REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
31-08-2014	Ph.D. Dissertation	-

4. TITLE AND SUBTITLE
Space--Time Fluid--Structure Interaction Computation of Flapping-Wing Aerodynamics

5a. CONTRACT NUMBER
W911NF-12-1-0162

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS
Nikolay Kostov

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES
William Marsh Rice University
6100 Main St., MS-16

Houston, TX 77005 -1827

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS (ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
62037-MA.19

12. DISTRIBUTION AVAILIBILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT
The focus of this thesis is computational fluid-structure interaction (FSI) analysis of flapping-wing aerodynamics of a micro aerial vehicle (MAV). The wing motion and deformation data, whether prescribed fully or partially, is from an actual locust, extracted from high-speed, multi-camera video recordings of the locust in a wind tunnel. The core computational FSI technology is based on the Deforming-Spatial-Domain/Stabilized Space--Time (DSD/SST) formulation. This is

15. SUBJECT TERMS
No keywords

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Tayfun Tezduyar
UU	UU	UU	UU		19b. TELEPHONE NUMBER 713-348-6051

Report Title

Space--Time Fluid--Structure Interaction Computation of Flapping-Wing Aerodynamics

ABSTRACT

The focus of this thesis is computational fluid-structure interaction (FSI) analysis of flapping-wing aerodynamics of a micro aerial vehicle (MAV). The wing motion and deformation data, whether prescribed fully or partially, is from an actual locust, extracted from high-speed, multi-camera video recordings of the locust in a wind tunnel. The core computational FSI technology is based on the Deforming-Spatial-Domain/Stabilized Space--Time (DSD/SST) formulation. This is supplemented with using NURBS basis functions in temporal representation of the wing and mesh motion, and in remeshing. Here we use the version of the DSD/SST formulation derived in conjunction with the variational multiscale (VMS) method, and this version is called "DSD/SST-VMST." The structural mechanics computations are based on the Kirchhoff-Love shell model. We use a sequential coupling technique, which is applicable to some classes of FSI problem, especially those with temporally-periodic behavior. We show that sequential coupling performs well in FSI computation of the flapping-wing aerodynamics we consider here. We analyze cases where the MAV body has rolling, pitching, or rolling and pitching motion. We study how all these influence the lift and thrust generated by the MAV.

RICE UNIVERSITY

Space–Time Fluid–Structure Interaction Computation of Flapping-Wing Aerodynamics

by

Nikolay M. Kostov

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree

Doctor of Philosophy

Approved, Thesis Committee:

T. E. Tezduyar, Chair
Professor of Mechanical Engineering and
Materials Science

K. Takizawa, Co-advisor
Associate Professor in Department of
Modern Mechanical Engineering
Waseda University, Tokyo, Japan

J. E. Akin
Professor of Mechanical Engineering and
Materials Science

A. J. Meade
Professor of Mechanical Engineering and
Materials Science

M. Embree
Professor of Computational and
Applied Mathematics

Houston, Texas
December, 2013

Abstract

Space–Time Fluid–Structure Interaction Computation of Flapping-Wing Aerodynamics

by

Nikolay M. Kostov

The focus of this thesis is computational fluid-structure interaction (FSI) analysis of flapping-wing aerodynamics of a micro aerial vehicle (MAV). The wing motion and deformation data, whether prescribed fully or partially, is from an actual locust, extracted from high-speed, multi-camera video recordings of the locust in a wind tunnel. The core computational FSI technology is based on the Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) formulation. This is supplemented with using NURBS basis functions in temporal representation of the wing and mesh motion, and in remeshing. Here we use the version of the DSD/SST formulation derived in conjunction with the variational multiscale (VMS) method, and this version is called "DSD/SST-VMST." The structural mechanics computations are based on the Kirchhoff-Love shell model. We use a sequential coupling technique, which is applicable to some classes of FSI problem, especially those with temporally-periodic behavior. We show that sequential coupling performs well in FSI computation of the flapping-wing aerodynamics we consider here. We analyze cases where the MAV body has rolling, pitching, or rolling and pitching motion. We study how all these influence the lift and thrust generated by the MAV.

Acknowledgment

I want to begin by thanking Dr. Tayfun Tezduyar for giving me the great opportunity to work on cutting edge applied engineering research in computational fluid–structure interactions. His standards for all work done at the Team for Advanced Flow Simulation and Modeling (T★AFSM) created a very professional work environment and resulted in a steep learning curve for which I am truly grateful. I am sure that all the knowledge gained in the lab, both professional and personal, will be of great help in my future career.

I am extremely grateful to Dr. Kenji Takizawa whose help was invaluable every step along the way. He was always generous with his great wealth of knowledge and with his time. His expertise in every single aspect of the work process was inspirational. He is, most certainly, one of the most hard working and dedicated people I have ever worked with. While his success so far is exceptional, I am certain that he will continue having an unparalleled career and great achievements in all his future endeavors.

I want to thank Dr. Ed Akin, Dr. Mark Embree, and Dr. Andrew Meade for being on my thesis committee and for their valuable feedback. They all provided the first glimpses into computational methods and fluid mechanics during my undergraduate years at Rice University and were part of the catalyst for my decision to choose Rice for my graduate studies.

I extend my gratitude to all my friends and coworkers from the T★AFSM who not only worked with me on a variety of challenging engineering problems, but also taught me a lot about camaraderie and teamwork. It would not be possible to finish my thesis if it was not for their help, insightful discussions, and acute observations. I want to thank Dr. Kenta Sugihara, Brad, Tim, Tyler, Tony, Darren, Matt, Kat, Spenser, Joe, Austin, Casey, Cody, Ryan, and Ken for their help and support on a daily basis.

Very special thanks go to my family for believing in me and encouraging me to finish what I have started. To my mother Dobrinka, my father Marin, my brother Konstantin and his wife Sibila, I am deeply thankful for your love and support.

Finally, I want to express my gratitude to the entire Rice community for the wonderful undergraduate and graduate experience and the most exciting 7.5 years of my life.

This work was supported by ARO Grant W911NF-12-1-0162. We thank Professor Fabrizio Gabbiani and Dr. Raymond Chan (Baylor College of Medicine) for providing us the digital data extracted from the videos of the locust in their wind tunnel.

Contents

Abstract ii

Acknowledgments iii

List of figures viii

List of tables xiv

Acronyms xv

1 Introduction 1

1.1 Project description 2

1.2 Overview 6

2 Governing equations and finite element formulations 8

2.1 Fluid mechanics governing equations 8

2.2 Structural mechanics governing equations 9

2.3 DSD/SST formulation of fluid mechanics 10

2.4 DSD/SST-VMST (ST-VMS) formulation of fluid mechanics 12

2.4.1 ST variational formulation 13

2.4.2 Scale separation 13

2.4.3 DSD/SST-VMST (ST-VMS) formulation 14

2.5 New-generation ST formulations 19

2.5.1 Mesh representation 20

2.6 Semi-discrete formulation of structural mechanics 21

3 Special techniques **22**

3.1 Time representation 22

3.1.1 Time marching problem 23

3.1.2 Design of temporal NURBS basis functions 26

3.1.3 Approximation in time 26

3.2 Simple-Shape Deformation Model (SSDM) 27

3.3 Mesh update techniques in the temporal NURBS representation . . . 29

3.3.1 Mesh computation and representation 29

3.3.2 Remeshing techniques 29

3.4 Fluid mechanics computation with temporal NURBS mesh 30

3.4.1 No-slip condition on a prescribed boundary 30

3.4.2 Starting condition 33

4 Flapping-motion and geometry representation **38**

4.1 Wing and body shape 38

4.2 Flapping-motion representation 40

4.2.1 Generating a periodic data set 42

4.2.2 Motion of the wings 42

5 Base MAV computation **44**

5.1 Surface and volume meshes 44

5.2 Mesh update technique 45

5.3 Computation of the base case 47

5.3.1 Computational conditions 47

5.4 Results 49

6 SCFSI computation **55**

6.1 Setup and computations 55

6.2 Results 59

7 Membrane model **64**

7.1 Setup and computations 64

7.2 Results 67

8 Prescribed body motion **70**

8.1 Locust body motion 71

8.1.1 Setup and computations 71

8.1.2 Results 72

8.2 Prescribed roll 72

8.2.1 Setup and computations 72

8.2.2 Results 78

8.2.3 Prescribed pitch 78

8.2.4 Prescribed roll and pitch 83

9 Flapping-pattern studies **87**

9.1 Setup and computations 88

9.1.1 FW advanced or delayed 88

9.1.2 Asymmetric flapping 89

9.2 Results 89

9.2.1 FW advanced or delayed 89

9.2.2 Asymmetric flapping 98

10 Conclusions **104**

Bibliography **106**

List of figures

2.1 ST slab in an abstract representation. 10
2.2 Temporal basis functions. 19

3.1 Data represented with NURBS. The data and control variables (top). The basis functions corresponding to each control variables (bottom). 23
3.2 The data represented with basis functions after the knot insertion. The data and control variables (top). The basis functions corresponding to each control variables (bottom). 24
3.3 The data and control variables (top). The basis functions that we separately form for a given interval for the ST computation (bottom). To integrate over the interval, in the NURBS representation of the data we need to search for the corresponding element and parametric coordinate for the time $t(\vartheta^g)$ of each quadrature point ϑ^g, and interpolate the value from the data. 25
3.4 NURBS representation for a time-varying spatial position vector. The circles are the spatial position vector at each sampling time. The squares are the temporal-control points and the smooth curve is represented by them. 27
3.5 SSDM. Complex shape is shaded. Circles are tracked points. SS is represented by squares (control points). 28

3.6 Remeshing and trimming NURBS. A set of un-remeshed meshes (top). A set of remeshed meshes (middle). Common basis functions (bottom). 31

3.7 Remeshing with knot insertion. For the set of un-remeshed meshes, there are p newly-defined basis functions and the corresponding control points are marked "New." We carry out the mesh moving computations for those meshes. 32

3.8 Mesh representation for the starting condition with quadratic NURBS in time. 34

3.9 Mesh representation for the starting condition with cubic NURBS in time. 35

4.1 Wing model construction. 39

4.2 MAV FW and HW surfaces represented by NURBS and the corresponding control mesh. 39

4.3 Locust body and wings, see [1], (left) and MAV body and wings (right). 40

4.4 MAV body patches. 40

4.5 Tracking points in data set provided by BCM. 41

4.6 Process used for building a periodic flapping cycle. 43

5.1 MAV wing and body surface meshes with triangular elements. 45

5.2 Surface meshes on some of the external computational boundaries and on the outer surface of the inner, cylindrical mesh region, which has been rotated to account for in-flight body angle. 46

5.3 Length scales involved in the model used in the computations. . . . 48

5.4 Total lift and thrust generated over one cycle. Positive value indicates that thrust exceeds drag. 49

5.5 Lift and thrust generated on the right FW. 50

5.6 Lift and thrust generated on the right HW. 50

5.7 Surface pressure in Pa (relative to the free-stream pressure) at the first four of the eight equally-spaced points during the third flapping cycle (top view on left, bottom view on right). 52
5.8 Surface pressure in Pa (relative to the free-stream pressure) at the last four of the eight equally-spaced points during the third flapping cycle (top view on left, bottom view on right). 53
5.9 Volume rendering of the vorticity magnitude for the eight equally-spaced points during the third flapping cycle (left to right and then top to bottom). 54

6.1 Spatial-control mesh for the HW SS. 56
6.2 MAV with the HW deformation extracted from the structural mechanics computation, seen at eight equally-spaced instants during the third flapping cycle. 58
6.3 Net force distribution (in Pa) at eight equally-spaced instants during the third flapping cycle. 60
6.4 Total lift (top) and thrust (bottom) at different SCFSI iterations. . . 61
6.5 Right HW lift (top) and thrust (bottom) at different SCFSI iterations. 62
6.6 Right FW lift (top) and thrust (bottom) at different SCFSI iterations. 63

7.1 Top view of the MAV with the HW deformation extracted from the structural mechanics computation, seen for the shell model at eight equally-spaced instants during the third flapping cycle.) 65
7.2 Top view of the MAV with the HW deformation extracted from the structural mechanics computation, seen for the membrane model at eight equally-spaced instants during the third flapping cycle.) 66
7.3 Total lift (top) and thrust (bottom) with the shell and membrane models. 67

7.4 Right HW lift (top) and thrust (bottom) with the shell and membrane models. 68

8.1 Flapping and body motions for the base (grey) and locust-body (blue) cases at the first four of eight equally-spaced instants during the flapping cycle. Top (left) and side (right) views. 73

8.2 Flapping and body motions for the base (grey) and locust-body (blue) cases at the last four of eight equally-spaced instants during the flapping cycle. Top (left) and side (right) views. 74

8.3 Vertical tip position and velocity for the right wings for the base and locust-body cases. 75

8.4 Longitudinal tip position and velocity for the right wings for the base and locust-body cases. 76

8.5 Total lift (top) and thrust (bottom) for the base and locust-body cases. 77

8.6 Prescribed-roll case with the HW deformation extracted from the structural mechanics computation, seen at eight equally-spaced instants during the third flapping cycle. 79

8.7 Total lift (top) and thrust (bottom) for the fixed-body and prescribed-roll cases. 80

8.8 Prescribed-pitch case with the HW deformation extracted from the structural mechanics computation, seen at eight equally-spaced instants during the third flapping cycle.) 81

8.9 Total lift (top) and thrust (bottom) for the fixed-body and prescribed-pitch cases. 82

8.10 Total lift (top) and thrust (bottom) for the fixed-body and prescribed roll and pitch cases. 84

8.11 Surface pressure in Pa (relative to the free-stream pressure) for the prescribed roll and pitch case, seen at the first four of eight equally-spaced instants during the third flapping cycle (top view on left, bottom view on right). 85

8.12 Surface pressure in Pa (relative to the free-stream pressure) for the prescribed roll and pitch case, seen at the last four of eight equally-spaced instants during the third flapping cycle (top view on left, bottom view on right). 86

9.1 Vertical tip position of the FW for the FWA and FWD cases compared to the FWO case. 88

9.2 Vertical tip positions of the FW and HW for the left and right sides in the asymmetric-flapping case. 90

9.3 Asymmetric-flapping case with the HW deformation extracted from the structural mechanics computation, seen at eight equally-spaced instants during the third flapping cycle. 91

9.4 Total lift (top) and thrust (bottom) for the FWA and FWO cases. . 92

9.5 Total lift (top) and thrust (bottom) for the FWD and FWO cases. . 93

9.6 Right FW lift (top) and thrust (bottom) for the FWA and FWO cases. 94

9.7 Right HW lift (top) and thrust (bottom) for the FWA and FWO cases. 95

9.8 Right FW lift (top) and thrust (bottom) for the FWD and FWO cases. 96

9.9 Right HW lift (top) and thrust (bottom) for the FWD and FWO cases. 97

9.10 Total lift (top) and thrust (bottom) for the symmetric and asymmetric flapping. 98

9.11 Right FW lift (top) and thrust (bottom) for the symmetric and asymmetric flapping. 99

9.12 Right HW lift (top) and thrust (bottom) for the symmetric and asymmetric flapping. 100

9.13 Left FW lift (top) and thrust (bottom) for the symmetric and asymmetric flapping. 101

9.14 Left HW lift (top) and thrust (bottom) for the symmetric and asymmetric flapping. 102

List of tables

5.1 Summary of computational setup. In each temporal patch we indicate the number of temporal control points and at which point we generate the tetrahedral volume mesh with the number of nodes and elements shown. 47

Acronyms

ALE	Arbitrary Lagrangian–Eulerian
DSD/SST	Deforming-Spatial-Domain/Stabilized Space–Time
DSD/SST-SUPS	DSD/SST version with SUPG/PSPG stabilization
DSD/SST-VMST	DSD/SST version with variational multiscale turbulence model
FSI	Fluid–structure interaction
FW	Forewing
FWA	Forewing advanced
FWD	Forewing delayed
FWO	Forewing orginal (locust data)
HW	Hindwing
LSIC	Least-squares on incompressibility constraint
MAV	Micro aerial vehicle
NURBS	Non-uniform rational B-splines
PSPG	Pressure-Stabilizing/Petrov-Galerkin
RBVMS	Residual-based variational multiscale method
SCFSI	Sequentially-coupled fluid–structure interaction
SS	Simple shape
SSDM	Simple-Shape Deformation Model
ST	Space–time
ST-SUPS	Space–time version with SUPG/PSPG stabilization
ST-VMS	Space–time version with variational multiscale method
SUPG	Streamline-Upwind/Petrov-Galerkin
T★AFSM	Team for Advanced Flow Simulation and Modeling
VMS	Variational multiscale

Chapter 1

Introduction

A micro aerial vehicle (MAV) is an aerial vehicle with a scale generally smaller than 60 cm. There are many current and potential applications for MAVs: exploration of large areas with minimal human monitoring, mapping of geographic areas, creation of 3D surface models based on image capturing, surveillance, monitoring of hazardous situations, reduction of human exposure to toxic materials, monitoring warehouse inventory, and many others. All of these rely on self-sufficient devices that can sustain long periods of flight without a need to recharge or refuel. This requires an energy-efficient flight system and it is only natural to look at biological examples of high efficiency flight at small scale. Fixed-wing aerodynamics is not very efficient at a small scale as the forward velocity is not enough to generate the aerodynamic forces necessary without increasing wing area too much. This requires having wings that move faster than the forward velocity. This can be achieved by either rotating blades or flapping wings. Flapping-wing insects are known to be very energy efficient. Locusts, in particular, can fly very long distances over several days on a small amount of calories. Bio-inspired flapping wings allow a high energy efficiency and maneuverability. However, to be able to understand and utilize these biological flapping patterns we need accurate simulation and modeling techniques. Computational modeling of

flapping-wing aerodynamics can allow a cost-effective way to develop future MAVs. In this thesis we will present the numerical methods used for MAV design including the fluid–structure interactions (FSI) involved in the motion and deformation of the wings in flight.

1.1 Project description

The ability to accurately predict the aerodynamic forces during flapping flight allows flexibility in the design of MAVs. With little additional cost many different wing configurations can be evaluated with no physical models being built. In addition, wing interactions can be studied to maximize beneficial aerodynamic effects. A variety of flight maneuvers can be studied, including different angles of attack, rolling, and pitching. The rest of the material in this section is from [2].

Space–time (ST) computational analysis of flapping-wing aerodynamics of an actual locust was first presented in [3]. The wing motion and deformation data was extracted from high-speed, multi-camera video recordings of the locust in a wind tunnel, which was part of a parallel, experimental research carried out by Professor Fabrizio Gabbiani and Dr. Raymond Chan at Baylor College of Medicine in Houston. The ST computational analysis was based on the Deforming-Spatial-Domain/Stabilized ST (DSD/SST) formulation [4, 5, 6, 7, 8, 9, 10, 11], which is the core computational method, and special ST techniques targeting flapping-wing aerodynamics.

The DSD/SST formulation is an interface-tracking (moving-mesh) method for computation of flows with moving interfaces, including FSI. The stabilization components of the formulation are the Streamline-Upwind/Petrov-Galerkin (SUPG) [12] and Pressure-Stabilizing/Petrov-Galerkin (PSPG) [4, 13] methods. The ST-VMS method [10] is the the variational multiscale version of the DSD/SST method, which was originally called “DSD/SST-VMST” (i.e. the version with the VMS turbulence

model) in [9]. The VMS components are from the residual-based VMS method given in [14, 15, 16, 17]. The original DSD/SST formulation was named "DSD/SST-SUPS" in [9] (i.e. the version with the SUPG/PSPG stabilization), which was also called "ST-SUPS" in [11].

In interface-tracking methods, as the interface moves and the spatial domain occupied by the fluid changes its shape, the mesh moves to accommodate this shape change and to follow (i.e. "track") the interface. The Arbitrary Lagrangian–Eulerian (ALE) finite element formulation [18] is the most widely used moving-mesh technique, with increased emphasis on FSI in recent years (see, for example, [19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 11, 47, 48, 49, 50, 51]).

Moving-mesh methods require mesh update methods. Mesh update typically consists of moving the mesh for as long as possible and remeshing as needed. With the key objectives being to maintain the element quality near solid surfaces and to minimize frequency of remeshing, a number of advanced mesh update methods [52, 53, 54, 8] were developed to be used with the DSD/SST method, including those that minimize the deformation of the layers of small elements placed near solid surfaces.

Though not as popular as the ALE formulation, the DSD/SST method has been applied to some of the most challenging flow problems with moving interfaces. The classes of problems solved include the free-surface and multi-fluid flows [4, 6, 52, 55, 54], fluid–object interactions [4, 5, 6, 55], flows with solid surfaces in fast, linear or rotational relative motion [55, 54, 56, 57, 37, 58], compressible flows [55], fluid–particle interactions [55, 54], and FSI [59, 60, 61, 62, 8, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 9, 73, 74, 75, 76, 10, 77, 78, 79].

One of the desirable features of the DSD/SST formulation is that for the temporal representation in ST computations we can use higher-order basis functions such as NURBS, which have been used very effectively as spatial basis functions in fluid and

structural mechanics and FSI computations [80, 21, 25, 81]. Using NURBS basis functions for the temporal representation provides a more accurate representation of the surface motion and deformation and a more robust and effective way of moving the mesh and remeshing. This is at the core of the special ST techniques targeting flapping-wing aerodynamics. The special ST techniques were demonstrated in [3] with the preliminary computation of the flapping-wing aerodynamics of an actual locust. A number of advances in the special ST techniques were presented in [1]. The advances include a more accurate and realistic representation of the wing geometry, a smoother temporal representation of the wing motion and deformation, a temporally-periodic representation of the wing motion and deformation, a more effective way of generating and moving an inner mesh around the locust, and a better starting condition for the mesh motion. A detailed computational analysis of bio-inspired flapping-wing aerodynamics of an MAV was presented in [82], where the wing shapes, motions and deformations were essentially the same as those of the actual locust in [1]. The MAV body was a simplified and streamlined version of the locust body in [1]. The detailed computational analysis included temporal and spatial refinement studies, and also using different combinations of wing configurations and investigating the beneficial and disruptive interactions between the wings and the role of wing camber and twist. A condensed version of the material from [1], concentrating on the flapping-motion modeling and computations, and a temporal-order study from [82] were presented in [83].

In an FSI computation with a moving-mesh method, the FSI coupling technique determines how the coupling between the equation blocks representing the fluid mechanics, structural mechanics, and mesh moving equations is handled. The coupling techniques used in ST FSI computations (i.e. FSI computations with the DST/SST method) evolved over the years from block-iterative FSI coupling [84] (see [62, 8] for the terminology) to a more robust version of block-iterative coupling [84, 62, 85] and

to quasi-direct coupling [62] and direct coupling [62] techniques. The quasi-direct and direct coupling techniques, which are applicable to cases with nonmatching fluid and structure meshes at the interface, yield more robust algorithms for FSI computations where the structure is light, such as parachute FSI computations. New versions of the quasi-direct and direct coupling techniques with upgraded and additional interface projection methods [8, 64, 69, 72, 86] have a substantially increased robustness in ST FSI computations, and rendered the earlier ST FSI solvers obsolete. These new quasi-direct and direct coupling techniques automatically reduce to "monolithic" coupling when the interface has matching fluid and structure meshes. Allowing nonmatching meshes at the interface substantially increases the scope of the FSI solver, leading to success in FSI modeling of challenging problems, such as ringsail spacecraft parachutes (see [64, 65, 72, 86, 87, 73, 75, 77, 78]).

The sequentially-coupled FSI (SCFSI) technique was introduced in [88, 89] in the context of arterial FSI computations. It is applicable to some classes of FSI problems, especially those with temporally-periodic FSI dynamics, where the FSI coupling can be achieved by a sequence of alternating fluid and structural mechanics computations, with each computation taking place over a cycle or more of the FSI dynamics. Multiscale versions of the SCFSI technique increased the effectiveness of ST FSI computations in arterial fluid mechanics [68] and parachute modeling [72, 87].

In this paper we present a SCFSI analysis of flapping-wing aerodynamics of an MAV. The wing motion and deformation data, whether prescribed fully or partially, is the same as those of the actual locust in [1]. The body is the same as the one in [82]. In the fluid mechanics computations, we use the ST-VMS method in combination with the ST-SUPS method. The structural mechanics computations are mostly based on the Kirchhoff–Love shell model [90, 91, 33]. We also carry out, for comparison purpose, some test computations with the membrane model. In addition to the straight-flight case, we analyze cases where the MAV body has rolling, pitching, or

rolling and pitching motion, where we assume that the MAV somehow performs such maneuvers. We study how these maneuvers influence the lift and thrust.

1.2 Overview

In Chapter 2 we review the governing equations for the fluid and structural mechanics models. MAV flight is at small Mach numbers, so the Navier–Stokes equations of incompressible flows are used. The DSD/SST and DSD/SST-VMST formulations are reviewed.

In Chapter 3 we review the special techniques used for the representation of the wing motion, mesh generation, and mesh motion. We use NURBS basis functions in time, which allows more accurate wing motion representation. In addition, we review the techniques for enforcing a no-slip boundary condition and for starting the fluid mechanics computation when the motion is described using NURBS in time.

In Chapter 4 we describe the steps involved in fluid mechanics computation of a flapping-wing MAV while using actual locust data.

In Chapter 5 we describe the mesh generation for the fluid mechanics computation of the MAV with locust wings. The aerodynamic forces generated are calculated and reported.

In Chapter 6 we describe the SCFSI technique used for MAV computations. For the computations in this chapter we use the Kirchhoff–Love shell model for the structure. The wing deformation pattern is based on prescribing the leading edge of the wing with the motion from the locust data and solving for the rest of the wing deformation. Lift and thrust results are presented and compared to the MAV with wing motion coming entirely from the locust data.

In Chapter 7 we report the results from the SCFSI computation of the MAV with the membrane model for the wings. The results for the lift and thrust for the

membrane case are compared to the shell case.

In Chapter 8 we discuss a number of cases where we prescribe the motion of the MAV. First, we discuss a case where we use the actual locust body motion and evaluate the effect of that motion on the aerodynamic forces generated. Next, we prescribe large rolling, pitching, and combined rolling and pitching motions. The lift and thrust generated in these cases are discussed. The SCFSI technique with the shell model is used for these large prescribed-motion cases.

In Chapter 9 we evaluate variations in the flapping motion. We alter the synchronization between the forewings and hindwings and study the effect on the aerodynamic forces generated. We also look at the effect of asymmetric flapping, where the motions of the left and right wings are not the same. The SCFSI technique with the shell model is used for these cases.

In Chapter 10 we provide our concluding remarks.

Chapter 2

Governing equations and finite element formulations

Flapping flight is inherently an FSI problem. The wing motion generates aerodynamic forces, which in turn deform the wings. This interaction yields the complex deformations and fluid flow that determine the dynamics of flapping flight. The fluid mechanics is governed by the Navier–Stokes equations of incompressible flows. These equations are reviewed in Section 2.1. The governing equations of motion for the structural mechanics are presented in Section 2.2. The DSD/SST formulation is reviewed in 2.3. The DSD/SST-VMST formulation is described in Section 2.4. The new-generation ST formulations, emphasizing temporal interpolation with NURBS basis functions, are described in Section 2.5. The semi-discrete formulation of structural mechanics is described in Section 2.6. The material in this chapter is from [10].

2.1 Fluid mechanics governing equations

Let $\Omega_t \subset \mathbb{R}^{n_{sd}}$ be the spatial domain with boundary Γ_t at time $t \in (0, T)$. The subscript t indicates the time-dependence of the domain. The Navier–Stokes equations

of incompressible flows are written on Ω_t and $\forall t \in (0,T)$ as

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\boldsymbol{\nabla}\mathbf{u} - \mathbf{f}\right) - \boldsymbol{\nabla}\cdot\boldsymbol{\sigma} = \mathbf{0}, \tag{2.1}$$

$$\boldsymbol{\nabla}\cdot\mathbf{u} = 0, \tag{2.2}$$

where ρ, $\mathbf{u}$ and $\mathbf{f}$ are the density, velocity and the external force, respectively. The stress tensor $\boldsymbol{\sigma}$ is defined as $\boldsymbol{\sigma}(p,\mathbf{u}) = -p\mathbf{I} + 2\mu\boldsymbol{\varepsilon}(\mathbf{u})$, with $\boldsymbol{\varepsilon}(\mathbf{u}) = \left((\boldsymbol{\nabla}\mathbf{u}) + (\boldsymbol{\nabla}\mathbf{u})^T\right)/2$. Here p is the pressure, $\mathbf{I}$ is the identity tensor, $\mu = \rho\nu$ is the viscosity, ν is the kinematic viscosity, and $\boldsymbol{\varepsilon}(\mathbf{u})$ is the strain-rate tensor. The essential and natural boundary conditions for Eq. (2.1) are represented as $\mathbf{u} = \mathbf{g}$ on $(\Gamma_t)_{\mathrm{g}}$ and $\mathbf{n}\cdot\boldsymbol{\sigma} = \mathbf{h}$ on $(\Gamma_t)_{\mathrm{h}}$, where $(\Gamma_t)_{\mathrm{g}}$ and $(\Gamma_t)_{\mathrm{h}}$ are complementary subsets of the boundary Γ_t, $\mathbf{n}$ is the unit normal vector, and $\mathbf{g}$ and $\mathbf{h}$ are given functions. A divergence-free velocity field $\mathbf{u}_0(\mathbf{x})$ is specified as the initial condition.

2.2 Structural mechanics governing equations

Let $\Omega_t^s \subset I\!R^{n_{xd}}$ be the spatial domain with boundary Γ_t^s, where $n_{xd} = 2$ for shells and membranes. The superscript "s" indicates the structure. The parts of Γ_t^s corresponding to the essential and natural boundary conditions are represented by $(\Gamma_t^s)_{\mathrm{g}}$ and $(\Gamma_t^s)_{\mathrm{h}}$. The equations of motion are written as

$$\rho^s\left(\frac{d^2\mathbf{y}}{dt^2} + \eta\frac{d\mathbf{y}}{dt} - \mathbf{f}^s\right) - \boldsymbol{\nabla}\cdot\boldsymbol{\sigma}^s = \mathbf{0}, \tag{2.3}$$

where ρ^s, $\mathbf{y}$, η, $\mathbf{f}^s$ and $\boldsymbol{\sigma}^s$ are the material density, structural displacement, damping coefficient, external force and the Cauchy stress tensor, respectively. The stresses are expressed in terms of the second Piola–Kirchoff stress tensor $\mathbf{S}$, which is related to the Cauchy stress tensor through a kinematic transformation.

For the flapping-wing MAV, what makes one structural element model different

from the other is the manner in which **S** is defined. These definitions can be found in [11].

2.3 DSD/SST formulation of fluid mechanics

A ST variational formulation of incompressible flows (see for example [4, 5, 6, 7, 8, 9]) is written over a sequence of N ST slabs Q_n, where Q_n is the slice of the ST domain between the time levels t_n and t_{n+1}, and P_n is the lateral boundary of Q_n (see Figure 2.1). We denote the trial and test functions spaces for the velocity

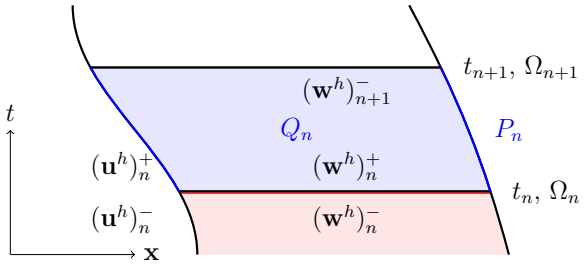


Figure 2.1: ST slab in an abstract representation.

and pressure as $\mathbf{u} \in \mathcal{S}_\mathbf{u}$, $p \in \mathcal{S}_p$, $\mathbf{w} \in \mathcal{V}_\mathbf{u}$ and $q \in \mathcal{V}_p$. The notation $(\cdot)^-_n$ and $(\cdot)^+_n$ denotes the function values at t_n as approached from below and above. At each time step, the integrations are performed over Q_n. The essential and natural boundary conditions are enforced over $(P_n)_\mathrm{g}$ and $(P_n)_\mathrm{h}$, the complementary subsets of the lateral boundary of the ST slab. In the DSD/SST method [4, 5, 6, 7, 8, 9, 10, 11], the ST finite element interpolation functions are continuous within a ST slab, but discontinuous from one ST slab to another. Each Q_n is decomposed into elements Q^e_n, where $e = 1, 2, \ldots, (n_{el})_n$. The subscript n used with n_{el} is for the general case where the number of ST elements may change from one ST slab to another. The finite-dimensional trial and test functions spaces are denoted as $\left(\mathcal{S}^h_\mathbf{u}\right)_n$, $\left(\mathcal{S}^h_p\right)_n$, $\left(\mathcal{V}^h_\mathbf{u}\right)_n$ and $\left(\mathcal{V}^h_p\right)_n$.

The DSD/SST formulation (from [7]) is written as follows: given $(\mathbf{u}^h)^-_n$, find

$\mathbf{u}^h \in (\mathcal{S}_{\mathbf{u}}^h)_n$ and $p^h \in (\mathcal{S}_p^h)_n$, such that $\forall \mathbf{w}^h \in (\mathcal{V}_{\mathbf{u}}^h)_n$ and $\forall q^h \in (\mathcal{V}_p^h)_n$:

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left(\frac{\partial \mathbf{u}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{u}^h - \mathbf{f}^h \right) dQ + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(p^h, \mathbf{u}^h) \; dQ \\
- &\int_{(P_n)_{\mathrm{h}}} \mathbf{w}^h \cdot \mathbf{h}^h \; dP + \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h \; dQ + \int_{\Omega_n} (\mathbf{w}^h)_n^+ \cdot \rho \left((\mathbf{u}^h)_n^+ - (\mathbf{u}^h)_n^- \right) d\Omega \\
+ &\sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \frac{1}{\rho} [\tau_{\mathrm{SUPG}} \rho \left(\frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h \right) + \tau_{\mathrm{PSPG}} \boldsymbol{\nabla} q^h] \cdot [Ł(p^h, \mathbf{u}^h) - \rho \mathbf{f}^h] dQ \\
+ &\sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \nu_{\mathrm{LSIC}} \boldsymbol{\nabla} \cdot \mathbf{w}^h \rho \boldsymbol{\nabla} \cdot \mathbf{u}^h \; dQ = 0,
\end{aligned} \tag{2.4}$$

where

$$Ł(q^h, \mathbf{w}^h) = \rho \left(\frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h \right) - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma}(q^h, \mathbf{w}^h). \tag{2.5}$$

This formulation is applied to all ST slabs Q_0, Q_1, Q_2, …, Q_{N-1}, starting with $(\mathbf{u}^h)_0^- = \mathbf{u}_0$. Here τ_{SUPG}, τ_{PSPG} and ν_{LSIC} are the SUPG, PSPG and LSIC stabilization parameters. There are various ways of defining these parameters. Here we provide the definitions given in [7]:

$$\tau_{\mathrm{SUPG}} = \left(\frac{1}{\tau_{\mathrm{SUGN12}}^2} + \frac{1}{\tau_{\mathrm{SUGN3}}^2} \right)^{-\frac{1}{2}}, \tag{2.6}$$

$$\tau_{\mathrm{SUGN12}} = \left(\sum_{a=1}^{n_{en}} \left| \frac{\partial N_a}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} N_a \right| \right)^{-1}, \tag{2.7}$$

$$\tau_{\mathrm{SUGN3}} = \frac{h_{\mathrm{RGN}}^2}{4\nu}, \tag{2.8}$$

$$h_{\mathrm{RGN}} = 2 \left(\sum_{a=1}^{n_{en}} |\mathbf{r} \cdot \boldsymbol{\nabla} N_a| \right)^{-1}, \quad \mathbf{r} = \frac{\boldsymbol{\nabla} \|\mathbf{u}^h\|}{\| \boldsymbol{\nabla} \|\mathbf{u}^h\| \|}, \tag{2.9}$$

$$\tau_{\mathrm{PSPG}} = \tau_{\mathrm{SUPG}}, \tag{2.10}$$

and in [8]:

$$\nu_{\mathrm{LSIC}} = \tau_{\mathrm{SUPG}} \left\| \mathbf{u}^h - \mathbf{v}^h \right\|^2, \tag{2.11}$$

where n_{en} is the number of (ST) element nodes and N_a is the ST shape function associated with the ST node a. As an alternative to the construction of τ_{SUPG} as given by Eqs. (2.6)–(2.7), another option was introduced in [8]. In that option, τ_{SUPG} is constructed based on separate definitions for the advection-dominated and transient-dominated limits:

$$\tau_{\text{SUPG}} = \left(\frac{1}{\tau_{\text{SUGN1}}^2} + \frac{1}{\tau_{\text{SUGN2}}^2} + \frac{1}{\tau_{\text{SUGN3}}^2} \right)^{-\frac{1}{2}}, \tag{2.12}$$

$$\tau_{\text{SUGN1}} = \left(\sum_{a=1}^{n_{en}} \left| \left(\mathbf{u}^h - \mathbf{v}^h \right) \cdot \boldsymbol{\nabla} N_a \right| \right)^{-1}, \tag{2.13}$$

$$\tau_{\text{SUGN2}} = \frac{\Delta t}{2}, \tag{2.14}$$

where $\mathbf{v}^h$ is the mesh velocity and Δt is the time-step size. It was noted in [8] that separating τ_{SUGN12} into its advection- and transient-dominated components as given by Eqs. (2.13)–(2.14) is equivalent to excluding the $\frac{\partial N_a}{\partial t}\big|_{\boldsymbol{\xi}}$ part of $\frac{\partial N_a}{\partial t}$ in Eq. (2.7), making that the definition for τ_{SUGN1}, and accounting for $\frac{\partial N_a}{\partial t}\big|_{\boldsymbol{\xi}}$ in the definition for τ_{SUGN2} given by Eq. (2.14). Here $\boldsymbol{\xi}$ is the vector of element coordinates. For more ways of calculating τ_{SUPG}, τ_{PSPG} and ν_{LSIC}, see [92, 7, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106].

2.4 DSD/SST-VMST (ST-VMS) formulation of fluid mechanics

The ST version of the residual-based VMS (RBVMS) method [14, 15, 16, 17] was introduced in [9] and compared to the original DSD/SST formulation [4, 5, 6, 7, 8]. We repeat that derivation and comparison in this section.

