right\|^{2}}-\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}} \frac{\left(\phi_{i}\right)_{k}-\left(\phi_{i}\right)_{k-1}}{\Delta t} \\
& -\frac{2}{\left\|\mathbf{U}_{\infty}\right\|^{2}}\left[\left\{\left(v_{x}\right)_{k},\left(v_{y}\right)_{k}\right\}+\omega \times\left(r_{i}\right)_{k}\right]  \tag{6.49}\\
& {\left[\left(u_{i}^{x}\right)_{k} \cos \beta_{k}+\left(u_{i}^{y}\right)_{k} \sin \beta_{k},-\left(u_{i}^{x}\right)_{k} \sin \beta_{k}+\left(u_{i}^{y}\right)_{k} \cos \beta_{k}\right] }
\end{align*}
$$

The pressure over the body is integrated to compute an overall force and torque acting on the instantaneous center of mass, and the swimmer's motion is determined by Newton's equations:

$$
\begin{gather*}
\frac{d v^{\{x, y\}}}{d t}=\frac{F^{\{x, y\}}}{m}  \tag{6.50}\\
\frac{d(I \omega)}{d t}=I \frac{d \omega}{d t}+\omega \frac{d I}{d t}=M . \tag{6.51}
\end{gather*}
$$

Note that since the body shape changes at each time step, the moment of inertia of the body is not constant. Equations (6.50) and (6.51) are discretized by a forward Euler scheme:

$$
\begin{align*}
& \frac{\left(v^{\{x, y\}}\right)_{k}-\left(v^{\{x, y\}}\right)_{k-1}}{\Delta t}=\frac{\left(F^{\{x, y\}}\right)_{k}}{m}  \tag{6.52}\\
& I_{k} \frac{\left(\omega_{k}-\omega_{k-1}\right)}{\Delta t}+\omega_{k} \frac{I_{k}-I_{k-1}}{\Delta t}=M_{k} \tag{6.53}
\end{align*}
$$

Rearranging Equations (6.52) and (6.53), we solve for the linear and rotational velocities at the $k^{\text {th }}$ timestep,

$$
\begin{align*}
\left(v^{\{x, y\}}\right)_{k} & =\left(v^{\{x, y\}}\right)_{k-1}+\frac{\left(F^{\{x, y\}}\right)_{k}}{m} \Delta t  \tag{6.54}\\
\omega_{k} & =\frac{M_{k} \Delta t+I_{k} \omega_{k-1}}{2 I_{k}-I_{k-1}} \tag{6.55}
\end{align*}
$$

### 6.4.7 Velocity Potential

In this section we describe how to numerically compute the velocity potential when the integration path is crossed by point vortices in the flow. Most unsteady panel codes in the literature did not encounter this issue since the models often consider just one body or in the cases of two bodies, they are often placed so as to not interact with the wake of the other. This is one contribution we have made to extend the functionality of unsteady panel codes.

The velocity potential is needed to compute the pressure distribution over the body, which in turn is used to compute the forces and moment on the swimmer. In a potential flow, the following expression for the velocity potential holds:

$$
\begin{equation*}
\phi_{1}=\phi_{0}+\int_{0}^{1} \mathbf{u} \cdot \mathrm{~d} s \tag{6.56}
\end{equation*}
$$

In order to compute the pressure on the body, the temporal change in the value of the velocity potential is needed. Hence, we choose a point far enough upstream and away from the body such that $\phi_{0}$ is essentially constant. The potential is then found by computing the integral of the fluid velocity from this far away point to the body. The procedure for finding the velocity at a point in the fluid is nearly identical to that for finding the velocity at a point vortex - using Equations (6.40) and (6.41). However, instead of computing the velocity at the location of the point vortices, the location is that of the discretized path from upstream to the leading edge of the bodies.

Since the potential is multi-valued, care must be taken when the path of integration is crossed by point vortices in the wake. For example, a path originating and ending at the same point that encircles a vortex will result in a different potential value depending on the number of encirclements. In complex analysis, branch cuts are the curves in the complex plane across which multi-valued functions are discontinuous. These branch cuts originate at the singularities, known as branch points.


Figure 6.4: Point vortex crossing integration path. The branch cut of the vortex is indicated by the dashed line to the left of the vortex. Since the velocity potential field is multi-valued, when a vortex crosses the line of integration, the computed velocity potential must be incremented by the circulation of the vortex.

Thus in our application, branch cuts at each point vortex are required to achieve a single-valued scalar potential field.

To demonstrate this, consider Figure 6.4 showing a point vortex with a branch cut as it crosses an integration path. Since the value of the potential has a discontinuous jump across the branch, the integration path must be deformed to travel around the vortex. The two straight line paths are infinitesimally near each other and cancel out, leaving just the contribution of the path around the vortex. The integral of the velocity around the vortex is equal to the circulation of the enclosed vortex. In practice, this means that when a vortex in the wake "crosses" the integration path, the value of the computed potential must jump by the value of the circulation of that vortex. In the code, a path is chosen from far upstream to the leading edge of each body. The path is discretized into many line segments, and at the center of each segment the fluid velocity is computed. The tangential component of the velocity at each point is multiplied by the length of the particular element and summed over all the elements to compute the velocity potential at the leading edge. The potential along the foil surface is computed by continuing the integration along the top and bottom surfaces, by multiplying the tangential fluid velocity components by the panel lengths and summing over prior panels. This provides the potential value along the edges of the panels, and linear interpolation is used to compute the potential at the
control points.

### 6.4.8 Diverting Vortices

Since the point vortices in the wake are singularities, numerical difficulties arise when they approach too closely to the surface of the swimmer. A small change in the position of a point vortex can result in a very large change in the velocity induced along the body, which may preclude the code from converging to a solution at that time step. Also, the vortices may even enter within the body boundaries, since as noted previously, the no pentration boundary condition is satisfied only at the control points, and fluid 'leaks' in and out of the body everywhere else. In this section we propose and describe a simple method to address these issues.

Most prior numerical work did not involve the interaction of a wake with a solid body. One notable exception was the work of Zhu et al. [130] where a threedimensional panel method was used to study the flow structures shed from a fish-like model. To avoid the various numerical issues, including wake singularities convecting inside the body, they adopted a regularization technique [61] for the wake and surface panels. We found that although regularization improved the situation, the vortices can still convect inside the body since the no-penetration boundary condition only holds at the control points. To avoid these complications, a region is prescribed about each body which vortices are not permitted to enter. Any vortex which would enter the region is instead advanced as demonstrated in Figure 6.5. The intersections of the edge of the region with the circle with origin at the original vortex position (outside the region) and radius equal to the distance travelled during the time step are found. The vortex is instead advanced to the intersection that is nearest the originally-predicted vortex position. While the motivation for this is numerical, there is a plausible physical justification based on the existence of a boundary layer in viscous flows, which can be thought to extend the physical boundaries of the body [103, 102]. The code


Figure 6.5: Foil with surrounding region. Vortices are not permitted to enter the region around the foil identified by the dark dotted line in order to avoid numerical difficulties. A point vortex is initially at the position indicated by the black dot. Its new position would place it at the red dot - inside the region. The vortex is instead advanced to the position of the blue dot - the nearest intersection to the red dot of the region border and the circle with origin at the original position and radius equal to the initially predicted distance traveled.
validation in Section 6.6 suggests this approach yields satisfactory results.

### 6.5 Numerical Algorithm

In this section we bring all the previous components together and present the overall algorithm for the code. Refer back to Figure 6.3 for a flowchart with equation numbers labeled next to the corresponding sections. The equations that need to be solved are nonlinear and require an iterative procedure. The circulation about the foils at the previous time step and the position and strength of any point vortices in the flow are known. The procedure begins by guessing values for the length and orientation of the wake panels: $\left(\Delta_{w 1}\right)_{k},\left(\Delta_{w 2}\right)_{k},\left(\theta_{1}\right)_{k}$ and $\left(\theta_{2}\right)_{k}$. A good guess for each is the value at the $k-1$ time step. Since the overall motion of the swimmer as it self-propels through the fluid needs to be determined, we begin by setting the linear and angular velocities of the center of the body to the values at the previous time step. This leaves $2 N$ source distribution values and two vorticity distribution values to be solved for.

First, the Neumann boundary condition is imposed along both bodies so that the fluid velocity along the body surfaces is tangent to the moving body. The source distribution is expressed as a function of the two vorticity density strengths and a
constant, per Equation (6.10). Likewise, the velocity along the trailing edge panels is expressed as a function of the two vorticity density strengths plus a constant, following Equations (6.26) and (6.28)-(6.30). The two Kutta condition Equations (6.33) and (6.35) are linearized about the circulation density values from the previous time step, $\left(\gamma_{1}\right)_{k-1}$ and $\left(\gamma_{2}\right)_{k-1}$ as an initial guess. Equations (6.38) and (6.39) are solved, new estimates of $\left(\gamma_{1}\right)_{k}$ and $\left(\gamma_{2}\right)_{k}$ are found, and those values are used yet again to linearize Equations (6.33) and (6.35). The procedure is repeated until the circulation density values converge within a desired tolerance.

With these values known, the source density along the body is found from Equation (6.10). With all values now known, the velocity induced at the midpoint of the wake panel is recomputed by summing the induced velocity due to the source and vorticity distributions, the point vortices in the wake and the freestream velocity. The velocity is used to update the wake panel lengths and orientation via Equations (6.5) and (6.6). The vorticity density along the wake panels is found from Equations (6.3) and (6.4). Since the length and orientation of the wake panel geometry variables has likely changed, the above procedure is repeated with the new values until all unknown variables converge to within a desired tolerance.

Once all values have converged, the pressure along the body is computed by Equation (6.48) and the forces and moment on the swimmer are computed. Since we seek to determine how the swimmer self-propels through the fluid, the resultant body velocity due to the computed forces must be consistent with the Neumann boundary condition imposed on the swimmer. This will require another iterative process, but extra care must be taken since a slight change in the boundary condition imposed on the body may result in a significant change in the computed forces, and the process may become unstable. As a result, a relaxation scheme is adapted from Carling [19] who used it to find the self-propulsion of anguilliform swimmers.

In Carling's relaxation scheme, instead of directly using the computed forces and
moment to update the velocity of the swimmer, weighted forces $\bar{F}$ and moment $\bar{M}$ are computed as follows:

$$
\begin{gather*}
\overline{(F}\{x, y\}_{k}=\xi\left(F^{\{x, y\}}\right)_{k}+(1-\xi)\left(F^{\{x, y\}}\right)_{k-1},  \tag{6.57}\\
\bar{M}_{k}=\xi M_{k}+(1-\xi) M_{k-1}, \tag{6.58}
\end{gather*}
$$

where $\xi$ is chosen between 0.5 and 1.0. Carling [19] found that a value of $\xi=0.75$ was optimal. Lower values resulted in unstable computations while computations with larger values were less accurate.

Given the weighted forces and moment on the body, $\overline{F_{x}}, \overline{F_{y}}$, and $\bar{M}$, we apply Equations (6.54) and (6.55) to compute new values for the body velocity components:

$$
\begin{equation*}
\left(\bar{v}^{\{x, y\}}\right)_{k}=\left(v^{\{x, y\}}\right)_{k-1}+\frac{\left(\bar{F}^{\{x, y\}}\right)_{k}}{m} \Delta t \tag{6.59}
\end{equation*}
$$

and

$$
\begin{equation*}
\bar{\omega}_{k}=\frac{\bar{M}_{k} \Delta t+I_{k} \omega_{k-1}}{2 I_{k}-I_{k-1}} \tag{6.60}
\end{equation*}
$$

Yet even this velocity computed from the weighted force may lead to large jumps in the forces. Carling [19] suggested cautiously adjusting the velocity value at each iteration as follows:

$$
\begin{gather*}
\left(v^{\{x, y\}}\right)_{k}^{p+1}=\gamma_{\{x, y\}}\left(\bar{v}^{\{x, y\}}\right)_{k}+\left(1-\gamma_{\{x, y\}}\right)\left(v^{\{x, y\}}\right)_{k}^{p}  \tag{6.61}\\
\omega_{k}^{p+1}=\gamma_{\omega} \bar{\omega}_{k}+\left(1-\gamma_{\omega}\right) \omega_{k}^{p} \tag{6.62}
\end{gather*}
$$

where $\gamma_{\{x, y, \omega\}}$ are the weighting parameters for the corresponding velocity components and $p$ is the iteration number. Equations (6.59)-(6.62) correspond to the step labeled "Compute body velocities" on the algorithm flowchart in Figure 6.3.

With the new velocity estimates $\left(v^{\{x, y\}}\right)_{k}^{p+1}$ and $\omega_{k}^{p+1}$, the new estimated position
and orientation of the body can be computed from Equations (6.11)-(6.13). With the new guess for the position on the body, a new boundary condition can be found from Equations (6.14) and (6.15), and the process iterated until the forces computed converge to a desired accuracy. Finally, the position of the vortices as well as the swimmer are advanced. In the numerical experiments presented in the next chapter, we used $\gamma_{x}=0.7$. In our study, we constrained the motion so that it could not rotate or translate laterally, so the other two parameters were not needed.

### 6.6 Code Validation

The two-body code was validated against results from Tuncer and Platzer [119], who utilized a multiblock Navier-Stokes solver to compute the flow about tandem NACA0012 airfoils. They also employed an unsteady potential flow solver (UPOT) for one flapping airfoil to validate their Navier-Stokes solver.

We used their result for a single heaving foil to validate our two foil code by placing the bodies far enough apart that there would be no noticeable hydrodynamic interaction between them. The two bodies are 10 chord lengths apart vertically and are both prescribed the same relative heaving motion $y(t)=-0.1 c \cos (3 t)$. Refer to Figure 6.6 for a schematic of the layout and coordinate system. The non-dimensional reduced frequency is defined as $[119,18]$ :

$$
\begin{equation*}
k=\frac{\omega c}{2 U_{\infty}} \tag{6.63}
\end{equation*}
$$

where $\omega$ is the frequency of oscillation, $c$ is the chord length of the foil and $U_{\infty}$ is the speed of the freestream flow. In this case $k=1.5$.

Figure 6.7 shows the locations of the two foils with their wakes near time $2 \pi \frac{c}{U_{\infty}}$. Also shown in dashed lines are the integration paths used to compute the potential originating from the point $(-10 c, 7 c)$ to the leading edge of both foils.


Figure 6.6: Schematic of layout and coordinate system for Tuncer and Platzer [119] validation case. Not to scale.


Figure 6.7: Layout and integration paths not crossing wake for two foil code validation. The dashed lines are the integration paths to compute the velocity potential. Neither integration path is crossed by wake vortices. The results of this validation case are shown in Figures 6.8 and 6.9.


Figure 6.8: Integration path not crossing vortex wake: Unsteady drag coefficient vs time compared to the potential flow solver of Tuncer and Platzer [119]. The geometry of the system is as shown in Figure 6.7 and each foil is moved with the relative motion $y(t)=-0.1 c \cos (3 t)$.

Since the bodies are sufficiently far apart, the unsteady drag and lift coefficients computed over the two bodies are nearly indistinguishable from each other and are shown in Figures 6.8 and 6.9, respectively. There is very good agreement with the potential flow results of Tuncer and Platzer.

To ensure that the algorithm for computing the velocity potential is able to accurately account for vortices that cross the path of integration, we repeated this test case but with the lower foil shifted back slightly behind the top foil, as shown in Figure 6.10. Now, the integration path crosses the vortex wake. The unsteady drag and lift coefficients for this case are shown in Figures 6.11 and 6.12. Again, there is excellent agreement with the Tuncer and Platzer results, and the results are unaffected by the integration path chosen.

Next, the two bodies were placed one behind the other - with one chord length spacing in between. This time, only the leading foil was moved with the same motion


Figure 6.9: Integration path not crossing vortex wake: Unsteady lift coefficient vs time compared to the potential flow solver of Tuncer and Platzer [119]. The geometry of the system is as shown in Figure 6.7 and each foil is moved with the relative motion $y(t)=-0.1 c \cos (3 t)$.


Figure 6.10: Layout and integration path crossing wake for two foil code validation. The dashed lines are the integration paths to compute the velocity potential. The integration path for the lower foil runs through the wake of the upper foil and is crossed by wake vortices. The results of this validation case are shown in Figures 6.11 and 6.12. This validation case was performed to demonstrate that the code accurately accounts for vortices crossing the integration path


Figure 6.11: Integration path crossing vortex wake: Unsteady drag coefficient vs time compared to the potential flow solver of Tuncer and Platzer [119]. The geometry of the system is as shown in Figure 6.10 and each foil is moved with the relative motion $y(t)=-0.1 c \cos (3 t)$. The results of this case match up with those in Figure 6.8


Figure 6.12: Integration path crossing vortex wake: Unsteady lift coefficient vs time compared to the potential flow solver of Tuncer and Platzer [119]. The geometry of the system is as shown in Figure 6.10 and each foil is moved with the relative motion $y(t)=-0.1 c \cos (3 t)$. The results of this case match up with those in Figure 6.9


Figure 6.13: Layout, integration paths and wake vortex distribution at $t=10 \frac{c}{U_{\infty}}$ for two foil code validation against Tuncer and Platzer [119] Navier Stokes solver. The leading foil is prescribed a heaving motion of $y(t)=-0.1 c \cos (3 t)$ while the trailing foil remains stationary. The simulation is compared to the results of Tuncer and Platzer in Figures 6.14 and 6.15.
as before, while the trailing foil was maintained stationary. The time step used was $\Delta t=0.1 \frac{c}{U_{\infty}}$ and a region of a normal distance of $0.08 c$ within which vortices were not allowed to enter was established around the trailing foil. This distance was found by numerical experimentation and chosen to ensure that all the numerical experiments converged for the desired parameter range considered in this study. No such region was established or needed for the leading foil since it did not encounter any point vortices. The geometry, velocity potential integration paths and wake after $t=10 \frac{c}{U_{\infty}}$ are shown in Figure 6.13. The unsteady drag and lift over the leading and trailing foils is compared to the result from the Navier-Stokes solver in Figures 6.14 and 6.15, respectively.

For the leading foil, the computed values for lift and drag are nearly identical to those found before when the two bodies were hydrodynamically isolated. The drag coefficients computed for the stationary trailing foil match up well. The peaks are slightly over-estimated. The lift coefficient is not estimated as well, perhaps due to the inability of the code to account for frictional forces. Still, the agreement is qualitatively sufficiently good since this may be considered a worst case scenario


Figure 6.14: Code validation: Unsteady drag coefficient vs time. The results of the current study are shown in blue and red for the leading and trailing foils, respectively. The results of Tuncer and Platzer [119] appear as dashed and solid black lines for the leading and trailing foils, respectively. The dotted black line indicates the heaving motion of the leading foil. Our results are superimposed on the actual figure in Tuncer and Platzer.


Figure 6.15: Code validation: Unsteady lift coefficient vs time. The results of the current study are shown in blue and red for the leading and trailing foils, respectively. The results of Tuncer and Platzer [119] appear as dashed and solid black lines for the leading and trailing foils, respectively. The dotted black line indicates the heaving motion of the leading foil. Our results are superimposed on the actual figure in Tuncer and Platzer.
where the stationary body experiences relatively lower pressure forces.
For a more quantitative comparison, consider the propulsive efficiency which we define as

$$
\begin{equation*}
\eta_{P}=\frac{\bar{T} \cdot\left\|\mathbf{U}_{\infty}\right\|}{\bar{P}} \tag{6.64}
\end{equation*}
$$

where $\bar{T}$ and $\bar{P}$ are the average thrust produced and power input into the system, respectively, over one period of motion. The power is time rate of change of work, $\frac{d W}{d t}$, or equivalently, the product of the bodies' instantaneous forces and velocities. For the case of a simple heaving body, the instantaneous work is the product of the derivative of the $y$ position with respect to time and the instantaneous lift. We computed the efficiency for the cases when one foil is heaving on its own as well as when a second stationary foil is added in its wake, as in Figure 6.13. The results are summarized in Table 6.3. For the isolated foil case, the computed efficiency matches that found by

Table 6.3: Propulsive efficiency comparison

|  | current code | Tuncer and Platzer |
| :--- | :---: | :---: |
| Heaving foil only | $37 \%$ | $37 \%$ |
| Heaving leading foil + stationary trailing foil | $53 \%$ | $56 \%$ |

Tuncer and Platzer's Navier-Stokes simulation. For the case where a second foil was added, we slightly underestimate the efficiency as $53 \%$ compared to the $56 \%$ found by Tuncer and Platzer's Navier-Stokes solver. The discrepancy may be attributed to the way in which the trailing foil interacts with the vorticity shed from the leading body or the fact that our code does not account for frictional forces. It should be noted again that point vortices were not allowed to enter within a distance of $0.08 c$ of the trailing foil, but were instead diverted around the body. Despite the fact that the Navier-Stokes solver also accounts for the effects of viscosity, while our solution only includes forces due to pressure, the results agree well.

## Chapter 7

## Control Through a Vortex Wake

This chapter presents a method for controlling the motion of a swimmer through a nearly-periodic vortex wake in order to produce a near-optimal thrust-generating gait.

Section 7.1 begins with a brief review of prior work on motion through a vortex wake. We discuss how the effective angle of attack is a useful metric for determining the lift on a body undergoing unsteady motion, and it is shown that the position of the leading stagnation point on a foil may serve as an acceptable proxy for determining the phase and frequency of the true effective angle of attack of a body. The results of the numerical experiments performed suggest that the steady state thrust coefficient for a two foil swimmer is nearly optimized when these two quantities - the effective angle of attack of the trailing foil (due to only the motion of the foil while ignoring the influence of the wake) and the stagnation point position on the leading edge of the trailing foil - are in phase.

Before presenting our own control strategy, we review recent attempts in the literature to model and control fluid systems in Section 7.2. While much of the prior work required estimating the location of flow structures throughout the flow field from measured quantities, our formulation is unique in that it requires knowing flow


Figure 7.1: Schematic of two foil swimmer. The shape of the swimmer is defined by the time-varying heaving amplitudes of the leading and trailing foils, $y_{1}(t)$ and $y_{2}(t)$, respectively along with their respective pitching amplitudes, $\beta_{1}(t)$ and $\beta_{2}(t)$. A uniform freestream velocity exists to the right, and the positive thrust direction is to the left. The filled circles indicate point vortices shed from the trailing edges of the foils.
quantities only on the surface of the swimmer so that the position of the stagnation point can be determined.

In Section 7.3 we describe the heuristic control objective based on the results of the numerical experiments performed on the swimmer - shown in Figure 7.1. A phase-locked loop controller is designed to synchronize the phase of the effective angle of attack of the trailing foil with the position of the leading stagnation point on the trailing foil by adjusting the foil's pitching and heaving frequency. We note that the classical phase-locked loop and the corresponding simplified theory do not strictly apply since the feedback variable is a more complex function of the phase than in the classical case. Despite this complication, an appropriately-tuned controller is able to achieve the control objective.

We demonstrate the effectiveness of the controller with an example in Section 7.4. Whereas the swimmer's overall position and orientation is fixed in the experiments performed in Section 7.1, we implement the controller on a swimmer that is free to propel itself in the forward direction in Section 7.4. Since the fluid is inviscid and we wish to reach a steady-state swimming speed, we impose a drag force opposing the motion of the swimmer and proportional to the square of the speed. The
appropriately-tuned controller is able to synchronize the phase of the effective angle of attack and stagnation point position, although the lock-in time varies depending on the initial starting conditions as well as the controller parameters. For comparison, a series of numerical experiments are conducted where the phase between the leading and trailing foil is fixed for each trial and the swimmer is allowed to reach a steady state swimming speed. It is shown that the highest thrust-producing phase value corresponds to nearly the same phase achieved by the controller. We also present a second example where the frequency of the swimmer's motion is changed to a smaller value to demonstrate that the controller is still able to achieve its objective.

Finally, we provide an example to demonstrate the benefits of controller-based feedback. This example compares the performance of two swimmers - one which is unable to adapt to changing conditions (no feedback) and another that uses the phaselocked loop controller. We have a swimmer that is initially prescribed a constantfrequency leading and trailing foil pitching and heaving motion with the appropriate leading to trailing foil phase lag to achieve maximum steady state thrust. The motion of the trailing foil remains constant, but the frequency of the leading foil is later reduced and then increased to simulate an unpredictable environment or unmodeled dynamics. We compare the resulting motion of the cases with and without control and show that not only does the controller generate a faster gait, but also that the propulsive efficiency is significantly improved.

### 7.1 Development of a Heuristic Control Objective Through Numerical Experiments

With the numerical model described in Chapter 6, we can prescribe various openloop gaits - cyclic shape changes in time - and observe how the swimmer propels through the fluid. This simply provides us with the resulting motion if the gait is
known in advance.
But the characteristics of an "optimal" gait are not immediately obvious. As the swimmer moves through the fluid, it may encounter a time-dependent vortex wake. It is reasonable to expect that introducing an appropriate control algorithm which takes into account the changing fluid environment could improve the swimmer's performance.

In this section we present the numerical experiments performed with our model to gain an improved understanding of the system hydrodynamics. We show that the effective angle of attack and the leading edge stagnation point position play an effective role in determining the performance of the swimmer. We use the results of the study to formulate a physically-based, heuristic control objective for improvedperformance swimming.

### 7.1.1 Prior Work: Swimming Through a Vortex Wake

We expect that the motion of the swimmer relative to the vortex wake plays an important role in the swimmer's performance. Previous experimental and numerical work on similar systems provide guidance for our own study.

Liao et al. [68] studied the motion of a trout in the periodic vortex wake of an upstream cylinder and found that the fish slalom between vortices as a way of extracting energy from the oncoming flow. Slaloming describes motion in which the fish moves with the lateral flow and against the downstream flow. They found that this approach minimizes power input and increases propulsive efficiency.

Beal et al. [11] summarized the main modes of vortex-foil interaction as vortex interception mode and slaloming mode. In vortex interception mode, the leading edge of the foil travels through the vortices, increasing thrust and significantly modifying the downstream wake. In slaloming mode, the foil travels between, but not through, the oncoming vortices. In this case the wake generated consists of pairs of vortices

- one from upstream and one generated by the foil. Vortex interception mode was found to increase thrust, while slaloming mode improved propulsive efficiency.

In experimental studies of a passive foil through a wake, Beal [11] identified two mechanisms responsible for improving performance by appropriately maneuvering through the oncoming flow. The first is the relative angle of attack due to the lateral flow which is responsible for producing lift and thrust. As observed by others, the forces are larger when moving against the lateral flow. The other mechanism generates thrust through suction from oncoming vortices interacting with the leading edge of the foil and is dependent on the timing between the lateral position of the foil and the position of the oncoming vortices.

Beal [11] also studied a euthanized fish in the wake of a cylinder and found that it was able to passively propel itself upstream by extracting energy from the vortex structures in the flow. Both fish and high aspect ratio foils were found to be highly efficient in slalom modes, or in an equivalent mode where opposite-sign vortices are shed from the trailing edge to pair with the oncoming vortex.

In a numerical study of tandem foils designed to mimic the motion of sunfish fins, Akhtar et al. [2] demonstrated that the thrust and propulsive efficiency are very sensitive to the timing (phase difference) between the leading and trailing foil.

### 7.1.2 Effective Angle of Attack

Two quantities will play an important role in the control algorithm of our swimmer. One is what we will refer to as the effective angle of attack, although strictly speaking, we really mean an approximate effective angle of attack that accounts for the unsteady motion of the foil but does not account for the influence of a non-uniform wake, such as vortices. Still, we will show how this quantity is very useful for controlling the motion of a swimmer through a vortex wake.

In steady motion with attached flow, the lift produced by a foil is proportional to
the angle of attack - the angle between the foil's chordline and the velocity of the foil motion relative to the freestream. But when a foil undergoes unsteady motion, the amount of lift it generates is a function of more than just the angle of attack. The unsteady motion generates additional forces on the foil, and these forces are accounted for by the effective angle of attack. This quantity serves as a meaningful metric of the lift-generating capability of a foil's unsteady motion. Similarly, we note that flow structures such as vortices around the foil may also enhance or detract from the lift produced by the body. It is beyond the scope of this study to attempt to model this very complicated relationship, so while we will not include the effect of non-uniform flow distributions in our definition of the effective angle of attack, we demonstrate that our definition is still useful in the context of the controller which we will introduce later.

We consider a foil that is propelling at a non-uniform speed in the forward direction as a result of vertical heaving and pitching about a fixed point relative to the foil. We define the effective angle of attack as the sum of three components: the angle of incidence $\left(\alpha_{\text {eff, }}\right)$ between the chordline of the foil and its motion relative to the freestream, a heaving motion component ( $\alpha_{\text {eff,h }}$ ) and a pitching motion component $\left(\alpha_{\text {eff,p }}\right)$. Refer to Figure 7.2. Since the freestream velocity is horizontal and to the right and we will constrain the swimmer so that it may only propel forward or backward relative to the freestrem velocity, the angle of incidence component is simply $\alpha_{\mathrm{eff}, \mathrm{i}}=\beta$, the prescribed angle of the foil relative to the freestream velocity. For the other two components, we use a assume a small angle approximation. The heaving component results in an angle of attack that is constant along the length of the chord: $\alpha_{\text {eff,h }}=$ $\dot{y} / V_{\text {rel }}$, where $V_{\text {rel }}=\left\|\mathbf{V}_{\infty}\right\|-v^{x}$ is the freestream velocity component in the $x$-direction relative to the foil and $\dot{y}$ is the heaving (lateral) velocity of the foil. This component has a fairly straightforward physical interpretation. In the frame of the foil, the relative freestream velocity due to the heaving motion is at an angle of $\tan ^{-1}\left(\dot{y} / V_{\text {rel }}\right)$,
$\qquad$
(a)

(b)

(c)

Figure 7.2: Three components of effective angle of attack. Here we show the three components for the trailing foil. (a) The angle of incidence between the chordline of the foil and its motion relative to the freestream: $\alpha_{\text {eff, },}=\beta_{2}$. (b) A heaving motion component: $\alpha_{\text {eff,h }}=\dot{y} / V_{\text {rel }}$. (c) A pitching component: $\alpha_{\text {eff,p }}(x)=\frac{\dot{\beta} c)}{V_{\text {rel }}}$, where $c$ is the chord length and $V_{\text {rel }}$ is the speed of the swimmer relative to the freestream velocity in the direction of the freestream velocity.
or $\dot{y} / V_{\text {rel }}$ in the small-angle approximation. Thus, the contribution to the effective angle of attack due to the heaving motion is $\dot{y} / V_{\text {rel }}$. The pitching component is less straightforward. Since the velocity of the foil relative to the fluid varies along the chord while pitching, the angle of attack is a function of the chord position:

$$
\begin{equation*}
\alpha_{\mathrm{eff}, \mathrm{p}}(x)=\frac{\dot{\beta}\left(x-x_{0}\right)}{V_{\mathrm{rel}}} \tag{7.1}
\end{equation*}
$$

where $x_{0}$ is the point about which the foil is pitching and $\dot{\beta}$ is the rotational velocity of the foil. In this study, $x_{0}=\frac{c}{4}$, the quarter-chord point. The pitching motion effectively changes the camber of the foil and hence, the effective angle of attack. For a foil with parabolic shape, it has been shown that the angle at the $\frac{3}{4}$-chord point is the best representation for the effective angle of attack [121]. Thus, substituting $x=\frac{3 c}{4}$ in Equation (7.1) yields a 'representative' pitching effective angle of attack component: $\alpha_{\text {eff,p }}=\frac{\dot{\beta} c / 2}{V_{\text {rel }}}$.

