REPORT DOCUMENTATION PAGE

2

Form Approved OMB NO. 0704-0188

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

4. TITLE AND SUBTITLE
Space--Time Interface-Tracking Computations with Contact Between Solid Surfaces

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

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS
Austin J. Buscher

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.20

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
To address the computational challenges associated with contact between moving solid surfaces, such as those in cardiovascular fluid--structure interaction (FSI), parachute FSI, and flapping-wing aerodynamics, we introduce a space--time (ST) interface-tracking method that can deal with topology change (TC). In cardiovascular FSI, our primary target is heart valves. The method is a new version of the Deforming-Spatial-Domain/Stabilized ST (DSD/SST) method, and we call it ST-TC. It includes a master--slave system that maintains the connectivity of the ``parent'' mesh when there is contact between the moving interfaces. It is an efficient, practical alternative to using

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 Interface-Tracking Computations with Contact Between Solid Surfaces

ABSTRACT

To address the computational challenges associated with contact between moving solid surfaces, such as those in cardiovascular fluid--structure interaction (FSI), parachute FSI, and flapping-wing aerodynamics, we introduce a space--time (ST) interface-tracking method that can deal with topology change (TC). In cardiovascular FSI, our primary target is heart valves. The method is a new version of the Deforming-Spatial-Domain/Stabilized ST (DSD/SST) method, and we call it ST-TC. It includes a master--slave system that maintains the connectivity of the ``parent'' mesh when there is contact between the moving interfaces. It is an efficient, practical alternative to using unstructured ST meshes, but without giving up on the accurate representation of the interface or consistent representation of the interface motion. We explain the method with conceptual examples and present 2D and 3D test computations with models representative of the classes of problems we are targeting.

RICE UNIVERSITY

Space–Time Interface-Tracking Computations with Contact Between Solid Surfaces

by

Austin J. Buscher, 2nd Lt, USAF

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

Master of Science

Approved, Thesis Committee:

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

J. E. Akin
Professor of Mechanical Engineering and
Materials Science and Professor of
Computational and Applied Mathematics

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

K. Takizawa
Associate Professor in Department of
Modern Mechanical Engineering and
Waseda Institute for Advanced Study
Waseda University, Tokyo, Japan

Houston, Texas
April 2014

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the United States Air Force, Department of Defense, or the U. S. Government.

Abstract

Space–Time Interface-Tracking Computations with Contact Between Solid Surfaces

by

Austin J. Buscher

To address the computational challenges associated with contact between moving solid surfaces, such as those in cardiovascular fluid–structure interaction (FSI), parachute FSI, and flapping-wing aerodynamics, we introduce a space–time (ST) interface-tracking method that can deal with topology change (TC). In cardiovascular FSI, our primary target is heart valves. The method is a new version of the Deforming-Spatial-Domain/Stabilized ST (DSD/SST) method, and we call it ST-TC. It includes a master–slave system that maintains the connectivity of the "parent" mesh when there is contact between the moving interfaces. It is an efficient, practical alternative to using unstructured ST meshes, but without giving up on the accurate representation of the interface or consistent representation of the interface motion. We explain the method with conceptual examples and present 2D and 3D test computations with models representative of the classes of problems we are targeting.

Acknowledgments

First I would like to thank Dr. Tayfun Tezduyar, who provided me with this amazing opportunity to pursue this graduate degree at Rice University. Memories of his kindness, motivating spirit, engineering intuition and work ethic will continue inspire me for the rest of my life.

I would also like to thank Dr. Kenji Takizawa. Dr. Takizawa is the smartest and hardest working person I have ever met. Without his intellect and hard work, this work would not have been possible.

I would also like to thank my fellow team members. Dr. Nikolay Kostov was an excellent mentor and a brilliant collaborator. Spenser Mcintyre and Joe Boben were motivating mentors. Ryan, Cody, Casey, and Ken were all hard-working comrades that I will greatly miss.

I would also like to thank my wife, Hafsah, who was and always will be supportive of my ambitions and dreams.

Lastly, I would like to thank Dr. Ed Akin and Dr. Andrew Meade for taking the time to sit on my thesis committee.

This work was supported in part by the Rice–Waseda Research Agreement.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The material in this chapter is from [59]. Flow problems with moving boundaries and interfaces (MBI) are encountered frequently in engineering analysis and design. They include fluid–structure interaction (FSI), fluid–object interaction (FOI), fluid–particle interaction (FPI), free-surface and multi-fluid flows, and flows with solid surfaces in fast, linear or rotational relative motion. These problems pose some of the most formidable computational challenges. The challenges include contact between moving solid surfaces and other cases of topology change (TC), such as those in cardiovascular FSI, parachute FSI, and flapping-wing aerodynamics. A method for flows with MBI can be viewed as an interface-tracking (moving-mesh) technique or an interface-capturing (nonmoving-mesh) technique, or possibly a combination of the two.

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 [26] is the most widely used moving-mesh technique, with increased emphasis on FSI in recent years (see, for example, [41, 78, 7, 30, 33, 17, 6, 20, 8, 10, 18, 14, 13, 9, 11, 4, 24, 40, 12, 3, 22, 25, 5, 37, 38, 79, 81, 32, 16, 15,

31, 82, 29, 23]). The Deforming-Spatial-Domain/Stabilized Space–Time (DSD/SST) method [65, 68, 69, 67, 70, 54, 56, 16] is also a general-purpose moving-mesh technique.

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 [63, 27, 66, 70] 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.

Over the past 20 years the DSD/SST method has been applied to some of the most challenging flow problems with MBI. The classes of problems solved include the free-surface and multi-fluid flows [65, 69, 63, 62, 66], FOI [65, 68, 69, 62], aerodynamics of flapping wings [39, 48, 50], flows with solid surfaces in fast, linear or rotational relative motion [62, 66, 49, 47, 12, 61], compressible flows [62], shallow-water flows [66, 44], FPI [62, 66], and FSI [39, 28, 64, 71, 70, 34, 72, 73, 43, 76, 74, 52, 35, 77, 75, 54, 53, 36, 55, 45, 56, 46, 51, 57].

As mentioned in [72], moving the fluid mechanics mesh to track a fluid–solid interface enables us, at least for interfaces with reasonable geometric complexity, to control the mesh resolution near that interface and obtain accurate solutions in such critical flow regions. As also mentioned in [72], sometimes the geometric complexity of the interface may require a fluid mechanics mesh that is not affordable or not desirable or just not manageable in mesh moving, and this is one of the most common reasons given for favoring an interface-capturing method. This approach can be seen as a special case of interface representation techniques where the interface geometry is somehow represented over a nonmoving fluid mechanics mesh, the main point being that the fluid mechanics mesh does not move to track the interfaces. However, as pointed out in [67], a consequence of the mesh not moving to track the interface is

that for fluid–solid interfaces, independent of how accurately the interface geometry is represented, the resolution of the boundary layer will be limited by the resolution of the fluid mechanics mesh where the interface is.

For interfaces with reasonable geometric complexity, if a moving-mesh method can be used with a reasonable frequency of remeshing (see [70] for various remeshing options), its fluid mechanics accuracy near the interface will be superior to that of an nonmoving-mesh method. As pointed out in [72], "while it is understandable that fixed-mesh methods become more favored when the interface geometric complexity appears to be too high for a moving-mesh method, we need to remember that there is a difference between making the problem computable and obtaining good fluid mechanics accuracy near the interface." Therefore, as also pointed out in [72], "it is not difficult to imagine that if we lower our expectations of good fluid mechanics accuracy near the interfaces with high geometric complexity, we can find a number of ways to make the problem computable also with moving-mesh methods, and can still expect to obtain good accuracy near the interfaces with reasonable geometric complexity." Examples of that were given in [72].

A robust moving-mesh method with effective mesh update can handle FSI or other MBI problems even when the solid surfaces undergo large displacements (see, for example, FPI [62, 66] with the number of particles reaching 1,000 [66], parachute FSI [70, 72, 73, 75, 53, 55, 46, 51, 57], flapping-wing aerodynamics [48, 50], and wind-turbine rotor and tower aerodynamics [61]. It can handle FSI or other MBI problems also even when the solid surfaces are in near contact or create near TC, if the "nearness" is sufficiently "near" for the purpose of solving the problem. Examples of such problems are FPI with collision between the particles [62, 66], parachute-cluster FSI with contact between the parachutes of the cluster [53, 55, 46, 57], flapping-wing aerodynamics with the forewings and hindwings crossing each other very close [48, 50], and wind-turbine rotor and tower aerodynamics with the blades passing the tower

close [61].

As mentioned in [16], one of course recognizes that certain classes of interfaces (such as free-surface and two-fluid flows with splashing) might be too complex to deal with an interface-tracking technique and therefore, for all practical purposes, require an interface-capturing technique. The Mixed Interface-Tracking/Interface-Capturing Technique (MITICT) [66] was introduced in 2001 for computation of flow problems that involve both fluid–solid interfaces that can be accurately tracked with a moving-mesh method and fluid–fluid interfaces that are too complex or unsteady to be tracked. Those fluid–fluid interfaces are captured over the mesh tracking the fluid–solid interfaces. The MITICT was successfully tested in 2D computations with solid circles and free surfaces [2, 19] and in 3D computation of ship hydrodynamics [3].

In some MBI problems with contact between the solid surfaces, the "nearness" that can be modeled with a moving-mesh method without actually bringing the surfaces into contact might not be "near" enough for the purpose of solving the problem. Cardiovascular FSI with heart valves, where the flow has to be completely blocked at contact, is an example. The Fluid–Solid Interface-Tracking/Interface-Capturing Technique (FSITICT) [75] was motivated by such cardiovascular FSI problems. In the FSITICT, we track the interface we can with a moving mesh, and capture over that moving mesh the interfaces we cannot track, specifically the interfaces where we need to have an actual contact between the solid surfaces. A specific application of the FSITICT was presented in [80], where the ALE method is used for interface tracking, and a fully Eulerian approach for interface capturing, with some 2D benchmark problems as test computations. This specific application was extended in [80] to 2D FSI models with flapping and contact, where the fully Eulerian interface-capturing is complemented with mesh adaptivity.

There are many types of nonmoving-mesh methods that can compute MBI problems involving an actual contact between solid surfaces or other cases of TC. The

immersed-boundary methods, X-FEM, and particle methods are the typical examples. Some of those methods give up on the accurate representation of the interface, and most give up on the consistent representation of the interface motion. The DSD/SST formulation does not need to give up on either, even where we have an actual contact or some other TC, provided that we can update the mesh even there. Using an ST mesh that is unstructured both in space and time, as proposed for contact problems in [66], would give us such a mesh update option. However, that would require a fully unstructured 4D mesh generation, and that is not easy in computing real-world problems.

We want to address the computational challenges associated with contact between moving solid surfaces and other cases of TC, including those in cardiovascular FSI, parachute FSI, and flapping-wing aerodynamics, with the primary target in cardiovascular FSI being heart valves. For this purpose, we introduce in this paper an ST interface-tracking method that can deal with TC. It is a new version of the DSD/SST method, and we call it ST-TC. It is a practical alternative to using unstructured ST meshes, but without giving up on the accurate representation of the interface or the consistent representation of the interface motion, even where there is an actual contact between solid surfaces or other TC. The ST-TC method is based on special mesh generation and update, and a master–slave system that maintains the connectivity of the "parent" mesh when there is a TC.

In Chapter 2, we provide, with two hypothetical cases, a context for TC and explain the master–slave system and its design. In Chapter 3, we provide two conceptual examples that help us explain the mesh update process. Three numerical examples are presented, a 2D flapping pair in Chapter 4, a 3D Micro air vehicle (MAV) in Chapter 5, and an aortic valve model with coronary arteries in Chapter 6. The concluding remarks are in Chapter 7.

Chapter 2

Topology change

The material in this chapter is from [59]. We consider two hypothetical cases of two bars to provide a context for TC. In the first case, shown in Figure 2.1, the bars

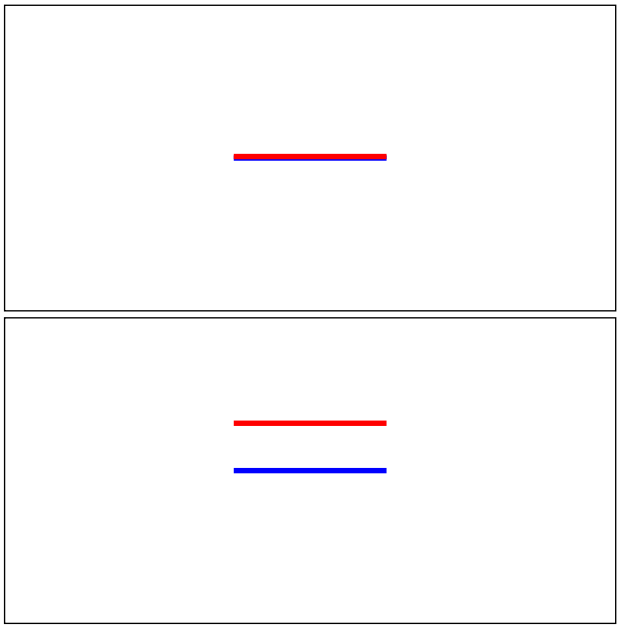

Figure 2.1: Hypothetical case of two bars that are initially coinciding, with one hole in the fluid mechanics domain (top). Then the red bar starts moving upward, creating a second hole in the domain (bottom).

are initially coinciding, with just one hole in the fluid mechanics domain. Then the red bar starts moving upward, creating a second hole in the domain. In the second case, shown in Figure 2.2, the bars are initially aligned with connected ends, again

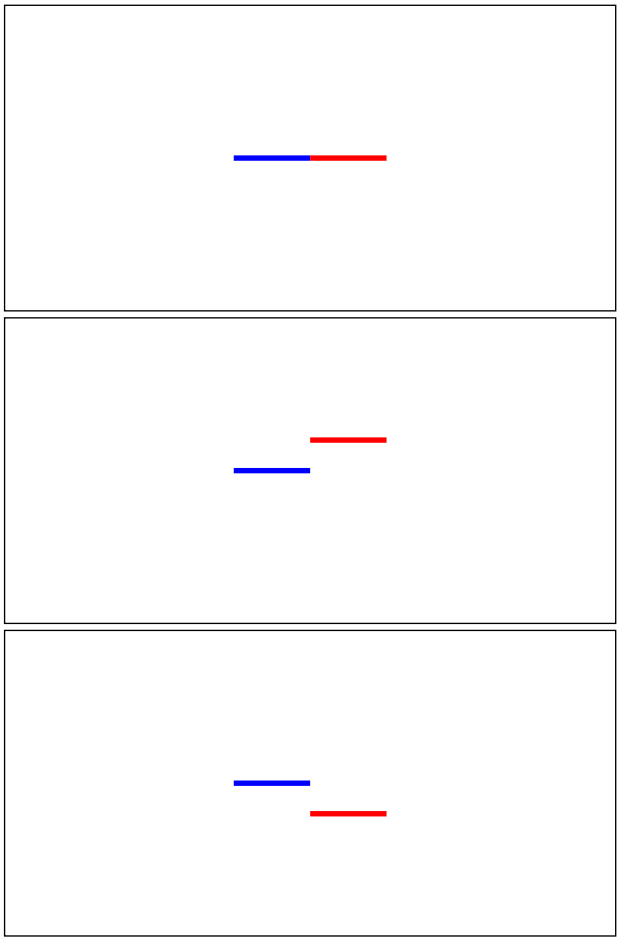

Figure 2.2: Hypothetical case of two bars that are are initially aligned with connected ends, with one hole in the domain (top). Then the red bar starts a flapping motion, up (middle) and down (bottom), creating a second hole in the domain, except when their ends become connected periodically during the flapping motion.

with a single hole in the domain. Then the red bar starts a flapping motion, up and down, creating a second hole in the domain, except when their ends become connected periodically during the flapping motion. When the red bar is in the upper position, the part of the domain below it is connected to the part of the domain above the blue bar. When the red bar is in the lower position, the part of the domain above it is connected to the part of the domain below the blue bar.

These two cases are representatives of the typical TC challenges we expect to see in the classes of MBI problems we are targeting. Especially the first case is really not possible to treat in a consistent way without using an ST method.

2.1 Master–slave system

We propose a very simple technique in the ST context. Having a constraint between nodes in a finite element formulation is quite common. These constraints reduce the number of unknowns, but in our implementation we delay that unknown elimination until the iterative solution of the linear systems encountered at nonlinear iterations of a time step. The iterative solution of the linear systems is performed with reduced number of unknowns. The technique is easy to manage in a parallel-computing environment, especially if the preconditioner is simple enough. Typically we assign a master node to each slave node, and we use only the unknowns of the master nodes in iterative solution of the linear systems.

We can use different master–slave relationships at different time levels. This is a practical alternative to, but less general than, using ST meshes that are unstructured in time. Still, we can use this concept to deal with the TC cases considered above, and the important point is that the connectivity of the "parent" mesh does not change. Consequently, the distribution model in the parallel-computing environment does not change during the computations with moving meshes.

With this technique, we need to implement one more functionality. We exclude certain elements from the integration of the finite element formulation. The exclusion principles are given below.

- Exclude all spatial elements with zero volume from the spatial integration.
- Exclude all ST elements with zero ST volume from the ST integration.
- We assume that checking if an ST element has zero ST volume is equivalent to checking if all the spatial elements associated with that ST element have zero volume. Therefore, for this purpose, we check the spatial-element volumes.
- To identify the spatial elements with zero volume, which should have zero Jaco-

bian at all the integration points, instead of evaluating the Jacobians, we make the determination for a given spatial element from the master–slave relationship of its nodes.

2.2 Design of the master–slave system

The data we need to provide to the solver is quite simple. It is just the master–slave relationship at each time level. However there are some restrictions, and here we explain the three that we want to emphasize.

The first restriction is that we cannot have a node which is not part of any active (nonzero volume) spatial element. This is because the values at such nodes would no longer be in our equation system, and therefore would become undefined. If because of another TC such a node comes back to the equation system later as part of an active element, it would add an undefined component to the equation system.

The second restriction is that when we construct the ST elements, we have to have matching lateral element-boundary faces between the active adjacent ST elements. This condition cannot be checked on the spatial mesh. Therefore we need to check it on the ST mesh.

The third one is related to implementation. The master–slave relationship also extends to cases when we have boundary conditions on the master and slave nodes. In other words, the conditions at the master node also apply to the slave nodes.

Chapter 3

Conceptual examples

The material in this chapter is from [59].

3.1 Contraction and expansion

This is related to the first one of the two cases of TC described in Chapter 2. Contraction and expansion are basically the same, except having different directions in time progression. Figure 3.1 shows a contraction example. The spatial element with

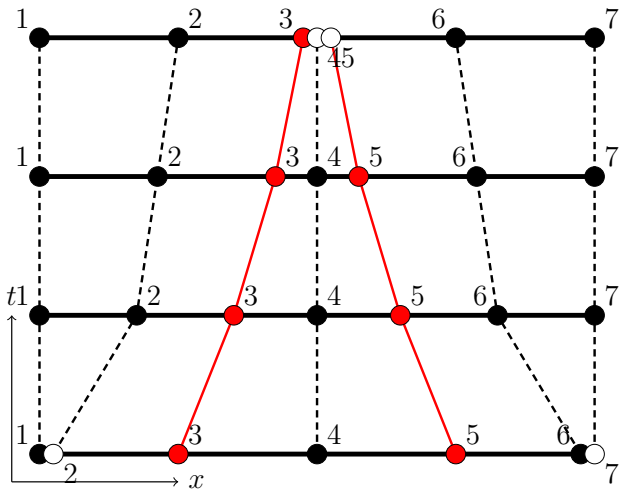

Figure 3.1: Contraction. The red nodes, 3 and 5, are on the contraction interface and are contacting. The white nodes are the slaves. They are in the same position as their masters, but for visualization purposes we slightly shift their positions in the figure. The numbers indicate the node numbers on the parent mesh.

nodes 1 and 2, for example, has zero volume at the first time level. However, it has nonzero volume at the second time level, and therefore the corresponding ST element has nonzero volume.

3.2 Flapping

This is related to the second one of the two cases of TC described in Chapter 2. Figure 3.2 shows the red and blue bars at three instants in time as the red bar crosses the blue bar. Figure 3.3 shows, for the flapping motion, the ST trajectories of the

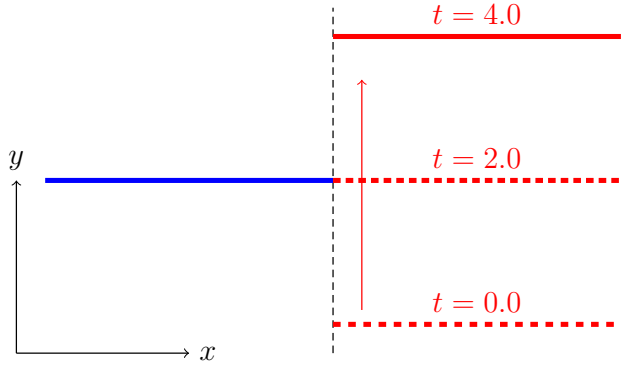

Figure 3.2: Flapping. Red and blue bars at different instants in time as the red bar crosses the blue bar.

neighboring ends of the blue and red bars. Figure 3.4 shows the ST element edges

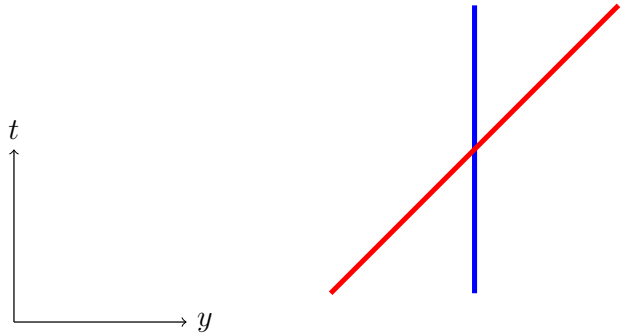

Figure 3.3: Flapping. The ST trajectories of the neighboring ends of the blue and red bars.

for the line separating the two sides of the domain containing the blue and red bars (shown as the vertical dashed line in Figure 3.2).

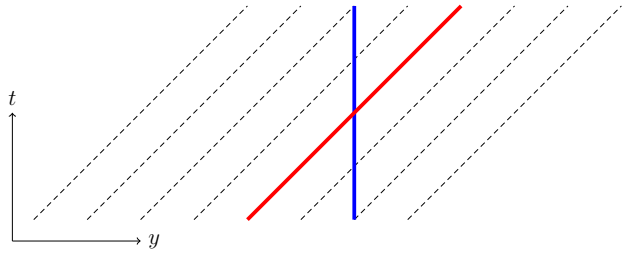

Figure 3.4: Flapping. The ST element edges for the vertical dashed line in Figure 3.2.

For each side of the domain, the spatial node motions along the ST element edges have to be designed in a fashion that does not lead to mesh entanglement. Figure 3.5 shows the master–slave relationship for the blue-bar side of the domain, and Figure 3.6 the red-bar side. In addition, those two sides are in a master–slave relationship along

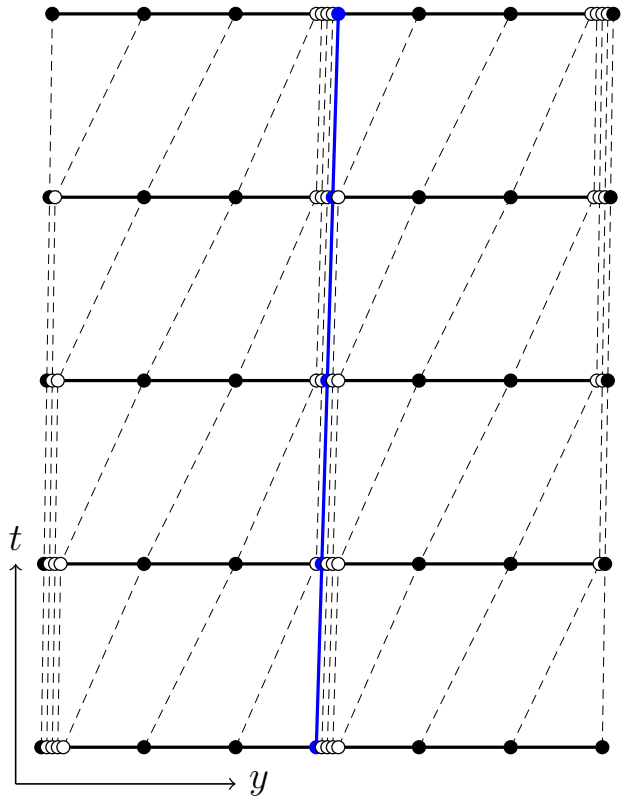

Figure 3.5: Flapping. Blue-bar side of the ST boundary between the two sides.

the vertical dashed line in Figure 3.2.