2.4.1 ST variational formulation

In deriving the variational formulation, we use the version of Eq. (2.1) where the advection term is in the conservation-law form. We multiply that version of Eq. (2.1) and Eq. (2.2) with the corresponding test functions and integrate them over Q_n:

$$\int_{Q_n} \mathbf{w} \cdot \rho \left(\frac{\partial \mathbf{u}}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{u}\mathbf{u}) - \mathbf{f} \right) dQ - \int_{Q_n} \mathbf{w} \cdot \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} dQ + \int_{Q_n} q \boldsymbol{\nabla} \cdot \mathbf{u} dQ = 0. \quad (2.15)$$

We integrate by parts all the terms except for the external force and enforce the essential and natural boundary conditions. That gives us the following variational formulation: find $\mathbf{u} \in \mathcal{S}_{\mathbf{u}}$ and $p \in \mathcal{S}_p$, such that $\forall\ \mathbf{w} \in \mathcal{V}_{\mathbf{u}}$ and $\forall\ q \in \mathcal{V}_p$

$$\begin{aligned}
&\int_{\Omega_{n+1}} \mathbf{w}_{n+1}^- \cdot \rho \mathbf{u}_{n+1}^- d\Omega - \int_{\Omega_n} \mathbf{w}_n^+ \cdot \rho \mathbf{u}_n^- d\Omega - \int_{Q_n} \frac{\partial \mathbf{w}}{\partial t} \cdot \rho \mathbf{u} dQ \\
&- \int_{(P_n)_{\mathrm{h}}} (\mathbf{w} \cdot \rho \mathbf{u}) (\mathbf{n} \cdot \mathbf{v}) \, dP + \int_{(P_n)_{\mathrm{h}}} (\mathbf{w} \cdot \rho \mathbf{u}) (\mathbf{n} \cdot \mathbf{u}) \, dP - \int_{Q_n} \boldsymbol{\nabla} \mathbf{w} : \rho \mathbf{u}\mathbf{u} dQ \\
&- \int_{Q_n} \mathbf{w} \cdot \rho \mathbf{f} dQ - \int_{(P_n)_{\mathrm{h}}} \mathbf{w} \cdot \mathbf{h} dP + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}) : \boldsymbol{\sigma} dQ + \int_{P_n} q \mathbf{n} \cdot \mathbf{u} dP \\
&- \int_{Q_n} \boldsymbol{\nabla} q \cdot \mathbf{u} dQ = 0. \qquad (2.16)
\end{aligned}$$

2.4.2 Scale separation

In the variational multiscale techniques [14, 15, 16, 17] the "coarse-scale" and "fine-scale" are separated as follows:

$$\mathcal{S}_{\mathbf{u}} = \overline{\mathcal{S}_{\mathbf{u}}} \oplus \mathcal{S}'_{\mathbf{u}}, \quad \mathcal{S}_p = \overline{\mathcal{S}_p} \oplus \mathcal{S}'_p, \quad \mathcal{V}_{\mathbf{u}} = \overline{\mathcal{V}_{\mathbf{u}}} \oplus \mathcal{V}'_{\mathbf{u}}, \quad \mathcal{V}_p = \overline{\mathcal{V}_p} \oplus \mathcal{V}'_p. \qquad (2.17)$$

The coarse-scale part of Eq. (2.16) is written as follows:

$$\begin{aligned}
&\int_{\Omega_{n+1}} \overline{\mathbf{w}}_{n+1}^{-} \cdot \rho \mathbf{u}_{n+1}^{-} d\Omega - \int_{\Omega_n} \overline{\mathbf{w}}_{n}^{+} \cdot \rho \mathbf{u}_{n}^{-} d\Omega - \int_{Q_n} \frac{\partial \overline{\mathbf{w}}}{\partial t} \cdot \rho \mathbf{u} dQ \\
&- \int_{(P_n)_{\mathrm{h}}} (\overline{\mathbf{w}} \cdot \rho \mathbf{u}) (\mathbf{n} \cdot \mathbf{v}) \, dP + \int_{(P_n)_{\mathrm{h}}} (\overline{\mathbf{w}} \cdot \rho \mathbf{u}) (\mathbf{n} \cdot \mathbf{u}) \, dP - \int_{Q_n} \boldsymbol{\nabla} \overline{\mathbf{w}} : \rho \mathbf{u} \mathbf{u} dQ \\
&- \int_{Q_n} \overline{\mathbf{w}} \cdot \rho \mathbf{f} dQ - \int_{(P_n)_{\mathrm{h}}} \overline{\mathbf{w}} \cdot \mathbf{h} dP + \int_{Q_n} \boldsymbol{\varepsilon}(\overline{\mathbf{w}}) : \boldsymbol{\sigma} dQ + \int_{P_n} \overline{q} \mathbf{n} \cdot \mathbf{u} dP \\
&- \int_{Q_n} \boldsymbol{\nabla} \overline{q} \cdot \mathbf{u} dQ = 0. \qquad (2.18)
\end{aligned}$$

From [14, 15, 16, 17], the fine-scale solutions are represented by the strong-form residuals of the coarse-scale:

$$\mathbf{u}' = -\frac{\tau_{\mathrm{M}}}{\rho} \mathbf{r}_{\mathrm{M}} (\overline{\mathbf{u}}, \overline{p}) , \qquad p' = -\rho \nu_{\mathrm{C}} r_{\mathrm{C}} (\overline{\mathbf{u}}) , \qquad (2.19)$$

where

$$\begin{aligned}
\mathbf{r}_{\mathrm{M}} (\mathbf{u}, p) &= \rho \left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \boldsymbol{\nabla} \mathbf{u} - \mathbf{f} \right) + \boldsymbol{\nabla} p - 2 \boldsymbol{\nabla} \cdot \mu \boldsymbol{\varepsilon}(\mathbf{u}), \qquad (2.20) \\
r_{\mathrm{C}} (\mathbf{u}) &= \boldsymbol{\nabla} \cdot \mathbf{u}, \qquad (2.21)
\end{aligned}$$

and τ_{M} and ν_{C} are stabilization parameters closely related to $\tau_{\mathrm{SUPG}}/\tau_{\mathrm{PSPG}}$ and ν_{LSIC}.

Remark 1 *More on the fine-scale approximation in conjunction with the Green's operator can be found in [14, 15, 16, 17].*

2.4.3 DSD/SST-VMST (ST-VMS) formulation

In this subsection we write from [9] the derivation of the DSD/SST-VMST formulation (i.e. the version with the variational multiscale turbulence model).

Fine-scale representation

The fine-scale solutions are represented over each element from Eq. (2.19) with $\mathbf{u}^h \in \left(\mathcal{S}^h_{\mathbf{u}}\right)_n$ and $p^h \in \left(\mathcal{S}^h_p\right)_n$:

$$\mathbf{u}' = -\frac{\tau_{\mathrm{M}}}{\rho}\mathbf{r}_{\mathrm{M}}\left(\mathbf{u}^h, p^h\right), \quad p' = -\rho\nu_{\mathrm{C}} r_{\mathrm{C}}\left(\mathbf{u}^h\right). \tag{2.22}$$

Remark 2 *When the polynomial order of the shape functions is less than two, the last term in Eq. (2.20) vanishes.*

There are various ways of defining τ_{M} and ν_{C}. For τ_{M} we use the τ_{SUPG} definition given by Eq. (2.6). For ν_{C}, we consider the ν_{LSIC} definition given by Eq. (2.11) and the definition from [17]:

$$\nu_{\mathrm{C}} = \left(\tau_{\mathrm{M}} \sum_{i=1}^{n_{sd}} G_{ii}\right)^{-1}, \quad G_{ij} = \sum_{k=1}^{n_{sd}} \frac{\partial \xi_k}{\partial x_i}\frac{\partial \xi_k}{\partial x_j}. \tag{2.23}$$

We evaluate the stabilization parameters at $\boldsymbol{\xi} = \mathbf{0}$.

Coarse-scale discretization

Using scale separation in Eq. (2.18) for velocity and pressure, we obtain:

$$\begin{aligned}
&\int_{\Omega_{n+1}} (\overline{\mathbf{w}})^-_{n+1} \cdot \rho\left((\overline{\mathbf{u}})^-_{n+1} + (\mathbf{u}')^-_{n+1}\right) d\Omega - \int_{\Omega_n} (\overline{\mathbf{w}})^+_n \cdot \rho\left((\overline{\mathbf{u}})^-_n + (\mathbf{u}')^-_n\right) d\Omega \\
&- \int_{Q_n} \frac{\partial \overline{\mathbf{w}}}{\partial t} \cdot \rho(\overline{\mathbf{u}} + \mathbf{u}') dQ + \int_{(P_n)_{\mathrm{h}}} \left(\overline{\mathbf{w}} \cdot \rho\left(\overline{\mathbf{u}} + \mathbf{u}'\right)\right)\left(\mathbf{n} \cdot \left(\overline{\mathbf{u}} + \mathbf{u}' - \mathbf{v}\right)\right) dP \\
&- \int_{Q_n} \boldsymbol{\nabla}\overline{\mathbf{w}} : \rho(\overline{\mathbf{u}} + \mathbf{u}')(\overline{\mathbf{u}} + \mathbf{u}') dQ - \int_{Q_n} \overline{\mathbf{w}} \cdot \rho\mathbf{f} dQ - \int_{(P_n)_{\mathrm{h}}} \overline{\mathbf{w}} \cdot \mathbf{h} dP \\
&+ \int_{Q_n} \boldsymbol{\varepsilon}(\overline{\mathbf{w}}) : \left(\boldsymbol{\sigma}(\overline{p}, \overline{\mathbf{u}}) + \boldsymbol{\sigma}'\right) dQ + \int_{P_n} \overline{q}\mathbf{n} \cdot (\overline{\mathbf{u}} + \mathbf{u}') dP - \int_{Q_n} \boldsymbol{\nabla}\overline{q} \cdot (\overline{\mathbf{u}} + \mathbf{u}') dQ = 0.
\end{aligned} \tag{2.24}$$

Here $\boldsymbol{\sigma}' \equiv \boldsymbol{\sigma} - \overline{\boldsymbol{\sigma}}$ is introduced temporarily. We set the fine-scale solution to zero at the spatial and temporal boundaries, use the assumption $\boldsymbol{\varepsilon}(\overline{\mathbf{w}}) : 2\mu\boldsymbol{\nabla}\mathbf{u}' = 0$ (see [107, 108]), and obtain the following form:

$$\begin{aligned}
&\int_{\Omega_{n+1}} (\overline{\mathbf{w}})^-_{n+1} \cdot \rho(\overline{\mathbf{u}})^-_{n+1} d\Omega - \int_{\Omega_n} (\overline{\mathbf{w}})^+_n \cdot \rho(\overline{\mathbf{u}})^-_n d\Omega \\
&- \int_{Q_n} \frac{\partial \overline{\mathbf{w}}}{\partial t} \cdot \rho(\overline{\mathbf{u}} + \mathbf{u}') dQ + \int_{(P_n)_{\mathrm{h}}} (\overline{\mathbf{w}} \cdot \rho\overline{\mathbf{u}}) \left(\mathbf{n} \cdot (\overline{\mathbf{u}} - \mathbf{v})\right) dP \\
&- \int_{Q_n} \boldsymbol{\nabla}\overline{\mathbf{w}} : \rho(\overline{\mathbf{u}} + \mathbf{u}')(\overline{\mathbf{u}} + \mathbf{u}') dQ - \int_{Q_n} \overline{\mathbf{w}} \cdot \rho\mathbf{f} dQ - \int_{(P_n)_{\mathrm{h}}} \overline{\mathbf{w}} \cdot \mathbf{h} dP \\
&+ \int_{Q_n} \boldsymbol{\varepsilon}(\overline{\mathbf{w}}) : \left(\boldsymbol{\sigma}(\overline{p}, \overline{\mathbf{u}}) - p'\mathbf{I}\right) dQ + \int_{P_n} \overline{q}\mathbf{n} \cdot \overline{\mathbf{u}} dP - \int_{Q_n} \boldsymbol{\nabla}\overline{q} \cdot (\overline{\mathbf{u}} + \mathbf{u}') dQ = 0. \quad (2.25)
\end{aligned}$$

We collect the fine-scale terms in one place and write the global integrations as sum of element-level integrals:

$$\begin{aligned}
&\int_{\Omega_{n+1}} (\overline{\mathbf{w}})^-_{n+1} \cdot \rho(\overline{\mathbf{u}})^-_{n+1} d\Omega - \int_{\Omega_n} (\overline{\mathbf{w}})^+_n \cdot \rho(\overline{\mathbf{u}})^-_n d\Omega - \int_{Q_n} \frac{\partial \overline{\mathbf{w}}}{\partial t} \cdot \rho\overline{\mathbf{u}} dQ \\
&+ \int_{(P_n)_{\mathrm{h}}} (\overline{\mathbf{w}} \cdot \rho\overline{\mathbf{u}}) \left(\mathbf{n} \cdot (\overline{\mathbf{u}} - \mathbf{v})\right) dP - \int_{Q_n} \boldsymbol{\nabla}\overline{\mathbf{w}} : \rho\overline{\mathbf{u}}\ \overline{\mathbf{u}} dQ - \int_{Q_n} \overline{\mathbf{w}} \cdot \rho\mathbf{f} dQ \\
&- \int_{(P_n)_{\mathrm{h}}} \overline{\mathbf{w}} \cdot \mathbf{h} dP + \int_{Q_n} \boldsymbol{\varepsilon}(\overline{\mathbf{w}}) : \boldsymbol{\sigma}(\overline{p}, \overline{\mathbf{u}}) dQ + \int_{P_n} \overline{q}\mathbf{n} \cdot \overline{\mathbf{u}} dP - \int_{Q_n} \boldsymbol{\nabla}\overline{q} \cdot \overline{\mathbf{u}} dQ \\
&- \sum_{e=1}^{(n_{el})_n} \int_{Q_n^e} \left[\left(\rho \frac{\partial \overline{\mathbf{w}}}{\partial t} + \boldsymbol{\nabla}\overline{q} \right) \cdot \mathbf{u}' + \boldsymbol{\nabla}\overline{\mathbf{w}} : \left(\rho(\mathbf{u}'\overline{\mathbf{u}} + \overline{\mathbf{u}}\mathbf{u}' + \mathbf{u}'\mathbf{u}') + p'\mathbf{I}\right) \right] dQ = 0. \quad (2.26)
\end{aligned}$$

Spatially discretized version of Eq. (2.26) is written as follows: find $\mathbf{u}^h \in \left(\mathcal{S}^h_{\mathbf{u}}\right)_n$ and $p^h \in \left(\mathcal{S}^h_p\right)_n$, such that $\forall\ \mathbf{w}^h \in \left(\mathcal{V}^h_{\mathbf{u}}\right)_n$ and $\forall\ q^h \in \left(\mathcal{V}^h_p\right)_n$:

$$\begin{aligned}
&\int_{\Omega_{n+1}} (\mathbf{w}^h)^-_{n+1} \cdot \rho(\mathbf{u}^h)^-_{n+1} d\Omega - \int_{\Omega_n} (\mathbf{w}^h)^+_n \cdot \rho(\mathbf{u}^h)^-_n d\Omega - \int_{Q_n} \frac{\partial \mathbf{w}^h}{\partial t} \cdot \rho \mathbf{u}^h dQ \\
&+ \int_{(P_n)_{\mathrm{h}}} \left(\mathbf{w}^h \cdot \rho \mathbf{u}^h\right) \left(\mathbf{n}^h \cdot (\mathbf{u}^h - \mathbf{v}^h)\right) dP - \int_{Q_n} \boldsymbol{\nabla} \mathbf{w}^h : \rho \mathbf{u}^h \mathbf{u}^h dQ - \int_{Q_n} \mathbf{w}^h \cdot \rho \mathbf{f}^h dQ \\
&- \int_{(P_n)_{\mathrm{h}}} \mathbf{w}^h \cdot \mathbf{h}^h dP + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(p^h, \mathbf{u}^h) dQ + \int_{P_n} q^h \mathbf{n}^h \cdot \mathbf{u}^h dP - \int_{Q_n} \boldsymbol{\nabla} q^h \cdot \mathbf{u}^h dQ \\
&- \sum_{e=1}^{(n_{el})_n} \int_{Q^e_n} \left[\left(\rho \frac{\partial \mathbf{w}^h}{\partial t} + \boldsymbol{\nabla} q^h \right) \cdot \mathbf{u}' + \boldsymbol{\nabla} \mathbf{w}^h : \left(\rho(\mathbf{u}' \mathbf{u}^h + \mathbf{u}^h \mathbf{u}' + \mathbf{u}' \mathbf{u}') + p' \mathbf{I} \right) \right] dQ = 0.
\end{aligned} \tag{2.27}$$

Comparison with the original DSD/SST formulation

To compare DSD/SST-VMST (ST-VMS) to a slightly modified DSD/SST-SUPS (where the advection term is in the conservation-law form), we further rearrange the terms in Eq. (2.27):

$$\begin{aligned}
&\int_{Q_n} \mathbf{w}^h \cdot \rho \left(\frac{\partial \mathbf{u}^h}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{u}^h \mathbf{u}^h) - \mathbf{f}^h \right) dQ + \int_{Q_n} \boldsymbol{\varepsilon}(\mathbf{w}^h) : \boldsymbol{\sigma}(p^h, \mathbf{u}^h) dQ \\
&- \int_{(P_n)_{\mathrm{h}}} \mathbf{w}^h \cdot \mathbf{h}^h dP + \int_{Q_n} q^h \boldsymbol{\nabla} \cdot \mathbf{u}^h dQ + \int_{\Omega_n} (\mathbf{w}^h)^+_n \cdot \rho \left((\mathbf{u}^h)^+_n - (\mathbf{u}^h)^-_n \right) d\Omega \\
&- \sum_{e=1}^{(n_{el})_n} \int_{Q^e_n} \left[\rho \left(\frac{\partial \mathbf{w}^h}{\partial t} + \mathbf{u}^h \cdot \boldsymbol{\nabla} \mathbf{w}^h \right) + \boldsymbol{\nabla} q^h \right] \cdot \mathbf{u}' dQ - \sum_{e=1}^{(n_{el})_n} \int_{Q^e_n} \boldsymbol{\nabla} \cdot \mathbf{w}^h p' dQ \\
&- \sum_{e=1}^{(n_{el})_n} \int_{Q^e_n} \rho \mathbf{u}' \cdot \left(\boldsymbol{\nabla} \mathbf{w}^h \right) \cdot \mathbf{u}^h dQ - \sum_{e=1}^{(n_{el})_n} \int_{Q^e_n} \rho \mathbf{u}' \cdot \left(\boldsymbol{\nabla} \mathbf{w}^h \right) \cdot \mathbf{u}' dQ = 0.
\end{aligned} \tag{2.28}$$

Remark 3 *The last two terms correspond to the cross stress and Reynolds stress.*

Remark 4 *If we exclude the last two terms, the formulation becomes the same as the modified DSD/SST-SUPS formulation (where the advection term is in the conservation-law form) under the conditions $\tau_{PSPG} = \tau_{SUPG}$ and $\nu_{\mathrm{C}} = \nu_{LSIC}$. The 6th and 7th terms are the SUPG/PSPG and LSIC terms.*

Equation (2.28) is conservative form of DSD/SST-VMST method. The convective form of DSD/SST-VMST is directly comparable to the unmodified DSD/SST-SUPS. For a shortcut derivation of the convective form, we integrate by parts two of the pieces in the 5th term of Eq. (2.25):

$$
\begin{aligned}
&-\int_{Q_n} \boldsymbol{\nabla}\overline{\mathbf{w}} : \rho\overline{\mathbf{u}}\overline{\mathbf{u}}dQ = -\int_{(P_n)_{\mathrm{h}}} \left(\overline{\mathbf{w}}\cdot\rho\overline{\mathbf{u}}\right)\left(\mathbf{n}\cdot\overline{\mathbf{u}}\right)dP + \int_{Q_n}\overline{\mathbf{w}}\cdot\rho(\overline{\mathbf{u}}\cdot\boldsymbol{\nabla}\overline{\mathbf{u}})dQ \\
&+\int_{Q_n}\left(\overline{\mathbf{w}}\cdot\rho\overline{\mathbf{u}}\right)\boldsymbol{\nabla}\cdot\overline{\mathbf{u}}dQ, \qquad (2.29)\\
&-\int_{Q_n} \boldsymbol{\nabla}\overline{\mathbf{w}} : \rho\mathbf{u}'\overline{\mathbf{u}}dQ = -\int_{(P_n)_{\mathrm{h}}} \left(\overline{\mathbf{w}}\cdot\rho\overline{\mathbf{u}}\right)\left(\mathbf{n}\cdot\mathbf{u}'\right)dP + \int_{Q_n}\overline{\mathbf{w}}\cdot\rho(\mathbf{u}'\cdot\boldsymbol{\nabla}\overline{\mathbf{u}})dQ \\
&+\int_{Q_n}\left(\overline{\mathbf{w}}\cdot\rho\overline{\mathbf{u}}\right)\boldsymbol{\nabla}\cdot\mathbf{u}'dQ. \qquad (2.30)
\end{aligned}
$$

The first term on the right-hand-side of Eq. (2.29) cancels the same term with opposite sign in Eq. (2.25). The first term on the right-hand-side of Eq. (2.30) vanishes because the fine-scale solution is zero at the spatial boundaries. The sum of the last terms of Eqs. (2.29) and Eq. (2.30) is zero. The discrete version of the second term on the right-hand-side of Eq. (2.29) replaces in the first term of Eq. (2.28) the advection piece in the conservation-law form. The discrete version of the second term on the right-hand-side of Eq. (2.30) replaces the 8th term of Eq. (2.28). With all that, we obtain the convective form of DSD/SST-VMST:

$$
\begin{aligned}
&\int_{Q_n}\mathbf{w}^h\cdot\rho\left(\frac{\partial\mathbf{u}^h}{\partial t}+\mathbf{u}^h\cdot\boldsymbol{\nabla}\mathbf{u}^h-\mathbf{f}^h\right)dQ+\int_{Q_n}\boldsymbol{\varepsilon}(\mathbf{w}^h):\boldsymbol{\sigma}(p^h,\mathbf{u}^h)dQ \\
&-\int_{(P_n)_{\mathrm{h}}}\mathbf{w}^h\cdot\mathbf{h}^h dP+\int_{Q_n}q^h\boldsymbol{\nabla}\cdot\mathbf{u}^h dQ+\int_{\Omega_n}(\mathbf{w}^h)_n^+\cdot\rho\left((\mathbf{u}^h)_n^+-(\mathbf{u}^h)_n^-\right)d\Omega \\
&-\sum_{e=1}^{(n_{el})_n}\int_{Q_n^e}\left[\rho\left(\frac{\partial\mathbf{w}^h}{\partial t}+\mathbf{u}^h\cdot\boldsymbol{\nabla}\mathbf{w}^h\right)+\boldsymbol{\nabla}q^h\right]\cdot\mathbf{u}'dQ-\sum_{e=1}^{(n_{el})_n}\int_{Q_n^e}\boldsymbol{\nabla}\cdot\mathbf{w}^h p'dQ \\
&+\sum_{e=1}^{(n_{el})_n}\int_{Q_n^e}\mathbf{w}^h\cdot\rho(\mathbf{u}'\cdot\boldsymbol{\nabla}\mathbf{u}^h)dQ-\sum_{e=1}^{(n_{el})_n}\int_{Q_n^e}\rho\mathbf{u}'\cdot\left(\boldsymbol{\nabla}\mathbf{w}^h\right)\cdot\mathbf{u}'dQ=0. \qquad (2.31)
\end{aligned}
$$

2.5 New-generation ST formulations

A ST basis function can be written as a product of its spatial and temporal parts:

$$N_a^\alpha = T^\alpha(\theta)\, N_a(\boldsymbol{\xi}), a = 1, 2, \ldots, n_{ens}, \quad \alpha = 1, 2, \ldots, n_{ent}, \tag{2.32}$$

where $\theta \in [-1,\ 1]$ is the temporal element coordinate, and n_{en} and n_{ent} are the number of spatial and temporal element nodes (see Figure 2.2 for examples of temporal basis functions). In general, the values

$$\phi_n^- = \lim_{t \to t_n^-} \phi^h(t), \qquad \phi_n^+ = \lim_{t \to t_n^+} \phi^h(t), \tag{2.33}$$

do not need to be equal to $\phi_{n-1}^{n_{ent}}$ and ϕ_n^1 (coefficients of the basis functions $T_{n-1}^{n_{ent}}$ and

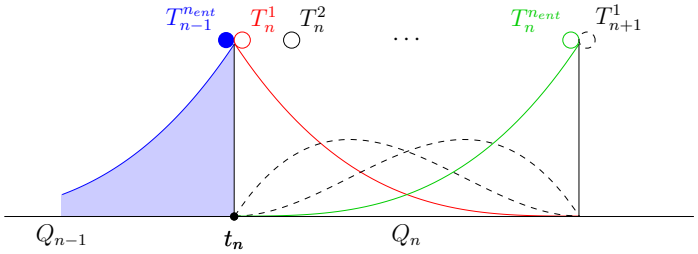


Figure 2.2: Temporal basis functions.

T_n^1). However, for the cases we consider here the basis functions are interpolatory at $\theta = -1$ and $\theta = 1$, and therefore $\phi_n^- = \phi_{n-1}^{n_{ent}}$ and $\phi_n^+ = \phi_n^1$.

As pointed out in [9], different components (i.e. unknowns), and the corresponding test functions, can be discretized with different sets of temporal basis functions. This was shown in [9] by introducing a secondary mapping $\Theta_\zeta(\theta) \in [-1,\ 1]$, where $\Theta_\zeta(\theta)$ is a strictly-increasing function, and rewriting the generalized ST basis function for the element indices $(a,\ \alpha)$ as

$$(N_a^\alpha)_\zeta = T^\alpha(\Theta_\zeta(\theta))\, N_a(\boldsymbol{\xi}). \tag{2.34}$$

Here ζ indicates the component, which can also be "t". We can also use different functions T^α for different components, which is in fact something we do in another context, and this could be in combination with the secondary mapping. Prescribed and unknown variables can be represented over different ST slabs, because we only need to supply the prescribed values at the integration points of the ST slab used in representing the unknown variables. Most of the time higher-order basis functions can represent complex functions with fewer number of control points. This is very helpful in decreasing the I/O intensity, such as in computations with the multiscale SCFSI techniques (see Section 7 of [10]).

2.5.1 Mesh representation

In general, for the moving-mesh and FSI cases, the temporal basis functions for the mesh represent the mesh path. As pointed out in [9], we need the flexibility to specify the speed along the path. Consider the mesh position vector given by

$$\mathbf{x}(\theta) = T^\alpha \left(\Theta_{\mathbf{x}}(\theta)\right) N_a \mathbf{x}_a^\alpha, \tag{2.35}$$

where the repeated indices imply summation over the applicable range. It was proposed in [9] to use $\Theta_{\mathbf{x}}(\theta)$ to specify the velocity. We seek a function $\Theta_{\mathbf{x}}$ such that

$$\mathbf{v} = \frac{d\theta}{dt}\frac{d\Theta_{\mathbf{x}}}{d\theta}\frac{dT^\alpha}{d\Theta_{\mathbf{x}}} N_a \mathbf{x}_a^\alpha \tag{2.36}$$

represents the desired mesh velocity along the path. This approach gives us the option to form a ST parametric space beyond the NURBS capability.

2.6 Semi-discrete formulation of structural mechanics

With $\mathbf{y}^h$ and $\mathbf{w}^h$ coming from appropriately defined trial and test function spaces, respectively, the semi-discrete finite element formulation of the structural mechanics equations (see [109, 110, 111]) is written as

$$\int_{\Omega_0^s} \mathbf{w}^h \cdot \rho^s \frac{d^2\mathbf{y}^h}{dt^2}\, d\Omega + \int_{\Omega_0^s} \mathbf{w}^h \cdot \eta\rho^s \frac{d\mathbf{y}^h}{dt}\, d\Omega + \int_{\Omega_0^s} \delta\mathbf{E}^h : \mathbf{S}^h\, d\Omega = \int_{\Omega_t^s} \mathbf{w}^h \cdot \left(\mathbf{t}^h + \rho^s \mathbf{f}^s\right) d\Omega. \tag{2.37}$$

The fluid mechanics forces acting on the structure are represented by vector $\mathbf{t}^h$. The above formulation is for structures represented by a shell or membrane model. The left-hand-side terms of Eq. (2.37) are referred to in the original configuration and the right-hand-side terms in the deformed configuration at time t. Time discretization of Eq. (2.37) is based on the Generalized-α method [112].

Chapter 3

Special techniques

The representation of the motion of the wings requires special mesh moving techniques as well as techniques to represent the complex deformation patterns of the wings. In this chapter we present several special techniques developed by the Team for Advanced Flow Simulation and Modeling (T★AFSM) to address these and other computational challenges related to modeling of a flapping-wing MAV. For completeness, we provide from [1] the material in this chapter.

3.1 Time representation

Let us represent time $t \in (0, T)$ with p^{th} order NURBS basis functions, R^β ($\beta = 0, \ldots, n_{\text{ct}} - 1$). The basis functions are defined on the parametric space described by the open knot vector $\{\vartheta_1, \ldots, \vartheta_{n_{\text{kt}}}\}$, where n_{ct} and n_{kt} are the number of control points and knots. Then, time t can be written as

$$t = \sum_{\beta=0}^{n_{\text{ct}}-1} t_c^\beta R^\beta(\vartheta), \tag{3.1}$$

where t_c^β represents the temporal-control point. In the case of the ST formulation there is a mesh corresponding to each temporal-control point t_c^β. As originally pro-

posed in [3], we use a strictly-increasing mapping function $\Theta_t(\theta)$ to relate the element coordinate for a knot span and the NURBS parametric space:

$$\vartheta = \frac{(1-\Theta_t(\theta))\vartheta_{e+p+1} + (1+\Theta_t(\theta))\vartheta_{e+p+2}}{2}, \tag{3.2}$$

where e represents the element index ($e = 0, \ldots, n_{elt} - 1$, where n_{elt} is the number of elements). We have $n_{\text{ct}} = n_{\text{kt}} - p - 1$, and assuming no knot multiplicity inside the knot vector, $n_{elt} = n_{\text{kt}} - 2p - 1$. The element shape functions are defined as

$$T_e^{\alpha}(\Theta_t(\theta)) = R^{e+\alpha-1}(\vartheta). \tag{3.3}$$

In the time interval of element e, we represent t with the local shape functions:

$$t(\Theta_t(\theta)) = \sum_{\alpha=1}^{p+1} t_c^{e+\alpha-1} T_e^{\alpha}(\Theta_t(\theta)). \tag{3.4}$$

3.1.1 Time marching problem

Consider a set of NURBS basis functions that is being used in representing the data that we will work with. Figure 3.1 shows an example. Starting from this, as proposed

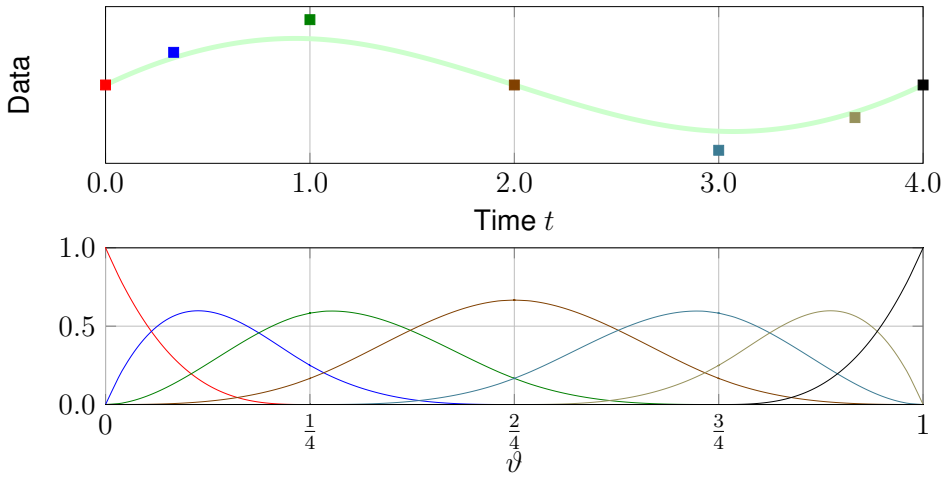


Figure 3.1: Data represented with NURBS. The data and control variables (top). The basis functions corresponding to each control variables (bottom).

in [3, 10], we can form a new basis set by knot insertion with the objective that all elements become patches as shown in Figure 3.2. After that, we can use this basis

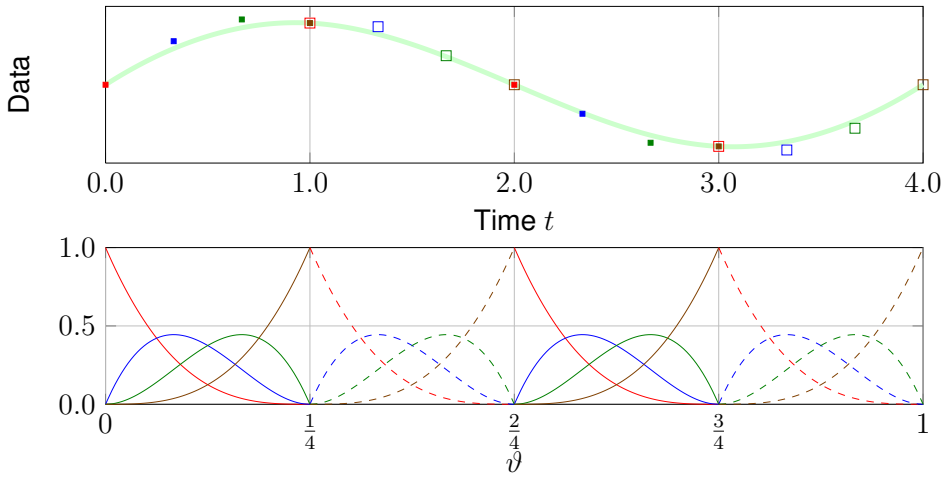


Figure 3.2: The data represented with basis functions after the knot insertion. The data and control variables (top). The basis functions corresponding to each control variables (bottom).

set in our ST computation, where the knot spans would be the time intervals of the ST slabs, and represent the data exactly. An alternative process, which was also proposed in [3, 10], has the same functionality, but without the need to explicitly represent the data we are working with using the new basis set. In that process, we separately form a new basis set where each element is a patch, and the data is represented in the formulation in terms of its own basis set. In general, the basis set that we separately form does not need to be the same as the one that could be obtained by the process described earlier, but if it is, then the two processes result in equivalent solution methods. Figure 3.3 shows the new basis functions that we separately form.

Since we need to work with different basis sets, we map one parametric space to the other through physical time t. With the function defined by Eq. (3.1), time t can be obtained from the parametric space ϑ. Here we consider the inverse mapping; i.e., $t \rightarrow \vartheta$. Finding the parametric space coordinate was proposed in [3, 10] as follows:

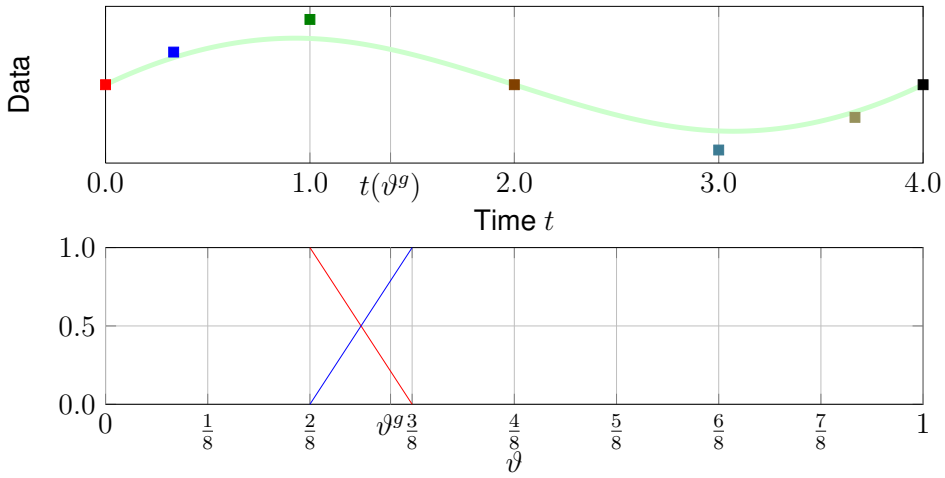


Figure 3.3: The data and control variables (top). The basis functions that we separately form for a given interval for the ST computation (bottom). To integrate over the interval, in the NURBS representation of the data we need to search for the corresponding element and parametric coordinate for the time $t(\vartheta^g)$ of each quadrature point ϑ^g, and interpolate the value from the data.

1. Find the element e that is represented by the knot span $(\vartheta_{e+p+1}, \vartheta_{e+p+2})$. The process requires only time values at each element boundary, and we can quickly obtain the element index e by using a binary search technique.

2. Calculate θ for a given t by using Newton–Raphson iterations as follows:

$$\theta^{i+1} = \theta^i - \left(t - t\left(\Theta_t\left(\theta^i\right)\right)\right)\left(\left.\frac{\mathrm{d}t}{\mathrm{d}\theta}\right|^i\right)^{-1}, \tag{3.5}$$

   where "i" is the iteration counter, $t\left(\Theta_t\left(\theta^i\right)\right)$ can be calculated from Eq. (3.4), and

$$\left.\frac{\mathrm{d}t}{\mathrm{d}\theta}\right|^i = \sum_{\alpha=1}^{p+1} t_c^{e+\alpha-1} \left.\frac{\mathrm{d}T_e^\alpha}{\mathrm{d}\Theta_t}\right|_{\Theta_t(\theta^i)} \left.\frac{\mathrm{d}\Theta_t}{\mathrm{d}\theta}\right|_{\theta^i}. \tag{3.6}$$

   We use as the initial guess $\theta^0 = 0$.

3. Compute ϑ from Eq. (3.2).

3.1.2 Design of temporal NURBS basis functions

In the previous section we described how to find the parametric space value corresponding to physical time. Here we describe from [3, 10] some specific temporal representations.

For implementation convenience and computational efficiency, we restrict the time interval of the ST slab such that the time interval does not step over a time corresponding to a temporal knot in the basis set used for representing the data or mesh. Thus the supporting set of meshes for each ST slabs consists of only specific $p+1$ meshes, where p is the order of basis used for representing the data or mesh. Because of that requirement, a uniform element size, i.e. $t(\vartheta_{e+p+2}) - t(\vartheta_{e+p+1}) = \Delta t$, where $\Delta t = \frac{T}{n_{elt}}$, is convenient. Moreover, we might have the following requirement:

$$\frac{\mathrm{d}t}{\mathrm{d}\theta} = \frac{\Delta t}{2}. \tag{3.7}$$

In the case of B-spline basis functions, for the identity mapping $\Theta_t(\theta) = \theta$, we can satisfy the condition expressed by Eq. (3.7) by selecting the control points as follows:

$$t_c^{\beta} = t_c^{\beta-1} + \frac{\vartheta_{\beta+p+1} - \vartheta_{\beta+1}}{p\left(\vartheta_{n_{\mathrm{kt}}} - \vartheta_1\right)} T, \tag{3.8}$$

for $\beta = 1, \ldots, n_{\mathrm{ct}} - 1$ and $t_c^0 = 0$.

3.1.3 Approximation in time

Let $\boldsymbol{\chi}_A^s$ be the sampling values of a time-varying spatial position vector $\mathbf{x}_A$ at sampling times t^s ($s = 0, \ \cdots, \ n_{\mathrm{sp}} - 1$, where n_{sp} is the number of sampling points). For example, $\mathbf{x}_A$ could be the position vector for spatial node A, or it could be the position vector for a point on a surface geometry extracted from video data. For each A, as proposed in [3, 10], we represent the path corresponding to the sampling

points with NURBS. This serves two purposes: bringing smoothness to the temporal representation, and better representation accuracy for less control points. First, we form a linear finite element mesh in time, consisting of two-node elements. Then, we use least-squares projection to translate that to NURBS representation:

$$\int_0^T \boldsymbol{R}_A^h \cdot \left(\mathbf{x}_A^h - \boldsymbol{\chi}_A^h\right) \mathrm{d}t = 0, \tag{3.9}$$

where $\boldsymbol{R}_A^h$ and $\mathbf{x}_A^h$ are the test function and NURBS representation in time, and $\boldsymbol{\chi}_A^h$ is the linear representation in time. Thus, we obtain the control point $\mathbf{x}_A^\beta$ corresponding to each time control point t_c^β. Figure 3.4 is an example.

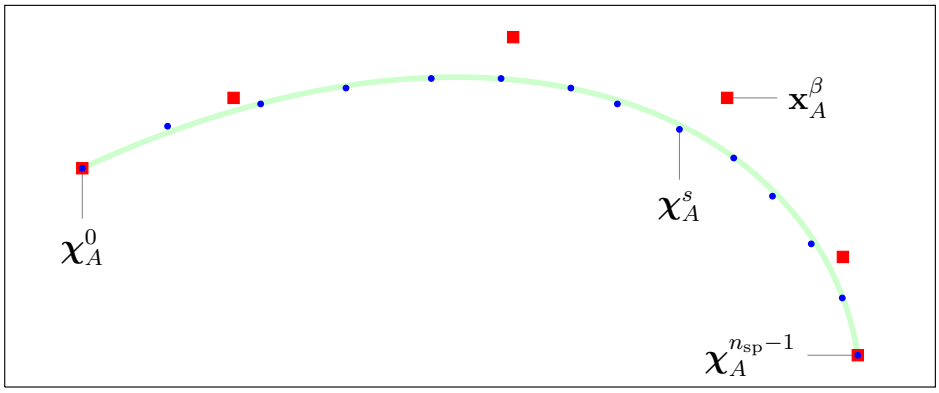


Figure 3.4: NURBS representation for a time-varying spatial position vector. The circles are the spatial position vector at each sampling time. The squares are the temporal-control points and the smooth curve is represented by them.