In the absence of upstream vorticity, the effective angle of attack for a foil pitching
about the quarter-chord point may then be expressed as

$$
\begin{align*}
\alpha_{\mathrm{eff}} & =\alpha_{\mathrm{eff}, \mathrm{i}}+\alpha_{\mathrm{eff}, \mathrm{~h}}+\alpha_{\mathrm{eff}, \mathrm{p}}  \tag{7.2}\\
& =\beta+\frac{\dot{y}+\dot{\beta} c / 2}{V_{\mathrm{rel}}} \tag{7.3}
\end{align*}
$$

### 7.1.3 Stagnation Point

Recall that in the previous section, our expression for the effective angle of attack consisted of just three components - the angle of incidence, a heaving component and a pitching component. In reality, other factors may influence the "true" effective angle of attack as well. For example, when a foil is moving through a fluid with non-uniform velocity distribution, determining the actual effective angle of attack is considerably more difficult. In this case, the velocity of the fluid along the body may vary spatially and temporally as a result of the vorticity in the fluid. Even if the precise velocity is known, it is still not obvious what the effective angle of attack should be. We will demonstrate that one potential alternative to computing this value from the fluid and kinematic properties is to consider the stagnation points on the foil. One stagnation point will always remain fixed at the trailing edge, while a second stagnation point will vary in position near the leading edge as the foil moves through the fluid. We will later demonstrate how the value of our simplified expression for the effective angle of attack along with the position of the stagnation point position can be used as part of a controller.

Anderson et al. [4] showed that the effective angle of attack can be related to the position of the stagnation point. For steady potential flow, Anderson's results indicate that the distance from the leading edge along the surface of the foil of the forward stagnation point is monotonically increasing with effective angle of attack, though the exact relationship varies depending on whether or not there is circulation around the foil. Since we consider the unsteady motion of a foil through a fluid for which
the circulation about the body will vary, it is not guaranteed that such a monotonic relationship exists.

Due to the geometry of our swimmer (see Figure 7.1), the trailing foil will interact with the vorticity wake shed from the leading foil. Thus, the effective angle of attack determined from Equation (7.3) is not a "true" effective angle of attack since it fails to account for the effect of the vorticity. On the other hand, the leading foil encounters no such vortex wake, so we expect this approximate expression to be close to the true effective angle of attack. Hence, we may use the leading foil to check the relationship between the leading stagnation point position and our expression for the effective angle of attack. The stagnation point is determined numerically by finding the zero velocity location through linear interpolation of the velocity values at the control points. We perform a numerical experiment and plot the resulting stagnation point position relative to the leading edge along with the effective angle of attack for the leading foil in Figure 7.3. Both quantities are normalized to unity at steady-state. Although there is some small leading or lagging between the two variables at times, they are sufficiently in phase that the stagnation point location serves as an acceptable proxy for determining the phase and frequency of the effective angle of attack for the parameter space considered in this study. This greatly simplifies the otherwise more difficult challenge of directly determining the effective angle of attack of the trailing foil as it travels through a non-uniform vortex wake.

### 7.1.4 Numerical Experiments

As observed in prior work, the timing of the motion of the body relative to the wake appears to play a critical role in determining swimming performance. A series of numerical experiments were performed to understand this relationship in our system.

Instead of studying the motion for a free-moving swimmer, we first consider the swimmer changing its shape by moving its front and rear foils while remaining pegged


Figure 7.3: Normalized effective angle of attack and stagnation point position along the surface of the leading foil versus time. In the parameter space considered here, the two values are nearly in phase, suggesting that the stagnation point location is an acceptable proxy for the effective angle of attack. For the red curve, the $y$ axis represents the (scaled) distance from the leading edge along the surface of the foil.
down and unable to propel itself. We are interested in measuring the average thrust coefficient of the swimmer.

Since the wake is periodic at steady state, the average thrust coefficient $C_{T}$ is defined as

$$
\begin{equation*}
C_{T}=\frac{\langle T\rangle}{\frac{1}{2} \rho V_{\mathrm{rel}}^{2}}, \tag{7.4}
\end{equation*}
$$

where the $\langle T\rangle$ represents the thrust averaged over one period of motion, $t_{\omega}=\frac{2 \pi}{\omega}$. We note that in this chapter (particularly in the results in Figure 7.6 where the swimmer is permitted to self-propel), we adopt a different non-dimensionalization of the thrust. Since a higher thrust-producing swimmer results in a higher swimming speed, nondimensionalizing by the relative velocity between the swimmer and the freestream would result in lower thrust coefficients for the fastest gaits. Instead, we use the chord length $c$ and period of motion, $t_{\omega}$ to non-dimensionalize the thrust as follows:
$C_{T}=\langle T\rangle /\left(\frac{1}{2} \rho\left(c / t_{\omega}\right)^{2}\right)$.
In these experiments, each foil is prescribed its own heaving $y(t)$ and pitching $\beta(t)$ motion, where we have assumed that the frequency of oscillation is the same for pitching and heaving of both foils:

$$
\begin{align*}
& \beta_{1}=A \sin (\omega t)  \tag{7.5}\\
& y_{1}=h \sin \left(\omega t-\frac{\pi}{2}\right)  \tag{7.6}\\
& \beta_{2}=A \sin \left(\omega t+\phi_{2}\right)  \tag{7.7}\\
& y_{2}=h \sin \left(\omega t-\frac{\pi}{2}+\phi_{2}\right) \tag{7.8}
\end{align*}
$$

where $-\frac{\pi}{2}$ is the phase between pitching and heaving for both foils and $\phi_{2}$ is the phase lag between the leading and trailing foil. We note that to avoid numerical convergence difficulties associated with the start of a gait, all pitching and heaving motions discussed in this thesis are also multiplied by an exponentially decaying envelope that quickly approaches unity $(1-\exp (-t))$. We fix the heaving amplitude to $h=0.05 c$ and vary the pitching amplitude $A$ and frequency $\omega$. The heaving amplitude was chosen such that the pitching and heaving effective angle of attack components would be of the same order of magnitude. Since we expect performance to be a strong function of the timing between the vortex wake of the trailing foil and the motion of the trailing foil, for each combination of $A$ and $\omega$, we vary the phase difference between the leading and trailing foil $\left(\phi_{2}\right)$ from 0 to $2 \pi$ radians in increments of $36^{\circ}$.

Six frequency values and two pitching amplitude values were chosen. We note in particular that the pitching amplitude values chosen preclude applying any results to large-amplitude motions without further investigation. Table 7.1 contains a summary of the parameter values for the 120 numerical experiments performed.

Table 7.1: Experiment parameter values

| $\omega$ | $A$ | $\phi_{2}$ |
| :---: | :---: | :---: |
| 2 | $6^{\circ}$ | $0^{\circ}, 36^{\circ}, 72^{\circ}, 108^{\circ}, 144^{\circ}, 180^{\circ}, 216^{\circ}, 252^{\circ}, 288^{\circ}, 324^{\circ}$ |
| 3 | $6^{\circ}$ | $0^{\circ}, 36^{\circ}, 72^{\circ}, 108^{\circ}, 144^{\circ}, 180^{\circ}, 216^{\circ}, 252^{\circ}, 288^{\circ}, 324^{\circ}$ |
| 4 | $6^{\circ}, 9^{\circ}$ | $0^{\circ}, 36^{\circ}, 72^{\circ}, 108^{\circ}, 144^{\circ}, 180^{\circ}, 216^{\circ}, 252^{\circ}, 288^{\circ}, 324^{\circ}$ |
| 5 | $6^{\circ}, 9^{\circ}$ | $0^{\circ}, 36^{\circ}, 72^{\circ}, 108^{\circ}, 144^{\circ}, 180^{\circ}, 216^{\circ}, 252^{\circ}, 288^{\circ}, 324^{\circ}$ |
| 6 | $6^{\circ}, 9^{\circ}$ | $0^{\circ}, 36^{\circ}, 72^{\circ}, 108^{\circ}, 144^{\circ}, 180^{\circ}, 216^{\circ}, 252^{\circ}, 288^{\circ}, 324^{\circ}$ |
| 7 | $6^{\circ}, 9^{\circ}$ | $0^{\circ}, 36^{\circ}, 72^{\circ}, 108^{\circ}, 144^{\circ}, 180^{\circ}, 216^{\circ}, 252^{\circ}, 288^{\circ}, 324^{\circ}$ |

### 7.1.5 Results

As an example, consider the set of gaits where $\omega 1=\omega_{2}=7, A=9^{\circ}$ and $\phi_{2}$ is varied between 0 and $2 \pi$ radians. In Figure 7.4, we show the effective angle of attack and the forward stagnation point position of the leading foil versus time for all ten cases. Both are normalized to unity at steady state. It appears that the thrust coefficient at steady state is maximized when the two curves are nearly in phase. To examine this possibility, we numerically determined the phase difference between the two curves, $\phi_{D}$, by identifying the first Fourier mode coefficients of each curve at steady state and subtracting the phases of the subsequent curve fits. Table 7.2 summarizes the results and includes the thrust coefficient, $C_{T}$ and phase difference between the stagnation point position and effective angle of attack curves at steady state, $\phi_{D}$ for each of the cases considered here. These results indicate that the thrust coefficient is near a maximum when the effective angle of attack and stagnation point positions are in phase. We repeat this procedure for all values of $\omega$ and $A$ noted in Table 7.1 and plot the thrust coefficient versus phase difference in Figure 7.5. Finally, a similar series of experiments are performed with the same parameters as in Table 7.1 and where the swimmer is able to self-propel. The results are shown in Figure 7.6.


Figure 7.4: Normalized effective angle of attack and leading edge stagnation point position along the surface of the trailing foil versus time for a series of experiments. In these experiments, $\omega_{1}=\omega_{2}=7$ and $A=9^{\circ}$ and $\phi_{2}$ is varied from 0 to $2 \pi$ radians. The steady state average thrust coefficient and phase angle between the effective angle of attack and leading foil stagnation point position for each case is summarized in Table 7.2. It appears that thrust is nearly maximized when the effective angle of attack and stagnation point positions are in phase.

Table 7.2: Results for $\omega_{1}=\omega_{2}=7, A=9^{\circ}$ experiments for a fixed swimmer. The swimmer is constrained to prevent self-propulsion. The thrust coefficient, $C_{T}$ versus $\phi_{D}$ is plotted in Figure 7.5 along with the corresponding data for other combinations of $\omega$ and $A$.

| $\omega$ | $A$ | $\phi_{2}$ | $\phi_{D}$ | $C_{T}$ |
| :---: | :---: | :---: | :---: | :---: |
| 7 | $9^{\circ}$ | $0^{\circ}$ | -1.67 | 0.508 |
| 7 | $9^{\circ}$ | $36^{\circ}$ | -2.37 | 0.446 |
| 7 | $9^{\circ}$ | $72^{\circ}$ | 3.09 | 0.446 |
| 7 | $9^{\circ}$ | $108^{\circ}$ | 2.44 | 0.485 |
| 7 | $9^{\circ}$ | $144^{\circ}$ | 1.80 | 0.5815 |
| 7 | $9^{\circ}$ | $180^{\circ}$ | 1.30 | 0.639 |
| 7 | $9^{\circ}$ | $216^{\circ}$ | 0.80 | 0.692 |
| 7 | $9^{\circ}$ | $252^{\circ}$ | 0.23 | 0.717 |
| 7 | $9^{\circ}$ | $288^{\circ}$ | -0.37 | 0.691 |
| 7 | $9^{\circ}$ | $324^{\circ}$ | -0.97 | 0.612 |

### 7.1.6 Analysis

The qualitative similarity between Figures 7.5 and 7.6 suggests that the non-dimensonalization for the thrust used in the free-swimming case is a reasonable one. It is evident from Figures 7.5 and 7.6 that for the parameter range considered in this study, the thrust coefficient is nearly maximum when the phase difference between the effective angle of attack and the position of the leading stagnation point of the trailing foil is close to zero.

For all cases considered, the thrust coefficient when $\phi_{D}=0$ is within a few percent of the maximum thrust coefficient. Although the exact operating condition for maximum thrust may be a function of the gait parameters, we will use the zero phase difference condition as a first approximation. Further, while this result is based on the average thrust coefficient at steady state, we use it to develop a controller to control the motion during the entire gait, including the initial transient period.

Based on this study, we will develop a controller which seeks to phase-synchronize the effective angle of attack with the measured stagnation point position on the leading edge of the trailing foil. We expect that such a controller would result in


Figure 7.5: Fixed velocity: Thrust coefficient versus the phase difference between $\alpha_{\text {eff }}$ and $x_{\text {stag }}$ at steady state for a series of experiments. The swimmer is constrained to prevent self-propulsion.
near-optimal thrust-producing gaits.

### 7.2 Literature review: Modeling and Control of

## Fluid Systems

There have been significant recent efforts to develop models of fluid systems suitable for the purposes of control. From the time history of a few pressure sensors, Suzuki et al. [107] developed an inverse imaging technique to identify the position and circulation of a vortex in a channel. For more complex systems, inverse algo-


Figure 7.6: Free-swimming: Thrust coefficient versus the phase difference between $\alpha_{\text {eff }}$ and $x_{\text {stag }}$ at steady state for a series of experiments where the swimmer is freeswimming in the $x$ direction
Note that a variation of the thrust coefficient defined by Equation (7.4) is used here.
rithms can quickly become difficult to implement. One common alternate approach involves the use of observers. Tadmor [108] used an extended Kalman filter to estimate the distance between two same-sign vortices as well as the position of their vorticity centroid and regulated these quantities through observer-based feedback. To suppress vortex shedding in the wake of a cylinder, Gerhard et al. [38] employed low-dimensional Galerkin models to develop an observer-based controller. And Pastoor et al. [92] used two first-order low pass filters as an observer to estimate the
phase and frequency of a nearly-periodic vortex wake behind a backward-facing step. A control method which we adopt in our own study is the phase-locked loop controller, which was successfully implemented by Joe et al. [50] to improve the lift performance of a flat plate at a fixed moderate to high angle of attack.

By contrast, our system does not require explicit knowledge or estimation of the fluid structures in the flow. The only information needed for the purposes of control is the position of the leading stagnation point and the effective angle of attack of the trailing foil.

### 7.3 Control Methodology

The control objective is based on the idea of exploiting the near-periodicity of the wake to optimize the swimmer's performance.

Based on the numerical experiments described in §7.1.4-7.1.6, a reasonable strategy to maximize thrust is to ensure that $x_{\text {stag }}$ and $\alpha_{\text {eff }}$ are in phase. Therefore, the control objective we pursue is to achieve $\phi_{D}=0$ during the motion of the swimmer. We implement this using a phase-locked loop controller.

Figure 7.7 is a block diagram of a classical phase-locked loop controller. Note that $\omega^{\star}$ is an initial value for the control variable $u$ which we usually set equal to $\omega_{1}$, the constant frequency of the leading foil. The signals $z_{i}$ and $z_{o}$ have the form $z_{i}(t)=A \sin \left(\omega_{i} t+\varphi_{i}\right)$ and $z_{o}(t)=b \cos \left(\omega_{o} t+\varphi_{o}\right)$. We note that in our application, the variables $z_{i}$ and $z_{o}$ correspond to the observed leading edge stagnation point position and approximate effective angle of attack of the trailing foil, respectively. In practice, the controller will modify the frequency of the trailing foil, $u$ which will produce a pitching and heaving motion as follows:

$$
\begin{equation*}
\beta_{2}(t)=A \cos \left(\int_{0}^{t} u(\tau) \mathrm{d} \tau+\varphi^{*}\right) \tag{7.9}
\end{equation*}
$$



Figure 7.7: Classical phase-locked loop block diagram.

$$
\begin{equation*}
y_{2}(t)=h \sin \left(\int_{0}^{t} u(\tau) \mathrm{d} \tau+\varphi^{*}\right) \tag{7.10}
\end{equation*}
$$

where $\varphi^{*}$ is an initial prescribed phase offset. Another difference with the classical phase-locked loop controller is that rather than producing an output $z_{o}(t)=$ $b \cos \left(\omega_{o} t+\varphi\right), z_{o}$ will be the effective angle of attack as determined by Equation (7.3). Multiplying the signals one gets:

$$
\begin{equation*}
z_{i} z_{o}=\frac{A b}{2}\left\{\sin \left(\left(\omega_{i}+\omega_{o}\right) t+\varphi_{i}+\varphi_{o}\right)+\sin \left(\left(\omega_{i}-\omega_{o}\right) t+\varphi_{i}-\varphi_{o}\right)\right\} \tag{7.11}
\end{equation*}
$$

In the ideal case when the signals have close to equal frequencies, the resulting signal will have a high frequency component of frequency $\omega_{i}+\omega_{o}$ plus a low frequency component of frequency $\omega_{i}-\omega_{o}$. An appropriately designed low-pass filter will eliminate the high-frequency component. Then, letting $\phi_{i}=\omega_{i} t+\varphi_{i}$ and $\phi_{o}=\omega_{o} t+\varphi_{o}$, we have:

$$
\begin{equation*}
z_{i} z_{o} \approx \frac{A b}{2} \sin \left(\phi_{i}-\phi_{o}\right) \tag{7.12}
\end{equation*}
$$

Finally, assuming the two phases are sufficiently close, we can use the small angle approximation to further simplify Equation (7.12):

$$
\begin{equation*}
z_{i} z_{o} \approx \frac{A b}{2}\left(\phi_{i}-\phi_{o}\right) . \tag{7.13}
\end{equation*}
$$

Under these assumptions, the block diagram can be equivalently represented as in


Figure 7.8: Simplified phase-locked loop block diagram.

Figure 7.8. The error is $e=\phi_{i}-\phi_{o}$, and the closed-loop transfer function is

$$
\begin{equation*}
\frac{\phi_{o}}{\phi_{i}}=\frac{\tilde{K} s+\tilde{K}_{i} K}{s^{2}+\tilde{K} s+K_{i} \tilde{K}} \tag{7.14}
\end{equation*}
$$

where $\tilde{K}=\frac{K A b}{2}$ accounts for the magnitudes of the signals $z_{i}$ and $z_{o}$. The error response is then

$$
\begin{equation*}
\frac{e}{\phi_{i}}=1-\frac{\phi_{o}}{\phi_{i}}=\frac{s^{2}}{s^{2}+\tilde{K} s+K_{i} \tilde{K}} \tag{7.15}
\end{equation*}
$$

This is in standard second-order form where $\omega_{n}=\sqrt{K_{i} \tilde{K}}$ and $\zeta=\sqrt{\frac{\tilde{K}}{4 K_{i}}}$. Then, the proportional-integral (PI) controller gains are given by:

$$
\begin{align*}
K & =\frac{4 \zeta \omega_{n}}{A b}  \tag{7.16}\\
K_{i} & =\frac{\omega_{n}}{2 \zeta} \tag{7.17}
\end{align*}
$$

As observed previously [21], tuning such a controller requires knowing the amplitudes of the signals in advance. We determined estimates for these values by performing trial numerical experiments in advance. For our system, we chose the damping ratio of $\zeta=1$ and tuned the gains by using intuition to manually adjust $\omega_{n}$ (held constant for each trial) during a series of numerical experiments to produce the desired response characteristics.

Recall that in practice $z_{i}$ and $z_{o}$ correspond to the observed stagnation point position and effective angle of attack, respectively. Unlike in the classical phase-locked
loop, our feedback variable ( $\alpha_{\text {eff }}$ ) has a more complex expression and the classical phase-locked loop construction does not strictly apply. Here, the frequency $u$ varies such that the corresponding pitch and heave values of the trailing foil are given by Equations (7.9) and (7.10). As shown in Equation (7.3), the effective angle of attack is a function of the pitch angle, the heaving velocity, the pitching rate and the speed of the swimmer relative to the freestream flow. Further, since the classical phase-locked loop controller synchronizes signals 90 degrees out of phase from each other, we feed back a "fictional" effective angle of attack that is phase-shifted by 90 degrees from the true effective angle of attack such that the stagnation point signal is synchronized with the true effective angle of attack.

Despite these differences between our system and the classical phased-locked loop configuration, we found that in practice the appropriately-tuned controller worked successfully and the tuning is similar to that of a second-order system. We caution that this fortunate result may not apply outside the limited parameter space explored here.

### 7.4 Example

In this section we present an example where a phase-locked loop controller is used to synchronize the effective angle of attack of the trailing foil with the observed position of its leading edge stagnation point. We demonstrate that the controller generates a gait which produces a near-optimal thrust coefficient at steady state.

A discrete-time $3^{\text {rd }}$-order low-pass Butterworth filter (see Figure 7.7) with cutoff frequency 0.023 was used for this example. The leading and trailing bodies have NACA 0012 profiles and are discretized into 100 straight line panels each. The leading edge of the trailing foil is placed 2.5 chord lengths behind the leading edge of the leading foil, leaving 1.5 chord lengths of space in between. Both foils pitch about
their quarter chord points. The pitching angle, $\beta_{1}(t)$, and heaving amplitude, $y_{1}(t)$, of the leading foil are prescribed as follows:

$$
\begin{gather*}
\beta_{1}(t)=\frac{9 \pi}{180} \sin (7 t)  \tag{7.18}\\
y_{1}(t)=0.05 \sin \left(7 t-\frac{\pi}{2}\right) . \tag{7.19}
\end{gather*}
$$

Refer to Figure 7.1 for a schematic of the swimmer geometry. To avoid numerical difficulties due to the singular nature of point vortices, wake vortices that approach within 0.03 chord lengths of the surface of the trailing foil are diverted around the body. For simplicity of analysis, the swimmer is constrained to propel only along the direction of the freestream. The swimmer was constrained to prevent overall lateral or angular motions resulting from fluid forces. Since the numerical model is inviscid, a simple drag model was adopted in order to ensure that the swimmer reaches a steady swimming speed. Viscous drag is modeled in only the freestream direction as proportional to the square of the relative velocity of the swimmer:

$$
\begin{equation*}
\operatorname{Drag}=\frac{1}{2} \rho C_{f} A_{s} V_{\mathrm{rel}}^{2}, \tag{7.20}
\end{equation*}
$$

where $C_{f}$ is a drag coefficient and $A_{s}$ is the surface area of the swimmer. In our model, the two foils represent the propulsors (or fins) on a larger swimmer with a body further upstream of the foils. Although the combined length of the two foils is 2 chord lengths; for simulation purposes, we take the total length of the body to be about 9 chord lengths for a total surface area of 18c.

The total body length to caudal fin length ratio varies greatly among swimmers. For two different morphologies of Zebrafish Danio rerio [94], the ratio is 2.99 and 5.34, while for the false killer whale Pseudorca crassidens [132] it is 8.08. Our chosen ratio of 4.5 falls comfortably within this range.

We assume $C_{f}=0.0146$, the experimentally-determined friction drag coefficient for a swimming dogfish Mustelus canis in a flume [4]. Using a different friction coefficient value will affect the steady state swimming speed.

The motion of the trailing foil is as follows:

$$
\begin{align*}
& \beta_{2}(t)=\frac{9 \pi}{180} \cos \left(\int_{0}^{t} u(\tau) \mathrm{d} \tau+\varphi^{*}\right)  \tag{7.21}\\
& y_{2}(t)=0.05 \sin \left(\int_{0}^{t} u(\tau) \mathrm{d} \tau+\varphi^{*}\right) \tag{7.22}
\end{align*}
$$

where $\varphi^{*}$ is an initial prescribed phase offset and $u(t)$ is determined by the controller. From the block diagram for the phase-locked loop, we see that $u=\omega^{\star}+K\left(1+\frac{K_{i}}{s}\right) e$. To avoid a long start-up time and allow $u$ to more quickly reach a constant steadystate value, $\omega^{\star}$ is set equal to $\omega_{1}$.

### 7.4.1 Results

We prescribe the gait in Equations (7.18), (7.19), (7.21) and (7.22) and use the phase-locked loop controller described in Section 7.3 to synchronize the motion of the trailing foil to the oncoming wake. Figure 7.9 shows that the normalized effective angle of attack and the position of the stagnation point versus time are effectively synchronized by the controller after a transient period. Time is non-dimensionalized by the period, $t_{\omega}=2 \pi / \omega_{1}$, and speed is non-dimensionalized by the chord length, $c$ and the period. The speed of the swimmer relative to the freestream versus time is shown in Figure 7.10. The average non-dimensional steady-state speed is 1.296, and the Strouhal number based on tip-to-tip excursion and speed relative to the freestream is 0.20 .

A series of several numerical experiments without control plus one with control on a self-propelling swimmer were performed to check how well the controller achieves its goal of generating the fastest gait. Most of the experiments performed were similar


Figure 7.9: With control (leading foil frequency, $\omega_{1}=7$ ): Normalized effective angle of attack, $\alpha_{\text {eff }}$, and the stagnation point position along the surface, $x_{\text {stag }}$, of the trailing foil vs non-dimensionalized time for a self-propelling fish-like swimmer using a phaselocked loop controller. Both quantities are scaled to facilitate visual comparison. After a transient period, the two signals are synchronized. The gait is given by Equations (7.18), (7.19), (7.21) and (7.22).
to those described previously and served as a "brute force" method for mapping the swimming speed at various operating conditions. The motion of the leading foil is again given by Equations (7.18) and (7.19), while in the experiments without control, the trailing foil is prescribed the gait:

$$
\begin{gather*}
\beta_{2}(t)=\frac{9 \pi}{180} \sin \left(7 t+\phi_{2}\right)  \tag{7.23}\\
y_{2}(t)=0.05 \sin \left(7 t-\frac{\pi}{2}+\phi_{2}\right), \tag{7.24}
\end{gather*}
$$

where the motion is the same as the leading foil but with a constant phase difference, $\phi_{2}$, between the leading and trailing foils. A total of 20 numerical experiments with


Figure 7.10: Non-dimensional swimmer speed relative to the freestream versus time. After a transient period, the swimmer reaches a steady-state non-dimensional average speed of 1.296. The swimmer gait is prescribed by Equations (7.18), (7.19), (7.21) and (7.22) and a phase-locked loop controller described in Section 7.3 is employed to synchronize the motion of the trailing foil to the oncoming wake.
evenly-spaced phase differences between 0 and $2 \pi$ were performed. The steady state speed as a function of the phase difference for these runs are indicated by blue dots in Figure 7.11. One additional experiment was performed where the motion of the trailing foil is regulated by the controller. On the same figure, a red plus sign indicates the steady-state speed achieved by the controller, which appears to be at or near the fastest achievable speed for this family of gaits. This speed was achieved by the controller at an effective constant phase difference between the leading and trailing foils of $\phi_{2}=0.635$ at steady state. This suggests that the controller is effective at achieving near-optimal speed gaits. To demonstrate that the controller is effective in a broader parameter space, another experiment was performed with nearly the same gait as prescribed by Equations (7.18), (7.19), (7.21) and (7.22) except the absolute frequency of the leading foil was changed from $\omega_{1}=7$ to 4 . A non-dimensional steady-


Figure 7.11: Non-dimensional steady-state speed vs. $\phi_{2}$ for constant phase gaits. The motion of the leading foil is given by Equations (7.18) and (7.19) while the trailing foil motion is prescribed by Equations (7.23) and (7.24), where $\phi_{2}$ is the phase difference between the leading and trailing foils. The red plus sign indicates the steady-state speed for value $\phi_{2}=0.635$, the equivalent phase difference at steady-state for the controlled case. Among all the experiments performed, the maximum steady-state speed is achieved when $\phi_{2}=0.635$, suggesting that the control algorithm is effective at maximizing speed.
state speed of 1.37 was achieved. Figure 7.12 shows how the controller synchronizes the effective angle of attack and the leading stagnation point position of the trailing foil (both normalized to unity at steady state).

### 7.4.2 Lock-in Time

Although we showed that the phase-locked loop controller works effectively to reach an optimum steady-state speed, there is no guarantee that it will do so quickly, or even that it will do so more quickly than simply prescribing the optimal constantphase gait with no controller. This difficulty partially arises from the fact that the controller is based on the linearized model of a phase-locked loop. Whereas the model


Figure 7.12: With control (leading foil frequency, $\omega_{1}=4$ ): Normalized effective angle of attack, $\alpha_{\text {eff }}$, and the stagnation point position along the surface, $x_{\text {stag }}$, of the trailing foil vs non-dimensionalized time for a self-propelling fish-like swimmer using a phase-locked loop controller. After a transient period, the two signals are synchronized. The gait is given by Equations (7.18), (7.19), (7.21) and (7.22) but with the absolute frequency of the leading foil changed to $\omega_{1}=4$ in order to demonstrate the effectiveness of the controller across a range of pitching/heaving frequency.
assumes that the error is equal to the difference in the phases between the signals, in reality the error is the sine of the difference of the phases. This means, for example, that when the two signals are $\pi$ radians out of phase, the "error" will be zero and the controller will not adjust the frequency as desired. This can substantially increase the lock-in time needed to synchronize the two signals.

To demonstrate this effect, the same controller was initialized with two different initial phase values, $\varphi^{*}$ (See Equations (7.21) and (7.22)). Figure 7.13 shows the relative swimmer speed versus time for these two cases as well as for the optimal constant-phase case with no control. In all three cases, the swimmer achieves the same steady-state speed. When $\varphi^{*}=5.293$, the controller is able to reach a steady-


Figure 7.13: Non-dimensional speed vs time for controlled and uncontrolled gaits. All gaits reach the same steady-state speed, though the choice of initial phase $\varphi^{*}$ significantly impacts performance in the controlled case.
state speed slightly faster than the constant-phase case with no control. On the other hand, when $\varphi^{*}=1.722$, the controller took considerably longer to reach steady-state. It is not apparent how one should choose $\varphi^{*}$ to optimize performance. It is also possible that without proper tuning, a poor initial phase choice could result in the controller being unable to converge.