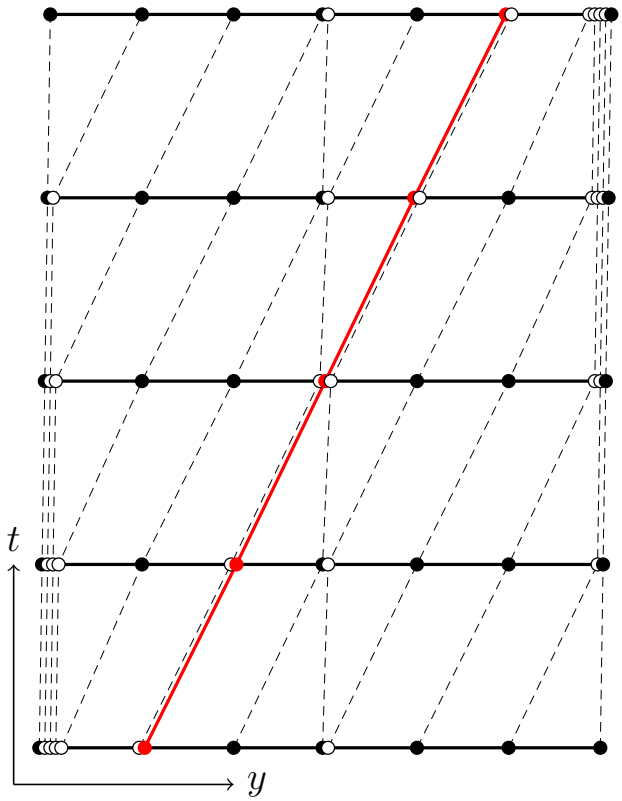

Figure 3.6: Flapping. Red-bar side of the ST boundary between the two sides.

Chapter 4

A pair of symmetrically-flapping surfaces ("Flapping pair")

The material in this chapter is from [59]. We use the DSD/SST method in the computations. The stabilization parameter τ_{SUPS} comes from the τ_{SUPG} definition in [67], specifically the definition given by Eqs. (107)–(109) in [67], which can also be found as the definition given by Eqs. (7)–(9) in [70], with h_{RGN} (= h_{RGNT}) and ν_{LSIC} (= $\nu_{\text{LSIC-HRGN}}$) given by Eqs. (15) and (19) in [61]. In solving the linear equation systems encountered at every nonlinear iteration of a time step, the GMRES search technique [42] is used with diagonal preconditioner.

4.1 Geometry, motion modeling and computational conditions

The pair of surfaces, with zero thickness, undergo a prescribed sinusoidal flapping, as shown in Figures 4.1 and 4.2, with a period of $T = 18.0$ s. The projected width of the flapping surfaces along the horizontal axis is 1.0 m. The density and kinematic viscosity are 1.0 kg/m^3 and 1.6667×10^{-5} m^2/s. The inflow velocity is 0.1 m/s. The

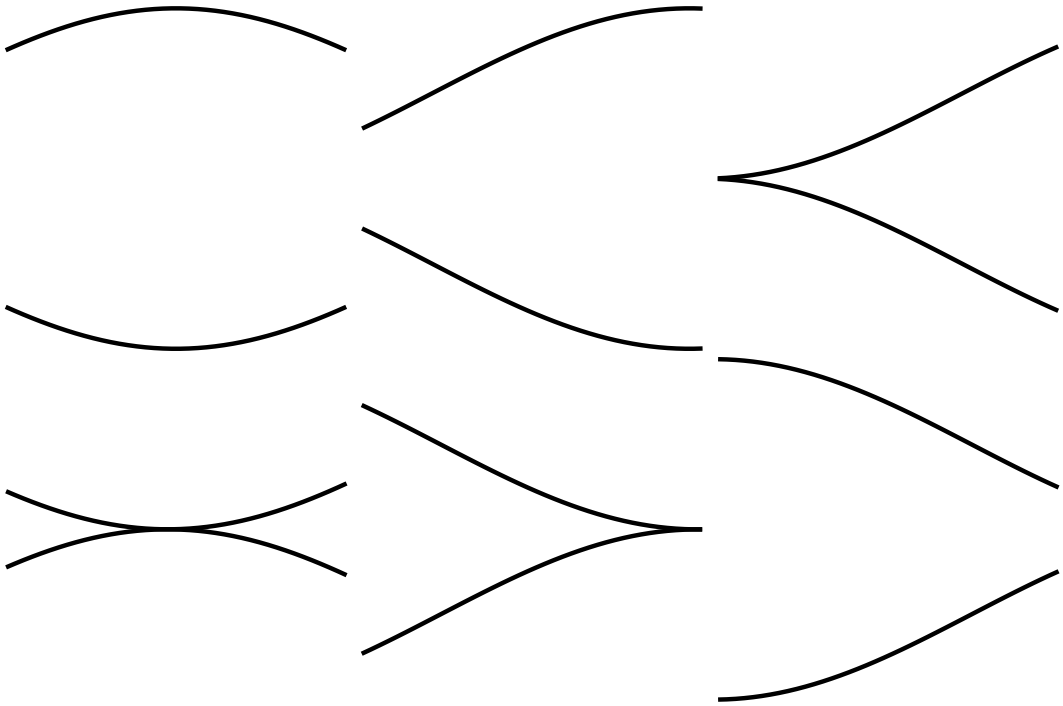

Figure 4.1: Flapping pair. Surface positions at $t = 0$ (also 18), 3, 6, 9, 12 and 15 s.

Reynolds number based on these length, viscosity and velocity scales is 6,000.

The dimensions of the computational domain, in m, are 40×20, and the distance between the inflow boundary and the leading edge is 15 m. The boundary conditions are no-slip on the flapping surfaces, uniform horizontal velocity at the inflow, zero-stress at the outflow, and slip at the upper and lower boundaries. We tested three different meshes to see the influence of increased refinement in space and time. The meshes have a structured inner zone and an unstructured outer zone, made of 4-node quadrilateral and 3-node triangular elements, respectively. Table 4.1 shows, for each mesh, the number nodes along each flapping surface, number of nodes in the inner zone, total number of nodes and elements, and the number of time steps per flapping cycle. Figures 4.3–4.6 show Mesh 1 and the inner zones for all three meshes.

During the flapping motion, only the inner mesh moves, with a special, algebraic mesh moving technique. Figures 4.7–4.9 show, for Mesh 1, the inner mesh at different instants during the flapping cycle.

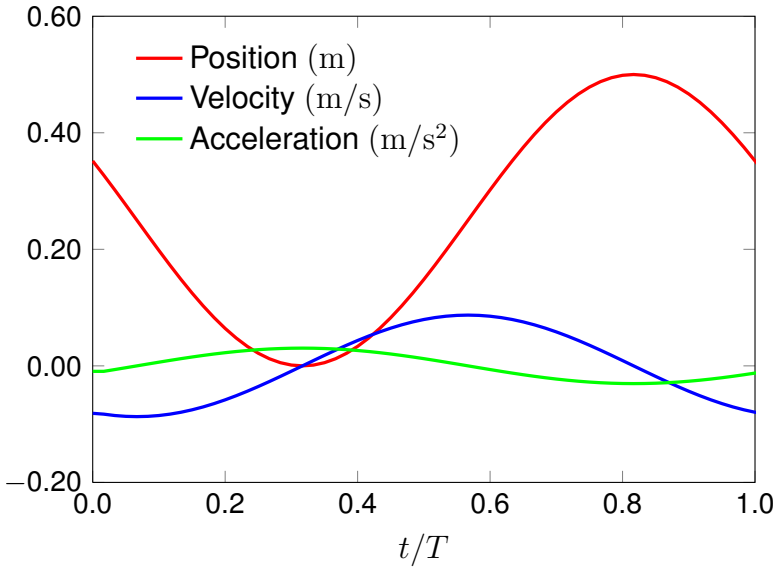

Figure 4.2: Flapping pair. Leading-edge position, velocity and acceleration for the upper surface.

Mesh	Surface Nodes	Inner Nodes	Total Nodes	Elements	Time Steps
1	40	28,379	37,911	47,628	60
2	80	32,439	42,432	52,590	120
3	120	36,499	47,156	57,958	180

Table 4.1: Flapping pair. Mesh data and the number of time steps per flapping cycle.

4.2 Computations

We use the SUPS version of the DSD/SST method (see [54, 56, 16] for the terminology). Prior to the flapping motion, we compute 2,500 time steps to develop the flow field. The time step size is 1.0 s, with 3 nonlinear iterations per time step, and the corresponding number of GMRES iterations are 50, 100 and 150. The flapping cycles are computed with 5 nonlinear iterations per time step, and the corresponding number of GMRES iterations are 150, 250, 450, 650 and 900. We note that since it is not easy to show that the solution in this test problem is accurate in terms of fluid physics, we decided to use large numbers of GMRES iterations. We did not explore reducing the number of GMRES iterations, because the 2D computations are not

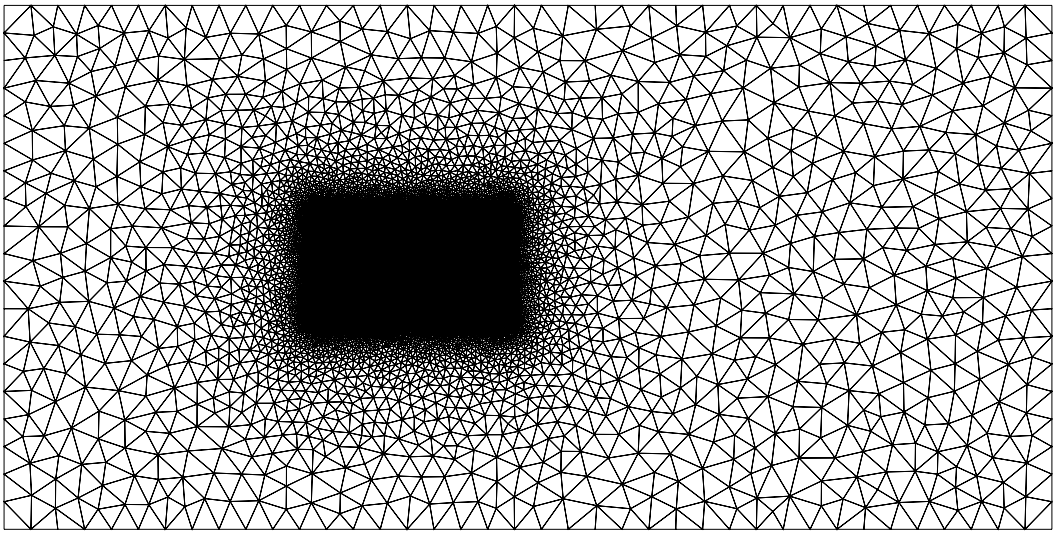

Figure 4.3: Flapping pair. Mesh 1.

costly. We compute the problem for two complete cycles and display the results for the second flapping cycle. Figures 4.10–4.12 show, for the three meshes, the lift and drag experienced by the flapping surfaces.

4.3 Results

Figures 4.13–4.15 show, for Mesh 1, the velocity magnitude at different instants during the flapping cycle.

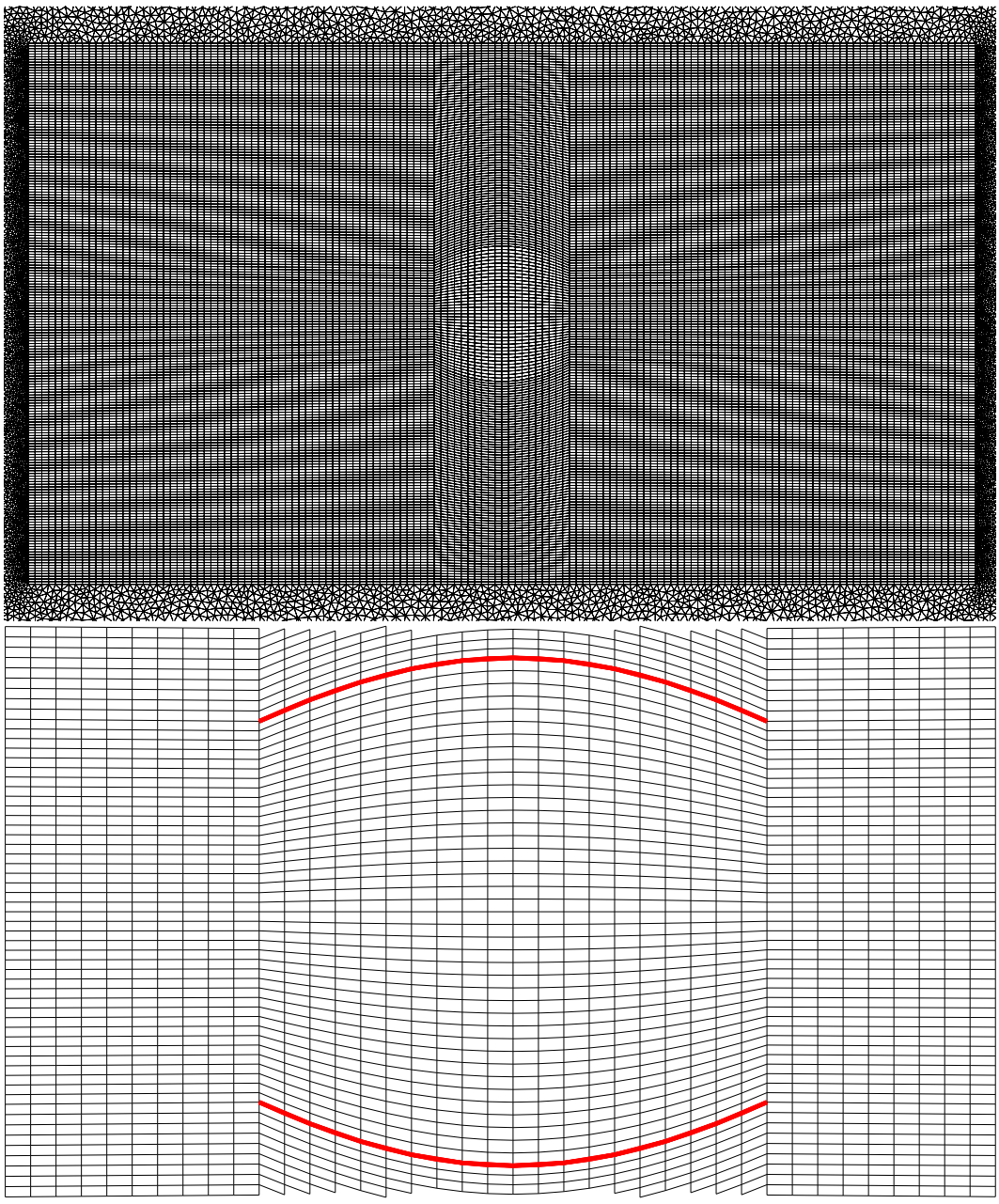

Figure 4.4: Flapping pair. Mesh 1. Inner zones.

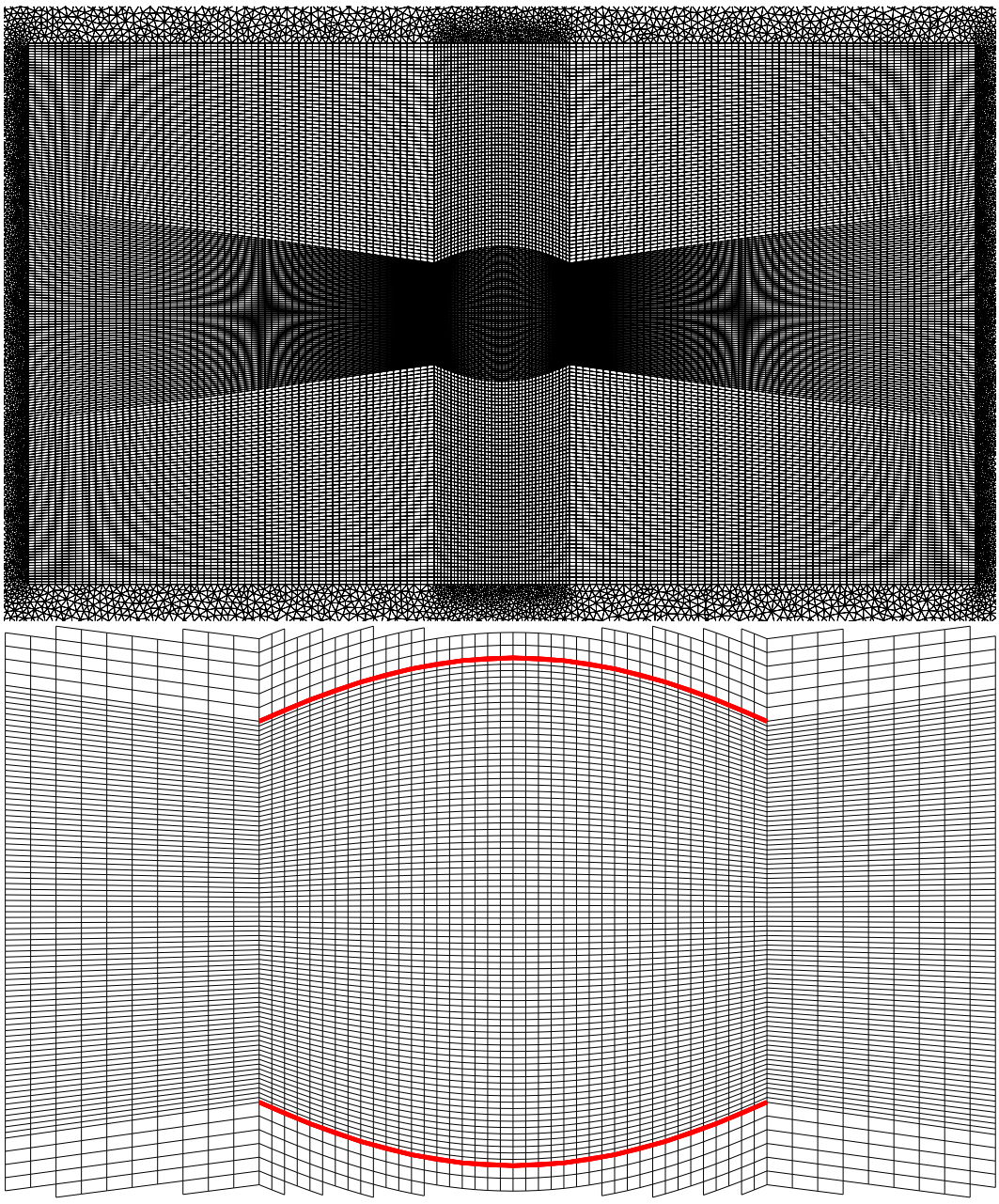

Figure 4.5: Flapping pair. Mesh 2. Inner zones.

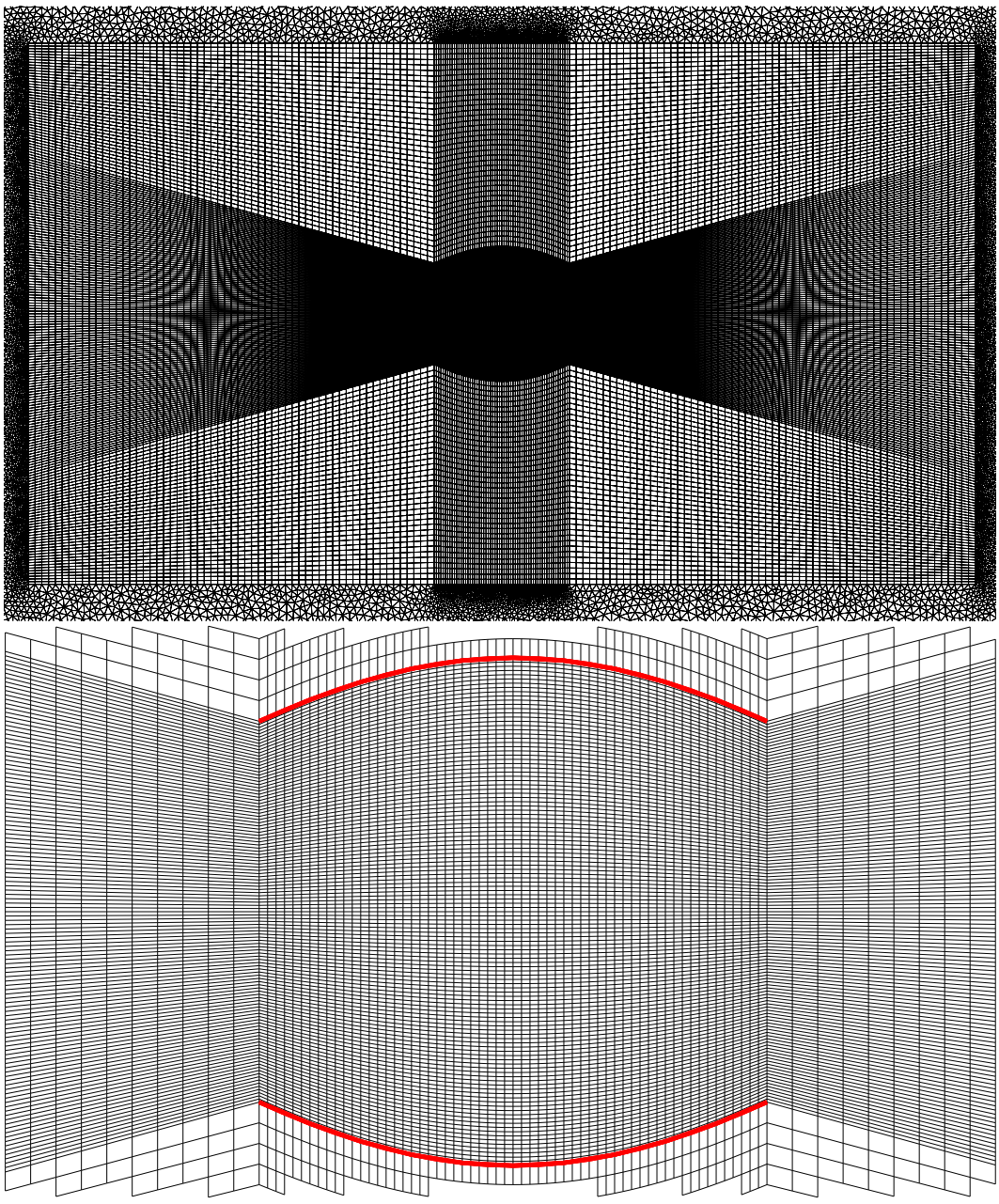

Figure 4.6: Flapping pair. Mesh 3. Inner zones.

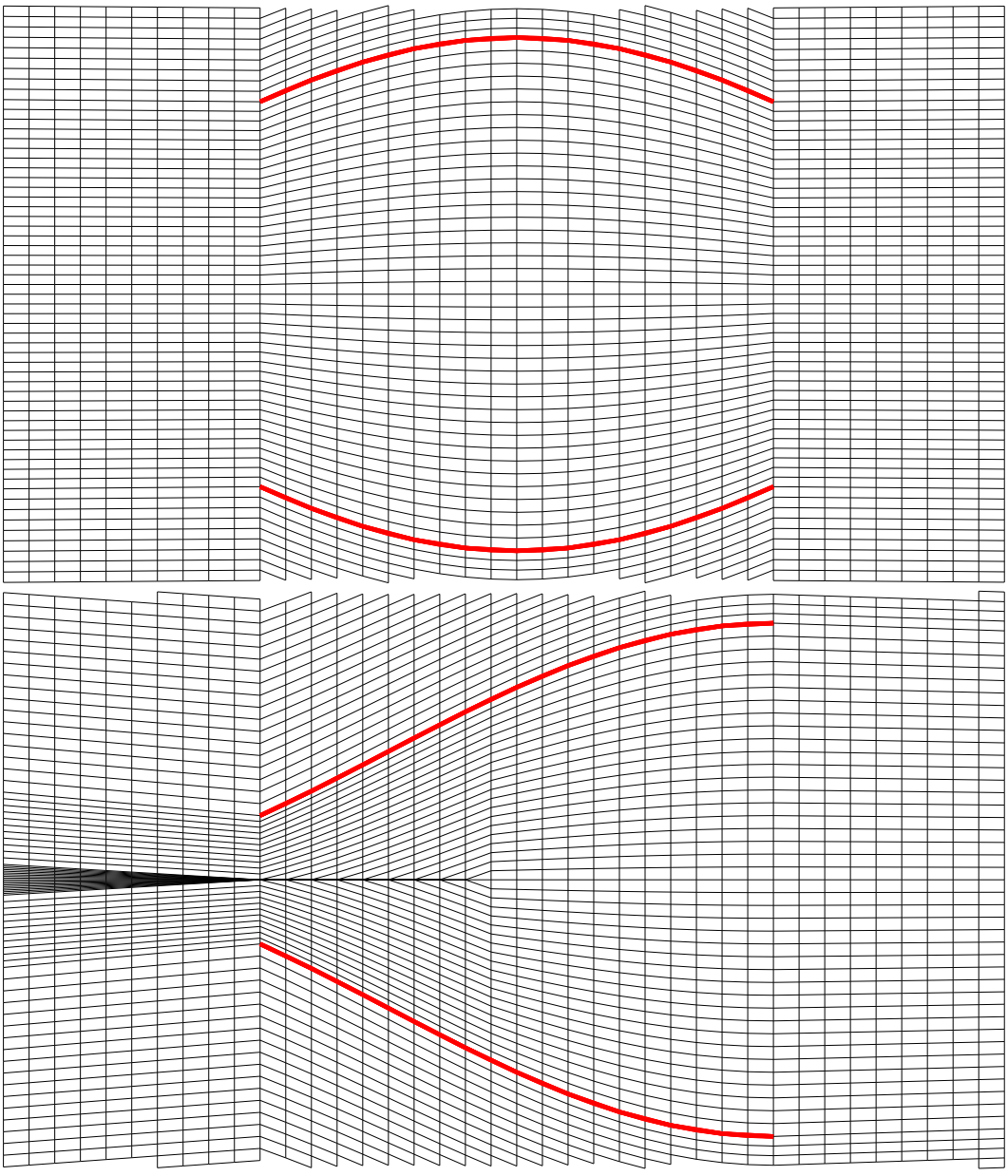

Figure 4.7: Flapping pair. Mesh 1 at $t = 0$ (also 18) and 3 s.