Remark 5 *This is a simple projection. However, the concept is applicable to more complicated formulations to obtain smoother motion.*

3.2 Simple-Shape Deformation Model (SSDM)

Suppose we want to track the motion/deformation of an object with surface shape that is too complex to track in full detail, giving us just the option of tracking only a finite number of points belonging to this complex shape. In the simple-shape deformation

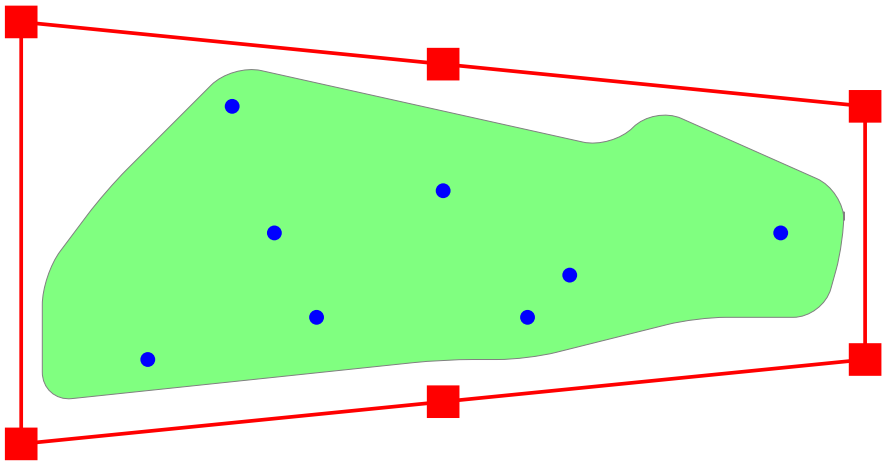


Figure 3.5: SSDM. Complex shape is shaded. Circles are tracked points. SS is represented by squares (control points).

model (SSDM), we assume that those tracked points are associated with a simple shape (SS) instead of the actual, complex shape. NURBS is used for the spatial representation of the SS. We note that the SS is larger than the complex shape. The complex shape can be represented by finite elements or NURBS.

Starting with the reference configuration, the SS, the complex shape and the tracked points all can be seen in a common parametric space (Figure 3.5). The complex shape can be represented by finite elements or NURBS. Control points of the SS at different times during tracking are determined by a least-squares fit. The fit minimizes the difference between the positions on the SS (with respect to the reference configuration) of the tracked points and the positions of the actual tracked points. The complex shape at a given temporal-control point is determined by interpolation from the parametric space in the case of the finite element representation, and by least-square projection in the case of NURBS representation. The least-squares integration is over the parametric space of the complex shape, and we minimize, with respect to the control points of the complex shape, the difference between the complex-shape and simple-shape representations.

In the full ST representation, the method we described above is applied to temporal-control values that are determined as described in Section 3.1.3, instead of the actual physical locations.

3.3 Mesh update techniques in the temporal NURBS representation

3.3.1 Mesh computation and representation

Given the surface mesh, we compute the volume mesh using the mesh moving technique we have been using [52]. Here, as proposed in [3], we apply this technique to computing the meshes that will serve as temporal-control points. This allows us to do mesh computations with longer time in between, but get the mesh-related information, such as the coordinates and their time derivatives, from the temporal representation whenever we need. Obviously this also reduces the storage amount and access associated with the meshes. However, because of the longer time between the control meshes, linear interpolation of the surfaces between control points in time might be needed in computing those meshes with the mesh moving technique mentioned.

Remark 6 *We note that getting the meshes used in the computations from the temporal representation can be done independent of which time direction was used in computing the control meshes. In other words, it does not matter if the control meshes were computed while the wings were flapping forward or backward in time.*

3.3.2 Remeshing techniques

In many computations remeshing becomes unavoidable. Two choices were proposed in [3]. To explain those two choices, let us assume that when we try to move from

control mesh M_c^β to $M_c^{\beta+1}$, we find the quality of $M_c^{\beta+1}$ to be less than desirable. In the first choice, which is called "trimming," we remesh going back to $M_c^{\beta-p+1}$. Then whenever our solution process needs a mesh, depending on the time, we use the control meshes belonging to either only the un-remeshed set or only the remeshed set (Figure 3.6).

In the second choice, we perform knot insertion p times in the temporal representation of the surface at the right-most knot before the maximum value of the basis function corresponding to $t_c^{\beta+1}$, making that knot a new patch boundary. Then we do the mesh moving computation for the control meshes associated with the newly-defined basis functions, not only the one at the new patch boundary, but also going back $(p-1)$ basis functions (Figure 3.7).

We use the second choice in computations, because we believe that in many cases the need for remeshing is generated by a topological change, which we can avoid going over with a large step if we use the knot insertion process.

3.4 Fluid mechanics computation with temporal NURBS mesh

We solve the fluid dynamics equations with the DSD/SST formulation. Here we describe two techniques related to the moving-mesh problem.

3.4.1 No-slip condition on a prescribed boundary

Suppose we have a prescribed mesh motion, and no-slip conditions on part of the boundary of that mesh. Those Dirichlet conditions can be obtained from the mesh boundary motion.

Prior to solving the equations using a ST slab Q_n, we use a least-squares projection

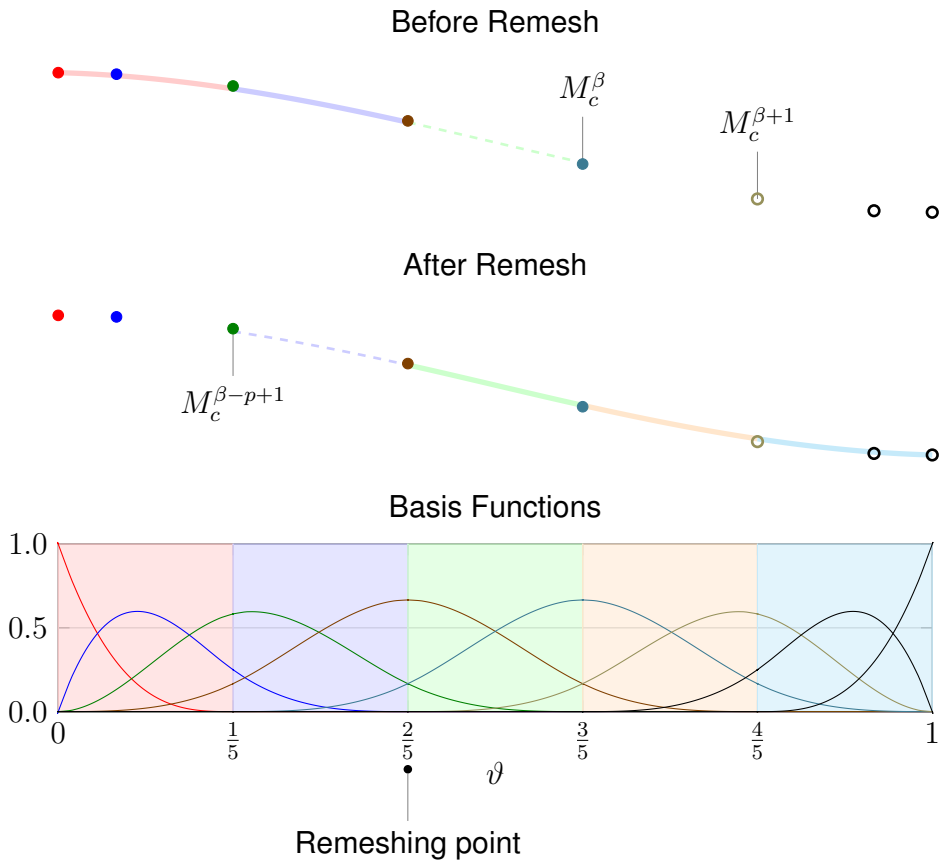


Figure 3.6: Remeshing and trimming NURBS. A set of un-remeshed meshes (top). A set of remeshed meshes (middle). Common basis functions (bottom).

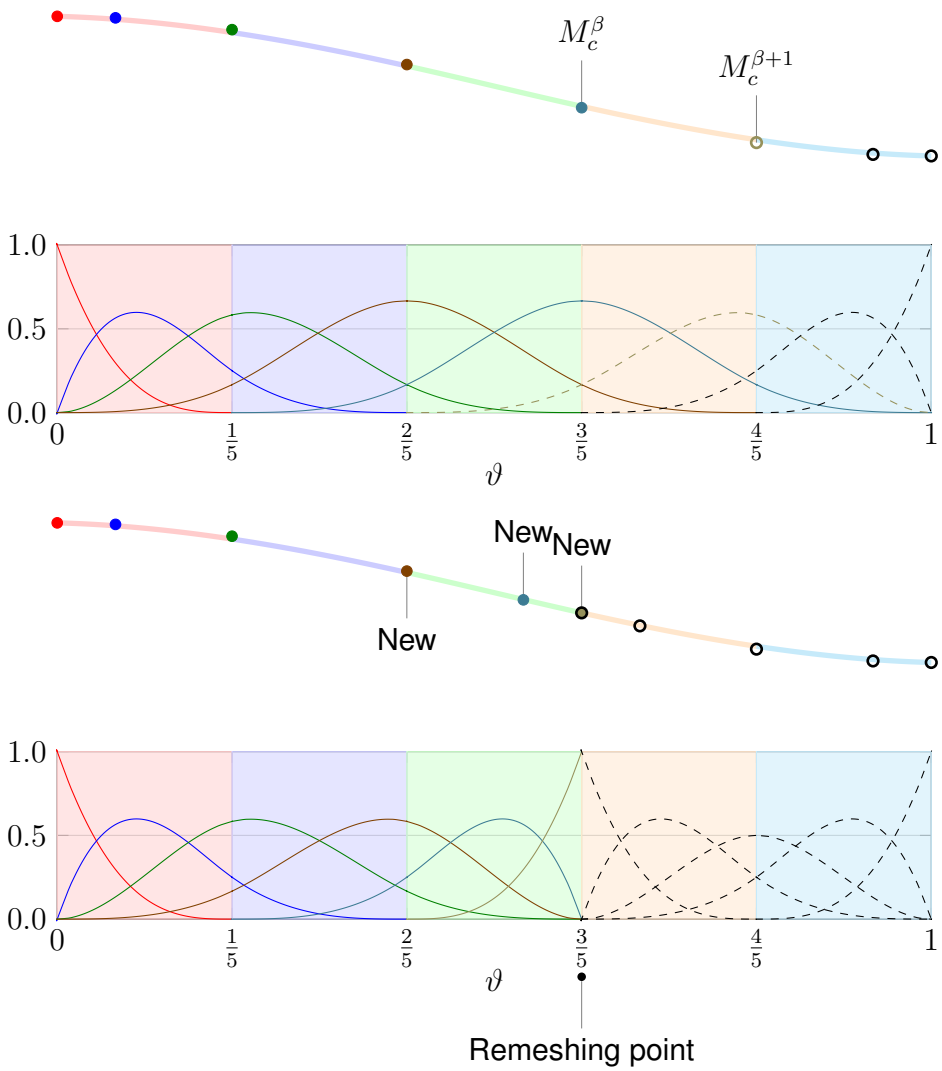


Figure 3.7: Remeshing with knot insertion. For the set of un-remeshed meshes, there are p newly-defined basis functions and the corresponding control points are marked "New." We carry out the mesh moving computations for those meshes.

for each prescribed node A as follows:

$$\int_{t_n}^{t_{n+1}} \boldsymbol{R}_A^h \cdot \left(\mathbf{u}_A^h - \frac{\mathrm{d}\mathbf{x}_A^h}{\mathrm{d}t} \right) \mathrm{d}t = 0, \tag{3.10}$$

where $\boldsymbol{R}_A^h$ represents the test function, $\mathbf{u}_A^h$ is represented by temporal-control velocities (unknown) and the corresponding basis functions in time, and the mesh velocity is obtained by the derivative of the mesh displacement, which is also represented by temporal-control positions and their basis functions. We note that $\mathbf{u}_A^h$ at time t_n approaching from below and above might be different.

3.4.2 Starting condition

Starting a fluid dynamics computation is not always easy, especially in the presence of moving boundaries. For example, pre-FSI computation techniques were developed in [113, 89, 86, 72] to build a starting condition for FSI computations. Here we describe the techniques we use as precomputation sequence for flow computations with prescribed boundary motion.

Suppose we want to compute with a mesh temporally represented with NURBS $(M_c^0,\ M_c^1,\ \cdots)$. We define the temporal-control value $\mathbf{x}_c^k$ to be used interchangeably with the k^{th} temporal-control mesh M_c^k.

With quadratic NURBS representation

We describe this option from [3]. Figure 3.8 is an example of mesh representation with quadratic NURBS in time. We generate two additional meshes as follows:

$$M_c^{-1} = \frac{M_c^0 - \alpha M_c^1}{1 - \alpha}, \tag{3.11}$$

$$M_c^{-2} = M_c^{-1}, \tag{3.12}$$

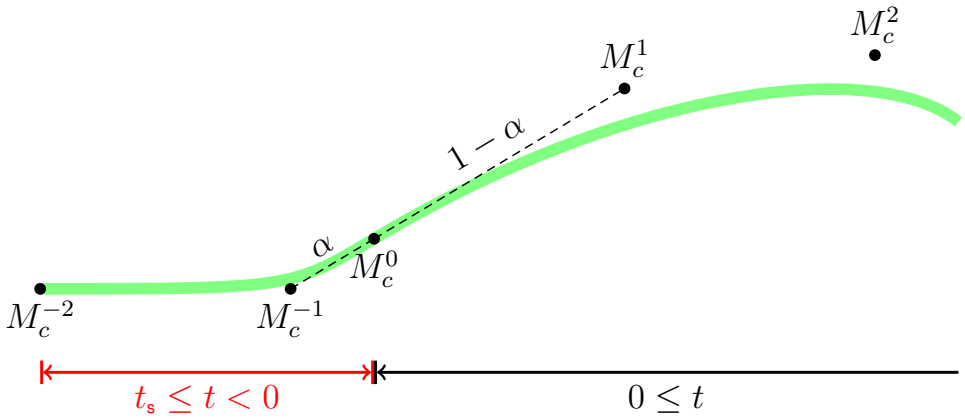


Figure 3.8: Mesh representation for the starting condition with quadratic NURBS in time.

where $0 < \alpha < 1$ is an extrapolation parameter. Mesh M_c^{-1} is an extrapolation. The corresponding temporal-control point for M_c^{-1} is

$$t_c^{-1} = \frac{t_c^0 - \alpha t_c^1}{1-\alpha}, \tag{3.13}$$

with the only requirement being $t_c^{-2} \equiv t_s < t_c^{-1}$, which determines the length of the precomputation. For the computations reported in this paper, in temporal representation of the mesh, as NURBS basis functions, we use quadratic B-spline functions defined by the knot vector $\{0, 0, 0, 1, 1, 1\}$. Based on the preliminary computations, using even higher-order NURBS basis functions was proposed in [3] so that the acceleration is continuous. We explain this in the next subsection.

With cubic NURBS representation

Figure 3.9 is an example of mesh representation with cubic NURBS in time. Here we construct a starting-condition temporal patch with C^2-continuity with the next

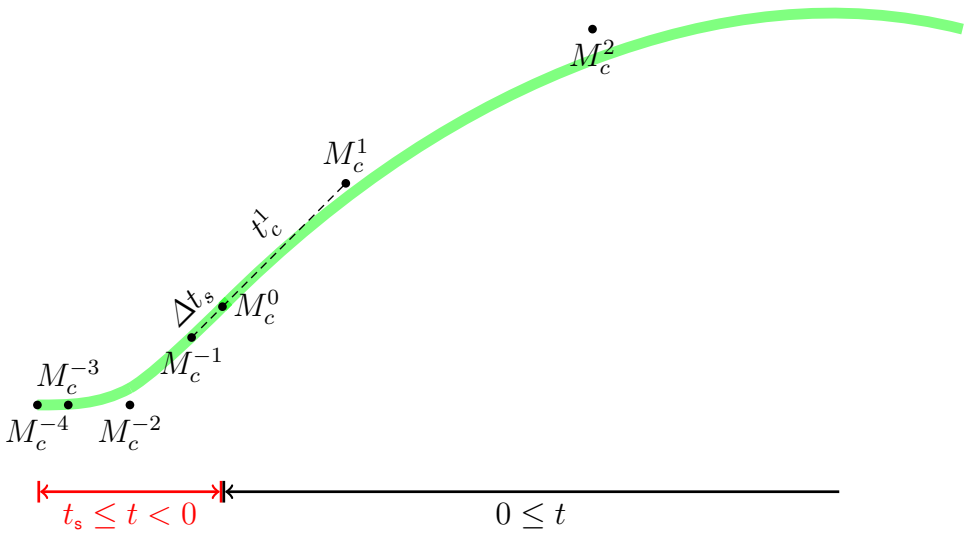


Figure 3.9: Mesh representation for the starting condition with cubic NURBS in time.

patch:

$$\lim_{t\to 0^-} \mathbf{x} = \lim_{t\to 0^+} \mathbf{x}, \tag{3.14}$$

$$\lim_{t\to 0^-} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \lim_{t\to 0^+} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}, \tag{3.15}$$

$$\lim_{t\to 0^-} \frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2} = \lim_{t\to 0^+} \frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2}, \tag{3.16}$$

where $\mathbf{x}$ represents the mesh position defined with temporal-control meshes, and we want the mesh velocity and acceleration to be zero at the beginning of that temporal patch:

$$\left.\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}\right|_{t=t_s} = \mathbf{0}, \tag{3.17}$$

$$\left.\frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2}\right|_{t=t_s} = \mathbf{0}. \tag{3.18}$$

Remark 7 *Because this condition provides zero velocity and acceleration, we can compute with a static mesh prior to the computation in this temporal patch to develop the flow field.*

To achieve above conditions we use cubic B-spline functions defined by the knot vector $\{0,0,0,0,\frac{1}{2},1,1,1,1\}$. Because the last temporal-control mesh for the starting-condition patch is M_c^0, the condition given by Eq. (3.14) is automatically satisfied. The conditions given by Eqs. (3.17) and (3.18) can be satisfied by setting $M_c^{-4} = M_c^{-3} = M_c^{-2}$, with any set of corresponding control points for time: $t_c^{-4} \equiv t^s < t_c^{-3} < t_c^{-2}$. However, satisfying the conditions given by Eqs. (3.15) and (3.16) is not trivial.

The following is an example of using cubic B-splines. The derivatives of the mesh position at $t \to 0^+$ are as follows:

$$\left.\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}\vartheta}\right|_{\vartheta=\vartheta_1} = \frac{3}{\vartheta_5-\vartheta_2}\left(\mathbf{x}_c^1-\mathbf{x}_c^0\right), \tag{3.19}$$

$$\left.\frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}\vartheta^2}\right|_{\vartheta=\vartheta_1} = \frac{6}{\vartheta_5-\vartheta_3}\left(\frac{\mathbf{x}_c^2-\mathbf{x}_c^1}{\vartheta_6-\vartheta_3}-\frac{\mathbf{x}_c^1-\mathbf{x}_c^0}{\vartheta_5-\vartheta_2}\right), \tag{3.20}$$

where ϑ is parametric coordinate[1] defined on the first patch. Similarly, the derivatives of the time are as follows:

$$\left.\frac{\mathrm{d}t}{\mathrm{d}\vartheta}\right|_{\vartheta=\vartheta_1} = \frac{3}{\vartheta_5-\vartheta_2}t_c^1, \tag{3.21}$$

$$\left.\frac{\mathrm{d}^2t}{\mathrm{d}\vartheta^2}\right|_{\vartheta=\vartheta_1} = \frac{6}{\vartheta_5-\vartheta_3}\left(\frac{t_c^2-t_c^1}{\vartheta_6-\vartheta_3}-\frac{t_c^1}{\vartheta_5-\vartheta_2}\right). \tag{3.22}$$

Thus, the mesh velocity and acceleration can be determined as follows:

$$\lim_{t\to0^+}\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \frac{\mathbf{x}_c^1-\mathbf{x}_c^0}{t_c^1}, \tag{3.23}$$

$$\lim_{t\to0^+}\frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2} = \left(\frac{\mathrm{d}t}{\mathrm{d}\vartheta}\right)^{-2}\left(\frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}\vartheta^2}-\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}\frac{\mathrm{d}^2t}{\mathrm{d}\vartheta^2}\right)\Bigg|_{\vartheta=\vartheta_1}, \tag{3.24}$$

with Eqs (3.20)–(3.22). We note that if the temporal-control points are given by Eq. (3.8), the second term of Eq. (3.24) is zero.

[1]We note that the index of the knot vector starts from 1.

Similarly, the derivatives at $t \to 0^-$ are as follows:

$$\lim_{t\to 0^-} \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = -\frac{\mathbf{x}_c^0 - \mathbf{x}_c^{-1}}{t_c^{-1}}, \tag{3.25}$$

$$\lim_{t\to 0^-} \frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2} = \frac{1}{3\left(t_c^{-1}\right)^2} \left(\mathbf{x}_c^{-2} - 3\mathbf{x}_c^{-1} + 2\mathbf{x}_c^0 + \frac{t_c^{-2} - 3t_c^{-1}}{t_c^{-1}}\left(\mathbf{x}_c^0 - \mathbf{x}_c^{-1}\right)\right). \tag{3.26}$$

With that, there are many ways of satisfying the conditions given by Eqs. (3.15) and (3.16). Here we set $t_c^{-1} = -\Delta t_s$. We also define the control points for the time in such a way that the part of Eq. (3.26) related to $\frac{\mathrm{d}^2 t}{\mathrm{d}\theta^2}$ at $t \to 0^-$ is zero; i.e., $t_c^{-2} = 3t_c^{-1}$. Thus, we can set $\mathbf{x}_c^{-1}$ and $\mathbf{x}_c^{-2}$ as

$$\mathbf{x}_c^{-1} = \mathbf{x}_c^0 - \frac{\Delta t_s}{t_c^1}\left(\mathbf{x}_c^1 - \mathbf{x}_c^0\right), \tag{3.27}$$

$$\mathbf{x}_c^{-2} = \mathbf{x}_c^0 - 3\left(\frac{\Delta t_s}{t_c^1}\left(\mathbf{x}_c^1 - \mathbf{x}_c^0\right) - \Delta t_s^2 \lim_{t\to 0^+} \frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2}\right). \tag{3.28}$$

Furthermore we set $t_c^{-3} = -5\Delta t_s$ and $t_s = -6\Delta t_s$ to have an overall linear relationship between the time and the parametric space. We leave Δt_s as a free parameter that we can adjust for various purposes, such as having smaller acceleration and smaller displacement from M_c^0.

Remark 8 *In addition to using NURBS basis functions for the temporal representation of the motion and deformation of the solid surfaces and volume meshes computed, we can use NURBS as temporal basis functions in our ST computations. With that, we would have temporal advantages similar to the spatial advantages one would have by using NURBS as spatial basis functions.*

Chapter 4

Flapping-motion and geometry representation

In this chapter, we describe from [1] the techniques used for modeling the body and wings of the MAV and the techniques used for representing the motion and deformation of the flapping wings, with the data coming from the actual locust.

4.1 Wing and body shape

Based on a digital image of a pair of locust forewing (FW) and hindwing (HW), we construct the surface geometries of the FW and HW using a NURBS-based design software (see Figure 4.1). The FW and HW are modeled as a single NURBS patch with degeneration at the wingtip. There are 21 control points for the FW, and 51 for the HW (see Figure 4.2). in [1]. The HWs for the MAV are attached to the body through the entire chord length while the HWs for the locust computation presented in [1] are not (see Figure 4.3).

The MAV body is a simplified and streamlined version of the locust body. The body is symmetric along the sagittal plane. The body is made up of 20 NURBS patches, 10 patches in each sagittal half (this is the minimum number needed to

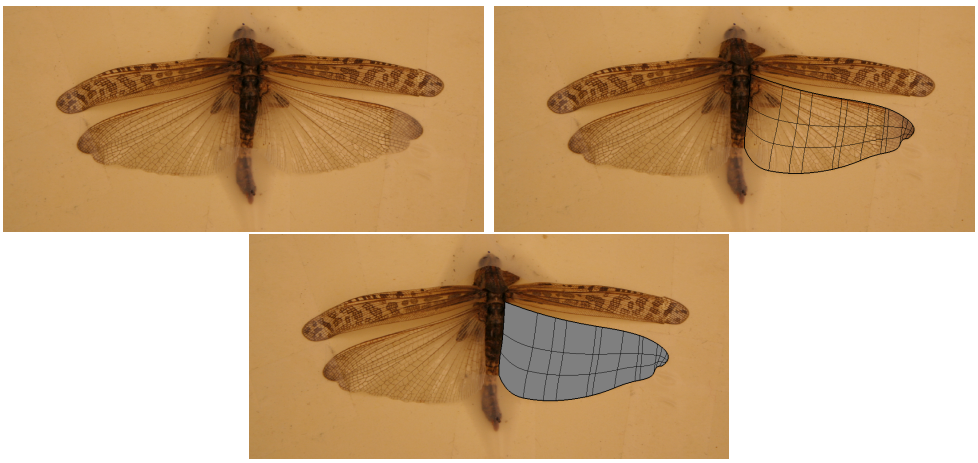


Figure 4.1: Wing model construction.

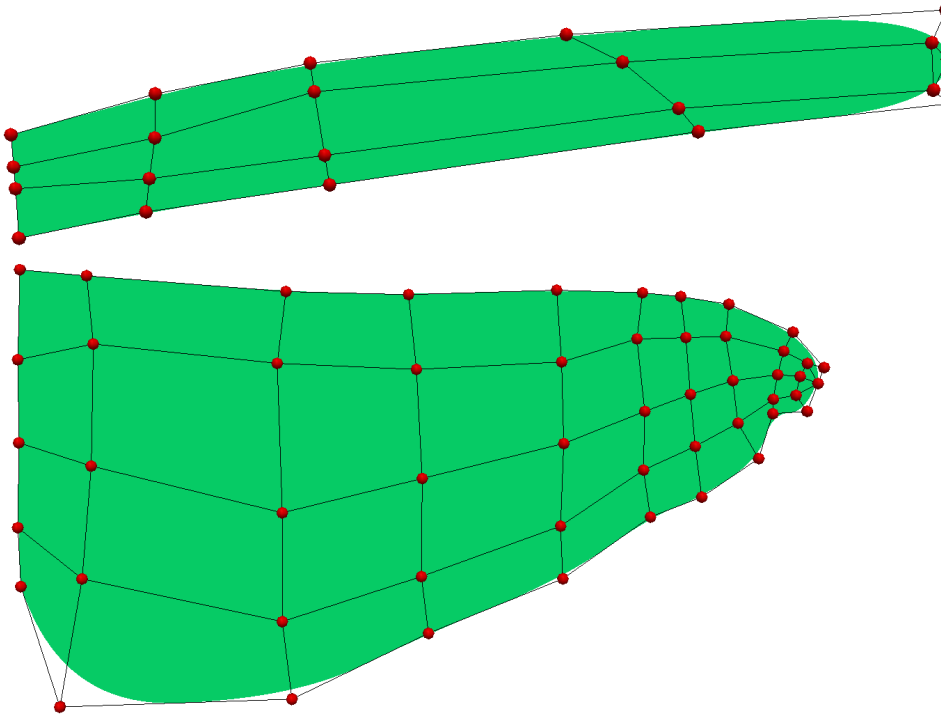


Figure 4.2: MAV FW and HW surfaces represented by NURBS and the corresponding control mesh.

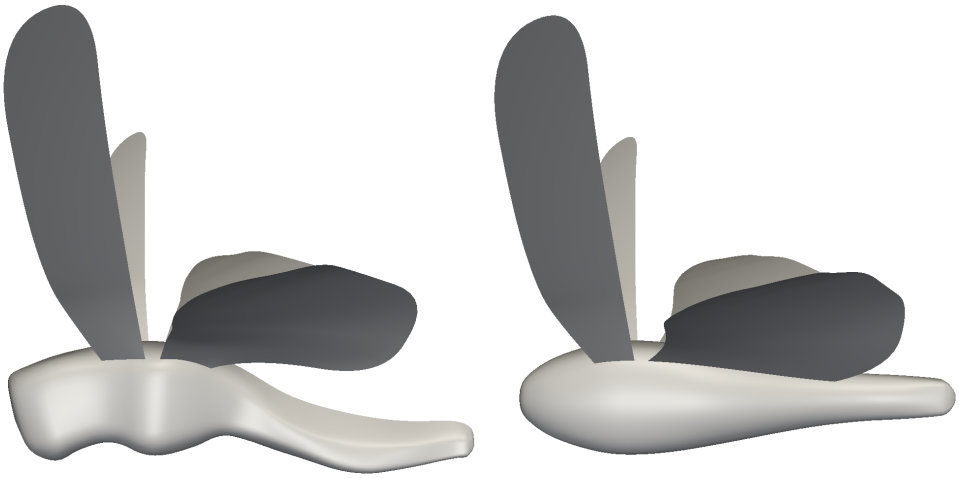


Figure 4.3: Locust body and wings, see [1], (left) and MAV body and wings (right).

attach the wings to the body). We shape the patches as shown in Figure 4.4. The

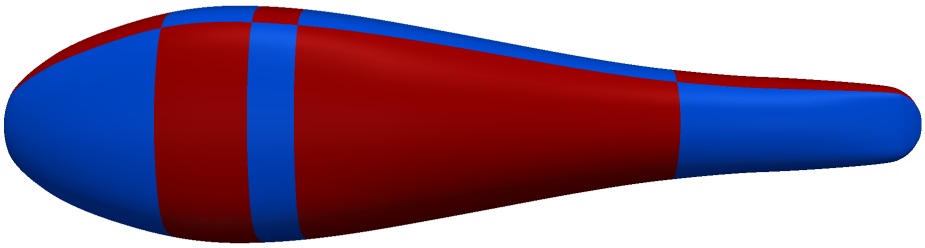


Figure 4.4: MAV body patches.

wing spatial-control meshes are deformed to the various positions in the flapping motion as we will describe in Section 4.2.

4.2 Flapping-motion representation

The wing motions we prescribe need to accurately represent the wing motion and deformation patterns during flight. Here we face the challenge of reconciling data acquired through photogrammetry with data that is suitable as an input for compu-

tational analysis. In this computation, we use a data set that is more recent than the one reported in [3]. Both data sets were provided by our collaborators at BCM. The location of the tracking points provided in this data set can be seen in Figure 4.5. By averaging the values corresponding to each other across the symmetry plane, we create a new data set of tracking points for one side. With the help of these tracking

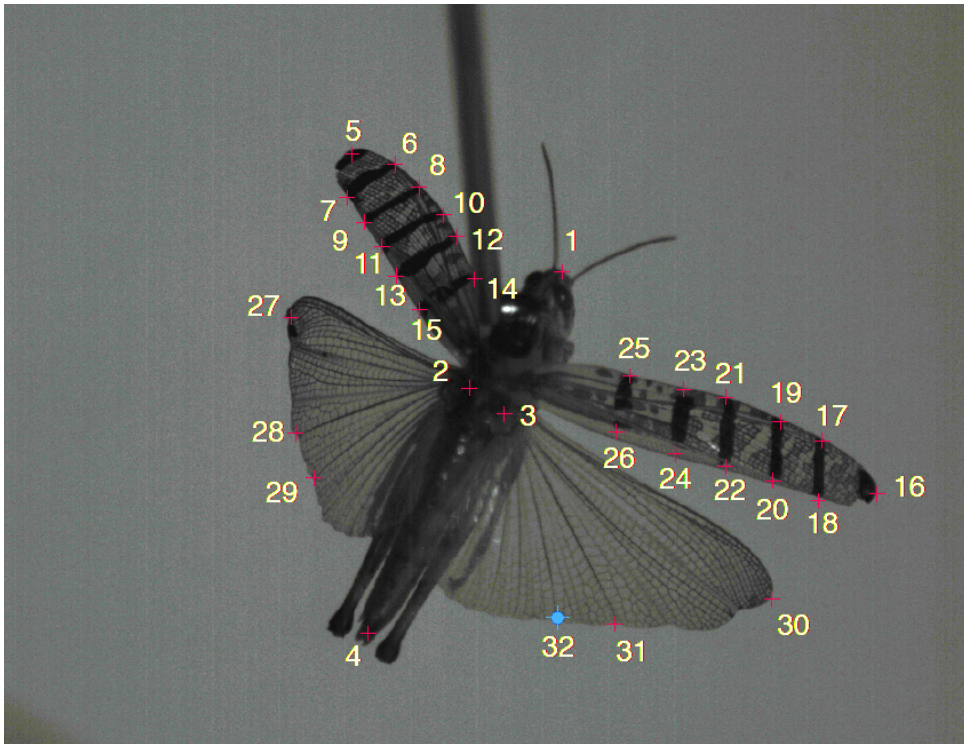


Figure 4.5: Tracking points in data set provided by BCM.

points, we generate additional "tracking points" to more closely represent locust flight characteristics observed in the video. After the body motion is remove, the motion of the wings is reflected across the sagittal plane of the MAV to create symmetric flapping motion. The total number of tracking points used in the computations is 68. We then use this symmetric data as the input for the least-squares projection used for the temporal representation. Next, we temporally interpolate our representative data set. For each tracking point, we apply a temporal NURBS representation as discussed in Section 3.1.2 and 3.1.3. Higher-order basis functions should be used in temporal interpolation to achieve a continuous acceleration. We use cubic B-spline

functions for temporal interpolation. Motion with cubic basis functions maintains C^2-continuity across knots. With this, acceleration will be continuous across temporal knots.

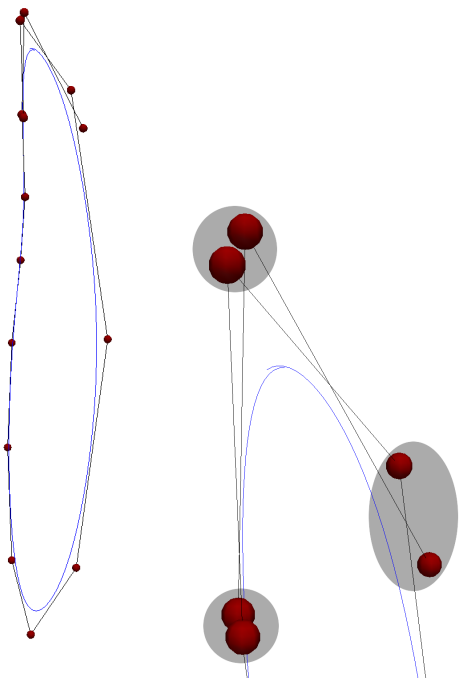
4.2.1 Generating a periodic data set

While flapping motion kinematics is inherently cyclical, it is not necessarily periodic. For example, the vertical position at the start of the first downstroke does not correspond to exactly the same position for subsequent strokes. This adds to the difficulty of computing flapping-flight aerodynamics. It is beneficial to have a periodic data set for a single flap cycle, where the first and last points of the cycle are colocated and have the desired continuity. Thus, a single set of deformed meshes can be appended to produce as many flapping cycles as are required. Because of the periodicity of the motion, we can compute a desire number of flapping cycles that we think would be sufficient to reach a solution that we would consider periodic.

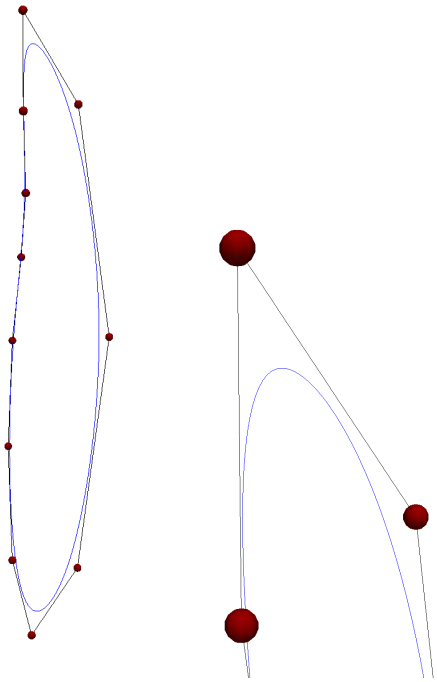
To obtain a periodic data set, after the least-squares projection, we extract one flapping cycle by averaging and inserting knots at the top of the HW cycle (end of the upstroke and beginning of the downstroke). To maintain continuity, the control points corresponding to the knot at the beginning of the flap cycle are colocated with the control points corresponding to the knot at the end of the flap cycle (for a cubic B-spline, three control points correspond to a given knot). To obtain such repetition we average those control points. Finally, we insert knots to extract a single cycle. We show the averaging process for a cubic B-spline in Figure 4.6.

4.2.2 Motion of the wings

We spatially represent the tracking points at each temporal-control point. Spatial interpolation is accomplished using the SSDM described in Section 3.2. The FW and HW SS consist of 6 and 9 control points, respectively.



(a) Flap cycle of interest before averaging control points (left) and enlarged view of top of cycle (right). The control points we average are highlighted.



(b) Flap cycle of interest after averaging control points (left) and enlarged view of top of cycle (right).

Figure 4.6: Process used for building a periodic flapping cycle.

Chapter 5

Base MAV computation

In this chapter, we describe the MAV computations where the wing deformation patterns come from the actual locust. These deformation patterns and the motion of the wings are prescribed as described in Chapter 4. In the rest of the chapter, from [82], we describe the setup for the computation, mainly the surface and volume meshes and the mesh update technique, and report the aerodynamic forces generated.

5.1 Surface and volume meshes

After the wing spatial-control surfaces are deformed to the various positions in the flapping motion as described in Section 4.2.2, we discretize the surfaces at each temporal-control point with triangular elements. The triangular surface mesh used in the computation is shown in Figure 5.1. We note that both FW and HW have a finite thickness of 1% of the FW root chord, which tapers to zero thickness at the wing edges.

We generate the tetrahedral element volume mesh with the following steps. First, we generate a one-layer refinement region near the wing surface. The element height of this layer is 10% of the FW root chord. In addition, we have a cylindrical region of increased element refinement around the MAV. We also specify increased volume-

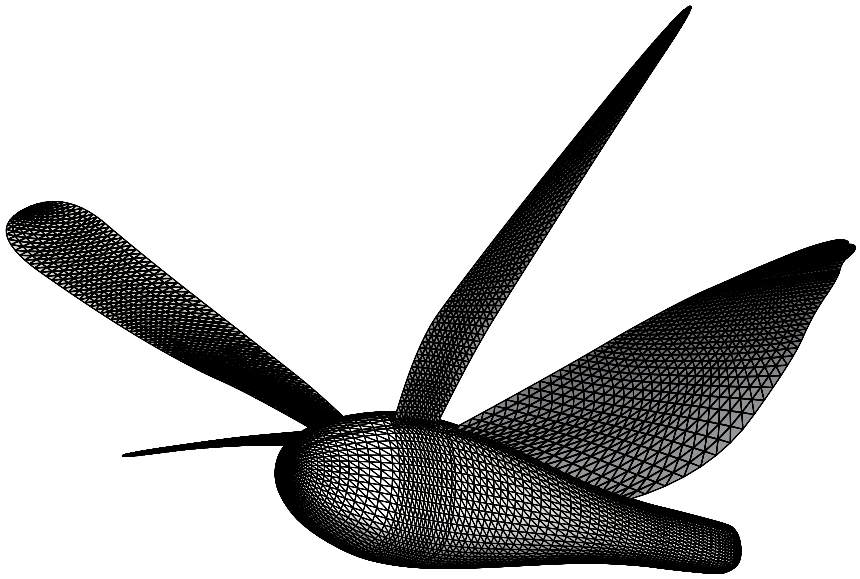


Figure 5.1: MAV wing and body surface meshes with triangular elements.

mesh refinement in the region between the FW and HW. This refinement reduces the mesh distortion due to the staggered motion of FW and HW. The volume mesh within this cylindrical region is then generated using an automatic mesh generator. We rotate the cylindrical region to an angle representing the approximate free-flight body angle of the locust and compute mesh motion only in this cylindrical region as discussed in Section 4.2. A volume mesh between the cylindrical region and the external domain boundaries is then generated with an automatic mesh generator. This region is fixed for the entire computation; therefore, it is used for all temporal patches. The volume mesh boundaries and cylindrical refinement region are shown in Figure 5.2.

5.2 Mesh update technique

Because of the topological changes caused by the two-wing motion, we cannot use the same connectivity during the entire computation. We know the wing motion in advance, so we use a "prepared" remeshing technique.

