A three-dimensional computation of fluid-structure interaction in bioprosthetic heart valves using immersed membrane method

Xia, Guohua

2006

Xia, Guohua. (2006). A three-dimensional computation of fluid-structure interaction in bioprosthetic heart valves using immersed membrane method. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/5251>

<https://doi.org/10.32657/10356/5251>

Nanyang Technological University

Downloaded on 26 Apr 2023 20:04:45 SGT

A THREE-DIMENSIONAL COMPUTATION OF FLUID- STRUCTURE INTERACTION IN BIOPROSTHETIC HEART VALVES USING IMMERSED MEMBRANE METHOD

XIA GUOHUA

School of Mechanical and Aerospace Engineering
Nanyang Technological University

2006

ACKNOWLEDGMENTS

I would like to express my most sincere appreciation to my thesis advisors, Associate Professor Zhao Yong and Associate Professor Yeo Joon Hock, for their immense unconditional support, guidance, knowledge, patience, encouragement and discussion sessions throughout the research project. Most importantly, the author wishes to thank them for their kind understanding towards the author's problems. Associate Professor Zhao Yong helped me master the knowledge of all the numerical techniques involved in this work and patiently explained every problem to me. Associate Professor Yeo helped me a lot to obtain the knowledge of prosthetic heart valves and heart physiology.

I would like to thank all the members of my supervisors' research groups for their kind help and inspiring suggestions given to me. Special thanks also go to Dr. Tai Chin Hoe for his invaluable advices and suggestions. Deepest appreciations are also expressed to the staffs of Centre for Advanced Numerical Engineering Simulations (CANES) for their generous help and the computing facilities provided. I would like to acknowledge the Scholarship provided by Nanyang Technological University to support my Ph.D. research works.

CONTENTS

ACKNOWLEDGMENTS	I
ABSTRACT	VII
LIST OF KEYWORDS	X
LIST OF TABLES	XI
LIST OF FIGURES	XII
NOMENCLATURE	XVIII
PUBLICATIONS	XXI
CHAPTERS	
CHAPTER 1 GENERAL INTRODUCTION	1
1.1 Aortic Heart Valve	2
1.1.1 Anatomy of Aortic Heart Valve	2
1.1.2 Valve Motion in One Cardiac Cycle	6
1.2 Aortic Heart Valve Protheses	8
1.2.1 Aortic Heart Valve Diseases	8
1.2.2 Heart Valve Protheses	9
1.3 Objective and Scope of Research	13
1.4 Thesis Structure	14
CHAPTER 2 LITERATURE REVIEW	15
2.1 Numerical Investigations of Heart Valves.....	15

2.2 Computational Fluid Dynamics	22
2.2.1 Incompressible Flow	23
2.2.2 Moving Boundary	25
2.2.2.1 Eulerian Methods	25
2.2.2.1.1 Immersed Boundary Method	25
2.2.2.1.2 Fictitious Domain Method	26
2.2.2.1.3 Ghost Cell Finite Difference Method	27
2.2.2.1.4 Ghost Fluid Method	28
2.2.2.2 Arbitrary Lagrangian-Eulerian (ALE) Method	29
2.2.2.2.1 Mesh Regeneration Method	30
2.2.2.2.2 Deforming Mesh Method	31
2.2.2.3 Hybrid Methods	33
2.2.2.3.1 Chimera Mesh Method	34
2.2.3 Fluid-Structure Interaction	34
2.2.4 Parallel Computation	38
2.3 Computational Structural Dynamics	41
2.3.1 Finite Volume Method for Structural Analysis	41
2.3.2 Triangular Shell Element	43
2.4 Remarks	44
 CHAPTER 3 COMPUTATIONAL FLUID DYNAMICS	 46
3.1 Governing equations	47
3.2 Artificial Compressibility Method	52
3.3 Finite Volume Formulation	53
3.4 Upwind-Biased Interpolation	58
3.5 Upwind Characteristics-Based Method	59
3.6 Matrix-Free Implicit Dual Time-Stepping Algorithm	66
3.7 Time Integration	71
3.8 Convergence Acceleration Techniques	72
3.8.1 Local Time Stepping in Pseudo Time	72

3.8.2 Implicit Residual Smoothing	73
3.9 Boundary Conditions	74
CHAPTER 4 COMPUTATIOAL STRUCTURAL DYNMICS	75
4.1 Two-dimensional Immersed Membrane	76
4.1.1 Governing Equations	76
4.1.2 Discretization of The Structural Equations	80
4.2 Bi-leaflet Mechanical Heart Valve	82
4.3 Bioprosthetic Aortic Heart Valve	84
4.3.1 Governing Equations	84
4.3.2 Triangular Shell Elements	86
4.3.3 Finite-Volume Scheme	87
4.3.4 Discretization of Structural Equations	91
4.3.5 Boundary Conditions	92
CHAPTER 5 IMMERSED MEMBRANE METHOD AND FLUID- STRUCTURE INTERACTION	93
5.1 Multiple Ghost Fluid Nodes and One-sided Extrapolations	94
5.2 Immersed Membrane Method (IMM).	95
5.2.1 Inter-relationship between Fluid Mesh and Structural Mesh.	95
5.2.2 Intersection Points between Fluid Mesh and Structural Mesh.	99
5.2.3 Multiple Ghost Fluid Nodes and One-Sided Extrapolations	100
5.3 Solution procedure for Fluid-structure interaction	105
5.3.1 Fluid Forces on The Fluid-Structure Interface.	106
5.3.2 Solution Procedure.	109
5.4 Extension of IMM to Arbitrary Object shapes	112
CHAPTER 6 PARALLEL COMPUTING	113

6.1 Single Program Multiple Data Parallel Programming	114
6.1.1 Message Passing Interface (MPI)	115
6.1.2 Domain Decomposition	116
6.1.2.1 METIS.	117
6.1.2.2 Identification of Ghost Nodes and Overlapping Elements	120
6.1.2.3 Grid Files with Local Numbering	121
6.1.2.4 Data Structure for the Exchange of Ghost Nodes Variables and Gradients	123
6.2 Data Communication for Fluid-structure interaction and immersed membrane method.	127
6.2.1 Sending Information of Ghost Nodes and Variables	128
6.2.2 Data Communication for Computation of Fluid Forces	130
6.3 Measuring Performance	132
6.3.1 Speed-up	133
6.3.2 Parallel Efficiency	134
CHAPTER 7 NUMERICAL RESULTS AND ANALYSIS	135
7.1 Two-Dimensional Immersed Membrane Computations	136
7.1.1 Grid Convergence Study	136
7.1.2 Validation.	138
7.1.3 Moving Rigid Membrane.	139
7.1.4 Moving Flexible Membrane	140
7.1.5 Remarks	141
7.2 Serial Computation of Bi- Leaflet Mechanical Heart Valve	141
7.2.1 Model Properties of St. Jude Bi-leaflet Mechanical Heart Valve	142
7.2.2 Results for Opening Phase.	143
7.2.3 Results for Closing Phase	145
7.2.4 Remarks	147
7.3 Validation of Parallel Solver with IMM	148
7.3.1 Grid Convergence and Accuracy Study.	148

7.3.2 Error Analysis	151
7.4 Validation of Parallel Solver	152
7.4.1 Unsteady Flows at Reynolds Number of 100	154
7.4.2 Steady Flows Due to A Rotating Sphere	155
7.5 Parallel Computation of Bio-Prosthetic Heart Valve	157
7.5.1 Opening Phase	158
7.5.1.1 Computational Model	158
7.5.1.2 Material and Geometrical Properties of Valve Leaflet	159
7.5.1.3 Initial and Boundary Conditions	160
7.5.1.4 Results	162
7.5.2 Closing Phase	170
7.5.2.1 Computational Model	171
7.5.2.2 Results	174
7.5.3 Remarks	174
 CHAPTER 8 CONCLUSIONS AND FUTURE WORK	 244
8.1 Conclusions	245
8.1.1 Numerical Methods	245
8.1.2 Simulation of A Stented Bioprosthetic Heart Valve	248
8.2 Future Works	250
 REFERENCE	 253