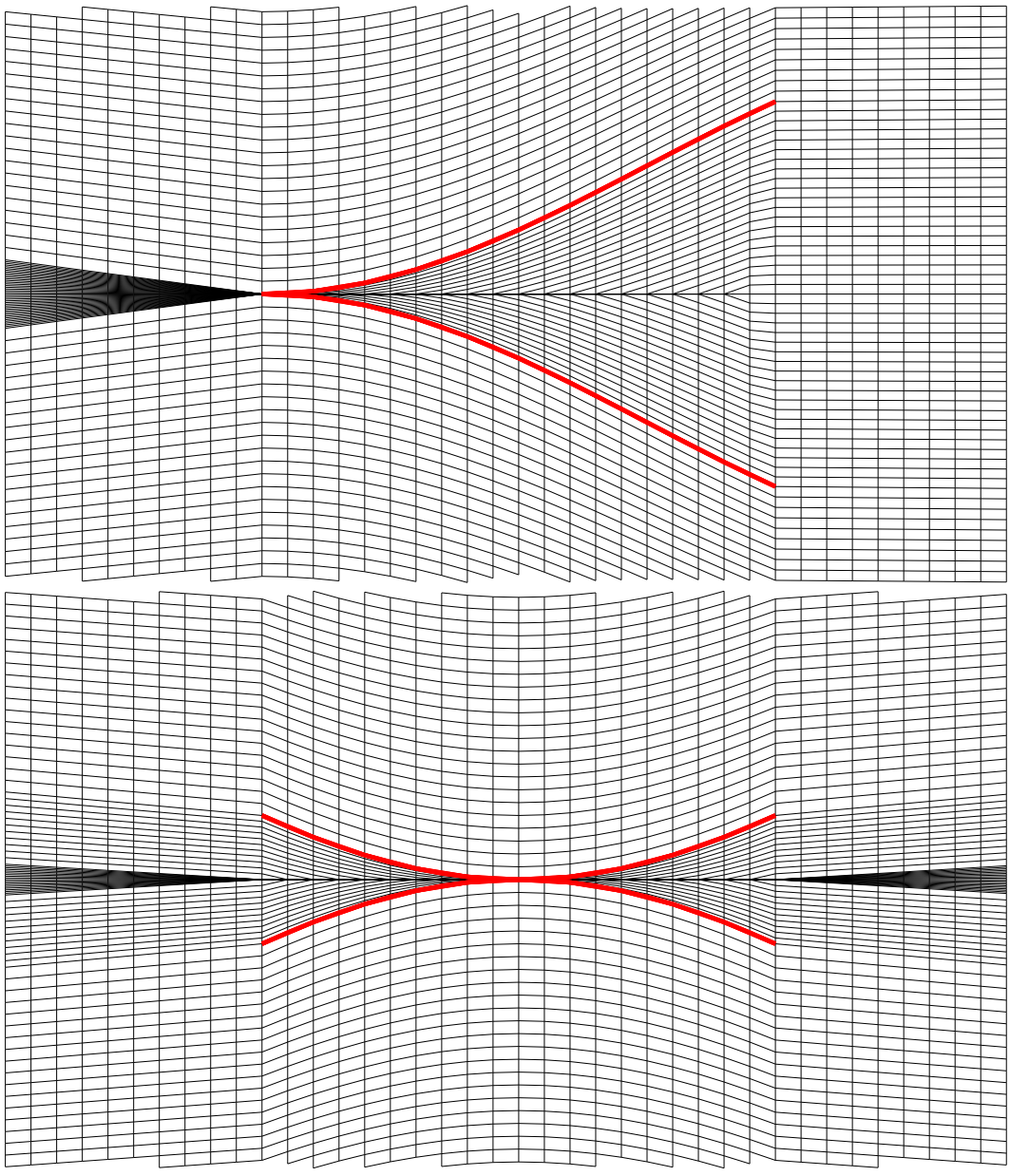

Figure 4.8: Flapping pair. Mesh 1 at $t = 6$ and 9 s.

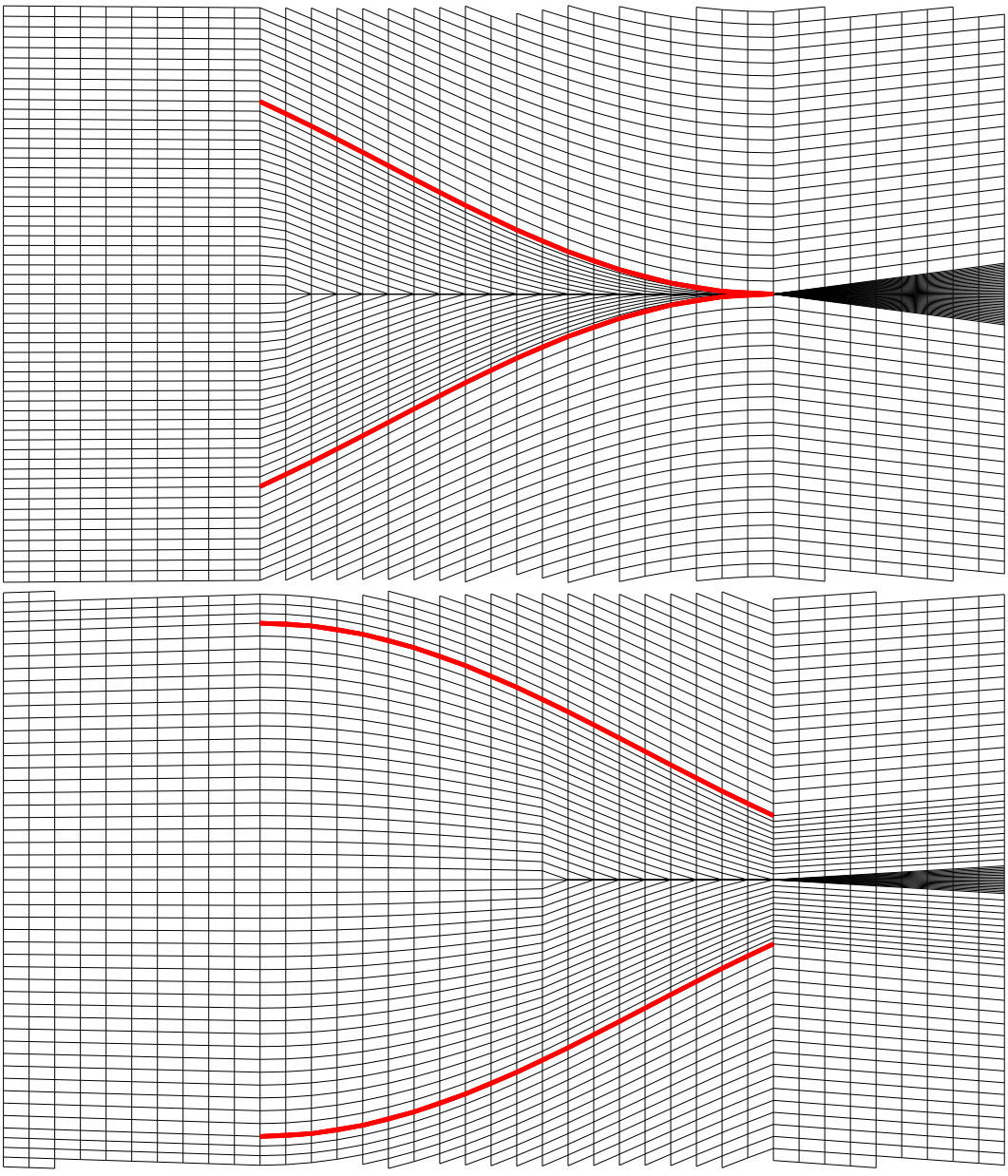

Figure 4.9: Flapping pair. Mesh 1 at $t = 12$ and 15 s.

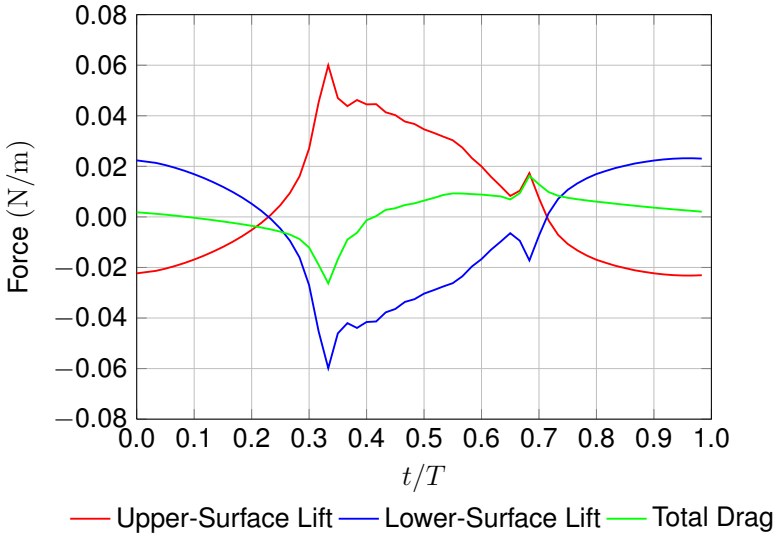

Figure 4.10: Flapping pair. Lift and drag for Mesh 1.

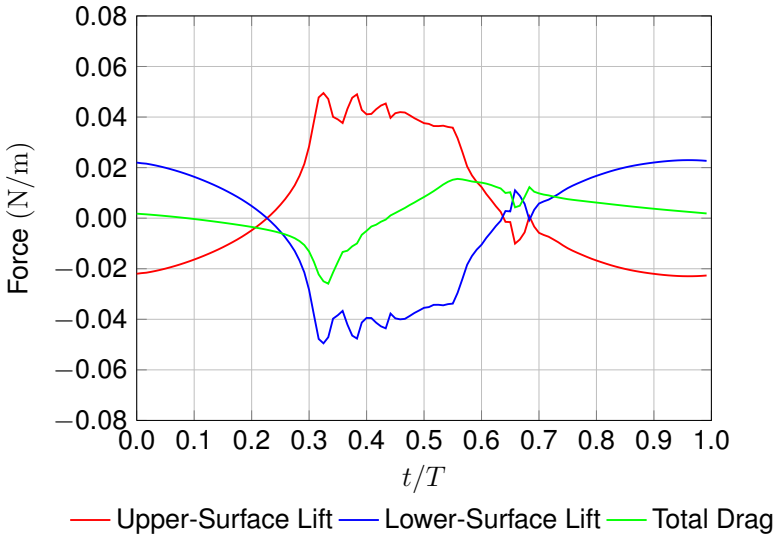

Figure 4.11: Flapping pair. Lift and drag for Mesh 2.

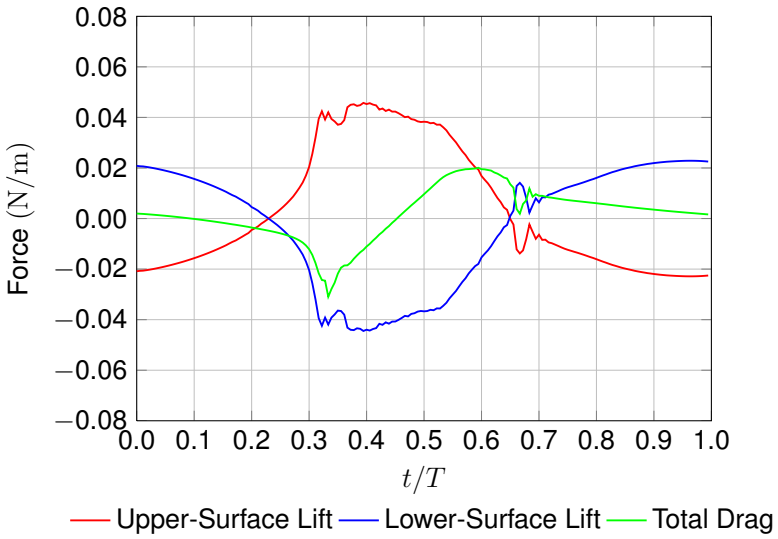

Figure 4.12: Flapping pair. Lift and for Mesh 3.

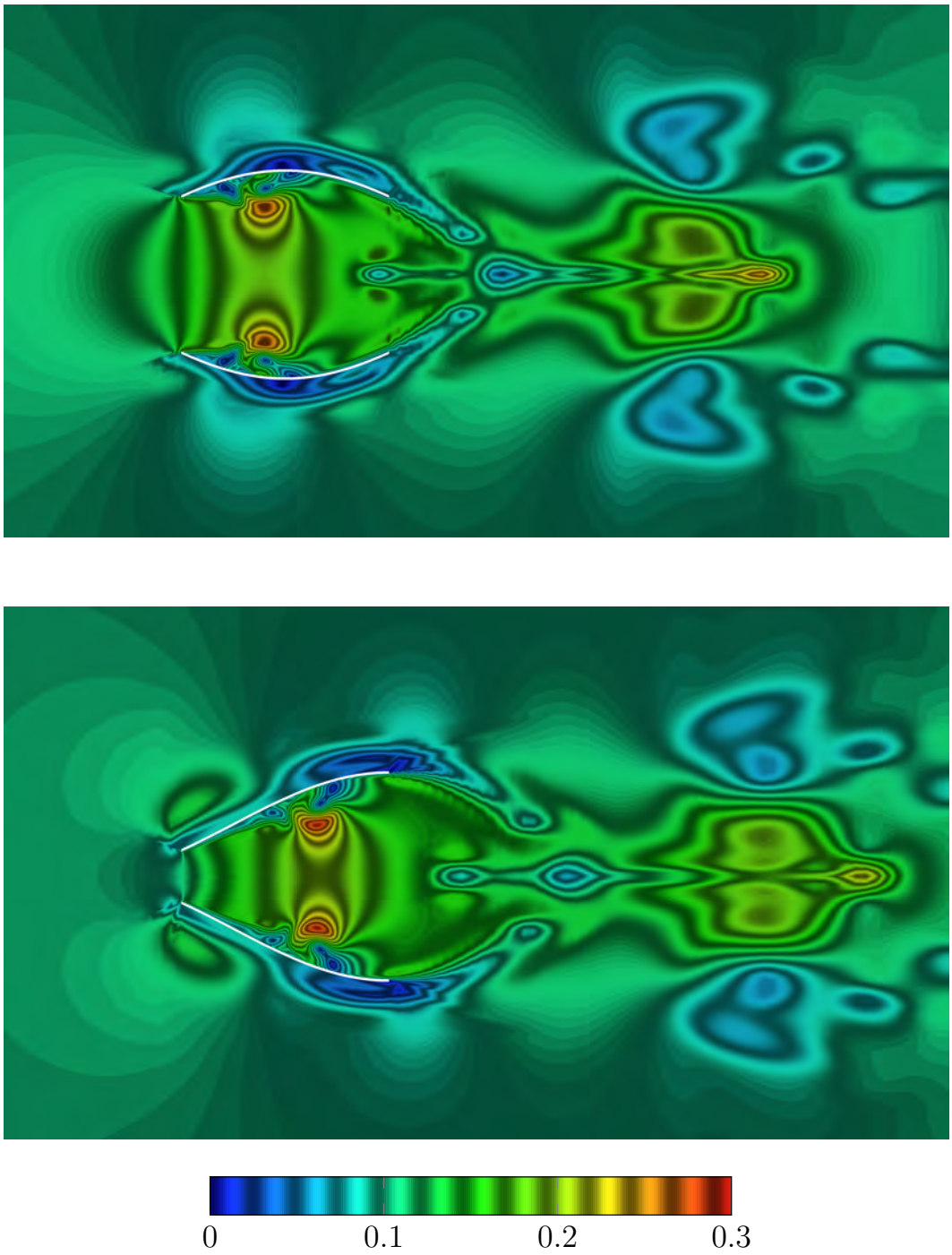

Figure 4.13: Flapping pair. Velocity magnitude (in m/s) for Mesh 1 at $t = 0$ and 3 s.

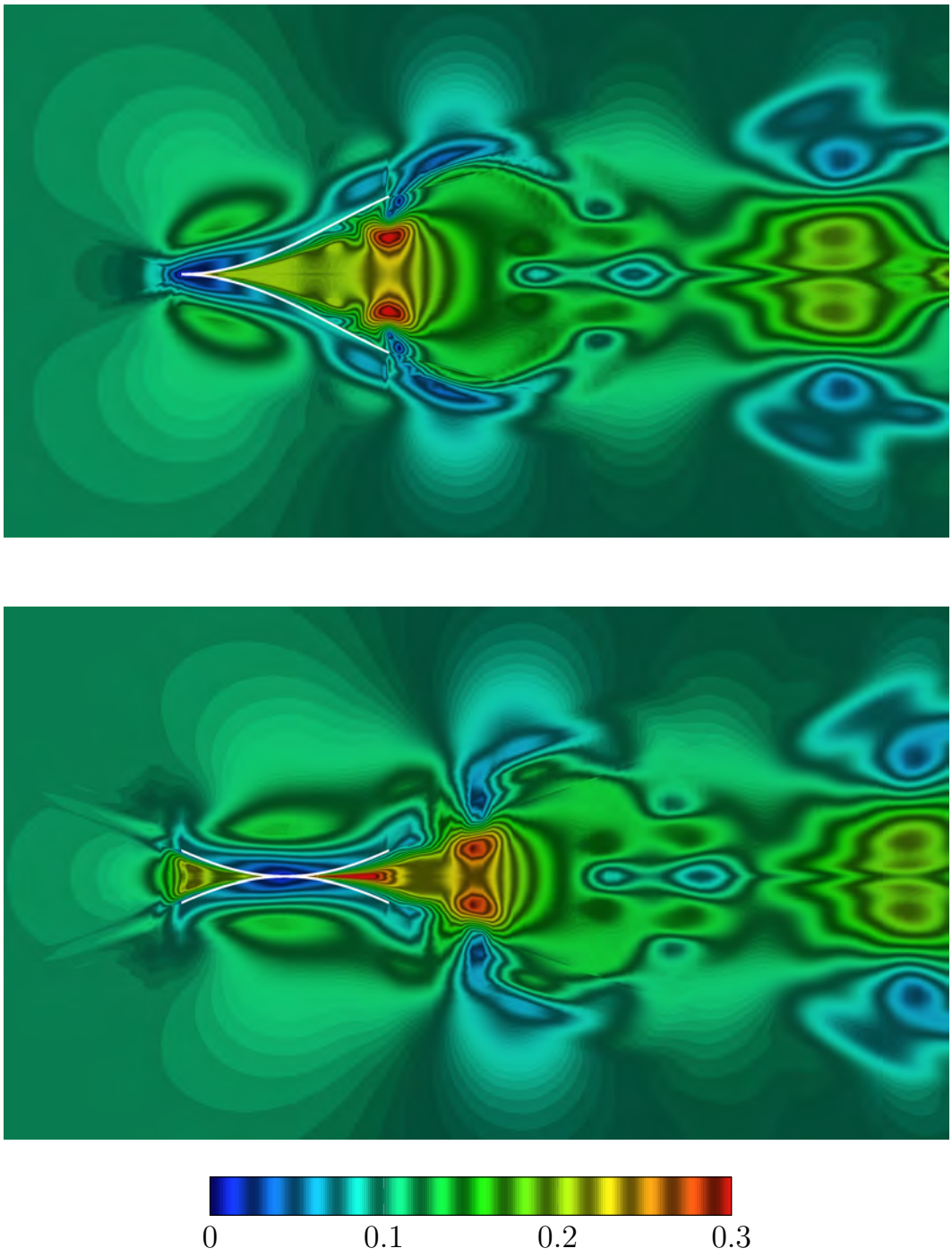

Figure 4.14: Flapping pair. Velocity magnitude (in m/s) for Mesh 1 at $t = 6$ and 9 s.

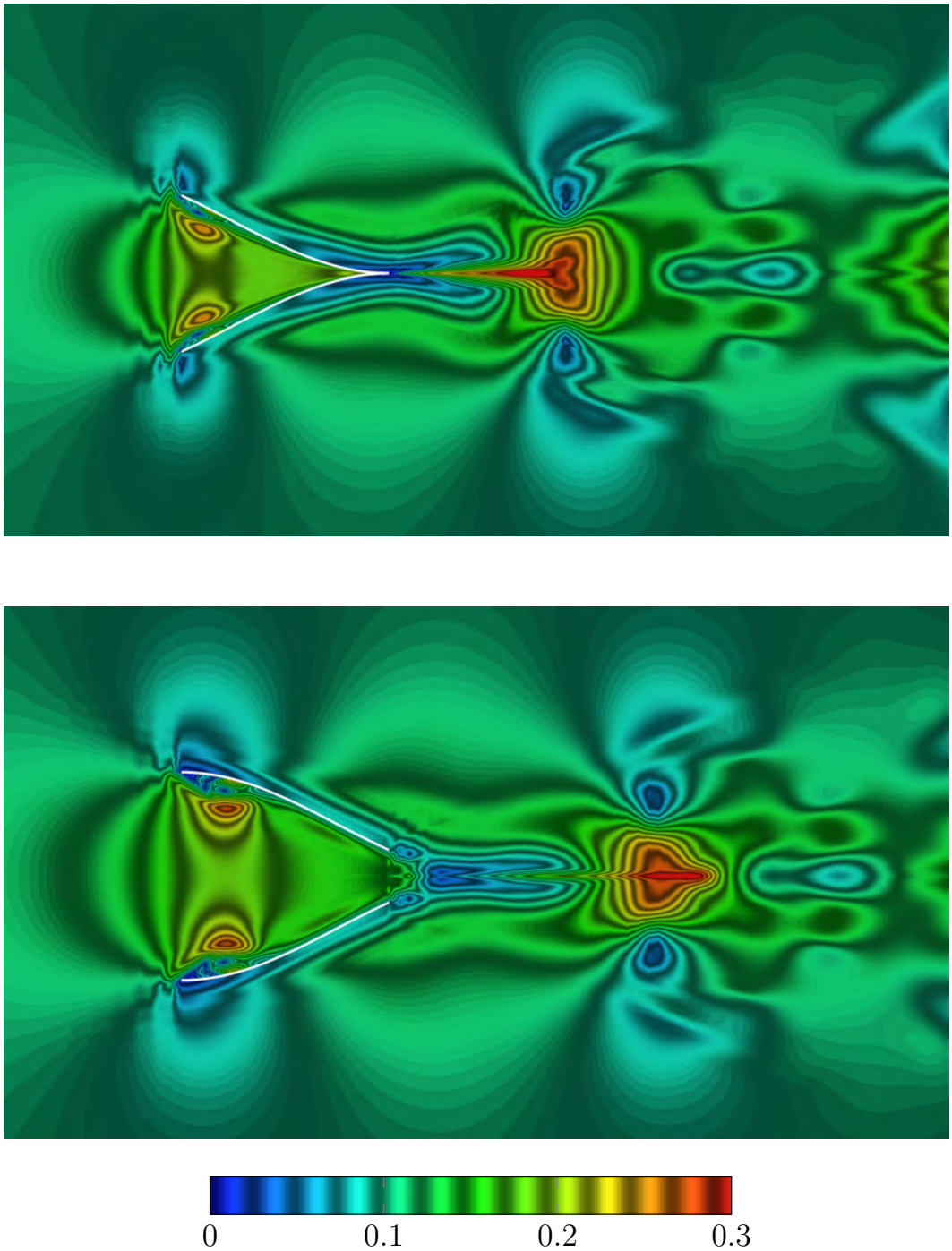

Figure 4.15: Flapping pair. Velocity magnitude (in m/s) for Mesh 1 at $t = 12$ and 15 s.

Chapter 5

Dragonfly MAV

The material in this chapter is from [60].

5.1 Geometry and flapping-motion modeling

The design of the wings is similar to the design in a toy MAV [1]. The body is the same as the MAV body in [50].

The span of the single wing is 46.7 mm and the minimum, maximum, and average chord lengths are 16.2, 19.2, and 17.6 mm, respectively (see Figure 5.1). The wings have zero thickness and undergo prescribed flapping, as shown in Figures 5.2 and 5.3, with a period of $T = 0.0365$ s. Figure 5.4 shows the position of the leading-edge contact point over time. The position is measured from the body.

The density and kinematic viscosity are 1.225 kg/m^3 and 1.461×10^{-5} m^2/s. The free-stream velocity is 4.5 m/s. The Reynolds number based on average chord length and free-stream velocity is 5,423. Three cases are computed, with the angle of attack $\alpha = 0^\circ,\ 5^\circ$, and 10°.

The dimensions of the computational domain, in spans of a single wing, are $30\times20\times20$, and the distance between the inflow boundary and the leading edge is 10 (see Figure 5.5). The boundary conditions are no-slip on the wings and body

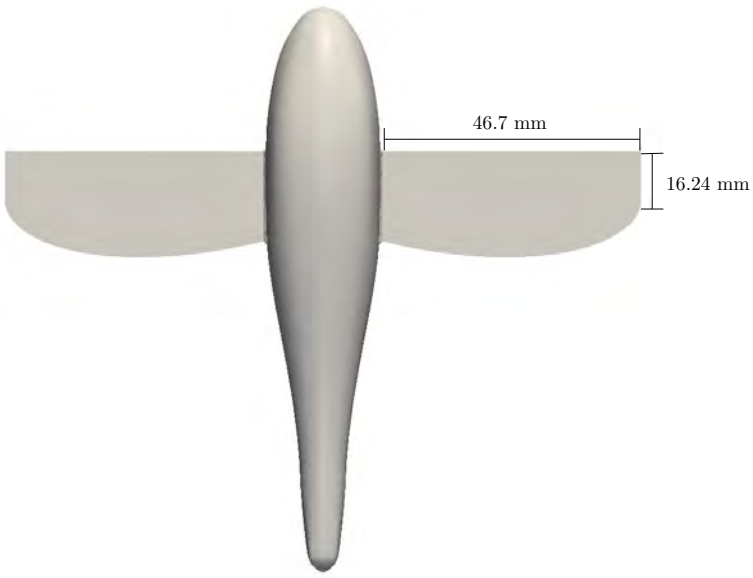

Figure 5.1: Dragonfly MAV. Wing dimensions.