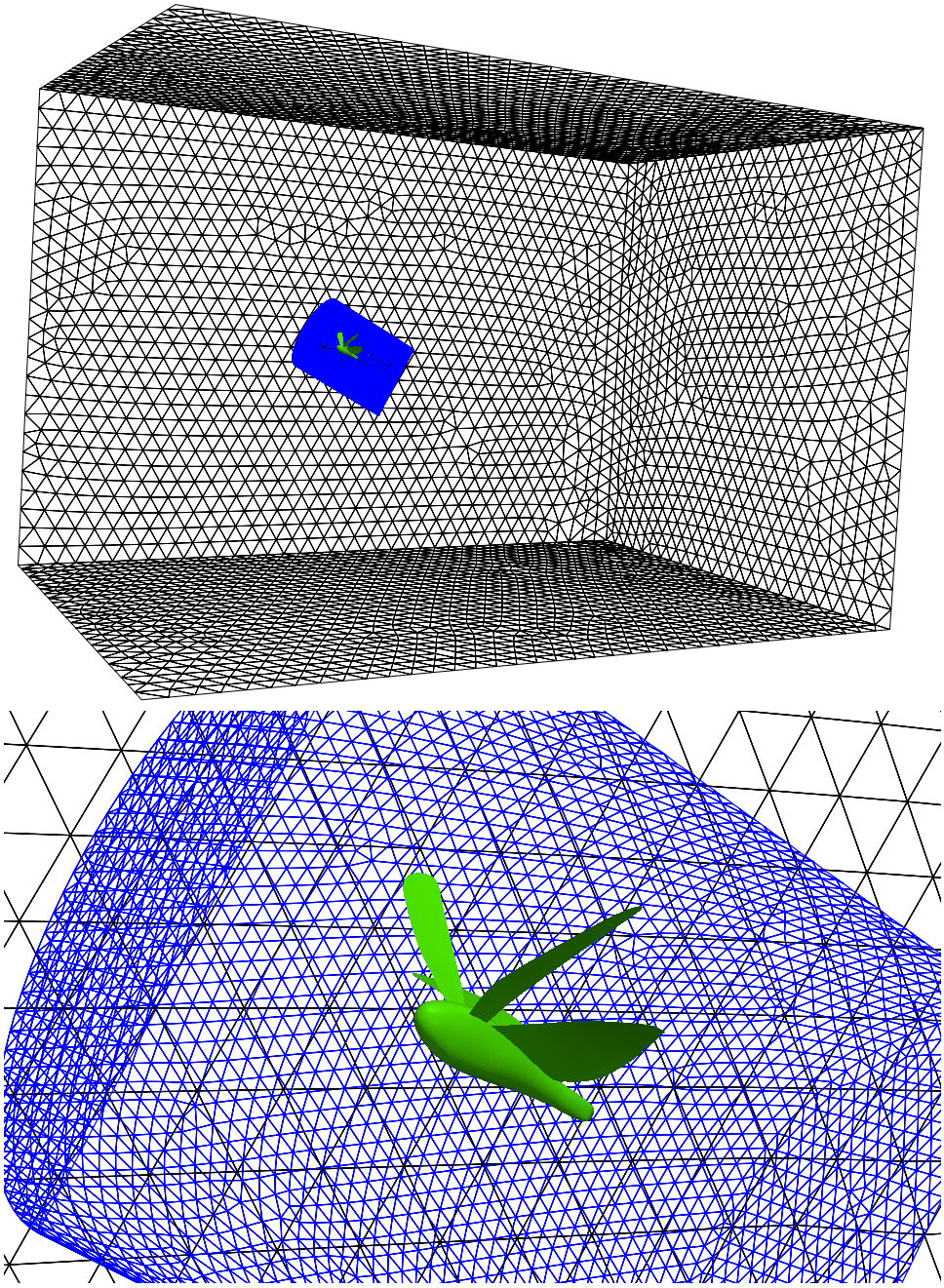


Figure 5.2: Surface meshes on some of the external computational boundaries and on the outer surface of the inner, cylindrical mesh region, which has been rotated to account for in-flight body angle.

We have a surface modeling using a temporal NURBS representation. First we decide the remeshing instants. Then, we use knot insertion to split the flapping cycle into patches as seen in Section 3.3.2. Those patches are called "temporal patches." The number of nodes and elements in the total volume mesh varies between each temporal patch. The average number of nodes and elements are 0.35 million and 2.1 million (see Table 5.1). Due to the relatively large change in deformation between

Temporal Patch	Control Points	Knot Spans
1	7	4
2	6	3
3	5	2
4	5	2

Temporal Patch	Meshing Point	Number of Nodes	Number of Elements
1	4	347,312	2,072,463
2	3	342,501	2,043,814
3	3	334,840	1,998,235
4	3	385,900	2,303,385

Table 5.1: Summary of computational setup. In each temporal patch we indicate the number of temporal control points and at which point we generate the tetrahedral volume mesh with the number of nodes and elements shown.

each temporal-control point, we use subiterations for the mesh computation to divide the steps between temporal-control points into 20 smaller steps. We move the mesh, which corresponds to the middle control point, backward and forward through each smaller step using 1,500 GMRES [114] iterations (this is the technique described in Section 3.3.1).

5.3 Computation of the base case

5.3.1 Computational conditions

Figure 5.3 shows the length scales involved in the model used in the computations.

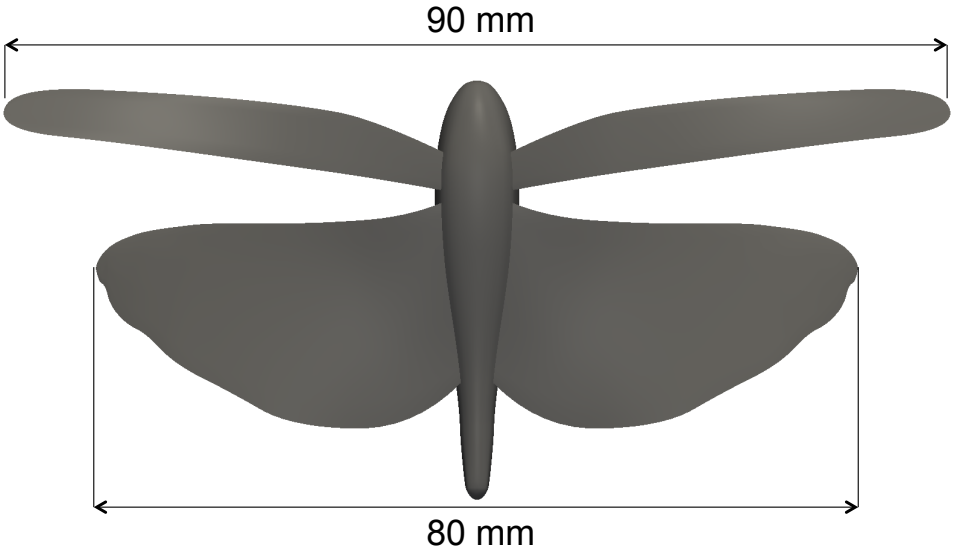


Figure 5.3: Length scales involved in the model used in the computations.

The air density and kinematic viscosity are 1.2251 kg/m^3 and 1.4606$\times$10^{-5} m^2/s. The period of flapping is $T = 0.047$ s. Prior to the beginning of the prescribed flapping motion, we compute 400 time steps to develop the flow field. Over the first 300 time steps of this computation, we use a Cosine form to smoothly increase the inflow velocity from 0.0 to 2.4 m/s, which represents the average wind tunnel velocity used for the empirical data collection. For the last 100 time steps we keep the velocity steady at 2.4 m/s. In this flow-development computation, the time step size is 1.71$\times$10^{-4} s, with 4 nonlinear iterations per time step. We use the ST-SUPS and ST-VMS techniques for the first two and last two nonlinear iterations, respectively, with the stabilization parameters as given by Eqs. (7)–(11) in [8] for $\tau_{\rm M} = \tau_{\rm SUPG}$ (which are also in Section 2.3) and Eqs. (35)–(37) in [3] for $\nu_{\rm C}$. The stabilization parameters are calculated after the predictor step and after the first two nonlinear iterations. The number of GMRES iterations for the nonlinear iterations are 30, 60, 120, and 180.

For the computation with flapping, we use 25 ST slabs (with linear basis functions) for each of the knot spans in the temporal representation of the mesh, and that gives us the time-step size of 1.71$\times$10^{-4} s. The rest of the computational parameters are

the same as those above.

Remark 9 *We have seldom observed close-to-zero or negative diagonal terms when the time-step size is large. This occurs when the fine-scale velocity is much larger than the coarse-scale (discrete) velocity. We believe this is due to the predictor, which assumes all velocities are the same from the previous time step except those with Dirichlet conditions.*

We partition this mesh to enhance parallel efficiency. Mesh partitioning is based on the METIS [115] algorithm. We partition it into 128 parts and use 128 parallel processes.

5.4 Results

We compute the problem for three complete flapping cycles. We display results for the third cycle. The lift and thrust (pressure component only) are shown in Figures 5.4–5.6 for the MAV base case. Figures 5.7–5.8 show the pressure on the wing surfaces

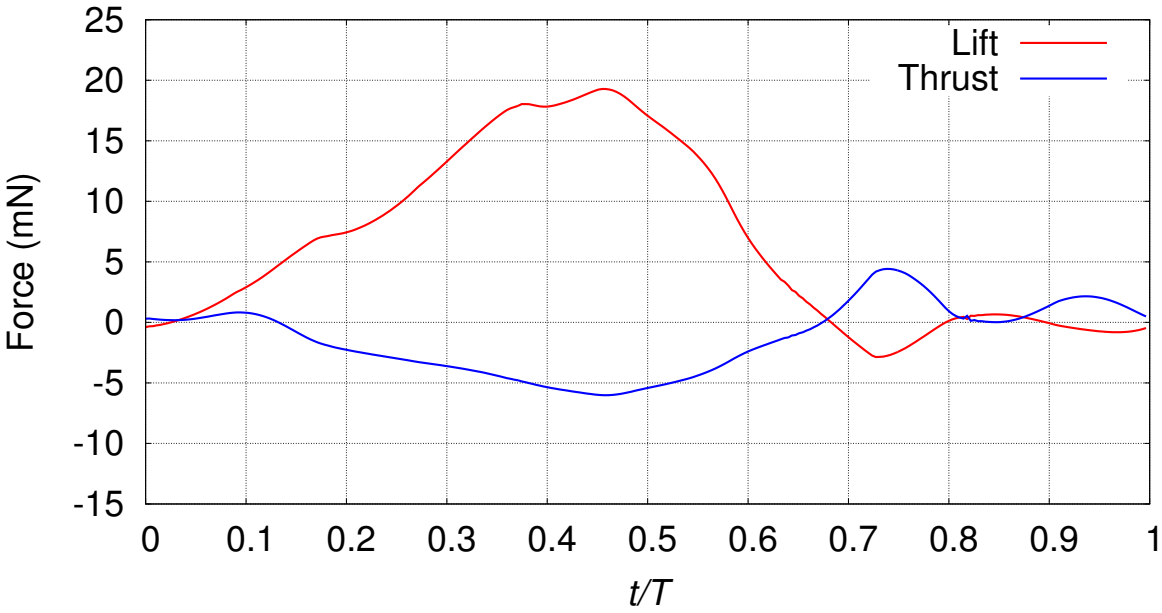


Figure 5.4: Total lift and thrust generated over one cycle. Positive value indicates that thrust exceeds drag.

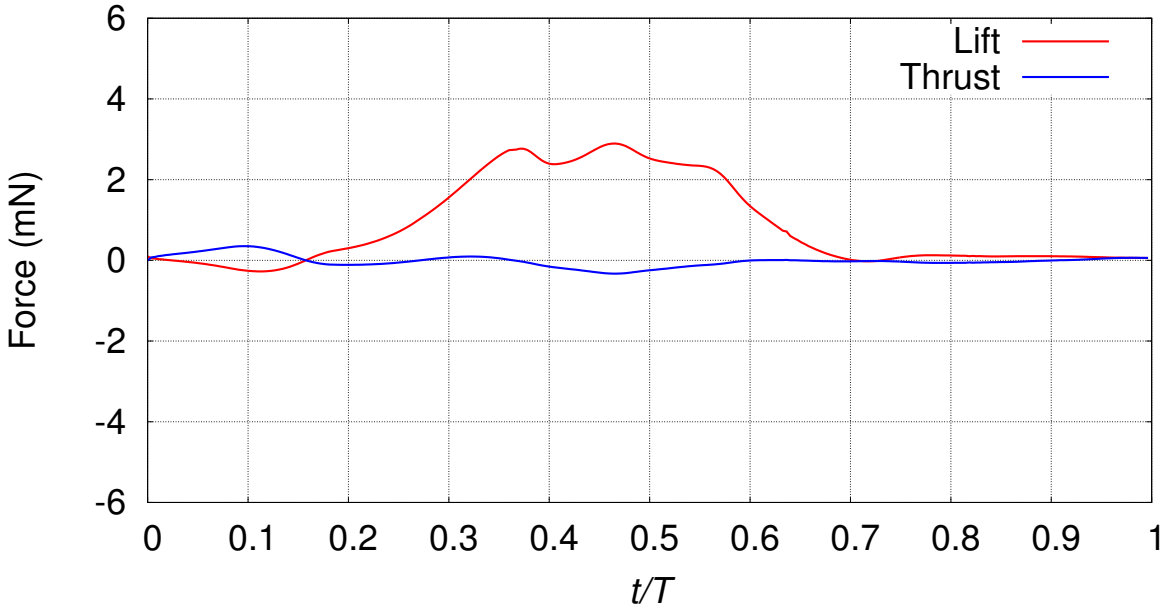


Figure 5.5: Lift and thrust generated on the right FW.

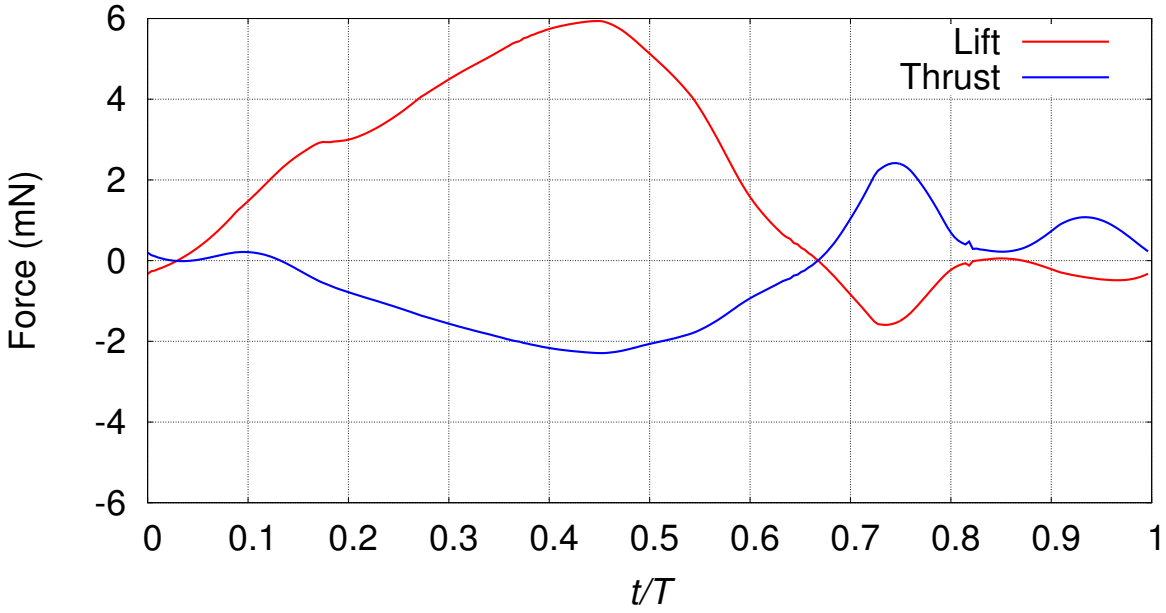


Figure 5.6: Lift and thrust generated on the right HW.

(relative to the free-stream pressure) at different instants during the third flapping cycle. Figure 5.9 shows the vorticity magnitude at different instants.

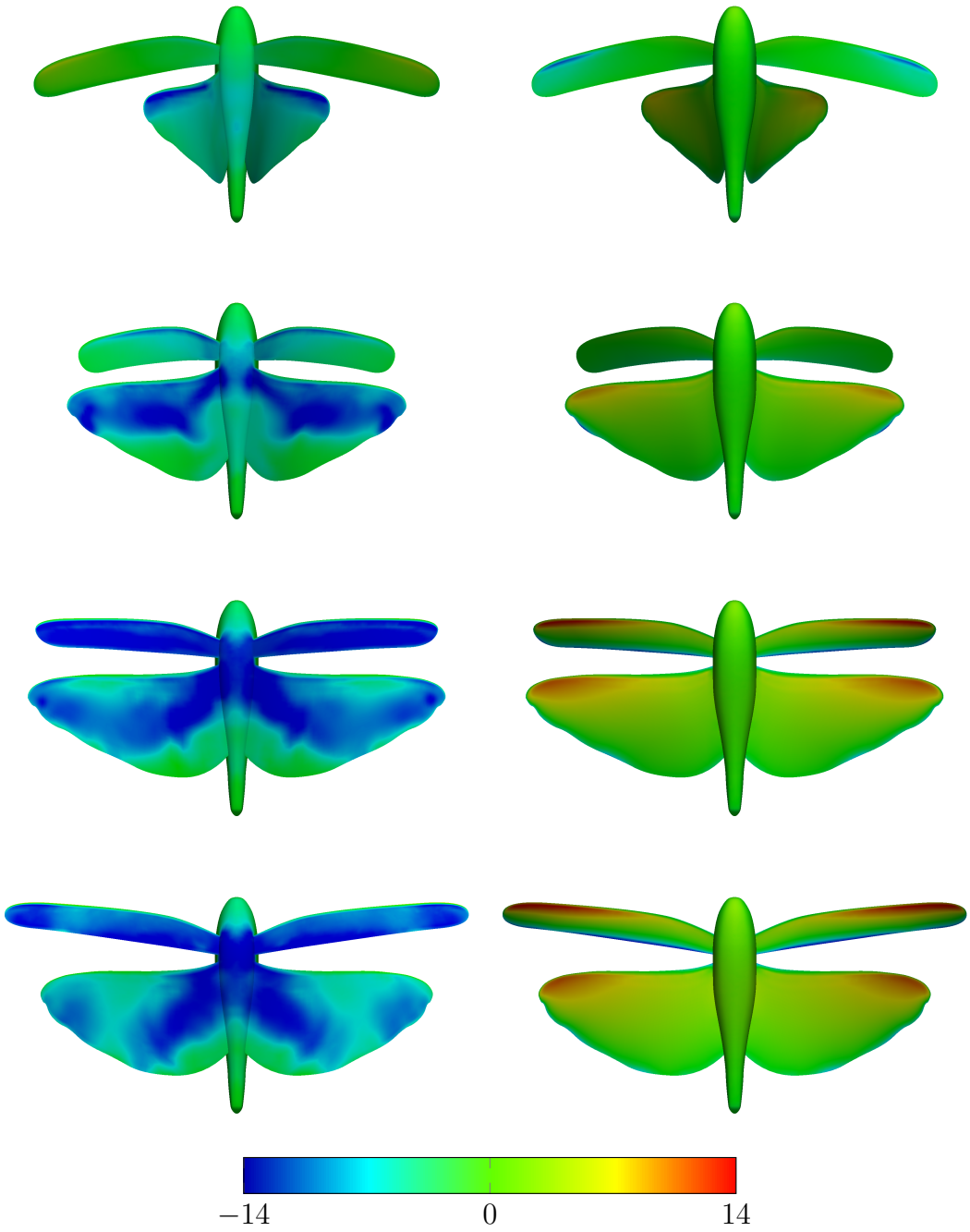


Figure 5.7: Surface pressure in Pa (relative to the free-stream pressure) at the first four of the eight equally-spaced points during the third flapping cycle (top view on left, bottom view on right).

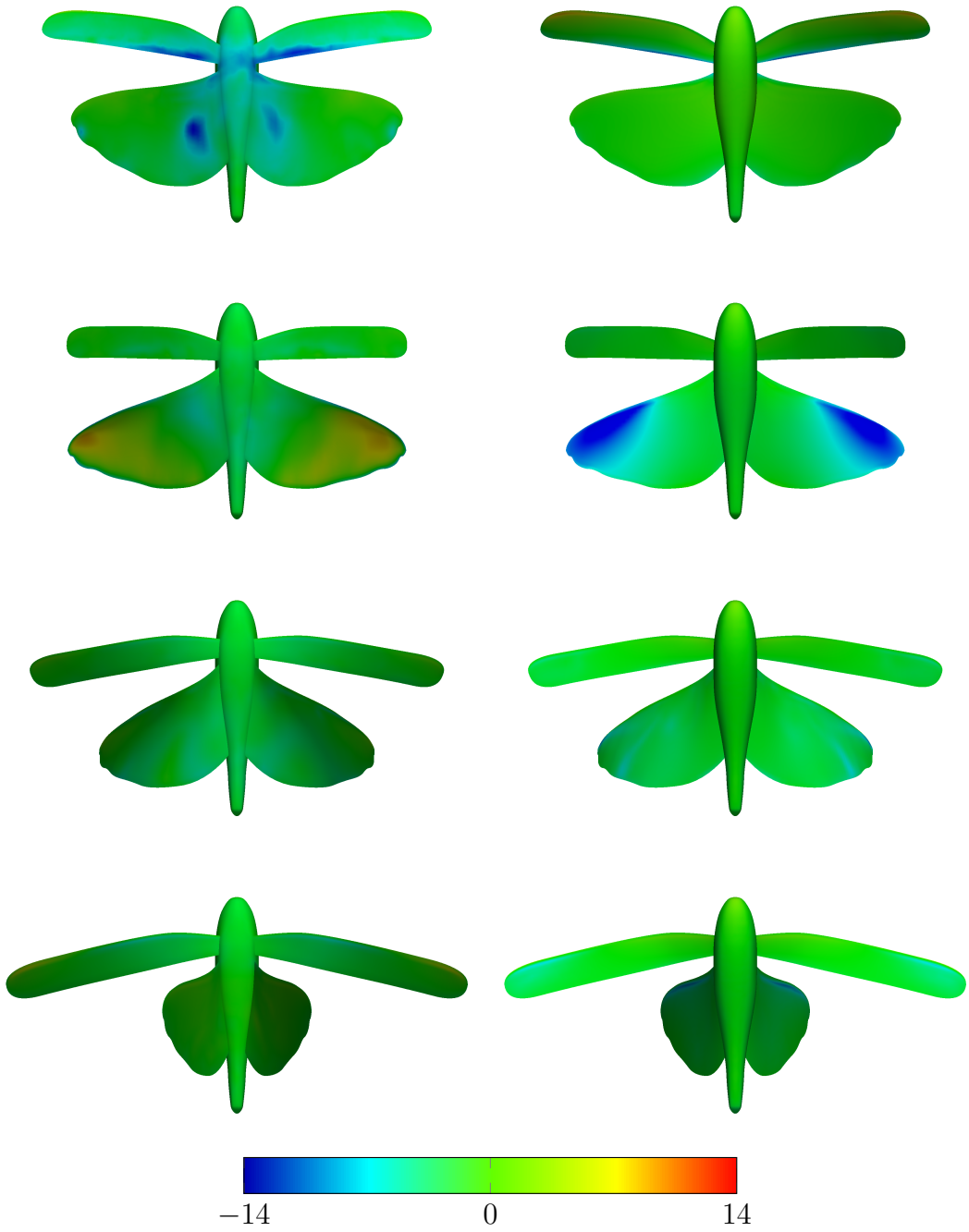


Figure 5.8: Surface pressure in Pa (relative to the free-stream pressure) at the last four of the eight equally-spaced points during the third flapping cycle (top view on left, bottom view on right).

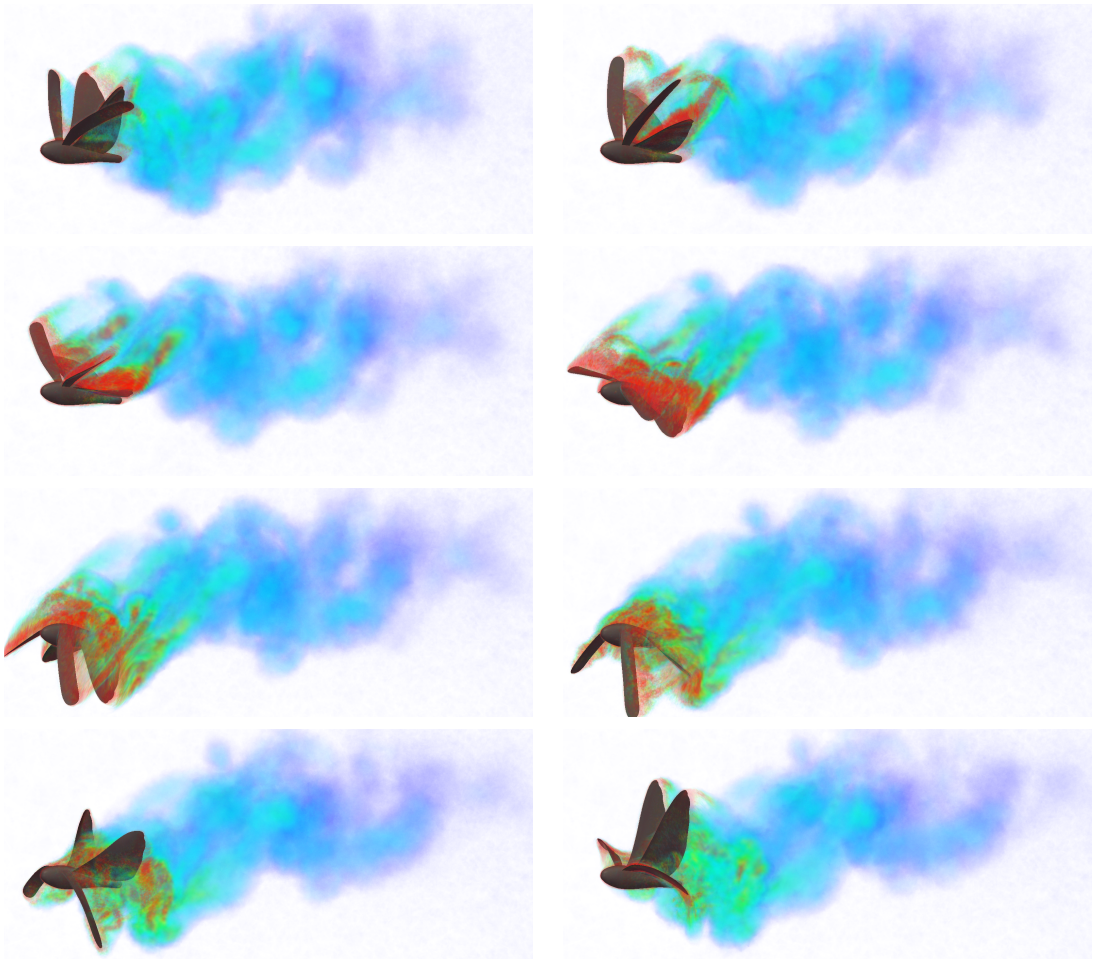


Figure 5.9: Volume rendering of the vorticity magnitude for the eight equally-spaced points during the third flapping cycle (left to right and then top to bottom).

Chapter 6

SCFSI computation

In this chapter, we describe from [2] the SCFSI computations for the MAV. We do the structural mechanics computation only for the HW, as it generates most of the lift and thrust. We use the Kirchhoff–Love shell model. The FW has prescribed motion and deformation as described in Chapter 5.

6.1 Setup and computations

To make the structural mechanics computations less challenging, instead of the complex shape for the HW, we use the SS. We use a higher spatial refinement for the SS than the one in [82]. It has 16 control pointed and is represented with quadratic NURBS in both directions (there were 9 control points in [82]). The SS for the FW has the same number of control points (6) as in [82]. Figure 6.1 shows the spatial-control mesh for the HW SS.

We start with a structural mechanics computation with no fluid mechanics forces applied. We prescribe the leading-edge motion of the wing. We can use either a desired leading-edge motion or the original leading-edge motion from the locust data, which is what we do here. We note that the SS leading-edge data is represented temporally by cubic NURBS and can easily be altered while maintaining the smooth

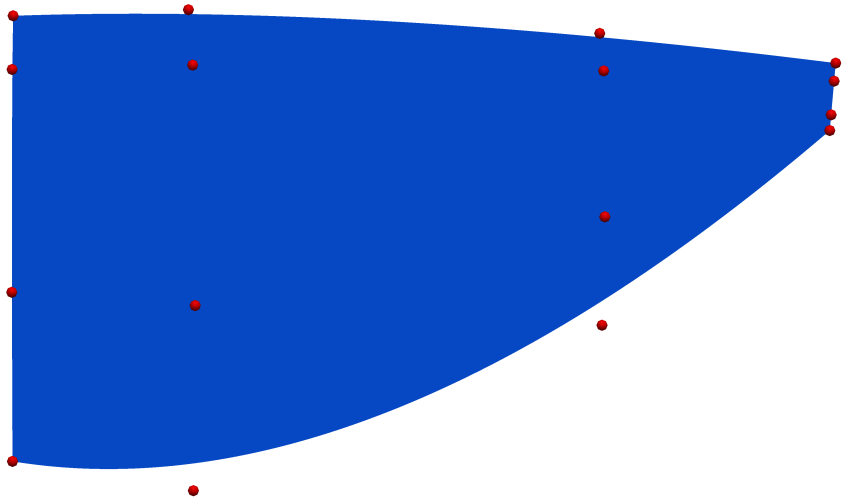


Figure 6.1: Spatial-control mesh for the HW SS.

motion and continuous acceleration. The structural mechanics material properties are based on measured values of the actual locust wing material properties, with the values altered to take into account the homogeneous modeling of the wing rather than using the actual wing morphology [116]. The Young's modulus is 4.3 GPa, the density is 560 kg/m^3, and the thickness is 1.1×10^{-3} m.

In the spatial discretization with quadratic NURBS in each direction and 16 control points (see Figure 6.1), we have 3 quadrature points in each direction, and the mass matrix is consistent. In the temporal discretization, we use the generalized-α method [112]. The parameters we use with the method, in the notation of [11], are $\alpha_\mathrm{m} = 2$, $\alpha_\mathrm{f} = 1$, $\gamma = 1.5$, and $\beta = 1$. These values result in a spectral radius of zero while maintaining second-order accuracy in time. The acceleration predictor is

$$\mathbf{a}_{n+1}^0 = \mathbf{a}_n, \tag{6.1}$$

which, because of the Newmark-velocity formula, leads to the following displacement

and velocity predictors:

$$\mathbf{y}^0_{n+1} = \mathbf{y}_n + \mathbf{u}_n \Delta t + \frac{1}{2}\mathbf{a}_n \Delta t^2, \tag{6.2}$$

$$\mathbf{u}^0_{n+1} = \mathbf{u}_n + \mathbf{a}_n \Delta t. \tag{6.3}$$

The time-step size, 1.71×10^{-4} s, is the same as that of the fluid mechanics computations, and the number of nonlinear iterations at each time step is 7. The number of GMRES iterations for each nonlinear iteration is sufficiently high to make the solution process equivalent to a direct one. The preconditioner is nodal-block-diagonal. We compute three full flapping cycles. We use the motion and deformation from the third cycle in the fluid mechanics computation. This computed data replaces the original tracking-point data for the locust HW, and the rest of the setup process, mesh generation and moving, and computational parameters are almost the same as those described in Section 5.3. The number of GMRES iterations for the nonlinear iterations are a little different: 30, 60, 120, and 200, and the preconditioner is nodal-block-diagonal, instead of diagonal. Also, we split the flapping cycle into five temporal patches instead of four, because the increased complexity of the wing deformation patterns requires one additional remesh. As shown in [82], increasing the number of temporal patches beyond four does not result in a significant change in the aerodynamic forces. Figure 6.2 shows the wing motion and deformation used in the fluid mechanics computation of the first SCFSI iteration. We do the fluid mechanics computation for three flapping cycles.

In the second SCFSI iteration, the structural mechanics computation is carried out with the fluid mechanics forces applied from the fluid mechanics computation for the third flapping cycle of the first iteration. We calculate the pressure difference between the top and bottom surface for each node of the wing at each time step. Figure 6.3 shows this net force distribution at different instants during the third flapping cycle.

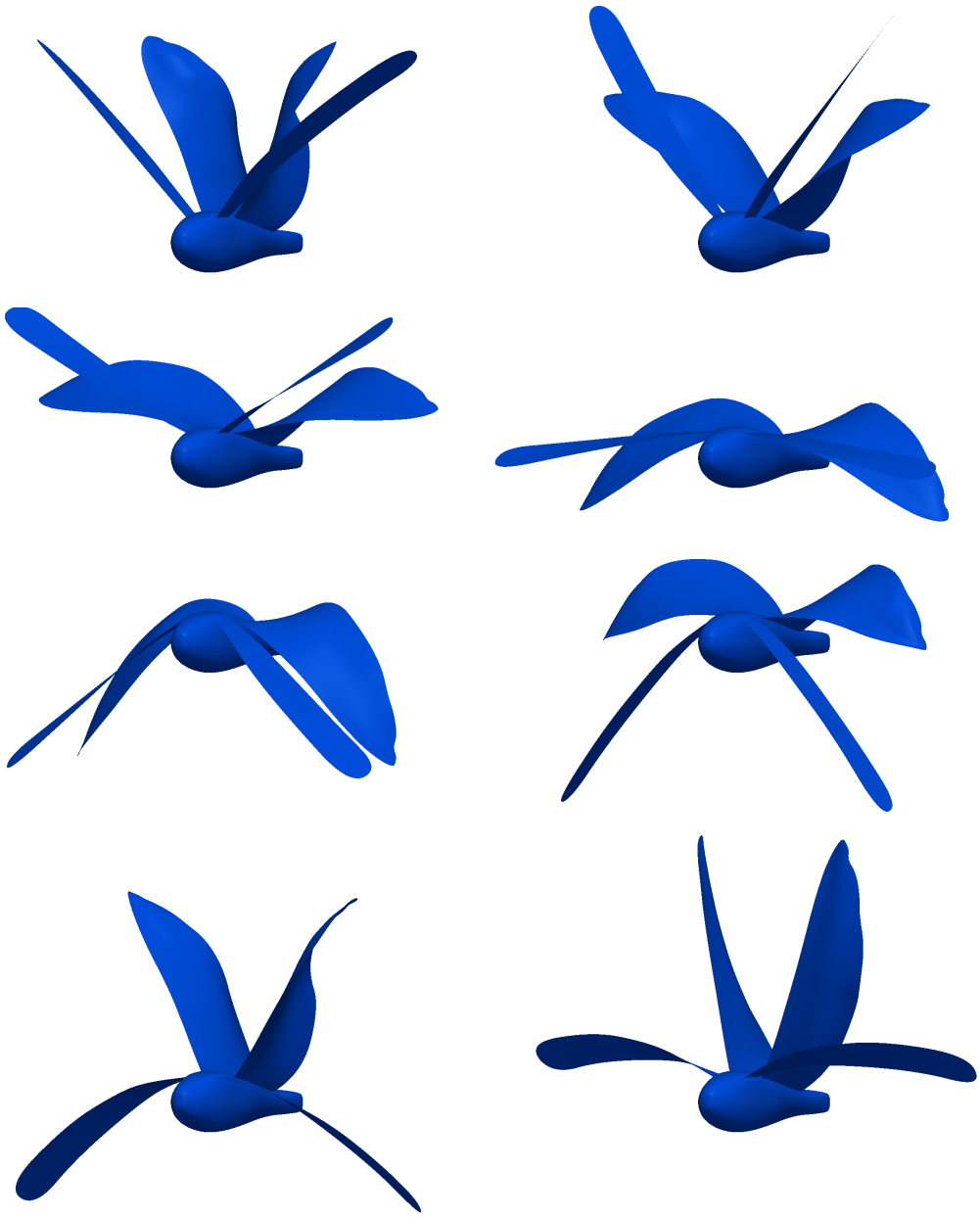


Figure 6.2: MAV with the HW deformation extracted from the structural mechanics computation, seen at eight equally-spaced instants during the third flapping cycle.

We project this force distribution to a temporal representation with cubic NURBS. Then we project that from the complex shape used in the fluid mechanics computation to the SS used in the structural mechanics computation. We can discretize that in time to be used in the structural mechanics computations. Temporal representation of the force with NURBS allows us to discretize in time with any time-step size, without loosing the smoothness of the force distribution data. We perform four SCFSI iterations where this process is repeated.

6.2 Results

We present the results for each SCFSI iteration compared to the base case. Figure 6.4 shows the total lift and thrust. Figures 6.5 and 6.6 show the lift and thrust generated by the HW and FW.

Even though the FW motion is not different from what it was in the base case, because of the interaction with the HW, the force the FW experiences is different. We note that there is not much change from the third iteration to the fourth, and not even from the second iteration to the third, and therefore we will stop the SCFSI iterations at three in the computations presented in the rest of the paper.

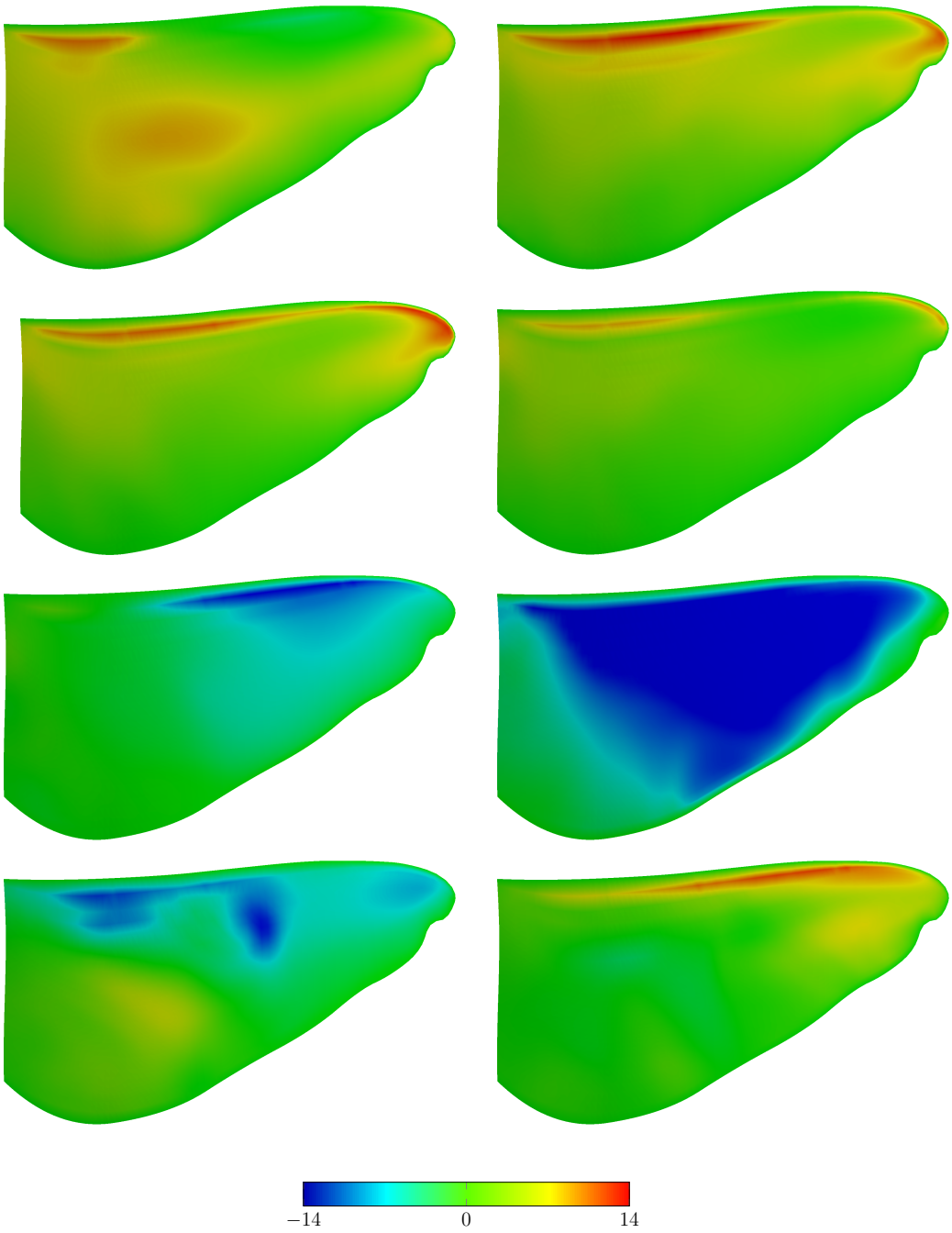


Figure 6.3: Net force distribution (in Pa) at eight equally-spaced instants during the third flapping cycle.