ABSTRACT

This work aims to apply Computational Fluid Dynamics (CFD) and Computational Structural Dynamics (CSD) to simulate the motions of bioprosthetic heart valves and the flow field around them. The major problem in the numerical simulation of heart valves is fluid-structure interaction between the valve leaflets and the surrounding blood flows. And due to the large displacement of heart valves, the flow phenomena during the dynamic opening and closing phases are highly complex. In this work, a novel and efficient immersed membrane method (IMM) is proposed for the simulation of fluid-structure interaction phenomena in heart valves. In the IMM, a stationary Eulerian mesh is used for the computation of fluid domain, while the heart valves are treated as thin structures immersed in the flow field moving on top of the background Eulerian mesh, without changing the topology and geometry of the mesh. To obtain accurate fluid conditions, the concept of ghost nodes with ghost values is introduced to account for discontinuities in pressures and fluid stresses across the valves. Velocity continuity at the fluid-structure interface is directly enforced by one-sided extrapolations of fluid velocity using the ghost nodes and the extrapolated velocities are treated as a boundary condition for the fluid domain. The fluid-structure interaction is solved in a staggered and iterative manner: first, the governing equations are solved for the fluid domain, after which fluid forces are calculated along the fluid-structure interface. Then the structure is updated under the influence of fluid forces to a new position with a new configuration. Lastly, the immersed membrane method is employed to obtain appropriate boundary conditions at the fluid-structure interface for the fluid domain with the newly updated

ABSTRACT

structure configuration. The above procedure is repeated to the desired time instant. In fluid domain, a high-order characteristics-based finite-volume scheme is employed to solve the governing Navier-Stokes (NS) equations on an unstructured grid using an implicit matrix-free dual time-stepping technique. In the structural domain, the Newton's second law of motion is directly solved for each element under the influences of fluid forces. The implicit dual time-stepping technique is also implemented to improve the convergence rate. In order to improve computational efficiency, the solver is parallized using the Single Program Multiple Data (SPMD) programming paradigm and Message Passing Interface (MPI). The proposed IMM is suitable for various kinds of fluid-structure interaction problems. It can solve FSI phenomena involving both rigid bodies and flexible bodies. It can also be extended to solve the FSI problems for objects with arbitrary shapes, where surfaces of the immersed objects are treated as immersed membranes and the insides of the objects do not have fluids.

The efficiency and robustness of the time-accurate formulation of the current numerical algorithm is tested by computing the motions of two-dimensional models of immersed thin structures in incompressible viscous flows. A rigid membrane and a flexible membrane driven by influences of fluid-structure interactions are simulated by the developed solver. In addition, it is also applied to compute the motions of a 29 mm St. Jude bi-leaflet mechanical heart valve and the associated blood flows under physiological conditions for both opening and closing phases. Results show that the current solver is efficient in solving large displacement fluid-structure interaction problems. Error analysis is carried out and the result indicates that the immersed membrane method is at least second order accurate. The developed solver is validated

ABSTRACT

by computing flow around a rotating sphere. Both unsteady and steady results agree well with analytical solutions and numerical results obtained by other researchers.

Finally, the solver is used to compute the motions of a stented bioprosthetic aortic heart valve and the blood flows around it under physiological conditions for both opening and closing phases. The key features of this problem are three-dimensionality, fully coupled blood-leaflet interaction and physiologically realistic flow conditions. The simulated pulsatile flow exhibits a very complicated, inherently three-dimensional behavior. The motions of the valve leaflets for both the opening and closing phases are driven by the blood-leaflet interaction effect. The general motions of the valve can be summarized as follows: The valve leaflet opens and closes very fast. Most of the time in one cardiac cycle it ejects blood into the aorta when it is open and blood fills in heart when the valve is closed. The transvalvular pressure difference is the major driving for the motions of the valve leaflet. High fluid stresses exist near the free edges of valve leaflets, indicating that these regions are high-risk areas for valve tearing.

LIST OF KEYWORDS

Computational fluid dynamics; Computational structural dynamics; prosthetic heart valve; fluid-structure interaction; immersed membrane method; ghost nodes; one-sided extrapolation; ghost values; finite volume scheme; artificial compressibility method; high-resolution characteristics-based scheme; dual time stepping scheme; parallel computing; 29mm St. Jude Medical aortic bileaflet mechanical heart valve; physiological conditions; stented bioprosthetic aortic heart valve.

LIST OF TABLES

1.1	Dimensions of human aortic valve.....	5
5.1	Real fluid nodes and ghost fluid nodes for the computations of viscous fluxes and gradients of a 2D fluid element intersected by the immersed membrane.	102
6.1	Communication between core and <i>ghost</i> nodes for partition 1 as depicted in Figure 5.3.....	125
7.1	Properties of the vortices of the three grids.	137
7.2	Comparison of flow field with immersed membrane and flow field with internal boundary (L=20mm).	139
7.3	Parameters of a flexible membrane.....	140
7.4	Polar inflow angles at different Reynolds numbers compared with [157] and [160]	156
7.5	Geometrical and material properties of the flow and valve leaflet.....	160

LIST OF FIGURES

1.1	Human heart valves.....	3
1.2	Aortic heart valve.....	3
1.3(a)	Anatomy of aortic heart valve [2].....	4
1.3(b)	Cross section of aortic heart valve leaflet [2]	4
1.4	Typical dimensions of the rest human aortic heart valve	4
1.5	Plot of the aortic and left ventricular pressure curves and the associated flow rate curve during a cardiac cycle.....	6
1.6(a)	Starr-Edwards caged-ball valve.....	10
1.6(b)	Medtronic Hall tilting mono-disk valve.....	10
1.6(c)	St. Jude Medical bi-leaflet valve.....	10
1.7	Bioprosthetic valves (a) St. Jude stented bioprostheses.....	11
1.7	Bioprosthetic valves (b) St. Jude stentless bioprostheses.....	11
3.1	Construction of control volume within a tetrahedron for a node P.....	55
3.2	τ - ξ Co-ordinate.....	61
4.1	Forces and moments acting on an element of the immersed membrane.....	77
4.2	Computation of curvature for a structure element.....	79
4.3	Hemodynamic forces acting on the surface of the valve leaflet [45].....	83
4.4	Mesh for one leaflet of a bioprosthetic heart valve.....	87
4.5	Cell-vertex control volume around vertex O	88
4.6	Forces acting on one control volume.....	88

4.7	Computation of tension forces.....	89
5.1	Geometry of a 2D fluid domain and an immersed structure with its bounding box for search process.....	96
5.2(a)	Structure node within a triangular fluid element.....	97
5.2(b)	Structure node not within a triangular fluid element.....	98
5.3	Intersection point between the immersed membrane and fluid elements....	100
5.4	Real fluid nodes and ghost fluid nodes of a 2D fluid element intersected by the immersed membrane.....	101
5.5	Real fluid nodes and ghost fluid nodes of a 3D tetrahedron element intersected by the immersed membrane.....	102
5.6(a)	Velocity extrapolation for ghost node g_{23}	104
5.6(b)	Pressure extrapolation for ghost node g_{23}	104
5.7	Relation between the structure domain and the fluid domain in fluid-structure interaction.....	106
5.8	Extrapolation of fluid pressure to FSI interface.....	107
5.9	Flowchart for solution of fluid-structure interaction.....	111
5.10	IMM for arbitrary object shapes.....	112
6.1	Flowchart depicting geometric domain decomposition parallelization strategy using MPI	113
6.2	General structure for an MPI parallel program.....	116
6.3	Flowchart depicting algorithm developed in program “ <i>Partition</i> ”	119
6.4	Mesh decomposed into 4 partitions showing each sub-domain extended with a layer of <i>ghost</i> nodes and overlapping elements.....	122
6.5	Cavity mesh decomposed into various partitions (a) original mesh; (b) 2 partitions; (c) 4 partitions; (d) 8 partitions; (e) 16 partitions; (f) 32	

partitions.....	123
7.1 Geometry of the 2D channel model with an immersed membrane.....	176
7.2 Flow field of grid (I) with 2,000 nodes.....	176
7.3 Flow field of grid (II) with 18,000 nodes.....	176
7.4 Flow field of grid (III) with 32,000 nodes.....	176
7.5 (a) X-component velocity profiles at $X=1$	177
7.5 (b) Convergence history plot for three different mesh densities.....	177
7.6 Computational model for validation of steady flow ($L=20$ mm)	178
7.7(a) Close-up view of immersed membrane and surrounding mesh.....	178
7.7(b) Close-up of internal boundary and surrounding mesh.....	178
7.8 Convergence history of immersed membrane and internal boundary.....	179
7.9(a) Flow field with an immersed membrane computed by the immersed membrane method.....	179
7.9(b) Flow field with an internal boundary computed by solver DELTINKE...	179
7.10(a) Comparison of X-component velocity profiles at $X=0$	180
7.10(b) Comparison of X-component velocity profiles at $X=1$	180
7.11 Motion of a rigid membrane, thickness of $0.5L/100$, driven by a constant inflow at $Re=100$	182
7.12 Steady flow with an immersed flexible membrane.....	182
7.13 Motion of an immersed flexible membrane, driven by constant viscous inflow at $Re=100$	184
7.14(a) Fully closed configuration of St. Jude Bi-leaflet MHV.....	184
7.14(b) Fully opened configuration of St. Jude Bi-leaflet MHV.....	185
7.15 Time-dependent inlet velocity profile for opening and closing phases.....	185
7.16(a) Geometry of the vessel for simulation of St. Jude 29mm Bi-leaflet MHV	186