### 7.5 Swimming with and without control

In this section we demonstrate by example the performance gain achieved from this control algorithm. Consider a case where the frequency of the leading foil is set to a constant value. From the results of a parametric study, such as in Figure 7.11, the phase of the trailing foil can be set to a fixed value to optimize the steady-state speed. Now, consider that the frequency of the leading foil deviates slightly from the


Figure 7.14: Leading foil frequency, $\omega_{1}(t)$ vs non-dimensionalized time. The frequency of the leading foil is varied from its initial value in order to evaluate the effectiveness of applying control to the trailing foil.
assumed value. We compare the resulting performance between two cases: (a) where the trailing foil has no ability to detect the change in frequency and maintains a constant phase difference and (b) where the trailing foil is able to detect the position of its leading stagnation point and a controller is implemented to modify its motion.

The frequency versus time of the leading foil in this example is shown in Figure 7.14. The trailing foil in case (a) maintains a constant absolute frequency of $\omega_{2}=7$. Two sets of parameters are used in the controlled case. Initially, the same gains are used as in Section 7.4.1. Less aggressive gains were needed to ensure that the controller did not go unstable. After the lock-in time (around $t / t_{\omega_{1}}=8$ ), the controller switches to a more aggressive set of parameters in order to more effectively track variations in the frequency of the flow. Without more aggressive control gains, the controller is unable to track the variations in the frequency of the flow.

Figure 7.15 shows the resulting non-dimensional forward speed versus time of the swimmer for the cases with and without control. A snapshot of the swimmer and its vortex wake for the controlled case is shown in Figure 7.17. The controlled case


Figure 7.15: Swimmer performance with and without control. Comparison of relative speed vs non-dimensionalized time for a swimmer with (blue) and without (red) control. The gait of the swimmer is given by Equations (7.18), (7.19), (7.21) and (7.22), except instead of a constant frequency motion for the leading foil, the prescribed frequency versus time is as shown in Figure 7.14. A snapshot of the swimmer and flow field for the controlled case is shown in Figure 7.17. In the uncontrolled case, the frequency of the trailing foil remains constant.
was able to achieve a higher speed at all times, even when the trailing foil frequency was lower than in the uncontrolled case. At steady state the controlled case reaches a constant average speed. On the other hand, the uncontrolled case reaches a limit cycle due to the fact that the frequency of the leading foil and vortex wake is different from that of the trailing foil.

Figures 7.16(a) and 7.16(b) show the normalized effective angle of attack and location of the leading stagnation point position versus time for the cases with and without control, respectively. As expected, in Figure 7.16(a), the two curves remain nearly in phase after the initial lock-in period. By contrast, the signals in Figure 7.16(b) go in and out of phase due to the lack of compensation for the frequency difference between the leading and trailing foils. Figures 7.19(a) and 7.19(b), which are plots of the stagnation point position versus the effective angle of attack, illustrate this result


Figure 7.16: Normalized effective angle of attack and stagnation point position along the surface vs time for the cases (a) with control and (b) without control corresponding to the results in Figure 7.15. Since the frequency of the leading and trailing foils are different in the uncontrolled case, the stagnation point position and effective angle of attack are not usually in phase. By comparing Figures 7.16(b) and 7.15, it is apparent that the higher speed portions of the limit cycle of the red curve in Figure 7.15 correspond to when the two curves in Figure 7.16(b) are in phase, while the slower speeds correspond to when the curves are out of phase. Refer to Figure 7.18 for more details.


Figure 7.17: Snapshot of swimmer and vortex wake. A portion of the wake of the swimmer is shown for the controlled case described in $\S 7.5$ at $t / t_{\omega}=44.5$. The red and blue circles represent clockwise and counterclockwise vorticity, respectively. The larger, darker circles correspond to stronger strength vortices while the smaller, lighter circles are weaker strength vortices.
more clearly. By comparing Figures $7.16(\mathrm{~b})$ and 7.15 , it is apparent that the higher speed portions of the limit cycle of the red curve in Figure 7.15 correspond to when the two curves in Figure 7.16(b) are in phase, while the slower speeds correspond to when the curves are out of phase. In Figure 7.18 we plot the speed of the swimmer in the uncontrolled case as well as a measure of the phase between $\alpha_{\text {eff }}$ and $x_{\text {int }}$. Higher swimming speeds correspond to when the two quantities are in phase while lower speeds occur when they are out of phase. See Figure 7.18 for more details.

The use of a controller has noticeable benefits in addition to generating faster gaits. At steady state, although the controlled case resulted in a higher trailing foil frequency than in the case without control, the power required in the uncontrolled case was still about $7 \%$ more than in the case with control. Overall, for the entire gait in this example, the controller improves the average propulsive efficiency from $28.6 \%$ in the uncontrolled case to $40.8 \%$ in the controlled case.

### 7.6 Summary

We have considered the problem of controlling the motion of a swimmer through a self-generated, nearly-periodic wake. It is apparent that the timing of the motion


Figure 7.18: Relative speed of swimmer for uncontrolled case from Figure 7.15 (red, solid) versus non-dimensionalized time along with the low-pass filtered product of $x_{\text {int }}$ and a $\pi / 2$ phase-shifted version of $\alpha_{\text {eff }}$ in Figure 7.16(b) (black, dashed). The low-pass filtered product of these terms acts as a phase detector and provides a measure of the phase difference between the two curves. The value is zero both when the curves are in phase and when they are $\pi$ radians out of phase. The maximum value occurs for a phase difference of $\pi / 2$ and $-\pi / 2$. The in-phase zero points are labeled by closed blue circles and the out of phase points by open red circles. Note that the swimmer speed is nearly maximum when the curves are in phase and nearly minimum when the curves are out of phase.
of the swimmer through such a wake will affect performance. The effective angle of attack plays an important role in determining the performance of the swimmer, however determining the effective angle of attack for a swimmer moving through a non-uniform wake is not trivial. For the range of parameters considered in this study, it was shown that the position of the leading stagnation point on the trailing foil serves as an effective proxy for determining the phase and frequency of the effective angle of attack of a nearly periodically oscillating foil. Based on a series of numerical experiments, the highest thrust coefficient gaits were achieved when the effective angle of attack of the swimmer (due to only the motion of the foil while ignoring the influence of the wake) was in or nearly in phase with the "true" effective angle of attack as indicated by the stagnation point position.


Figure 7.19: Normalized stagnation point position along the surface versus effective angle of attack for the cases (a) with control and (b) without control. The last three periods of motion are indicated by the black portion of the curves. These figures were produced from the same data in Figure 7.16 to more clearly visualize how the controller is able to effectively synchronize the two variables after an initial transient period. Perfect synchronicity would correspond to the curves lying directly along the diagonal dotted line.

A phase-locked loop controller was implemented to attain a nearly optimal steadystate thrust-producing swimming gait by synchronizing the two signals. It was shown that after a transient "lock-in" period, the controller successfully synchronized the two signals. To verify the performance of the controller, a parametric study was conducted on a family of constant-phase gaits in which only the phase between the leading and lagging foil was varied between runs. The fastest gaits indeed corresponded to the same phase difference achieved by the controller at steady state. In other words, after a transient period, the controller was able to find a near-optimal gait in terms of swimming speed.

To demonstrate the usefulness of such a controller, an example was presented in which the frequency of the leading foil deviated slightly from the presumed frequency. Two simulations were run: one with a controller to adapt to the changing frequency of the vortex wake, and another without control that assumed a known constant
leading foil frequency. It was shown that for this example, the controller led to a higher speed, decreased power use and higher propulsive efficiency.

Due to the assumptions made in the controller design, performance is limited. The transient time before the controller converges and is able to synchronize the signals may depend significantly on the initial phase of the system. A non-linear controller may be needed to further improve performance.

Additionally, the numerical experiments performed only explored a limited range of the parameter space. In particular, only relatively low-amplitude motions were considered. Additionally, some parameters such as the heaving amplitude were fixed to a constant value. Our conclusions which focused on the effective angle of attack may not extend to swimming outside of the parameter range considered here. In particular, we note that Beal et al. [11] suggested there were two main mechanisms for a moving foil to improve performance: the effective angle of attack and the suction created by the interaction of vortices with the leading edge. This later effect was not considered in this analysis and may play an important role in larger-amplitude swimming gaits.

## Chapter 8

## Conclusions and Future Work

In this chapter we highlight the contributions of this thesis and offer suggestions for future study.

In Chapters 3 and 4 we presented models for fish-like swimmers in potential and Stokes flow and used them to design forward and turning gaits. Our models accurately account for all hydrodynamical effects even for large-amplitude body deformations. This improves upon previous work, which assumed either small-amplitude deformations or hydrodynamically decoupled swimmer links. In both the potential flow and Stokes flow cases the local form of the connection - which relates shape changes to the body velocity - appears as an important term in the equations of motion. We demonstrated in Chapter 5 that the curvature of the connection plays a critical role in the gait design process by highlighting the areas of shape space corresponding to useful gaits. However, the large-amplitude gait design method we introduced is limited to motion in which the group characterizing the position and orientation of the swimmer is Abelian or in special cases when a semidirect product group is composed of an Abelian group and a vector space. In the latter case, our result only applies for the Abelian component of the semidirect product group but not the vector space directions. One particularly challenging open area for future study is identify-
ing approaches useful for developing large-amplitude gaits for motion corresponding to non-Abelian groups.

To model more realistic swimming that partially accounts for the effects of viscosity, we reviewed a numerical model of a potential flow swimmer with vorticity shedding in Chapter 6. Unlike most prior work, our model computed the overall self-propelling motion of the swimmer as part of the solution. We also extended these models by describing a method for computing the velocity potential in the presence of a vortex wake as well as for diverting vortices around a body to avoid some numerical issues. To facilitate the analysis, the swimmer was constrained to propel in only the freestream direction. Future work should permit full motion in the plane and attempt to develop techniques to achieve optimal turning gaits as well. In Chapter 7 we developed a control technique for maximizing thrust during forward swimming. Through a series of numerical experiments, we demonstrated that an optimal control strategy should ensure that two quantities on the trailing foil - the position of the leading stagnation point $\left(x_{\text {stag }}\right)$ and the approximate effective angle of attack, which ignores the effect of the vortex wake $\left(\alpha_{\text {eff }}\right)$ - should be in phase. We note that we do not expect these results to necessarily hold outside of the narrow parameter range considered in our study. In fact, Figures 7.5 and 7.6 suggest that the steady-state phase difference between $x_{\text {stag }}$ and $\alpha_{\text {eff }}$ may be a function of pitching amplitude, frequency or perhaps other parameters. In our work we employed a control technique based on steady state results even during the transient portion of the gait. In fact an optimal control technique may depend on a parameter such as the Strouhal number which accounts for the swimmer speed relative to the freestream. Other authors [11] have observed that performance in foil-like propulsion through a vortex wake is dependent on the effective angle of attack and the suction created by vortices interacting with the foil. This latter effect may be dominant for larger amplitude motions, and the ability to identify vortex structures upstream of the foil may prove necessary for
effective control algorithms.
As an alternative to the heuristic approach considered in this work, one might pursue the use of numerical optimal control methods [83] to navigate along a desired path or around obstacles. Since the numerical model adopted here increases the state space dimension of the system at each time step due to the creation of vortices, we anticipate that one challenge to using such tools will be the requirement that the state dimension remain constant. Although a full Navier-Stokes simulation could address this particular issue, the computational cost associated with such a large state may be prohibitively high. A more fruitful approach might involve the adaption of a relatively low-order model like the one presented in this thesis or reduced order models, perhaps based on proper orthogonal decomposition techniques.

Ultimately, these results should be tested and applied experimentally. We anticipate that the turning gaits identified in the potential flow case may serve as a reasonable first approximation for relatively large, fast-moving robotic swimmers. In the field of nanomedicine, the design of drug delivery or cell repair machines might benefit from an improved understanding of motion in Stokes flow. Finally, based on the control techniques we have introduced for swimming through a vortex wake, we envision fish-like robots - with a series of surface sensors mimicking the lateral line in their biological counterparts - that are aware of their surrounding fluid environments and navigate through them efficiently.

## Bibliography

[1] R. Abraham, J. Marsden, and T. Ratiu. Manifolds, Tensor Analysis, and Applications. Springer-Verlag, 2nd edition, 1991.
[2] I. Akhtar, R. Mittal, G. Lauder, and E. Drucker. Hydrodynamics of a biologically inspired tandem flapping foil configuration. Theoretical and Computational Fluid Dynamics, 21(3):155-170, 2007.
[3] M. Allen. Bush's gaffes are back as debates near. The Washington Post, page A8, October 12000.
[4] E. Anderson, W. McGillis, and M. Grosenbaugh. The boundary layer of swimming fish. J. Exp. Biol., 204(1):81-102, 2001.
[5] J. Anderson and N. Chhabra. Maneuvering and Stability Performance of a Robotic Tuna 1. Integrative and Comparative Biology, 42(1):118-126, 2002.
[6] J. Anderson and P. Kerrebrock. The vorticity control unmanned undersea vehicle (VCUUV) —An autonomous vehicle employing fish swimming propulsion and maneuvering. Proc. 10th Int. Symp. Unmanned Untethered Submersible Technology, pages 189-195, 1997.
[7] J. Anderson and P. Kerrebrock. The vorticity control unmanned undersea vehicle (VCUUV) performance results. Eleventh International Symposium on Unmanned Untethered Submersible Technology, 369, 1999.
[8] D. Barrett, M. Grosenbaugh, and M. Triantafyllou. The optimal control of a flexible hull robotic undersea vehicle propelled by an oscillating foil. Proceedings of the 1996 Symposium on Autonomous Underwater Vehicle Technology, pages 1-9, 1996.
[9] D. Barrett, M. Triantafyllou, D. Yue, M. Grosenbaugh, and M. Wolfgang. Drag reduction in fish-like locomotion. Journal of Fluid Mechanics, 392:183-212, 2000.
[10] B. Basu and G. Hancock. The unsteady motion of a two-dimensional aerofoil in incompressible inviscid flow. J. Fluid Mech., 87(1):159-178, 1978.
[11] D. Beal, F. Hover, M. Triantafyllou, J. Liao, and G. Lauder. Passive propulsion in vortex wakes. Journal of Fluid Mechanics, 549:385-402, 2006.
[12] L. Becker, S. Koehler, and H. Stone. On self-propulsion of micro-machines at low Reynolds number: Purcell's three-link swimmer. Journal of Fluid Mechanics, 490:15-35, 2003.
[13] T. Benjamin and A. Ellis. The Collapse of Cavitation Bubbles and the Pressures thereby Produced against Solid Boundaries. Philosophical Transactions for the Royal Society of London. Series A, Mathematical and Physical Sciences,, 260(1110):221-240, 1966.
[14] W. Birnbaum. Der Schlagflügelpropeller und die kleinen Schwingungen elastisch befestigter Tragflügel. Z. f. Flugtechn. u. Motorluftschiffahrt, 15:128-134, 1924.
[15] J. Blake. Self propulsion due to oscillations on the surface of a cylinder at low Reynolds number. Bull. Austral. Math. Soc, 3:255-264, 1971.
[16] R. Blake, L. Chatters, and P. Domenici. Turning radius of yellowfin tuna(Thunnus albacares) in unsteady swimming manoeuvres. Journal of fish biology, 46(3):536-538, 1995.
[17] T. Brower. Design of a Manta Ray Inspired Underwater Propulsive Mechanism For Long Range, Low Power Operation. PhD thesis, Tufts University, 2006.
[18] J. H. J. Buchholz. The Flowfield and Performance of a Low Aspect Ration Unsteady Propulsor. PhD thesis, Princeton University, Princeton, NJ, September 2006.
[19] J. Carling, T. L. Williams, and G. Bowtell. Self-propelled anguilliform swimming: simultaneous solution of the two-dimensional navier-stokes equations and Newton's laws of motion. J. Exp. Biol., 201(23):3143-3166, 1998.
[20] J. Cheng and G. Chahine. Computational hydrodynamics of animal swimming: boundary element method and three-dimensional vortex wake structure. Comparative Biochemistry and Physiology, Part A, 131(1):51-60, 2001.
[21] D. Clarke. Designing phase-locked loops for instrumentation applications. Measurement, 32(3):205-227, 2002.
[22] J. Cortes, S. Martinez, J. Ostrowski, and K. McIsaac. Optimal Gaits for Dynamic Robotic Locomotion. The International Journal of Robotics Research, 20(9):707, 2001.
[23] D. Crighton. The Kutta Condition in Unsteady Flow. Annual Review of Fluid Mechanics, 17(1):411-445, 1985.
[24] G. de Araujo and J. Koiller. Self-Propulsion of N-Hinged 'Animats' at Low Reynolds Number. Qualitative Theory of Dynamical Systems, 1:1-28, 2003.
[25] R. Djojodihardjo and S. Widnall. A numerical method for the calculation of nonlinear, unsteady lifting potential flow problems. AIAA Journal, 7(10):20012009, 1969.
[26] E. G. Drucker and G. V. Lauder. Locomotor function of the dorsal fin in teleost fishes: experimental analysis of wake forces in sunfish. J. Exp. Biol., 204:29432958, 2001.
[27] J. Eldredge. Numerical simulations of undulatory swimming at moderate Reynolds number. Bioinspiration and Biomimetics,, 1(4):S19-S24, 2006.
[28] J. Eldredge, T. Colonius, and A. Leonard. A vortex particle method for twodimensional compressible flow. Journal of Computational Physics, 179(2):371399, 2002.
[29] S. Feldkamp. Swimming in the California sea lion: morphometrics, drag and energetics, 1987.
[30] R. FINN. Stationary Solutions of the Navier-Stokes Equations. Proceedings of Symposia in Applied Mathematics, 35, 1965.
[31] F. Fish. Kinematics and estimated thrust production of swimming harp and ringed seals, 1988.
[32] F. Fish. Power Output and Propulsive Efficiency of Swimming Bottlenose Dolphins (Tursiops Truncatus), 1993.
[33] F. Fish. Performance constraints on the maneuverability of flexible and rigid biological systems. Proceedings of the Eleventh International Symposium on Unmanned Untethered Submersible Technology, pages 394-406, 1999.
[34] F. Fish, J. Hurley, and D. Costa. Maneuverability by the sea lion Zalophus californianus: turning performance of an unstable body design. J. Exp. Biol., 206(4):667-674, 2003.
[35] S. F. Galls and O. K. Rediniotis. Computational simulation of the autonomous navigation of a biomimetic underwater vehicle. AIAA Journal, 41(4):605-611, April 2003.
[36] A. Galper and T. Miloh. Generalized Kirchhoff Equations for a Deformable Body Moving in a Weakly Non-Uniform Flow Field. In Mathematical and Physical Sciences, volume 446, pages 169-193. JSTOR, 1994.
[37] A. Galper and T. Miloh. Dynamic equations of motion for a rigid or deformable body in an arbitrary non-uniform potential flow field. J. Fluid Mech., 295:91120, 1995.
[38] J. Gerhard, M. Pastoor, R. King, B. Noack, A. Dillmann, M. Morzynski, and G. Tadmor. Model-based control of vortex shedding using low-dimensional Galerkin models. AIAA Paper, 4262, 2003.
[39] J. Giesing. Unsteady Two-Dimensional Potential Flow with Lift. Douglas Aircraft Report No. CB-32144, March, 1965.
[40] J. Giesing. Unsteady Two-Dimensional Potential Flow For Two Bodies With Lift. Technical Report AD0654200, DTIC Research Report, 1967.
[41] J. Giesing. Nonlinear Interaction of Two Lifting Bodies in Arbitrary Unsteady Motion. Journal of Basic Engineering, 90(3):387-394, 1968.
[42] J. Giesing. Nonlinear Two-Dimensional Unsteady Potential Flow with Lift. Journal of Aircraft, 5(2):135-143, 1968.
[43] H. Glauert. The force and moment on an oscillating airfoil. Rep. Memo. Aeronaut. Res. Comm.(Great Britain, 1929.
[44] J. Graver, P. University, D. of Mechanical, and A. Engineering. Underwater Gliders: Dynamics, Control and Design. PhD thesis, Princeton University, 2005.
[45] J. Gray. Studies In Animal Locomotion VI. The propulsive powers of the dolphin. J. Exp. Biol., 13:192-199, 1936.
[46] M. Hausner and J. Schwartz. Lie groups, Lie algebras. Gordon and Breach, New York, 1968.
[47] J. Hess and A. Smith. Calculation of potential flow about arbitrary bodies. Prog. Aeronaut. Sci., 8:1-138, 1967.
[48] F. Houssay. Forme, puissance et stabilité des poissons. A. Hermann et fils, 1912.
[49] G. Hsiao and R. Kress. Integral equation for the two-dimensional exterior Stokes problem. Appl. Numer. Math., 1(1):77-93, 1985.
[50] W. T. Joe, K. Taira, T. Colonius, D. G. MacMynowski, and G. Tadmor. Closedloop control of vortex shedding on a two-dimensional flat-plate airfoil at a low reynolds number. In 46 th AIAA Aerospace Sciences Meeting and Exhibit, January 2008.
[51] K. Jones, C. Dohring, and M. Platzer. Wake Structures Behind Plunging Airfoils: A Comparison of Numerical and Experimental Results. AIAA Paper, 1996.
[52] K. Jones and M. Platzer. An experimental and numerical investigation of flapping-wing propulsion. AIAA Paper, pages 99-0995, 1999.
[53] K. Jones, M. Platzer, and C. Monterey. Time-Domain Aeroelastic Analysis of a Two Airfoil System with Application to Unsteady Rotary Wing Flowfields. In AIAA 33rd Aerospace Sciences Meeting and Exhibit. Reno, NV, 1995.
[54] E. Kanso and J. Marsden. Optimal Motion of an Articulated Body in a Perfect Fluid. Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on, pages 2511-2516, 2005.
[55] E. Kanso, J. E. Marsden, C. W. Rowley, and J. B. Melli-Huber. Locomotion of articulated bodies in a perfect fluid. J. Nonlin. Sci., 2005.
[56] T. Karman and W. Sears. Airfoil theory for non-uniform motion. Journal of the Aeronautical Sciences, 5:6-17, 1938.
[57] J. Katz and A. Plotkin. Low-Speed Aerodynamics. Cambridge University Press, 2nd edition, May 2001.
[58] S. Kelly and R. Murray. The geometry and control of dissipative systems. Decision and Control, 1996., Proceedings of the 35th IEEE, 1, 1996.
[59] S. D. Kelly. The Mechanics and Control of Robotic Locomotion with Applications to Aquatic Vehicles. PhD thesis, California Institute of Technology, Pasadena, California, May 1998.
[60] G. Kirchhoff. Ueber die bewegung eines rotationskörpers in einer flüssigkeit. Crelle, lxxi:237, 1869.
[61] R. Krasny. Desingularization of periodic vortex sheet roll-up. Journal of Computational Physics, 65(2):292-313, 1986.
[62] H. Kussner. Untersuchung der Bewegung einer Platte beim Eintritt in eine Strahlgrenze. Luftfahrt Forschung, 13:425, 1936.
[63] H. Kussner. Zusammenfassender Berichtuber den instationaren Auftrieb von Flugeln. Luftfahrtforschung, 13(12):410-424, 1936.
[64] H. Lamb. Hydrodynamics. Dover, 1945.
[65] N. Leonard and J. Graver. Model-based feedback control of autonomous underwater gliders. Oceanic Engineering, IEEE Journal of, 26(4):633-645, 2001.
[66] N. E. Leonard and P. S. Krishnaprasad. Motion control of drift-free, leftinvariant systems on lie groups. IEEE Trans. Automat. Contr., 40, 1995.
[67] A. Leroyer and M. Visonneau. Numerical methods for RANSE simulations of a self-propelled fish-like body. Journal of Fluids and Structures, 20(7):975-991, 2005.
[68] J. Liao, D. Beal, G. Lauder, and M. Triantafyllou. Fish Exploiting Vortices Decrease Muscle Activity. Science, 302(5650):1566-1569, November 2003.
[69] M. Lighthill. On the squirming motion of nearly-spherical deformable bodies through liquids at very small reynolds numbers. Communications on Pure and Applied Mathematics, 5:100-118, 1952.
[70] M. J. Lighthill. Note on the swimming of slender fish. J. Fluid Mech., 9, 1960.
[71] M. J. Lighthill. Large-amplitude elongated-body theory of fish locomotion. Proc. R. Soc. Lond. B, 179, 1971.
[72] H. Liu. A computational fluid dynamics study of tadpole swimming. J. Exp. Biol., 199(6):1245-1260, 1996.
[73] H. Liu. The three-dimensional hydrodynamics of tadpole locomotion. J. Exp. Biol., 200(22):2807-2819, 1997.
[74] W. Magnus. On the exponential solutions of differential equations for a linear operator. Communications on Pure and Applied Mathematics, VII:649-673, 1954.
[75] J. Marsden and T. Ratiu. Introduction to Mechanics and Symmetry, volume 17. Springer-Verlag, New York, second edition, 2003.
[76] J. Marsden, T. Ratiu, and A. Weinstein. Semidirect products and reduction in mechanics. Trans. Amer. Math. Soc., 281(1), January 1984.
[77] J. E. Marsden. Lectures on Mechanics, volume 174 of London Mathematical Society Lecture Note Series. Cambridge University Press, 1990.
[78] J. E. Marsden, R. Montgomery, and T. S. Ratiu. Reduction, symmetry, and phases in mechanics. Mem. Amer. Math. Soc., 88(436), November 1990.
[79] N. Maslov. Maneuverability and controllability of dolphins. Bionika, 4:46-50, 1970.
[80] R. Mason. Fluid locomotion and trajectory planning for shape-changing robots. PhD thesis, California Institute of Technology, Pasadena, California, 2003.
[81] R. Mason and J. W. Burdick. Propulsion and control of deformable bodies in an ideal fluid. In IEEE International Conference on Robotics and Automation, 1999.
[82] J. B. Melli, C. W. Rowley, and D. S. Rufat. Motion planning for an articulated body in a perfect planar fluid. SIAM J. Appl. Dyn. Sys., 5(4):650-669, 2006.
[83] M. Milam, K. Mushambi, and R. Murray. A new computational approach to real-time trajectory generation forconstrained mechanical systems. Decision and Control, 2000. Proceedings of the 39th IEEE Conference on, 1, 2000.
[84] T. Miloh and A. Galper. Self-Propulsion of General Deformable Shapes in a Perfect Fluid. In Mathematical and Physical Sciences, volume 442, pages 273299, 1993.
[85] R. Montgomery. How much does the rigid body rotate? A Berry's phase from the 18th century. American Journal of Physics, 59:394, 1991.
[86] K. Morgansen, V. Duidam, R. Mason, J. Burdick, and R. Murray. Nonlinear control methods for planar carangiform robot fish locomotion. In IEEE International Conference on Robotics and Automation, volume 1, 2001.
[87] R. M. Murray, J. W. Burdick, S. D. Kelly, and J. Radford. Trajectory generation for mechanical systems with application to robotic locomotion. In Proc. Workshop on Algorithmic Foundations of Robotics, 1998.
[88] S. Nowacek. High Performance Turning Capabilities During Foraging By Bottlenose Dolphins (Tursios Truncatus). Marine Mammal Science, 20(3):498-509, 2004.
[89] C. Oseen. Über die Stokessche Formel und über die verwandte Aufgabe in der Hydrodynamik. Arkiv Mat., Astron., Fysik, 6:29, 1910.
[90] J. Ostrowski, J. Desai, and V. Kumar. Optimal Gait Selection for Nonholonomic Locomotion Systems. The International Journal of Robotics Research, 19(3):225, 2000.
[91] C.-K. Pang. A Computer Code (USPOTF2) for Unsteady Incompressible Flow Past Two Airfoils. Master's thesis, Naval Postgraduate School, Monterey, California, September 1988.
[92] M. Pastoor, B. Noack, R. King, and G. Tadmor. Spatiotemporal Waveform Observers and Feedback in Shear Layer Control. 44 th AIAA Aerospace Sciences Meeting and Exhibit, pages 1-9, 2006.
[93] M. Platzer, K. Neace, and C. Pang. Aerodynamic analysis of flapping wing propulsion. AIAA, Aerospace Sciences Meeting and Exhibit, 31 st, Reno, NV, page 1993, 1993.
[94] I. Plaut. Effects of fin size on swimming performance, swimming behaviour and routine activity of zebrafish Danio rerio. J. Exp. Biol., 203(4):813-820, 2000.
[95] H. Power. The completed double layer boundary integral equation method for two-dimensional stokes flow. IMA J. Appl. Math., 51:123-145, 1993.
[96] C. Pozrikidis. Boundary integral and singularity methods for linearized viscous flow. Cambridge University Press, New York, 1992.
[97] L. Prandtl. Applications of modern hydrodynamics to aeronautics. Technical report, NACA TR-116, June 1921.
[98] E. M. Purcell. Life at low Reynolds number. Am. J. Phys., 45(1):3-11, Jan. 1977.
[99] J. E. Radford. Symmetry, Reduction and Swimming in a Perfect Fluid. PhD thesis, California Institute of Technology, Pasadena, California, 2003.
[100] J. E. Radford and J. W. Burdick. Local motion planning for nonholonomic control systems evolving on principal bundles. In Proc. Mathematical Theory of Networks and Systems, 1998.
[101] P. Saffman. The self-propulsion of a deformable body in a perfect fluid. Journal of Fluid Mechanics, 28:385-389, 1967.
[102] T. Sarpkaya. An inviscid model of two-dimensional vortex shedding for transient and asymptotically steady separated flow over an inclined plate. Journal of Fluid Mechanics, 68(01):109-128, March 1975.
[103] J. W. Schaefer and S. Eskinazi. An analysis of the vortex street generated in a viscous fluid. Journal of Fluid Mechanics, 6:241-260, 1959.
[104] W. SEARS. Some Aspects of Non-Stationary Airfoil Theory and Its Practical Application. AIAA Journal, 8(3):104-108, 1941.
[105] A. Shapere and F. Wilczek. Geometry of self-propulsion at low Reynolds number. J. Fluid Mech., 198:557-585, 1989.
[106] D. Smith. Estimates at infinity for stationary solutions of the Navier-Stokes equations in two dimensions. Archive for Rational Mechanics and Analysis, 20(5):341-372, 1965.
[107] T. Suzuki, T. Colonius, and D. MacMartin. Inverse technique for vortex imaging and its application to feedback flow control. AIAA, 4260, 2003.
[108] G. Tadmor. Observers and feedback control for a rotating vortex pair. IEEE Transactions on Control Systems Technology, 12(1):36-51, 2004.
[109] D. Tam and A. Hosoi. Optimal Stroke Patterns for Purcell's Three-Link Swimmer. Physical Review Letters, 98(6):68105, 2007.
[110] G. Taylor. Analysis of the Swimming of Microscopic Organisms. Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 209(1099):447-461, 1951.
[111] G. Taylor. Analysis of the Swimming of Long and Narrow Animals. Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 214(1117):158-183, 1952.
[112] G. Taylor. The Action of Waving Cylindrical Tails in Propelling Microscopic Organisms. Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences, 211(1105):225-239, 1952.
[113] A. Techet. Experimental visualization of the near-boundary hydrodynamics about fish-like swimming bodies. PhD thesis, Massachusetts Institute of Technology, 2001.
[114] A. Techet and M. Triantafyllou. Boundary Layer Relaminarization in Swimming Fish. Proceedings of the International Offshore and Polar Engineering Conference, 1999.
[115] N. Teng. The Development of a Computer Code for the Numerical Solution of Unsteady, Inviscid and Incompressible Flow over an Airfoil,". Master's Thesis, Naval Postgraduate School, Monterey, CA, June, 1987.
[116] T. Theodorsen. General theory of aerodynamic instability and the mechanism of flutter. Technical report, NACA TR-496, 1935.
[117] D. Thompson. Historia Animalium, Book XI, 1910.
[118] M. Triantafyllou and G. Triantafyllou. An efficient swimming machine. Scientific American, 272(3):64-70, 1995.
[119] I. H. Tuncer and M. F. Platzer. Thrust generation due to airfoil flapping. AIAA J., 34(2), February 1996.
[120] M. Vezza and R. Galbraith. A method for predicting unsteady potential flow about an aerofoil. International Journal for Numerical Methods in Fluids, 5:347-356, 1985.
[121] T. von Karman and J. Burgers. General Aerodynamic Theory - Perfect Fluids, volume II. Dover Publications, New York, 1963.
[122] H. Wagner. Uber die entstehung des dynamischen auftriebes von tragflugeln. Zeitschrift für Angewandte Mathematik und Mechanik, 5(1):17-31, 1925.
[123] P. Webb. Efficiency of pectoral fin propulsion of Cymatogaster aggregata. Swimming and Flying in Nature, 2:573-584, 1974.
[124] P. Webb. Speed, Acceleration and Manoeuvrability of Two Teleost Fishes. Journal of Experimental Biology, 102(1):115-122, 1983.
[125] T. Williams, G. Bowtell, J. Carling, K. Sigvardt, and N. Curtin. Interactions between muscle activation, body curvature and the water in the swimming lamprey. Symp Soc Exp Biol, 49:49-59, 1995.
[126] D. Willis, J. Peraire, and J. White. FastAero-A Precorrected FFT-Fast Multipole Tree Steady and Unsteady Potential Flow Solver. High Performance Computation for Engineered Systems, 2005.
[127] M. Wolfgang, J. Anderson, M. Grosenbaugh, D. Yue, and M. Triantafyllou. Near-body flow dynamics in swimming fish. Journal of Experimental Biology, 202(17):2303-2327, 1999.
[128] T. Y. Wu. Swimming of a waving plate. J. Fluid Mech., 10, 1961.
[129] T. Y. Wu. Hydrodynamics of swimming propulsion. part 1. swimming of a two-dimensional flexible plate at variable forward speeds in an inviscid fluid. $J$. Fluid Mech., 46, 1971.
[130] Q. ZHU, M. Wolfgang, D. Yue, and M. Triantafyllou. Three-dimensional flow structures and vorticity control in fish-like swimming. Journal of Fluid Mechanics, 468:1-28, 2002.
[131] S. Zimmerman. Submarine Technology for the 21st Century. Trafford Publishing, 2000.
[132] M. Zylber, G. Failla, and A. Le Bas. Stenurus globicephalae Baylis et Daubney, 1925 (Nematoda: Pseudaliidae) from a False Killer Whale, Pseudorca crassidens (Cetacea: Delphinidae), Stranded on the Coast of Uruguay. Memórias do Instituto Oswaldo Cruz, 97:221-225, 2002.