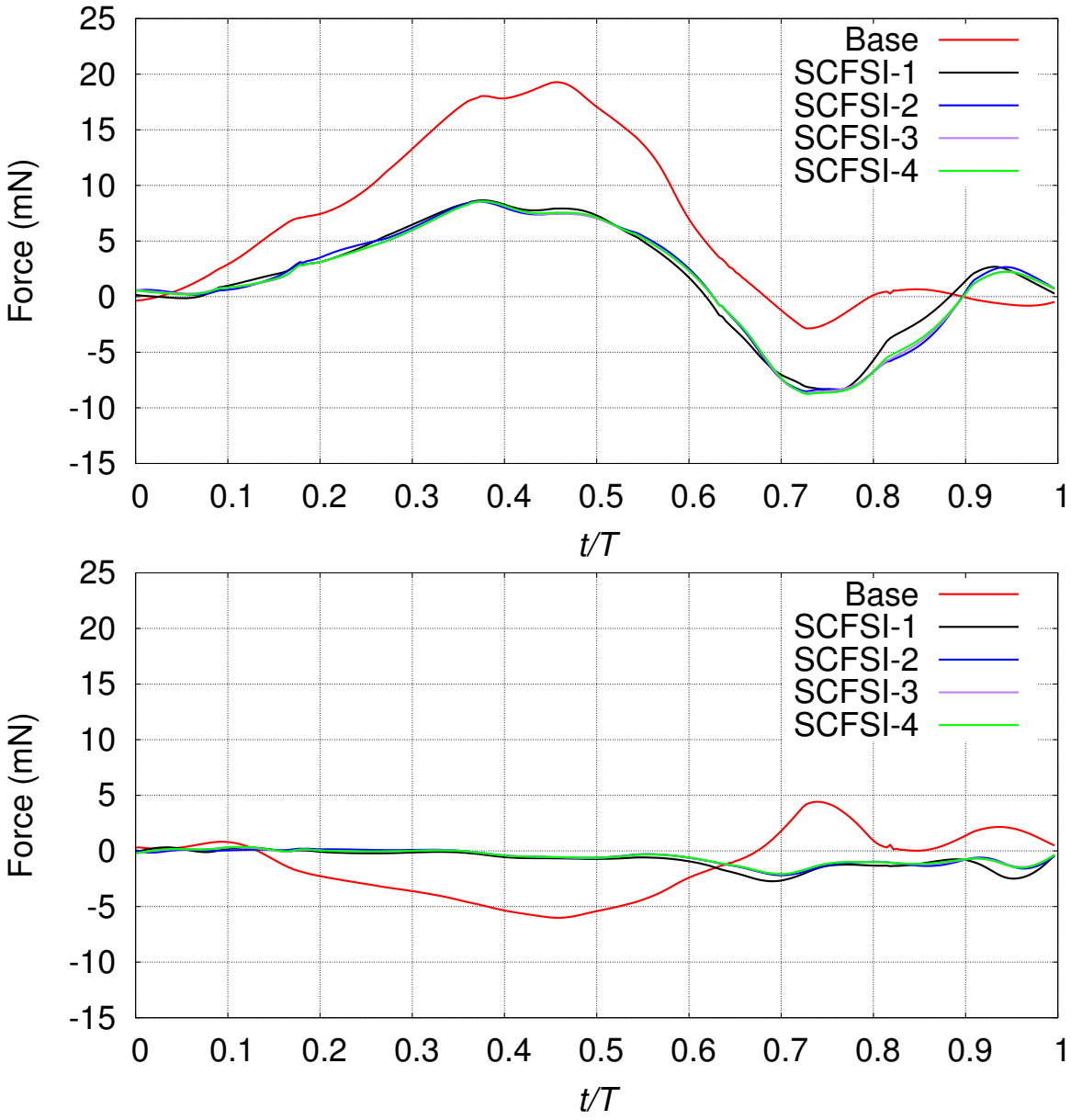


Figure 6.4: Total lift (top) and thrust (bottom) at different SCFSI iterations.

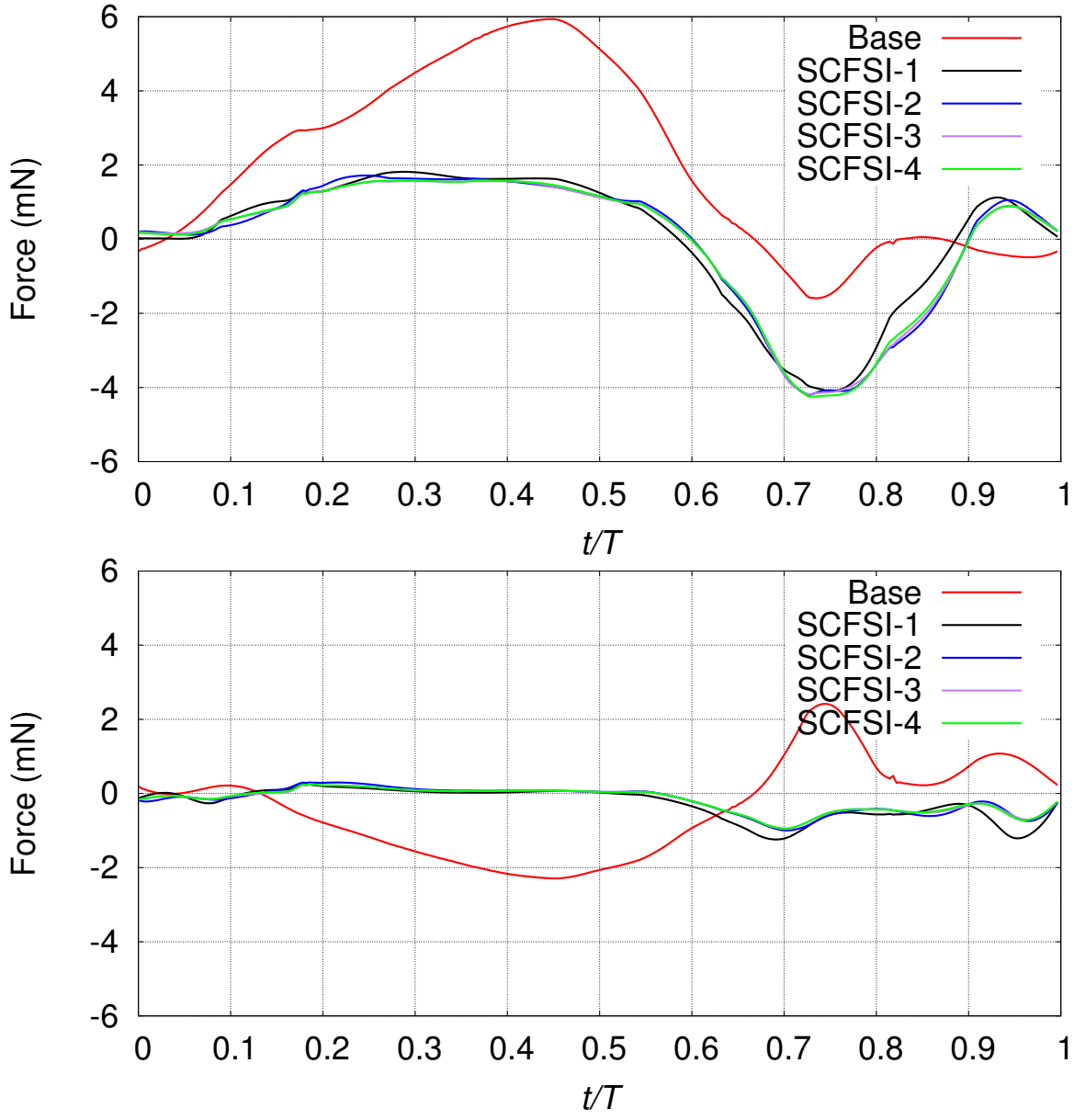


Figure 6.5: Right HW lift (top) and thrust (bottom) at different SCFSI iterations.

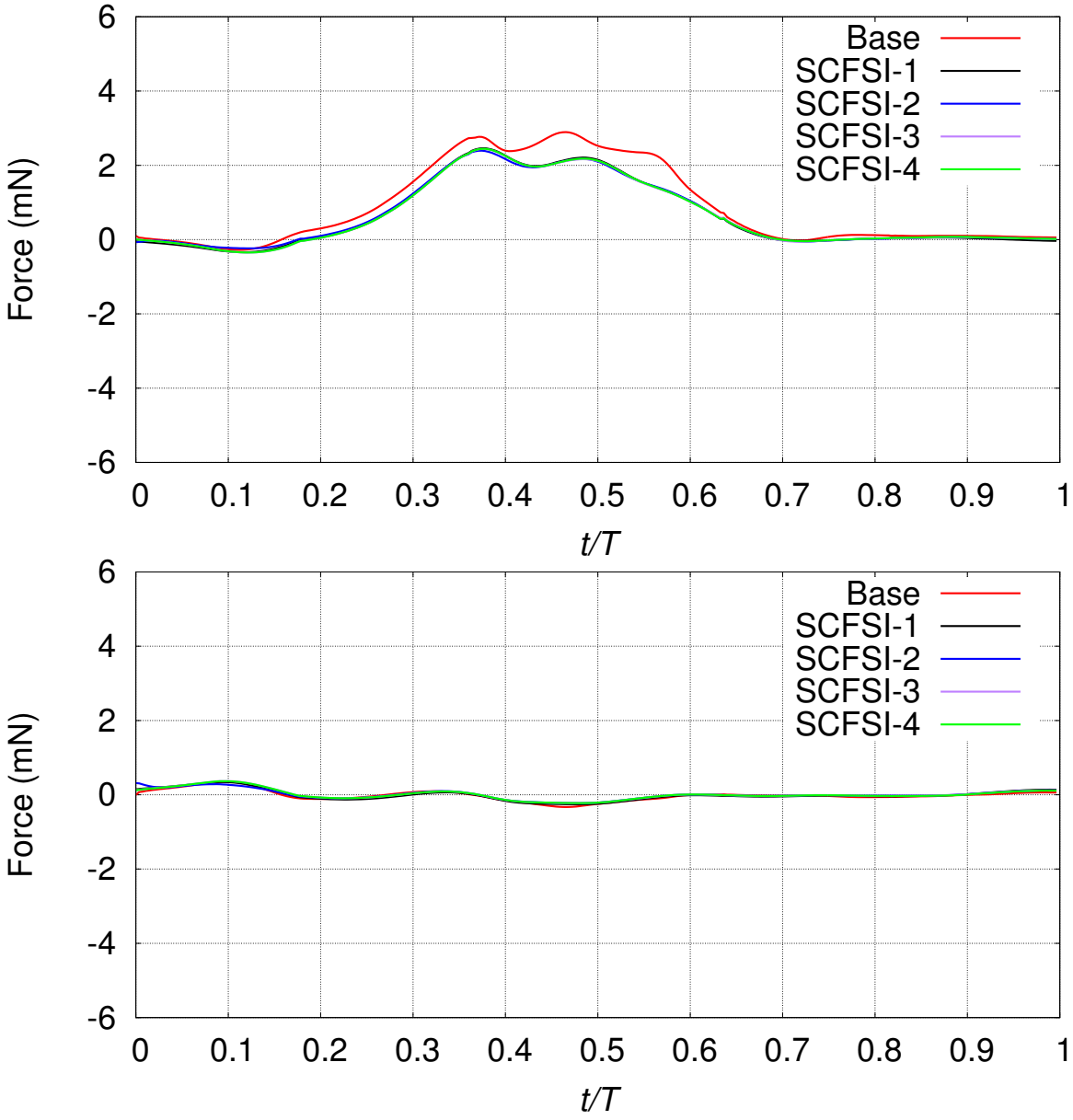


Figure 6.6: Right FW lift (top) and thrust (bottom) at different SCFSI iterations.

Chapter 7

Membrane model

It is interesting to investigate what effect the structural model has on the performance of the flapping-wing MAV. In this chapter, from [2], we describe the computations with the membrane model. This is conceivable if the MAV wings are made of very thin flexible material (on the order of micrometers) with negligible bending stiffness. The leading edge, with its motion prescribed, behaves as a beam-like structural member. In an actual flapping-wing MAV, such a design is quite plausible.

7.1 Setup and computations

With the membrane model we use the same procedure and computational settings as with the shell model in Chapter 6. We also use the same material properties. After three cycles of structural mechanics computation for the first SCFSI iteration, we proceed as before with three cycles of fluid mechanics computation. Figures 7.1 and 7.2 show, for the shell and membrane models, the wing motion and deformation used in the fluid mechanics computation of the first SCFSI iteration. As can be seen, the membrane model results in larger deformations. We perform three SCFSI iterations, and present the results from that.

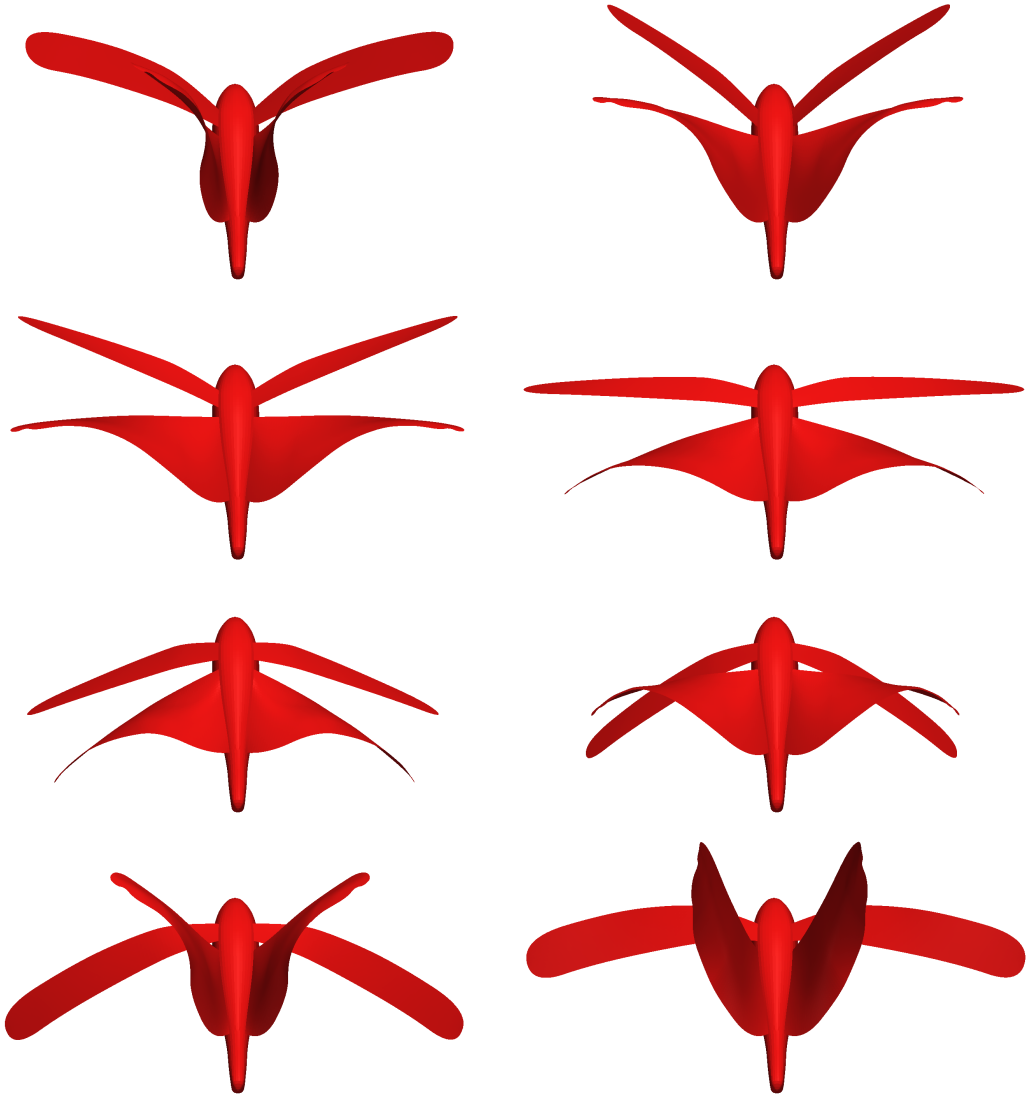


Figure 7.1: Top view of the MAV with the HW deformation extracted from the structural mechanics computation, seen for the shell model at eight equally-spaced instants during the third flapping cycle.)

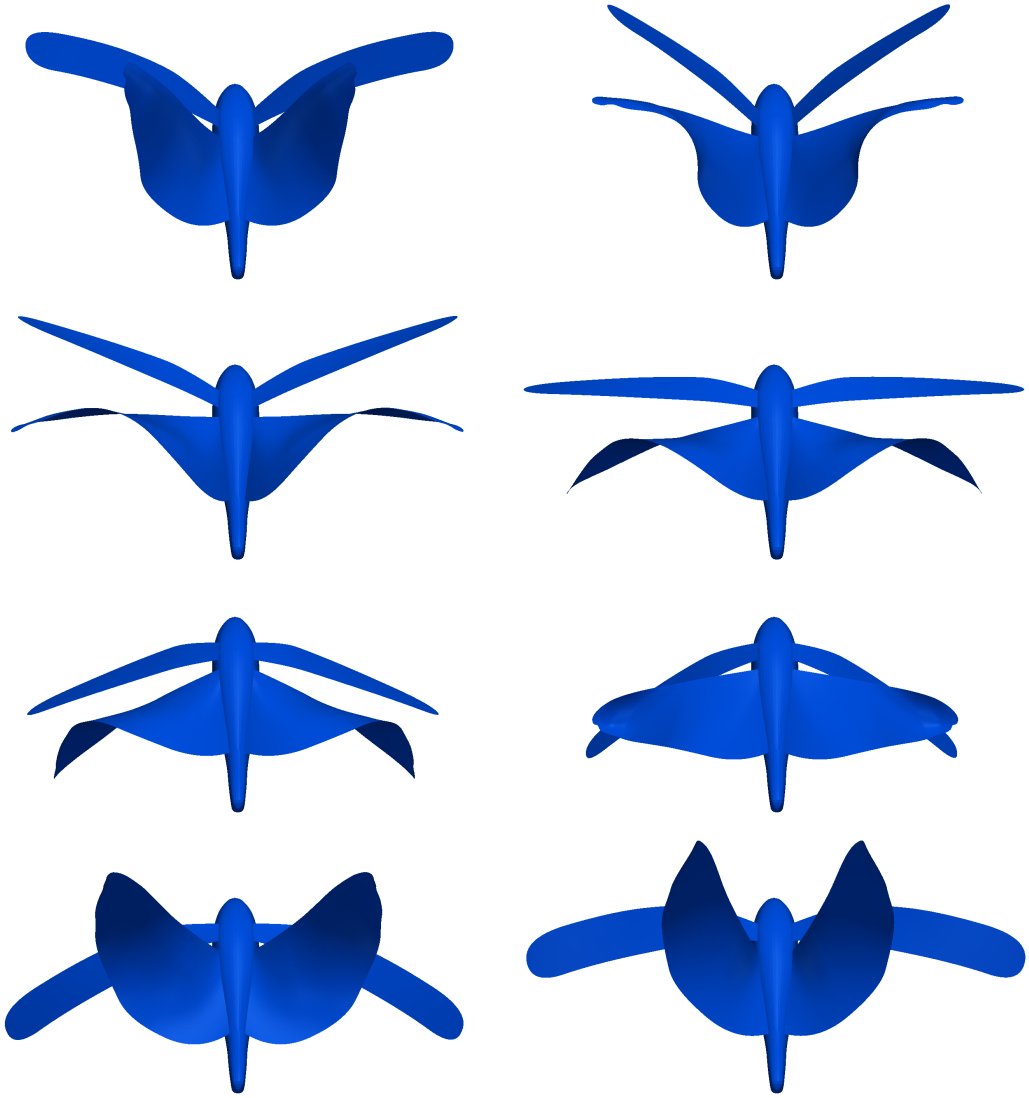


Figure 7.2: Top view of the MAV with the HW deformation extracted from the structural mechanics computation, seen for the membrane model at eight equally-spaced instants during the third flapping cycle.)

7.2 Results

Figures 7.3 and 7.4 show the total and right HW lift and thrust with the shell and membrane models. The average total lift with the membrane model is 1.70 mN

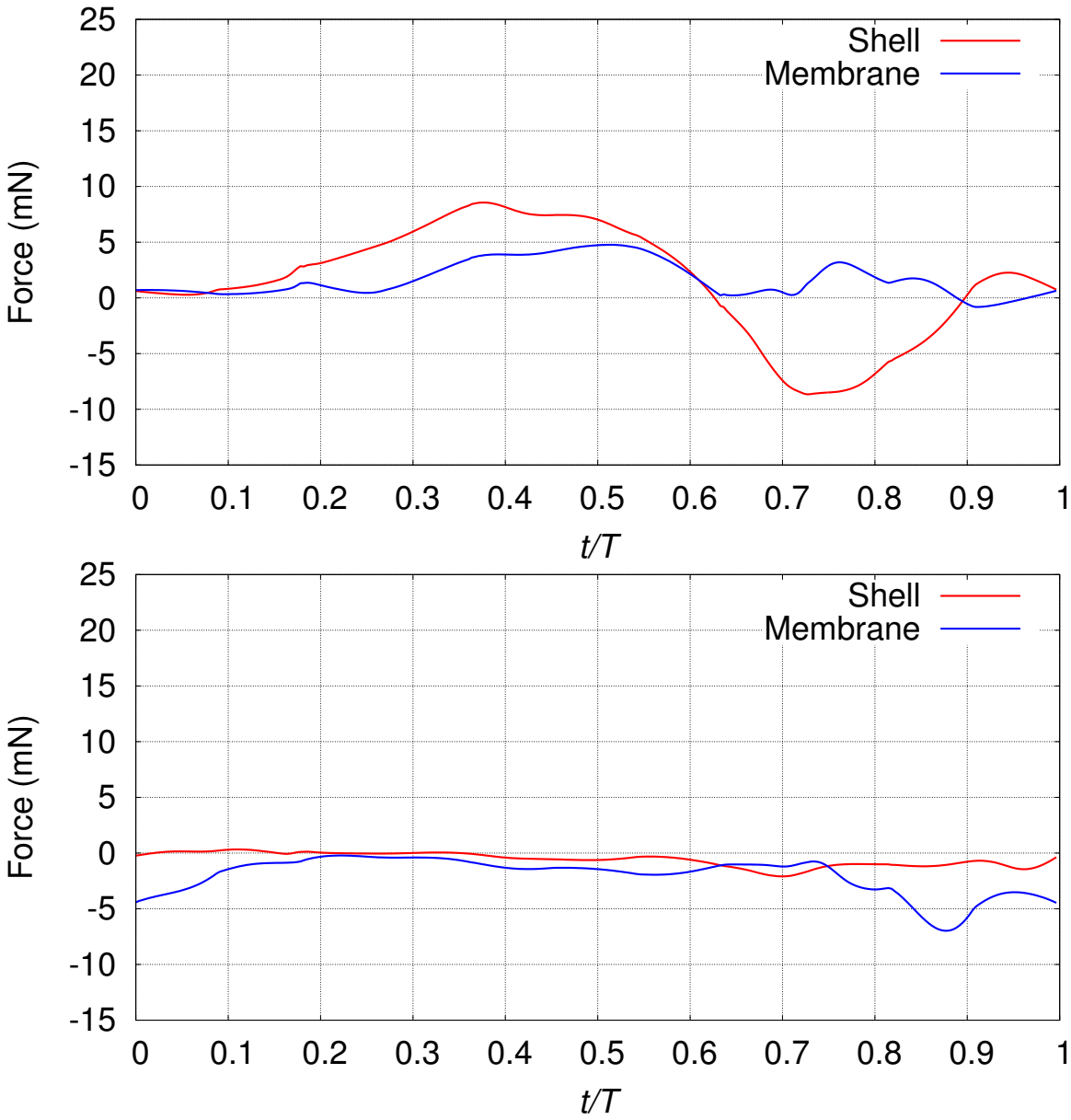


Figure 7.3: Total lift (top) and thrust (bottom) with the shell and membrane models.

compared to 1.43 mN with the shell model. The average thrust is −2.04 mN with the membrane model and −0.56 mN with the shell model. This shows a small increase in lift with a significant decrease in thrust. However, it is interesting to note that while

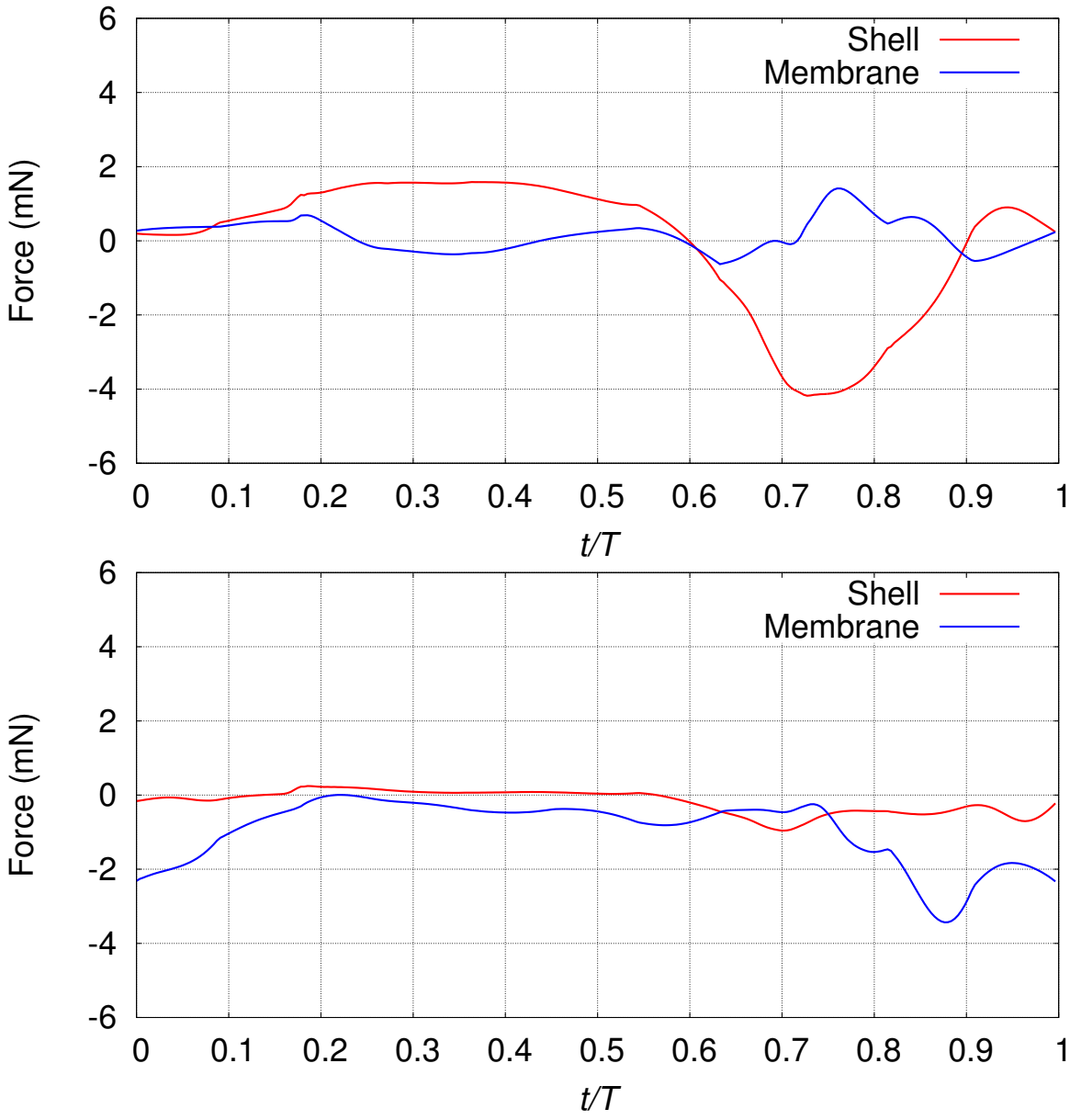


Figure 7.4: Right HW lift (top) and thrust (bottom) with the shell and membrane models.

the downstroke with shell model produces more lift, the upstroke with the membrane model produces quite a bit of positive lift instead of negative lift. This suggests that an optimal flapping-wing MAV design should perhaps have a shell-like behavior on the downstroke and membrane-like behavior on the upstroke. Such a wing would also more accurately mirror the actual locust wing deformation in flight. Adding structural members that provide stiffness to a membrane wing on the downstroke is a possible solution.

Chapter 8

Prescribed body motion

This chapter is from [2]. To fully simulate an MAV under various flying conditions, we want to be able to not only change the wing deformation patterns and statically change the angle of attack, but also have the ability to apply any prescribed body motion to the MAV. We can add heave, sway, surge, roll, pitch, and yaw, and investigate their effect on the forces generated by the MAV, assuming that the MAV somehow performs such maneuvers. Thanks to the refinement cylinder, we can add all of these without impacting the mesh quality around the MAV wings. All the mesh motion is taken up by the external region where the elements can undergo large deformations without experiencing issues with mesh quality. The computations presented so far had no body motion. We consider several test cases to observe the effect of prescribed body motion on the aerodynamic forces generated. First we study if the small amount of body motion seen during the locust flight has an effect on the overall lift and thrust production. In addition, we study the cases of larger and more general prescribed body motion. Specifically, we consider three cases: rolling, pitching, and combined rolling and pitching.

8.1 Locust body motion

We prescribe for the MAV the body motion that the locust had in the flight from which we extracted the wing motion and deformation data.

8.1.1 Setup and computations

In [3, 1, 82], the body motion of the locust was removed from the motion of the wing tracking points. Here we include that body motion separately, which consists of a translational-displacement vector and three rotations. The setup for the representation of the wing motion is identical to that in the base case. In the temporal representation of the body motion, we use cubic NURBS basis functions for the position vector and the three angles. With this, we have separate temporal representations for the body and wing motions. To make the entire motion periodic, the body position vector and three angles are adjusted just like how the wing motion was made periodic in [1]. The body position and orientation cycle is split into the same temporal patches that the flapping cycle was split.

The wing flapping is handled, as before, with the mesh motion inside the cylindrical refinement region. The body motion is handled by moving the cylindrical refinement region, which in turn drives the motion of the mesh between the cylindrical region and the external boundaries. For the base case, there is no mesh motion outside the cylindrical refinement region, so that part of the mesh does not change. For the cases with body motion, because the locust body motion is fairly small, that mesh changes (i.e. moves), but its connectivity does not. As before, we subdivide the prescribed motion into smaller steps and use the same method to deform the outer region as we did for the mesh motion around the wings. We perform the fluid mechanics and mesh moving computations with the same settings as the base case. Figures 8.1 and 8.2 show, in a comparative setting, the flapping and body motions for the base

and locust-body cases. Figures 8.3 and 8.4 show the vertical and longitudinal wing tip positions and velocities for the base and locust-body cases.

8.1.2 Results

Figure 8.5 shows the total lift and thrust for the base and locust-body cases. On the downstroke, the lift for the locust-body case is somewhat smaller than that of the base case. Also, the magnitude of the thrust is smaller. This is most likely due to the fact that on the downstroke, the body motion causes the body, and also the wings, to move up and back thus decreasing the actual wind velocity around the wings. This can also be seen during the upstroke, with the magnitude of the lift and thrust being smaller than those in the base case. Any body motion opposite the motion of the wings leads to a decreased relative velocity and thus decreased force magnitude. The body motion for this case is, for the most part, opposite to that of the wing motion, and, therefore, leads to magnitude loss for both lift and thrust. The average lift and thrust for the locust-body case is 6.04 mN and -1.27 mN, respectfully, compared to 6.50 mN and -1.31 mN for the base case.

8.2 Prescribed roll

8.2.1 Setup and computations

We model a dynamic roll where the MAV starts at 18.5°, rotates to 45°, then down to 0°, and back to 18.5°. Such a large range of motion would have required far more frequent remeshing if the body motion drove the mesh motion directly. Using the external mesh to take up the motion makes the mesh moving process more efficient. The large range of rotation requires that we use a new external mesh for each patch. However, this does not increase the remeshing cost that much (the external mesh has roughly 30% of the nodes). The HW deformation is computed with the

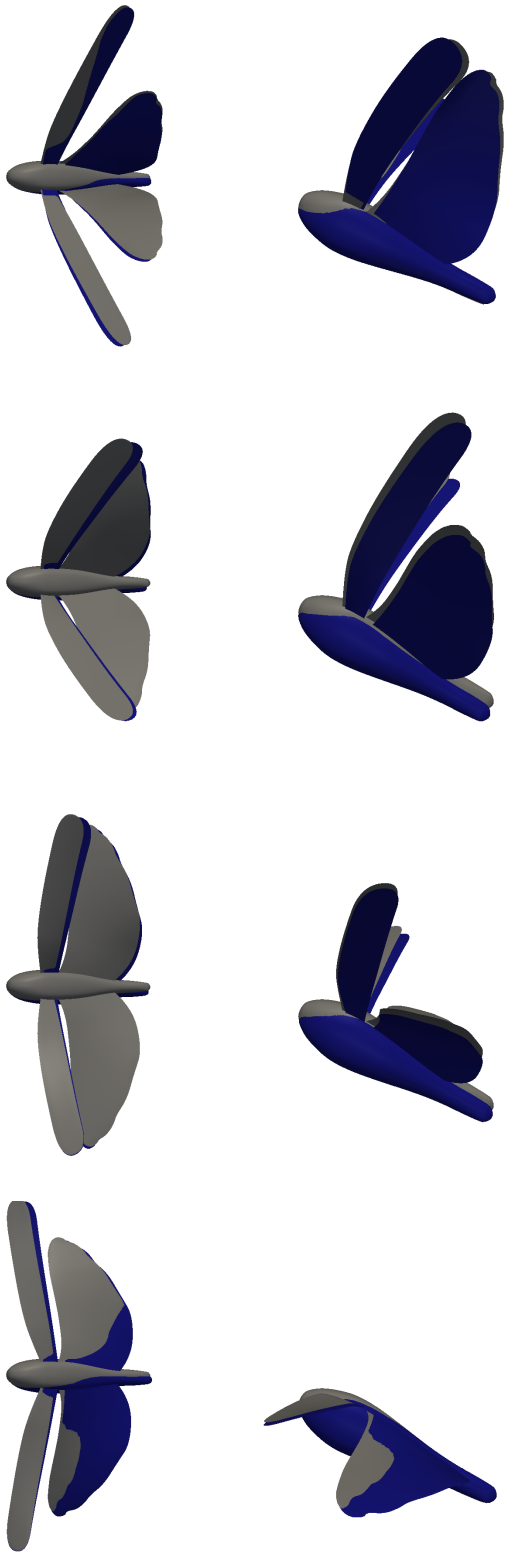


Figure 8.1: Flapping and body motions for the base (grey) and locust-body (blue) cases at the first four of eight equally-spaced instants during the flapping cycle. Top (left) and side (right) views.

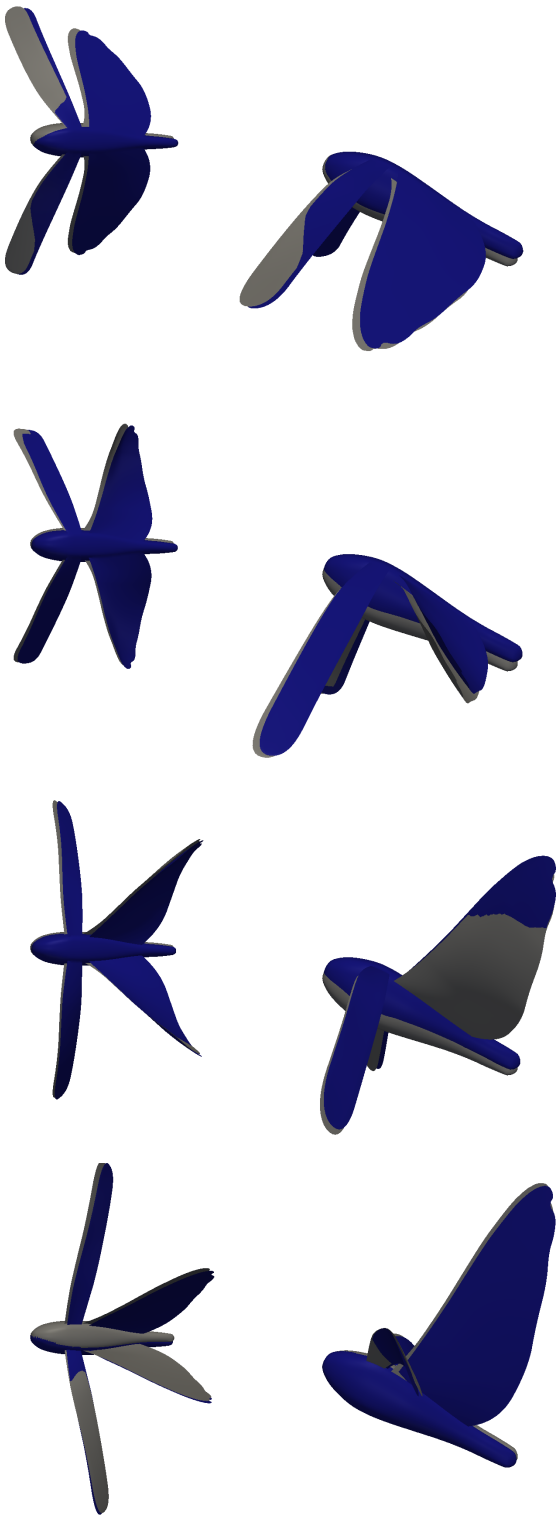


Figure 8.2: Flapping and body motions for the base (grey) and locust-body (blue) cases at the last four of eight equally-spaced instants during the flapping cycle. Top (left) and side (right) views.

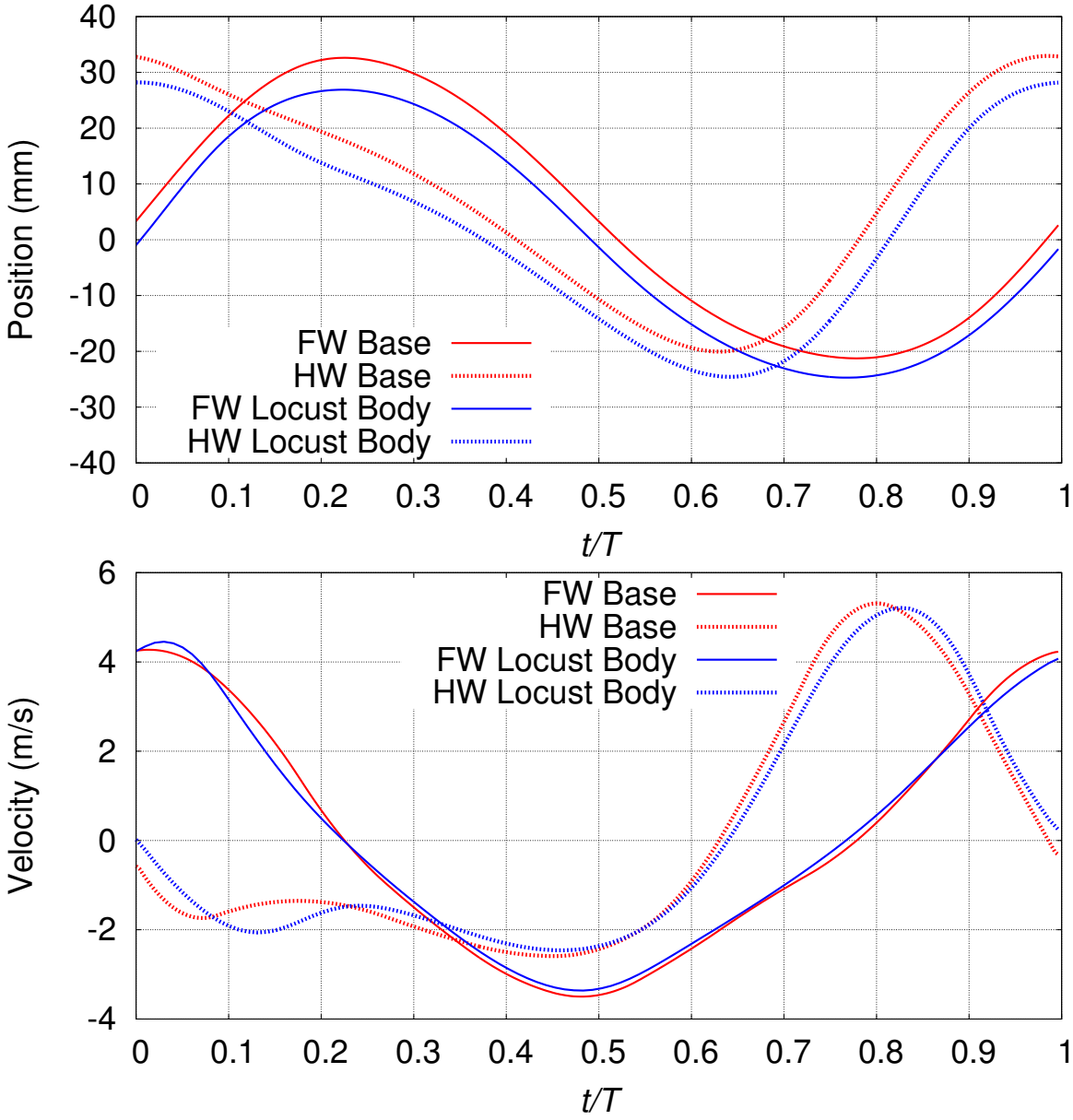


Figure 8.3: Vertical tip position and velocity for the right wings for the base and locust-body cases.