7.16(b)	Geometry and position of the valve leaflet.....	186
7.17	Pressure contours of the x - y plane at $z=0$ during opening phase at different time instants.....	187
7.18	Velocity vectors of the x - y plane at $z=0$ and volume stream around the valve leaflet during opening phase at different time instants.....	188
7.19	Pressure and fluid shear stress distributions on leaflet surface for opening phase.	191
7.20	Pressure contours of the x - y plane at $z=0$ around valve leaflets at different time instants of closing phase.....	191
7.21	Velocity vector of the x - y plane at $z=0$ and volume stream around the valve leaflet during closing phase at different time instants	195
7.22	Pressure and fluid shear stress distributions on leaflet surface for closing phase.	197
7.23	Computational domain for oscillating sphere.....	198
7.24	Grid generation for consecutively finer meshes.....	198
7.25	Mesh partition for grid convergence study.....	199
7.26	Flow fields at the moment of the end of one oscillation time period.....	201
7.27	Convergence history of L_1 , L_2 and L_∞ error norms of x -component velocity for the three coarser grids with respect to the finest grid.....	201
7.28	Computational domain for rotating sphere.....	202
7.29	Mesh partition for steady flows induced by a rotating sphere.....	202
7.30	Evolution history for flow field due to a rotating sphere at $Re = 100$	204
7.31	Unsteady results reported by Gilmonov [157] for flow due to a rotating sphere at $Re = 100$	205
7.32	Steady flow due to rotating sphere at different Reynolds numbers.....	206

7.33	Numerical results by Gilmonov [157] of steady flow due to rotating sphere at different Reynolds numbers.....	207
7.34	Computational domain for opening phase.....	208
7.35	Configuration of a closed and unloaded porcine aortic valve leaflet	209
7.36	Mesh for a bioprosthetic aortic valve leaflet.....	209
7.37(a)	Fluid mesh, immersed leaflet mesh for opening phase.....	210
7.37(b)	Fluid mesh extracted from the center of $y - z$ plane with the immersed leaflet mesh.....	210
7.38	Fluid pressure and stress distributions on a valve leaflet at different time instants for opening phase.....	214
7.39	Pressure contours of $y = 0$ plane at different time instants for opening phase.....	218
7.40	Velocity vector plots at different time instants for opening phase.....	225
7.41	Valve orifice area of opening phase.	225
7.42	Extracted positions of velocity profiles.....	226
7.43	Velocity profiles extracted for opening phase.....	228
7.44	Perspective view of the computational model for closing phase.....	229
7.45	Pressure contours of $y = 0$ plane at different time instants for closing phase.....	221
7.46	Fluid pressure and stress distribution on valve leaflet at different time instants for opening phase.....	234
7.47	Velocity vector plots at different time instants for opening phase.....	239
7.48	Positions for velocity extraction of plane $y = 0$	240
7.49	Velocity profiles extracted for opening phase.....	241
7.50	Valve orifice area for closing phase.....	242

7.51	Numerical results excerpted from De Hart's thesis [3] for comparison...	243
------	---	-----

NOMENCLATURE

ACM	Artificial Compressibility Method.
BHV	Bioprosthetic Heart Valve.
C	Preconditioning matrix.
c	Speed of sound.
CFD	Computational Fluid Dynamics.
CFL	Courant-Friedrichs-Lewy number.
CSD	Computational Structural Dynamics
E_p	Efficiency for parallel computing.
E	Young's modulus
FSI	Fluid-Structure Interaction
$\vec{F}_v$	Viscous flux vectors.
$\vec{F}_c$	Convective flux vectors.
$\vec{F}_c^+$	Positive term of flux vector of convected quantities.
$\vec{F}_c^-$	Negative term of flux vector of convected quantities.
IMM	Immersed Membrane Method.
K	Unit matrix.
L	Characteristic length.
MHV	Mechanical Heart Valve.
MPI	Message Passing Interface.
Pr	Prandtl number.
$R(W_p)$	Flux residual.

NOMENCLATURE

$\tilde{\mathbf{R}}(\mathbf{W}_p^{n+1})$	Modified flux residual in implicit time stepping.
Re	Reynolds number.
$\bar{\mathbf{r}}$	Distance vector which points from the pivot axis of the mechanical heart valve to the node under consideration.
S_p	Speed-up for parallel computing.
SPMD	Single Program Multiple Data.
S_t	Strouhal number.
T	Period for a cardiac cycle.
t	Time.
$\Delta\tau$	Pseudo-time step.
Δt	Real time step.
t_p	Total simulation wall-clock time.
t_{cpu}	CPU time.
t_{comm}	Communication time.
t_{comp}	Computation time.
t_{idle}	Idling time.
$t_{\text{comm}}^{\text{FFV}}$	Communication time for exchanging of flow field variables.
$t_{\text{comm}}^{\text{FT}}$	Communication time for exchanging of forcing terms.
$t_{\text{comm}}^{\text{NG}}$	Communication time for exchanging of nodal gradients.
U_∞	Free-stream velocity.
u, v, w	Velocity component in the x, y and z direction, respectively.
$\mathbf{W}$	Vector of conserved flow variables.
$\mathbf{W}_p$	Vector of conserved flow variables at node P.
x, y, z	Cartesian co-ordinates.

NOMENCLATURE

Greek Symbols

ρ	Density.
ρ_{∞}	Free stream density.
μ	Dynamic viscosity.
τ	Pseudo-time.
α_{RK}	Stage coefficients for Runge-Kutta time integration.
ε	Smoothing coefficient.
β	Artificial compressibility parameter.
κ	Weight used in MUSCL interpolation.
ν	Poisson's ratio

PUBLICATIONS

The following papers are related to the present research:

- (1) Xia G.H., Zhao Y. and Yeo J. H., Numerical Simulation of 3D Fluid-Structure Interaction Using An Immersed Membrane Method, Modern Physics Letters B, Vol. 19 No. 28-29, 2005.
- (2) X. Lv, Y. Zhao, X. Y. Huang, G. H. Xia and Z. J. Wang, An Efficient Parallel/Unstructured-Multigrid Preconditioned Implicit Method for Simulating 3D Unsteady Compressible Flows with Arbitrarily Moving Objects, (in press), Journal of Computational Physics, 2005.
- (3) Xia G.H., Zhao Y. and Yeo J. H., Numerical Simulation of 3D Fluid-Structure Interaction Using An Immersed Membrane Method, International Symposium on Physics of Fluids, Huangshan, China, 9-12 June, 2005.
- (4) Xia G.H., Zhao Y. and Yeo J. H., An Immersed Membrane Method For Simulation of Fluid-Structure Interaction in Bio-Fluid Flows, 1st International BioEngineering Conference (**IBEC 2004**), Singapore, 08-10, September, 2004.

CHAPTER ONE

GENERAL INTRODUCTION

This chapter presents the background information for current research work. In section one, a short introduction is given about aortic heart valve and its motion in a cardiac cycle, followed by a summary of aortic valve diseases and prostheses in section two. The objective and scope of this research are defined in section three, followed by the structure of thesis given in the last section for clarity.

1.1 AORTIC HEART VALVE

1.1.1 Anatomy of Aortic Heart Valve

A human heart has four valves regulating the internal blood flow. They are mitral valve, aortic valve, pulmonary valve and tricuspid valve as shown in Figure 1.1. The aortic heart valve is located at the outlet of the left ventricle just prior to the ascending aorta. The natural aortic heart valve comprises of three highly flexible leaflets and three sinuses, see Figure 1.2.

An aortic valve leaflet has two functional parts: non-coaptation part and coaptation part. The area near the free edge is called coaptation. When the valve closes, the neighboring leaflets snap together and the coaptation parts coapt against each other, sealing firmly to provide a safety margin for the valve to close without regurgitation. The remaining area of valve leaflet is the non-coaptation part which bears the load of aortic pressure. The aortic valve leaflet is a complex multi-layer structure as depicted in Figure 1.3(a). The two main structural layers are the fibrosa that covers the whole cusp and the ventricularis that covers all but the coaptation region. In cross-section, the leaflet has three distinct layers, the fibrosa, spongiosa and ventricularis as shown in Figure 1.3(b). The fibrosa is on the top surface of the leaflet if looking down from the aorta, with the ventricularis on the bottom. The fibrosa is considered to be the primary structural layer and contains a large amount of collagen organized into large bundles and fibers which are oriented in a circumferential direction. This directionality results in a structure that is considerably stiffer in the circumferential direction than the radial. A matrix of elastin surrounds the collagen bundles to maintain the valve's microstructure during unloading. The ventricularis, while less organized than the fibrosa, contains a

significant amount of collagen and elastin. However, because the collagen is not oriented in any specific direction, it tends to be less stiff than the ventricularis. The spongiosa is primarily water, but also contains glycosaminoglycans (GAG's) and small amounts of collagen and elastin which connect the fibrosa and ventricularis together. The sinuses are dilated pouches of the aortic root, see Figure 1.2, which are thinner than the aortic wall. It is generally believed that this cavity plays an important role in the mechanism of valve closure [1].