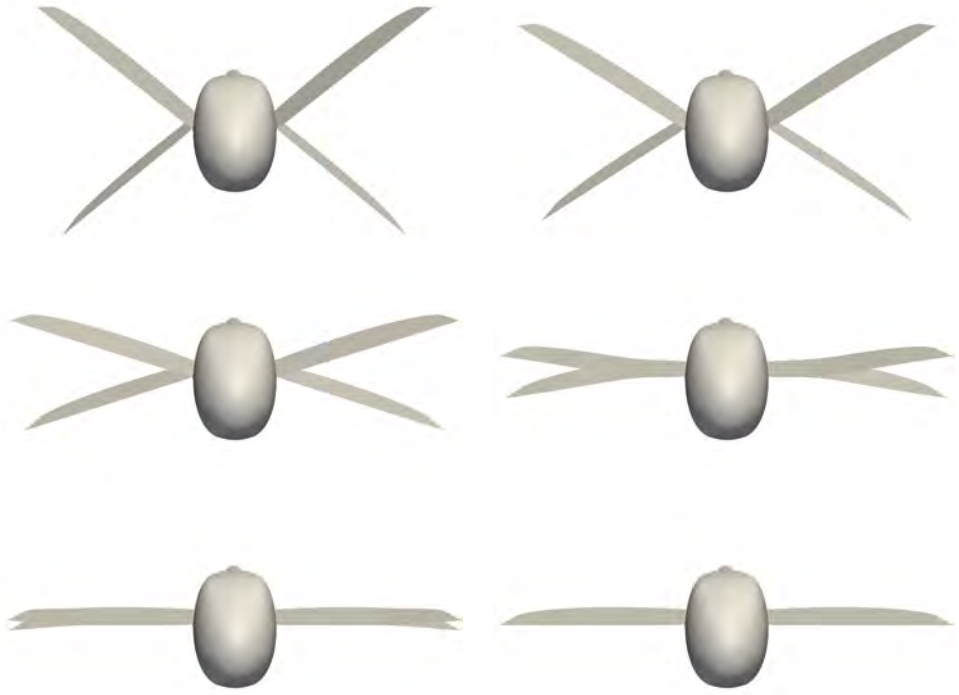

Figure 5.2: Dragonfly MAV. Wing configurations at $t/T = 0.0$, 0.1, 0.2, 0.3, 0.4, and 0.5 (left to right and then top to bottom).

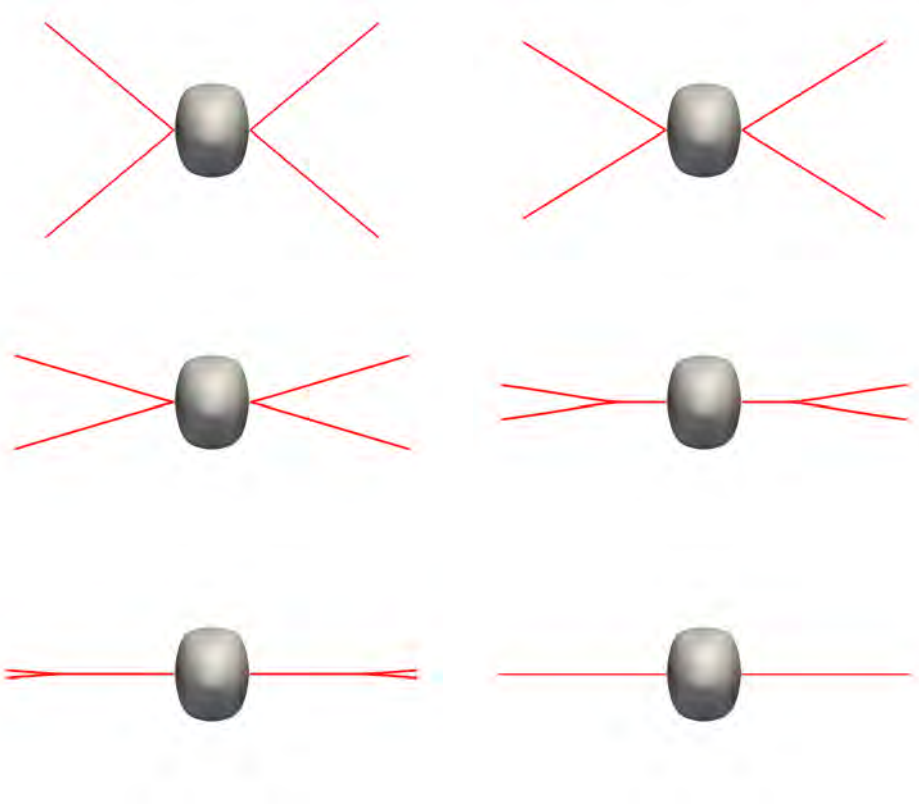

Figure 5.3: Dragonfly MAV. Wing leading edges at the same instants as in Figure 5.2.

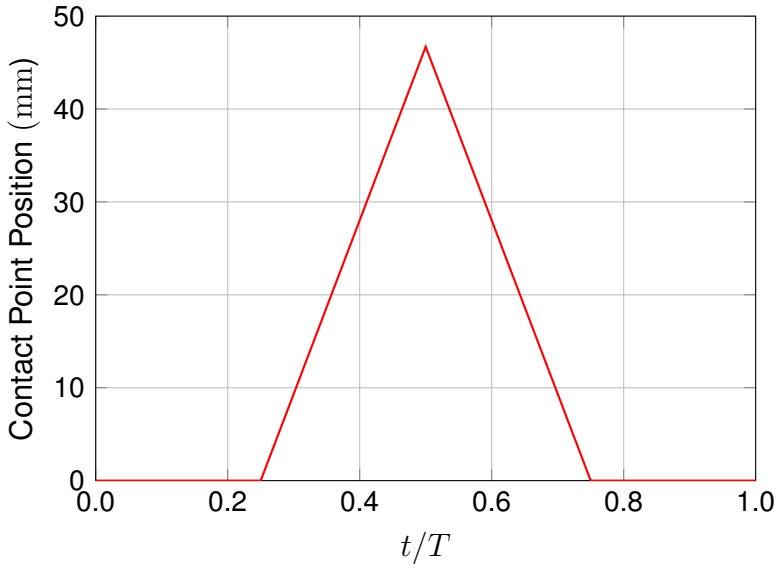

Figure 5.4: Dragonfly MAV. Contact point position along the leading edge over a flapping cycle.

of the MAV, uniform horizontal velocity at the inflow boundary, zero-stress at the outflow boundary, and slip at the upper, lower, and side boundaries.

The meshes have structured, inner zones around the wings and an unstructured, outer zone. Both the structured and unstructured zones consist of tetrahedral elements. For each mesh, Table 5.1 shows the number of nodes and elements. Figure 5.6 shows top view of the wing and body surface meshes. During the flapping motion, only the mesh in the inner zones move, with a special, algebraic mesh moving technique.

The structured, inner zones consist of four parts corresponding to each wing. Those parts each have $3\times2\times2$ structured zones. Each zone has $20\times20\times20$ hexahedral clusters made of 6 tetrahedral elements. Figures 5.7 and 5.8 show, for $\alpha = 0°$, the mesh at six equally-spaced instants in time while the wings are closing. The zones between the upper and lower wings collapse when the wings close, and the nodes in the neighboring zones also collapse accordingly. We note that a wing has split nodes except on the leading and trailing edges. However, when the wings are closed, the nodes on the upper surface of the upper wings and the lower surface of the lower wings become masters. When the wings are partially closed, at the contact point, the nodes on the lower surface of the upper wing are also masters while the nodes on the upper surface of the lower wings are slaves.

5.2 Computational conditions

We use the DSD/SST-SUPS and DSD/SST-VMST (convective) techniques for the first two and last two nonlinear iterations of each time step. The stabilization parameter τ_{SUPS} comes from the τ_{SUPG} definition in [67], specifically the definition given by Eqs. (107)–(109) in [67], which can also be found as the definition given by Eqs. (7)–(9) in [70], with ν_{LSIC} from Eq. (17) in [70]. The time-step size is 4.51×10^{-4} s.

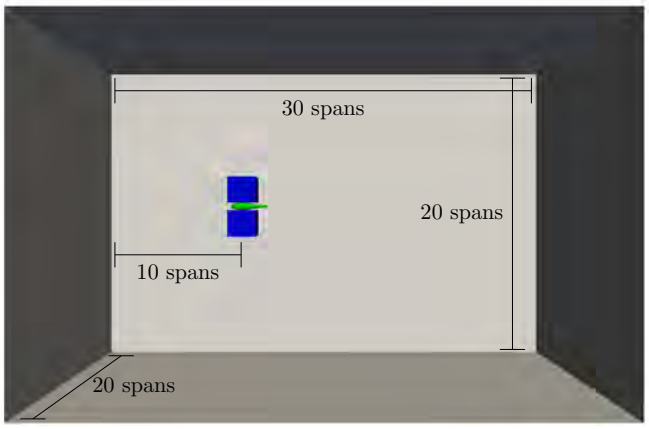

Figure 5.5: Dragonfly MAV. Computational domain and mesh setup. Outer boundaries (gray), boundaries of the inner, structured meshes (blue) and body (green).

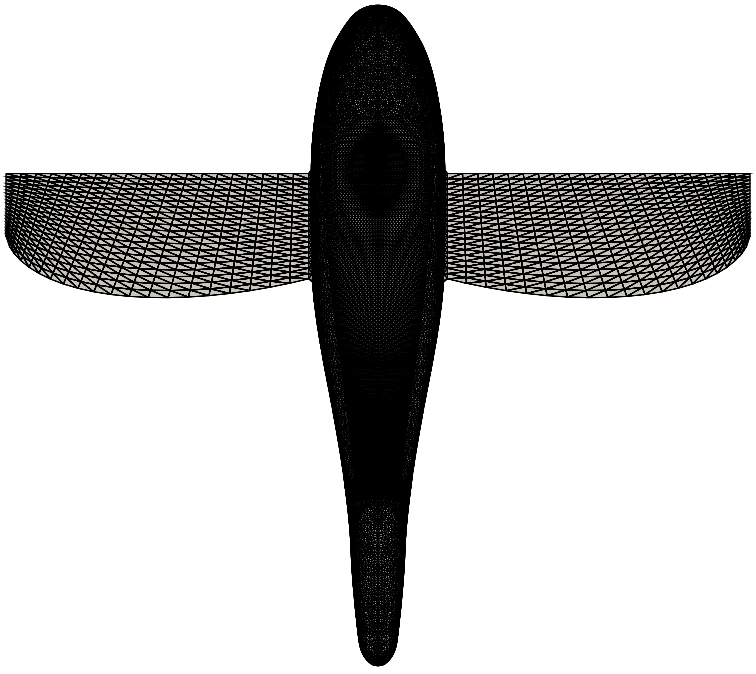

Figure 5.6: Dragonfly MAV. Surface mesh at $t/T = 0.5$.

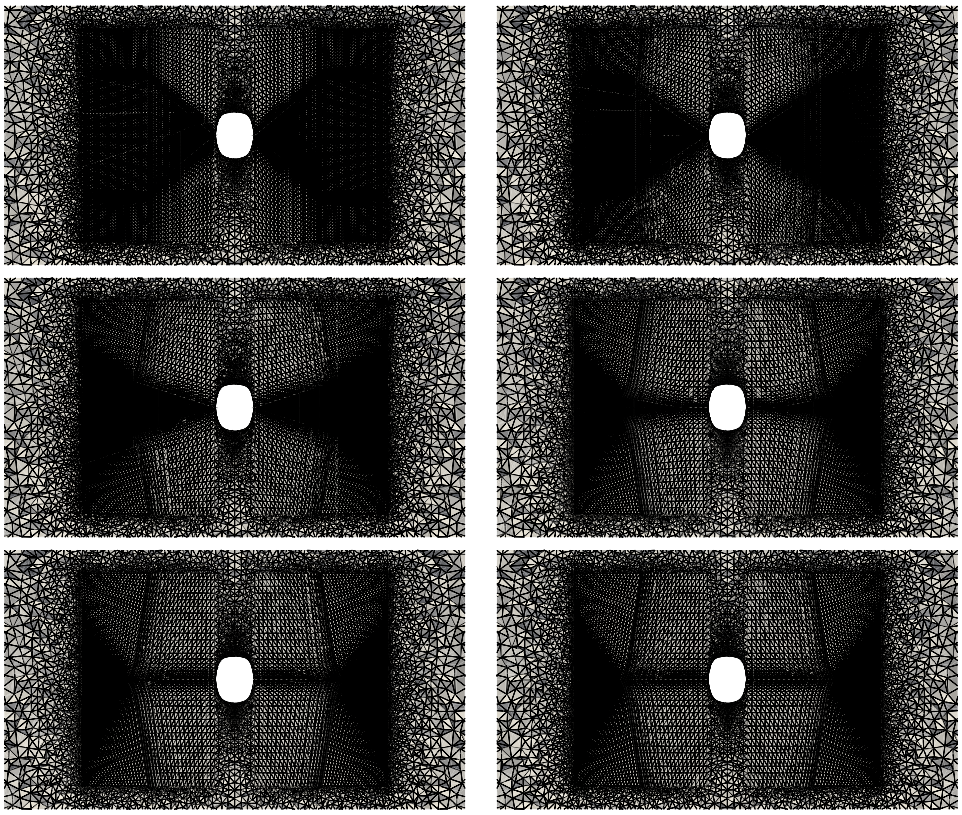

Figure 5.7: Dragonfly MAV. Mesh (cut mid-chord) at the same instants as in Figure 5.2 (left to right then top to bottom).

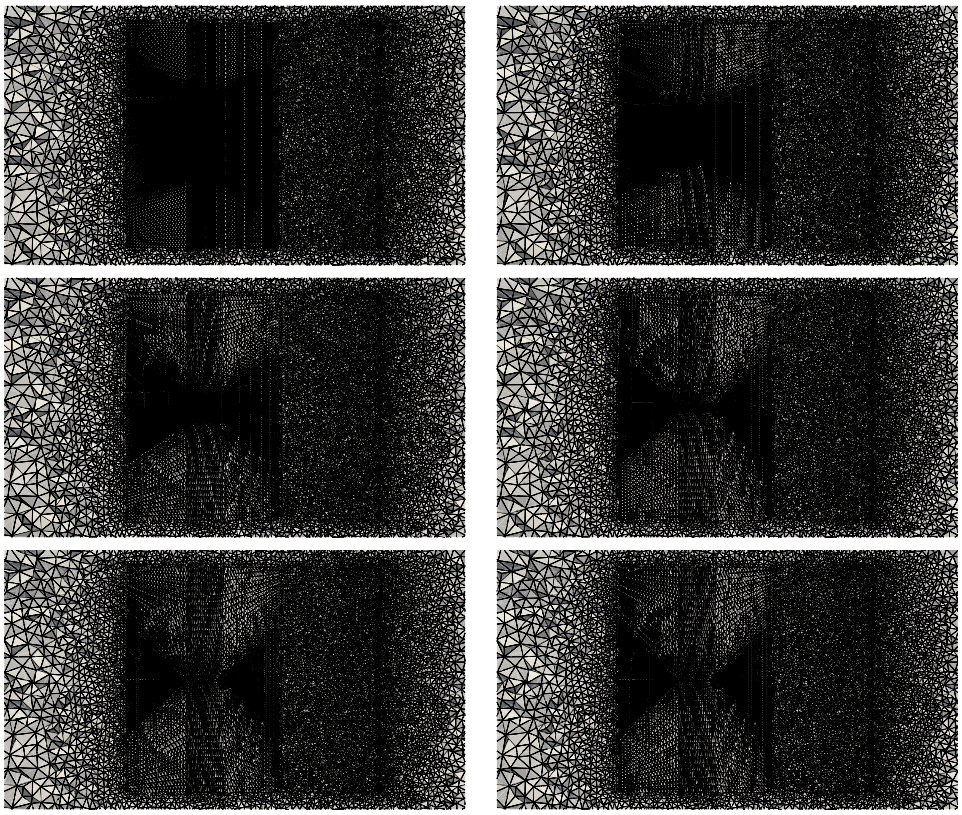

Figure 5.8: Dragonfly MAV. Mesh (cut mid-span) at the same instants as in Figure 5.2 (left to right then top to bottom).

Table 5.1: Dragonfly MAV. Number of nodes (nn) and elements (ne) in the meshes used.

Interface Mesh		
Single Wing	nn	802
	ne	1,600
Body	nn	8,521
	ne	16,630
Volume Mesh		
Inner	nn	405,002
	ne	2,303,920
Full ($\alpha = 0°$)	nn	1,143,613
	ne	6,844,706
Full ($\alpha = 5°$)	nn	1,227,618
	ne	7,353,540
Full ($\alpha = 10°$)	nn	1,152,367
	ne	6,896,762

For each angle of attack, prior to the flapping motion, we compute 550 time steps with the geometry at $t = 0$ to develop the flow field. For the first 500 time steps, only half of the computational domain is used and a slip boundary condition is enforced on the symmetry plane. The results are then copied to the other half of the mesh for the final 50 time steps of flow field development. The inflow velocity of 4.5 m/s is reached by a sinusoidal ramping over the first 150 time steps, starting from 0.0 m/s. In computing the developed flow field, the number of GMRES iterations per nonlinear iteration is 150, 350, 450, and 800. In computing the flapping cycles, the number of GMRES iterations is 250, 500, 750, and 1,000. We compute three flapping cycles and display the results for the third cycle.

5.3 Results

We first present (in Figures 5.9–5.14), only for $\alpha = 10°$, results over (or in relationship to) the MAV body and wing surfaces. Figure 5.9 shows the helicity isosurfaces. The flow field near the wings is almost symmetric, but the flow behind the MAV is not.

Figures 5.10 and 5.11 show pressure on the body and the wing surfaces. The pressure is almost symmetric, and therefore we use the left and right sides of the wing pictures for the upper and lower wings. For the body, however, both sides show the upper surface. Figures 5.12 and 5.13 show the magnitude of shear stress on the body and wing surfaces. Again, we use the left and right sides of the wing pictures for the upper and lower wings, and both sides of the body for the upper surface. Figure 5.14 shows the pressure difference between the upper and lower surfaces. The left side is for the upper wing, and the right side for the lower wing. For the closed parts of the wings, both sides show the difference between the lower surface of the lower wing and the upper surface of the upper wing. Lift and drag forces are shown in Figures 5.15–5.17. The forces are separated into the upper and lower wings, the closed wings, and the body.

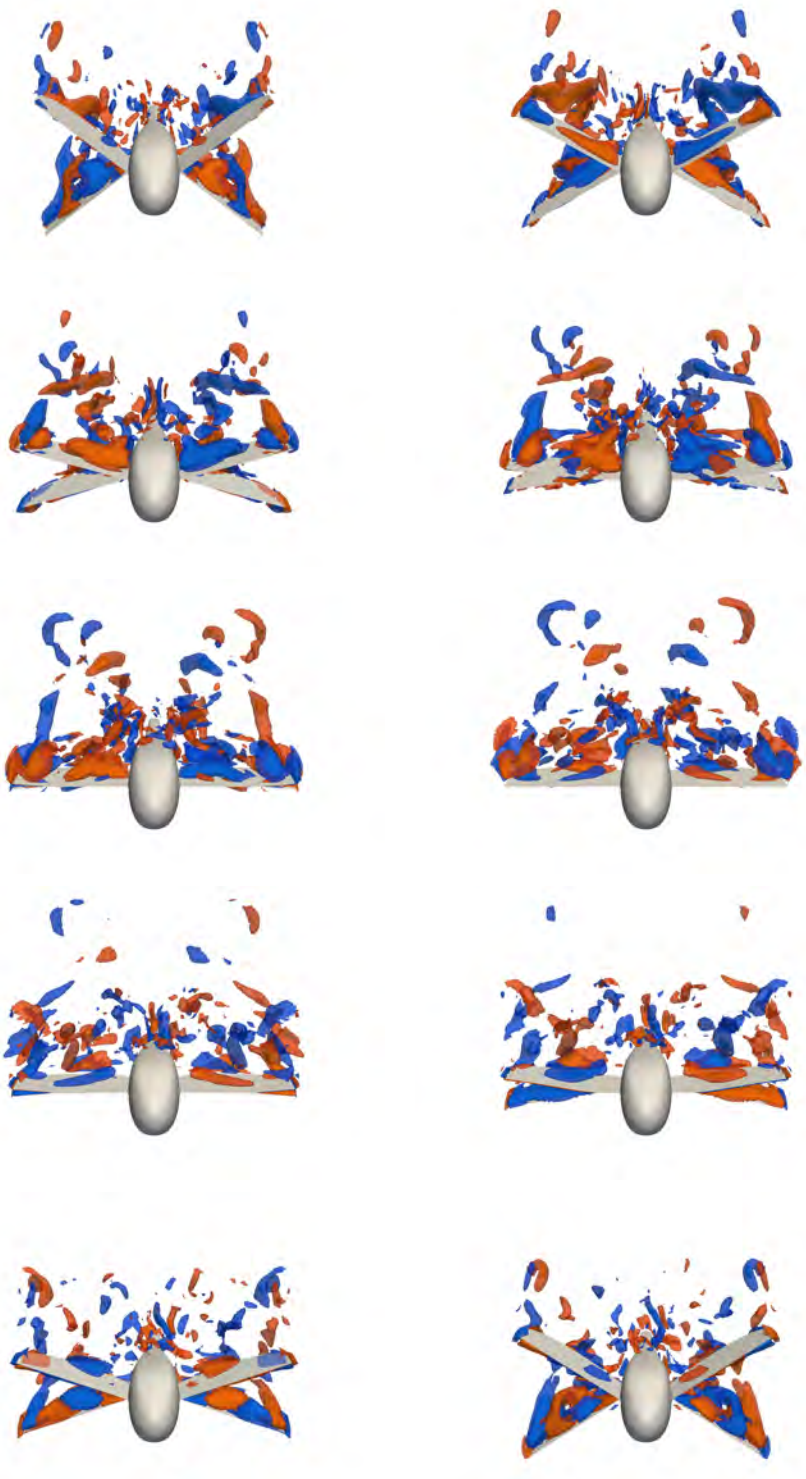

Figure 5.9: Dragonfly MAV. Helicity isosurfaces (± 5 and ± 10 m^2/s^2) for $\alpha = 10°$ at $t/T = 0.0$, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9 (left to right and then top to bottom). Blue is for negative values, and red for positive.

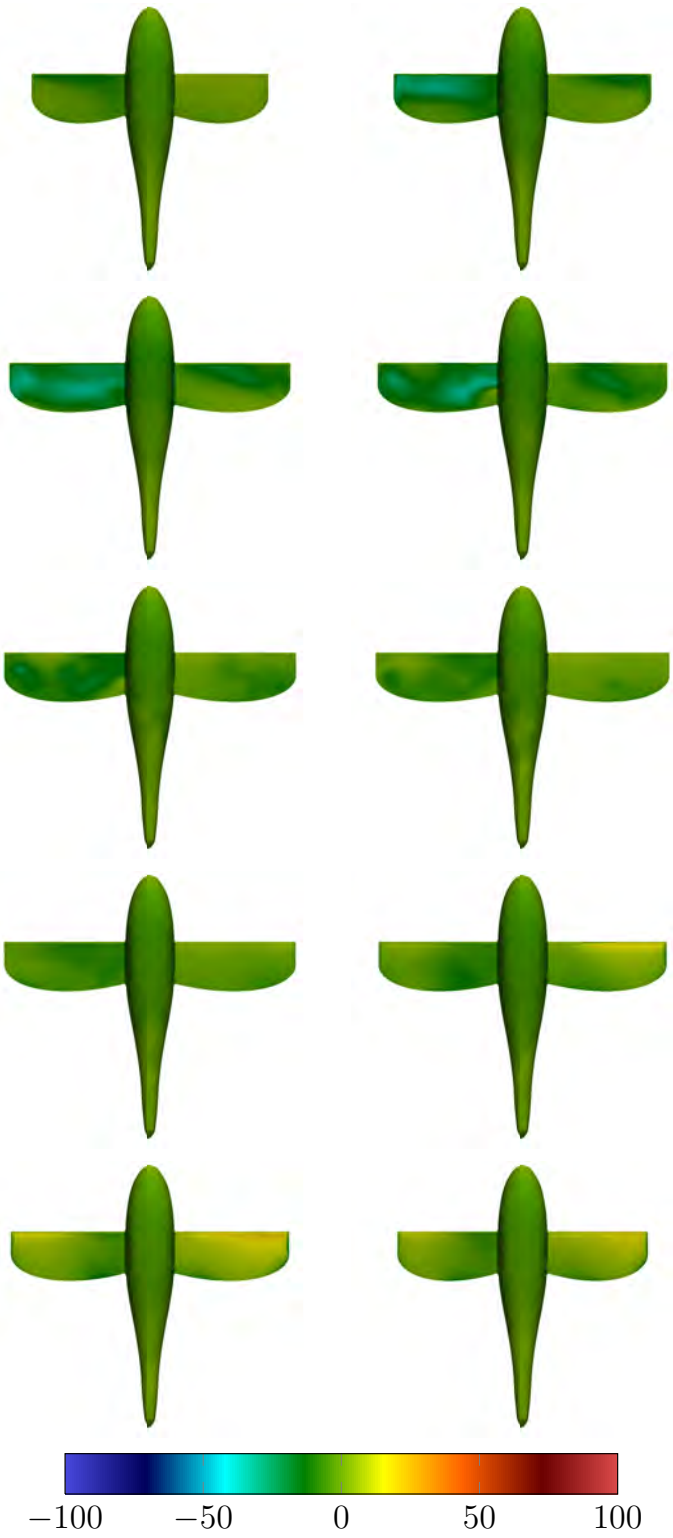

Figure 5.10: Dragonfly MAV. Pressure (Pa) for $\alpha = 10°$ on the body and wings at the same instants as in Figure 5.9. The upper surface of the upper wing (left side) and the lower surface of the lower wing (right side).