## Appendix A

## Potential flow code

This potential flow code written in MATLAB was developed in collaboration with Eva Kanso. Refer to Chapter 3 for a description of the potential flow swimmer model.

```
A.1 driver.m
% Driver file for running potential flow swimmer simulations
% INPUTS: a, e, c, N, tsteps, mode1, mode2, phi
% OUTPUTS: g (group motion)
% ellipse semi-major (a) and semi-minor (e) axes length
a}=1; e=.1
% e: offset distance between joint and tip of ellipse
c = . 2;
% N: # of panels per ellipse
N = 50;
% tsteps: # of time steps/period
tsteps = 50;
% mode1, mode2 and phi specify shape changes (see shape_var.m)
mode1 = 1; mode2=1; phi = 0;
l = a+e;
% time vector (one period)
h}=2*\mathbf{pi}/\mathrm{ tsteps; t = 0:h:2*pi; nbt = length(t);
% initialize group variables
g = zeros(nbt,3);
% discretization of each ellipse
[zcg1,t1,n1, del1] = ellipse(a,b,N);
% Actual mass and inertia of each ellipse
m}=\textrm{a}*\textrm{b}/\mp@subsup{\textrm{l}}{}{\wedge}2
global II
II = [(m*(a^2+b^2)/4)/1^2, 0, 0; 0,m, 0; 0, 0,m];
```

```
% Integration by classical 4th-order Runge-Kutta method
for i = 1:nbt-1
    g(i+1,: ) = rk4(t(i),g(i,:),h,l, zcg1,t1,n1, del1,N,mode1,mode2,phi );
end
```


## A. 2 rk4.m

```
function g1 = rk4(t0,g0,h,l, zcg1, t1, n1, del1, npts,mode1,mode2, phi)
% Fourth order fixed time step Runge-Kutta scheme to advance motion in
% time.
% shape variables
[th1_0, th2_0, th 1dot_0, th2dot_0] = shape_var(t0,mode1,mode2,phi);
[th1_1,th2_1, th1dot_1, th2dot_1] = shape_var(t0+h/2,mode1,mode2, phi);
[th1_2,th2_2, th1dot_2, th2dot_2]= shape_var(t0+h,mode1,mode2, phi );
% connection
A0 = connection(th1_0,th2_0,th1dot_0,th2dot_0, l, zcg1,t1,n1, del1, npts);
A1 = connection(th1_1,th2_1, th1dot_1, th2dot_1, l, zcg1,t1,n1, del1, npts);
A2 = connection(th1_2,th2_2,th1dot_2,th2dot_2, l, zcg1, t1, n1, del1, npts);
% non-dimensionalize connection - multiply u & v velocities by l
A0}=[\begin{array}{ccccccc}{1}&{0}&{0;0}&{1}&{0;0}&{0}&{1}\end{array}]*\textrm{A}0
A1 = [[11 0 0;0
A2}=[\begin{array}{llllllll}{1}&{0}&{0;0}&{1}&{0;0}&{0}&{1}\end{array}]*\textrm{A}2
K1 = vel_fun(A0,g0);
K2 = vel_fun(A1,g0 + h*K1/2);
K3 = vel_fun(A1,g0 + h*K2/2);
K4 = vel_fun(A2,g0 + h*K3);
g}1=\textrm{g}0+\textrm{h}*(\textrm{K}1+2*\textrm{K}2+2*\textrm{K}3+\textrm{K}4)/6
```


## A. 3 shape_var.m

```
function [th1, th2, th1dot,th2dot] = shape_var(t,mode1,mode2, phi)
% Prescribe shape changes of angles between bodies
% A1 = Amplitude of motion
% offset = Offset along diagonal
% phi = phase shift
switch mode1
    case 1
        % circles in shape space
        switch mode2
            case 1
                    % for circle about origin
                    A1=1.5;
                    offset=0;
            case 2
                    % for circle shifted,
                    A1 = 0.8;
                offset = 0.8;
            case 3
                % for circle shifted,
                A1 = - 0.45;
                offset = - 1.5;
        end
        th1 = -offset+A1*(\boldsymbol{cos}(\textrm{t}-\textrm{phi}));
        th2 = offset+A1*(sin(t-phi));
        th1dot = - A1*sin(t-phi);
        th2dot = A1* cos(t-phi);
```

case 2
Amp $=2.0$;
\% counter-clockwise square in shape space
if $\quad \mathrm{t}<\mathbf{p i} / 2$
th $1=0$;
th2 $=\operatorname{Amp} *(\mathrm{t}) /(\mathbf{p i} / 2) ;$ th1dot $=0$;
th2dot $=\mathrm{Amp} /(\mathbf{p i} / 2) ;$
else if $\mathrm{t}<$ pi \& $\mathrm{t}>=\mathbf{p i} / 2$
$\operatorname{th} 1=-\operatorname{Amp} *((\mathrm{t}-\mathbf{p i} / 2) /(\mathbf{p i} / 2)) ;$
th2 $=$ Amp;
th1dot $=-\operatorname{Amp} /(\mathbf{p i} / 2)$;
th2dot $=0$; else if $\mathrm{t}>=\mathbf{p i} \& \mathrm{t}<3 * \mathbf{p i} / 2$
th1 $=-$ Amp;
$\operatorname{th} 2=\operatorname{Amp}-\operatorname{Amp} *((\mathrm{t}-\mathbf{p i}) /(\mathbf{p i} / 2)) ;$ th1dot $=0$; th 2 dot $=-\operatorname{Amp} /(\mathbf{p i} / 2)$;
else if $\mathrm{t}<=2 * \mathbf{p} \mathbf{i}+\mathbf{e p s} \& \mathrm{t}>=3 * \mathbf{p i} / 2$

## th1 $=-\operatorname{Amp}+\operatorname{Amp} *((\mathrm{t}-3 * \mathbf{p i} / 2) /(\mathbf{p i} / 2))$;

th2 $=0$;
th1dot $=\mathrm{Amp} /(\mathbf{p i} / 2)$;
th $2 \operatorname{dot}=0$;
end
end end
end
end

## A. 4 connection.m

```
function A = connection(th1, th2, th1dot,th2dot, l, zcg1,t1, n1, del1, npts);
% Computes the hydromechanical connection which relates the shape changes
% to the group motion
global II;
% non-dimensional added inertias
[I33,I11,I22,I31, I32,I12] = getadmass(th1, th2, l, zcg1,t1,n1, del1, npts);
% motions of 1 and 2 relative to 3
x1 = [th1,(1+\boldsymbol{cos}(th1)), sin(th1)];
x2 = [th2,-(1+\operatorname{cos}(th2)),-\operatorname{sin}(th2)];
zeta1 = [th1dot; 0; th1dot]; % (th1dot, x1dot, y1dot)
zeta2=[th2dot; 0; -th2dot];
% Inverse adjoint action
Adx1inv = adjointinv(x1);
Adx2inv = adjointinv(x2);
% Adjoint action
Adx1 = adjoint(x1);
Adx2 = adjoint(x2);
% % actual mass
I33 = I33 + II;
I}22=\textrm{I}22+Adx2inv'* *II*Adx2inv;
I11 = I11 + Adx1inv'* II *Adx1inv;
% locked moment of inertia
IIloc}=\textrm{I}11+\textrm{I}22+\textrm{I}33+2*\textrm{I}12+2*\textrm{I}31+2*I32
% momenta of ellipses 1 and 2
zeta1_temp = Adx1*zeta1;
zeta2_temp = Adx2*zeta 2;
h1p2 = (I11+I12+I31)*zeta1_temp + (I2 2+I32+I12)*zeta2_temp;
```

```
% connection
A= inv(IIloc)*h1p2;
% switch components around to put in form (u,v,omega) rather than (omega,u,v)
Atemp = A;
A(1,:) = Atemp (2,:);
A(2,:) = Atemp (3,:);
A(3,:) = Atemp (1,:);
```


## A. 5 getadmass.m

```
function [m11,m22,m33,m12,m13,m23]= getadmass(th1,th2,l, zcg1, t1, n1, del1,N)
% define geometry
[zc,zcg,t,n,del] = threebody(th1,th2,l, zcg1,t1,n1, del1,N);
% calculate the influence matrices and velocity potential
[An,Bt,phi] = influence(zc,t,n, del', 3*N);
% calculate the added masses
[m11,m22,m33,m12,m13,m23]= admass}(An,phi, zc,n, del , l,N)
```


## A. 6 threebody.m

```
function [zc, zcg,t,n, del] = threebody(th1,th2,l, zcg1,t1,n1,del1, npts)
% Computes location of control points wrt inertial frame (zc) and wrt frame
% for 3 bodies.
% Also finds the normal (n) and tangent (t) vectors at each control point
% ____OUTPUT
% zc position of control pts w.r.t inertial frame
% components of vectors tangent to panels
% n components of outward normal vectors
% panel length
N1 = npts; N1p1 = N1 + 1;
N2 = 2*npts; N2p1 = N2 + 1;
N = 3* npts;
% orientation of front and rear ellipses
cth1 = \boldsymbol{cos}(th1); sth1= 汸(th1);
```



```
% position of c.o.m of ellipses (l=a+c)
zg}(1,1)=0
zg}(1,2)=0
zg}(2,1)=1*(1+cth1)
zg(2,2) = l*sth1;
zg}(3,1)=-1*(1+\operatorname{cth}2)
zg}(3,2)=-1*sth2
% initialize
xg = zeros(N,1); yg = zeros(N,1);
zcg = zeros(N,2); zc = zeros(N,2);
t}=\operatorname{zeros}(\textrm{N},2);\quad\textrm{n}=\boldsymbol{zeros}(\textrm{N},2)
Del = zeros(N,1);
% c.o.m
```

```
xg(1:npts,1) = ones(npts,1)*zg(1,1);
xg(N1p1:N2,1) = ones (npts,1)*zg(2,1);
xg}(\textrm{N}2\textrm{p}1:N,1)=oones(npts,1)*zg(3,1)
yg(1:npts,1) = ones(npts,1)*zg(1,2);
yg(N1p1:N2,1) = ones(npts,1)*zg(2,2);
yg(N2p1:N,1) = ones(npts,1)*zg(3,2);
% orientation of front of rear ellipses
orient2(:,1) = ones(npts,1).* cth1;
orient2(:,2) = ones(npts,1).*sth1;
orient3(:,1) = ones(npts,1).* cth2;
orient3(:,2) = ones(npts,1).*sth2;
% ellipses
zcg}(1:npts,1)= zcg1(:, 1)
zcg(1:npts,2) = zcg1(:, 2);
zcg(N1p1:N2,1) = zcg1(:, 1).* orient2(:,1) - zcg1(:, 2).*orient2(:, 2);
zcg(N1p1:N2,2) = zcg1(:,1).*orient2(:,2) + zcg1(:, 2).*orient2(:,1);
zcg(N2p1:N,1) = zcg1(:,1).*orient3(:,1) - zcg1(:, 2).*orient3(:, 2);
zcg(N2p1:N,2) = zcg1(:,1).*orient3(:,2) + zcg1(:, 2).*orient3(:,1);
zc}=[\textrm{xg},\textrm{yg}]+\textrm{zcg}
% tangent vectors
t(1:npts,1) = t1 (:, 1);
t(1:npts,2) = t1 (:, 2);
t(N1p1:N2,1) = t1 (:, 1).*orient2(:, 1) - t1 (:, 2).*orient2(:, 2);
t(N1p1:N2,2) = t1 (:, 1).*orient2(:,2) + t1 (:, 2).*orient2(:, 1);
t(N2p1:N,1) = t1 (:, 1).*orient3(:, 1) - t1 (:, 2).*orient3(:, 2);
t(N2p1:N,2) = t1 (:, 1).*orient3(:, 2) + t1 (:, 2).*orient3(:, 1);
% normal vectors
n(:, 1) = -t (:, 2);
n(:, 2) = t (:, 1);
% panel length
del(1:npts,:) = del1;
del(N1p1:N2,:) = del1;
del(N2p1:N,:) = del1;
```


## A. 7 influence.m

| $\begin{aligned} & \text { function }[\mathrm{An}, \mathrm{Bt}, \mathrm{Phi}]=\text { influence }(\mathrm{zc}, \mathrm{t}, \mathrm{n}, \text { del }, \mathrm{N}) \\ & \% \xrightarrow[I N P U T]{ } \end{aligned}$ |  |
| :---: | :---: |
|  |  |
| \% |  |
| \% zc position of control pts |  |
| \% $t$ components of vectors tangent to panels |  |
| \% $n$ components of outward normal vectors |  |
| \% del panel length |  |
| \% |  |
| \% zc, $t$ and $n$ are w.r.t inertial frame |  |
| \% |  |
| \% |  |
| \% _ INTERNAL VARIABLES |  |
| \% |  |
| \% Xrel $(i, j) \& \operatorname{Yrel}(i, j)$ | j) $\}$ Yrel ( $i, j$ coordinates of control pt i (Ci) relative |
| \% | to control pt j (Cj) w.r.t inertial frame |
| \% |  |
| \% Cn ( $i, j$ ) \& Ct $(i, j)$ | \& Ct $(i, j)$ normal and tangential coordinates of Ci relative |
| \% | to Cj w.r.t. a frame attached to the panel j |
| \% |  |
| \% Vn(i, j) छ Vt (i, j) | 8 Vt(i, $)$ normal and tangential velocities induced |

```
%
                at Ci due to a constant source distribution 
        velocities Vn(i,j) and Vt(i,j)
        expressed w.r.t. inertial frame
    % -_OUTPUT
% An(i,j) छBt(i,j) normal and tangential velocities induced at Ci
% An(i,j) & Bt(i,j) lormal and tangential velocities induced at Ci 
% Phi potential function
% initialize
Xc}=\boldsymbol{zeros}(N,N); Yc= zeros(N,N)
tx = zeros(N,N); ty = zeros(N,N);
nx}=\boldsymbol{zeros}(N,N); ny = zeros(N,N)
Ds = zeros(N,N);
% assign
Xc = zc (:, 1)* ones(1,N); Yc = zc (:, 2)*ones(1,N);
tx}=\textrm{t}(:,1)*\operatorname{ones}(1,\textrm{N});\quad\textrm{ty}=\textrm{t}(:,2)*\operatorname{ones}(1,\textrm{N})
nx}=\textrm{n}(:,1)*\mathrm{ ones (1,N); ny = n(:,2)*ones(1,N);
del=del';
Ds = 0.5.*ones(N,1)* del';
% compute Xrel(i,j) and Yrel(i,j)
XREL = Xc-Xc';
YREL = Yc-Yc';
% compute Cn(i,j) and Ct(i,j)
Cn = XREL.*nx' + YREL.*ny';
Ct = XREL.*tx' + YREL.*ty';
% compute Vn(i,j) and Vt(i,j)
temp1 = Cn.^2;
temp2 = ( Ct + Ds ).^ 2;
temp3 = (Ct - Ds).^ 2;
Vt_Num = temp2 + temp1;
Vt_Den = temp3 + temp1;
Vt = log(Vt_Num./Vt_Den );
Vn_Num = 2.*Cn.*Ds;
Vn_Den = Ct.^2 + temp1 - Ds.^2;
angle = 2.*atan(Vn_Num./Vn_Den);
Vn = angle + 2* pi*eye(N,N);
% compute Vx(i,j) and Vy(i,j)
Vx}= Vn.*nx'+Vt.*tx'
Vy = Vn.*ny' + Vt.*ty';
% compute An(i,j) and Bt(i,j)
An=Vx.*nx + Vy.*ny;
Bt = Vx.*tx + Vy.*ty;
% compute Phi
Phi = - Ct.*Vt - Cn.*Vn - Ds.*log((temp2 + temp1).*(temp3 + temp1));
```


## A. 8 admass.m

function $[\mathrm{m} 11, \mathrm{~m} 22, \mathrm{~m} 33, \mathrm{~m} 12, \mathrm{~m} 13, \mathrm{~m} 23]=\operatorname{admass}(\mathrm{An}, \mathrm{Phi}, \mathrm{zcg}, \mathrm{n}, \mathrm{Del}, \mathrm{l}, \mathrm{npts})$
$\%$
$\%$
$\%$ INPUT
$\%$
\% An $(i, j) \quad$ normal velocity induced at Ci due

```
% N
                to a constant source distribution at panel j
                Expressed w.r.t a frame attached to the panel i
                                potential function
                                normal vectors to panels
                position of the control points relative to c.o.m
                    length of panels
% __INTERNAL VARIABLES
% 
% -OUTPUT
N1 = npts; N N1p1 = N1 + 1;
N2 = 2* npts; N2p1 = N2 + 1;
N3 = 3* npts;
density = 1/pi;
% define boundary conditions
% angular rotation at unit speed
avel = zcg(:,1).*n(:, 2) - zcg(:,2).*n(:,1);
% translations at unit speed in the BODY1 fixed frame
xvel = n (:,1);
yvel = n(:, 2);
% body 1
vfn_1x = zeros(N3,1); vfn_1y = zeros(N3,1); vfn_1a= zeros(N3,1);
vfn_1x(1:N1,1) = xvel(1:N1, 1);
vfn_1y(1:N1,1) = yvel(1:N1,1);
vfn_1a(1:N1,1) = avel(1:N1,1);
% body 2
vfn_2x = zeros(N3,1); vfn_2y= zeros(N3,1); vfn_2a= zeros(N3,1);
vfn_2x(N1p1:N2,1) = xvel(N1p1:N2,1);
vfn_2y(N1p1:N2,1)= yvel(N1p1:N2,1);
vfn_2a(N1p1:N2,1) = avel(N1p1:N2,1);
% body 3
vfn_3x = zeros(N3,1); vfn_3y = zeros(N3,1); vfn_3a}=\mp@code{zeros(N3,1);
vfn_3x(N2p1:N3,1) = xvel(N2p1:N3,1);
vfn_3y(N2p1:N3,1) = yvel(N2p1:N3,1);
vfn_3a(N2p1:N3,1) = avel(N2p1:N3,1);
% Source density distribution
inv_An = inv(An);
% body 1
sigma_1x = inv_An*vfn_1x;
sigma_1y = inv_An*vfn_1y;
sigma_1a = inv_An*vfn_1a;
```

```
% body 2
sigma_2x = inv_An*vfn_2x;
sigma_2y = inv_An*vfn_2y;
sigma_2a= inv_An*vfn_2a;
% body 3
sigma_ 3x = inv_An*vfn_3x;
sigma_3y = inv_An*vfn_3y;
sigma_3a}= inv_An*vfn_3a;
% compute potential functions
% body 1
phi_1x = Phi*sigma_1x;
phi_1y = Phi*sigma_1y;
phi_1a= Phi*sigma_1a;
% body 2
phi_2x = Phi*sigma_2x;
phi_2y = Phi*sigma_2y;
phi_2a}= Phi*sigma_2a;
% body 3
phi_3x = Phi*sigma_3x;
phi_3y = Phi*sigma_3y;
phi_3a}= Phi*sigma_3a
% compute the added masses
% body 1
% BODY1 fixed frame
m11_xx = density*sum(phi_1x.*vfn_1x.*Del);
m11_yy = density*sum(phi_1y.*vfn_1y.*Del);
m11_aa= density*sum(phi_1a.*vfn_1a.*Del);
m11_xy = density*sum(phi_1x.*vfn_1y.*Del);
m11_xa= density *sum(phi_1x.*vfn_1a.*Del);
m11_ya = density*sum(phi_1y.*vfn_1a.*Del);
% body 2
% BODY2 fixed frame
m22_xx = density *sum(phi_2x.*vfn_2x.* Del);
m22_yy = density*sum(phi_2y.*vfn_2y.*Del);
m22_aa}=\mathrm{ density *sum(phi_2a.*vfn_2a.*Del);
m22_xy = density*sum(phi_2x.*vfn_2y.*Del);
m22_xa= density*sum(phi_2x.*vfn_2a.*Del);
m22_ya= density*sum(phi_2y.*vfn_2a.*Del);
% body 3
% BODY3 fixed frame
m33_xx = density *sum(phi_3x.*vfn_3x.*Del);
m33_yy = density*sum(phi_3y.*vfn_3y.*Del);
m33_aa= density*sum(phi_3a.*vfn_3a.*Del);
m33_xy = density*sum(phi_3x.*vfn_3y.*Del);
m33_xa= density*sum(phi_3x.*vfn_3a.*Del);
m33_ya}=\mathrm{ density*sum(phi_3y.*vfn_3a.*Del);
% influence of body 1 on body 2
m12_xx = density*sum(phi_1x.*vfn_2x.*Del);
m12_yy = density*sum(phi_1y.*vfn_2y.*Del);
m12_aa= density*sum(phi_1a.*vfn_2a.*Del);
m12_xy = 0.5* density*(sum(phi_1x.*vfn_2y.* Del) + sum(phi_ 2x.*vfn_1y.*Del));
m12_xa = 0.5*density*(sum(phi_1x.*vfn_2a.*Del) + sum(phi_2x.*vfn_1a.*Del));
m12_ya = 0.5* density*(sum(phi_1y.*vfn_2a.*Del) + sum(phi_2y.*vfn_1a.*Del));
% influence of body 2 on body 3
m23_xx = density*sum(phi_2x.*vfn_3x.*Del);
```

```
m23_yy = density *sum(phi_2y.*vfn_3y.*Del);
m23_aa = density*sum(phi_2a.*vfn_3a.*Del);
m23_xy = 0.5*density*(sum(phi_2x.*vfn_3y.*Del) + sum(phi_3x.*vfn_2y.*Del));
m23_xa = 0.5*density*(sum(phi_2x.*vfn_3a.*Del) + sum(phi_ 3x.*vfn_2a.*Del));
m23_ya = 0.5*density*(sum(phi_2y.*vfn_3a.*Del) + sum(phi_3y.*vfn_2a.*Del));
% influence of body 1 on body 3
m13_xx = density*sum(phi_1x.*vfn_3x.*Del);
m13_yy = density*sum(phi_1y.*vfn_3y.*Del);
m13_aa = density*sum(phi_1a.*vfn_3a.*Del);
m13_xy = 0.5*density*(sum(phi_1x.*vfn_3y.*Del) + sum(phi_3x.*vfn_1y.*Del));
m13_xa = 0.5*density*(sum(phi_1x.*vfn_3a.*Del) + sum(phi_ 3x.*vfn_1a.*Del));
m13_ya}=0.5*\operatorname{density*(sum(phi_1y.*vfn_3a.*Del) + sum(phi_3y.*vfn_1a.*Del));
% % non-dimensionlize
m}=\mp@subsup{l}{}{\wedge}2; j = m*l^^2; d=m*l; %m=density*pi*l`^2;
% body 1
m11_xx = m11_xx/m;
m11_yy = m11_yy/m;
m11_aa = m11_aa/j;
m11_xy = m11_xy/m;
m11_xa = m11_xa/d;
m11_ya}=m11_ya/d
% body 2
m22_xx = m22_xx/m;
m22_yy = m22_yy/m;
m22_aa= m22_aa/j;
m22_xy = m22_xy/m;
m22_xa = m22_xa/d;
m22_ya}=m22_ya/d
% body 3
m33_xx = m33_xx/m;
m33_yy = m33_yy/m;
m33_aa = m33_aa/j;
m33_xy = m33_xy/m;
m33_xa = m33_xa/d;
m33_ya = m33_ya/d;
% influence of body 1 on body 2
m12_xx = m12_xx/m;
m12_yy = m12_yy/m;
m12_aa = m12_aa/j;
m12_xy = m12_xy/m;
m12_xa = m12_xa/d;
m12_ya= m12_ya/d;
% influence of body 2 on body 3
m23_xx = m23_xx/m;
m23_yy = m23_yy/m;
m23_aa = m23_aa/j;
m23_xy = m23_xy/m;
m23_xa = m23_xa/d;
m23_ya}=m23_ya/d
% influence of body 1 on body 3
m13_xx = m13_xx/m;
m13_yy = m13_yy/m;
m13_aa= m13_aa/j;
m13_xy = m13_xy/m;
m13_xa = m13_xa/d;
m13_ya = m13_ya/d;
```

```
% assign
m11 = [m11_aa, m11_xa, m11_ya; ...
    m11_xa, m11_xx, m11_xy;...
    m11_ya, m11_xy, m11_yy];
m22 = [m22_aa, m22_xa, m22_ya; ...
    m22_xa, m22_xx, m22_xy ; ...
    m22_ya, m22_xy, m22_yy];
m33 = [m33_aa, m33_xa, m33_ya;...
    m33_xa, m33_xx, m33_xy ; ...
    m33_ya, m33_xy, m33_yy];
m12 = [m12_aa, m12_xa, m12_ya;...
    m12_xa, m12_xx, m12_xy ; ...
    m12_ya, m12_xy, m12_yy];
m23 = [m23_aa, m23_xa, m23_ya;...
    m23_xa, m23_xx, m23_xy;...
    m23_ya, m23_xy, m23_yy];
m13 = [m13_aa, m13_xa, m13_ya;...
    m13_xa, m13_xx, m13_xy;..
    m13_ya, m13_xy, m13_yy];
```


## A. 9 adjointinv.m

```
function Adginv = adjointinv(g)
theta = g(1);
x = g(2);
y = g(3);
ct = \boldsymbol{cos(theta);}
st = sin(theta);
Adginv = [ 1, 0, 0; 0. 
    -y*ct+x*st,
```


## A. 10 adjoint.m

```
function Adg = adjoint(g)
theta = g(1);
x = g (2);
y = g(3);
ct = соs(theta);
st = sin(theta);
Adg = [ 1, 0, 0;...
    y, ct, -st;...
    -x, st, ct];
```


## A. 11 vel fun.m

```
function gdot = vel_fun(A,g)
% g = (x,y,beta)
% A = (u,v,omega)
% differential equations
```

```
gdot = -[A(1)*\boldsymbol{cos}(g(3)) - A(2)*\boldsymbol{\operatorname{sin}}(\textrm{g}(3)),..
    A(1)*\boldsymbol{\operatorname{ln}}(\textrm{g}(3))+\textrm{A}(2)*\boldsymbol{\operatorname{cos}}(\textrm{g}(3)),\ldots
    A(3)];
```


## Appendix B

## Stokes flow code

This MATLAB code is an implementation of the Stokes flow swimmer model described in Chapter 4.