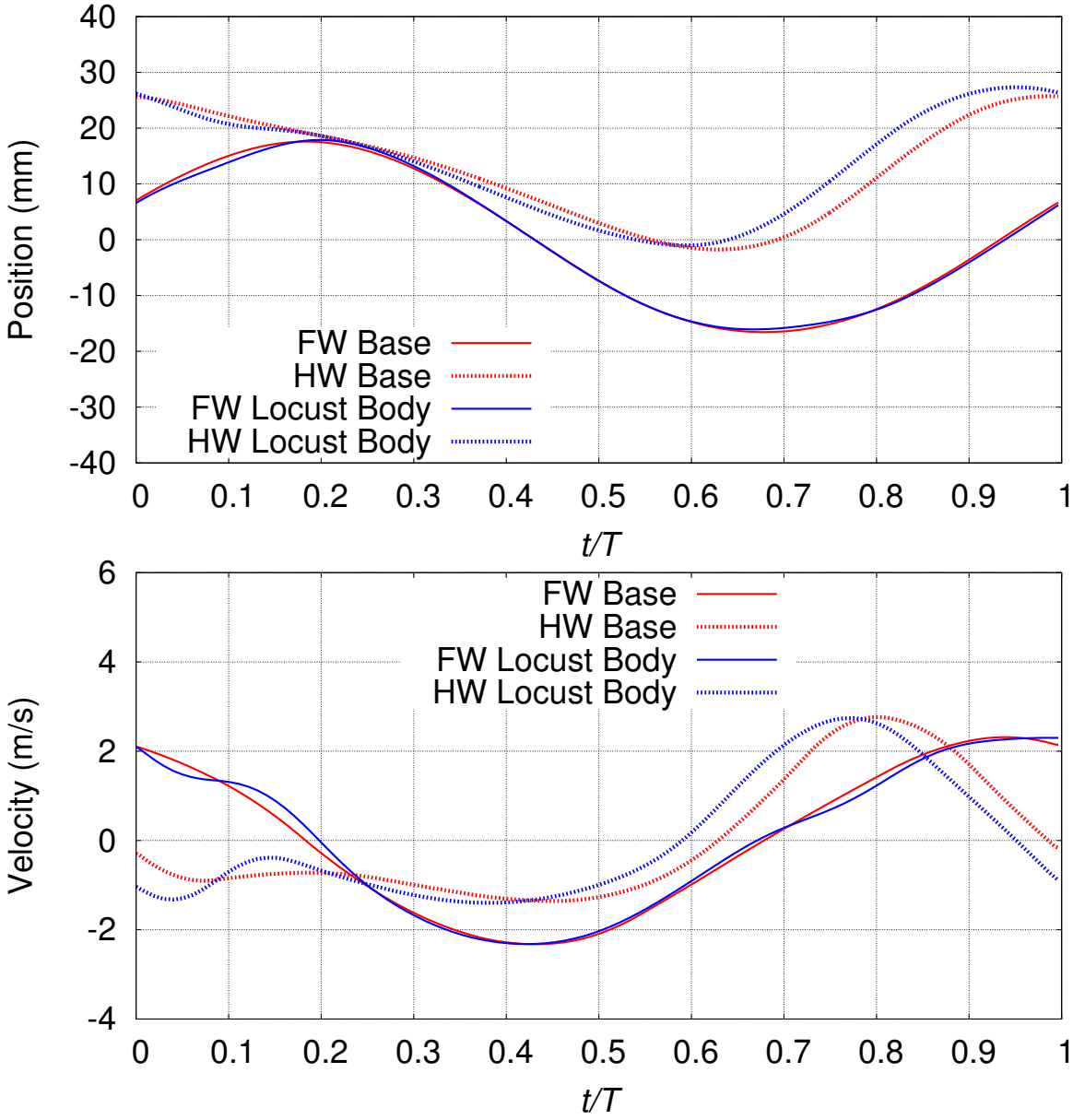


Figure 8.4: Longitudinal tip position and velocity for the right wings for the base and locust-body cases.

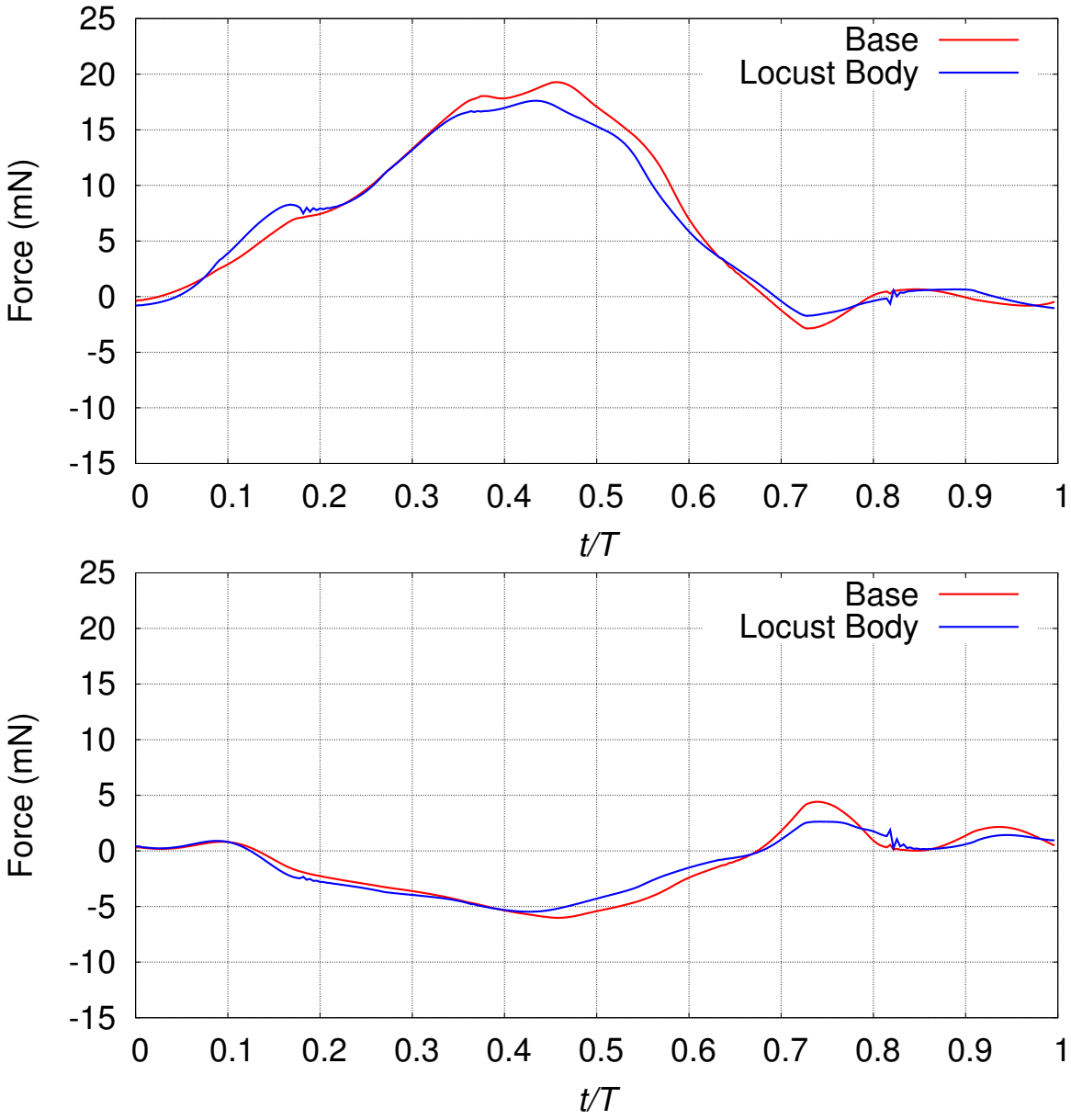


Figure 8.5: Total lift (top) and thrust (bottom) for the base and locust-body cases.

Kirchhoff–Love shell model. The computational parameters for the fluid and structural mechanics and mesh motion are the same as in Chapter 6. Figure 8.6 shows the body and wing motions used in the fluid mechanics computation of the first SCFSI iteration. We perform three SCFSI iterations and present the results from that.

8.2.2 Results

Figure 8.7 shows the total lift and thrust for the fixed-body and prescribed-roll cases. The rolling motion changes the aerodynamic forces generated significantly. The prescribed roll results in a noticeable decrease in the lift, but at the same time the thrust production is improved during the first part of the flapping cycle (HW downstroke). This suggests that rolling maneuvers can be useful in some cases when increased thrust is needed and lift is not as important.

8.2.3 Prescribed pitch

Setup and computations

The setup for the prescribed-pitch case is very similar to the setup for the prescribed-roll case. Instead of varying the roll angle, we are changing the body angle, which dynamically alters the angle of attack during the flight. We start at a body angle of 11.7°, pitch the MAV up to 27.3°, down to 0°, and back to 11.7°. Again, because the refinement cylinder has a large motion, a new external mesh is used for each patch. The HW deformation is computed with the Kirchhoff–Love shell model. The computational parameters for the fluid and structural mechanics and mesh motion are the same as in Chapter 6. Figure 8.8 shows the body and wing motions used in the fluid mechanics computation of the first SCFSI iteration. We perform three SCFSI iterations and present the results from that.

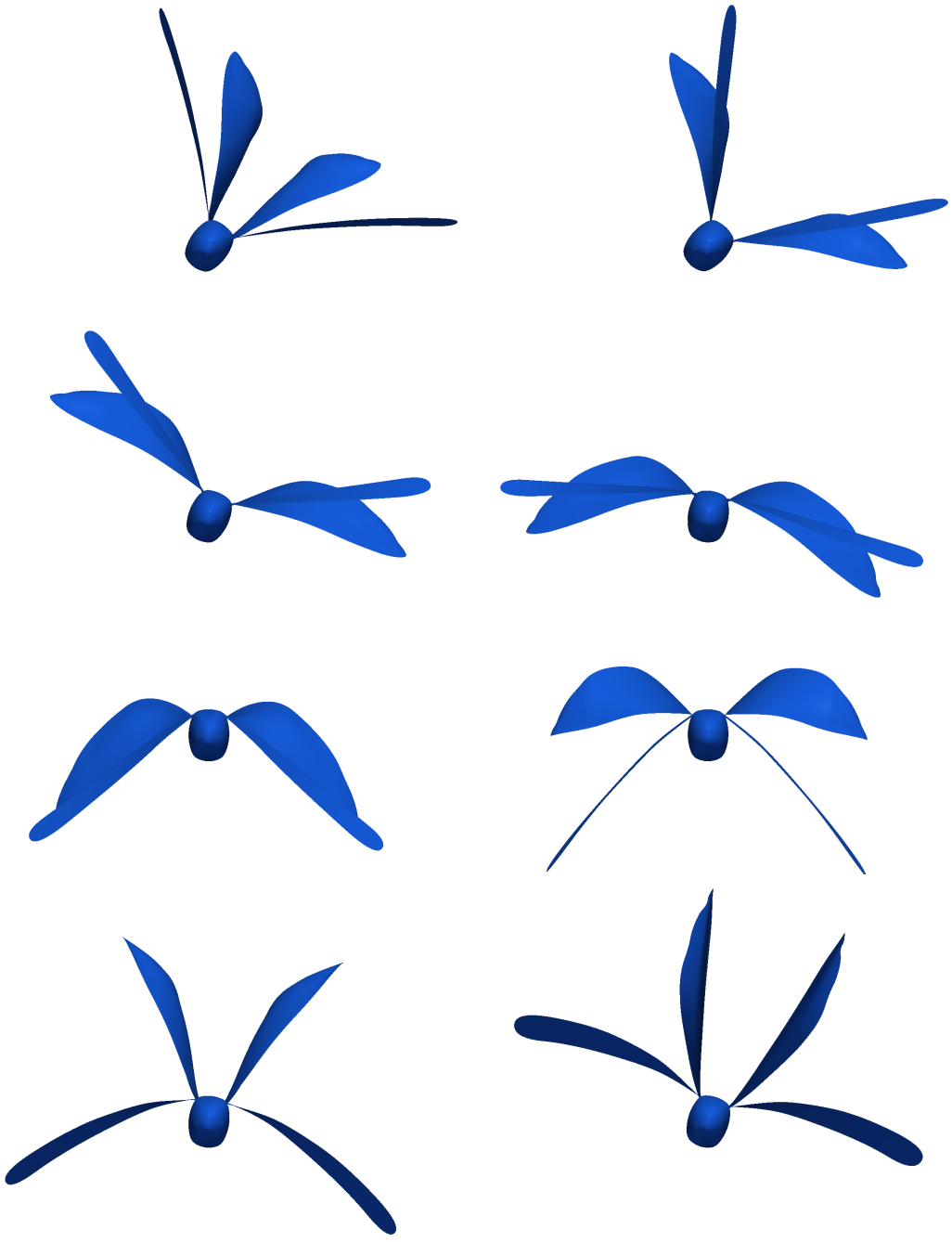


Figure 8.6: Prescribed-roll case with the HW deformation extracted from the structural mechanics computation, seen at eight equally-spaced instants during the third flapping cycle.

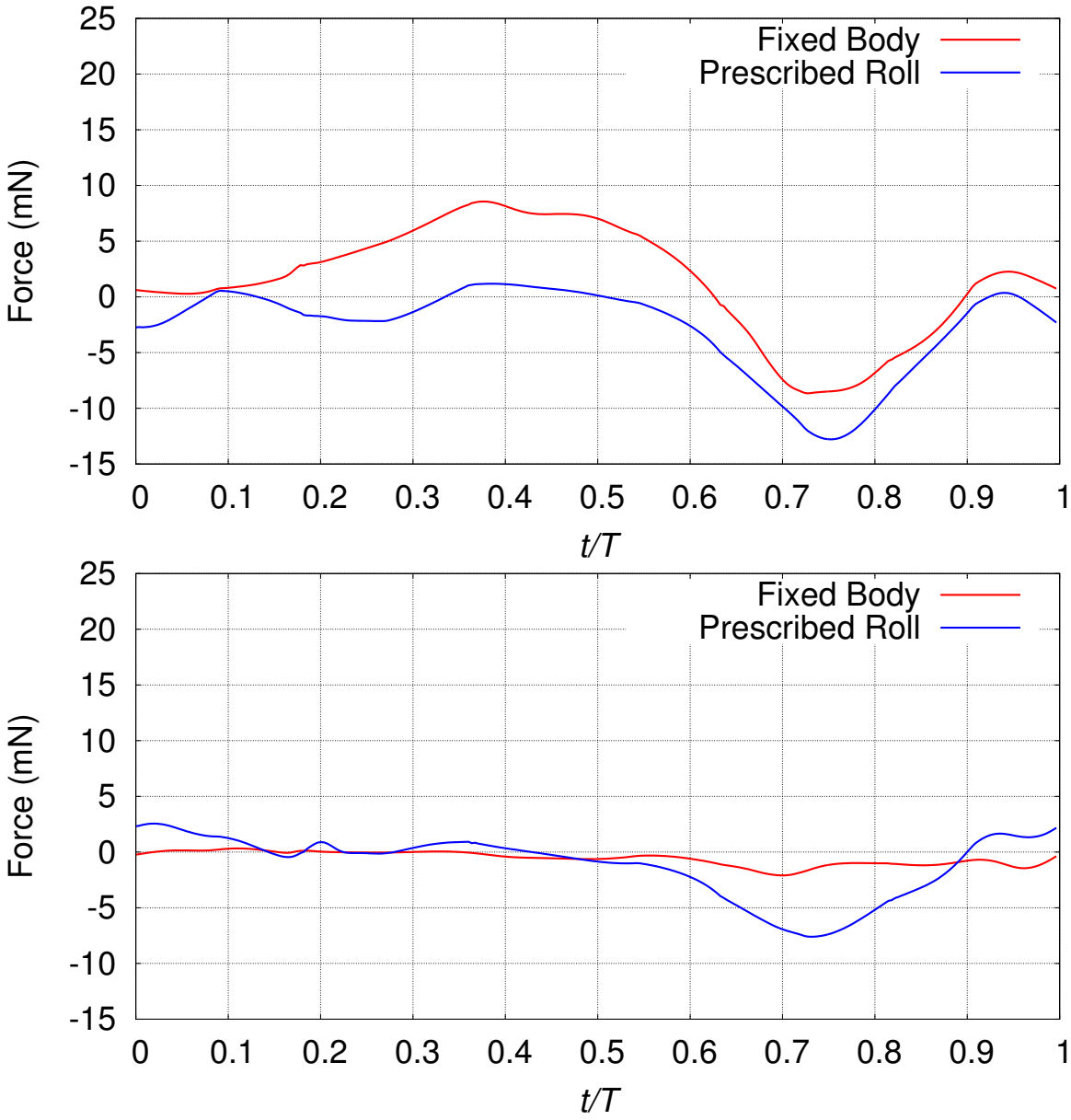


Figure 8.7: Total lift (top) and thrust (bottom) for the fixed-body and prescribed-roll cases.

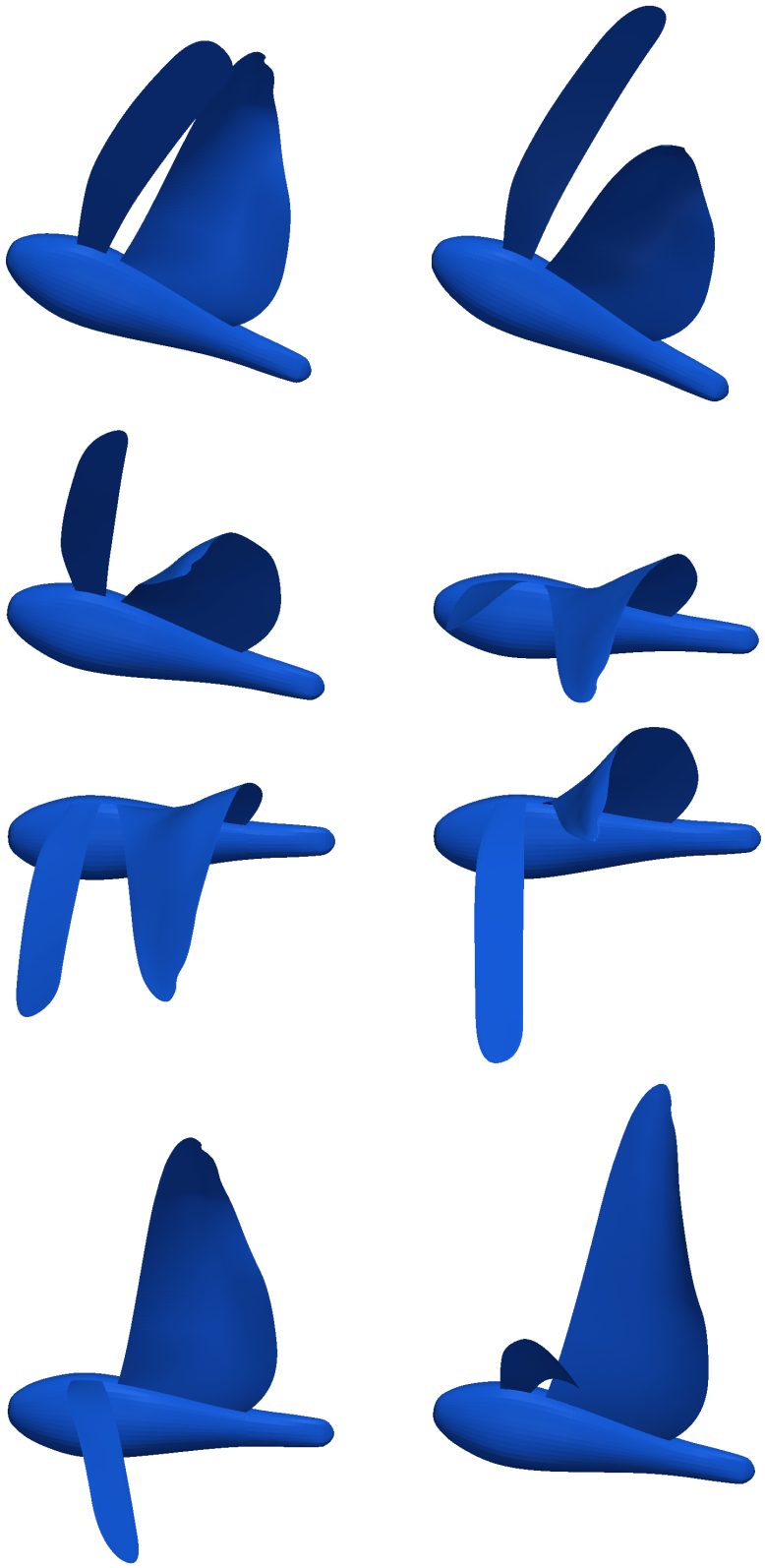


Figure 8.8: Prescribed-pitch case with the HW deformation extracted from the structural mechanics computation, seen at eight equally-spaced instants during the third flapping cycle.)

Results

Figure 8.9 shows the total lift and thrust for the fixed-body and prescribed-pitch cases. We see an increase in both the lift and thrust in the initial part of the HW

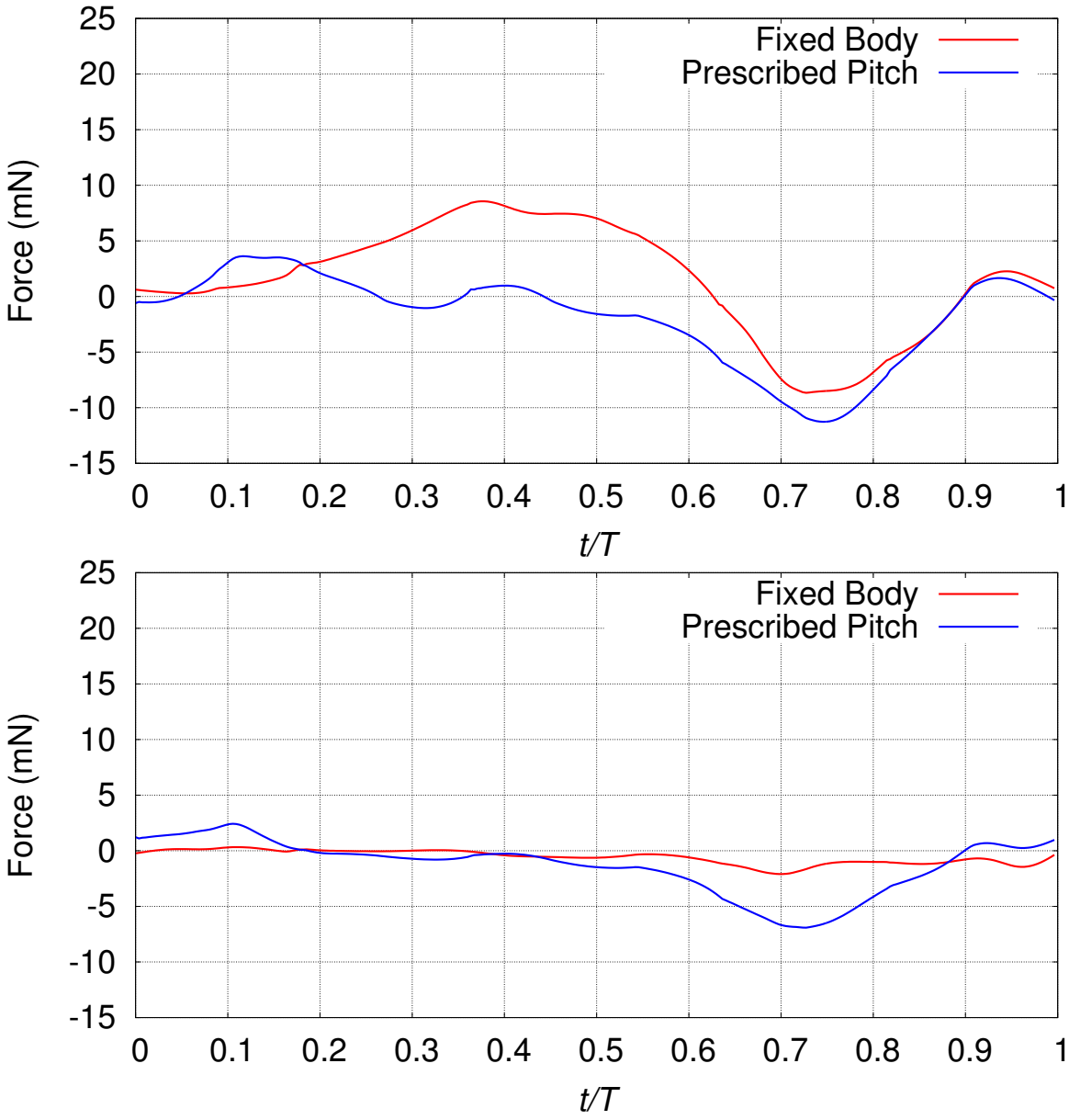


Figure 8.9: Total lift (top) and thrust (bottom) for the fixed-body and prescribed-pitch cases.

downstroke, but the overall lift and thrust both decrease. During that first part of the cycle the pitching motion is upwards (increases the angle of attack). By changing

the direction of the pitching motion with respect to the HW motion, we can influence the total lift and thrust production of the MAV.

8.2.4 Prescribed roll and pitch

Setup and computations

The prescribed body motion in this case is a synchronized combination of the prescribed roll and pitch in the two earlier cases. Again, a new external mesh is used for each patch. The HW deformation is computed with the Kirchhoff–Love shell model. The computational parameters for the fluid and structural mechanics and mesh motion are the same as in Chapter 6. We perform three SCFSI iterations and present the results from that.

Results

Figure 8.10 shows the total lift and thrust for the fixed-body and prescribed roll and pitch cases. At the beginning of the flapping cycle there is an increase in the lift, but the overall lift production is decreased. As before, the average lift and thrust both decrease. Figures 8.11 and 8.12 show the pressure on the wing surfaces (relative to the free-stream pressure) at eight equally-spaced instants during the third flapping cycle. As can be seen, the pressure distribution is quite asymmetrical compared to the base case.

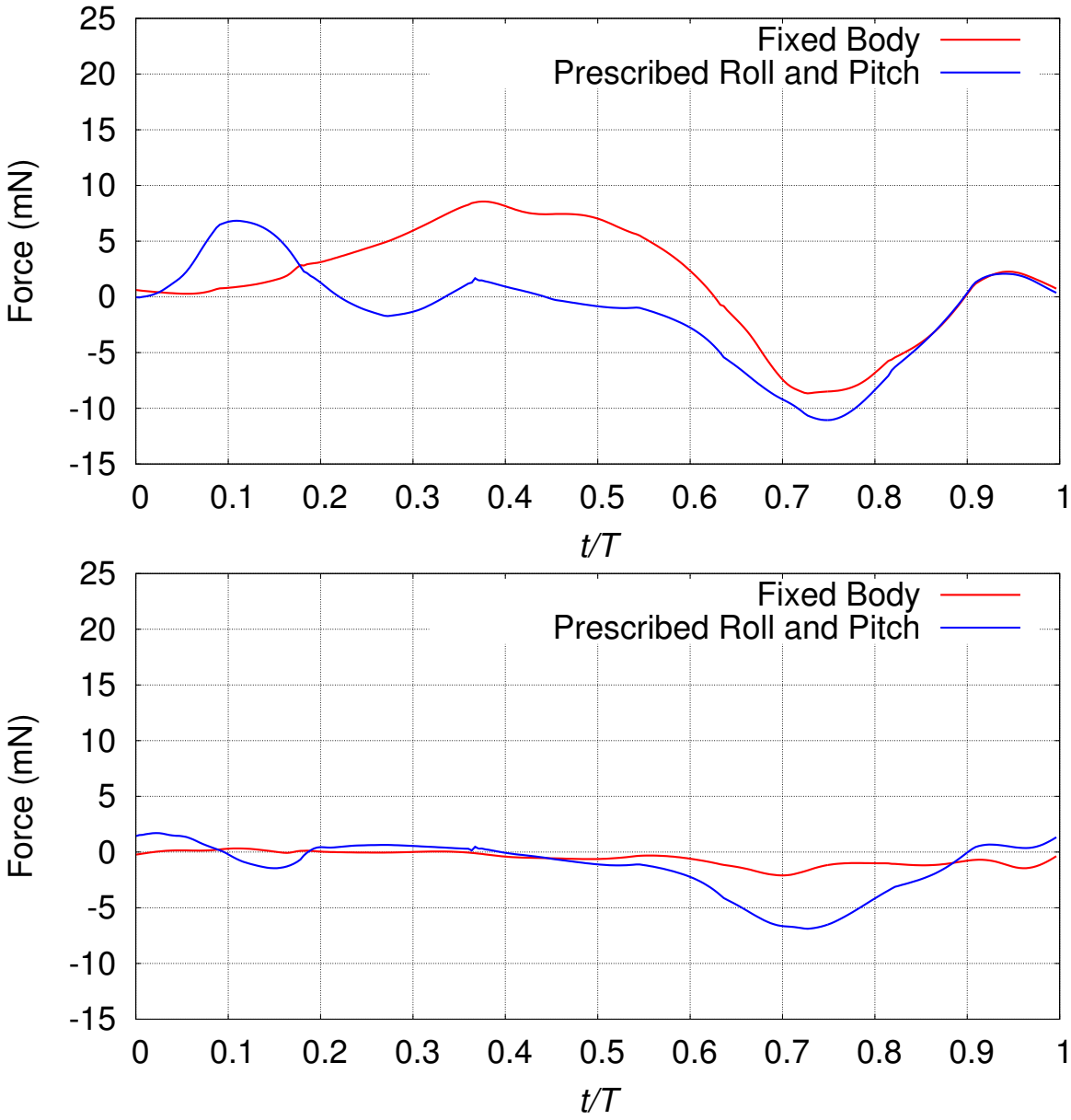


Figure 8.10: Total lift (top) and thrust (bottom) for the fixed-body and prescribed roll and pitch cases.

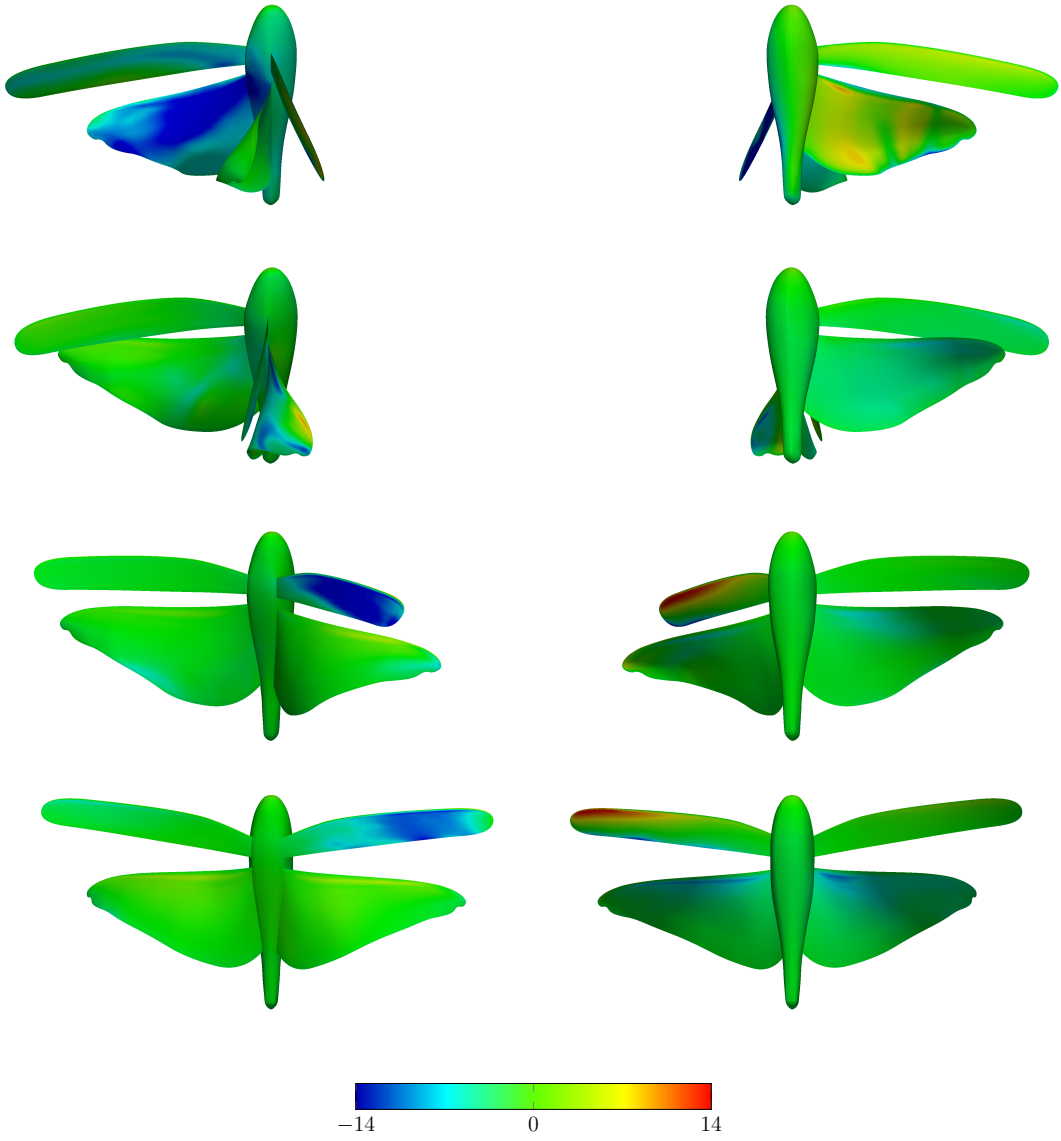


Figure 8.11: Surface pressure in Pa (relative to the free-stream pressure) for the prescribed roll and pitch case, seen at the first four of eight equally-spaced instants during the third flapping cycle (top view on left, bottom view on right).

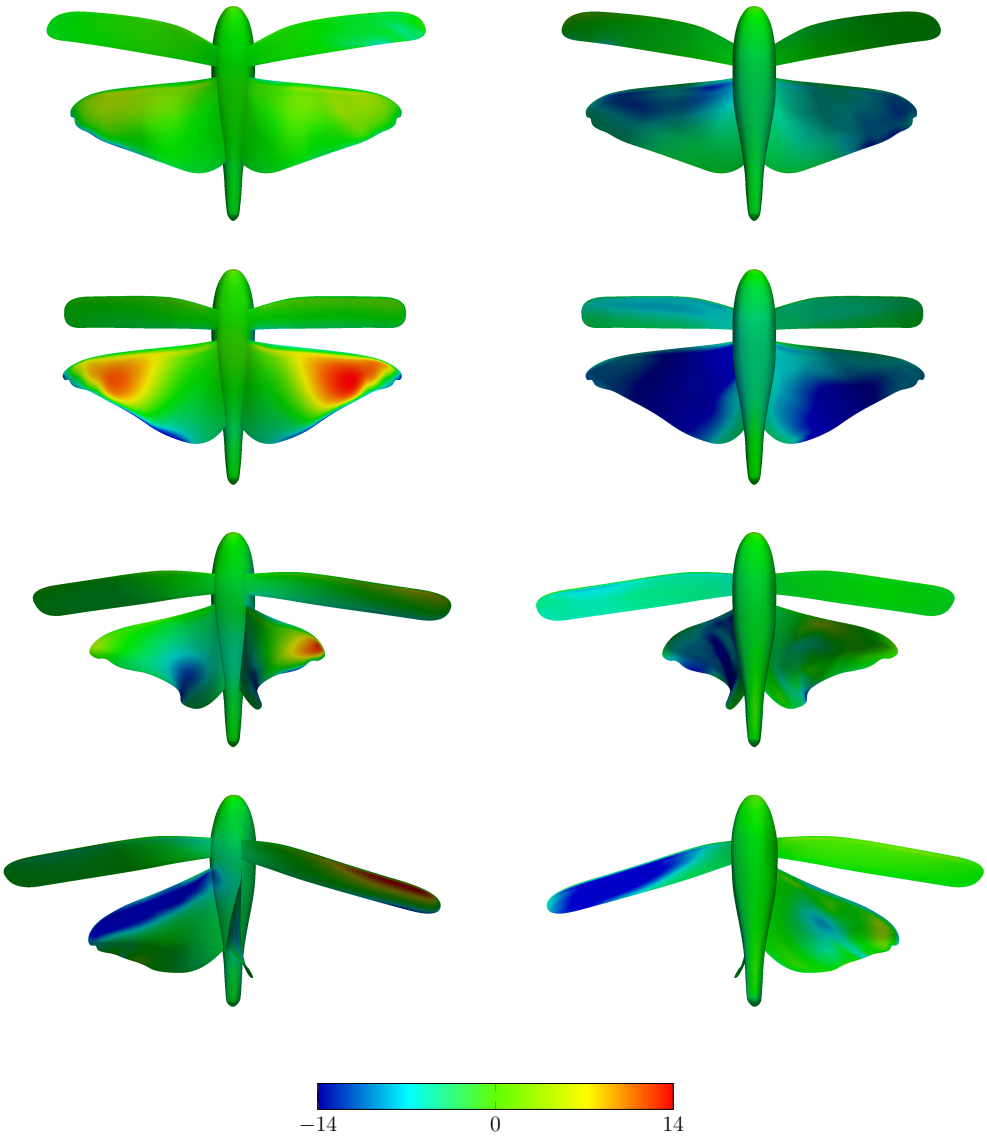


Figure 8.12: Surface pressure in Pa (relative to the free-stream pressure) for the prescribed roll and pitch case, seen at the last four of eight equally-spaced instants during the third flapping cycle (top view on left, bottom view on right).

Chapter 9

Flapping-pattern studies

It is interesting to study the effect of changing the synchronization between the FW and HW and introduce asymmetry between the motions of the left and right wings. In this chapter we present such studies from [2]. In the first study, we alter the synchronization between the FW and HW by advancing or delaying the FW motion relative to what it is in the original locust data. This allows us to seek an optimum point for the lift and thrust production and evaluate the possibility of using the variation in the synchronization to alter the flight performance. In the second study, we introduce asymmetry (across the sagittal plane) to the flapping. From a practical standpoint, it is unlikely to have asymmetric flapping in the MAV design, at least in the current ones. Still, it would be valuable to have the ability to study the effect of such asymmetry for the purpose of evaluating more complex MAV designs. We created a test case where the left and right wings are flapping still with the same frequency, but one side is delayed with respect to the other.

9.1 Setup and computations

9.1.1 FW advanced or delayed

For the case where the FW is advanced ("FWA") or delayed ("FWD"), the setup is very similar to the case with the original locust data ("FWO"), other than just advancing or delaying the FW by 9.1% of the flapping period. The FW and HW come very close to each other during flapping, so it is not possible to change the synchronization that much because the wings would collide unless the distance between the wings along the body is increased. As this would cause a variety of other changes and make the comparisons less consistent, we use a value close to the maximum possible advancing and delaying that does not require increasing the distance between the wings. Figure 9.1 shows the vertical tip position of the FW for the FWA and FWD cases compared to the FWO case. The HW deformation is computed with the

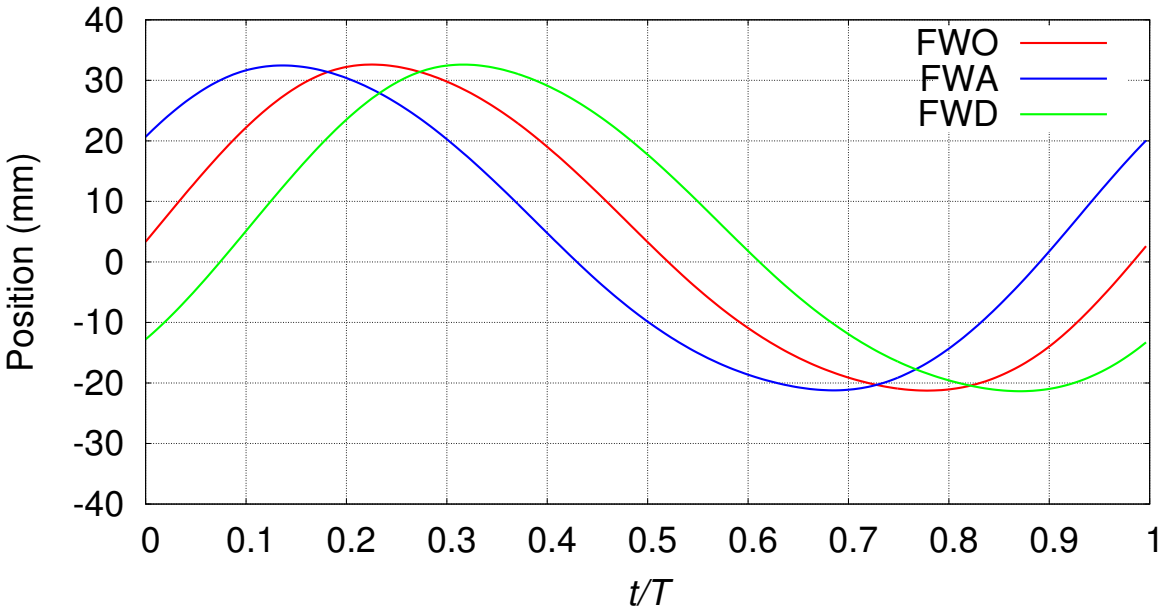


Figure 9.1: Vertical tip position of the FW for the FWA and FWD cases compared to the FWO case.