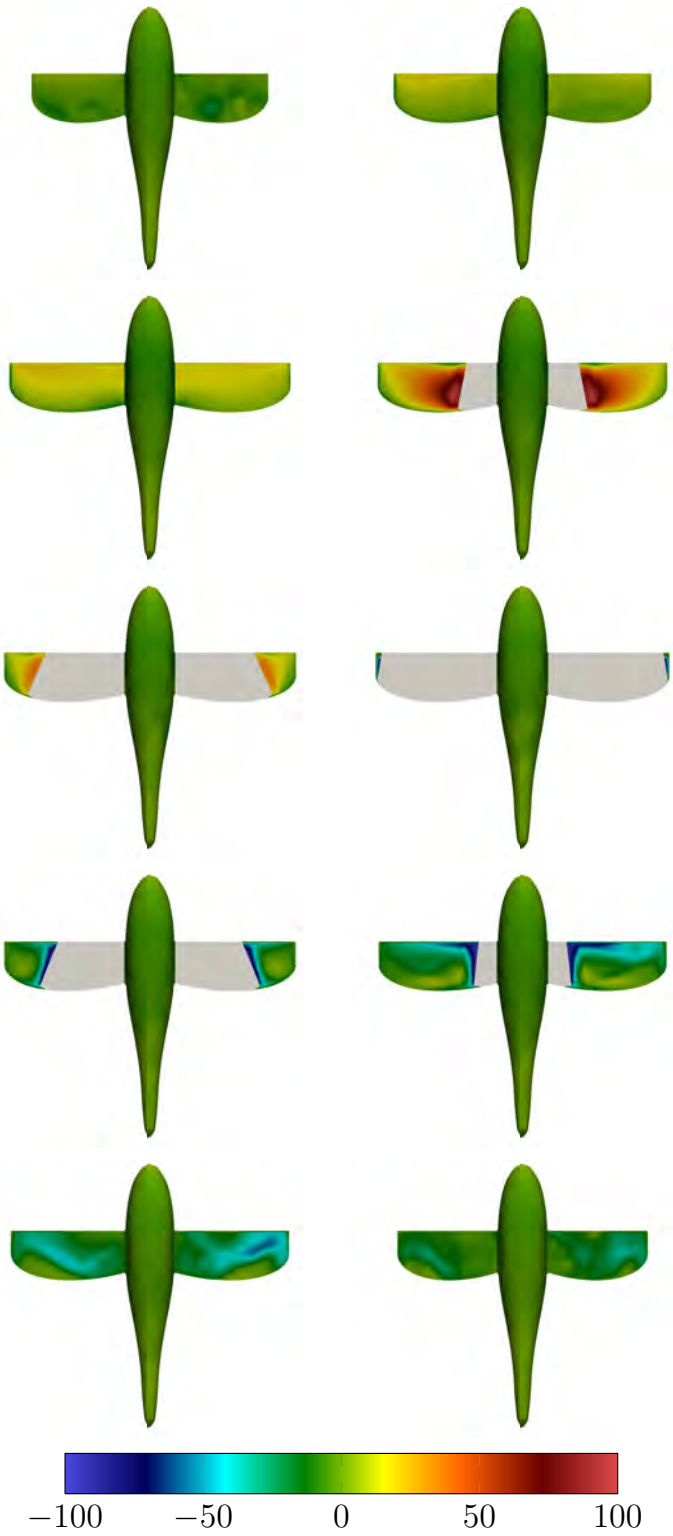

Figure 5.11: Dragonfly MAV. Pressure (Pa) for $\alpha = 10°$ on the body and wings at the same instants as in Figure 5.9. The lower surface of the upper wing (left side) and the upper surface of the lower wing (right side). The white regions are the closed parts of the wings.

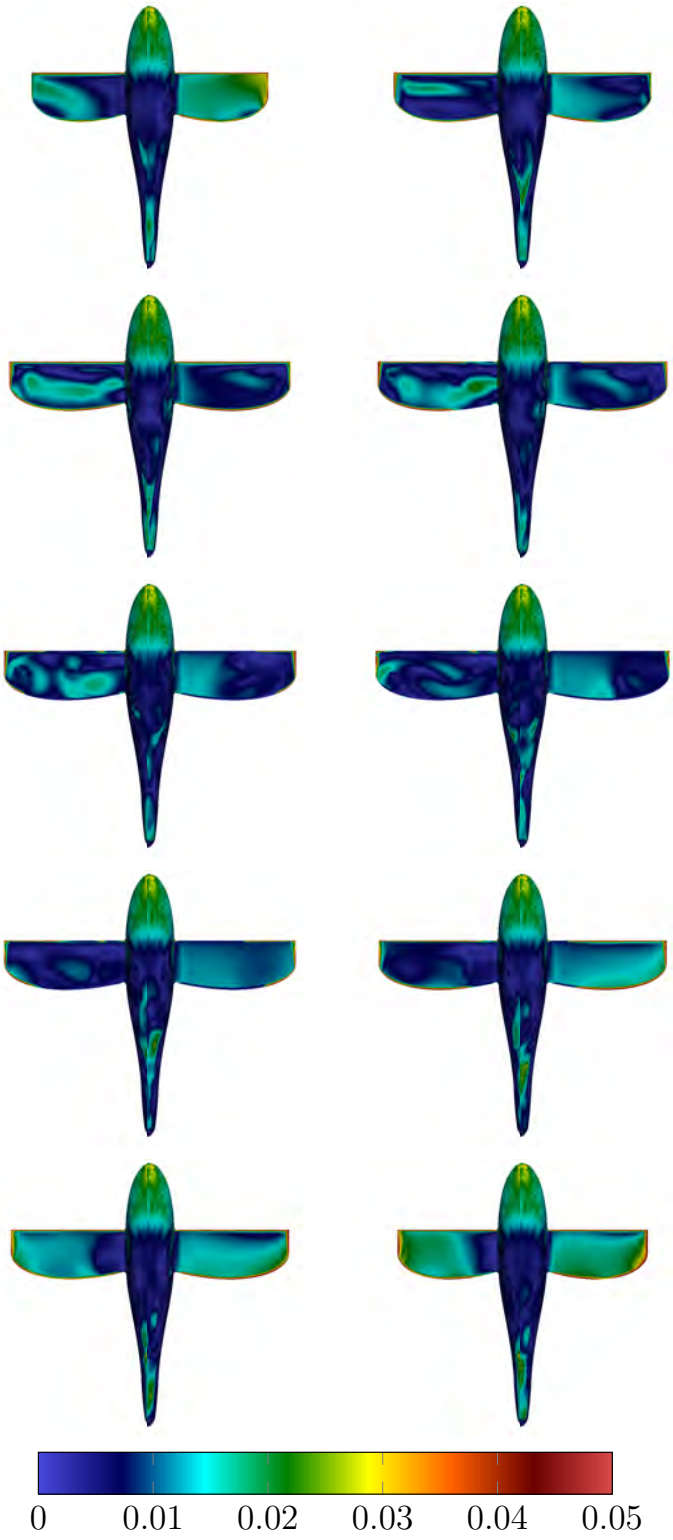

Figure 5.12: Dragonfly MAV. Magnitude of the shear stress (Pa) on the body and the wing surfaces at the same instants as in Figure 5.9. The upper surface of the upper wing (left side) and the lower surface of the lower wing (right side).

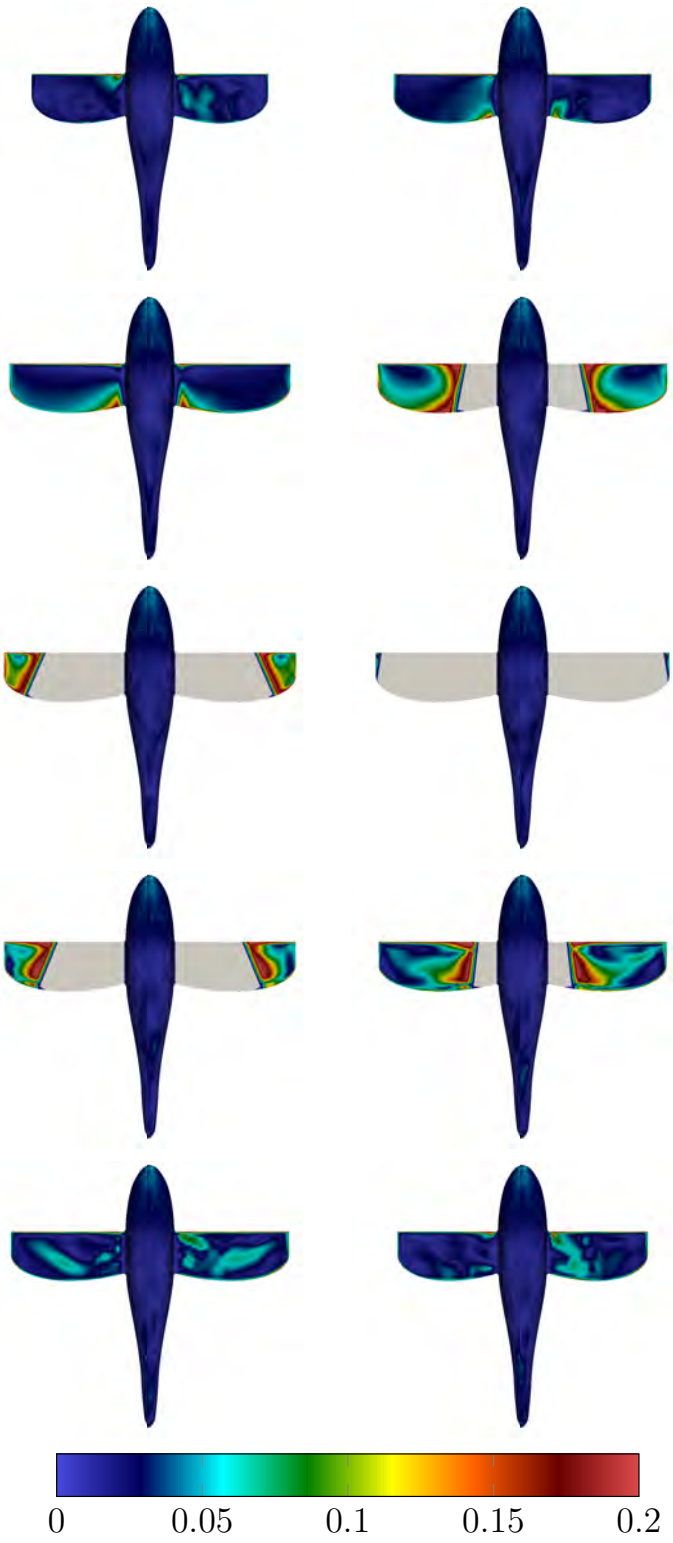

Figure 5.13: Dragonfly MAV. Magnitude of the shear stress (Pa) on the body and the wing surfaces at the same instants as in Figure 5.9. The lower surface of the upper wing (left side) and the upper surface of the lower wing (right side). The white regions are the closed parts of the wings.

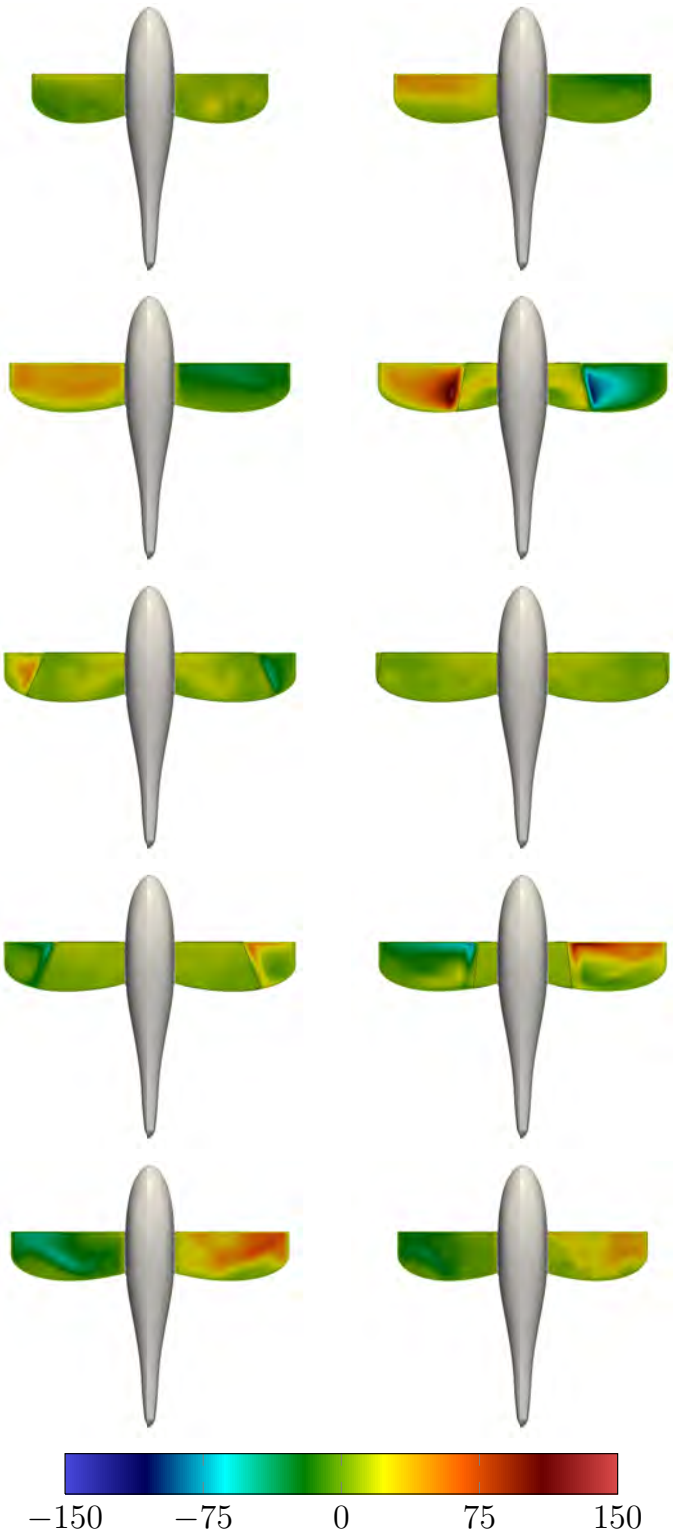

Figure 5.14: Dragonfly MAV. Pressure difference (Pa) between the lower and upper surfaces for $\alpha = 10°$ at the same instants as in Figure 5.9. For the upper wing and closed wings (left side) and for the lower wing and closed wings (right side).

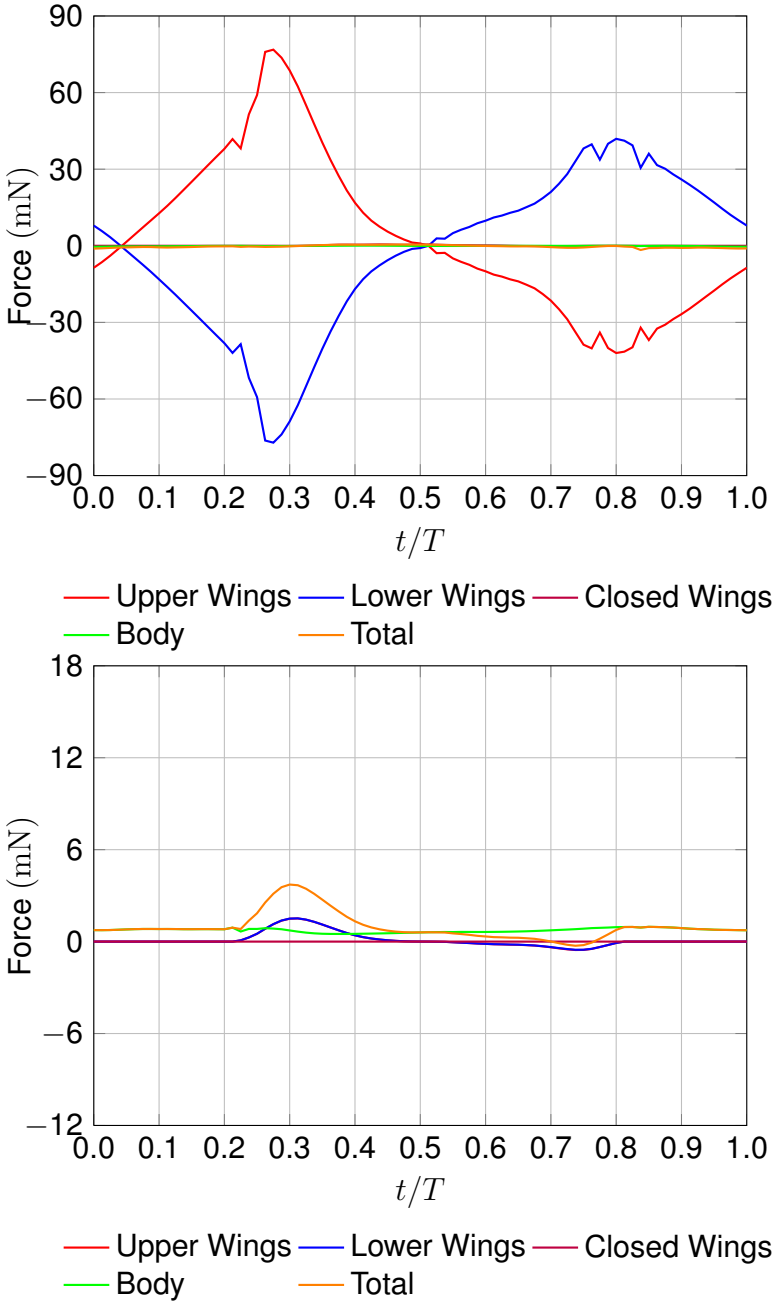

Figure 5.15: Dragonfly MAV. Lift (top) and drag (bottom) for $\alpha = 0°$.

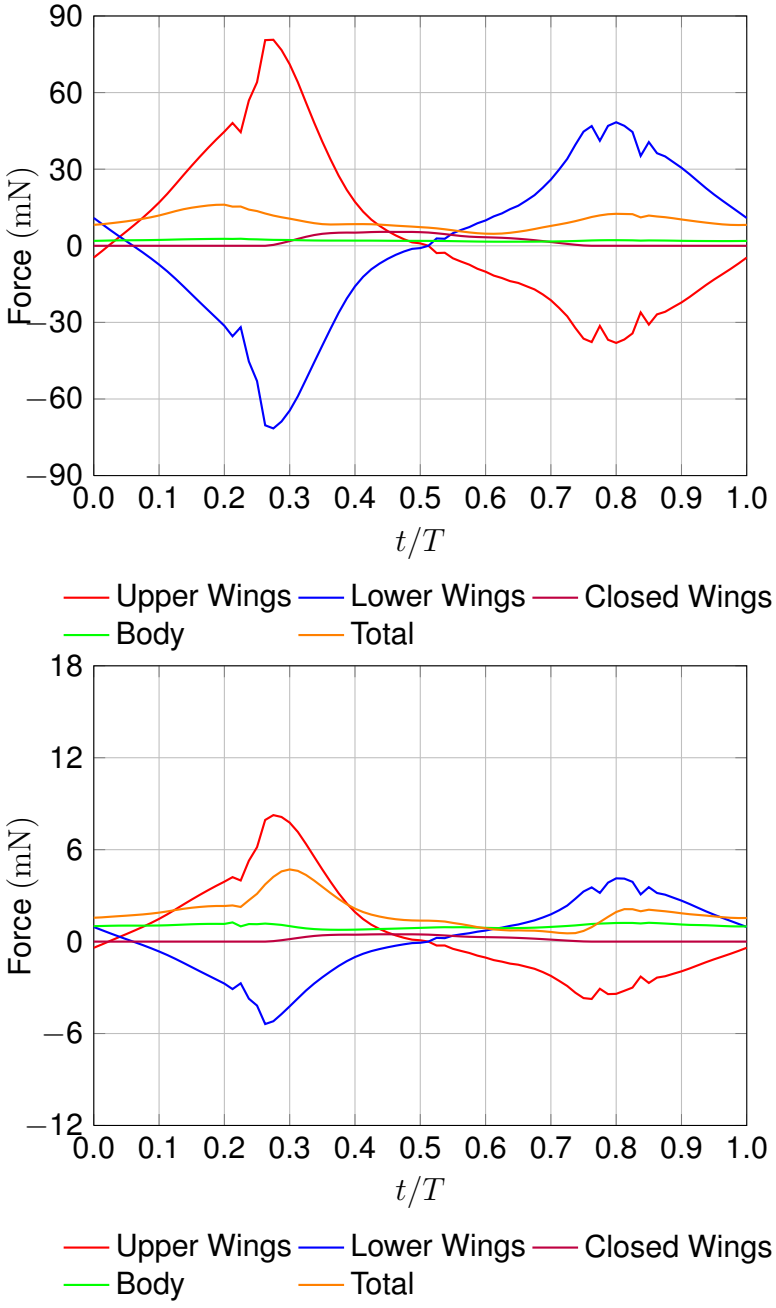

Figure 5.16: Dragonfly MAV. Lift (top) and drag (bottom) for $\alpha = 5°$.

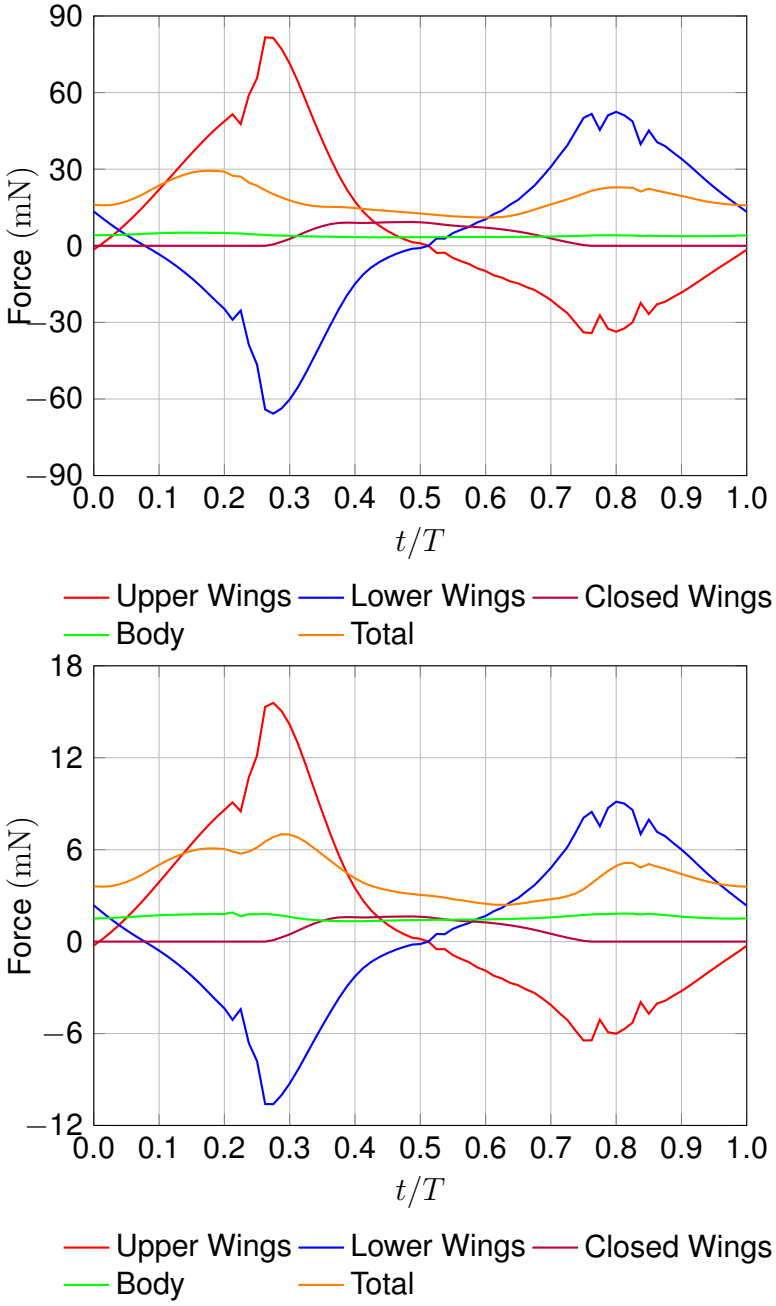

Figure 5.17: Dragonfly MAV. Lift (top) and drag (bottom) for $\alpha = 10°$.