Figure 1.1 Human heart valves

(a) [2]

(b)

Figure 1.2 Aortic heart valve

Figure 1.3 (a) Anatomy of aortic heart valve leaflet. (b) Cross section of aortic heart valve leaflet

Figure 1.4 Typical dimensions of the rest human aortic heart valve extracted from Ref [1]

The available measured geometry of aortic heart valve varies according to different measuring techniques the researchers used. However, the normalized dimensions of valves, in a single species or in a group of species, do not vary greatly.

Variations in the dimensions appear to be in the range of natural variations expected in any given species. The typical geometry of human aortic valve is sketched in Figure 1.4 with some important values listed in Table 1.1. All the values are normalized by 12mm. Although the three leaflets are not precisely identical, it is assumed that they are similar enough to permit a general description of valve with tri-leaflet symmetry [3].

R_{valve}	0.95-1.17	H_{com}	0.5-0.7
R_{aorta}	0.7-1.1	T_{wall}	0.086
H_{valve}	1.1-1.4	T_{valve}	0.017
D_{sinus}	0.2-0.7	α	28°-34°
H_{coapt}	0.34	θ	20°-22°
H_{sinus}	1.2-2.0		

Table 1.1 Dimensions of human aortic valve

R_{valve} = radius of valve, R_{aorta} = radius of aorta, H_{valve} = height of valve, D_{sinus} = depth of sinus, H_{coapt} = height of coaptation, H_{sinus} = height of sinus, H_{com} = height of commissure, T_{wall} = thickness of aorta wall, T_{valve} = thickness of valve, α = free edge angle of the leaflet, θ = bottom surface angle of the leaflet.

1.1.2 Valve Motion in One Cardiac Cycle

The cardiac cycle can be divided into a systolic and diastolic interval, as shown in Figure 1.5, as extracted from Ref. [4]. During diastole, the relaxation phase of the cardiac cycle, blood fills the left ventricle from atrium while the aortic valve remains closed. During systole, the pumping phase of the cardiac cycle starts by contracting the left ventricle and thus increases the intraventricular pressure. The leaflets start bulge towards the aorta just before ventricular ejection begins. The valve reaches the fully open position when the peak flow in the ascending aorta has reached 75% of its maximum [5]. Although the Reynolds number reaches up to 4,500 at peak flow, it

Figure 1.5 Plot of the aortic and left ventricular pressure curves and the associated flow rate curve during a cardiac cycle [4].

appears that in the case of properly working heart valves, the blood flow remains laminar [6]. Only when a valve is malfunctioning, some turbulence might be expected. Near the end of systole, valve gradually closes due to the deceleration of the forward aortic flow, resulting in about 80% valve closure at the moment of zero flow in the ascending aorta at the end of the systole [1]. The flow phenomena in closing phase are far more complex than the opening phase. In this phase vortices are initiated in the sinus cavities, which are thought to instigate early valve closure. Followed by the adverse pressure gradient during flow deceleration, a small reverse flow finally completes the closure. Being a thin and highly flexible membrane-like structure, the aortic valve can not withstand any significant pressure difference during the opening and closing phases of the systole. While in the course of diastole the leaflets have to withstand a slowly varying but considerable pressure load (see in Figure 1.5).

Generally, the aortic valve opens and closes very rapidly at the beginning and end of the systole which takes 17 to 20ms. The valve motion may be pictured as follows: a sudden opening of the valve, then only a little movement while the valve remains open, a sudden closure of the valve, and then almost no movement while the valve remains closed. Little time is spent in the opening and closing process during a cardiac cycle, most of the time is spent in filling the heart when the valve is closed and ejecting blood when the valve is open. This magnet-like to-and-from motion is the most prominent feature of the aortic valve in action [1].

1.2 AORTIC HEART VALVE PROTHESES

1.2.1 Aortic Heart Valve Diseases

The valves open when the upstream pressure is greater than the downstream pressure, and close tightly when the downstream pressure is higher than the upstream pressure. They work under pressure differences between 80mmHg and 100mmHg at a very high frequency. And with their thin-walled structures, heart valves are prone to diseases. Although diseases may happen to any valves, the aortic valve diseases occur most frequently and with more serious consequences, since the aortic valve works under the highest pressure difference and controls the blood pumped to the whole body. A diseased valve may cause restriction of blood flow (being stenotic) or regurgitant flow of blood (being incompetent), or both [1].

When the aortic valve becomes stenotic, resistance to systolic ejection occurs and a systolic pressure gradient develops between the left ventricle and the aorta. Stenotic aortic valves have a decreased aperture that leads to a progressive increase in left ventricular systolic pressure. This leads to pressure overload in the left ventricle. Over time, the pressure overload causes an increase in ventricular wall thickness (concentric hypertrophy). Eventually the left ventricle will dilate and reduce compliance of heart. Consequently an increase in left ventricular end-diastolic pressure will happen, which is increased further by a rise in atrial systolic pressure. This ever-developing pressure overload eventually leads to myocardial decompensation. The contractility of the myocardium diminishes, which leads to a decrease in cardiac output. The elevated left ventricular end-diastolic pressure causes a corresponding increase in pulmonary capillary arterial pressures and a decrease in ejection fraction and cardiac output.

Ultimately, congestive heart failure (CHF) develops. Aortic regurgitation is the diastolic flow of blood from the aorta into the left ventricle. Regurgitation is due to incompetence of the aortic valve or any disturbance of the valvular apparatus (e.g. leaflets, annulus of the aorta) resulting in diastolic flow of blood into the left ventricular chamber. Diastolic reflux through the aortic valve can lead to left ventricular volume overload, which increases systolic stroke volume and consequently increases pulse pressure.

Some degree of stenosis and regurgitation is allowed without compromising health. However, when valve diseases severely affect heart function, valve replacements are required using prosthetic valve devices. Although prosthetic valve devices have been on the market for several decades, the natural aortic valve is unique in its accomplishments, to date, no one can achieve the same efficiency and durability of the natural aortic valve. Therefore, efforts and knowledge are desired for more successful designs of prosthetic valves.

1.2.2 Heart Valve Prostheses

The development and clinical application of heart valve prostheses could be one of the most prominent fields of biomedical engineering and one of the focuses in academic research. In the past four decades, progress and collaboration among different disciplines related to the development of artificial heart valves, such as medicine, macromolecular chemistry, mechanical engineering and computer sciences, etc., have greatly contributed to the successive progress in this field. Currently, there mainly exist two types of prosthetic heart valves: mechanical heart valves and bioprosthetic heart valves.

Commercially available mechanical heart valves include caged-ball valves, mono-leaflet and bi-leaflet valves, see Figure 1.6. Bi-leaflet mechanical heart valves are the most commonly used mechanical prosthetic heart valves. They are notable mainly for providing the closest approximation to central flow achieved in a mechanical heart valve. Mechanical heart valves are generally durable, but their flow pattern disruptions can lead to thrombus formation, a fact that requires valve recipients to take anticoagulants, or blood-clot inhibitors, for the rest of their lives. Also, mechanical stresses can damage blood cells and bacterial infections can lead to further heart damage. If these valves could be made on non-thrombogenic in future, their use could become a much more desirable treatment for diseased heart valves in human beings.

(a)

(b)

(c)

Figure 1.6 (a) Starr-Edwards caged-ball valve; (b) Medtronic Hall tilting mono-disk valve; (c) St. Jude Medical bi-leaflet valve.

Alternatively, bioprosthetic valves have more resemblance to the natural heart valves as shown in Figure 1.7. They are made of chemically treated biological tissues and have better hemodynamic performance than mechanical heart valves. Their central flow characteristics enhance hemodynamic efficiency and their biological surfaces

enhance thromboresistance compared with mechanical prostheses. However, bioprostheses are haunted by their short durability. Most bioprostheses fail by calcific degeneration or tissue degeneration caused by mechanical stress in a relatively short period. Future research to improve bioprostheses will be focused on long durability and tissue strength.

Figure 1.7 Bioprosthetic valves (a) St. Jude stented bioprostheses;
(b) St. Jude stentless bioprostheses.