Kirchhoff–Love shell model. The computational parameters for the fluid and structural mechanics and mesh motion are the same as in Chapter 6. The mesh motion

strategy in the refinement cylinder is the same as it was in the FWO case. We perform three SCFSI iterations and present the results from that.

9.1.2 Asymmetric flapping

In this case we are delaying the left wings by 54% of the flapping period. Figure 9.2 shows the vertical tip positions of the FW and HW for both left and right sides. The HW deformation is computed with the Kirchhoff–Love shell model. The computational parameters for the fluid and structural mechanics and mesh motion are the same as in Chapter 6. The mesh motion strategy in the refinement cylinder is the same as it was in the symmetric-flapping case, except for the mesh asymmetry between the two sides across the sagittal plane. Figure 9.3 shows the body and wing motions used in the fluid mechanics computation of the first SCFSI iteration. We perform three SCFSI iterations and present the results from that.

9.2 Results

9.2.1 FW advanced or delayed

Figures 9.4–9.9 show the lift and thrust for the FWA and FWD cases compared to the FWO case. The average lift is 1.43 mN, 1.40 mN and 1.39 mN for the FWO, FWA and FWD cases. The peak lift occurs at different times. The average thrust is -0.56 mN, -0.50 mN and -0.52 mN for the FWO, FWA and FWD cases. The original synchronization has the most lift, while both the FWA and FWD cases show an increase in thrust.

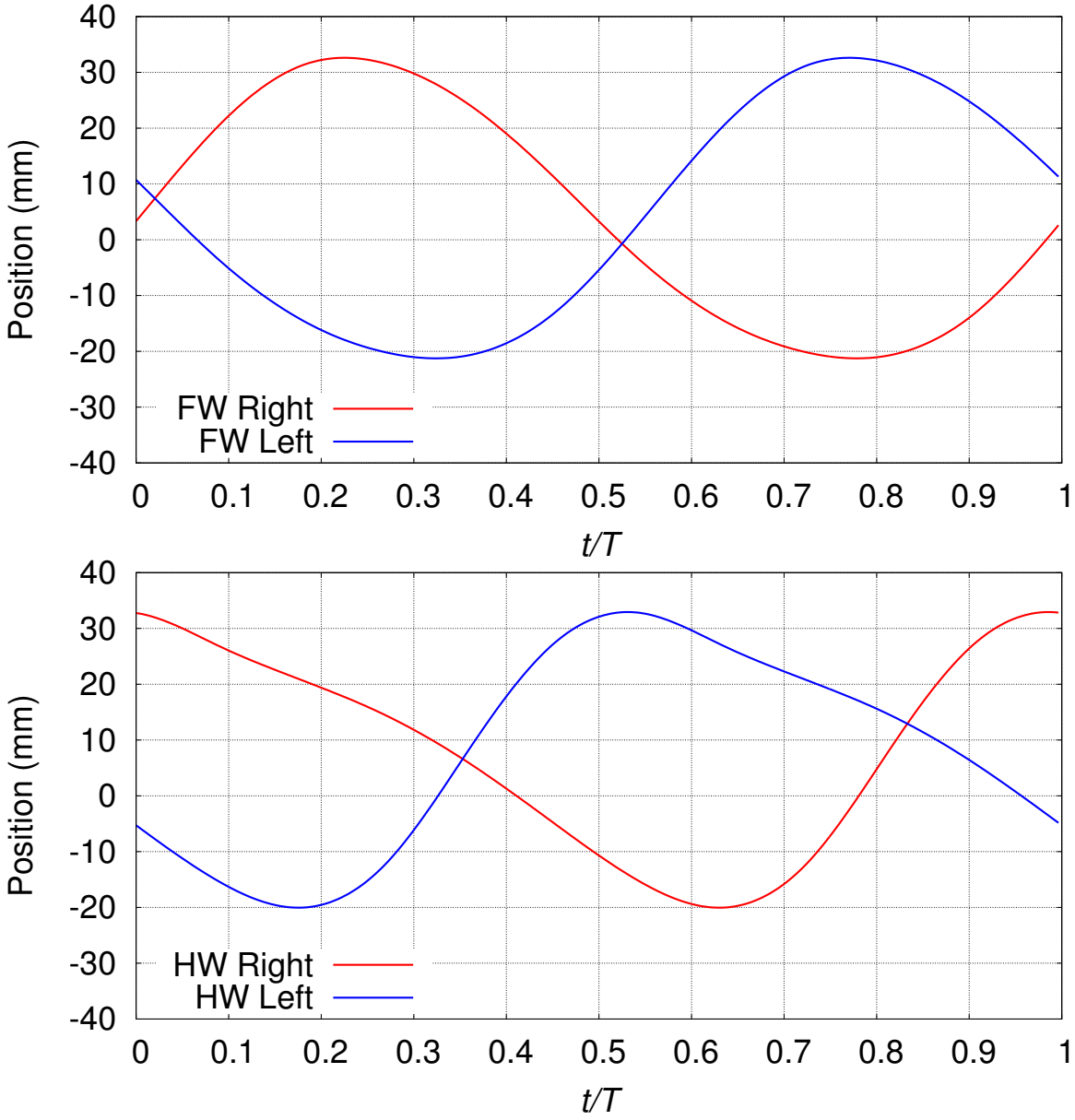


Figure 9.2: Vertical tip positions of the FW and HW for the left and right sides in the asymmetric-flapping case.

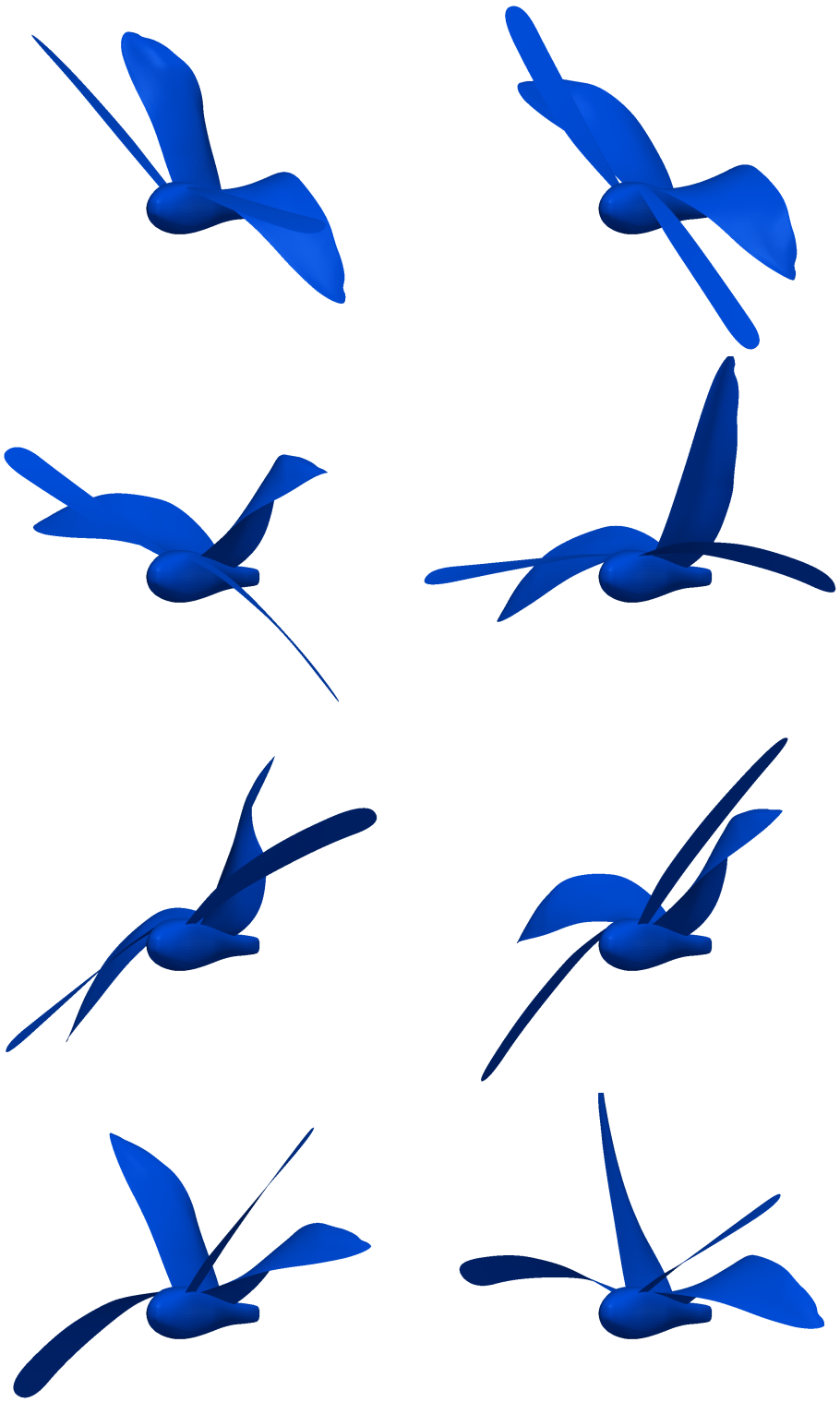


Figure 9.3: Asymmetric-flapping case with the HW deformation extracted from the structural mechanics computation, seen at eight equally-spaced instants during the third flapping cycle.

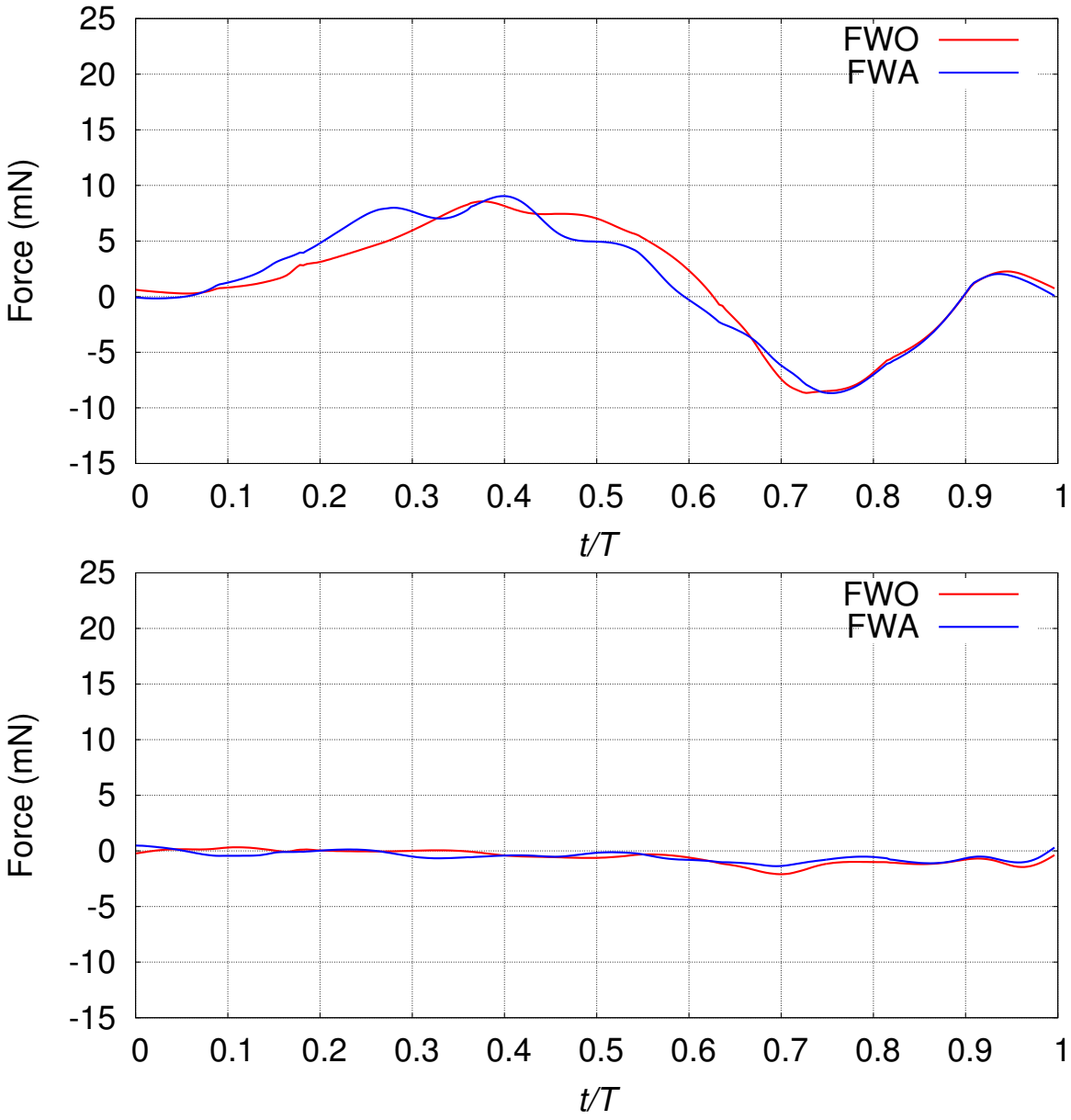


Figure 9.4: Total lift (top) and thrust (bottom) for the FWA and FWO cases.

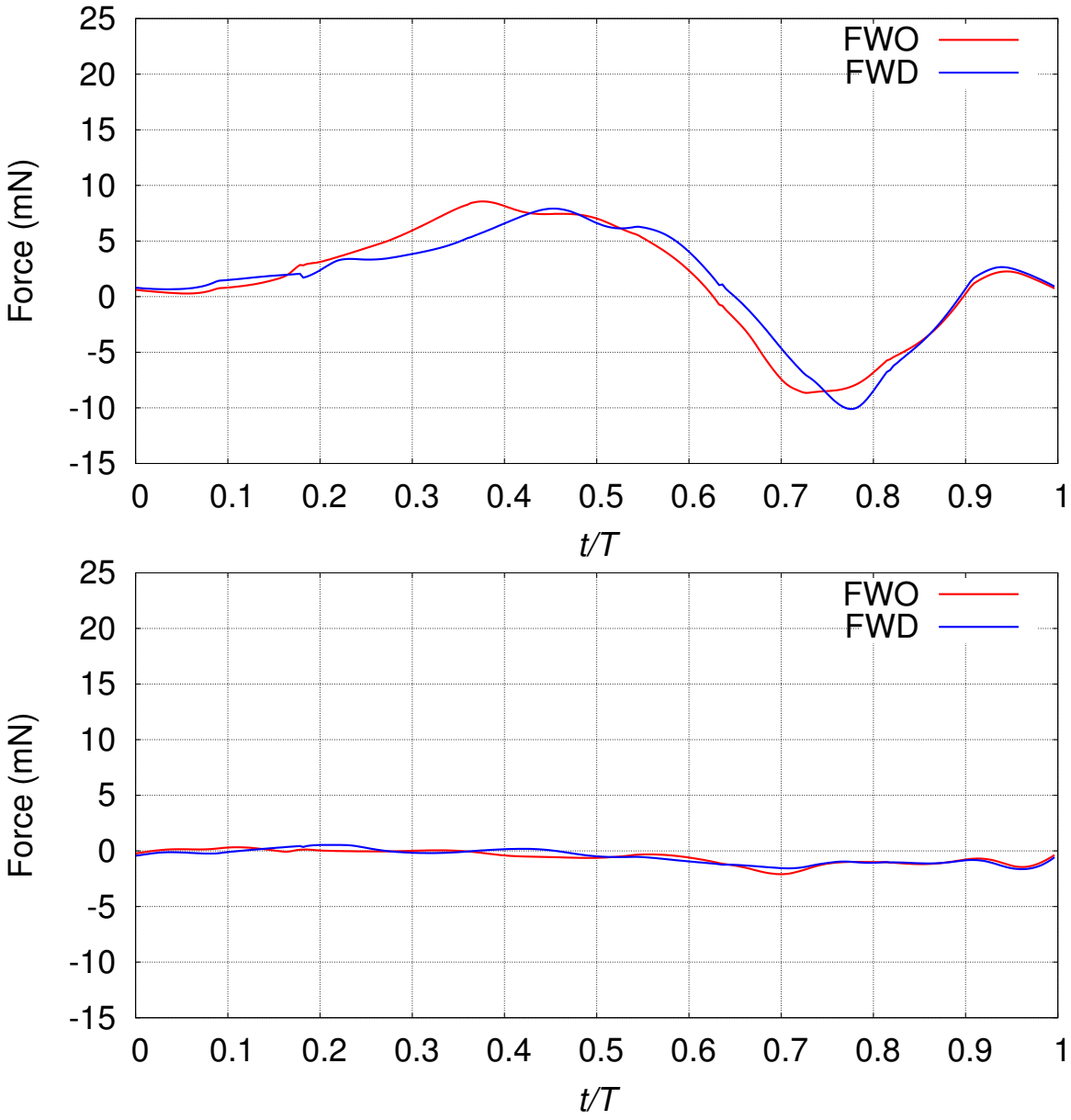


Figure 9.5: Total lift (top) and thrust (bottom) for the FWD and FWO cases.

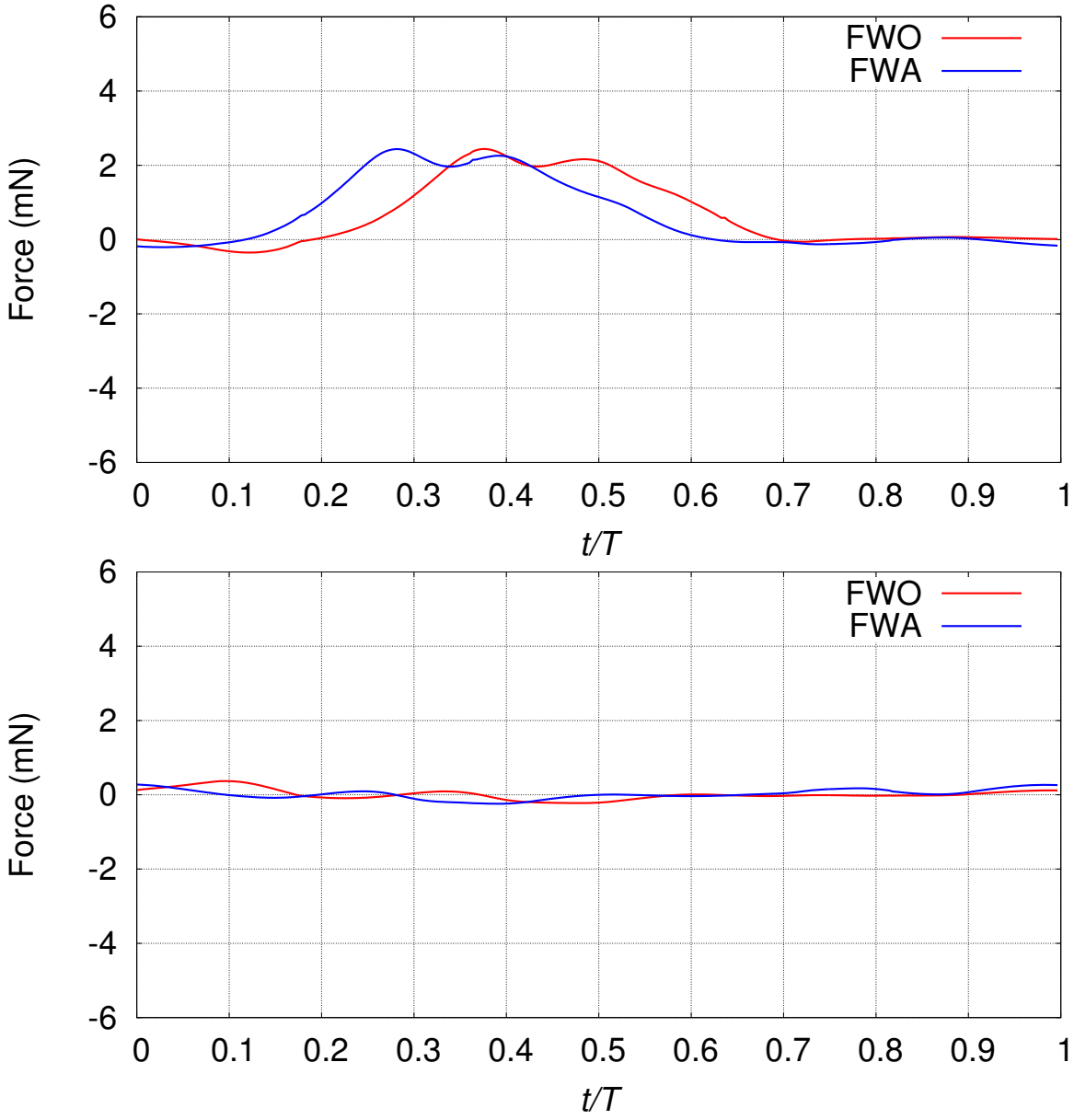


Figure 9.6: Right FW lift (top) and thrust (bottom) for the FWA and FWO cases.

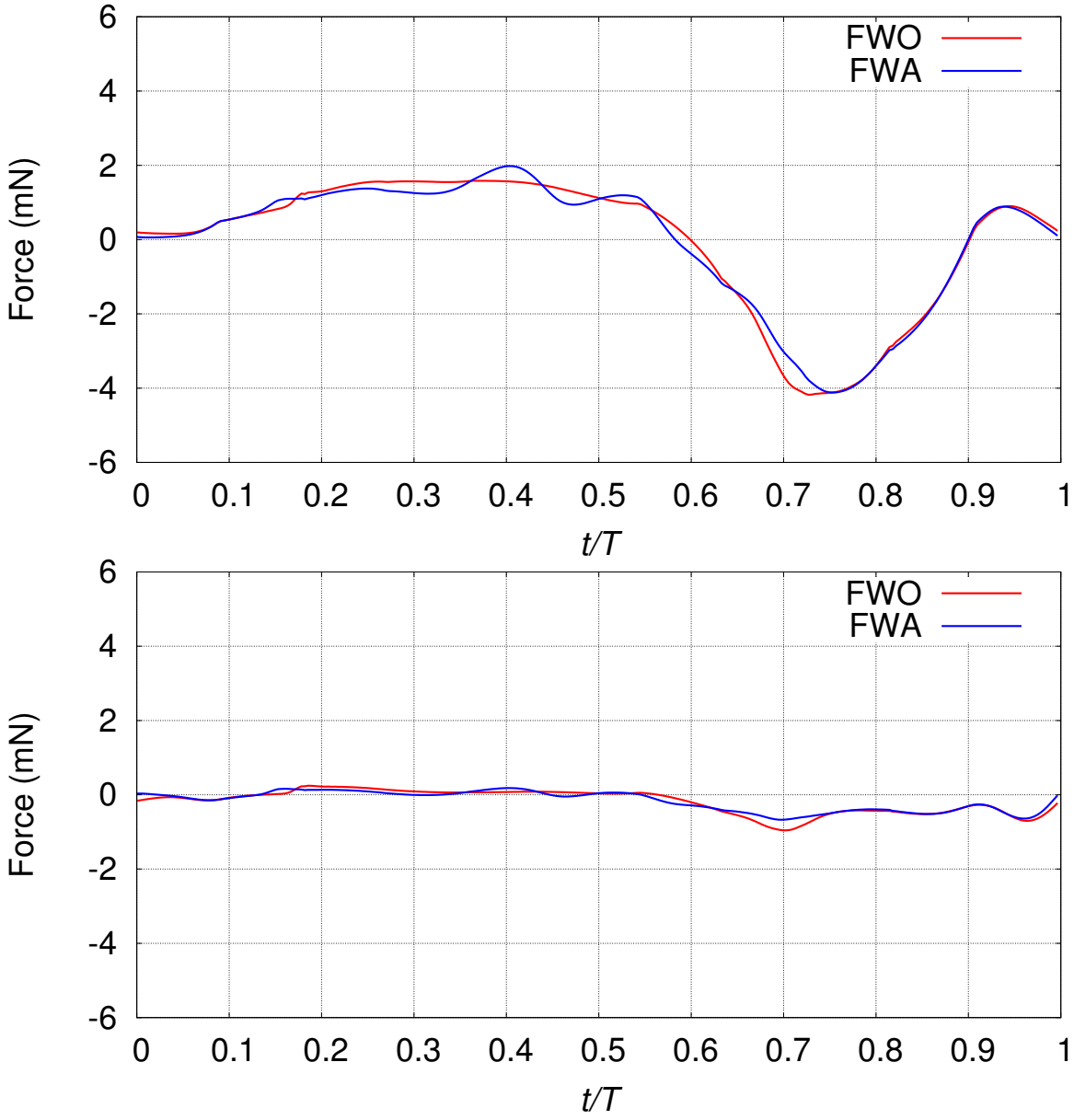


Figure 9.7: Right HW lift (top) and thrust (bottom) for the FWA and FWO cases.

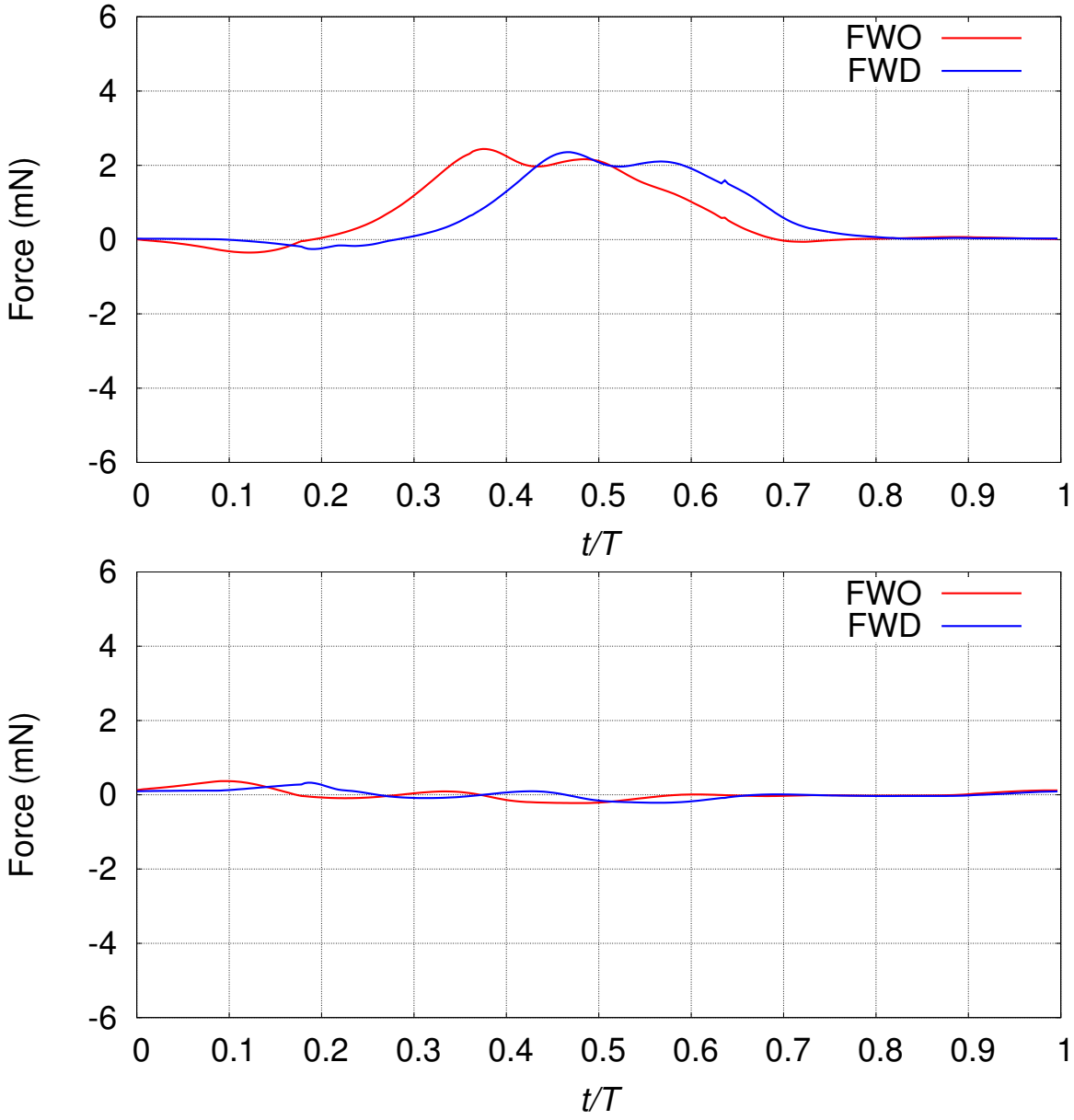


Figure 9.8: Right FW lift (top) and thrust (bottom) for the FWD and FWO cases.

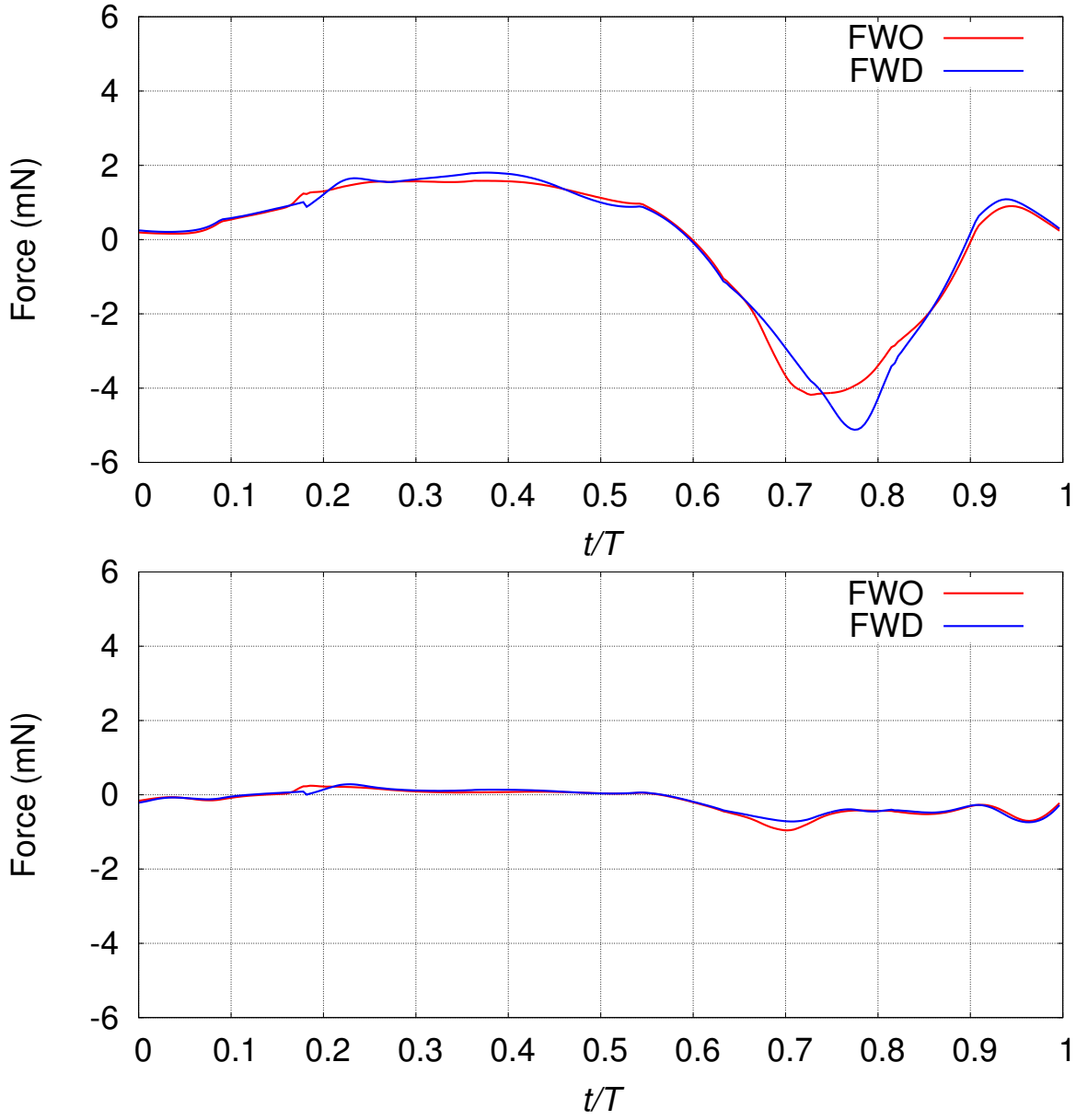


Figure 9.9: Right HW lift (top) and thrust (bottom) for the FWD and FWO cases.

9.2.2 Asymmetric flapping

Figure 9.10 shows the total lift and thrust for the symmetric and asymmetric flapping. Figures 9.11 and 9.12 show, for the right FW and HW, the lift and thrust for the symmetric and asymmetric flapping. Figures 9.13 and 9.14 do that for the left FW and HW.

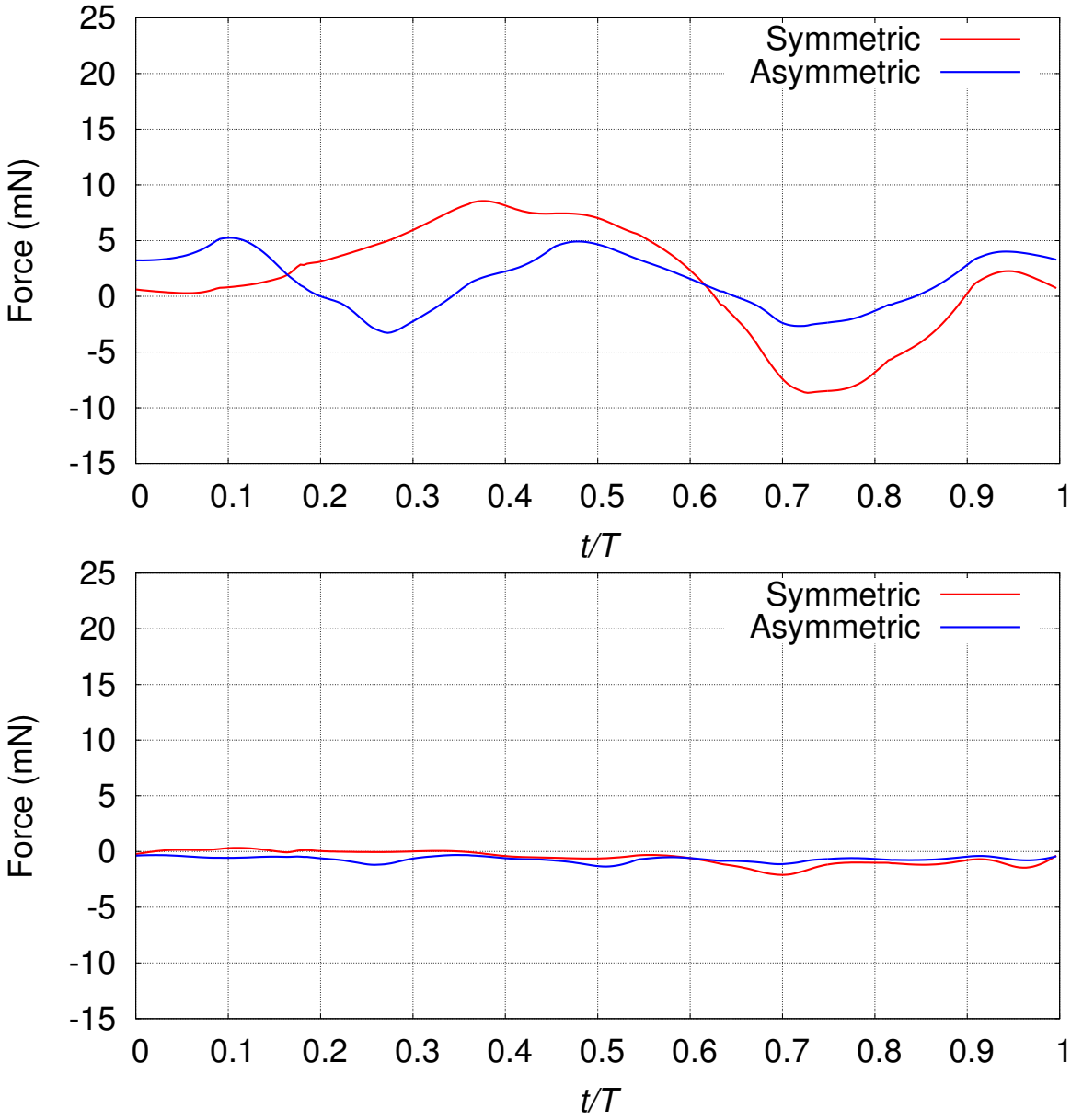


Figure 9.10: Total lift (top) and thrust (bottom) for the symmetric and asymmetric flapping.

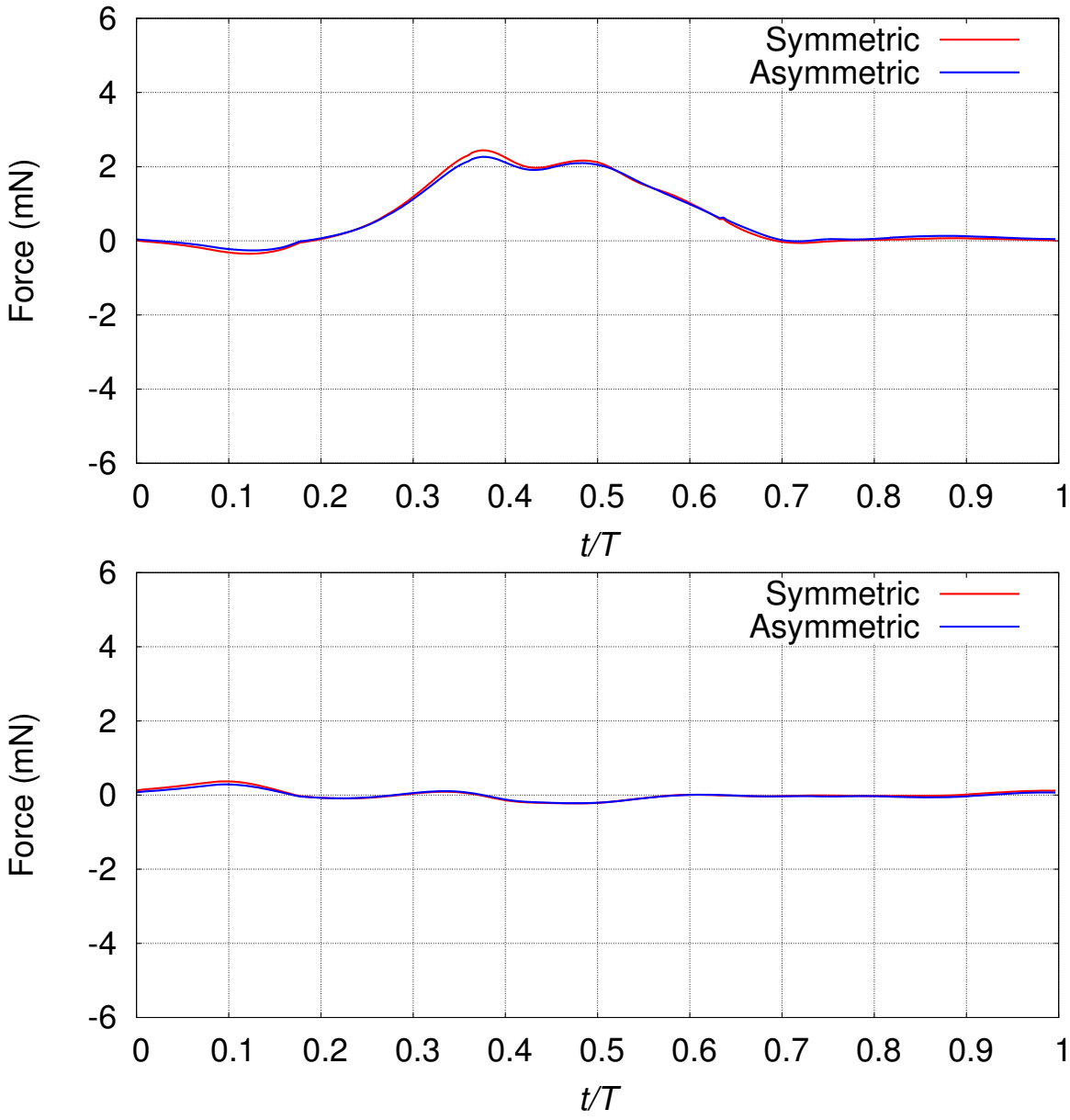


Figure 9.11: Right FW lift (top) and thrust (bottom) for the symmetric and asymmetric flapping.