## B. 1 driverStokes.m

```
function [g,gdot,A,s,sdot,time] = driverStokes(N,P,tsteps,mode,a,b,eta)
%
% [g,gdot,A,s,sdot,time] = driverStokes(N,P,mode,a,b,e)
% INPUTS
% N : # of panels on body
% P : # of periods to run simulation
% mode: 1 : Shapere छ Wilzcek, (JFM 1989) validation case
            2 : Becker, Koehler, Stone, (JFM 2003) validation case
            3 : 3-link swimmer forward gait from thesis
            4:3-link swimmer turning gait from thesis
    OPTIONAL INPUT VARIABLES (for mode=2, 3-link body)
% a : semi-major axis length
% b : semi-minor axis length
% eta : ratio of middle link length to outer link length (for mode 2)
%
% OUTPUTS
%}g=[x(t)] - motion in body group where beta is defined positive OCW
    y(t)] from horizontal
    [beta(t)]
gdot: swimmer velocity in inertial frame (xdot,ydot,betadot)
% A : swimmer velocity in body frame (u,v,omega)
% s: shape variable values
% sdot: shape variable derivatives
% time: time vector
%
% tsteps: # of time steps per period
global i
g}=[0;0;0]; % x, y, bet
h}=2*\mathbf{pi}/(\mathrm{ tsteps - 1);
```

```
time = 0:h:2* pi*P; nbt = length(time);
for i=1:nbt
    [s, sdot] = shape_var(time(i),mode);
    [f0 g0 h0] = StokesConnection(N,s,mode,a,b,eta);
    A(:, i) = [f0 g0 h0]*sdot';
    gdot = vel_fun(A(:, i ),g(:, i ));
    g(:, i+1)=g(:, i )+gdot '*h;
end
```


## B. 2 shape_var.m

```
function [s,sdot] = shape_var(t,mode)
% Prescribe shape changes of angles between bodies
% mode 1 : Shapere &G Wilzcek, (JFM 1989) validation case
% mode 2 : Becker, Koehler, Stone, (JFM 2003) validation case
% mode 3 : 3-link swimmer forward gait from thesis
% mode 4 : 3-link swimmer turning gait from thesis
switch mode
    case 1
        % Shapere छ Wilzcek, JFM, 1989
        s0 = 0;
        s2=}\boldsymbol{\operatorname{cos}}(\textrm{t})
        s3=}\operatorname{sin}(t)
        s0dot = 0;
        s2dot = -sin(t);
        s3dot = \boldsymbol{cos(t);}
        s = [s0, s2, s3];
        sdot = [s0dot, s2dot, s3dot];
    case 3
        % Becker, Koehler, Stone (JFM 2003) validation case of 3-link
        % swimmer - forward gait
        gamma1 = pi/3;
        if t < < * pi / 4;
            th2 = gamma1*((4/pi)*t-1);
            th1 = gamma1;
            th2dot = (4/ pi );
            th1dot = 0;
        else if (t<2*pi/2)
                    th2 = gamma1;
                        th1 = -gamma1*(4/pi)*t+3*gamma1;
                        th2dot = 0;
                th1dot = -gamma1*(4/pi);
            else if (t<2*pi*(3/4))
                    th2 = -gamma1*(4/ pi)*t+5*gamma1;
                        th1 = -gamma1;
                        th2dot = -gamma1*(4/pi);
                        th1dot = 0;
                else if (t<=2*pi)
                                    th2 = -gamma1;
                                    th1 = gamma1*(4/pi)*t - 7*gamma1;
                                    th2dot = 0;
                                    th1dot = gamma1 }*(4/\mathbf{pi})
                            end
                end
            end
        end
        s}=[\mathrm{ th1 th2 0];
        sdot = [th1dot th2dot 0];
    case 3 % forward gait, example in Thesis
        th1 = 1.5*\boldsymbol{cos}(t-\mathbf{pi}/4);
        th2 = 1.5*\operatorname{sin}(\textrm{t}-\mathbf{pi}/4);
        th1dot = -1.5*sin(t-pi/4);
        th2dot = 1.5* cos(t-pi/4);
```

```
        s = [lh1 th2 0];
        sdot = [th1dot th2dot 0];
    case 4 % turning gait, example in Thesis
    th1 = -0.8+0.8* cos(t-pi/4);
    th2 = 0.8+0.8* sin(t-pi/4);
    th1dot = -0.8*sin(t-pi/4);
    th2dot = 0.8* cos(t-pi/4);
    s = [lh1 th2 0]
    sdot}=[\mathrm{ th1dot th2dot 0}
end
```


## B. 3 StokesConnection.m

```
function [f0 g0 h0] = StokesConnection(N, s,mode,a,b, eta)
% compute control points zc, t and n vectors at each point, panel lengths
% and mode velocities at unit speed shape change
if mode==1
    [zc,t,n,del,vel_s0, vel_s2, vel_s3] = surfacemodes(N,s,mode);
else if mode>1
            [zc,t,n,del,vel_s0, vel_s2, vel_s3] = ...
                surfacemodes(N, s, mode, a, b, eta);
            end
end
N1 = size(zc,1);
[M] = influencematrix(zc,t,n, del,N1);
%translation is trivial, rigid motion of whole fluid
vel_x = [ ones(N1, 1); zeros(N1,1)];
vel_y = [zeros(N1, 1); ones(N1,1)];
%angular rotation at unit speed
vel_w = [- zc (:, 2); zc (:, 1)];
inv_M = inv(M);
%compute source distributions
phi_x = inv_M*vel_x;
phi_y = inv_M*vel_y;
phi_w = inv_M*vel_w;
phi_s1 = inv_M*vel_s0;
phi_s2 = inv_M*vel_s2;
phi_s3 = inv_M*vel_s3;
a1 = [-zc(:,2); zc(:,1)]; % corresponds to unit CW rotational velocity
```



```
    0 1 sum(phi_w (N1+1:2*N1).* del);
    0 0 (phi_w.*[del;del])'*a1;];
Fs=[\operatorname{sum}(phi_s1( 1: N1).*del) sum(phi_s2( 1: N1).*del) ...
sum(phi_s3( 1: N1).* del);
    sum(phi_s1( N1+1:2*N1).*del)
    [phi_s1.*[del;del]]'*a1
sum(phi_s2( N1+1:2*N1).* del )..
sum(phi_s3( N1+1:2*N1).* del);
[phi_s2.*[del;del]]'*a1
                                    [phi_s3.*[del; del]]'*a1;];
% connection
A}=-\mathbf{inv}(\textrm{Fg})*\textrm{Fs}
f0=A(:, 1); g0=A(:, 2); h0=A(:, 3);
```


## B. 4 surfacemodes.m

function $[z c, t, n, d e l, d 0 P, d 2 P, d 3 P]=$ surfacemodes $(N, s$, mode, $a, b, e t a)$

```
% Prescribe shape changes
% function [P, d1P, d2P, d3P] = STOKES_surface(theta, s);
    s1 s2, s3 - shape variables
    theta - radial angle
    OUTPUT
    P - coordinates in Body Frame
    d1P - derivative with respect to 1st mode variable
    d2P - derivative with respect to 2nd mode variable
    d3P - derivative with respect to 3rd mode variable
    Optional variables needed only for mode 2
    a - semi-major axis
    b - semi-minor axis
    e - ratio of middle link length to outer link length
s0=s(1); s2=s(2); s 3 = s (3);
if mode==1 % Shapere & Wilzcek, JFM 1989 (3.56)
    a = 3/10;
    b}=3/200
    beta = linspace (0,2*pi,N+1);
    theta = (beta (1:N)+\boldsymbol{beta}(2:N+1))/2;
    %MODES:
    F0 = s0*[0*theta, 0*theta ];
    F1 = [\boldsymbol{cos}(theta), sin(theta )];
    F2 = s2*[a*\operatorname{cos(theta) + b*sin}(2*theta) ,\ldots
        -a*\operatorname{sin}(theta) + b*\operatorname{cos}(2*theta)];
    F3 = s3*[-a*\operatorname{cos}(2*theta) + b*sin( theta) ,...
        a*\operatorname{sin}(2*theta) +b*\operatorname{cos}(theta)];
    %SHAPE
    P}=\textrm{F}0+\textrm{F}1+\textrm{F}2+\textrm{F}3
    zc}=[P(1:N)',P(N+1:2*N )']
    %DERIVATIVES
    d0P = F0';
    d2P}=[\textrm{a}*\boldsymbol{\operatorname{cos}}(\textrm{theta})+b*\boldsymbol{sin}(2*\mathrm{ theta ) ,...
        -a*\operatorname{sin}(theta) + b*\boldsymbol{cos}(2*theta)]';
    d3P}=[-a*\boldsymbol{cos}(2*\mathrm{ theta) + b*sin( theta) ,...
        a*\operatorname{sin}(2*theta) + b*\boldsymbol{cos}(theta)]';
    % body geometry to compute normal and tangent vectors and panel lengths
    %MODES:
    beta = beta';
    G0 = s0*[0*beta, 0*beta];
    G1 = [\boldsymbol{cos}(beta), sin(beta)];
    G2 = s 2*[a*\boldsymbol{cos}(beta)+b*\operatorname{sin}(2*\boldsymbol{beta}),-a*\operatorname{sin}(\quad\mathbf{beta})+b*\boldsymbol{cos}(2*\mathbf{beta})];
    G3 = s 3*[-a*\operatorname{cos}(2*\boldsymbol{beta})+b*\operatorname{sin}(beta),a*\operatorname{sin}(2*\boldsymbol{beta})+b*\operatorname{cos}(\mathrm{ beta ) ];}
    Q = G0 + G1 + G2 + G3;
    % Panel length
    Xrel = diff(Q(:, 1)); Yrel = diff(Q(:, 2));
    del = sqrt(Xrel.^2+Yrel.^2);
    % Tangent and normal vectors to panels
    tx = Xrel./del; ty = Yrel./del;
    t = [tx, ty]; n = [ty,-tx];
    t = - t;
else if mode>1
    th1 = s0;
        th2 = s2;
        dt = 0.000001;
        [zc,t,n,del] = threelinkbody(a,b,eta,N,th1, th2);
```

```
        % 2 lines below needed to compute numerical derivative
        [zc_th1, t,n, del] = threelinkbody (a,b,eta,N,th1+dt, th2 );
        [zc_th2,t,n,del] = threelinkbody(a,b,eta,N,th1, th2+dt);
```



```
        d2P}=[zc_th2(:,1) - zc (:, 1); zc_th2 (:, 2) - zc (:, 2)]/dt;'
        d}3\textrm{P}=0*\textrm{d}0\textrm{P};%0\mathrm{ (there is no third mode)
    end
end
```


## B. 5 threelinkbody.m

```
function [zc,t,n, del] = threelinkbody(a,b,eta,N,th1, th2);
% a = semi-major axis length
%b=semi-minor axis length
% N = number of points in ellipse (modified within function)
% th1,th2 = joint angles. th1 = -ive CCW from 0+, th2 = +ive CCW from 0-
% eta = ratio of middle link length to outer link length.
N}=\textrm{N}-\operatorname{mod}(\textrm{N}-2,4)
% needed to ensure a coordinate point at the center
% of the sides of the ellipse ends
beta = linspace(0,2*pi,N+1);
alpha = beta';
% define end ellipses
Px}=(\textrm{a}/\textrm{eta})*\boldsymbol{\operatorname{cos}(alpha); Py = b*sin(alpha);
Px = Px (N+1:-1:1); Py = Py(N+1:-1:1);
xcg1 = (Px (1:1:N) + Px (2:1:N+1))/2;
xcg1(N+1)=xcg1(1);
ycg1=(Py(1:1:N)+Py(2:1:N+1))/2;
ycg1(N+1) = ycg1 (1);
zcg1 = [xcg1,ycg1];
% orientation of front and rear ellipses
cth1= \boldsymbol{cos}(th1); sth1= sin(th1);
```



```
l = 2*a; % middle link length unit (middle unit length is 2a+2e1
12=2*(a/eta); % outer link length unit (actual outer unit length is 2a+e
% position of c.o.m of ellipses
zg(1,1) = 0;
zg}(1,2)=0
zg}(2,1)=1/2+(12/2)*\operatorname{cth}1
zg}(2,2)=(12/2)*sth1
zg}(3,1)=-1/2-(12/2)*\operatorname{cth}2
zg}(3,2)=-(12/2)*sth2
% c.o.m
xg1 = ones(N,1)*zg(1,1);
xg2 = ones(N,1)*zg(2,1);
xg3 = ones (N,1)*zg(3,1);
yg1 = ones(N,1)*zg(1, 2);
yg2 = ones(N,1)*zg(2,2);
yg3 = ones (N,1)*zg(3,2);
% orientation of front of rear ellipses
orient2(:,1) = ones (N,1).* cth1;
orient2(:,2) = ones (N,1).*sth 1;
orient3(:,1) = ones(N,1).* cth2;
orient3(:,2)=}=\mathrm{ ones(N,1).*sth2;
```

```
% cordinates used to compute tangent, normal, and panel length
zcg2(:,1)= zcg1(:, 1).*[orient2 (:, 1);orient2(1,1)] - ...
    zcg1(:, 2).*[orient2 (:,2);orient2(1, 2)];
zcg2(:,2)= zcg1(:, 1).*[orient2(:, 2);orient2(1,2)] + ...
    zcg1 (:, 2).*[ orient2 (:, 1); orient2 (1,1)];
zcg2 = [[xg2;xg2(1)],[yg2;yg2(1)]] + zcg2;
zcg3(:, 1) = zcg1 (:, 1).*[orient3(:, 1);orient3(1,1)] - ...
        zcg1(:, 2).*[orient3 (:, 2); orient3 (1, 2)];
zcg3(:,2) = zcg1(:,1).*[orient3 (:, 2); orient3(1,2)] + ...
        zcg1 (:, 2).*[orient3(:, 1); orient3(1,1)];
zcg3 = [[xg3;xg3(1)],[yg3;yg3(1)]] + zcg3;
dx1 = \boldsymbol{tan}(th1/2)*b;
dy1 = b;
dx2=坓利(th2/2)*b;
dy2 = b;
PT3 = [l/2-dx1, dy1]; PT6 = [l/2+dx1,- dy1];
PT7 = [-1/2+dx2,-dy2]; PT2 = [-1/2-dx2, dy2];
lower = ceil (N/4); upper = ceil ( }3*N/4); max = size(zcg2,1)
PT4 = [zcg2(upper ,:) ]; PT5 = [zcg2(lower,:)];
PT8 = [zcg3(lower,:)]; PT1 = [zcg3(upper,:)];
P1a = [-1/2+12/4 PT7 (2)]; % bottom
P1b}=[-1/2+12/4 PT2(2)]; % to
P1x = [-1/2+12/4 1e8]; % below
P1y = [-1/2+12/4 -1e8]; % above
%order: x a b y
P2a}=[-1/2-(12/4)*\textrm{cth}2+\textrm{b}*\boldsymbol{\operatorname{sin}}(\textrm{th}2) -(12/4)*sth2-b*\operatorname{cos}(th2)]; % bottom
P2b}=[-1/2-(12/4)*\operatorname{cth}2-\textrm{b}*\operatorname{sin}(\textrm{th}2) -(12/4)*sth2+b*\operatorname{cos}(th2)]; % to
P2x = [-l/2-(12/4)*cth2+1e8*\operatorname{sin}(th2) -(12/4)*sth2-1e8*\boldsymbol{cos}(th2)]; % below
P2y = [-l/2-(12/4)* cth2-1e8*\operatorname{sin}(th2) - (12/4)*sth2+1e8*\boldsymbol{cos}(th2)]; % above
P3a = [1/2-12/4 PT7(2)]; % bottom
P3b = [1/2-12/4 PT2 (2)]; % top
P3x = [l/2-12/4 1e8]; % below
P3y = [l/2-12/4 -1e8]; % above
P4a}=[l/2+(12/4)* cth1+b*sin(th1) (l2/4)*sth1-b*\operatorname{cos}(th1)]; % bottom
P4b}=[1/2+(12/4)*\textrm{cth}1-\textrm{b}*\operatorname{sin}(\textrm{th}1)\quad(12/4)*\operatorname{sth}1+\textrm{b}*\boldsymbol{\operatorname{cos}(th1)}];%\mathrm{ top
P}4\textrm{x}=[1/2+(12/4)*\operatorname{cth}1+1\textrm{e}8*\boldsymbol{\operatorname{sin}}(\textrm{th}1)\quad(12/4)*\operatorname{sth}1-1\textrm{e}8*\boldsymbol{cos}(\textrm{th}1)]; % belo
P4y = [l/2+(12/4)*cth1-1e8*\operatorname{sin}(th1) (12/4)*sth1+1e8*\boldsymbol{cos}(th1)]; % above
```

```
x1=[P1x(1) P1a(1) P1y(1)];
```

x1=[P1x(1) P1a(1) P1y(1)];
y1=[P1x(2) P1a(2) P1y(2)];
y1=[P1x(2) P1a(2) P1y(2)];
x2=[P2x(1) P2a(1) P2y(1)];
x2=[P2x(1) P2a(1) P2y(1)];
y2=[P2x(2) P2a(2) P2y(2)];
y2=[P2x(2) P2a(2) P2y(2)];
x3=[P3x(1) P3a(1) P3y(1)];
x3=[P3x(1) P3a(1) P3y(1)];
y3=[P3x(2) P3a(2) P3y (2)];
y3=[P3x(2) P3a(2) P3y (2)];
x4=[P4x(1) P4a(1) P4y(1)];
x4=[P4x(1) P4a(1) P4y(1)];
y4=[P4x(2) P4a(2) P4y(2)];
y4=[P4x(2) P4a(2) P4y(2)];
if abs(th2)>0
if abs(th2)>0
% center of circle;
% center of circle;
[xo,yo]=curveintersect(x1,y1,x2,y2);
[xo,yo]=curveintersect(x1,y1,x2,y2);
% short and long radius length of circles
% short and long radius length of circles
ro1 = P1a(2)-yo;
ro1 = P1a(2)-yo;
ro2 = P1b(2)-yo;
ro2 = P1b(2)-yo;
% th = th2 ->0
% th = th2 ->0
tho1 = linspace(th2,0,N/4);
tho1 = linspace(th2,0,N/4);
xro1 = [xo-ro1*sin(tho1)];
xro1 = [xo-ro1*sin(tho1)];
yro1=[yo+ro1*\operatorname{cos}(tho1)];
yro1=[yo+ro1*\operatorname{cos}(tho1)];
xro2 = [xo-ro2*sin(tho1)];
xro2 = [xo-ro2*sin(tho1)];
yro2=[yo+ro2*\operatorname{cos}(tho1)];

```
    yro2=[yo+ro2*\operatorname{cos}(tho1)];
```

```
else
    xro1 = linspace(P2a(1), P1a(1),N/4);
    yro1 = linspace(P2a(2),P1a(2),N/4);
    xro2 = linspace(P2b(1),P1b(1),N/4);
    yro2= linspace(P2b(2), P1b (2),N/4);
end
if abs(th1)>0
[xo2,yo2]=curveintersect(x3,y3,x4,y4);
ra1 = P3a(2)-yo2;
ra2 = P3b(2)-yo2;
tho2= linspace(th1,0,N/4);
xro3 = [xo2-ra1*sin(tho2)];
yro3=[yo2+ra1* cos(tho2)];
xro4 = [xo2-ra2*sin(tho2)];
yro4=[yo2+ra2*\operatorname{cos}(tho2)];
else
    xro3 = linspace(P4a(1),P3a(1),N/4);
    yro3 = linspace(P4a(2),P3a(2),N/4);
    xro4 = linspace(P4b(1), P3b(1),N/4);
    yro4= linspace(P4b (2), P3b(2),N/4);
end
pts1=[linspace(PT1(1),P2b(1),N/8)' linspace(PT1(2), P2b(2),N/8)'];
pts2 = [linspace(P1b (1), P3b(1),N/2)' linspace(P1b(2), P3b (2),N/2)'];
pts3 = [linspace(P4b (1),PT4(1),N/8)' linspace(P4b(2),PT4(2),N/8)'];
pts4=[linspace(PT5 (1), P4a(1),N/8)', linspace(PT5 (2), P4a(2),N/8)'];
pts5 = [linspace(P3a(1), P1a(1),N/2), linspace(P3a(2), P1a(2),N/2),'];
pts6 = [linspace(P2a(1),PT8(1),N/8)' linspace(P2a(2),PT8(2),N/8)'];
body = [zcg2(upper:max, 1), zcg2(upper:max, 2);
    zcg2(2:lower,1), zcg2(2:lower,2);
    pts4(2:length(pts4),1),pts4(2:length(pts4),2);
    xro3(2:length(xro3))' yro3(2:length(xro3))';
    pts5(2:length(pts5),1),pts5(2:length(pts5),2);
    xro1(length(xro1)-1:-1:1)', yro1(length(xro1) - 1: - 1:1)';
    pts6(2:length(pts6),1),pts6(2:length(pts6),2);
    zcg3(lower+1:upper,1), zcg3(lower+1:upper, 2);
    pts1(2:length(pts1),1),pts1(2:length(pts1), 2);
    xro2(2:length(xro2))', yro2(2:length(xro2))';
    pts2(2:length(pts2),1),pts2(2:length(pts2), 2);
    xro4(end-1:-1:1)' yro4(end-1:-1:1)';
    pts3(2:length(pts3),1),pts3(2:length(pts3), 2)];
bsize = size(body,1);
t = [difff(body (:, 1)), difff(body (:, 2))];
n = [-t (:, 2), t (:, 1)];
zc}=[(\operatorname{body}(1:\operatorname{bsize}-1,1)+\operatorname{body}(2:\operatorname{bsize},1))/2,
    (body (1: bsize - 1,2)+body (2: bsize, 2))/2];
del = sqrt(t (:, 1).^2+t(:, 2).^2 );
t}=[\textrm{t}(:,1)./del,\textrm{t}(:,2)./del]
n = [n(:, 1)./del,n(:, 2)./del];
```


## B. 6 influencematrix.m

```
function [M] = influencematrix(zc,t,n, del,N)
```

for $\mathrm{j}=1: \mathrm{N}$
$[\operatorname{Kxx}(:, j), \operatorname{Kxy}(:, j), \operatorname{Kyy}(:, j)]=\ldots$
stressletinfluence $(z c(:, 1), \operatorname{zc}(:, 2), \operatorname{zc}(j,:), \quad t(j,:), \quad n(j,:), \operatorname{del}(j)) ;$
end

```
R2= zc(:, 1).^ 2 + zc(:, 2).^ 2;
a1 = [-zc(:, 2)./R2; zc (:, 1)./R2];
a2 = [-zc(:, 2).* del; zc (:, 1).* del];
rotat = kron(a1, a2');
```

D1 $=$ ones ( $\mathrm{N}, 1) *$ del $^{\prime}$;
$\mathrm{M}=[\mathrm{Kxx}+\mathrm{D} 1 \mathrm{Kxy} ; \mathrm{Kxy} \mathrm{Kyy}+\mathrm{D} 1]+$ rotat;

## B. 7 stressletinfluence.m

```
function [Kxx, Kxy,Kyy] = stressletinfluenceAlt(X,Y,o,t,n,a)
%translate to panel frame
[N,M] = size(X);
X = X - ones(N,M)*o(1);
Y}=\textrm{Y}-\operatorname{ones}(\textrm{N},\textrm{M})*o(2)
%rotate to a coordinate sytem of panel
x}=\textrm{t}(1)*\textrm{X}+\textrm{t}(2)*\textrm{Y}
y}=\textrm{n}(1)*\textrm{X}+\textrm{n}(2)*\textrm{Y}
% shift y slightly above panel to avoid division by zero
% this is equivalent to computing the principle value of the integral on
% the panel itself
y = y+1.e-16;
A1 = atan ((x+(a/2))./y);
A2 = 埥an}((\textrm{x}-(\textrm{a}/2))\cdot/\textrm{y})
B1 = 1./(( a+2*x).^ 2+(2*y).^^2);
B2 = 1./((a-2*x).^ 2+(2*y).^^2);
C1 = 2*(2*x+a).* B1;
C}2=2*(2*\textrm{x}-\textrm{a})\cdot*\textrm{B}2
% influence coefficients in local panel frame
kxx = - (1/(2*\mathbf{pi}))*( -y .*(C1-C2) + (A1-A2));
kxy = - (1/(2*\mathbf{pi}))*(-4*y.^ 2.*(B1-B2) );
kyy = -(1/(2*\mathbf{pi}))*( y .*((C1-C2) +(A1-A2));
% transform back to observer (inertial) frame
Kxx = kxx*t(1)^2 + kyy*n(1)^2 + 2* kxy*n(1)*t(1);
Kxy = kxx*t(1)*t(2) + kyy*n(1)*n(2) + ...
    kxy*(t(1)*n(2)+ n(1)*\textrm{t}(2));
Kyy }= \textrm{kxx}*\textrm{t}(2)^2+\operatorname{kyy}*\textrm{n}(2)^2+2*\textrm{kxy}*\textrm{n}(2)*\textrm{t}(2)
```


## B. 8 vel_ful.m

```
function gdot = vel_fun(A,g)
%g= (x,y,beta)
% A = (u, v, omega)
% differential equations
gdot = [A(1)*\boldsymbol{cos}(\textrm{g}(3))-\textrm{A}(2)*\boldsymbol{\operatorname{sin}}(\textrm{g}(3)),\ldots
    A(1)*\operatorname{sin}(g(3))+A(2)*\operatorname{cos}(g(3)),\ldots
    A(3)];
```


## Appendix C

## Potential flow + vortex shedding

## code

This MATLAB code is an implementation of the swimmer model described in Chapters 6 and 7.

## C. 1 driver.m

```
% driver.m
% Feb 22, 2007 - For vortices about to enter into area too close to
% airfoil, velocity is adjusted to move them parallel to airfoil surface
% Feb 22, 2007 - Revert to point vortices.
% Jun 18, 2007 - Only advance motion in x direction (y and beta fixed)
% Pitching angle of foils is positive in CCW direction above horizontal
% Pitching velocity is positive in CCW direction
% Vorticity strength is position in CW direction
clear all;
warning off MATLAB: divideByZero
inputs; % load user-specified variables in file input.m
global N Vinf k l1 l2 xn yn a area Mass J gammatol thetatol NT upstream ...
        deltatol deltat gammax gammay gammaw Cf TL rho TotalCirc constrained;
initialize;
for k=2:length(time)-1
    text1 = sprintf('Time = %g', time(k)); disp(text1);
    [xp,yp,betap,vxp,vyp,vbetap] = nexttimestepinitialguess(x,y,beta,vx,\ldots
        vy,vbeta);
    % Define body geometry at t(k-1) (known in advance)
    [zc1,t1, n1, del1, I (k-1)] = swimmershape(y1, y2, beta1, beta2,beta, x, y,k-1);
```

```
Fxb}=[\begin{array}{lll}{0}&{0}&{0}\end{array}];\textrm{Fyb}=[\begin{array}{lll}{0}&{0}&{0}\end{array}];\textrm{Mb}=[\begin{array}{lll}{0}&{0}&{0}\end{array}]; % initialize forces to zer
%% Iterate until body motion and boundary condition converge
[theta1,theta2,gamma1,gamma2, delta1,delta2] = initialguess(theta1, ...
        theta2,gamma1,gamma2, delta1, delta2);
i = 1; Fxsolved = 0; Fysolved = 0; Msolved = 0;
while (i <=2|( Fxsolved ==0||Fysolved ==0| Msolved==0))
        x(k)=xp; y (k)=yp; beta (k)=betap;
        % Guess body geometry at t(k) (based on guess or estimate for
        % overall position and orientation)
        [zc2,t2,n2,del2,I(k), xnode1k,ynode1k, xnode2k,ynode 2k] = ...
            swimmershape(y1, y2, beta1, beta2,beta, x,y,k);
        % Boundary condition - velocity using diff from t(k-1) to t(k)
        % (See Sec 6.4.3)
    [Vdn(:,k),Vdt(:, k),Vdx (:, k),Vdy(:,k)] =bodyvelocity(zc1, zc2,n1,t1);
    % if any vortices are within gap around body, move them outside
    % See Sec 6.4.8
    vortexposorig = vortexpos;
    if size(vortexpos, 1)>0
            [vortexpos] = adjustvortexpos(zc2,t2,n2,del2,d,Vdx(:,k),\ldots
                Vdy(:,k), vortexposorig);
    end
    [q,gamma1,gamma2,theta1,theta2,delta1, delta2,D1,D2,E,F,zcwake1, ...
        zcwake2, converged] = flowsolution(zc2,t2,n2,del2, xnode1k,\ldots
        xnode2k, ynode1k,ynode2k, theta1, theta2, delta1, delta2, gamma1,\ldots
        gamma2, Vdn(:,k),vortexpos, vortexstrength);
    % center of mass of swimmer, assuming both foils have same mass
    comx = (cx1k+cx2k)/2; comy = (cy1k+cy2k)/2;
    [cp(:,k), cl1(k), cl2(k),cd1(k),cd2(k),cm001(k),cm002(k),cm251(k), ...
        cm252(k), lift1(k), drag1(k), lift2(k), drag2(k),Moment(k), phi1,\ldots
        phi2, vortin1, vortin2, vortcrossedover, Vt (:, k)] = ...
        computeforces(zc2,t2,n2,del2,D1,D2,E,F,gamma1,gamma2,delta1,...
        delta2,
        beta1, beta2,beta, xnode1k,ynode1k, xnode2k, ynode2k,Vdx (:,k),\ldots
        Vdy(:,k),comx, comy, phi1, phi2, vortin1km1, vortin2km1,\ldots
        vortcrossedover);
    [Fxb}(\textrm{i}+3),\operatorname{Fyb}(\textrm{i}+3),Mb(i+3),xp,yp,betap,vxp,vyp,vbetap]=\ldots
            computemotion(i, drag1,drag2,lift1, lift2,Moment,vxp,vyp,\ldots
            vbetap, Fx, Fy,M, vx, vy,vbeta, x, y,beta, I );
    Fxsolved = (abs (Fxb (i+3)-Fxb(i+2))< Fxtol );
    Fysolved = (abs(Fyb}(\textrm{i}+3)-\textrm{Fyb}(\textrm{i}+2))<\mathrm{ Fytol );
    Msolved = (abs(Mb(i+3)-Mb(i+2))<Mtol);
    if constrained
        Fysolved = 1; Msolved = 1; %
    end
    % if solution is not converged, revert to old vortex positions
    if Fxsolved =0 | Fysolved == 0 | Msolved =0
        vortexpos = vortexposorig;
    end
    i=i +1;
    convergencecheck1
end
convergencecheck
[x(k) y(k) beta(k) vx(k) vy(k) vbeta(k) Fx(k) Fy(k) M(k)] = ...
        updatemotionvars(xp,yp, betap,vxp,vyp, vbetap, Fxb,Fyb,Mb);
% if body is not permitted to self-propel...
if stationary
        x(k)=x(k-1); y(k)=y(k-1); beta (k)= beta(k-1);
```

```
        vx(k) = 0; vy(k) = 0; vbeta (k)=0;
    end
    [vortexpos, vortexstrength, vortexindex] = replacewakepanel(vortexpos,\ldots
    vortexstrength, vortexindex, zcwake1, zcwake2,gamma1,gamma2);
    % identify stagnation point position on trailing foil
    xint(k)= stagpoint(del2,Vt)-12/2;
    [aeff] = angleofattack(y2,beta2,vx,c, aeff);
    if control==1
    [beta2,y2,beta290,y290,y2dot90,beta 2dot90, aeff90,u,u1,u2] = ...
                controller(NF,Wn, control, beta2,y2,y290,y2dot90, beta290,\ldots
                beta2dot90, aeff90,c,vx, xint,a2,b2, phi0,omega1,K,Ki,K2,Ki2,\ldots
                kswitch,switchcontroller,u,u1,u2,time);
    end
    if size(vortexpos,1)>0
        [vortexpos] = convectvortices(vortexpos, vortexstrength, q, ...
            zc2,t2,n2,del2,gamma1,gamma2,d,redirect,x,y,beta,vx,vy,\ldots
            vbeta, y1, y2, beta1, beta2,a,xn,yn);
    end
    % index of vortices at current time step ( k-1 during next time step)
    vortexindexkm1 = vortexindex;
    vortin1km1 = vortin1;
    vortin2km1 = vortin2;
    savedata
    plotandsave
end
```