Besides these available prosthetic types there is a growing interest in the development of synthetic valves. They allow more freedom in optimizing the design in terms of durability and valve hemodynamics. Numerical studies by De hart et.al [7] and Cacciola [8] have shown that leaflet fiber-reinforcement significantly reduces peak stresses at areas that are susceptible to degeneration. Another particularly exciting in concept, yet early in practice are tissue engineered heart valves. Recently, there is considerable activity in basic research and clinical investigation to develop this technique in which an anatomically appropriate construct containing cells seeded on a resorbable scaffold is fabricated *in vitro*. Stimulated and guided by appropriate

mechanical and chemical signals, biological remodeling is intended to recapitulate normal function architecture [9-10]. Moreover, remodeling of the tissue is inclined to proceed after implantation of the construct. Possibly, these features make tissue-engineered valves desirable substitutes in the future.

Generally, ideal heart valve prosthesis should meet the following requirements [11]:

- (1) The physical structure of the heart valve prosthesis should conform to the heart structure (i.e. the size and shape of the prosthesis should not interfere with the cardiac function);
- (2) Minimum flow resistance to prevent a significant pressure drop across the valve;
- (3) Minimize reverse flow for valve closure;
- (4) Resistance to mechanical and structural wear;
- (5) Durability and maintaining implanted valve function throughout the patient's lifetime;
- (6) Minimum trauma to blood cell and the endothelial tissue surrounding the valve;
- (7) Low probability for thromboembolic complications without the use of anti-coagulants;
- (8) Material compatibility with blood.

The perfect heart valve prosthesis design would have contained all the features of normal human valves, including resistance to disease. To date, no known substitute device can meet all these requirements and almost ideal heart valve prosthesis has not yet been designed. However, the majority of prosthetic valves in current clinical use do provide patients reasonable hope for several years of event-free survival, even though they do not resemble the natural valves.

1.3 OBJECTIVE AND SCOPE OF RESEARCH

From the previous analysis, it is clear that there is a strong need for an aortic valve prosthesis which can mimic the function of the natural valve with such an efficiency and durability that postoperative treatment becomes redundant. At this stage, the development of improved prostheses involves many expensive and time-consuming experiments to gain the knowledge and understanding of such a complex system. And the experiments are always limited to regions of the flow field around valve leaflets. The delicate and strong interaction between the surrounding blood and valve leaflets requires that both mechanical and hemodynamical factors of the valve must be considered in experiments, which are important to the performance and durability of a prosthetic valve.

Numerical simulation provides a promising way to develop improved valve prostheses by significantly reducing the number of experiments required and giving insights into system responses that can not or hardly obtained by experiments. The objective of this work is to develop an efficient high performance computational flow solver for numerical simulation of blood flows in prosthetic heart valves under pulsatile flow and blood-leaflet interaction. In general, this solver is able to solve the problems associated with natural valves and currently used prosthetic devices. If applying real material properties and appropriate boundary conditions, the solver can analyse the performance of different valve designs under different loading conditions, so it can be used as a design tool for valve prosthesis. An immersed membrane method (IMM) is proposed to solve the fluid-structure interaction problem, which allows for sharp change in fluid conditions across the immersed structure without mesh update, thus improving the efficiency and accuracy of the numerical models. In this work, all the developed

numerical methods and algorithms are validated by comparing the results obtained with a wide range of well-documented published results as well as experimental measurements wherever available. This is to demonstrate the accuracy, effectiveness and robustness of the solver in order to study the flow around a bioprosthetic aortic heart valve for both its opening and closing phases under physiological condition and blood-leaflet interaction.

1.4 THESIS STRUCTURE

In this thesis, Chapter 2 reviews some of the work relevant to this work contributed by researchers of this field. Chapter 3 describes the mathematical models for unsteady incompressible Navier-Stokes equations and the numerical approach employed in the solver. Chapter 4 describes the structural dynamics for immersed structures involved in this work. Chapter 5 describes the immersed membrane method and the solution strategy for fluid-structure interaction problem based on IMM. Chapter 6 describes the implementation of Single Program Multiple Data (SPMD) parallel computing for the fluid solver, the algorithms for identifying *ghost* nodes and overlapping elements and the data communication for fluid-structure interaction. Chapter 7 presents the validation of the developed numerical schemes and algorithms against well-documented published results. The computed results for various flow problems are showcased and discussed. And the results and discussion for the computations of a bioprosthetic aortic heart valve for both opening and closing phases is also presented in this chapter. Finally, Chapter 8 gives some concluding remarks and suggestions for possible future work. (The contents of all these chapters are based on separate articles, i.e. Refs [151, 152, 171,172], which are published or have been submitted for publication.)

CHAPTER TWO

LITERATURE REVIEW

This chapter reviews relevant published research works. First, the previous numerical simulations of heart valves are reviewed in section one, which followed by reviews on various techniques of incompressible flows and moving boundaries, research works about fluid-structure interaction phenomena and parallel computing involved in computational fluid dynamics in section two. Previous works on development of computational structural dynamics based on finite volume scheme and works about structural analysis by using triangular shell elements are reviewed in section three. The chapter is concluded with remarks in section four.

2.1 NUMERICAL INVESTIGATIONS OF HEART VALVES

The successful development of prosthetic heart valves requires not only the knowledge of the behavior of valves but also the associated blood flow field around them. Many *in-vitro* experimental studies provided useful field information on different heart valves; however, the physical and cost limitation of experimental investigations make it difficult to give a full and detailed description of the complex transvalvular flow phenomena. Numerical simulation provides an important alternative to the study of heart valves. Researchers can obtain details of the heart valve at relatively low costs by numerical simulations, and these results always have higher resolution than experimental results. However, numerical study depends much on the mathematical model and the computational method used, and it is limited by the performance of the computing facilities available. Using these two methods together, researchers have recently achieved significant progresses in the study of heart valves.

Under real flow conditions, the flow is pulsatile and the leaflet movement induced by blood-leaflet interaction should be taken into consideration. It is difficult to solve these conditions due to the overwhelming computational resources required. And it remains as one of the challenging problems till now. In his Ph.D. thesis, Peskin [26] studied blood and heart-valve-leaflet interaction by solving the Navier-Stokes equations on a structured mesh with the finite difference method. In his study, the heart valve was assumed as to be neutrally buoyant and force generating structure immersed in viscous incompressible fluid. The immersed boundary (IB) method was proposed to track the moving boundary and to calculate the blood-cardiac-tissue interaction. The structure was represented by a “link formalism”. The structure was discretized into a series of

points, which are connected by links. The physical properties of the structure are stored on these points. The length-tension relation for each link was fixed for the passive valve leaflet and varying for active heart muscle. An analogue of the delta function was used to interpolate the fluid velocity field to points of the cardiac tissue and also to spread the cardiac forces onto the fluid mesh. The IB method proposed by Peskin was remarkable and up to the present date, this method is still used by many researchers [27-32]. After Peskin, many researchers published their researches of flow fields around heart valves. In 1975, Underwood [33] studied steady, axisymmetric flow through a disk-type prosthetic heart valve. He solved the NS equations of two simple flow fields at different Reynolds numbers by the finite difference method. Then he compared these results at different Reynolds numbers. Huang [34] investigated the vortex shedding in a tilting disk valve in 1994. Shim [35] studied 3-D flow field around a Björk-Shiley valve in 1997. King *et al.* [36] studied effects of valve leaflet opening angles on the flow field in bi-leaflet valves in 1997. Many other researchers also studied the similar cases [37-40]. Most of these work mentioned above concentrated on steady flows in heart valves, little was done on the simulations of real flow fields in heart valves till recent years.

Chen and Zhang [41-43] studied the two-dimensional blood-leaflet interaction problem with finite element method in 1995 and 1996. Arbitrary Lagrangian-Eulerian (ALE) method was used to calculate the moving boundary in their work. For fluid-structure interaction calculation, the governing equations for the fluid and structure domains were coupled to form a matrix equation, and the equation was solved with the Newmark implicit integration scheme. With these methods, they studied the opening and closing processes of artificial heart valves and compared the opening and closing processes of St. Jude valve and a downstream direction valve. They analyzed the blood-

leaflet interaction system subjected to a step pressure when the valve was at closing position. In their studies, the ALE method was always valid only for small mesh deformation, and the computational mesh was too simple.