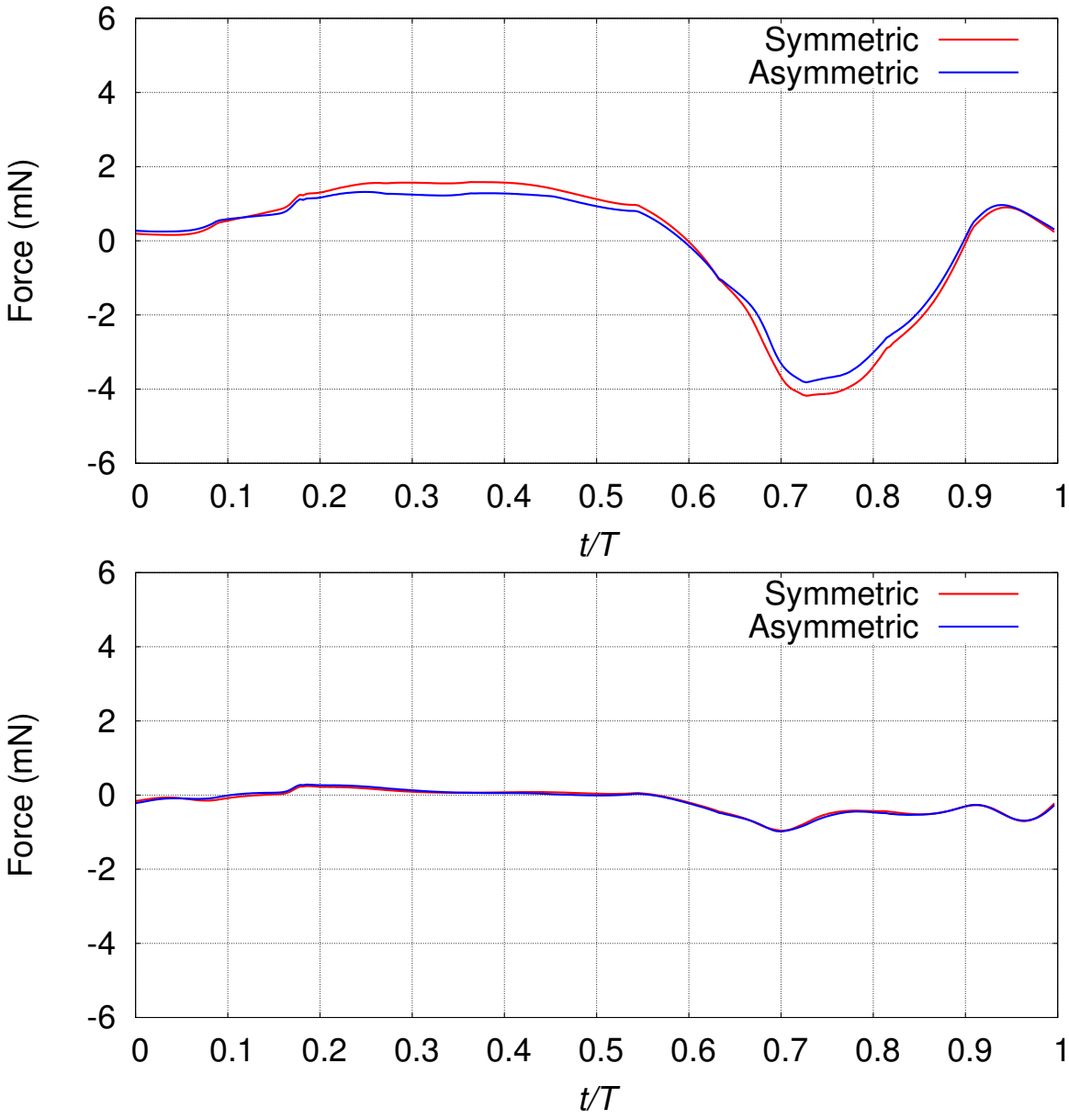


Figure 9.12: Right HW lift (top) and thrust (bottom) for the symmetric and asymmetric flapping.

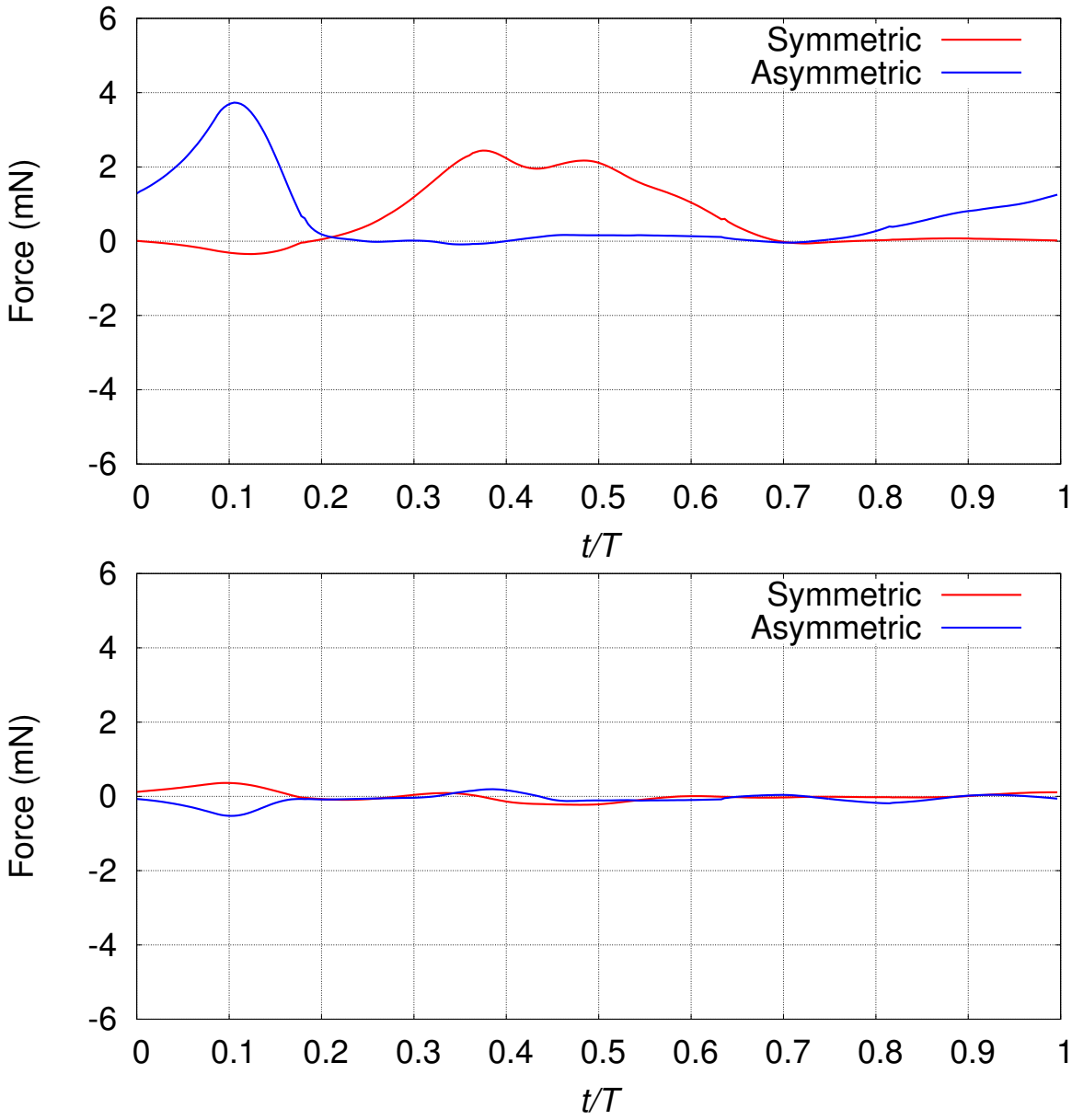


Figure 9.13: Left FW lift (top) and thrust (bottom) for the symmetric and asymmetric flapping.

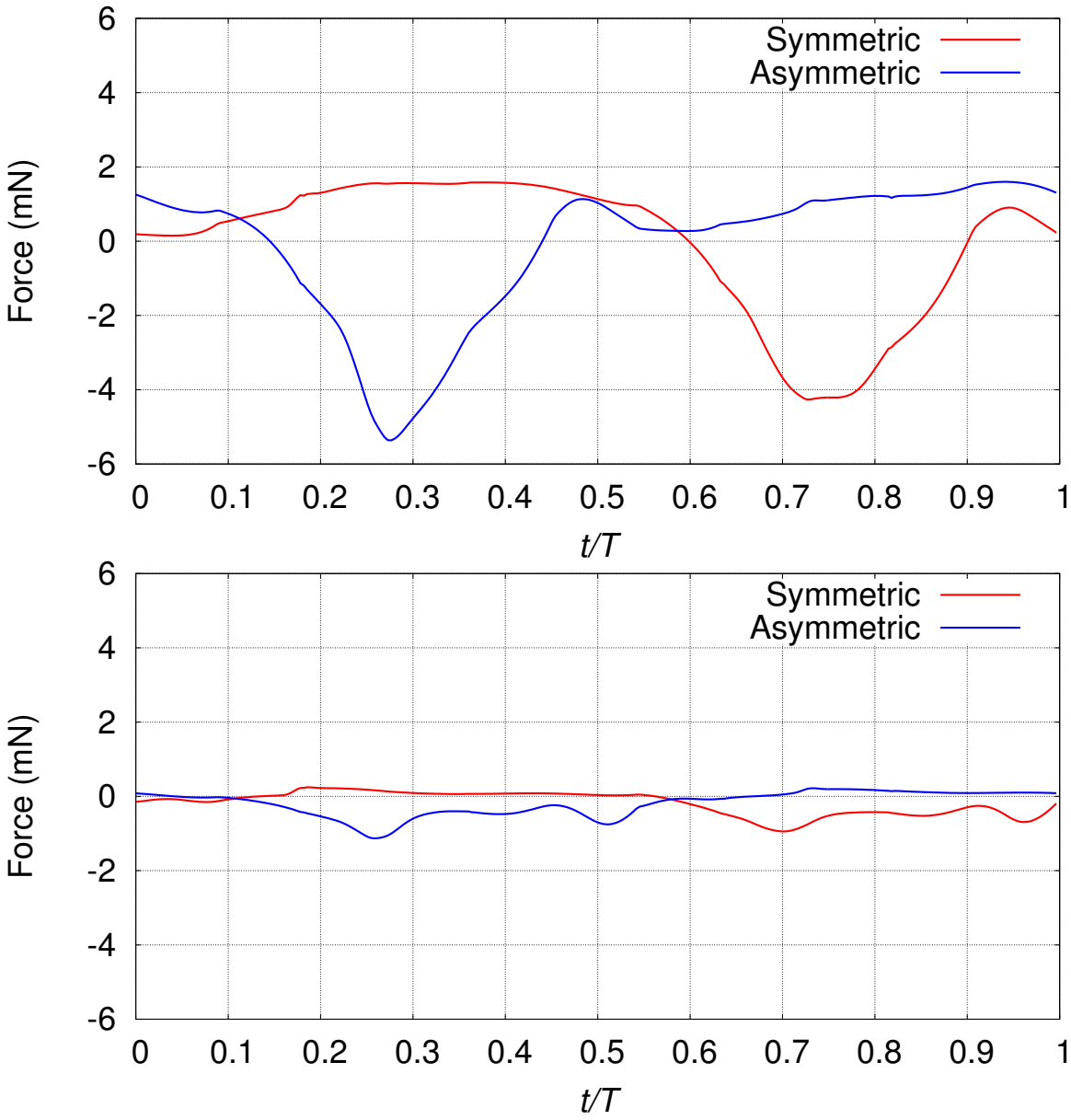


Figure 9.14: Left HW lift (top) and thrust (bottom) for the symmetric and asymmetric flapping.

As expected, for the right wings the lift and thrust for the symmetric and asymmetric flapping are very similar. For the left wings they are quite different. The left HW lift and thrust are basically shifted in time in a fashion that reflects the flapping shift in time. The left FW, on the other hand, has very different lift and thrust profiles. The average left FW lift and thrust for the symmetric flapping are 0.63 mN and 0.01 mN. For the asymmetric flapping, the values are 0.64 mN and -0.08 mN. The total lift and thrust for the symmetric flapping are 1.43 mN and -0.56 mN. For the asymmetric flapping, the values are 1.40 mN and -0.68 mN.

Chapter 10

Conclusions

We repeat here the concluding remarks from [2]. We have focused on a SCFSI analysis of flapping-wing aerodynamics of an MAV. The wing motion and deformation data, whether prescribed fully or partially, was from an actual locust, extracted from high-speed, multi-camera video recordings of the locust in a wind tunnel. The core computational FSI technology is based on the DSD/SST formulation, specifically, DSD/SST-VMST (also called ST-VMS), which is the version of the DSD/SST formulation derived in conjunction with the VMS method. This was supplemented with special ST techniques targeting flapping-wing aerodynamics, and the common feature for much of these special ST techniques is using NURBS basis functions in temporal representation. That includes temporal representation of the wing and mesh motion. Temporal representation with NURBS is also a key feature of the remeshing process. The structural mechanics computations were mostly based on the Kirchhoff–Love shell model. We also carried out, for comparison purpose, some test computations with the membrane model. The sequential-coupling technique, which is applicable to some classes of FSI problems, especially those with temporally-periodic behavior, performed well in FSI computations of the flapping-wing aerodynamics we considered here. In addition to the straight-flight case, we analyzed cases where the MAV

body had rolling, pitching, or rolling and pitching motion, where we assumed that the MAV was somehow able to perform such maneuvers. We showed how these maneuvers influence the lift and thrust.

Bibliography

[1] K. Takizawa, B. Henicke, A. Puntel, N. Kostov, and T.E. Tezduyar, "Space–time techniques for computational aerodynamics modeling of flapping wings of an actual locust", *Computational Mechanics*, **50** (2012) 743–760.

[2] K. Takizawa, T.E. Tezduyar, and N. Kostov, "Sequentially-coupled space–time FSI analysis of bio-inspired flapping-wing aerodynamics of an MAV", *Computational Mechanics*, to be submitted, 2014.

[3] K. Takizawa, B. Henicke, A. Puntel, T. Spielman, and T.E. Tezduyar, "Space–time computational techniques for the aerodynamics of flapping wings", *Journal of Applied Mechanics*, **79** (2012) 010903.

[4] T.E. Tezduyar, "Stabilized finite element formulations for incompressible flow computations", *Advances in Applied Mechanics*, **28** (1992) 1–44.

[5] T.E. Tezduyar, M. Behr, and J. Liou, "A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space–time procedure: I. The concept and the preliminary numerical tests", *Computer Methods in Applied Mechanics and Engineering*, **94** (1992) 339–351.

[6] T.E. Tezduyar, M. Behr, S. Mittal, and J. Liou, "A new strategy for finite element computations involving moving boundaries and interfaces – the deforming-spatial-domain/space–time procedure: II. Computation of free-surface flows,

two-liquid flows, and flows with drifting cylinders", *Computer Methods in Applied Mechanics and Engineering*, **94** (1992) 353–371.

[7] T.E. Tezduyar, "Computation of moving boundaries and interfaces and stabilization parameters", *International Journal for Numerical Methods in Fluids*, **43** (2003) 555–575.

[8] T.E. Tezduyar and S. Sathe, "Modeling of fluid–structure interactions with the space–time finite elements: Solution techniques", *International Journal for Numerical Methods in Fluids*, **54** (2007) 855–900.

[9] K. Takizawa and T.E. Tezduyar, "Multiscale space–time fluid–structure interaction techniques", *Computational Mechanics*, **48** (2011) 247–267.

[10] K. Takizawa and T.E. Tezduyar, "Space–time fluid–structure interaction methods", *Mathematical Models and Methods in Applied Sciences*, **22** (2012) 1230001.

[11] Y. Bazilevs, K. Takizawa, and T.E. Tezduyar, *Computational Fluid–Structure Interaction: Methods and Applications*. Wiley, 2013.

[12] A.N. Brooks and T.J.R. Hughes, "Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations", *Computer Methods in Applied Mechanics and Engineering*, **32** (1982) 199–259.

[13] T.E. Tezduyar, S. Mittal, S.E. Ray, and R. Shih, "Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements", *Computer Methods in Applied Mechanics and Engineering*, **95** (1992) 221–242.

[14] T.J.R. Hughes, "Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles, and the origins of stabilized methods", *Computer Methods in Applied Mechanics and Engineering*, **127** (1995) 387–401.

[15] T.J.R. Hughes, A.A. Oberai, and L. Mazzei, "Large eddy simulation of turbulent channel flows by the variational multiscale method", *Physics of Fluids*, **13** (2001) 1784–1799.

[16] Y. Bazilevs, V.M. Calo, J.A. Cottrell, T.J.R. Hughes, A. Reali, and G. Scovazzi, "Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows", *Computer Methods in Applied Mechanics and Engineering*, **197** (2007) 173–201.

[17] Y. Bazilevs and I. Akkerman, "Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and the residual–based variational multiscale method", *Journal of Computational Physics*, **229** (2010) 3402–3414.

[18] T.J.R. Hughes, W.K. Liu, and T.K. Zimmermann, "Lagrangian–Eulerian finite element formulation for incompressible viscous flows", *Computer Methods in Applied Mechanics and Engineering*, **29** (1981) 329–349.

[19] R. Ohayon, "Reduced symmetric models for modal analysis of internal structural-acoustic and hydroelastic-sloshing systems", *Computer Methods in Applied Mechanics and Engineering*, **190** (2001) 3009–3019.

[20] E.H. van Brummelen and R. de Borst, "On the nonnormality of subiteration for a fluid-structure interaction problem", *SIAM Journal on Scientific Computing*, **27** (2005) 599–621.

[21] Y. Bazilevs, V.M. Calo, Y. Zhang, and T.J.R. Hughes, "Isogeometric fluid–structure interaction analysis with applications to arterial blood flow", *Computational Mechanics*, **38** (2006) 310–322.

[22] R.A. Khurram and A. Masud, "A multiscale/stabilized formulation of the incompressible Navier–Stokes equations for moving boundary flows and fluid–structure interaction", *Computational Mechanics*, **38** (2006) 403–416.

[23] R. Lohner, J.R. Cebral, C. Yang, J.D. Baum, E.L. Mestreau, and O. Soto, "Extending the range of applicability of the loose coupling approach for FSI simulations", in H.-J. Bungartz and M. Schafer, editors, *Fluid–Structure Interaction*, volume 53 of *Lecture Notes in Computational Science and Engineering*, 82–100, Springer, 2006.

[24] K.-U. Bletzinger, R. Wuchner, and A. Kupzok, "Algorithmic treatment of shells and free form-membranes in FSI", in H.-J. Bungartz and M. Schafer, editors, *Fluid–Structure Interaction*, volume 53 of *Lecture Notes in Computational Science and Engineering*, 336–355, Springer, 2006.

[25] Y. Bazilevs, V.M. Calo, T.J.R. Hughes, and Y. Zhang, "Isogeometric fluid–structure interaction: theory, algorithms, and computations", *Computational Mechanics*, **43** (2008) 3–37.

[26] W.G. Dettmer and D. Peric, "On the coupling between fluid flow and mesh motion in the modelling of fluid–structure interaction", *Computational Mechanics*, **43** (2008) 81–90.

[27] Y. Bazilevs, J.R. Gohean, T.J.R. Hughes, R.D. Moser, and Y. Zhang, "Patient-specific isogeometric fluid–structure interaction analysis of thoracic aortic blood flow due to implantation of the Jarvik 2000 left ventricular assist device", *Computer Methods in Applied Mechanics and Engineering*, **198** (2009) 3534–3550.

[28] Y. Bazilevs, M.-C. Hsu, D. Benson, S. Sankaran, and A. Marsden, "Computational fluid–structure interaction: Methods and application to a total cavopulmonary connection", *Computational Mechanics*, **45** (2009) 77–89.

[29] R. Calderer and A. Masud, "A multiscale stabilized ALE formulation for incompressible flows with moving boundaries", *Computational Mechanics*, **46** (2010) 185–197.

[30] Y. Bazilevs, M.-C. Hsu, Y. Zhang, W. Wang, X. Liang, T. Kvamsdal, R. Brekken, and J. Isaksen, "A fully-coupled fluid–structure interaction simulation of cerebral aneurysms", *Computational Mechanics*, **46** (2010) 3–16.

[31] Y. Bazilevs, M.-C. Hsu, Y. Zhang, W. Wang, T. Kvamsdal, S. Hentschel, and J. Isaksen, "Computational fluid–structure interaction: Methods and application to cerebral aneurysms", *Biomechanics and Modeling in Mechanobiology*, **9** (2010) 481–498.

[32] Y. Bazilevs, M.-C. Hsu, I. Akkerman, S. Wright, K. Takizawa, B. Henicke, T. Spielman, and T.E. Tezduyar, "3D simulation of wind turbine rotors at full scale. Part I: Geometry modeling and aerodynamics", *International Journal for Numerical Methods in Fluids*, **65** (2011) 207–235.

[33] Y. Bazilevs, M.-C. Hsu, J. Kiendl, R. Wüchner, and K.-U. Bletzinger, "3D simulation of wind turbine rotors at full scale. Part II: Fluid–structure interaction modeling with composite blades", *International Journal for Numerical Methods in Fluids*, **65** (2011) 236–253.

[34] I. Akkerman, Y. Bazilevs, C.E. Kees, and M.W. Farthing, "Isogeometric analysis of free-surface flow", *Journal of Computational Physics*, **230** (2011) 4137–4152.

[35] M.-C. Hsu and Y. Bazilevs, "Blood vessel tissue prestress modeling for vascular fluid–structure interaction simulations", *Finite Elements in Analysis and Design*, **47** (2011) 593–599.

[36] S. Nagaoka, Y. Nakabayashi, G. Yagawa, and Y.J. Kim, "Accurate fluid–structure interaction computations using elements without mid-side nodes", *Computational Mechanics*, **48** (2011) 269–276.

[37] Y. Bazilevs, M.-C. Hsu, K. Takizawa, and T.E. Tezduyar, "ALE-VMS and ST-VMS methods for computer modeling of wind-turbine rotor aerodynamics and fluid–structure interaction", *Mathematical Models and Methods in Applied Sciences*, **22** (2012) 1230002.

[38] I. Akkerman, Y. Bazilevs, D.J. Benson, M.W. Farthing, and C.E. Kees, "Free-surface flow and fluid–object interaction modeling with emphasis on ship hydrodynamics", *Journal of Applied Mechanics*, **79** (2012) 010905.

[39] M.-C. Hsu, I. Akkerman, and Y. Bazilevs, "Wind turbine aerodynamics using ALE-VMS: Validation and role of weakly enforced boundary conditions", *Computational Mechanics*, **50** (2012) 499–511.

[40] M.-C. Hsu and Y. Bazilevs, "Fluid–structure interaction modeling of wind turbines: simulating the full machine", *Computational Mechanics*, **50** (2012) 821–833.

[41] I. Akkerman, J. Dunaway, J. Kvandal, J. Spinks, and Y. Bazilevs, "Toward free-surface modeling of planing vessels: simulation of the Fridsma hull using ALE-VMS", *Computational Mechanics*, **50** (2012) 719–727.

[42] S. Minami, H. Kawai, and S. Yoshimura, "Parallel BDD-based monolithic approach for acoustic fluid–structure interaction", *Computational Mechanics*, **50** (2012) 707–718.

[43] T. Miras, J.-S. Schotte, and R. Ohayon, "Energy approach for static and linearized dynamic studies of elastic structures containing incompressible liquids with capillarity: a theoretical formulation", *Computational Mechanics*, **50** (2012) 729–741.

[44] T.M. van Opstal, E.H. van Brummelen, R. de Borst, and M.R. Lewis, "A finite-element/boundary-element method for large-displacement fluid–structure interaction", *Computational Mechanics*, **50** (2012) 779–788.

[45] J.Y. Yao, G.R. Liu, D.A. Narmoneva, R.B. Hinton, and Z.-Q. Zhang, "Immersed smoothed finite element method for fluid–structure interaction simulation of aortic valves", *Computational Mechanics*, **50** (2012) 789–804.

[46] A. Larese, R. Rossi, E. Onate, and S.R. Idelsohn, "A coupled PFEM–Eulerian approach for the solution of porous FSI problems", *Computational Mechanics*, **50** (2012) 805–819.

[47] Y. Bazilevs, K. Takizawa, and T.E. Tezduyar, "Challenges and directions in computational fluid–structure interaction", *Mathematical Models and Methods in Applied Sciences*, **23** (2013) 215–221.

[48] A. Korobenko, M.-C. Hsu, I. Akkerman, J. Tippmann, and Y. Bazilevs, "Structural mechanics modeling and FSI simulation of wind turbines", *Mathematical Models and Methods in Applied Sciences*, **23** (2013) 249–272.

[49] J.Y. Yao, G.R. Liu, D. Qian, C.L. Chen, and G.X. Xu, "A moving-mesh gradient smoothing method for compressible CFD problems", *Mathematical Models and Methods in Applied Sciences*, **23** (2013) 273–305.

[50] K. Kamran, R. Rossi, E. Onate, and S.R. Idelsohn, "A compressible Lagrangian framework for modeling the fluid–structure interaction in the underwater im-

plosion of an aluminum cylinder", *Mathematical Models and Methods in Applied Sciences*, **23** (2013) 339–367.

[51] M.-C. Hsu, I. Akkerman, and Y. Bazilevs, "Finite element simulation of wind turbine aerodynamics: Validation study using NREL Phase VI experiment", *Wind Energy*, published online, DOI: 10.1002/we.1599, 2013.

[52] T. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, and S. Mittal, "Parallel finite-element computation of 3D flows", *Computer*, **26** (1993) 27–36.

[53] A.A. Johnson and T.E. Tezduyar, "Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces", *Computer Methods in Applied Mechanics and Engineering*, **119** (1994) 73–94.

[54] T.E. Tezduyar, "Finite element methods for flow problems with moving boundaries and interfaces", *Archives of Computational Methods in Engineering*, **8** (2001) 83–130.

[55] T. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, V. Kalro, and M. Litke, "Flow simulation and high performance computing", *Computational Mechanics*, **18** (1996) 397–412.

[56] K. Takizawa, B. Henicke, T.E. Tezduyar, M.-C. Hsu, and Y. Bazilevs, "Stabilized space–time computation of wind-turbine rotor aerodynamics", *Computational Mechanics*, **48** (2011) 333–344.

[57] K. Takizawa, B. Henicke, D. Montes, T.E. Tezduyar, M.-C. Hsu, and Y. Bazilevs, "Numerical-performance studies for the stabilized space–time computation of wind-turbine rotor aerodynamics", *Computational Mechanics*, **48** (2011) 647–657.

[58] K. Takizawa, T.E. Tezduyar, S. McIntyre, N. Kostov, R. Kolesar, and C. Habluetzel, "Space–time VMS computation of wind-turbine rotor and tower aerodynamics", *Computational Mechanics*, published online, DOI: 10.1007/s00466-013-0888-x, July 2013.

[59] S. Mittal and T.E. Tezduyar, "Parallel finite element simulation of 3D incompressible flows – Fluid-structure interactions", *International Journal for Numerical Methods in Fluids*, **21** (1995) 933–953.

[60] V. Kalro and T.E. Tezduyar, "A parallel 3D computational method for fluid–structure interactions in parachute systems", *Computer Methods in Applied Mechanics and Engineering*, **190** (2000) 321–332.

[61] T. Tezduyar and Y. Osawa, "Fluid–structure interactions of a parachute crossing the far wake of an aircraft", *Computer Methods in Applied Mechanics and Engineering*, **191** (2001) 717–726.

[62] T.E. Tezduyar, S. Sathe, R. Keedy, and K. Stein, "Space–time finite element techniques for computation of fluid–structure interactions", *Computer Methods in Applied Mechanics and Engineering*, **195** (2006) 2002–2027.

[63] M. Manguoglu, A.H. Sameh, T.E. Tezduyar, and S. Sathe, "A nested iterative scheme for computation of incompressible flows in long domains", *Computational Mechanics*, **43** (2008) 73–80.

[64] T.E. Tezduyar, S. Sathe, J. Pausewang, M. Schwaab, J. Christopher, and J. Crabtree, "Interface projection techniques for fluid–structure interaction modeling with moving-mesh methods", *Computational Mechanics*, **43** (2008) 39–49.

[65] T.E. Tezduyar, S. Sathe, M. Schwaab, J. Pausewang, J. Christopher, and J. Crabtree, "Fluid–structure interaction modeling of ringsail parachutes", *Computational Mechanics*, **43** (2008) 133–142.

[66] S. Sathe and T.E. Tezduyar, "Modeling of fluid–structure interactions with the space–time finite elements: Contact problems", *Computational Mechanics*, **43** (2008) 51–60.

[67] R. Torii, M. Oshima, T. Kobayashi, K. Takagi, and T.E. Tezduyar, "Fluid–structure interaction modeling of a patient-specific cerebral aneurysm: Influence of structural modeling", *Computational Mechanics*, **43** (2008) 151–159.

[68] T.E. Tezduyar, K. Takizawa, C. Moorman, S. Wright, and J. Christopher, "Multiscale sequentially-coupled arterial FSI technique", *Computational Mechanics*, **46** (2010) 17–29.

[69] K. Takizawa, C. Moorman, S. Wright, J. Christopher, and T.E. Tezduyar, "Wall shear stress calculations in space–time finite element computation of arterial fluid–structure interactions", *Computational Mechanics*, **46** (2010) 31–41.

[70] M. Manguoglu, K. Takizawa, A.H. Sameh, and T.E. Tezduyar, "Solution of linear systems in arterial fluid mechanics computations with boundary layer mesh refinement", *Computational Mechanics*, **46** (2010) 83–89.

[71] R. Torii, M. Oshima, T. Kobayashi, K. Takagi, and T.E. Tezduyar, "Role of 0D peripheral vasculature model in fluid–structure interaction modeling of aneurysms", *Computational Mechanics*, **46** (2010) 43–52.

[72] T.E. Tezduyar, K. Takizawa, C. Moorman, S. Wright, and J. Christopher, "Space–time finite element computation of complex fluid–structure interactions", *International Journal for Numerical Methods in Fluids*, **64** (2010) 1201–1218.

[73] K. Takizawa, T. Spielman, and T.E. Tezduyar, "Space–time FSI modeling and dynamical analysis of spacecraft parachutes and parachute clusters", *Computational Mechanics*, **48** (2011) 345–364.

[74] M. Manguoglu, K. Takizawa, A.H. Sameh, and T.E. Tezduyar, "A parallel sparse algorithm targeting arterial fluid mechanics computations", *Computational Mechanics*, **48** (2011) 377–384.

[75] K. Takizawa and T.E. Tezduyar, "Computational methods for parachute fluid–structure interactions", *Archives of Computational Methods in Engineering*, **19** (2012) 125–169.

[76] K. Takizawa, Y. Bazilevs, and T.E. Tezduyar, "Space–time and ALE-VMS techniques for patient-specific cardiovascular fluid–structure interaction modeling", *Archives of Computational Methods in Engineering*, **19** (2012) 171–225.

[77] K. Takizawa, M. Fritze, D. Montes, T. Spielman, and T.E. Tezduyar, "Fluid–structure interaction modeling of ringsail parachutes with disreefing and modified geometric porosity", *Computational Mechanics*, **50** (2012) 835–854.

[78] K. Takizawa, D. Montes, M. Fritze, S. McIntyre, J. Boben, and T.E. Tezduyar, "Methods for FSI modeling of spacecraft parachute dynamics and cover separation", *Mathematical Models and Methods in Applied Sciences*, **23** (2013) 307–338.

[79] K. Takizawa, T.E. Tezduyar, J. Boben, N. Kostov, C. Boswell, and A. Buscher, "Fluid–structure interaction modeling of clusters of spacecraft parachutes with modified geometric porosity", *Computational Mechanics*, **52** (2013) 1351–1364.

[80] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs, "Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement", *Computer Methods in Applied Mechanics and Engineering*, **194** (2005) 4135–4195.

[81] Y. Bazilevs and T.J.R. Hughes, "NURBS-based isogeometric analysis for the computation of flows about rotating components", *Computational Mechanics*, **43** (2008) 143–150.

[82] K. Takizawa, N. Kostov, A. Puntel, B. Henicke, and T.E. Tezduyar, "Space–time computational analysis of bio-inspired flapping-wing aerodynamics of a micro aerial vehicle", *Computational Mechanics*, **50** (2012) 761–778.

[83] K. Takizawa, B. Henicke, A. Puntel, N. Kostov, and T.E. Tezduyar, "Computer modeling techniques for flapping-wing aerodynamics of a locust", *Computers & Fluids*, **85** (2013) 125–134.

[84] T.E. Tezduyar, "Finite element methods for fluid dynamics with moving boundaries and interfaces", in E. Stein, R.D. Borst, and T.J.R. Hughes, editors, *Encyclopedia of Computational Mechanics*, Volume 3: Fluids, Chapter 17, John Wiley & Sons, 2004.

[85] T.E. Tezduyar, "Finite elements in fluids: Special methods and enhanced solution techniques", *Computers & Fluids*, **36** (2007) 207–223.

[86] K. Takizawa, C. Moorman, S. Wright, T. Spielman, and T.E. Tezduyar, "Fluid–structure interaction modeling and performance analysis of the Orion spacecraft parachutes", *International Journal for Numerical Methods in Fluids*, **65** (2011) 271–285.

[87] K. Takizawa, S. Wright, C. Moorman, and T.E. Tezduyar, "Fluid–structure interaction modeling of parachute clusters", *International Journal for Numerical Methods in Fluids*, **65** (2011) 286–307.

[88] T.E. Tezduyar, S. Sathe, M. Schwaab, and B.S. Conklin, "Arterial fluid mechanics modeling with the stabilized space–time fluid–structure interaction tech-

nique", *International Journal for Numerical Methods in Fluids*, **57** (2008) 601–629.

[89] T.E. Tezduyar, M. Schwaab, and S. Sathe, "Sequentially-Coupled Arterial Fluid–Structure Interaction (SCAFSI) technique", *Computer Methods in Applied Mechanics and Engineering*, **198** (2009) 3524–3533.

[90] J. Kiendl, K.U. Bletzinger, J. Linhard, and R. Wüchner, "Isogeometric shell analysis with Kirchhoff–Love elements", *Computer Methods in Applied Mechanics and Engineering*, **198** (2009) 3902–3914.

[91] J. Kiendl, Y. Bazilevs, M.-C. Hsu, R. Wüchner, and K.-U. Bletzinger, "The bending strip method for isogeometric analysis of Kirchhoff–Love shell structures comprised of multiple patches", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 2403–2416.

[92] T.E. Tezduyar and Y. Osawa, "Finite element stabilization parameters computed from element matrices and vectors", *Computer Methods in Applied Mechanics and Engineering*, **190** (2000) 411–430.

[93] J.E. Akin, T. Tezduyar, M. Ungor, and S. Mittal, "Stabilization parameters and Smagorinsky turbulence model", *Journal of Applied Mechanics*, **70** (2003) 2–9.

[94] J.E. Akin and T.E. Tezduyar, "Calculation of the advective limit of the SUPG stabilization parameter for linear and higher-order elements", *Computer Methods in Applied Mechanics and Engineering*, **193** (2004) 1909–1922.

[95] L. Catabriga, A.L.G.A. Coutinho, and T.E. Tezduyar, "Compressible flow SUPG parameters computed from element matrices", *Communications in Numerical Methods in Engineering*, **21** (2005) 465–476.

[96] T.E. Tezduyar, M. Senga, and D. Vicker, "Computation of inviscid supersonic flows around cylinders and spheres with the SUPG formulation and YZβ shock-capturing", *Computational Mechanics*, **38** (2006) 469–481.

[97] T.E. Tezduyar and S. Sathe, "Enhanced-discretization selective stabilization procedure (EDSSP)", *Computational Mechanics*, **38** (2006) 456–468.

[98] E. Onate, A. Valls, and J. Garcia, "FIC/FEM formulation with matrix stabilizing terms for incompressible flows at low and high Reynolds numbers", *Computational Mechanics*, **38** (2006) 440–455.

[99] A. Corsini, F. Rispoli, A. Santoriello, and T.E. Tezduyar, "Improved discontinuity-capturing finite element techniques for reaction effects in turbulence computation", *Computational Mechanics*, **38** (2006) 356–364.

[100] L. Catabriga, A.L.G.A. Coutinho, and T.E. Tezduyar, "Compressible flow SUPG parameters computed from degree-of-freedom submatrices", *Computational Mechanics*, **38** (2006) 334–343.

[101] T.E. Tezduyar, "Finite elements in fluids: Stabilized formulations and moving boundaries and interfaces", *Computers & Fluids*, **36** (2007) 191–206.

[102] F. Rispoli, A. Corsini, and T.E. Tezduyar, "Finite element computation of turbulent flows with the discontinuity-capturing directional dissipation (DCDD)", *Computers & Fluids*, **36** (2007) 121–126.

[103] A. Corsini, C. Iossa, F. Rispoli, and T.E. Tezduyar, "A DRD finite element formulation for computing turbulent reacting flows in gas turbine combustors", *Computational Mechanics*, **46** (2010) 159–167.

[104] M.-C. Hsu, Y. Bazilevs, V.M. Calo, T.E. Tezduyar, and T.J.R. Hughes, "Improving stability of stabilized and multiscale formulations in flow simulations at

small time steps", *Computer Methods in Applied Mechanics and Engineering*, **199** (2010) 828–840.

[105] A. Corsini, F. Rispoli, and T.E. Tezduyar, "Stabilized finite element computation of NOx emission in aero-engine combustors", *International Journal for Numerical Methods in Fluids*, **65** (2011) 254–270.

[106] A. Corsini, F. Rispoli, and T.E. Tezduyar, "Computer modeling of wave-energy air turbines with the SUPG/PSPG formulation and discontinuity-capturing technique", *Journal of Applied Mechanics*, **79** (2012) 010910.

[107] T.J.R. Hughes and G. Sangalli, "Variational multiscale analysis: the fine-scale Green's function, projection, optimization, localization, and stabilized methods", *SIAM Journal of Numerical Analysis*, **45** (2007) 539–557.

[108] T.J.R. Hughes and A.A. Oberai, "Calculation of shear stress in Fourier–Galerkin formulations of turbulent channel flows: projection, the Dirichlet filter and conservation", *Journal of Computational Physics*, **188** (2003) 281–295.

[109] A. Lo, *Nonlinear Dynamic Analysis of Cable and Membrane Structure*, Ph.D. thesis, Department of Civil Engineering, Oregon State University, 1982.

[110] R.J. Benney, K.R. Stein, J.W. Leonard, and M.L. Accorsi, "Current 3-D structural dynamic finite element modeling capabilities", in *Proceedings of AIAA 14th Aerodynamic Decelerator Systems Technology Conference*, AIAA Paper 97-1506, San Francisco, California, (1997).

[111] K. Stein, R. Benney, V. Kalro, T.E. Tezduyar, J. Leonard, and M. Accorsi, "Parachute fluid–structure interactions: 3-D Computation", *Computer Methods in Applied Mechanics and Engineering*, **190** (2000) 373–386.

[112] J. Chung and G.M. Hulbert, "A time integration algorithm for structural dynamics withimproved numerical dissipation: The generalized-α method", *Journal of Applied Mechanics*, **60** (1993) 371–75.

[113] T.E. Tezduyar, S. Sathe, T. Cragin, B. Nanna, B.S. Conklin, J. Pausewang, and M. Schwaab, "Modeling of fluid–structure interactions with the space–time finite elements: Arterial fluid mechanics", *International Journal for Numerical Methods in Fluids*, **54** (2007) 901–922.

[114] Y. Saad and M. Schultz, "GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems", *SIAM Journal of Scientific and Statistical Computing*, **7** (1986) 856–869.

[115] G. Karypis and V. Kumar, "A fast and high quality multilevel scheme for partitioning irregular graphs", *SIAM Journal of Scientific Computing*, **20** (1998) 359–392.

[116] S.A. Combes and T.L. Daniel, "Flexural stiffness in insect wings: I. Scaling and the influence of wing venation", *The Journal of Experimental Biology*, **206** (2003) 2979–2987.