## C. 2 inputs.m

```
% Specifies if body is fixed (true) or allowed to self-propel (false)
stationary = false;
% Constrain to prevent lateral/rotational motion? (can only translate in x)
% All self-propelling results in thesis constrained motion to only x-dir
constrained = true;
% Determines whether or not to control motion of trailing foil
control = 1;
% TIME DISTRIBUTION
t i = 0; % INITIAL TIME
tf = 40; % END TIME
deltat = 1/40; % 0.025; % TIME STEP SIZE
%
    FLOW PARAMETERS
rho = 1; % fluid density
Vinf =[[11 0}];\quad % freestream velocit
%_SOLVER CONVERGENCE ACCURACY PARAMETERS
thetatol = 0.00001;
deltatol=norm(Vinf)*(deltat (1))/10000;
gammatol = 0.00001;
% PRESCRIBED AIRFOIL MOTION & geometry
N = 100; % # panels per foil
nacanum = 12; % width ratio for NACA 00XX foil
a}=2.5; % horizontal spacing between foi
shift = 0; % vertical distance between foil 1 and 2
c}=1; % chord length
mode = 4; % specifies type of motion to prescribe
omega1 = 7; % frequency of heaving/pitching of foil 1
omega2 = 7; % frequency of heaving/pitching of foil 2
```

```
a1 = 0.05*c; % heaving amplitude of foil 1
a2 = 0.05*c; % heaving amplitude of foil 2
b1 = 9*\mathbf{pi}/180; % pitching amplitude of foil 1 (radians)
b2 = 9*\mathbf{pi/180; % pitching amplitude of foil 2 (radians)}
phase1 = -pi/2; % phase btwn heaving and pitching
phase2 = -pi; % phase btwn leading and trailing foil
% POINT FAR UPSTREAM USED TO COMPUTE POTENTIAL
upstream = [l-30 7}]
% distance about body from which point vortices are diverged
d = 0.03*c;
%determines whether to redirect vortices around body
redirect = true; % true/false
% Viscous drag force parameters
Cf = 0.0146; % dogfish in a flume, EJ Anderson, McGillis, Grosenbaugh, ...
% J Exp Biol, 2000
L}=9; % total length of body - a larger body is imagined to exist..
    % in front of the foils
TL}=2*(L); % total wetted body area
% Tolerance values for iterative loop to converge to forces acting on
% swimmer
Fxtol = 1e-7;
Fytol = Fxtol;
Mtol= Fxtol*100;
% relaxation parameters for estimating forces
gammax = 0.9; gammay = 0.2; gammaw = 0.2;
% Newton solver tolerance
tol = 1e-9;
%-_ CONTROLLER PARAMETERS
Aaeff = 0.6; % estimated amplitude of aeff (found numerically in advance)
Axint = 0.03; % estimated amplitude of xint (found numerically in advance)
zeta = 1; % damping coefficient
wn}=.2;% natural frequency
K}=(4*zeta*wn)/(Axint*Aaeff); % PI controller proportional gain
Ki =wn/(2*zeta); % PI controller integral gain
% flag determines whether to switch controller at time t(kswitch)
% may be used to switch to more aggressive controller after "lock-in"
switchcontroller=1;
% time step at which to switch to new controller
kswitch = 300;
% new controller characteristics
zeta2 = 1.0; % damping coefficient
wn2 = 1.0; % natural frequency
K2 = (4*zeta2*wn2)/(Axint*Aaeff); % PI controller proportional gain
Ki2 = wn/(2*zeta2); % PI controller integral gain
phi0 = pi; % initial phase offset for controlled y2, beta2
% Butterworth low-pass filter parameters
NF}=3; % filter order
Wn}=0.023; % cut-off frequency
% Flags that specify whether to display or save data at each time step
displayflag = 0; % plot flow snapshot?
savedisplayflag}=0;% save image file of snapshot
savefileflag = 0; % save file with all data at end of simulation?
savefilename =','; % file name
dir ='frames1,;' % directory in which to save data and snapshots
filename = 'frame'; % name of snapshot files
```


## C. 3 initialize.m

```
% Initialize variables
% vortex sheets
vortexstrength = [0];
vortexpos = [];
vortexindex = [];
vortin1km1 = [];
vortin2km1 
vortcrossedoverkm1 = [];
time = [0:deltat:tf];
deltat = diff(time);
[y1,y2,beta1,beta2] = prescribedmotion(mode,a1,a2,b1,b2,omega1,omega2,\ldots
    phase1, phase2,time, shift);
% initialize unknown variables in flow computation
theta1 = [beta1(1) beta1 (1)];
theta2 = [beta2(1) beta2(1)];
delta1 = [norm(Vinf)*deltat(1) norm(Vinf)*deltat(1)];
delta2 = [norm(Vinf)*deltat(1) norm(Vinf)*deltat (1)];
gamma1 = [\begin{array}{ll}{0}&{0}\end{array}];
gamma2 = [l0}00]
%% initialize orientation and velocity to zero.
```



```
vx = [0 0 0]; vy = vx; vbeta = vx;
phi1 = zeros(N,1);
phi2 = zeros(N,1);
Fx(1)=0; Fy(1)=0; M(1) = 0;
ltime = length(time);
Vdn=zeros (2*N, ltime); Vdt=zeros(2*N,ltime );
Vdx=zeros(2*N, ltime); Vdy=zeros}(2*N, ltime )
cp=zeros(2*N,ltime);
cl1=zeros(ltime, 1); cl2=zeros(ltime,1);
cd1=zeros(ltime,1); cd2=zeros(ltime,1);
cm001=zeros(ltime,1); cm002=zeros(ltime, 1);
cm251=zeros(ltime,1); cm252=zeros(ltime,1);
lift1=zeros(ltime,1); drag1=zeros(ltime, 1);
lift2=zeros(ltime, 1); drag2=zeros(ltime,1);
Moment=zeros(ltime,1);
Vt=zeros (2*N,ltime);
Fx=zeros(ltime, 1); Fy=zeros(ltime,1); M=zeros(ltime,1);
x=zeros(ltime,1); y=zeros(ltime,1); beta=zeros(ltime,1);
vx=zeros(ltime,1); vy=zeros(ltime,1); vbeta=zeros(ltime, 1);
vortexposx=zeros(2*ltime, ltime-1);
vortexposy=zeros}(2*\mathrm{ ltime, ltime - 1);
vortexstrengthk=zeros(2*ltime);
zcx = zeros(2*N, ltime); zcy = zcx;
% approx # of timesteps per period
NT = round}((2*\mathbf{pi})/(\mathrm{ omega }1*\mathrm{ deltat (1)));
%% define body shape
[xn,yn] = nacacoords(N, nacanum); % defines airfoil geometry
xa1k,ya1k,xa2k,ya2k] = shape(y1 (1),y2(1), beta1(1), beta2(1),a,xn,yn);
[area] = foilarea(xn,yn);
Mass = 2*area*rho;
% defines absolute coordinates of body in inertial reference frame
[zc1,t1,n1, del1, l1, l2, xnode1k,ynode1k, xnode 2k,ynode 2k] = geometry (xa1k,\ldots
        ya1k,xa2k,ya2k,beta(1),x(1),y(1));
[cx1k,cy1k, cx 2k, cy 2k] = centroid (xnode1k, ynode1k, xnode 2k,ynode 2k, area);
J = momentofinertia(xnode1k, ynode1k, cx1k, cy1k);
I(1) = totalinertia(cx1k,cx2k, cy 1k, cy 2k,Mass,J);
aeff=0;
```

| if control | $==1$ |
| ---: | :--- |
| $\quad y 2$ | $=0 * y 2 ;$ |
| $\quad$ beta 2 | $=0 *$ beta $2 ;$ |
| end | $=[] ;$ |
| y290 | $=[] ;$ |
| y2dot 90 | $=[] ;$ |
| beta 290 | $=[] ;$ |
| beta2dot 90 | $=[] ;$ |
| aeff 90 | $=[] ;$ |
| u | $=[] ;$ |
| u1 | $=[] ;$ |

## C. 4 prescribedmotion.m

function $[y 1, y 2$, beta1, beta 2$]=$ prescribedmotion (mode, a1, a $2, \mathrm{~b} 1, \mathrm{~b} 2$, omega1, $\ldots$ omega2, phi1, phi2, time, shift)
\% prescribe airfoil pitching and heaving
\% mode - selects the type of motion
$\%$
\% a $\quad=$ horizontal shift from center of body
\% alpha1 $=$ max amplitude of pitching for airfoil 1
\% alpha2 $=$ max amplitude of pitching for airfoil 2
\% h $\quad=\max$ amplitude of heaving for airfoil 2
\% omega1b $=$ frequency of pitching for airfoil 1
\% omega2b $=$ frequency of pitching for airfoil 1
\% phiy $=$ phase lag between pitching1 and heaving2
\% phib $=$ phase lag between pitching1 and pitching2
\% time $=$ time vector or scalar
\%
switch mode
case 1
$\mathrm{y} 1=0.0+\mathrm{a} 1 * \sin ($ omega $1 *$ time $) ;$
$\mathrm{y} 2=\mathrm{a} 2 * \sin ($ omega $1 *$ time + phi1 $) ;$
beta1 $=\mathrm{b} 1 * \sin ($ omega2 $2 *$ time $)$;
beta $2=\mathrm{b} 2 * \sin ($ omega $2 *$ time + phi2 $) ;$
case 2 \% pang validation heaving case $\mathrm{y} 1=\mathrm{a} 1 * \sin ($ omega1*time $)$;
$\mathrm{y} 2=-1+\mathrm{a} 2 * \sin ($ omega $1 *$ time + phi1 $) ;$
\% y2 $=-1+\ldots$.
beta $1=\mathrm{b} 1 * \sin ($ omega $2 *$ time $) ;$
beta2 $=\mathrm{b} 2 * \sin ($ omega $2 *$ time + phi2 $)$;
case $3 \%$ not shifted down in $y$ direction (no -1) $\mathrm{y} 1=\mathrm{a} 1 * \sin ($ omega $1 *$ time + phi1 $) \cdot *(1-\exp (-1 *$ time $)) ;$ $\mathrm{y} 2=\mathrm{a} 2 * \sin ($ omega1 $*$ time + phi $2+\mathrm{phi} 1) \cdot *(1-\exp (-1 *$ time $)) ;$ beta $1=\mathrm{b} 1 * \sin ($ omega $2 *$ time $) \cdot *(1-\exp (-1 *$ time $))$; beta $2=\mathrm{b} 2 * \sin ($ omega $2 *$ time + phi 2$) \cdot *(1-\exp (-1 *$ time $)) ;$ case $4 \%$ shifted in $y$ direction $\mathrm{y} 1=\mathrm{a} 1 * \sin ($ omega1 $1 *$ time + phi1 $) \cdot *(1-\exp (-1 *$ time $)) ;$ $\mathrm{y} 2=\mathrm{shift}+\mathrm{a} 2 * \sin ($ omega $1 * \mathrm{time}+\mathrm{phi} 2+\mathrm{phi} 1) . *(1-\exp (-1 * \mathrm{time})) ;$ beta1 $=\mathrm{b} 1 * \sin ($ omega $2 *$ time $) \cdot *(1-\exp (-1 *$ time $))$; beta $2=\mathrm{b} 2 * \sin ($ omega $2 *$ time + phi2 $) . *(1-\exp (-1 *$ time $)) ;$
case $5 \%$ shifted in $y$ direction, no exponential buildup in motion $\mathrm{y} 1=\mathrm{a} 1 * \sin ($ omega $1 *$ time + phi1 $) ;$ $\mathrm{y} 2=\mathrm{shift}+\mathrm{a} 2 * \sin ($ omega1 $*$ time + phi $2+$ phi1 $) ;$ beta1 $=\mathrm{b} 1 * \sin ($ omega2 2 time $)$; beta $2=\mathrm{b} 2 * \sin ($ omega $2 *$ time + phi2 $) ;$ case 6
\% FIXED ANGLE OF ATTACK $\mathrm{y} 1=0+$ time $* 0$;
beta1 $=\mathrm{b} 1+\operatorname{time} * 0$;
$\mathrm{y} 2=\mathrm{shift}+$ time $* 0$;
beta $2=\mathrm{b} 2+\mathrm{time} * 0$;
end

## C. 5 nacacoords.m

```
    function [xn,yn] = nacacoords(nodtot, nacanum)
    % ____OUTPUTS
% xm, ym: control points
% x, y : node points
% sinthe, costhe: panel angles
%ds: panel length
nlower = nodtot/2;
nupper = nodtot-nlower;
np1 = nodtot+1;
tau=nacanum / 100;
npoint = nlower;
nstart = 0;
% loop over lower surface
for n=1:npoint
    z = (1+\mathbf{cos}(\mathbf{pi}*(n-1)/npoint))/2;
    i= nstart + n;
    % compute thickness, camber and angular location of an airfoil point
    thick(i) = tau*5*(.2969*z.^.5 - . 1260*z - . 3537*z.^2 + ...
            .2843*z.^ 3 - . 1015*z.^4);
        camber(i) = 0;
        dcamdx(i) = 0;
        beta(i)=\boldsymbol{atan}(dcamdx(i ));
        xn(i) = z-(-1)*thick(i)*sin(beta(i ));
        yn(i)}=\operatorname{camber(i)}+(-1)*\operatorname{thick}(i)*\boldsymbol{cos}(\operatorname{beta}(\textrm{i}))
end
% initialize indexing for upper surface
npoint = nupper;
nstart = nlower;
% loop over upper surface
for n=1:npoint
    z = (1-cos (pi*(n-1)/npoint))/2;
    i= nstart + n;
    % compute thickness, camber and angular location of an airfoil point
    thick(i) = tau*5*(.2969*z.^.5 - . 1260*z - . 3537*z.^2 + ...
            2843*z.^3 - . 1015*z.^4);
        camber(i) = 0;
        dcamdx(i) = 0;
    \boldsymbol{beta}(i)=\boldsymbol{\operatorname{tan}}(\operatorname{dcamdx}(\textrm{i}));
    xn(i) = z-(1)*thick(i)*sin(beta(i));
    yn(i)}=\mathrm{ camber(i) +(1)*thick(i)*cos(beta(i));
end
xn(np1) = xn(1);
% shift to make quarter cord point at origin
xn = xn' - 0.25;
yn(np1) = yn(1);
yn = yn';
```


## C. 6 shape.m

```
function [xa1,ya1,xa2,ya2] = shape(y1,y2,beta1, beta2,a,xn,yn)
% Returns the coordinates for the leading (#1) and trailing (#2) foils of
% the swimmer shape
%
% ———— INPUTS
% xn, yn : node coordinates of airfoil at origin with zero rotation
% beta1,2 : orientation of airfoil wrt horizontal, positive is CW
% below positive horizontal axis
%y1,2 : vertical position of airfoil
% a : distance between front and rear foil
nodtot = length(xn)-1;
xnrot1 = zeros(nodtot +1,1);
xnrot2 = zeros(nodtot +1,1)
ynrot1 = zeros(nodtot+1,1)
ynrot2 = zeros(nodtot +1,1)
% rotate node points about quarter chord point
for i=1:nodtot+1
    xnrot1(i) = xn(i)* cos(beta1) + yn(i)*sin(beta1);
    ynrot1(i) = -xn(i)*sin(beta1) + yn(i)*\operatorname{cos}(beta1);
    xnrot2(i) = xn(i)*\boldsymbol{cos(beta2) + yn(i)*sin(beta2);}
    ynrot2(i)}=-xn(i)*\operatorname{sin}(beta2)+yn(i)*\boldsymbol{cos}(beta2)
end
% shape relative to body frame
xa1 = xnrot1 - a/2;
ya1 = ynrot1 + y1;
xa2 = xnrot2 + a/2;
ya2 = ynrot2 + y2;
```


## C. 7 foilarea.m

```
function [area] = foilarea(xn,yn)
% computes area (and hence, mass) of airfoils
area = 0;
for i=1:length(xn)-1
    area}=\mathrm{ area - 0.5*(xn(i)*yn(i+1)-xn(i+1)*yn(i));
end
```


## C. 8 geometry.m

```
function [zc,t,n, del, l1, l2, xnode1,ynode1, xnode2,ynode2] = ...
    geometry(xa1, ya1, xa2,ya2,beta, x, y)
% Computes the body geometry
nodtot = length(xa1)-1;
% initialize, preallocate memory
xr1 = zeros(nodtot+1,1);
yr1 = xr1;
xr2 = xr1
yr2 = xr1;
xm1 = zeros(nodtot,1);
ym1 = xm1;
xm2 = xm1;
ym2 = xm1;
sinthe1 = xm1;
costhe1 = xm1;
sinthe2 = xm1;
costhe2 = xm1;
```

```
del1 = xm1;
del2 = xm1;
% move node points to absolute position and orientation in space
% rotate node point by angle beta
cb = \boldsymbol{cos(beta); sb = sin(beta);}
for i=1:nodtot+1
    xr1(i) = xa1(i)*cb + ya1(i)*sb;
    yr1(i) = -xa1(i)*sb + ya1(i)*cb;
    xr2(i) = xa2(i)*cb + ya2(i)*sb;
    yr2(i)}=-\textrm{xa}2(\textrm{i})*\textrm{sb}+\textrm{ya}2(\textrm{i})*\textrm{cb}
end
% translate node points
xa1 = xr1 + x;
ya1 = yr1 + y;
xa2 = xr2 + x;
ya2 = yr2 + y;
xnode1 = xa1;
ynode1 = ya1;
xnode2 = xa2;
ynode2 = ya2;
% compute slope of panel and arc length of airfoil skin
for i=1:nodtot
        % AIRFOIL 1 %%%%0%%%%%%%%%%%%%%%%%%
        % control points
        xm1(i) = (xa1(i+1)+xa1(i))/2;
        ym1(i) = (ya1(i+1)+ya1(i))/2;
        % arc length
        dx1 = xa1(i+1)-xa1(i);
        dy1 = ya1(i+1)-ya1(i);
        del1(i) = sqrt(dx1^2+dy1^2);
        % slope
        sinthe1(i) = dy1/del1(i);
        costhe1(i) = dx1/del1(i);
        % AIRFOIL 2 %%%%%%%%%%%%%%%%%%%%%
        % control points
        xm2(i) = (xa2(i+1)+xa2(i))/2;
        ym2(i) = (ya2(i+1)+ya2(i))/2;
        % arc length
        dx2 = xa2(i+1)-xa2(i);
        dy2 = ya2(i+1)-ya2(i);
        del2(i) = sqrt(dx2^2+dy2^2);
        % slope
        sinthe2(i) = dy2/del2(i);
        costhe2(i) = dx2/del2(i);
end
del1 = del1;
xm1 = xm1;
ym1 = ym1;
del2 = del2;
xm2 = xm2;
ym2 = ym2;
% normal and tangent vectors at each panel
t1 = [ costhe1 sinthe1];
n1 = [-sinthe1 costhe1];
t2 = [ costhe2 sinthe2];
n2 = [-sinthe2 costhe2];
```

```
l1 = sum(del1); 12=sum(del2 );
zc1 = [xm1 ym1];
zc2 = [xm2 ym2];
zc}=[\textrm{zc}1;\textrm{zc}2]
t = [t1; t2];
n = [n1;n2];
del=[del1;del2];
```


## C. 9 centroid.m

```
function [cx1,cy1,cx2,cy2] = centroid(xnode1,ynode1,xnode2,ynode2, area)
% computes the centroid of both foils in inertial coordinate frame
% initialize
cx1 = 0; cy1 = 0;
cx2 = 0; cy2 = 0;
for i=1:length(xnode1)-1
    cx1 = cx1 - (1/(6* area ) )*(( xnode1 (i)+xnode1 (i +1)) *...
        (xnode1(i)*ynode1(i+1)-xnode1(i+1)*ynode1(i )));
    cy1 = cy1 - (1/(6* area ) )*(( ynode1 (i)+ynode1 (i +1)) *...
        (xnode1(i)*ynode1(i+1)-xnode1(i+1)*ynode1(i )));
    cx2 = cx2 - (1/(6* area) )*((xnode2(i)+xnode2(i + 1))*...
        (xnode2(i)*ynode2(i+1)-xnode2(i+1)*ynode2(i )));
    cy2 = cy2 - (1/(6* area ))*((ynode2(i)+ynode2(i +1))*...
        (xnode2(i)*ynode2(i+1)-xnode2(i+1)*ynode2(i )));
end
```


## C. 10 momentofinertia.m

```
function J = momentofinertia(xnode1, ynode1, cx1, cy1);
```

\% compute moment of inertia of one polygon, given node coordinates and
\% center of mass coordinates
$\mathrm{J}=0$;
for $\mathrm{i}=1$ :length (xnode1) -1
yip1 = ynode1 $(\mathrm{i}+1)-\mathrm{cy} 1$;
yi $=$ ynode1 (i) $-\operatorname{cy1}$;
xip1 $=$ xnode1 $(i+1)-c x 1 ;$
$\mathrm{xi}=\operatorname{xnode1}(\mathrm{i})-\mathrm{cx} 1$;
$\mathrm{J}=\mathrm{J}+0.5 *(1 / 12) *\left((\mathrm{yip} 1-\mathrm{yi}) *(\mathrm{xip} 1+\mathrm{xi}) *\left(\mathrm{xip} 1^{\wedge} 2+\mathrm{xi}^{\wedge} 2\right) \ldots\right.$
$\left.-(\operatorname{xip} 1-x i) *(y i p 1+y i) *\left(y i p 1^{\wedge} 2+y i^{\wedge} 2\right)\right)$;
end
\% to make value positive (may be -ive, depending on orientation of nodes)
$\mathrm{J}=\mathbf{a b s}(\mathrm{J})$;

## C. 11 totalinertia.m

```
function I = totalinertia(cx1, cx2, cy1,cy2,Mass,J)
% Computes the moment of inertia of the swimmer
% center of mass of entire swimmer, assuming both have same mass
comx = (cx1+cx2)/2;
comy = (cy1+cy2)/2;
r1 = [cx1-comx, cy1-comy ];
r2 = [cx2-comx, cy2-comy];
I}=\textrm{J}+0.5*\operatorname{Mass}*(\boldsymbol{norm}(\textrm{r}1\mp@subsup{)}{}{\wedge}2)+J + 0.5*Mass*(norm(r2 )^2)
```


## C. 12 nexttimestepinitialguess.m

```
function [xp,yp, betap, vxp,vyp,vbetap] = ...
    nexttimestepinitialguess(x,y,beta,vx,vy, vbeta)
% Use previous time step values to guess position, orientation and
% velocities of the swimmer at the current time step
global k deltat
% initial guess for next time step
% xp should converge to x(k)
xp }=\textrm{x}(\textrm{k}-1)\quad+\textrm{vx}(\textrm{k}-1)*\operatorname{deltat}(\textrm{k})
yp = y(k-1) + vy(k-1)*deltat(k);
betap = beta(k-1)+ vbeta(k-1)*deltat (k);
vxp = vx(k-1);
vyp = vy(k-1);
vbetap = vbeta (k-1);
```


## C. 13 swimmershape.m

```
function [zc1,t1,n1, del1,I,xnode1,ynode1, xnode2,ynode2] = ...
    swimmershape(y1, y2, beta1, beta2, beta, x,y,tstep)
% Returns the control points (zc1), normal (n1) and tangent (t1) vectors at
% the control points and panel lengths (del1) for the swimmer given the
% internal shape variables y1, y2, beta1, beta2, the position (x,y),
% orientation (beta), &% the current time step (tstep)
global a xn yn area Mass J
%%% Define Body and geometry at t(tstep)
% body shape at t(tstep)
[xa1,ya1,xa2,ya2] =
    shape(y1(tstep),y2(tstep),beta1(tstep),beta2(tstep),a, xn, yn);
% geometry at time t(tstep)
[zc1,t1,n1, del1, l1, l2, xnode1,ynode1,xnode2,ynode2] =...
    geometry(xa1, ya1, xa2, ya2,beta(tstep ),x(tstep),y(tstep ));
[cx1, cy1, cx2, cy2] = centroid(xnode1, ynode1, xnode2, ynode2, area);
I = totalinertia(cx1, cx2,cy1, cy2,Mass,J);
```


## C. 14 initialguess.m

function [theta1, theta 2 , gamma1, gamma2, delta1, delta2] $=\ldots$
initialguess (theta1, theta2, gamma1, gamma2, delta1, delta2);
\% initial guess for new time step is value from previous time step
global k

| $\operatorname{theta} 1(k)$ | $=\operatorname{theta} 1(k-1) ;$ |  | delta1 $(k)=\operatorname{delta} 1(k-1) ;$ |
| ---: | :--- | ---: | :--- |
| theta2 $(k)$ | $=\operatorname{theta} 2(k-1) ;$ |  | delta2 $(k)=\operatorname{delta} 2(k-1) ;$ |
| $\operatorname{gamma}(k)$ | $=\operatorname{gamma}(k-1) ;$ |  | $\operatorname{gamma} 2(k)=\operatorname{gamma} 2(k-1) ;$ |

## C. 15 bodyvelocity.m

```
function [Vdn,Vdt,Vdx,Vdy]= bodyvelocity(zc1, zc2,n,t);
global k deltat
% Compute velocity of body boundary (See Section 6.4.3)
% compute boundary condition
Vdx = (zc2(:,1)-zc1(:,1))/ deltat(k); % See Eqn 6.14
Vdy = (zc2(:,2)-zc1(:,2))/ deltat(k); % See Eqn 6.15
```

```
Vdn = Vdx.*n(:,1)+Vdy.*n(:, 2);
Vdt = Vdx.*t(:,1)+Vdy.*t(:, 2);
```


## C. 16 flowsolution.m

```
function [q,gamma1,gamma2,theta1,theta2,delta1, delta2,D1,D2,E,F,zcwake1,\ldots.
    zcwake2, converged] = flowsolution(zc,t,n,del, xnode1, xnode2,ynode1,\ldots
    ynode2,theta1,theta2, delta1, delta2,gamma1,gamma2,Vdn, vortexpos,...
    vortexstrength)
% Computes the source and vorticity distribution over two bodies & wake
% panel lengths and orientations given a specified body velocity
global N Vinf k l1 12 gammatol thetatol deltatol deltat NT;
% INITIAL GUESS FOR FLOW VARIABLES TO BE SOLVED FOR
gamma1new = gamma1(k); gamma2new = gamma2(k);
theta1new = theta1(k); theta2new = theta2(k);
delta1new = delta1(k); delta2new = delta2(k);
% initialize error values to zero
errgamma1 = 0; errgamma2 = 0;
errtheta1 = 0; errtheta2 = 0;
errdelta1 = 0; errdelta2 = 0;
```