Recently, Shi [26] has studied the pulsatile flow field around a St. Jude Medical size 29mm bi-leaflet MHV for both the opening and closing phases under blood-leaflet interaction. In his work, he used a marker method to track the moving boundary and applied it to blood-leaflet interaction computation. In his marker method, the velocity of the moving leaflet is transferred to the ambient fluid as boundary condition, and the fluid forces acting on the structure are integrated with the corresponding distance to generate the driving moment of leaflet rotations. As overwhelming computing resources are required in the CFD calculation, parallel processing technique is used in his work. The work gave great insights into the real flow fields in heart valves under pulsatile flow conditions with dynamic blood-leaflet interaction. However, the boundary of the leaflets in the fluid domain is defined by a devised geometric equation, which is based on the assumption that the leaflet is of semi-elliptical shape. This geometric equation might not be the true representation of the leaflets and this will lead to inaccurate flow fields around the leaflets. For parallel computation, only a maximum of three processors can be employed and he has partitioned the computational mesh into three sub-domains for the computation of both the opening and closing processes. In the computed results obtained for both the opening and closing processes, there are no significant sign of vortices behind the leaflets. Tai [45] later on proposed his immersed object method (IOM) with overlapping grids (OG) for the simulation of unsteady incompressible viscous flows around moving rigid bodies. Instead of just using the direct forcing method along the object boundary, the fluid velocity within the whole object is set to the

corresponding local object velocity by using a forcing term in the momentum equations. Compared with the other existing methods, this direct approach has the advantage that one does not need to perform elaborate interpolation to impose the boundary condition. For high Reynolds-number flows where the boundary layer is thin and has to be accurately captured, an OG surrounding the object can be added and the Eulerian solution can be transferred to the boundary of the OG to perform further calculation to obtain the fined solution and as well as the lift and drag coefficients. The closing and opening phases of a St. Jude bi-leaflet mechanical valve under physiological conditions were studied by parallel computation on 16 processes. With improved accuracy and enough mesh density, apparent vortices were observed to be shedding from the valve leaflets. The work proposed by Tai has achieved a great improvement towards the heart valve research under pulsatile flow conditions and dynamic blood-leaflet interaction. Recently, Xia [151] proposed an immersed membrane method to study the opening and closing phases of a bi-leaflet mechanical heart valve. The immersed membrane method is accurate in capturing the sharp changes of fluid conditions across the heart valves. And it is also capable to obtain the information of fluid pressure and stress distribution on the leaflet surfaces.

CFD Research Corp. (CFDRC), a consulting company devoted to CFD research and consultancy, marketed its new software called CFD-ACE [46]. CFDRC claimed that their CFD-ACE could efficiently tackle the fluid-structure interaction. They have cooperated with University of Miami and Georgia Institute of Technology to carry out research on pulsatile flows in the heart and heart valves. They calculated the squeezing flow for MHV cavitations in 1994 and 1997 [47,48]. In this calculation, the leaflet was supposed to move according to a prescribed change based on experimental results. In

1996, they calculated squeeze flow in tilting disc MHV with fluid-structure interaction [49]. In 1997, they carried out three-dimensional coupled fluid-structure interaction simulation of pericardial bio-prosthetic aortic valve function [50]. In their calculation, the moving boundary was always calculated by using deforming mesh method, which was just valid for small mesh deformation cases. With the introduction of Chimera mesh method for the moving boundary in their work, accuracy of their calculation can be expected to improve.

Professor Hwang's research group at University of Miami investigated the unsteady flows around MHVs [51, 52]. Gao *et al.* [51] studied the unsteady flow field in the hinge position of On-X, St. Jude Medical and Medtronic Parallel bi-leaflet MHVs with a commercial software called CFD2000 STORM. In the study a user defined function was added to the software in which the leaflet movement was obtained from experimental results. Yun [52] developed a code to calculate the unsteady flow in a mono-leaflet MHV in her Ph.D. work. In the developed code, the valve region was separated from the upstream and downstream regions. Static meshes were generated for the upstream and downstream regions, while mesh for the valve region was generated and allowed to rotate along the pivot of the leaflet. During calculation information exchange between the moving mesh and the static mesh was achieved through interpolation. The movement of the leaflet could be assigned either from experimental results or through fluid-structure interaction. The proposed method was useful and interesting, but it was not properly implemented due to maybe the limitation of the computing facility, and the information exchange between two meshes may also cause errors. As a result, the results were greatly compromised. Yun compared the results with the moving boundary and that with only fixed boundary. Although she claimed that

great difference was observed between them, actually the difference was too minor to be observed. Professor Yoganathan's research group in Georgia Institute of Technology has also used the commercial software CFD2000 STORM to study micro-structural flow within the hinges of a bi-leaflet MHV [53]. They developed a user-defined subroutine for use with CFD2000 STORM for computing the three-dimensional moving boundary. The movement of the leaflets was prescribed based on experimental results.

Kiris *et. al* [54] calculated flow in an artificial heart with moving boundary in 1997. In this artificial heart, two tilting-disk valves are mounted in mitral and aorta valve positions. The rotation of the aortic valve leaflets was obtained by approximating the mitral valve rotation, with a phase difference. In order to handle the geometric complexity and the moving boundary problem, a zonal method and an overlapped grid-embedding scheme were employed respectively. In the zonal method, the complex computational domain was divided into several simple sub-domains. In each sub-domain containing valve leaflets, the Chimera grid-embedding technique was employed by using two overlapped grids. Grid one occupied the whole sub-domain, and grid two wrapped around the tilting-disk moving with the disk. In the Chimera grid embedding technique, grid points that lie within the disk geometry and outside the channel grid are excluded from the solution process. These excluded points are called hole points, and the immediate neighbors of the hole points are called fringe points. The information is passed from one grid to another via fringe and grid boundary points by interpolating the dependent variables. An array was used to mark the hole, grid boundary and fringe points, and exclude hole and grid boundary points from solution procedure. The calculation result was compared with experimental measurements and was proved to be satisfactory.

In his Ph.D. works, De Hart [3] at Eindhoven University of Technology modeled the 2-D and 3-D aortic valve motion using a so-called fictitious domain method based on Galerkin finite element method to tackle the fluid-structure interaction. Lagrange multiplier was used to couple the fluid domain and the structure domain, and the system was solved in a fully coupled way leading to a simultaneously obtained solution of the fluid and structure unknowns. He studied the flow fields in natural heart valves and bio-prosthetic heart valves and analyzed mechanical properties of the valves during the systolic phase. This research is useful in the sense that both the mechanical responses of heart valves and the information of the surrounding flow fields are obtained together, which are important to the design and failure analysis of bio-prosthetic heart valves. However, the fictitious domain method is a quasi-static method for the structure domain, which compromises the accuracy of transient structure motion, especially when the speed of the leaflets is high.

2.2 COMPUTATIONAL FLUID DYNAMICS

With the advent of high-speed computers and the development of analysis algorithms, computational fluid dynamics (CFD) commonly used in aerospace engineering has gained much popularity in biological fields in recent years. As an independent research method, it can provide much information that experimental and theoretical methods cannot achieve in the biological and biomedical researches. An in-depth review on historical CFD works about incompressible flows, moving boundaries, fluid-structure interaction and parallel computation employed in this work is detailed in this section.

2.2.1 Incompressible Flow

The algorithms used to solve incompressible flows can be classified into three major categories: vorticity-based formulations, artificial compressibility and pressure correction methods [12]. For two-dimensional calculations using the vorticity-based approach, the governing equations are formulated in terms of a stream function and vorticity [13]. Direct extension of this method to three dimensions is not possible. However, different formulations have been used in three-dimensions, such as the vorticity–velocity approach [14], the vector-potential approach [15], and the vector stream function method [16]. The selection of boundary conditions is often quite challenging for these methods. The artificial compressibility method (ACM) [17,18] adds a pseudo time derivative of the pressure to the continuity equation and the incompressible flow field is treated as compressible during the transient stage. Initially, ACM has been used to calculate steady flows since this artificial term vanishes when the steady state is achieved [18]. However, it has been demonstrated recently that this approach can be made time accurate by considering the time-dependent terms in the momentum equations as illustrated in Tai and Zhao [19] and Nithiarasu [20]. Then the time accurate solution can be obtained by iterating to the pseudo-time convergence at each physical time step. The third class of algorithms, called pressure correction methods, uses a Poisson equation for the pressure field. This procedure computes an initial velocity field using the momentum equations and then uses the pressure Poisson equation to correct the velocity field so that the continuity equation is satisfied [21]. Historically, the pressure correction method was initially the most extensively used while ACM has only received comparable research attention in recent years. This is due to the successful achievement with transonic flow time-marching schemes. The main advantages of the ACM are memory efficiency and suitability for application to

massively parallel environments, and mutual technology transfer with compressible time-marching systems.

Time dependent calculations can be performed with either explicit or implicit schemes. One major drawback of using an explicit scheme is that the maximum time step must be less than some limit imposed by some stability constraints. This can result in utilizing a large amount of computational resources to make calculations over a given period of time if the stability requirement is much less than the minimum time step size for an accurate solution. The size of the maximum time step is directly related to the computational cell size, and meshes with large variations in cell size will have the global time step limited by the smallest cell [22]. Typically, cells at the outer boundaries of a mesh are orders of magnitude larger than the cells close to the interior surfaces, especially for viscous flow applications. The algorithm in [23] employed the multigrid method to advance implicitly in time in order to avoid the explicit time step limitation. An implicit dual time stepping scheme [23-25] is employed to obtain the numerical solutions for the multigrid method. This method is found to be much more effective in ensuring convergence than the original explicit Runge-Kutta time stepping scheme for both steady and unsteady flows. For time dependent problems, calculations performed within each physical time step are equivalent to a steady-state problem in pseudo-time where a number of sub-iterations are performed and the residuals are reduced to a low level. This allows various convergence acceleration techniques, such as local time stepping, and implicit residual smoothing, to be used during the convergence to steady state in pseudo-time. These techniques destroy time accuracy in pseudo-time, but since they are only used to accelerate convergence in pseudo-time, accuracy in real time is not affected.