Chapter 6

Aortic valve with coronary arteries

The material in this chapter is from [58].

6.1 Geometry and motion modeling

We create a typical aortic-valve model based on pictures, such as the one in [21]. The model, shown in Figures 6.1 and 6.2, has three leaflets with two outlets, corresponding to coronary arteries, and one main outlet, corresponding to the beginning of the aorta. The outlets are extended straight in each direction. The bulges are called sinuses. The left and right coronary arteries are attached to the left and right aortic sinuses, respectively, and the other sinus is called the posterior aortic sinus.

The inlet and main outlet diameters are 23 mm, which correspond to a typical aorta. The two coronary artery diameters are 2.9 mm.

We prescribe the motion of the three leaflets so that the valve closes and opens with a period of $T = 0.6$ s. The inflow velocity is specified such that the average flow rate is 5,000 ml/min. The flow rate is a time-variant function of the horizontal projection of the open mouth area of the valve when it is open, and of the inlet-side volume change while the valve is changing shape after it closes (see Figure 6.3). The density and kinematic viscosity of the blood are 1,000 kg/m^3 and 4.0×10^{-6} m^2/s.

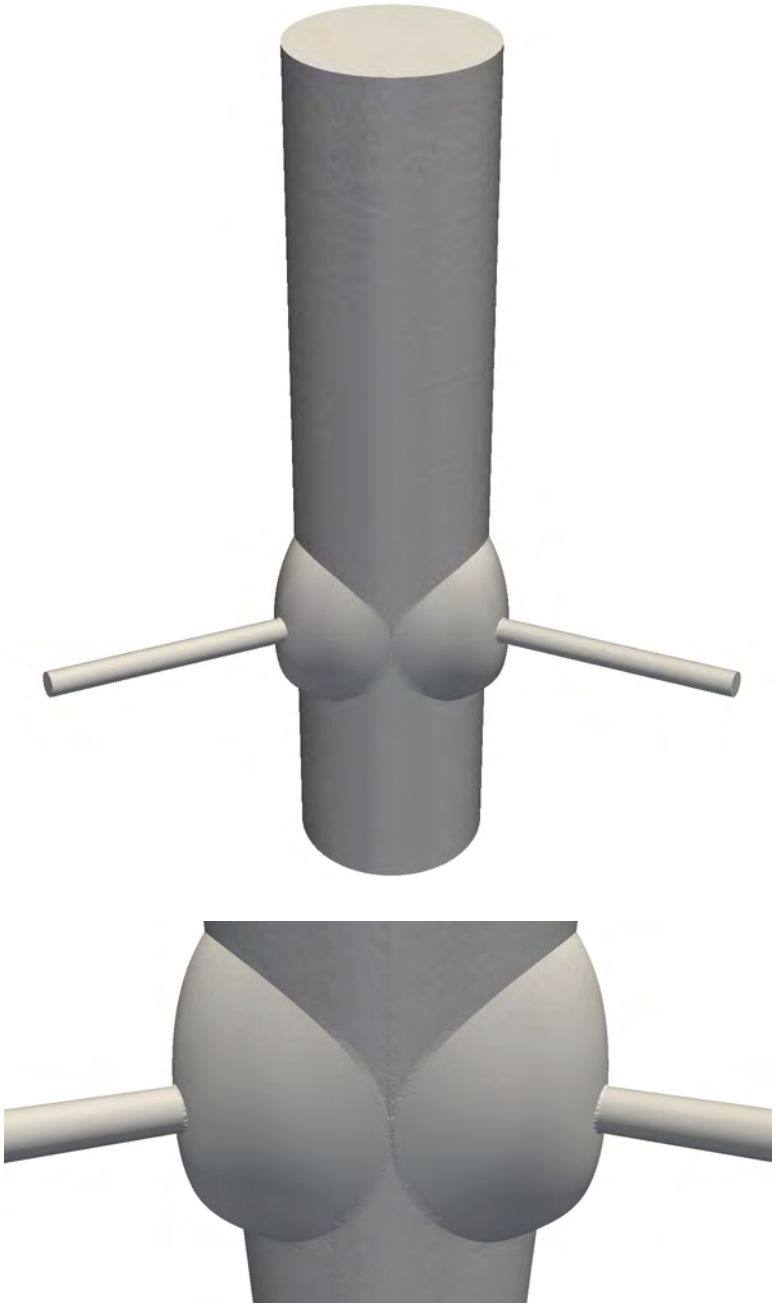

Figure 6.1: Aortic valve with coronary arteries. Model geometry. Aorta, leaflets, sinuses, and coronary arteries. The left coronary artery is on the right in the figure, and the right coronary artery is on the left in the figure.

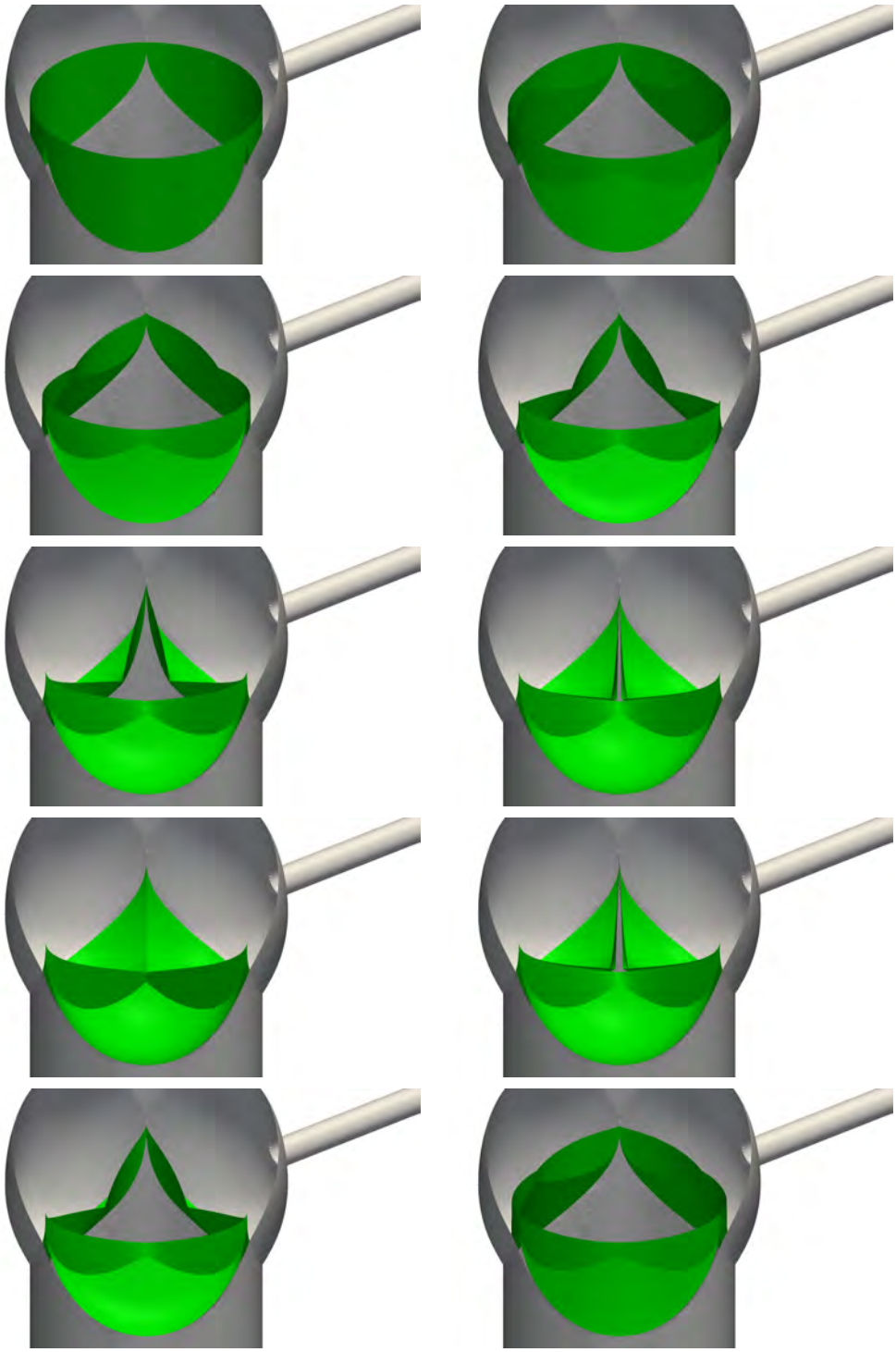

Figure 6.2: Aortic valve with coronary arteries. Leaflets at $t/T = 0.0$, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, and 0.9 (left to right and then top to bottom).

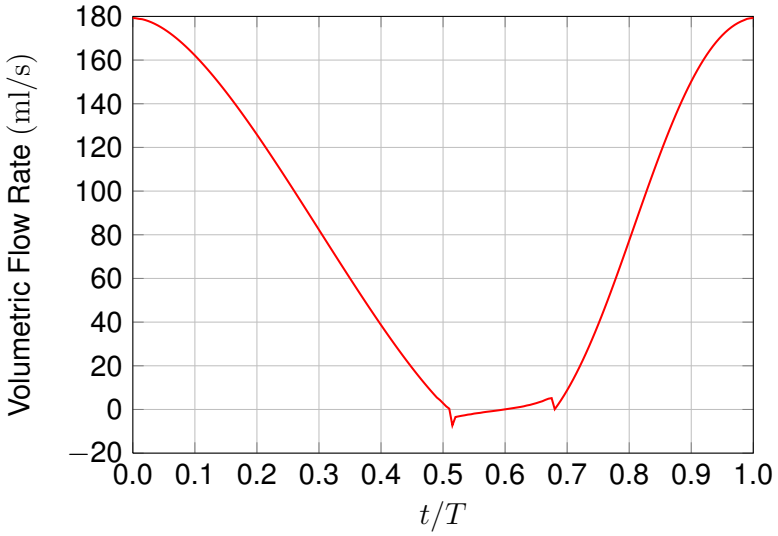

Figure 6.3: Aortic valve with coronary arteries. Inflow profile.

The boundary conditions are no-slip on the arterial walls and valves, traction-free at the outflow boundaries, and uniform velocity at the inflow boundary. The surface mesh on the leaflets (shown in Figure 6.4) is made up of 8,448 nodes and 16,440 triangular elements. The arterial-wall surface mesh is shown in Figure 6.5. The mesh has structured, inner zones around the leaflets and an unstructured, outer zone. The inner zones consist of tetrahedral, pyramid-shaped, and wedge-shaped elements, and the outer zone consists of tetrahedral elements. The volume mesh is made up of 1,417,910 nodes and 4,184,614 mixed elements. The mesh near the valve is shown in Figure 6.6.

During the prescribed motion, only the inner zones move with a special, algebraic mesh-moving technique. The positions of the nodes in the inner zones are created by linearly interpolating the surface mesh of the zones from the closed position to the open position. Seventy-nine layers of nodes are extruded from both the upper and lower surfaces of the leaflets.

When the valve is completely open, all of the fluid nodes extruded from the upper surface are slaves to the upper surface, and all of the nodes extruded from the lower surface are masters. As the valve closes, it leaves one layer of nodes attached to the

upper surface behind, and the leaflet surface moves to the next node position, making those nodes slaves to the lower surface. When the valve is completely closed, all of the fluid nodes extruded from the upper surface are masters and all of the nodes extruded from the lower surface are slaves to the lower surface.

6.2 Computational conditions

We use the DSD/SST-SUPS and DSD/SST-VMST (convective) techniques for the first two and last two nonlinear iterations of each time step. The stabilization parameter τ_{SUPS} comes from the τ_{SUPG} definition in [67], specifically the definition given by Eqs. (107)–(109) in [67], which can also be found as the definition given by Eqs. (7)–(9) in [70], with ν_{LSIC} from Eq. (17) in [70]. Prior to the prescribed motion, we compute 150 time steps with the geometry at $t = 0$ to develop the flow field. The viscosity of 4.0×10^{-6} m^2/s is reached by ramping over the first 50 time steps starting from the viscosity 1.31×10^{-3} m^2/s. The ramping profile for the viscosity is designed to result in a linear ramping for the Reynolds number. The time-step size is 6.33×10^{-3} s during flow-field development, and 3.00×10^{-3} s for the prescribed-motion cycles. In computing the developed-flow field, the number of GMRES iterations per nonlinear iteration is 150, 350, 450, and 800. In computing the flapping cycles, the number of GMRES iterations is 250, 500, 750, and 1,000. We compute two cycles and display the results for the second cycle.

6.3 Results

The global mass-balance error, normalized by the average flow rate, is less than 10%. Figure 6.7 shows the flow rate at the outlets of the coronary arteries. There is some "negative outflow" (i.e. inflow) from the coronary arteries, however, the wall shear stress (WSS) on the long pipes are large enough to stabilize the overall system.

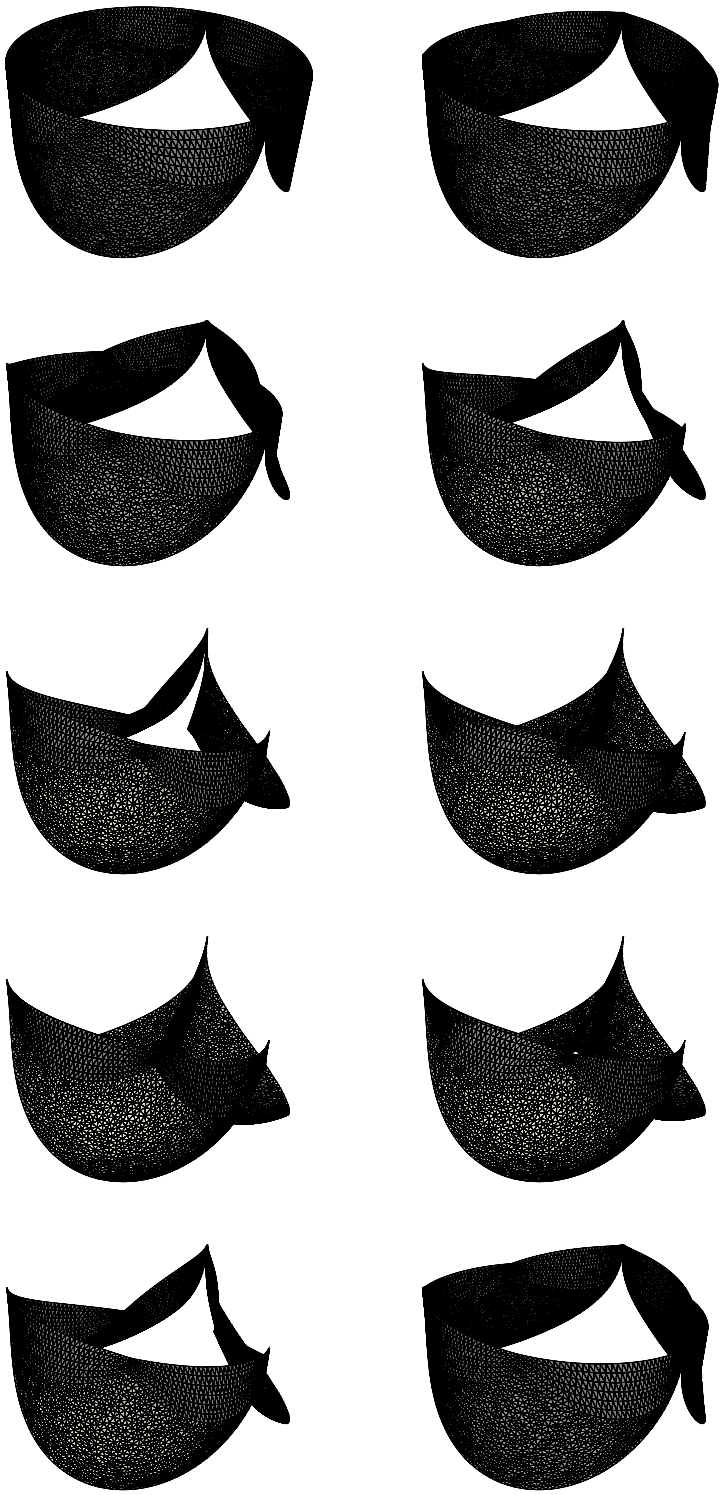

Figure 6.4: Aortic valve with coronary arteries. The leaflets surface mesh at the same instants as in Figure 6.2.

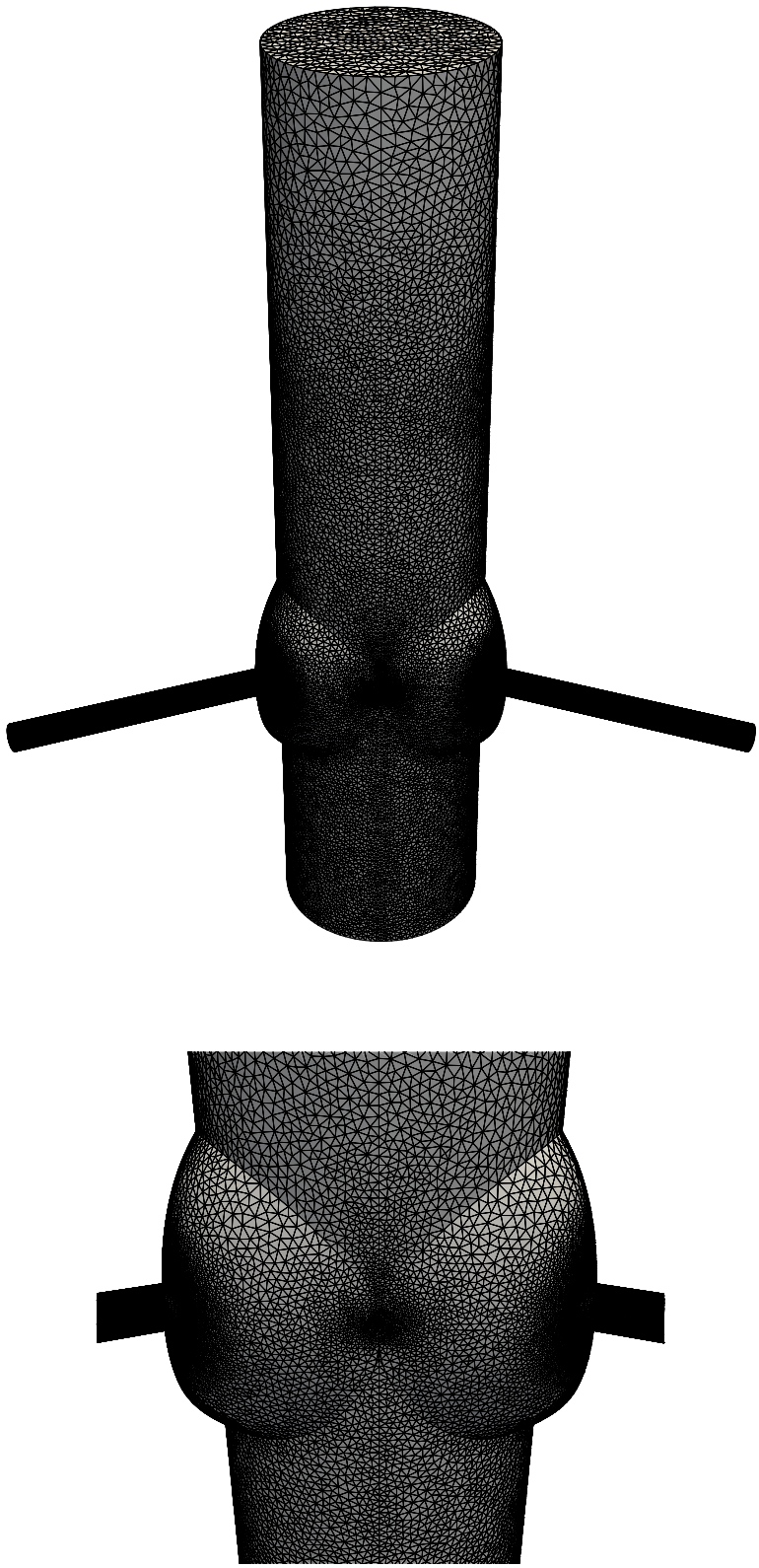

Figure 6.5: Aortic valve with coronary arteries. Mesh of the aortic valve, sinuses, and coronary arteries.

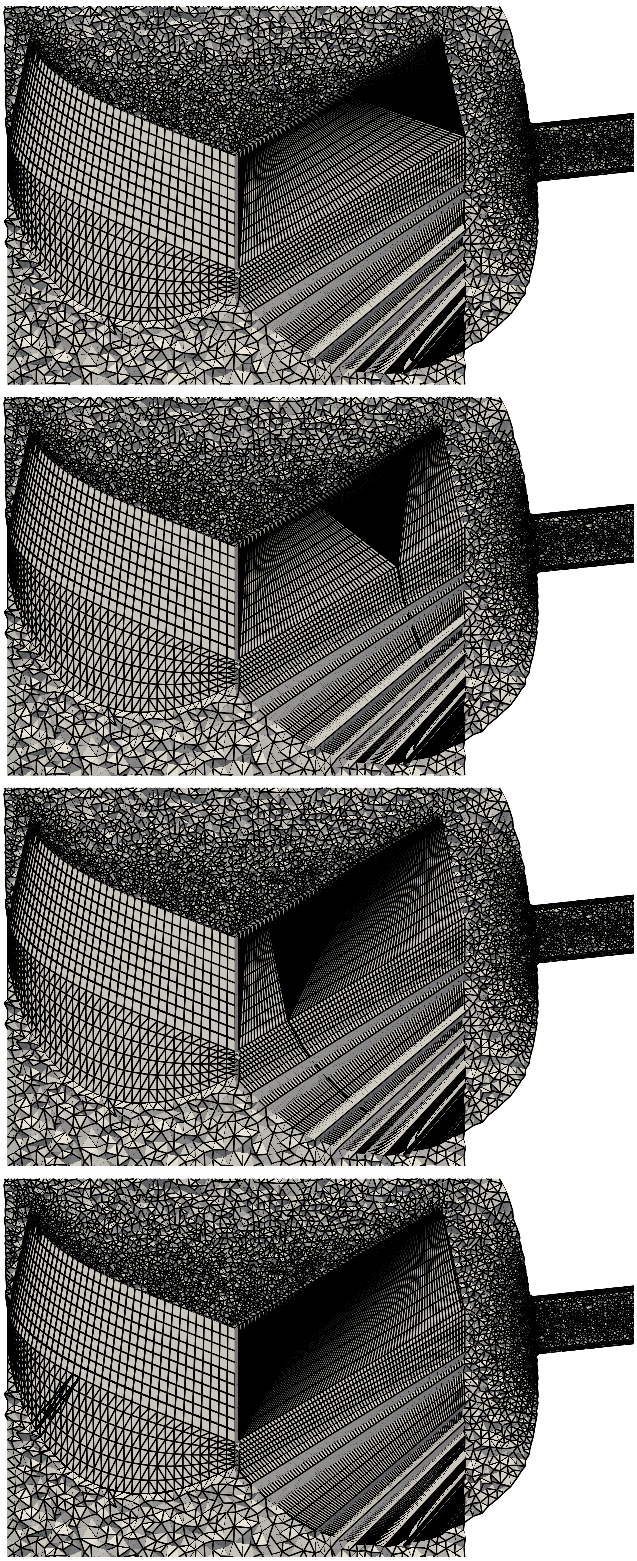

Figure 6.6: Aortic valve with coronary arteries. Mesh around the leaflets at $t/T =$ 0.0, 0.2, 0.4, and 0.6.

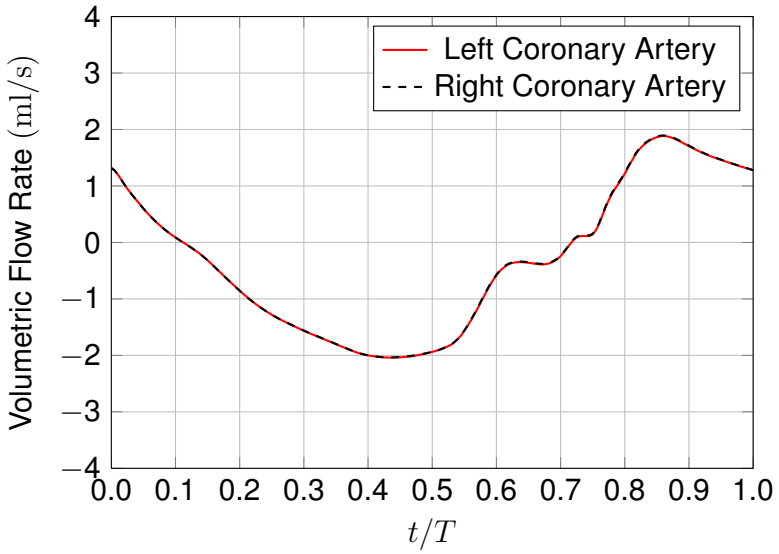

Figure 6.7: Aortic valve with coronary arteries. Flow rate at the outlets of the coronary arteries.

Figures 6.8-6.10 show a volume rendering of the velocity magnitude. Nonsymmetric and complex flow patterns are observed behind the valve. Figures 6.11 and 6.12 show the velocity magnitude on the "coronary plane," where the coronary arteries are connected to the aorta, and the "above-sinus plane," which is 18 mm downstream from the coronary plane. The mainstream flow oscillates away from the sinuses. This is mainly due to the jet from the contact of the leaflets of the sinuses.