[Ans, Ats, Axs, Ays, Bnv, Btv, Bxv, Byv] = panelinfluence(zc, $\mathrm{t}, \mathrm{n}, \mathrm{del}, \mathrm{zc})$;
\% influence coefficients due to circulation on airfoil 1
$\operatorname{Bnv} 1=\operatorname{Bnv}(:, 1: N) * \operatorname{ones}(N, 1)$;
Btv1 $=\operatorname{Btv}(:, 1: N) * \operatorname{ones}(N, 1) ;$
\% influence coefficients due to circulation on airfoil 2
$\operatorname{Bnv} 2=\operatorname{Bnv}(:, N+1: 2 * N) *$ ones (N, 1$) ;$
Btv2 $=\operatorname{Btv}(:, \mathrm{N}+1: 2 * \mathrm{~N}) *$ ones $(\mathrm{N}, 1) ;$
if $\operatorname{size}($ vortexpos, 1$)>0$
zcsize $=\operatorname{size}(z c, 1)$;
zcbody $=\mathrm{zc}(1: \mathrm{zcsize} / 2,:) ; \quad$ zcbody $2=\mathrm{zc}(\operatorname{zcsize} / 2+1: \mathrm{zcsize},:) ;$
tbody $1=\mathrm{t}(1: \operatorname{zcsize} / 2,:) ; \quad \operatorname{tbody} 2=\mathrm{t}(\operatorname{zcsize} / 2+1: \mathrm{zcsize},:) ;$
nbody $=\mathrm{n}(1: \operatorname{zcsize} / 2,:) ; \quad$ nbody $2=\mathrm{n}(\operatorname{zcsize} / 2+1: \mathrm{zcsize},:) ;$
\% vortices acting on body 1 will be considered point vortices
$[\mathrm{Cx} 1, \mathrm{Cy} 1, \mathrm{Cn} 1, \mathrm{Ct} 1]=\mathrm{ptvortinfluence}($ vortexpos, $\operatorname{zcbody} 1, \operatorname{tbody} 1, \operatorname{nbody} 1,1) ;$
\% vortices shed by body 1 act on body 2 as if they are vortex blobs
\% vortices shed by body 2 act on body 2 as if they are point vortices
$[\mathrm{Cx} 2 \mathrm{a}, \mathrm{Cy} 2 \mathrm{a}, \mathrm{Cn} 2 \mathrm{a}, \mathrm{Ct} 2 \mathrm{a}]=$ ptvortinfluence (vortexpos, zcbody 2, tbody $2, \ldots$
nbody2,4);
$[\mathrm{Cx} 2 \mathrm{~b}, \mathrm{Cy} 2 \mathrm{~b}, \mathrm{Cn} 2 \mathrm{~b}, \mathrm{Ct} 2 \mathrm{~b}]=\mathrm{ptvortinfluence}($ vortexpos, zcbody $2, \operatorname{tbody} 2, \ldots$
nbody2,1);
Multa $=\operatorname{repmat}([$ ones $(\mathrm{N}, 1) \quad \operatorname{zeros}(\mathrm{N}, 1)], 1, \mathrm{k}-2)$;
Multb $=\operatorname{repmat}([\operatorname{zeros}(\mathrm{N}, 1)$ ones $(\mathrm{N}, 1)], 1, \mathrm{k}-2)$;
Cx2 = Cx2a.*Multa+Cx2b.*Multb;
$\mathrm{Cy} 2=\mathrm{Cy} 2 \mathrm{a} \cdot *$ Multa+Cy2b.*Multb;
Cn2 $=$ Cn2a.*Multa+Cn2b.*Multb;
$\mathrm{Ct} 2=\mathrm{Ct} 2 \mathrm{a} . *$ Multa+Ct2b.*Multb;
$\mathrm{Cx}=[\mathrm{Cx} 1 ; \mathrm{Cx} 2]$;
$\mathrm{Cy}=[\mathrm{Cy} 1 ; \mathrm{Cy} 2]$;
$\mathrm{Cn}=[\mathrm{Cn} 1 ; \mathrm{Cn} 2]$;
$\mathrm{Ct}=[\mathrm{Ct} 1 ; \mathrm{Ct} 2]$;
else
$\mathrm{Cn}=0 ; \quad \mathrm{Ct}=0 ;$
end
$\operatorname{invAns}=\operatorname{inv}($ Ans $) ;$
firstrun $=1 ; \%$ ensures that loop will run at least once

```
counter = 0;
```

while abs (errgamma1) > gammatol \| abs (errgamma2) >gammatol \| $\|$.
$\operatorname{abs}($ errtheta 1$)>$ thetatol $\| \mathbf{a b s}($ errtheta 2$)>$ thetatol $\| \ldots$
$\mathbf{a b s}($ errdelta 1$)>$ deltatol $\| \mathbf{a b s}($ errdelta 2$)>$ deltatol $\|$...
firstrun $=1$
gamma1(k) $=$ gamma1new; gamma2(k) = gamma2new ;
theta1 $(k)=$ theta1new ; theta2 $(k)=$ theta2new;
delta1 (k) $=$ delta1new; delta2 $(k)=$ delta2new;
counter $=$ counter +1 ;
\% if solution is unable to converge within specified number of
\% iterations, break out of function with flag converged $=0$
if counter $>1000$
converged $=2$;
disp('wake panel parameters not converged');
return
end
firstrun $=0$;
\% Source distribution as function of circulation values
\% Ans*q $=Q 1 * g 1+Q 2 * g 2+Q 3$
$\% q=B 1 * g 1+B 2 * g 2+C$
\% wake panel influence on body panels
[zcwake1, tx1, ty $1, \operatorname{nx} 1$, ny1] = wakepanel (xnode1, ynode1, theta1, delta1);
[zcwake2,tx2,ty2, nx2, ny2] = wakepanel(xnode2,ynode2, theta 2 , delta 2 );
$[\mathrm{X}, \mathrm{X}, \mathrm{X}, \mathrm{X}, \mathrm{X}, \mathrm{X}, \mathrm{B} w x v 1$, Bwyv1] $=$ panelinfluence (zcwake1, $[\mathrm{tx} 1$ ty 1$], \ldots$
[nx1 ny1], delta1 (k), zc);
$[X, X, X, X, X, X, B w x v 2, B w y v 2]=$ panelinfluence (zcwake2, [tx2 ty2],...
[nx2 ny2], delta2(k), zc);
Bwnv1 = Bwxv1.*n (:, 1) + Bwyv1.*n $(:, 2)$;
Bwnv2 $=$ Bwxv2.*n $(:, 1)+$ Bwyv2.*n $(:, 2) ;$
Bwtv1 $=$ Bwxv1.*t $(:, 1)+$ Bwyv1.*t $(:, 2)$;
Bwtv2 $=$ Bwxv2.*t $(:, 1)+\operatorname{Bwyv2} \cdot * \mathrm{t}(:, 2) ;$
$\mathrm{B} 1=\operatorname{invAns} *((\operatorname{Bwnv} 1 * \mathrm{l} 1) /$ delta1 $(\mathrm{k})-\operatorname{Bnv} 1)$;
$\mathrm{B} 2=\operatorname{invAns} *((\operatorname{Bwnv} 2 * \mathrm{l} 2) /$ delta2 $(\mathrm{k})-\operatorname{Bnv} 2) ;$
$\mathrm{C}=\operatorname{invAns} *(-(\operatorname{Bwnv} 1 * \operatorname{gamma}(\mathrm{k}-1) * \mathrm{l} 1) / \operatorname{delta} 1(\mathrm{k})-\ldots$
(Bwnv2*gamma2 $(\mathrm{k}-1) * \mathrm{l} 2) /$ delta2 $(\mathrm{k})-\mathrm{Cn} *$ vortexstrength $\left.\quad-\mathrm{n} * \operatorname{Vinf}^{\prime}+\mathrm{Vdn}\right)$;
op\%
\% TANGENT VELOCITY Vt $=$ D1*g1 + D2*g2 $+E$ (See Eqn 6.25/6.26/6.27)
$\mathrm{D} 1=\mathrm{Ats} * \mathrm{~B} 1+\mathrm{Btv} 1-(\mathrm{B} w t v 1 * \mathrm{l} 1) / \mathrm{delta} 1(\mathrm{k})$;
$\mathrm{D} 2=\mathrm{Ats} * \mathrm{~B} 2+\mathrm{Btv} 2-(\mathrm{Bwtv} 2 * 12) / \mathrm{delta} 2(\mathrm{k})$;
$\mathrm{E}=\operatorname{Ats} * \mathrm{C}+($ Butv1*gamma1 $(\mathrm{k}-1) * \mathrm{l} 1) /$ delta1 $(\mathrm{k})+\ldots$
$(\operatorname{Bwtv} 2 * \operatorname{gamma} 2(\mathrm{k}-1) * \mathrm{l} 2) / \operatorname{delta} 2(\mathrm{k})+\mathrm{Ct} *$ vortexstrength${ }^{\prime}+\mathrm{t} * \operatorname{Vinf}{ }^{\prime}$;
\% NORMAL VELOCITY $F=V d n$
$\mathrm{F}=\mathrm{Vdn}$;


```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% IMPOSE KUTTA CONDITION TO GET TWO NONLINEAR EQUATIONS IN GAMMA1,2
% a1*g1^2 + b1*g2^2 + c1*g1*g2 + d1*g1 + e1*g2 + f1 = 0, where
% (See Sec 6.4.4 छGEqns 6.33/6.34/6.35/6.36)
a1 = D1(1)^2 - D1(N)^2;
b}1=\textrm{D}2(1)^2 - D2(N)^2
c1 = 2*(D1 (1)*D2(1)-D1(N)*D2(N));
d1 = 2*(D1(1)*E(1)-D1(N)*E(N)-l1/deltat (k));
e}1=2*(\textrm{D}2(1)*\textrm{E}(1)-\textrm{D}2(\textrm{N})*\textrm{E}(\textrm{N}))
f1 = E(1)^2-E(N)^2+(2*gamma1 (k-1)*l1)/deltat (k) + F (1)^2-F(N)^ 2;
a2 = D1 (N+1)^2 - D1 (2*N )^ 2;
b2 = D2(N+1)^2 - D2(2*N)^}2
c2 = 2*(D1 (N+1)*D2(N+1)-D1 (2*N)*D2(2*N));
d2 = 2*(D1 (N+1)*E(N+1)-D1(2*N)*E(2*N));
e2 = 2*(D2(N+1)*E(N+1)-D2(2*N)*E(2*N)-12/deltat (k));
f2= E(N+1)^2-E (2*N)^}2+(2*\operatorname{gamma}2(\textrm{k}-1)*\textrm{l}2)/\textrm{deltat}(\textrm{k})+\textrm{F}(\textrm{N}+1\mp@subsup{)}{}{\wedge}2-\textrm{F}(2*\textrm{N}\mp@subsup{)}{}{\wedge}2
```

```
gamma1o = gamma1(k);
gamma2o = gamma2(k);
gamma1onew = gamma1o;
gamma2onew = gamma2o;
firstrun2 = 1; % to ensure that while loops runs at least once
iter2 = 0;
while (abs(gamma1o-gamma1onew)>gammatol)||..
    (abs(gamma2o-gamma2onew)}>\mathrm{ gammatol )| |firstrun2==1
    gamma1o = gamma1onew ;
    gamma2o = gamma2onew;
    % (See Eqns 6.38/6.39)
    q1 = 2*a1*gamma1o + c1*gamma2o + d1;
    q}2=2*\textrm{b}1*\mathrm{ gamma2o + c1*gamma1o + e1;
    q}3=\textrm{a}1*\mathrm{ gamma1o^2 + b1*gamma2o^2 + c1*gamma1o*gamma2o + ...
        d}1*\mathrm{ gamma1o + e1*gamma2o + f1;
    q4 = 2*a2*gamma1o + c 2*gamma2o + d2;
    q}5=2*\textrm{b}2*\mathrm{ gamma2o + c 2*gamma1o + e2;
    q6 = a 2*gamma1o^2 + b 2*gamma2o^2 + c 2*gamma1o*gamma2o + ...
        d}2*\mathrm{ gammalo + e2*gamma2o + f2;
    % solve set of equations for dg1, dg2
    %dg=-inv([q1 q2;q4 q5])*[q3;q6]; % the line below is equivalent,
    % but faster
    dg = -[q41 q2;q4 q5 ] \[q3;q6];
    % update circulation strength values - See Eqn 6.37
    gamma1onew = gamma1o + dg(1);
    gamma2onew = gamma2o + dg(2);
    firstrun2 = 0;
    iter2 = iter2 + 1;
    if iter2 > 100 % checks if Newton solver not converging to solution
        converged = 4;
        disp('Newton solver not converged')
        return
    end
end
gamma1new = gamma1onew;
gamma2new = gamma2onew;
%0%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% COMPUTE SOURCE DISTRIBUTION - See Eqn 6.10
q = B1*gamma1new + B2*gamma2new + C;
% source distribution with Cnv (assuming no wake vortices)
% qnv = B1*gamma1new + B2*gamma2new + Cnv;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
$0%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% NOW, RE-COMPUTE VELOCITY AT TRAILING EDGE WAKE PANELS
% influence on wake panel due to source and vort dist on two bodies
    [X,X,Axsw1, Aysw1,X,X, Bxvw1, Byvw1] = panelinfluence(zc,t,n,del, zcwake1);
    [X,X,Axsw2,Aysw2,X,X, Bxvw2,Byvw2] = panelinfluence(zc,t,n,del, zcwake2);
% influence on wake panel 1 due to wake panel 2
[X,X,X,X,X,X, Bxvw12, Byvw12] = panelinfluence(zcwake2,[tx2 ty2],\ldots
    [nx2 ny2], delta2(k), zcwake1);
% influence on wake panel 2 due to wake panel 1
[X,X,X,X,X,X, Bxvw21, Byvw21] = panelinfluence(zcwake1,[tx1 ty1],\ldots
    [nx1 ny1], delta1(k),zcwake2);
if size(vortexpos, 1)>0
            [Cxw1,Cyw1] = ptvortinfluence(vortexpos, zcwake1,[tx1 ty1],\ldots
            [nx1 ny1],1);
            [Cxw2,Cyw2] = ptvortinfluence(vortexpos, zcwake2,[tx2 ty2],\ldots.
                [nx2 ny2],5); % (Note mode=5, not 4)
else
    Cxw1=0;Cyw1=0;
```

```
    Cxw2=0;Cyw2=0
    end
    % Wake panel circulation: (See Eqns 6.3/6.4)
    gammaw1 = ((gamma1 (k-1)-gamma1new)*l1)/delta1 (k);
    gammaw2 = ((gamma2(k-1)-gamma2new)*l2)/delta2 (k);
    % Velocity on wake panel 1
    Vx1 = Axsw1*q + (Bxvw1 (1:N)*ones (N,1))*gamma1new + ...
    (Bxvw1(N+1:2*N)* ones (N,1))*gamma2new + Bxvw 12*gammaw2 + ...
    Cxw1*vortexstrength' + Vinf(1);
Vy1 = Aysw1*q + (Byvw1 (1:N)*ones (N,1))*gamma1new + ...
    (Byvw1(N+1:2*N)* ones (N,1))*gamma2new + Byvw12*gammaw2 + ...
    Cyw1*vortexstrength' + Vinf(2);
Vx2 = Axsw2*q + (Bxvw2(1:N)*ones (N,1))*gamma1new + ...
    (Bxvw2 (N+1:2*N)* ones (N,1))*gamma2new + Bxvw21*gammaw1 + ...
    Cxw2*vortexstrength' + Vinf(1);
Vy2 = Aysw2*q + (Byvw2 (1:N)* ones (N,1))*gamma1new + ...
    (Byvw2(N+1:2*N)* ones (N,1))*gamma2new + Byvw21*gammaw1 + ...
    Cyw2*vortexstrength' + Vinf(2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% COMPUTE NEW PANEL LENGTH AND ORIENTATION (positive is below
% horizontal, CW, negative is above horizontal, CCW)
theta1new =-atan2(Vy1,Vx1); % See Eqn 6.6
theta2new =-atan2(Vy2,Vx2); % See Eqn 6.6
delta1new = norm(Vx1,Vy1)*deltat(k); % See Eqn 6.5
delta2new = norm(Vx2,Vy2)*deltat(k); % See Eqn 6.5
% determine if new wake panel lies inside body
[zcwake2t,tx2t,ty2t, nx2t, ny2t] = wakepanel(xnode2,ynode 2,\ldots
    [theta2(1:k-1) theta2new],[delta2(1:k-1) delta2new]);
in = inpolygon(zcwake2t(1), zcwake2t(2), xnode2, ynode2);
% set lower limit on delta so wake pane does not come too close to
% trailing edge (to avoid numerical difficulties)
delta2newMin = mean(delta1 (max (1,k-2*NT):k)) - ...
    (max(delta1 (max (1,k-2*NT):k)) - min(delta1 (max (1,k-2*NT):k )) )/ 2;
delta1newMin = mean(delta2 (max (1,k-2*NT):k ) ) - ...
    (max( delta2 (max (1,k-2*NT):k)) - min(delta2 (max (1,k-2*NT):k)) )/ 2;
delta2newMin = deltat(1);
if delta2new<delta2newMin
    delta2new=delta2newMin;
end
if delta1new<delta1newMin
    delta1new=delta1newMin;
end
errgamma1 = gamma1(k) - gamma1new;
errgamma2 = gamma2(k) - gamma2new;
errtheta1 = theta1(k) - theta1new;
errtheta2 = theta2(k) - theta2new;
errdelta1 = delta1(k) - delta1new;
errdelta2 = delta2(k) - delta2new;
end
converged = 1;
```


## C. 17 panelinfluence.m



```
%
% zc position of collocation pts
% zcpts points at which to compute influence coefficients
%t components of vectors tangent to panels
% n components of outward normal vectors
% if zcpts are not zc, then t, n are with respect to a horizontal
% del panel length
%
% zc, t and n are w.r.t inertial frame
%
```



```
%_OUTPUT 
coordinates of control pt i (Ci) relative 
```

```
sizezc = size(zc);
```

sizezc = size(zc);
sizept = size(pt);
sizept = size(pt);
M}=\mathbf{size}(zc,1)
M}=\mathbf{size}(zc,1)
N}=\operatorname{size}(\textrm{pt},1)
N}=\operatorname{size}(\textrm{pt},1)
% initialize (all have same dimensions)
% initialize (all have same dimensions)
Xrel = zeros(N,M); Yrel = zeros(N,M);
Xrel = zeros(N,M); Yrel = zeros(N,M);
Nx = Xrel; Ny = Xrel;
Nx = Xrel; Ny = Xrel;
Tx = Xrel; Ty = Xrel;
Tx = Xrel; Ty = Xrel;
D = Xrel; D2 = Xrel;
D = Xrel; D2 = Xrel;
Cn = Xrel; Ct = Xrel;
Cn = Xrel; Ct = Xrel;
Vn = Xrel; Vt = Xrel;
Vn = Xrel; Vt = Xrel;
Axs = Xrel; Ays = Xrel;
Axs = Xrel; Ays = Xrel;
Ans = Xrel; Ats = Xrel;
Ans = Xrel; Ats = Xrel;
Vtv = Xrel; Vnv = Xrel;
Vtv = Xrel; Vnv = Xrel;
Axv = Xrel; Ayv = Xrel;
Axv = Xrel; Ayv = Xrel;
Anv = Xrel; Atv = Xrel;
Anv = Xrel; Atv = Xrel;
zx = zc (:, 1);
zx = zc (:, 1);
zy = zc(:, 2);
zy = zc(:, 2);
nx}=\textrm{n}(:,1)
nx}=\textrm{n}(:,1)
ny = n(:,2);
ny = n(:,2);
tx = t (:, 1);
tx = t (:, 1);
ty=t(:, 2);
ty=t(:, 2);
px = pt (:, 1);
px = pt (:, 1);
py = pt(:, 2);

```
py = pt(:, 2);
```

```
Xrel = px(:,ones(1,M))-zx (:,ones(1,N))';
Yrel = py(:,ones (1,M))-zy(:,oones (1,N))';
NX = nx (:, ones (1,N) )';
NY = ny (:,ones(1,N))';
TX = tx (:,ones(1,N))';
TY = ty (:, ones (1,N))';
D = del (:,ones (1,N))';
D2 = D/2;
% normal and tangent distance from panel to pts as measured from panel
Cn = Xrel.*NX + Yrel.*NY;
Ct = Xrel.*TX + Yrel.*TY;
%Vt(i,j), Vn(i,j): tangent and normal velocity components at point i due
% to panel at point j, relative to panel j
Vt = log(((Ct+D2).^2+Cn.^ 2)./(((Ct-D2).^2+Cn.^ 2))}/(4*\mathbf{pi})
Vn}=2*\boldsymbol{atan2}((\textrm{Cn}.*\textrm{D}),(\textrm{Ct.^}2+\textrm{Cn}.^2-D2..^2))/(4*\mathbf{pi})
if sizezc=sizept
    if zc=pt
        Vn = Vn-diag(diag(Vn));
    end
end
% Vx(i,j), Vy(i,j): x and y velocity components at point i due to panel j
Axs = Vt.*TX + Vn.*NX;
Ays = Vt.*TY + Vn.*NY;
%0%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% for unit vorticity distribution, instead of source
Vtv = Vn;
Vnv}=-\textrm{Vt}
Axv = Vtv.*TX + Vnv.*NX;
Ayv = Vtv.*TY + Vnv.*NY;
%0%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if sizezc=sizept
    if zc=pt
        Ans = Axs.*NX' + Ays.*NY';
        Ans = Ans-diag(diag(Ans))+diag (0.5*ones (N,1));
        Ats = Axs.*TX' + Ays.*TY';
        Anv = Axv.*NX' + Ayv.*NY';
        Atv = Axv.*TX' + Ayv.*TY';
        Atv = Atv-diag(diag(Atv))+diag (0.5*ones (N,1));
    end
end
```


## C. 18 wakepanel.m

```
function [zcwake, tx, ty, nx, ny] = wakepanel(xnode,ynode, theta, delta)
% Returns wake panel coordinates and tangent/normal vectors given panel
% length and orientation
global k
% midpoint of wake panel expressed wrt inertial frame;
zcwake = [(xnode(1) + (delta (k)/2)* cos(theta(k)))
    (ynode(1) - (delta(k)/2)*sin (theta (k)))];
tx = cos(theta(k));
ty = -sin(theta(k));
nx = -ty;
```

ny $=t x$;

## C. 19 computeforces.m

```
function [cp,cl1, cl2,cd1,cd2,cm001,cm002,cm251,cm252, lift1, drag1, lift2,\ldots
    drag2,Moment, phi1, phi2, vortin1,vortin2, vortcrossedover, Vt] = ...
    computeforces(zc,t,n,del,D1,D2,E,F,gamma1,gamma2,delta1,delta2 ,..
    theta1, theta 2, vortexpos, vortexstrength, vortexindex, q, beta1, beta 2,\ldots
    beta, xnode1, ynode1, xnode2,ynode2,Vdx,Vdy,comx, comy, phi1, phi2,\ldots.
    vortin1km1,vortin2km1, vortcrossedoverkm1)
% Compute forces acting on bodies
global k N deltat Vinf
% tangential velocity on body
Vt = D 1*gamma1(k) + D 2*gamma2(k) + E;
% normal velocity of body
Vn}=\textrm{F}
% Velocities expressed with respect to inertial frame
VX = Vt.*t(:,1) + Vn.*n(:,1);
VY}=\textrm{Vt.*t(:,2) + Vn.*n(:, 2);
% Velocities expressed wrt frame fixed to moving body 1 E\mathcal{2}
vx1 = VX (1:N)* cos(beta1(k)+beta(k)) - VY(1:N)*\boldsymbol{sin}(beta1(k)+beta(k));
vx2 = VX (N+1:2*N)*\boldsymbol{cos}(\operatorname{beta}2(k)+\boldsymbol{beta}(k))-VY(N+1:2*N)*\boldsymbol{sin}(\operatorname{beta}2(k)+\boldsymbol{beta}(k));
vy1 = VX (1:N)*sin}(\operatorname{beta1}(\textrm{k})+\boldsymbol{beta}(\textrm{k})))+VY(1:N)*\operatorname{cos}(\operatorname{beta}1(k)+\boldsymbol{beta}(\textrm{k}))
vy2 = VX(N+1:2*N)*\operatorname{sin}(beta2(k)+beta (k))+VY(N+1:2*N)*\boldsymbol{cos}(\mathrm{ beta 2 (k)+beta(k));}
% Compute potential along both foils (See Sec 6.4.7)
[phi1(:,k), phi2(:,k),vortin1, vortin2, vortcrossedover] = ...
    potential(xnode1,ynode1, xnode2,ynode2,zc,t,n, del, Vt, vortexpos,\ldots
    vortexstrength, vortexindex,q,delta1, delta2,theta1,theta 2,gamma1,\ldots
    gamma2, vortin1km1, vortin2km1,vortcrossedoverkm1);
dphidt1 = (phi1 (:, k)-phi1(:, k - 1))/deltat (k);
dphidt2 = (phi2(:,k)-phi2(:,k-1))/deltat (k);
if k==2% to avoid spike due to zero initial value of potential function
    dphidt1=dphidt1*0;
    dphidt2=dphidt2*0;
end
dphidt = [dphidt1; dphidt2];
Vref = Vinf;
% Compute pressure coefficient distribution (See Sec 6.4.6, Eqn 6.4.9)
cp = 1- ([vx1;vx2].^ 2+[vy1;vy2].^ 2)/(norm(Vref)^2) ...
    -(2/(norm(Vref)^2))*((-Vdx.*[vx1;vx2])+(-Vdy.*[vy1;vy2])) ...
    - (2/(norm(Vref)^2))*dphidt;
[cl1, cl2, cd1, cd2, cm001, cm002, cm251, cm252, lift1, drag1, lift2, drag2, Moment]...
    = liftdragmoment (cp,xnode1, ynode1, xnode2, ynode2, zc, del,n,comx, comy );
```


## C. 20 potential.m

function [phi1, phi2, vortin1, vortin 2 , vortcrossedover] = potential(xnode1, ..
ynode1, xnode 2 , ynode $2, \mathrm{zc}, \mathrm{t}, \mathrm{n}$, del, Vt, vortexpos, vortexstrength, ..
vortexindex, q, delta1, delta2, theta1, theta2, gamma1, gamma2, vortin1km1, ..
vortin 2 km 1 , vortcrossedoverkm1)
\% computes the velocity potential along the surface of the swimmer body global $k \mathrm{~N}$ upstream TotalCirc

```
if isempty(vortexstrength) % (if there are no elements in vortexstrength)
    vortexstrength=0;
end
% nodal point at panel leading edge
midpoint = ceil(length(xnode1)/2);
tinygap = 0.00001;
leadingedge1 = [xnode1(midpoint) +n(midpoint, 1)* tinygap ...
    ynode1(midpoint) +n(midpoint,2)* tinygap ];
leadingedge 2 = [xnode 2(midpoint ) +n(midpoint +N, 1)* tinygap - tinygap ...
    ynode2(midpoint })+\textrm{n}(\operatorname{midpoint}+\textrm{N},2)*\mathrm{ tinygap ];
Vt1 = Vt(1:N);
Vt2 = Vt(N+1:2*N);
del1 = del (1:N);
del2 = del (N+1:2*N);
gap = 0.1;
%0%%%%%%%%%% Path to leading foil. First go down to leading edge, then across
philinit = 0;
% N1a and N1b = # of points along path going down and across
N1a}=40
N1b = 100;
% down
x0 = upstream(1);
yrange = [upstream(2) leadingedge1 (2)];
ypts1 = linspace(yrange(1), yrange(2),N1a+1)';
xpts1 = x0+0*ypts1;
delpts1= diff(ypts1);
xpts1=(xpts1(1:end-1)+xpts1(2: end ))/2;
ypts1}=(ypts1(1:end-1)+ypts1(2:end))/2
% across
y0 = yrange (2);
range = leadingedge1(1)-upstream (1);
xpts2 = - logspace(log10(1+range), log10(1),N1b+1)';
xpts2 = xpts2 - (min(xpts2)-upstream (1));
ypts2=y0 + xpts 2*0;
delpts2 = diff(xpts2);
xpts2 = (xpts2(1:end-1)+xpts2(2: end)) / 2;
ypts2}=(ypts2(1:\mathrm{ end - 1) +ypts2(2: end ))/2;
points = [[xpts1 ypts1];[xpts2 ypts2]];
mode=1;
[Vx1 Vy1] = velatpts(zc,t,n, del, points,xnode1, xnode 2,ynode1,ynode 2,\ldots
    gamma1,gamma2, theta1, theta2, delta1, delta2, vortexpos, vortexstrength,\ldots
    q, mode );
phi1b = phi1init + sum(Vy1(1:N1a).*delpts1) + ...
    sum(Vx1(N1a+1:N1a+N1b).* delpts2);
%%%%% FOR TRAILING FOIL
midpt = leadingedge2(1) - gap ;
if size(midpt,1)==0
    midpt = leadingedge2(1) - gap ;
end
%%%%%%%%%%%% Path to trailing foil. First go across, then down, then across
% integration path for 2nd body defined by points which separate domain
pt1 = upstream;
pt2 = [midpt upstream (2)];
pt3 = [pt2(1) leadingedge2(2)];
pt4 = leadingedge2;
pt5 = [(xnode2(1)+xnode2(N))/2 (ynode2(1)+ynode2(N))/2];
```

```
% other points to define Area 1
pt01 = [pt1(1) pt4(2)-2];
pt02 = [pt5(1) pt01(2)];
% other points to define Area 2
pt03 = [pt5(1) pt1(2)+2];
pt04 = [pt1(1) pt03(2)];
phi2init = 0;
% N1a and N1b = # of points along path going down and across
N2a = 200;
N2btop = 80; % grid points outside of influence of vortices
N2bbot = 1400; % grid points inside of influence of vortices
N2c = 500;
% across
xpts1b = linspace(pt1(1), pt2(1),N2a+1)';
ypts1b}= pt1(2)+xpts1b*0
delpts1b = diff(xpts1b);
xpts1b}=(\operatorname{xpts}1\textrm{b}(1:\mathrm{ end }-1)+\operatorname{xpts}1\textrm{b}(2:\mathrm{ end ) )/2;
ypts1b}=(ypts1b(1:end-1)+ypts1b(2:end))/2
% position of highest vortex within distance "d" of integration path +d;
d = 0.05;
if size(vortexpos, 1)>0
    if sum((vortexpos (:,1)<pt2(1)+d)&(vortexpos (:, 1)> pt2(1)-d))
            maxy = d + max(vortexpos((vortexpos (:, 1)<pt2(1)+d)&\ldots
                        (vortexpos(:,1)>pt2(1)-d),2));
            if maxy<pt4(2)
            maxy=pt4(2)+d;
            end
        else
            maxy = pt4(2)+d;
        end
else
    maxy = pt4(2)+d;
end
ypts2b1 = linspace(pt2(2), maxy, round ((pt2(2) - maxy)*N2btop));
ypts2b2 = linspace(maxy, pt3(2), round ((maxy-pt3(2))*N2bbot));
ypts2b}=[ypts2b1(1:end-1),ypts2b2]';
if min(ypts2b)<pt4(2)
    figure(2);clf;plot(ypts2b2,'.');pause(0.01);
    disp (maxy)
    disp(round ((maxy-pt3(2))*N2bbot))
    disp(ypts2b2)
end
xpts2b}= pt2(1) + ypts2b*0
delpts2b = diff(ypts2b);
xpts2b}=(\operatorname{xpts}2b(1:\mathrm{ end }-1)+xpts2b(2:end))/2
ypts2b}=(ypts2b(1:end-1)+ypts2b(2:end))/2
N2b}= length(xpts2b)
% across
xpts3b}= linspace(pt3(1),pt4(1),N2c+1)';
ypts3b}= pt3(2)+xpts3b*0
delpts3b}=\mathbf{diff}(xpts3b)
xpts 3b = (xpts3b (1:end-1)+xpts3b(2:end))/2;
ypts3b}=(ypts3b(1:\mathrm{ end - 1) +ypts3b (2: end ))/2;
points2 = [[xpts1b ypts1b];[xpts2b ypts2b];[xpts3b ypts3b]];
[Vx2 Vy2] = velatpts(zc,t,n, del, points2, xnode1, xnode 2,ynode1,ynode2,\ldots
    gamma1,gamma2, theta1, theta2, delta1, delta2, vortexpos, vortexstrength,\ldots
    q,mode );
```

```
phi2c = phi2init + sum(Vx2(1:N2a).* delpts1b) + ...
    sum(Vy2(N2a+1:N2a+N2b).* delpts2b) + ...
    \operatorname{sum}(Vx2(N2a+N2b+1:N2a+N2b+N2c).* delpts3b);
phin1 = zeros(length(xnode1), 1);
phin2 = zeros(length(xnode1),1);
phin1(midpoint) = phi1b;
phin2(midpoint) = phi2c;
% adjust for point vortices in the flow whose branch cuts cross integration
% path (See Sec 6.4.7)
if size(vortexpos,2)>0
    % Identify index #s of vortices in Area 1 and Area 2 at current time
    % vortindex1km1 & vortindex2km1 correspond to indices at previous time
    % step
    [vortin1 vortin2] = vortin(vortexpos, vortexindex, pt1, pt2, pt 3, pt4,\ldots
                pt5,pt01,pt02,pt03,pt04);
    % check which vortices crossed from Area 1 to Area 2 (circulation of
    % these vortices is subtracted from phi2)
    Area1to2 = intersect(vortin1km1, vortin2);
    % add index number to list of vortices across integration path
    vortcrossedover = [vortcrossedoverkm1 Area1to2];
    Area2to1 = intersect(vortin2km1, vortin1);
    % remove those vortices that have crossed back over
    vortcrossedover = setdiff(vortcrossedover, Area2to1);
    TotalCirc1to 2 = sum(vortexstrength(ismember(vortexindex,\ldots
        vortcrossedover )) ;
    phin2(midpoint) = phin2(midpoint) + TotalCirc1to2;
    TotalCirc(k) = TotalCirc1to2;
else
    vortin1 = [];
    vortin2 = [];
    vortcrossedover = [];
end
for count = midpoint -1:-1:1
    phin1(count) = phin1(count+1) - Vt1(count)*del1(count);
    phin2(count) = phin2(count+1) - Vt2(count)*del2(count);
end
for count = midpoint +1:length(xnode1)
    phin1(count) = phin1(count - 1) + Vt1 (count - 1)*del1 (count - 1);
    phin2(count) = phin2(count - 1) + Vt2(count - 1)*del2(count - 1);
end
phi1 = (phin1 (1: length(xnode1) - 1) + phin1 (2:length(xnode1)))/2;
phi2 = (phin2(1: length(xnode2) - 1) + phin2(2:length(xnode2)) )/ 2;
```