2.2.2 Moving Boundary

Numerical simulation is always implemented on a certain computational mesh with appropriate governing equations being discretized and solved on it. Resolving governing equations on stationary meshes for fixed geometries is straightforward. However in many biological cases and many engineering applications, the numerical simulation involves moving boundaries, such as flexible artery and moving heart valves. In such a situation, the initial computational mesh will deteriorate with the changing geometries. Therefore some special techniques are needed to tackle this problem. Various techniques devoted to solving the moving boundary problems have been developed by researchers in different fields. Generally these techniques can be classified into three major groups according to the properties of computational meshes used in the computation. These three groups are: Eulerian Methods, Arbitrary Lagrangian-Eulerian Methods and Hybrid Methods.

2.2.2.1 Eulerian Methods

In the Eulerian methods, the mesh is stationary. It will not move or deform, so the fluid motion can be conveniently described with respect to this Eulerian reference frame. The Eulerian methods include the Immersed Boundary Method (IBM), Fictitious Domain Method (FDM), Ghost Cell Method and Ghost Fluid Method.

2.2.2.1.1 Immersed Boundary Method

The term “immersed boundary method” was first used in reference to a method developed by Peskin [55, 56] to simulate the natural heart and heart valves combining Eulerian flow equations and a Lagrangian description of heart walls and valves. The

most distinguished feature of this method is that the entire simulation is carried out on a Cartesian grid which does not conform to the geometry of the heart. In this method, the fluid flow is governed by the incompressible Navier-Stokes equations which is solved on the stationary Cartesian grid. The immersed boundary is represented by a set of elastic fibres and the locations of these fibres are tracked in a Lagrangian manner by using a collection of massless points that move with the local fluid velocity. The stress and deformation of these elastic fibres are related by a constitutive law such as the Hooke's Law. The effect of the immersed boundary is essentially captured by transmitting the fibre stress to the fluid, which is implemented by adding a localized forcing term in the Navier-Stokes equations. Because the locations of the fibres do not generally coincide with the nodal points of the Cartesian grid, the forcing is distributed over a band of cells around each Lagrangian points and this distributed force is imposed on the Navier-Stokes equations of the surrounding nodes. Thus in effect, the originally sharp fluid-structure interface is essentially replaced by a smoothing function. The immersed boundary method has been applied to a wide range of problems, mostly in biofluid dynamics, including blood flow in the heart [57], platelet aggregation during blood clotting [27], flow in arteries [29], and simulating the motion of flexible pulp fibers [30]. Later Peskin and his co-researchers developed the method further and published an adaptive version [31] and a second-order extension [32] of the immersed boundary method.

2.2.2.1.2 Fictitious Domain Method

Glowinski *et. al* [58, 59] studied a Lagrange multiplier based fictitious domain method for the numerical solutions of three-dimensional elliptic problems with Dirichlet boundary conditions and also for the Navier-Stokes equations modeling incompressible

viscous flow. The method is based on the imposition of velocity constraints associated with moving internal boundaries by means of a Lagrange multiplier. It is similar to the immersed boundary method in which at a number of control points tension forces are imposed and they are distributed to the neighbouring points. The fictitious domain method allows coupling of domains with dissimilar element distribution and/or interpolation order using Lagrange multipliers on a fictitious boundary representing actual structure [60]. The main advantage of the fictitious domain method is that the different mathematical descriptions for the fluid and structure can be maintained, allowing convenient classical formulations (Lagrangian or Eulerian formulations) for each of these subsystems. Moreover, the fluid mesh is not altered or interrupted by the presence of the immersed domain, and therefore preserving its original quality.

The aforementioned Eulerian methods are attractive techniques for moving boundaries due to their inherent simplicity with the avoidance of mesh treatment, and their applications to low and moderate Reynolds number flows have been successful. However, their extension to higher Reynolds number flows is challenging due to the need to accurately resolve the boundary layers on the immersed surfaces not aligned with the grid lines. In such cases the local accuracy of the solution assumes greater importance, and the spreading of the effect of the immersed boundary introduced by the smooth force distribution function is less desirable.

2.2.2.1.3 Ghost Cell Finite Difference Method

To retain a sharp fluid-structure interface and achieve a higher local accuracy near the immersed moving boundary, the computational stencil near the interface is modified to directly impose the boundary conditions on the immersed boundary. Majumdar *et. al*

[61] proposed a ghost fluid cell method to treat the immersed boundary for structured grids. Ghost cells are defined as cells in the solid domain that have at least one neighbour in the fluid domain. For each ghost cell, an interpolation scheme that implicitly incorporates the boundary condition on the immersed boundary is then devised. This method has been used for simulating a wide variety of flows including compressible flow past a circular cylinder and an airfoil [62] at Reynolds numbers up to $O(10^5)$, aquatic propulsion [63], flow through a rib-toughened serpentine passage [64] and turbulent flow past a road vehicle [65].

2.2.2.1.4 Ghost Fluid Method

Fedkiw [66] proposed another ghost fluid method to tackle the moving boundary problems. In his method, the fluid-structure interfaces are treated in Eulerian scheme that maintains a Heaviside profile of the density with no numerical smearing. A level set function is used to define two separate domains for the two separate fluids. Then a ghost cell at every point in the computational domain is imposed. In this way, each grid point contains the mass, momentum and energy for the real fluid that exists at that point (according to the sign of the level set function) and a ghost mass, momentum and energy for the other fluid that does not really exist at the point (it is on the other side of the interface). With the help of the ghost fluid, the integrity of the background Eulerian mesh can be maintained even though the fluid-structure interface cuts part of the mesh. So that the interface boundary conditions or jump conditions are properly captured.

The methods reviewed in the previous two sections introduce the boundary conditions directly into the discrete equations. They enable a sharp representation of the immersed boundary and this is desirable, especially for high Reynolds numbers.

Furthermore, the direct enforcement of boundary conditions does not introduce any extra stability constraints in the representation of the solid bodies. Finally, this approach decouples the equations for the fluid nodes from those for the nodes in the solid, thereby obviating the solutions of the governing equations for the solid grid nodes. However, one disadvantage of this approach is that inclusion of boundary motion can be more difficult than the FD and IB methods. Furthermore, the ghost fluid cell finite difference method and ghost fluid method are limited to structured grids so far, hence their applications to complex geometries are also limited.

2.2.2.2 Arbitrary Lagrangian-Eulerian (ALE) Method

The arbitrary Lagrangian-Eulerian method, first proposed by Donea et al. [67], involves a continuous adaptation of the mesh without modifying the mesh topology in solving the fluid-structure interaction and moving boundary problems. The mathematical formulation of the equation of motion for a fluid is most conveniently described with respect to a Eulerian reference frame. However, this is incompatible with the Lagrangian formulation which is more appropriate to describe a solid phase. The ALE methodology combines the best of the features of these two different formulations. The essence of the ALE is that the mesh motion can be chosen arbitrarily [69]. It includes three phases: an explicit Lagrangian update, an implicit iteration of the momentum equation with the equation of state, and a rezone/map phase [69]. The Lagrangian phase provides an explicit update of the equation of motion. When the material velocities are much smaller than the fluid sound speed, the optional implicit phase allows sound waves to move many cells per cycle, thereby significantly improving computational efficiency. Finally, a rezone algorithm may prescribe mesh velocities relative to the

fluid, thus necessitating remap phase in which the solution from the end of phase two is mapped onto the new mesh. A particularly important property of ALE is that it provides a means of minimizing advection errors [70]. ALE techniques have gained popularity for transient, high speed, moderately large deformation problems of solid mechanics [71]. They are most commonly used in the aerospace field, where solid structure are subjected to complex airflows with thin boundary layers, e.g. see [72]. In heart valves, however, due to large-scale movement of a thin leaflet within the computational domain, it is generally difficult to adapt the mesh in such a way that a proper mesh quality is maintained without changing the mesh topology. Alternatively, remeshing can be performed. It is performed if the mesh quality is degenerated too much. This technique has been used by Horsten [73] to model the aortic valve in two dimensions. Remeshing, however, not only introduces artificial diffusivity, it may also be difficult to perform with sufficient robustness and accuracy for three-dimensional problems. Generally ALE methods can use Mesh Regeneration Method or Deforming Mesh Method according to different requirements.