Figures 6.13 and 6.14 show pressure difference between the lower and upper surfaces of the leaflets. We exclude the parts where the leaflets are in contact. Figures 6.15-6.18 show WSS on the leaflet surfaces. The WSS on the lower surfaces of the three leaflets are somewhat similar to each other. However, on the upper surfaces, the WSS for the leaflets of the coronary sinuses are different from the WSS for the leaflet of the posterior sinus. Figure 6.19 shows OSI on the leaflet surfaces. The WSS vector is projected onto the open configuration to calculate OSI (see [52]).

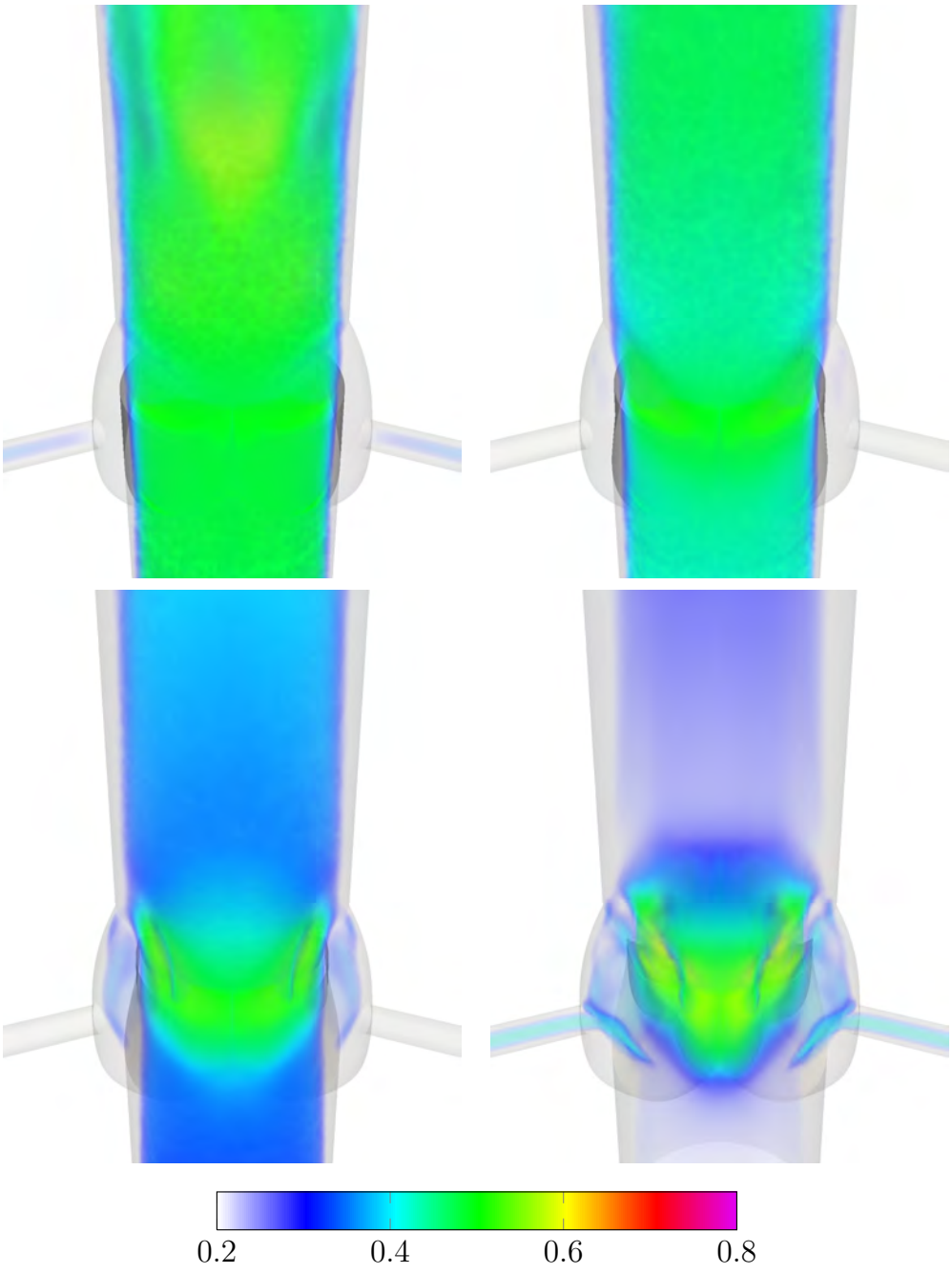

Figure 6.8: Aortic valve with coronary arteries. Volume rendering of the velocity magnitude (m/s) at $t/T = 0.0$, 0.1, 0.2, and 0.3.

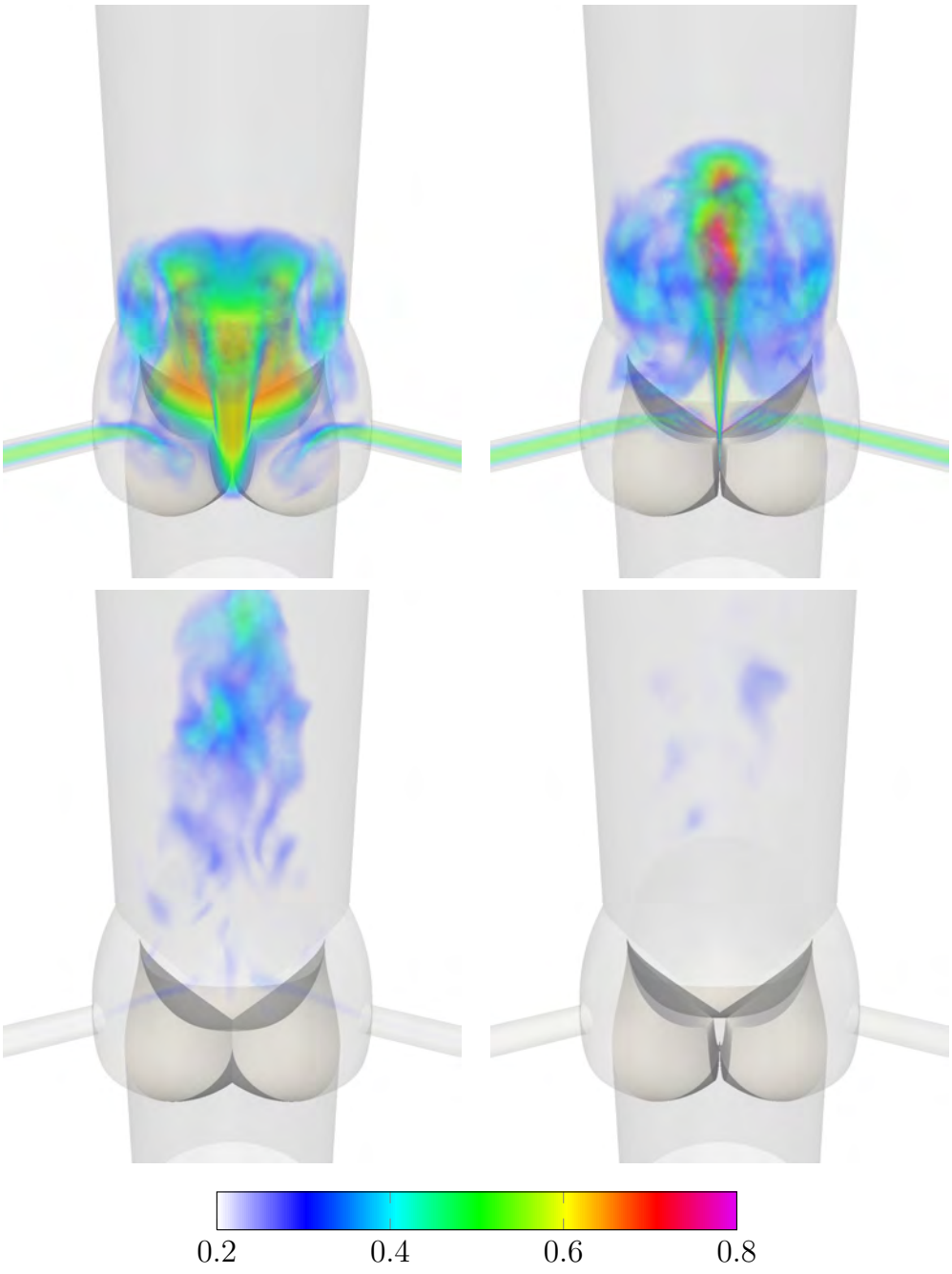

Figure 6.9: Aortic valve with coronary arteries. Volume rendering of the velocity magnitude (m/s) at $t/T = 0.4$, 0.5, 0.6, and 0.7.

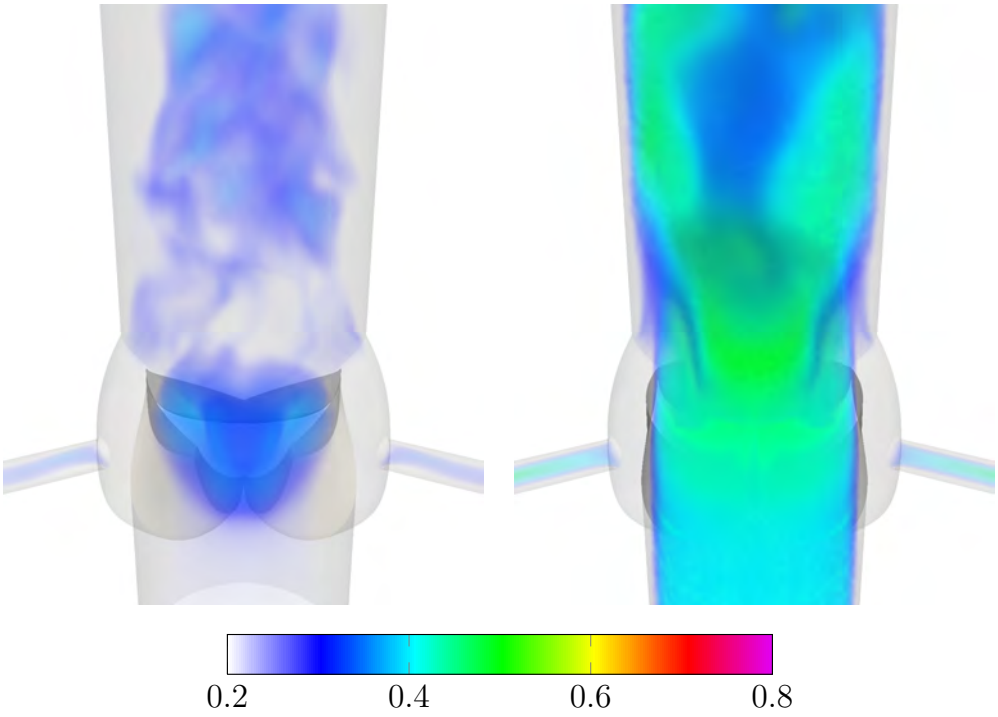

Figure 6.10: Aortic valve with coronary arteries. Volume rendering of the velocity magnitude (m/s) at $t/T = 0.8$, and 0.9.

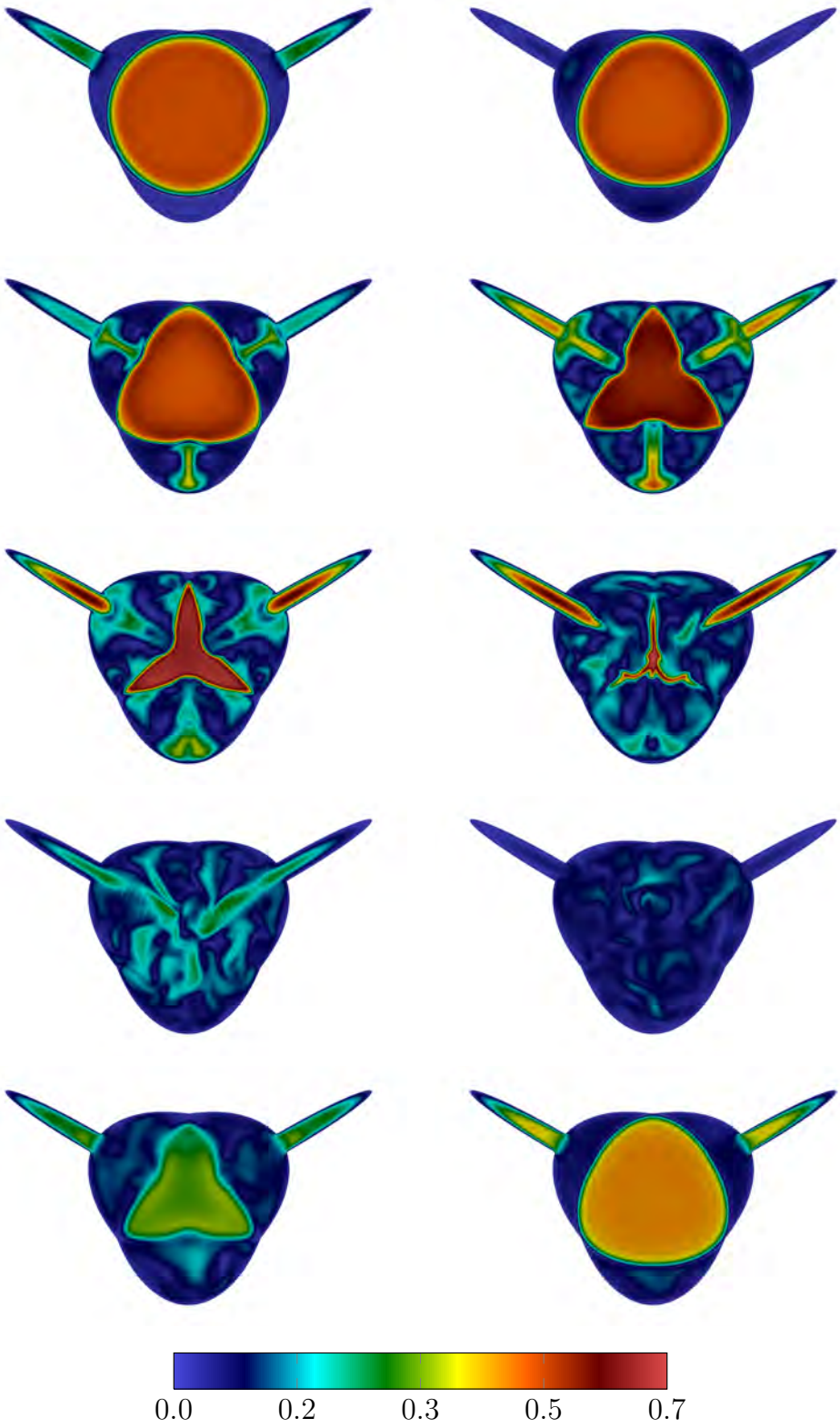

Figure 6.11: Aortic valve with coronary arteries. The velocity magnitude on the coronary plane (m/s) at the same time instants as in Figure 6.2.

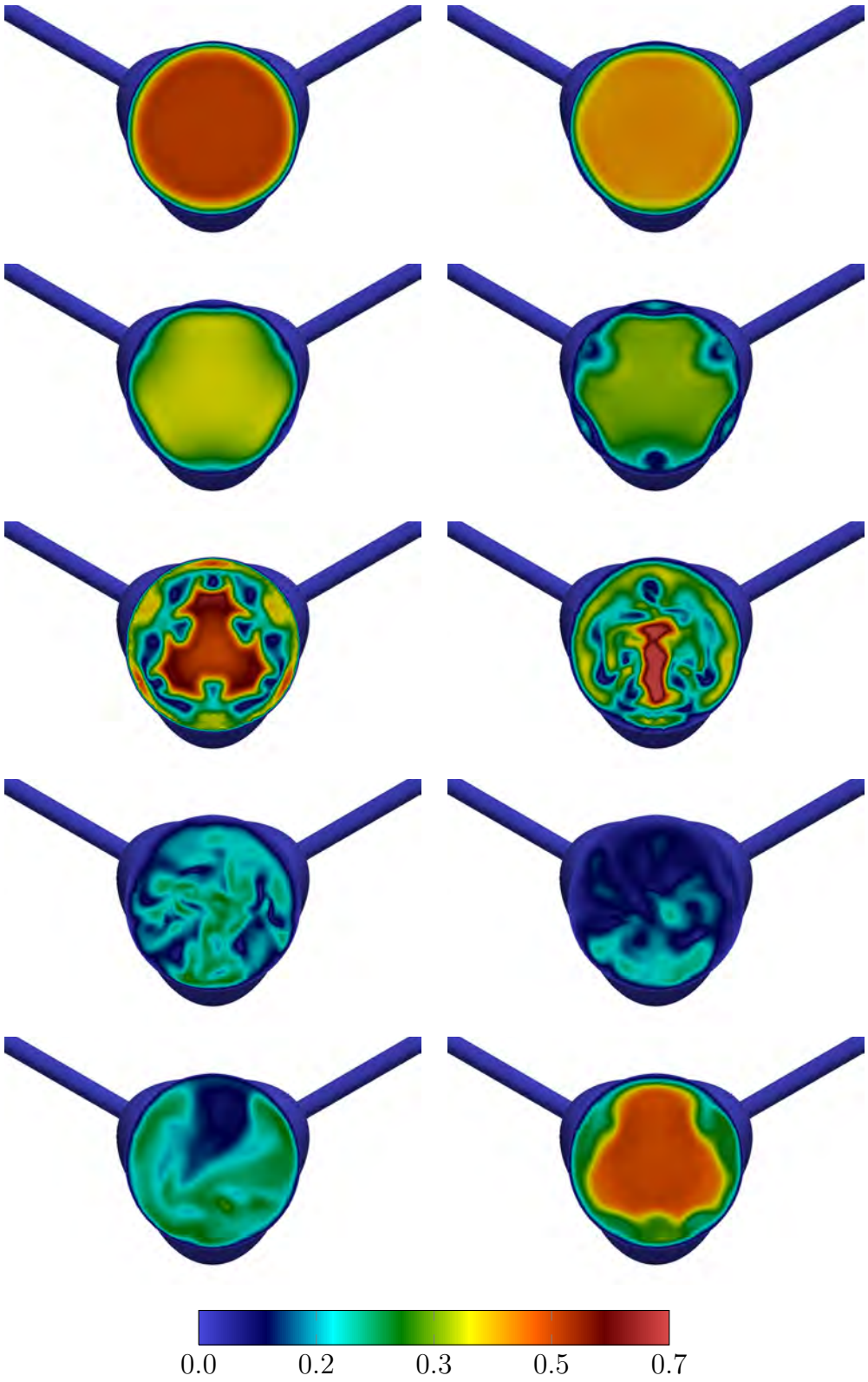

Figure 6.12: Aortic valve with coronary arteries. The velocity magnitude (m/s) on the above-sinus plane at the same time instants as in Figure 6.2.

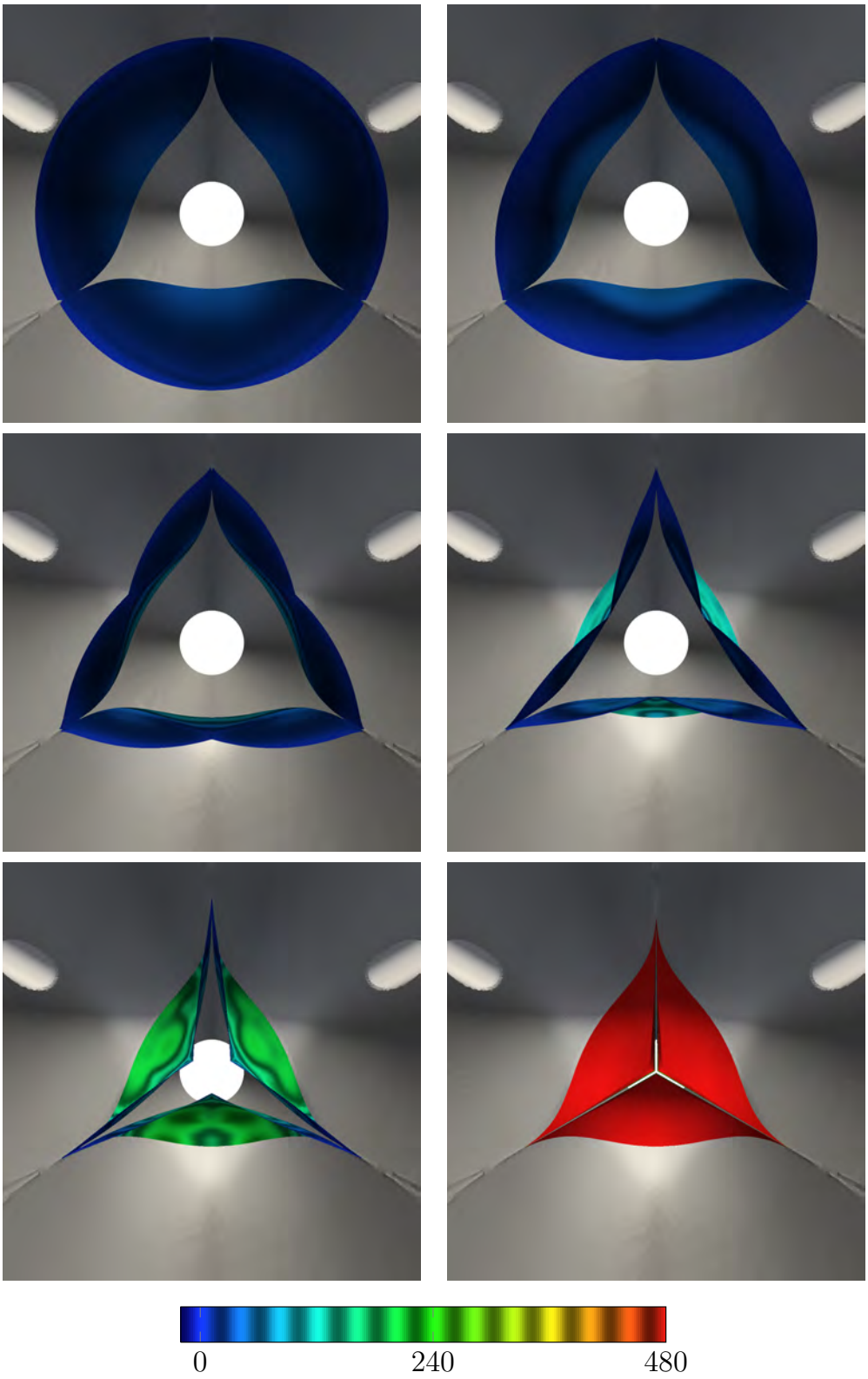

Figure 6.13: Aortic valve with coronary arteries. Pressure difference (Pa) between the lower and upper surfaces of the three leaflets at $t/T = 0.0$, 0.1, 0.2, 0.3, 0.4, and 0.5.

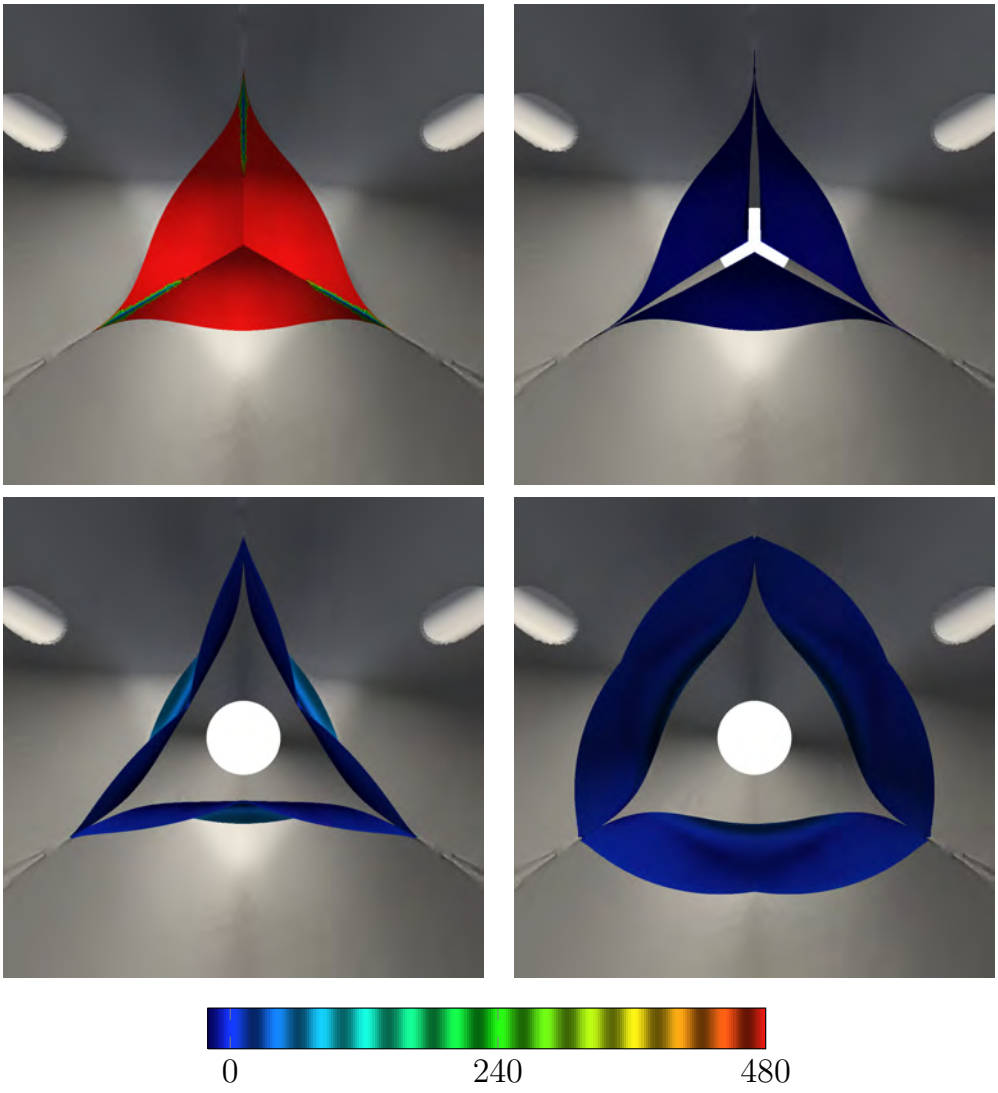

Figure 6.14: Aortic valve with coronary arteries. Pressure difference (Pa) between the lower and upper surfaces of the three leaflets at $t/T = 0.6$, 0.8, 0.8, and 0.9.