## C. 21 velatpts.m


ynode2, gamma1, gamma2, theta 1 , theta 2 , delta 1 , delta 2 , vortexpos, ..
vortexstrength, $q$, mode)
global Vinf $k$ N

```
[zcwake1,tx1,ty1, nx1, ny1] = wakepanel(xnode1, ynode1,theta1, delta1);
```

[zcwake2,tx2,ty2, nx2, ny2] = wakepanel(xnode 2 , ynode 2 , theta 2 , delta 2 );
[Ans, Ats, Axstemp, Aystemp, Bnv, Btv, Bxvtemp, Byvtemp ] = ...
panelinfluence ([zc; [zcwake1; zcwake2]], [t; [tx1 ty1];[tx2 ty2]],...
$[\mathrm{n} ;[\mathrm{nx} 1 \mathrm{ny} 1] ;[\mathrm{nx} 2 \mathrm{ny2}]],[$ del; delta1 $(\mathrm{k}) ;$ delta2 (k)], points);

```
Axs = Axstemp (:, 1:2*N);
Ays = Aystemp (:, 1:2*N);
Axsw = Axstemp (:,2*N+1:2*N+2);
Aysw = Aystemp (:, 2*N+1:2*N+2);
Bxv = Bxvtemp(:, 1:2*N);
Byv = Byvtemp(:, 1:2*N);
if mode==[]
    mode=1;
end
if size(vortexpos, 1)>0
    [Cx,Cy,Cn,Ct] = ptvortinfluence(vortexpos, points,0* points, 0* points,mode);
else
    Cx=0; Cy=0;
end
Vx = Axs*q + Axsw (1)*(gamma1(k-1)-gamma1(k)) + ...
    Axsw (2)*(gamma2(k-1)-gamma2(k)) + ...
    Bxv*[gamma1(k)*ones (N,1);gamma2(k)*ones(N,1)] + ...
    Cx*vortexstrength' + Vinf(1);
Vy = Ays*q + Aysw(1)*(gamma1(k-1)-gamma1(k)) + ...
    Aysw (2)*(gamma2(k-1)-gamma2(k)) + ...
    Byv*[gamma1(k)* ones (N,1);gamma2(k)*ones (N,1)] + ...
    Cy*vortexstrength' + Vinf(2);
```


## C. 22 liftdragmoment.m

```
function [cl1, cl2, cd1, cd2, cm001, cm002, cm251, cm252, lift1, drag1, lift2, ...
```

    drag2, Moment] \(=\) liftdragmoment (cp,xnode1, ynode1, xnode 2, ynode \(2, z c, \ldots\)
    del, n, comx , comy)
    global Vinf rho N

```
midpoint = ceil(length(xnode1)/2);
leadingedge1 = [xnode1(midpoint) ynode1(midpoint)];
leadingedge2 = [xnode2(midpoint) ynode2(midpoint)}]
xspan1 = xnode1(1)-xnode1(midpoint);
yspan1 = ynode1(1)-ynode1(midpoint);
xspan2 = xnode2(1)-xnode2(midpoint);
yspan2 = ynode2(1)-ynode2(midpoint);
quarterchord1 = [leadingedge1(1)+xspan1*.25 leadingedge1(2)+yspan 1*.25];
quarterchord 2 = [leadingedge2(1)+xspan 2*.25 leadingedge2(2)+yspan 2*. 25];
cl1 = sum(-cp(1:N).* del(1:N).*n(1:N,2));
cd1 = sum(-cp(1:N).*\operatorname{del}(1:N).*n(1:N,1));
cl2 = sum(-cp(N+1:2*N).*\operatorname{del}(\textrm{N}+1:2*N).*n(N+1:2*N,2));
cd2= sum(-cp(N+1:2*N).*\operatorname{del}(\textrm{N}+1:2*N).*n(N+1:2*N,1));
rx001 = zc (1:N,1) - leadingedge1 (1);
ry001 = zc(1:N,2)-leadingedge1 (2);
rx002 = zc(N+1:2*N,1) - leadingedge2 (1);
ry002 = zc(N+1:2*N,2)- leadingedge2(2);
rx251 = zc(1:N,1)-quarterchord1 (1);
ry251= zc(1:N,2)-quarterchord1(2);
rx252 = zc(N+1:2*N,1)-quarterchord2(1);
ry252= zc(N+1:2*N,2)-quarterchord2 (2);
cm001 = sum(((rx001.*n(1:N,2) - ry001.*n(1:N,1)).* cp(1:N).*\operatorname{del}(1:N));
cm251 = sum((rx251.*n(1:N,2) - ry251.*n(1:N,1)).*cp(1:N).* del (1:N));
cm002 = sum(( rx002.*n(N+1:2*N,2)-\ldots
```

```
        ry002.*n(N+1:2*N,1)).*cp(N+1:2*N).* del (N+1:2*N));
cm252 = sum(( rx252.*n(N+1:2*N,2) - ..
    ry252.*n(N+1:2*N,1)).* cp (N+1:2*N).* del (N+1:2*N));
% airfoil chord length
% chord = (xspan^2+yspan^^2)^.5;
% dynamic pressure at infinity
q1 = 0.5*rho*norm(Vinf)^2;
% compute moment about CENTER OF MASS (comx,comy)
rx = zc (:,1) - comx;
ry = zc(:,2) - comy;
p = cp*q1;
Moment = sum((rx.*n(:, 2) - ry.*n(:,1)).*p.*del);
lift1= cl1*q1;
drag1 = cd1*q1;
lift2 = cl2*q1;
drag2 = cd2*q1;
moment001 = cm001*q1;
moment251 = cm251*q1;
moment002 = cm002*q1;
moment252 = cm252*q1;
```


## C. 23 computemotion.m

```
function [Fxb,Fyb,Mb, xp,yp, betap,vxp,vyp,vbetap]=computemotion(i, drag1,\ldots
```

    drag2, lift 1 , lift 2 , Moment, vxp, vyp, vbetap, Fx, Fy, M, vx, vy, vbeta, x, y, beta, I)
    $\%$ determine forces, velocity and position $\mathcal{G}$ orientation of swimmer
global $k$ Cf TL Vinf gammax gammay gammaw Mass deltat constrained

```
% total force
Fxp = drag1(k)+drag2(k);
Fyp = lift1(k)+lift2(k);
Mp = Moment(k);
% drag force (due to skin friction) - See Section 7.4
Drag = -sign(-Vinf(1)+vxp)*0.5* Cf*TL*(-Vinf(1)+vxp)^2; % (See Eqn 7.18)
Fxp = Fxp+Drag;
% weighted average of force
w}=.75
% See Eqns 6.57 & 6.58
Fxb}=\textrm{w}*\textrm{Fxp}+(1-\textrm{w})*\textrm{Fx}(\textrm{k}-1)
Fyb}=\textrm{w}*\textrm{Fyp}+(1-\textrm{w})*\textrm{Fy}(\textrm{k}-1)
Mb}=\textrm{w}*\textrm{Mp}+(1-\textrm{w})*\textrm{M}(\textrm{k}-1)
```



```
% velocity - See Eqns 6.59 8% 6.60
vxb = vx(k-1) + Fxb*deltat (k)/ Mass;
vyb = vy(k-1) + Fyb*deltat(k)/Mass;
vbetab}=(\operatorname{vbeta}(\textrm{k}-1)*\textrm{I}(\textrm{k})+\textrm{Mb}*\operatorname{deltat}(\textrm{k}))/(2*I(\textrm{k})-\textrm{I}(\textrm{k}-1))
% update velocity - See Eqns 6.61 & 6.62
vxp = gammax*vxb + (1-gammax )*vxp;
vyp = gammay*vyb + (1-gammay)*vyp;
vbetap = gammaw *vbetab + (1-gammaw )*vbetap;
if constrained
    vyp = 0; vbetap = 0;
end
```

```
% compute new position with updated velocity (See Eqns 6.11, 6.12, 6.13)
xp = x (k-1) + vxp*deltat (k);
yp =y(k-1) + vyp*deltat(k);
betap = beta(k-1) + vbetap*deltat(k);
```


## C. 24 convergencecheck.m

```
% if code can't converge to a solution for various reasons,
% end the loop and display error message
if converged == 2| converged ==3 converged }==
    switch converged
        case 2
            disp('Failed to converge in function flowsolution.m');
            case 3
                    disp('Forces fail to converge in main program');
            case 4
                    display('Newton solver not converging in flowsolution.m')
        end
        break
end
```


## C. 25 convergencecheck1.m

```
% breaks out of loop if computation does not require converging to
% self-propulsion solution or if code does not converge
if stationary | i>250| converged==2| converged==4
    if i>250
        converged = 3;
    end
    break
end
```


## C. 26 updatemotionvars.m

```
function [x y beta vx vy vbeta Fx Fy M] = ..
    updatemotionvars(xp,yp, betap ,vxp, vyp, vbetap, Fxb,Fyb,Mb)
global i
x = xp;
y = yp;
beta = betap;
vx = vxp;
vy = vyp;
vbeta = vbetap;
Fx = Fxb(end);
Fy = Fyb(end);
M = Mb(end );
```


## C. 27 replacewakepanel.m

```
function [vortexpos, vortexstrength, vortexindex] = ...
    replacewakepanel(vortexpos, vortexstrength , vortexindex, zcwake1,\ldots.
    zcwake2,gamma1,gamma2)
% REPLACE WAKE PANEL WITH POINT VORTEX OF EQUAL CIRCULATION AT MIDPOINT OF
% WAKE PANEL
```

```
% J Melli 1/3/07
% added index numbers for tracking each vortex generated
global l1 l2 k
Gammaw1 = 11*(gamma1(k-1) - gamma1 (k));
Gammaw2 = l2*(gamma2(k-1) - gamma2(k));
Nptvorts = length(vortexstrength);
if k==2 % should be==2
    Nptvorts = 0;
end
vortexstrength(Nptvorts+1) = Gammaw1;
vortexstrength(Nptvorts+2) = Gammaw2;
vortexpos(Nptvorts +1,:) = zcwake1;
vortexpos(Nptvorts +2,:) = zcwake2;
vortexindex(Nptvorts+1) = 2*k-1;
vortexindex (Nptvorts+2) = 2*k;
```


## C. 28 stagpoint.m

```
function xint=stagpoint(del2,Vt)
global k N
L1(1)= del2 (1)/2;
for q=2:length(del2)/2;
    L1 (q) =L1 (q-1)+\operatorname{del2}(q-1)/2+\operatorname{del2}(q)/2;
end
Vtk=Vt(N+1:2*N,k);
signch=sign(Vtk(1: end - 1).*Vtk (2 : end ));
% sign change index #
SCI = find(signch==-1); % assume only one sign change occurs
if length(SCI)>1
    % choose value nearest leading edge
    SCI=SCI (find (min (\boldsymbol{abs}(SCI-N/2))==(\boldsymbol{abs}(SCI-N/2))));
end
x1 = L1(SCI); x2 = L1(SCI +1);
z1 = Vtk(SCI); z2 = Vtk(SCI +1);
% find x-intercept
xint = x1 - z1/(( z2-z1)/(x2-x1));
```


## C. 29 angleofattack.m

```
function [aeff] = angleofattack(y2, beta2,vx,c,aeff)
% computes the effective angle of attack - See Eqn 7.3
global k deltat Vinf
y2dot (k) = (y2(k)-y2(k-1))/deltat (1);
beta2dot (k) = (beta2(k)-beta2(k-1))/deltat (1);
% Eqn 7.3
aeff(k) = - beta2(k)'-(((-beta 2dot (k)'*c/2)+y2dot (k))/(Vinf(1)-vx(k)));
```


## C. 30 controller.m

```
function [beta2,y2,beta290,y290,y2dot90,beta2dot90, aeff90,u,u1,u2] = ...
    controller (NF,Wn, control, beta2,y2, y290, y2dot90, beta290, beta2dot90,\ldots.
    aeff90,c,vx, xint,a2,b2,phi0,omega1,K,Ki,K2,Ki2,kswitch,\ldots.
    switchcontroller,u,u1,u2,time)
% Controller to modify the motion of the trailing foil - See Section 7.3
global deltat Vinf k
% low-pass filter
[B,A] = butter(NF,Wn, 'low');
if (k>7)&(control==1)
    y2dot90(k) = (y290(k)-y290(k-1))/deltat (1);
    beta2dot90(k) = (beta290(k)-beta290(k-1))/deltat (1);
    aeff90(k) = -beta290(k)'-(((-beta2dot90(k)'*c/2)+...
        y2dot90(k))/(Vinf(1)-vx(k)));
    E = filter (B,A,- aeff90 (2:end).*xint (2:end));
else
    E = zeros(1,k);
end
u1(k) = omega1+ K*(E(end)+Ki*sum(E (1:k-2))*deltat (1));
if (switchcontroller==1)&(k>kswitch)
    u2(k) = omega1+ K2*(E(end)+Ki2*sum(E (1:k-2))*deltat (1));
else
    u2(k) = u1(k);
end
if k<=kswitch
    u(k)=u1(k);
end
if k>kswitch & k<kswitch+101
    u(k)}=\textrm{u}2(\textrm{k})*((\textrm{k}-\textrm{kswitch})/100)+\textrm{u}1(\textrm{k})*(1-(\textrm{k}-\textrm{kswitch})/100)
end
if k>kswitch +100
    u(k)=u2(k);
end
if control==1
    beta2(k+1)= b2* cos(sum(u)*deltat (1)+phi0)*(1-\operatorname{exp}(-1*\operatorname{time(k)));}
```



```
    beta290(k+1)= b2*\boldsymbol{cos}(\boldsymbol{sum}(\textrm{u})*\mathrm{ deltat(1)+pi/2+phi0)*(1-㒸p(-1*time(k)));}
    y290(k+1)= a 2*\operatorname{sin}(\boldsymbol{sum}(\textrm{u})*\operatorname{deltat}(1)+\mathbf{pi}/2+\textrm{phi}0)*(1-\boldsymbol{exp}(-1*\operatorname{time}(\textrm{k})));
end
```


## C. 31 convectvortices.m

function [vortexposnew] = convectvortices (vortexpos, vortexstrength, q, ...
zc, $\mathrm{t}, \mathrm{n}$, del, gamma1, gamma2, d, redirect, $\mathrm{x}, \mathrm{y}$, beta, vx , vy, vbeta, $\mathrm{y} 1, \mathrm{y} 2, \ldots$ beta1, beta2, a, xn, yn)
\% Convects wake vortices forward in time by computing velocity induced at \% each point
global k N Vinf deltat
[Xns, Xts, Axs, Ays, Xnv, Xtv, Bxv, Byv] = panelinfluence (zc, t, n, del, vortexpos);
$[\mathrm{Cx}, \mathrm{Cy}, \mathrm{Xn}, \mathrm{Xt}]=\mathrm{ptvortinfluence}($ vortexpos, vortexpos, vortexpos, vortexpos, 1 );
$\mathrm{Vx}=\mathrm{Axs} * \mathrm{q}+\mathrm{Bxv} *[$ ones $(\mathrm{N}, 1) * \operatorname{gamma}(\mathrm{k}) ;$ ones $(\mathrm{N}, 1) * \operatorname{gamma} 2(\mathrm{k})]+\ldots$
Cx*vortexstrength ${ }^{\prime}+$ Vinf(1);
$\mathrm{Vy}=$ Ays $* \mathrm{q}+\operatorname{Byv} *[$ ones $(\mathrm{N}, 1) * \operatorname{gamma}(\mathrm{k}) ;$ ones $(\mathrm{N}, 1) * \operatorname{gamma} 2(\mathrm{k})]+\ldots$ Cy*vortexstrength ${ }^{\prime}+\operatorname{Vinf}(2) ;$
\% (Eqns 6.40 8 6.41)
vortexposnew $=$ vortexpos $+[\mathrm{Vx} \mathrm{Vy}] *$ deltat $(\mathrm{k})$;
if redirect \% See Section 6.4.8)

```
% position and orientation of body (guess) at next time step
xg}=\textrm{x}(\textrm{k})+\textrm{vx}(\textrm{k})*\mathrm{ deltat (k);
yg}=\textrm{y}(\textrm{k})+\textrm{vy}(\textrm{k})*\mathrm{ deltat (k);
betag}=\boldsymbol{beta}(\textrm{k})+\operatorname{vbeta}(\textrm{k})*\operatorname{deltat}(\textrm{k})
% body geometry at next time step (guess)
[xa1kp1, ya1kp1, xa2kp1,ya2kp1] = shape(y1(k),y2(k),beta1(k),beta2(k),\ldots
        a,xn,yn);
[zcp1,tp1, np1, delp1, l1p1, l2p1, xnode1kp1, ynode1kp1, xnode2kp1,\ldots
        ynode2kv] = geometry(xa1kp1,ya1kp1, xa2kp1,ya2kp1, betag, xg, yg );
% Check if new vortex positions are inside unallowable area
% define perimeter with space of 'd' about bodies
zcbig = expandbody(zcp1(N+1:2*N,:), np1 (N+1:2*N,: ), d);
p1a=zcbig(1,:);
p1b=zcbig(2,:);
m1 = (p1a(2)-p1b(2))/(p1a(1)-p1b (1));
b}1=(\textrm{p}1\textrm{b}(1)*\textrm{p}1\textrm{a}(2)-\textrm{p}1\textrm{a}(1)*\textrm{p}1\textrm{b}(2))/(\textrm{p}1\textrm{b}(1)-\textrm{p}1\textrm{a}(1))
% new point far downstream
xc = 100;
p1c}=[\textrm{xc m}1*\textrm{xc}+\textrm{b}1]
p2b=zcbig(end - 1,:);
p2a=zcbig(end,:);
m2 = (p2a(2)-p2b(2))/( p2a(1) - p2b (1));
b}2=(\textrm{p}2\textrm{b}(1)*\textrm{p}2\textrm{a}(2)-\textrm{p}2\textrm{a}(1)*\textrm{p}2\textrm{b}(2))/(\textrm{p}2\textrm{b}(1)-\textrm{p}2\textrm{a}(1))
p2c}=[\begin{array}{lll}{\textrm{xc}}&{\textrm{m}2*\textrm{xc}+\textrm{b}2}\end{array}]
[zxi, zyi]=curveintersect([p1a(1) p1c(1)],[p1a(2) p1c(2)],\ldots
        [p2a(1) p2c(1)],[p2a(2) p2c(2)]);
% extended zcbig for purposes of finding intersection points
zcbigInt = [[zxi zyi];zcbig;[zxi zyi]];
% identify vortices within specified area defined by zcbig
[in1 on1] = inpolygon(vortexposnew (:,1), vortexposnew (:, 2),\ldots
        zcbig(:,1), zcbig(:, 2));
in = in1+on1;
invals = find (in==1);
if sum(in )>0
        for q=1:length(invals)
        index = invals(q);
        % original vortex position
        a1 = [vortexpos(index,:)];
        % 'new' vortex position
        b1 = [vortexposnew (index,:)];
        % radius of circle based at a1
        r = norm(b1-a1);
        Xi=[];
        while length(Xi)<1
            % circle defined by center a1 and radius r
            Cx}=\textrm{a}1(1)+\textrm{r}*\boldsymbol{cos}(linspace ( 0, 2*\mathbf{pi}))
            Cy=a1(2)+r*sin(linspace(0,2*\mathbf{pi}));
            [Xi,Yi]=curveintersect(Cx, Cy, zcbigInt (:,1), zcbigInt (:, 2));
            % slowly increase r until circle intersects zcbig
            r=r*1.001;
        end
        % distance from new vortex position to all intersection points
        dist = (sum(((ones (length (Xi), 1)*b1-[Xi Yi]).^ 2 )')).`..5;
        % index of intersection point nearest assumed vortex position
```

```
                dsindex = find(dist==min(dist));
                % update vortex position to furthest downstream intersection
                % point
                vortexposnew (index,:)=[Xi(dsindex) Yi(dsindex)];
                end
    end
end
```


## C. 32 ptvortinfluence.m

```
function [Cx, Cy, Cn, Ct] = ptvortinfluence(vortexpos, points, t, n, mode)
% The influence on "points" due to unit vortex blobs at positions specified
% by "vortexpos" is returned.
m= size(vortexpos,1);
N = size(points,1);
% compute velocities induced at zc due to wake vortices
%m= number of pt vortices in wake
% N = number of pts at which induced velocity is computed
% compute angle between panel i and vortex m, relative to horizontal and
% from a ray originating at vortex m through panel i
% px(:,ones(1,M))-zx(:, ones(1,N))';
px=points(:,1);
py=points(:, 2);
vposx = vortexpos(:,1);
vposy = vortexpos(:, 2);
delx = px(:,ones(1,m))-vposx (:,oones (1,N) )';
dely = py(:,ones (1,m))-vposy(:,ones (1,N))';
d}=(\mathrm{ delx.^2+dely.^ 2).^. . 5;
%alpha = atan2(dely,delx);
% OLD COMPUTATION METHOD
% x and y components of induced velocity wrt inertial frame
%Cx = ones (1,N)'*ones(1,m)./(2*pi*d).* sin(alpha);
%Cy=-ones(1,N)'*ones (1,m)./(2*pi*d).*\operatorname{cos(alpha);}
% faster computation (avoids computing angles) (POINT VORTICES)
%Cx = dely./(2*pi*d.^2);
%Cy = -delx./(2*pi*d.^2);
% faster computation; (VORTEX BLOBS)
% explicit 4th order velocity kernel for vortex blob
% Beale and Majda, High Order Vortex Methods, J. Comp Phys (58), 1985 p.192
% using a^2 = 2
% delta = core radius
%
if mode==4
    delta = 0.1; % core radius for blob vortex distribution
end
if mode==5
    delta = 1.0
end
switch mode % treats vortex as point vortex or blob vortex
    case 1 % point vortex
        K = 1;
    case 4; % blob vortex distribution
```

```
        K = (1-2*\boldsymbol{exp}(-(d.^2)/(delta^2))+\boldsymbol{exp}((-d.^2)/(2* delta` 2) ));
    case 5; % blob vortex distribution
    K = (1-2*\operatorname{exp}(-(d.^2)/(delta^2))+\boldsymbol{exp}((-d.^2)/(2* delta` 2)));
    otherwise
    K =1;
end
Cx}=(\mathrm{ dely./(2*pi*d.^ 2)).*K;
Cy = (-delx./(2*pi*d.^2)).*K;
Cx}(\textrm{d}==0)=0
Cy (d==0)=0;
% ith row of the mth column of Vn is the normal component of velocity induced
% at the ith control point of the foil due to point vortex m
% normal and tangential induced velocity relative to local coordinate frame
if N=size(n,1)
    nx=n(:,1); ny=n(:, 2);
    tx=t (:, 1); ty=t (:,2);
    Cn = Cx.*nx (:,ones(1,m))+Cy.*ny(:,ones(1,m));
    Ct = Cx.*tx (:,ones (1,m))+Cy.*ty (:,ones (1,m));
else
    Cn=0;
    Ct=0;
end
```


## C. 33 expandbody.m

```
function zcbig = expandbody(zc,n,d);
% defines control points of larger body to be a distance d away from
% original body. This is used as a buffer zone. Point vortices entering
% this zone will be annihalated.
zcbig(:,1) = zc(:,1)+n(:,1)*d;
zcbig(:, 2) = zc(:,2)+n(:, 2)*d;
```


## C. 34 savedata.m

```
% Saves body geometry information and vortex position and strengths at each
% time step in large matrices. Useful for post processing.
vortexposx (1:size(vortexpos,1),k) = vortexpos(:, 1);
vortexposy(1:size(vortexpos,1),k)= vortexpos(:, 2);
vortexstrengthk(1:size(vortexpos,1),k) = vortexstrength ';
zcx (:, k) = zc2(:, 1);
zcy(:,k)= zc2(:, 2);
```


## C. 35 plotandsave.m

```
if displayflag = 1
    figure(1); clf;hold on;
    load mycmap; colormap(mycmap);
    patch(zcx(1:N,k), zcy(1:N,k),'k');
    patch(zcx (N+1:2*N,k), zcy (N+1:2*N,k),'k');
    scatter (vortexposx (:,k), vortexposy (:, k) , ...
        700*abs(vortexstrengthk (:, k))+eps, vortexstrengthk(:,k),'filled');
    if length(signchange1)}>
```

```
        scatter(c1center(:,1), c1center (:, 2),...
        500*\mathbf{abs}(c1strength')+eps,0.35*c1strength',' filled');
    end
    if length(signchange2)>1
    scatter(c2center (:,1), c2center (:, 2) , ...
        500*\mathbf{abs}(\textrm{c}2\mathrm{ strength')+eps,0.35*c2strength',' filled');}
    end
    axis equal; axis([min(min(zcx)) max(max(zcx)) -1 1]);
    set(gcf,'PaperPosition',[[\begin{array}{llll}{0}&{0}&{1}&{1}\end{array}]);
    set(gca,'Xtick', []);
    set(gca,'Ytick',[]);
    box on;
    pause(0.01);
    if savedisplayflag=1
    filename = [dirname,'/', filename];
    psprint(filename,k);
    end
end
if savefileflag =1 & k= length(time)-1
    savefile = ['save ', dirname,'/', savefilename];
    eval(savefile)
end
```


## C. 36 adjustvortexpos.m

```
function [vortexposnew] = adjustvortexpos(zc,t,n, del,d,Vdx,Vdy,vortexpos);
% prevents vortices from entering within region near trailing foil
global N
vortexposnew = vortexpos;
% define perimeter with space of 'd' about bodies
zcbig = expandbody(zc(N+1:2*N,:),n(N+1:2*N,:),d);
% identify vortices within specified area defined by zcbig
[in1 on1] = inpolygon(vortexpos(:,1), vortexpos(:, 2), zcbig(:,1), zcbig(:, 2));
in = in1+on1;
invals= find(in== 1);
if sum(in )>0
```

for $q=1$ :length(invals)
index $=$ invals (q);
\% original vortex position
a1 $=[$ vortexpos (index,$:)]$;
$\mathrm{d} 2=\operatorname{sum}\left(([\mathrm{a} 1(1) * \operatorname{ones}(\mathrm{~N}, 1) \quad \mathrm{a} 1(2) * \operatorname{ones}(\mathrm{~N}, 1)]-\mathrm{zcbig}) .^{\wedge} 2^{\prime}\right)$;
\% control point number of point nearest to a1
cpnum $=\boldsymbol{f i n d}(\mathrm{d} 2=\boldsymbol{\operatorname { m i n }}(\mathrm{d} 2))$;
\% distance from vortex to zcbig
Velnorm $=\operatorname{norm}([\operatorname{Vdx}(\mathrm{N}+$ cpnum) $\operatorname{Vdy}(\mathrm{N}+$ cpnum $)])$;
\% point outside zcbig in direction of normal vector
$\mathrm{b} 1=\mathrm{a} 1+\mathrm{d} *[\mathrm{n}(\mathrm{N}+$ cpnum, 1$) \mathrm{n}(\mathrm{N}+$ cpnum, 2$)] ;$
\% point of intersection.
$[\mathrm{Xi}, \mathrm{Yi}]=\mathrm{curveintersect}([\mathrm{a} 1(1) ; \mathrm{b} 1(1)],[\mathrm{a} 1(2) ; \mathrm{b} 1(2)], \operatorname{zcbig}(:, 1), \mathrm{zcbig}(:, 2))$;
if length $(\mathrm{Xi})>0$
$\%$ vector from a1 to [Xi Yi]
$\mathrm{v}=[\mathrm{Xi} \mathrm{Yi}]-\mathrm{a} 1$;

```
                % 'new' vortex position - just outside (0.1d) of zcbig
                a2 = [Xi Yi]+ 0.1*d*(v/norm(v));
                    vortexposnew(index,:) = a2;
        end
    end
end
```


## C. 37 vortin.m

```
function [vortin1 vortin2] = vortin(vortexpos, vortexindex, pt1,pt2,\ldots
    pt3,pt4,pt5,pt01,pt02,pt03,pt04)
% J Melli, 1/6/07
%
% Finds vortices in the regions Area 1 and Area 2, which are separated by
% the integration path to the trailing airfoil.
%
% The purpose is to keep track of the vortices moving from one side of the
% integration path to the other in order to adjust the value of the
% potential.
%
% Area 1 is defined by (in CW order): pt1, pt2, pt3, pt4, pt5, pt02, pt01
% Area 2 is defined by (in OCW order): pt1, pt2, pt3, pt4, pt5, pt03, pt04
xv1 = [pt1 (1);pt2(1);pt3(1);pt4(1);pt5 (1);pt02(1);pt01 (1);pt1 (1)];
yv1 =[pt1(2);pt2(2);pt3(2);pt4(2);pt5 (2);pt02(2);pt01(2);pt1 (2)];
xv2 =[pt1(1);pt2(1);pt3(1);pt4(1);pt5(1);pt03(1);pt04(1);pt1 (1)];
yv2 =[pt1(2);pt2(2);pt3(2);pt4(2);pt5(2);pt03(2);pt04(2);pt1(2)];
[vortinarea1 vortonarea1] = inpolygon(vortexpos(:, 1), vortexpos(:, 2), xv1, yv1);
[vortinarea2] = inpolygon(vortexpos(:, 1), vortexpos(:, 2), xv2, yv2);
vortin1 = vortexindex(logical(vortinarea1+vortonarea1 ));
vortin2 = vortexindex(vortinarea 2);
```