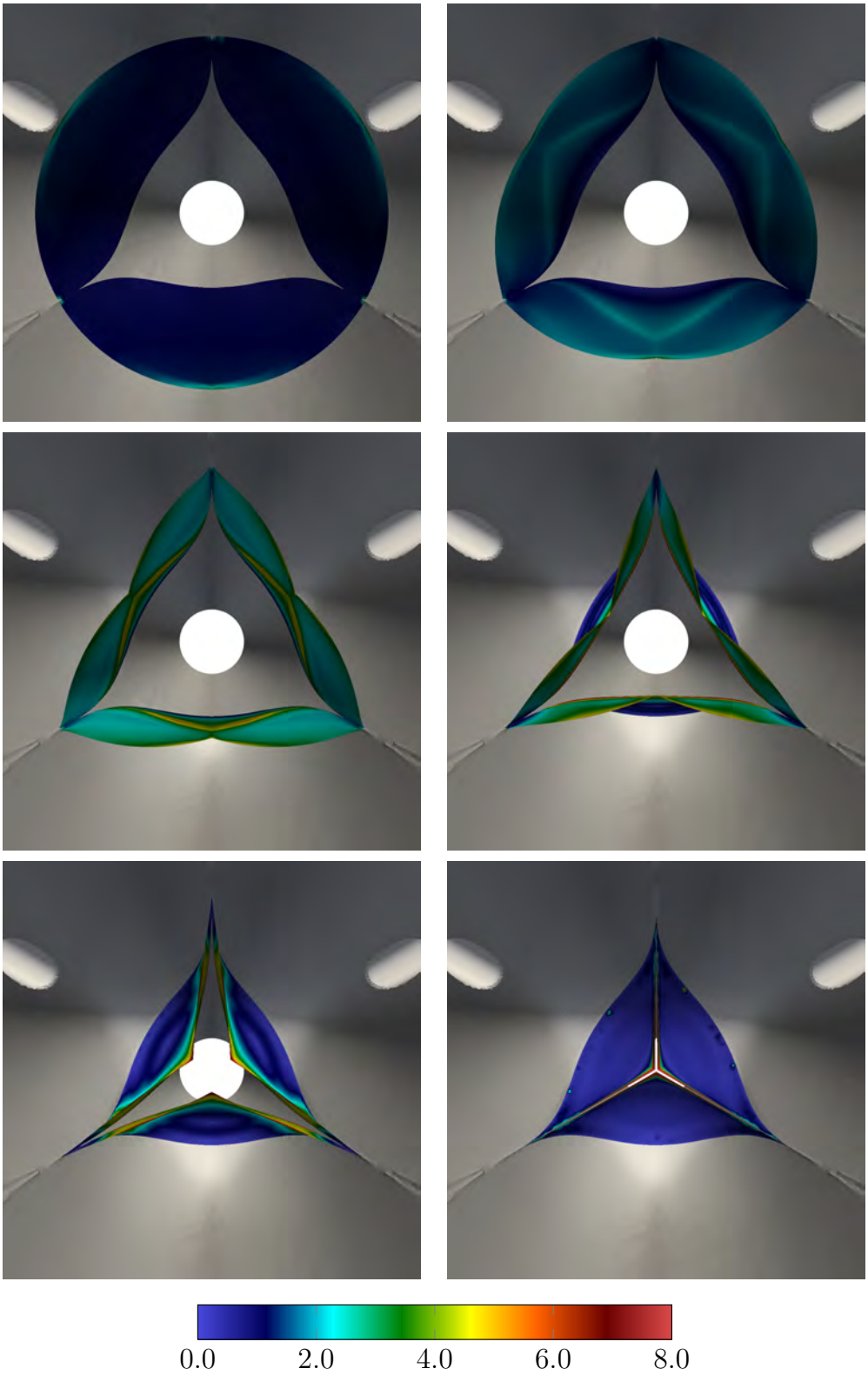

Figure 6.15: Aortic valve with coronary arteries. WSS (Pa) on the lower surface of the three leaflets at the same time instants as in Figure 6.13.

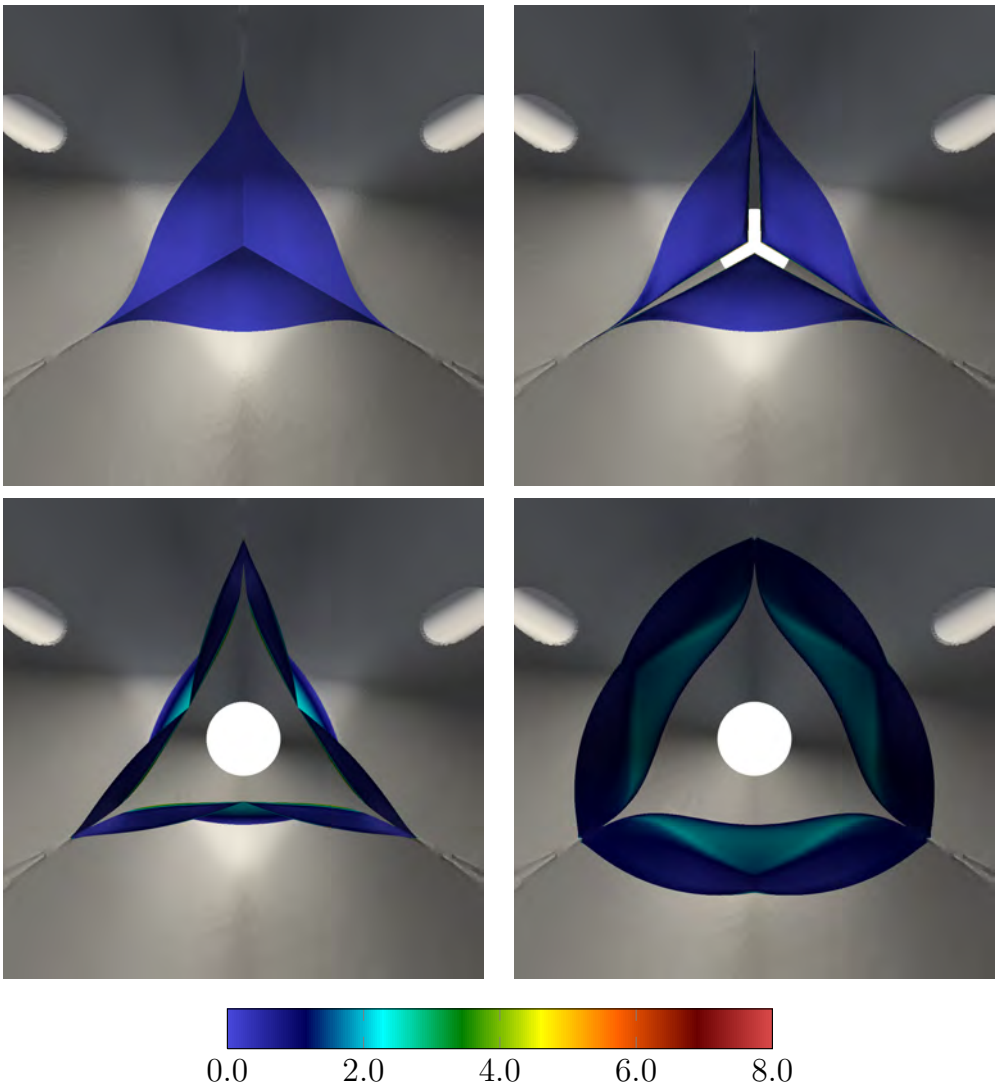

Figure 6.16: Aortic valve with coronary arteries. WSS (Pa) on the lower surface of the three leaflets at the same time instants as in Figure 6.14.

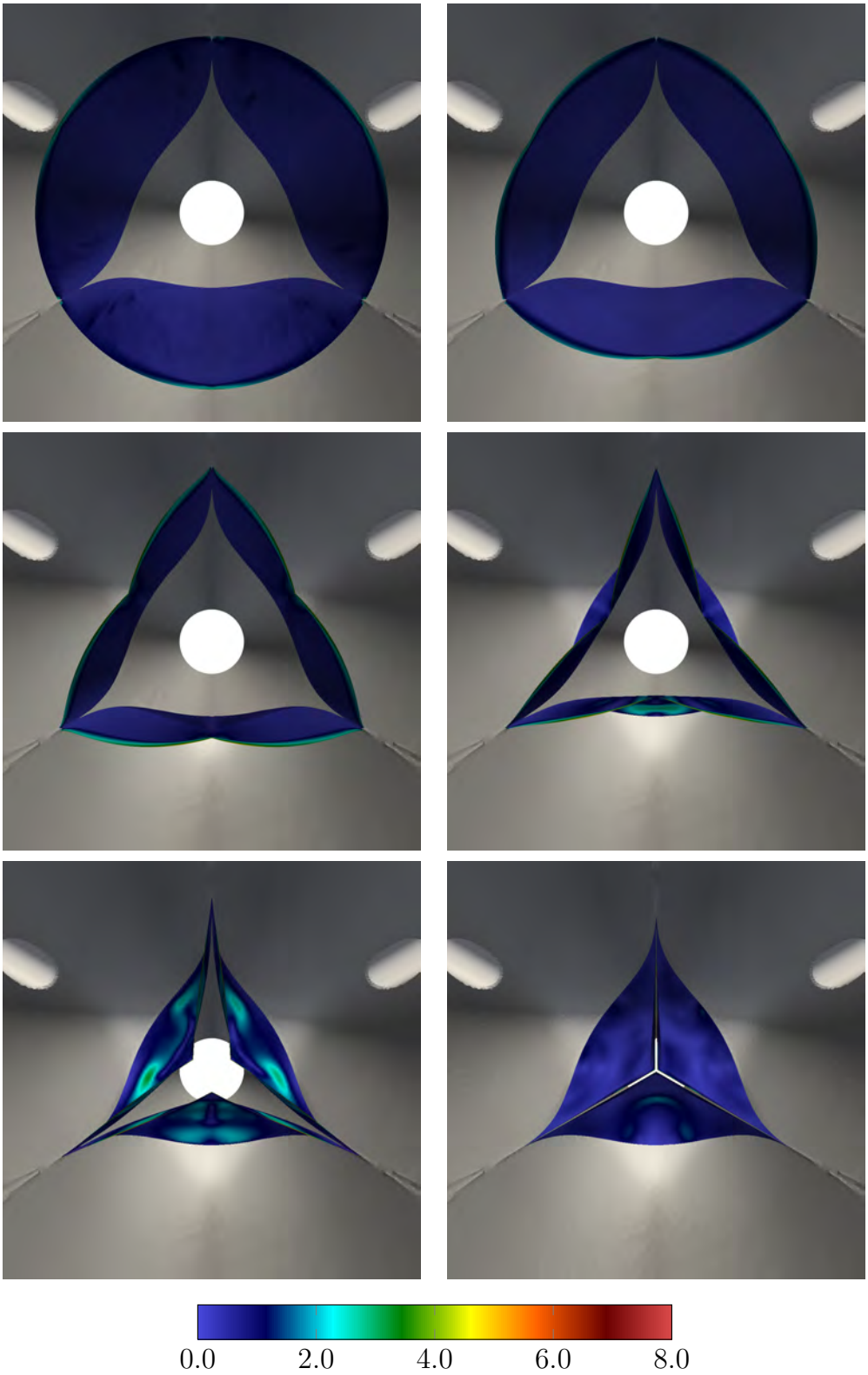

Figure 6.17: Aortic valve with coronary arteries. WSS (Pa) on the upper surface of the three leaflets at the same time instants as in Figure 6.13.

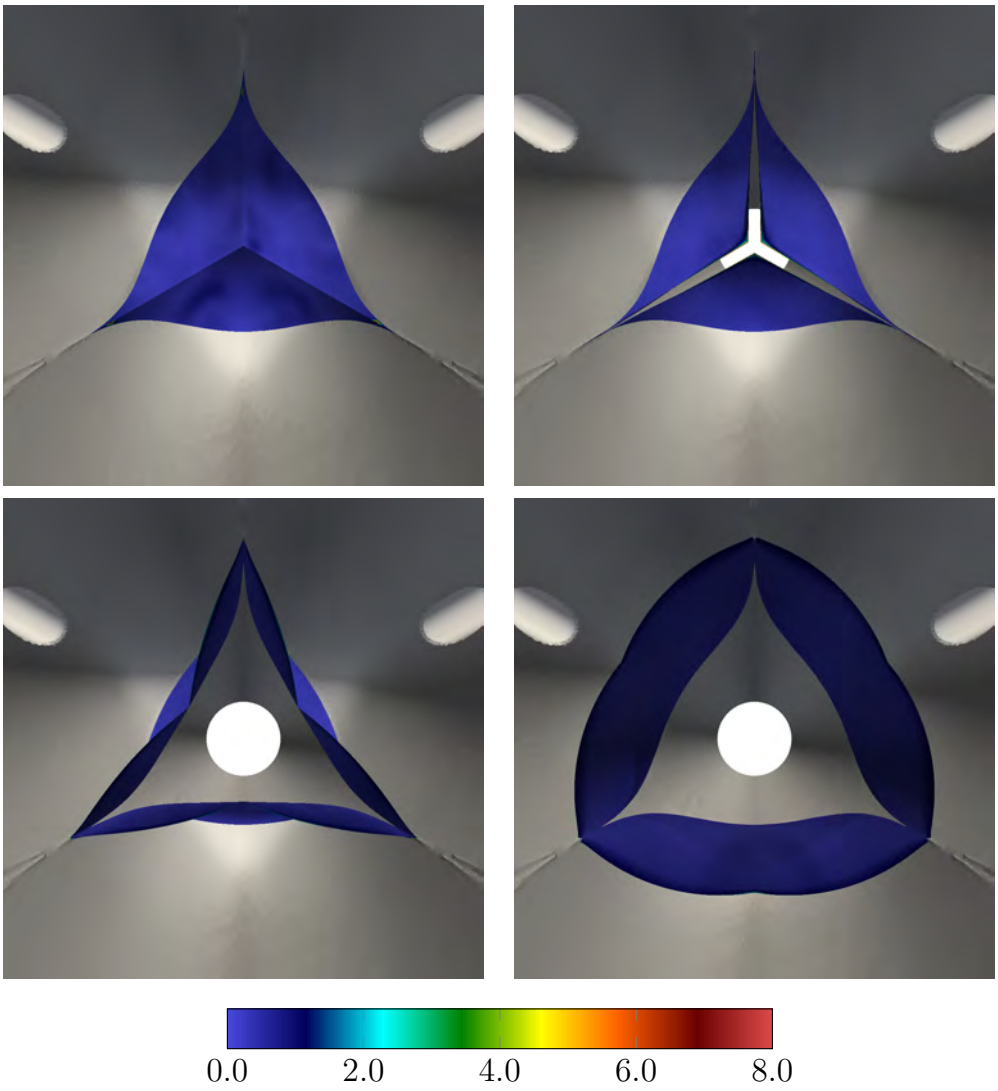

Figure 6.18: Aortic valve with coronary arteries. WSS (Pa) on the upper surface of the three leaflets at the same time instants as in Figure 6.14.

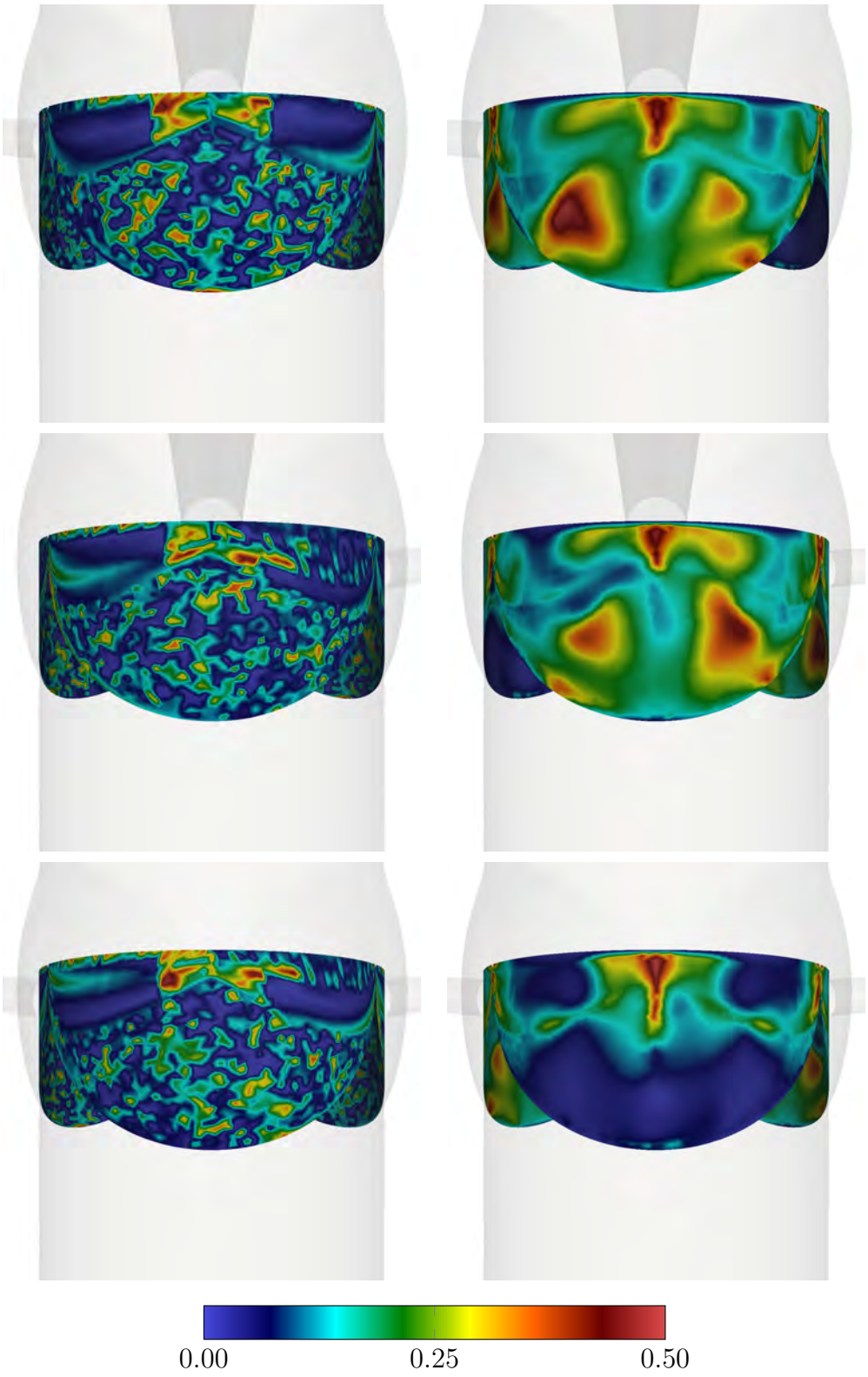

Figure 6.19: Aortic valve with coronary arteries. OSI on the lower (left) and upper (right) surfaces of the leaflets. The leaflets are in the fully open configuration, and left, right, and posterior aortic sinuses from top to bottom, respectively.

Chapter 7

Concluding Remarks

The material in this chapter is from [59]. We have presented an interface-tracking (moving-mesh) method that addresses the computational challenges associated with contact between moving solid surfaces and other cases of TC, including those in cardiovascular FSI, parachute FSI, and flapping-wing aerodynamics, with the primary target in cardiovascular FSI being heart valves. It is a new version of the DSD/SST method, and we call it ST-TC. It is based on special mesh generation and update, and a master–slave system that maintains the connectivity of the parent mesh when there is contact or other TC. This makes the method an efficient, practical alternative to using unstructured ST meshes, but without giving up on the accurate representation of the interface or consistent representation of the interface motion, even where there is contact or other TC. We explained the method with conceptual examples, and presented successful 2D and 3D computations with models representative of the classes of problems we are targeting. We are comfortable with concluding that the ST-TC method has the interface-tracking accuracy, the TC flexibility, and the computational practicality.

Bibliography

[1] Flytech wowwee dragonfly. http://rcmania.com/reviews/flytech-dragonfly-wowwee/, Sept. 2007.

[2] J. E. Akin, T. E. Tezduyar, and M. Ungor. Computation of flow problems with the mixed interface-tracking/interface-capturing technique (MITICT). *Computers & Fluids*, 36:2–11, 2007.

[3] 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*, 79:010905, 2012.

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

[5] 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*, 50:719–727, 2012.

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

[7] 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*, 38:310–322, 2006.

[8] 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*, 198:3534–3550, 2009.

[9] 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*, 65:207–235, 2011.

[10] 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*, 45:77–89, 2009.

[11] 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*, 65:236–253, 2011.

[12] 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(supp02):1230002, 2012.

[13] 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*, 9:481–498, 2010.

[14] 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*, 46:3–16, 2010.

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

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

[17] 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*, pages 336–355. Springer, 2006.

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

[19] M. A. Cruchaga, D. J. Celentano, and T. E. Tezduyar. A numerical model based on the Mixed Interface-Tracking/Interface-Capturing Technique (MITICT) for flows with fluid–solid and fluid–fluid interfaces. *International Journal for Numerical Methods in Fluids*, 54:1021–1030, 2007.

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

[21] B. E. Griffith, X. Luo, D. M. McQueen, and C. S. Peskin. Simulating the fluid dynamics of natural and prosthetic heart valves using the immersed boundary method. *International Journal of Applied Mechanics*, 1:137–177, 2009.

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

[23] 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.

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

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

[26] 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*, 29:329–349, 1981.

[27] 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*, 119:73–94, 1994.

[28] 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*, 190:321–332, 2000.

[29] K. Kamran, R. Rossi, E. Onate, and S. R. Idelsohn. A compressible Lagrangian framework for modeling the fluid–structure interaction in the underwater implosion of an aluminum cylinder. *Mathematical Models and Methods in Applied Sciences*, 23:339–367, 2013.

[30] 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*, 38:403–416, 2006.

[31] 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*, 23:249–272, 2013.

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

[33] 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*, pages 82–100. Springer, 2006.

[34] 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*, 43:73–80, 2008.

[35] 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*, 46:83–89, 2010.

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

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

[38] 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*, 50:729–741, 2012.

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

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

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

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

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

[44] S. Takase, K. Kashiyama, S. Tanaka, and T. E. Tezduyar. Space–time SUPG finite element computation of shallow-water flows with moving shorelines. *Computational Mechanics*, 48:293–306, 2011.

[45] 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*, 19:171–225, 2012.

[46] 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*, 50:835–854, 2012.

[47] 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*, 48:647–657, 2011.

[48] 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*, 50:743–760, 2012.

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

[50] 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*, 50:761–778, 2012.

[51] 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*, 23:307–338, 2013.

[52] 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*, 46:31–41, 2010.

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

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

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

[56] K. Takizawa and T. E. Tezduyar. Space–time fluid–structure interaction methods. *Mathematical Models and Methods in Applied Sciences*, 22(supp02):1230001, 2012.

[57] 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*, 52:1351–1364, 2013.

[58] K. Takizawa, T. E. Tezduyar, A. Buscher, and S. Asada. Space–time fluid mechanics computation of heart valve models. in preparation, 2014.

[59] K. Takizawa, T. E. Tezduyar, A. Buscher, and S. Asada. Space–time interface-tracking with topology change (ST-TC). *Computational Mechanics*, published online, DOI: 10.1007/s00466-013-0935-7, October 2013.

[60] K. Takizawa, T. E. Tezduyar, A. Buscher, and K. Montel. Space–time aerodynamic and FSI analysis of MAV flapping wings. in preparation, 2014.

[61] 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*, 53:1–15, 2014.

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

[63] T. Tezduyar, S. Aliabadi, M. Behr, A. Johnson, and S. Mittal. Parallel finite-element computation of 3D flows. *Computer*, 26(10):27–36, 1993.

[64] 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*, 191:717–726, 2001.

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

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

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

[68] 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(3):339–351, 1992.

[69] 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(3):353–371, 1992.

[70] 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*, 54:855–900, 2007.

[71] 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*, 195:2002–2027, 2006.

[72] 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*, 43:39–49, 2008.

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

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

[75] 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*, 64:1201–1218, 2010.

[76] 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*, 43:151–159, 2008.

[77] 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*, 46:43–52, 2010.

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

[79] 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*, 50:779–788, 2012.

[80] T. Wick. Coupling of fully Eulerian and arbitrary Lagrangian–Eulerian methods for fluid-structure interaction computations. *Computational Mechanics*, 52:1113–1124, 2013.

[81] 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*, 50:789–804, 2012.

[82] 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*, 23:273–305, 2013.