2.2.2.2.1 Mesh Regeneration Method

For large deformation or displacement cases, the computational mesh needs to be regenerated. With different complexity of the flow field, mesh regeneration might be carried out as local mesh regeneration or global mesh regeneration. When dealing with large deformation, mesh regeneration methods could provide better mesh for the moving boundary calculation than any other methods, but its computational expense is out of the reach of most researchers.

In recent years, Professor Tezduyar's research group in University of Minnesota has been using this method. With the help of a 512-node Thinking Machines CM-5, they did much research of this kind, such as simulation of multiple spheres falling in a liquid-filled tube [74, 76], airflow past the parachute [74, 75], and flow past a flapping wing [77]. In fact their method was a combination of deforming mesh and mesh regeneration. In their work usually the deforming mesh method was first used when boundary movement was relatively small. When the mesh became overly skewed, grid regeneration was deployed and the flow variables were interpolated from the old mesh to the new one. In some examples, such as the simulation of 1,000 spheres falling in a liquid-filled tube [75], the total mesh was regenerated every several steps. This may be expensive work done for moving boundary problems. Hassan et al. [78] calculated transient motion of a 3-dimensional body subjected to large displacement and pseudo transient motion of a missile configuration mounted initially underneath a wing using mesh regeneration combined with deforming mesh in 1998. They also used deforming mesh when the displacement was relatively small, and regenerated the mesh when it was severely skewed. As their configuration was relatively simple, they only regenerated the mesh in certain regions of the flow field where the mesh was severely skewed.

2.2.2.2.2 Deforming Mesh Method

The Deforming Mesh Method is mainly used in systems with small displacements or small deformations. In this method, when the boundary of flow field has certain small changes, the grid points lying on the moving boundary move with the boundary. Integral grid points near the moving region also adjust their positions to smooth the movement from the moving boundary to the internal region of the flow field. With this

method, the computational expense is not excessive. It is also used for calculations of aeroelasticity and elastic tubes.

Heil [79] studied stokes flow in an elastic tube with this method in 1998. In his calculation, a structured fluid mesh was generated in the underformed cylindrical tube. In this reference configuration, the fluid nodes on the tube wall coincide with certain material points in the tube. When the tube wall deformed, the fluid mesh was adjusted by keeping the fluid mesh “attached” to the same material points in the tube. The interior mesh points changed their radial displacements with each other to suit the change in the boundary. In this implementation, the coordinates of grid points were changed, but their relative positions were kept.

Hassan et al. [74] calculated pseudo-transient motion of a missile mounted initially underneath a wing with the deforming mesh method and the local mesh regeneration. The deforming mesh method was used when small displacement assumption was satisfied. In their deforming mesh method, the mesh was modeled as a spring network, with nodal points on the outer boundary held fixed while the instantaneous locations of points on the moving boundary were specified. The locations of the inner nodes of the mesh were then determined by solving the static equilibrium equations that resulted from summing for the forces in the spring system at each node. The mesh-smoothing algorithm thus altered the positions of the interior nodes without changing the topology of the mesh.

Farhat *et. al* [80] calculated a flat panel with infinite aspect ratio in supersonic airstreams in 1995 with the ALE method. In their work, differential equation was used

to calculate the new positions of grid points, and the mathematical description of moving boundary was used as boundary condition to this differential equation for mesh movement. In their work the ALE method was just the deforming mesh method. Sculz and Kallinderis [81] calculated an elastically mounted cylinder vibrating freely in the flow field with deformable hybrid grids in 1998. In their work, dynamic deformation of the hybrid mesh was achieved by matching the computed structural displacement with the mesh displacements immediately surrounding the body. The displacement of the remaining mesh was then linearly reduced based on the distance away from the structure such that the mesh near all far field boundaries remained undisturbed. Zhao and Ahmed [150] calculated the self-excited oscillations in a 2D collapsible channel using the ALE method. A dynamic mesh method was adopted in their work to handle the large deformation of the collapsible channel wall. With the combination of ALE and dynamic mesh method, the efficiency of the unsteady flow simulation was greatly improved and large deformation of the moving boundaries could be tackled.

The ALE method is accurate and allows for jumps of gradients of fluid variables across the membrane. But it can only tackle small displacements. When the deformation is too large, regeneration of the computational grid is needed.

2.2.2.3 Hybrid Methods

The hybrid method combines the Lagrangian and Eulerian reference. The Chimera Mesh Method is a typical hybrid method.

2.2.2.3.1 Chimera Mesh Method

When the boundary moves without changing its geometry, Chimera mesh might be a suitable method for cases with large displacements. In this method there are two sets of meshes, a Eulerian mesh and a Lagrangian mesh. The Lagrangian mesh surrounds the moving boundary and moves with it. The Eulerian background mesh occupies other regions of the flow field. The two meshes have an overlapping region or a transient region. When the moving boundary moves to a new position, the transition region is regenerated, or the overlapping region is redefined. Flow variables on the Eulerian mesh are computed in a Eulerian manner, and those on the Lagrangian mesh are evaluated in a Lagrangian manner. Kiris *et al.* [54] calculated the flow through artificial heart devices with this method in 1997. In their work, structured Chimera mesh was used around the heart valves in mitral and aorta positions. The chimera mesh and the background mesh exchanged flow information through an overlapping zone. Comparison between numerical results and experimental measurements showed that they related well.

For moving boundary with fixed geometry, Chimera mesh method could be a good reference. It is less computational expensive than the mesh regeneration method, and could effectively deal with large displacements. The shortcoming is the introduction of errors during value exchanging between the two meshes.

2.2.3 Fluid-Structure Interaction

Fluid-structure interaction (FSI) is a common phenomenon in nature, such as unsteady viscous flow in elastic (collapsible) tubes [79], flapping flags [82] and others. There

exist many examples where the motion of the thin-walled, leaflet-like structure is driven by the motion of fluid. A typical example in biomechanics is the opening and closing behaviours of aortic heart-valves, which is a delicate interaction between blood flow and geometrical and stiffness properties of the heart-valve leaflets.

In fluid-structure interaction, the movement of the structure totally depends on the forces, especially the fluid forces, acting on the structure. Due to the action of these forces, the structure will have some kind of motion, including rotation, deformation or translation or has some combinations or even all of these changes. Due to the corresponding changes of the structure in spatial positions and geometric shapes, the flow field around the structure is also changed. Fluid forces acting on the structure evolve with the changing flow field; this brings the structure to a new state of movement. In this way, fluid and structure interact with each other in a loop fashion, and the system evolves with very complex nonlinear dynamics.

In the numerical simulation of fluid-structure interaction phenomenon, the fluid phase is most conveniently described with respect to a Eulerian reference frame, while a Lagrangian formulation is more appropriate for the solid phase. However, these two formulations are incompatible. Many methods have been proposed to tackle this incompatibility, see the previous review on moving boundary problems. With the help of these methods, many researchers have studied this phenomenon. In most of these works, the flow field and the structure are computed separately with fluid and structure dynamics. The study always begins by computing the flow field with the Navier-Stokes equations or Euler equations, calculating fluid forces acting on the structure. Depending on the complexity of the problem and the geometric change of the structure, the response of the structure is calculated either by the finite element method or really as

simple as an ordinary differential equation. The corresponding displacements or geometric changes of the structure are then fed into the flow field, and the computational mesh of the flow field is refreshed, using one or two of the techniques discussed in the previous section. The flow field is then recalculated on the refreshed mesh and fluid forces are then passed to the structure again for calculating new structure states. In this way the calculation is carried out for several prescribed iterations or until the desired error limit is satisfied, then the interaction between fluid and structure is deemed to be fully coupled and current values of system variables can satisfy the governing equations of both the fluid and the structure. Now the system can step forward to a new stage and the above procedures are repeated in an iterative manner.

In recent years, Baaijens' [60] research team at Eindhoven University of Technology, Netherlands, has been active in studying fluid-structure interaction in the aortic valve. In 2001, Baaijens proposed a new numerical method: the fictitious domain/mortar method for studying fluid-structure interaction. In his work, this method was used to study the motions of flexible thin beams in a Stokes flow and a Navier-Stokes flow. Two thin beams of different thickness were studied, and the results showed that the motions of these two beams were different. This illustrated the feasibility of the fictitious domain/mortar method for investigating the impact of structural changes on fluid-structure interaction. The key advantage of this method is that it does not require any updating of mesh of the fluid domain. This makes the fictitious domain/mortar method attractive for 3D calculations, where a robust updating of the mesh to follow the motion of the thin-shell-like structures poses major difficulties. In year 2000, De Hart et al. [83] studied the two-dimensional fluid-structure interaction model of aortic valve using the fictitious domain method and validated the numerical results by LDA

measurements and high-speed video recordings. In the work, the 2D model was analyzed based on finite element method. The fluid and structure were coupled by the fictitious domain method. A staggered approach was used to solve the system uncoupled [84]. In 2003, De Hart et al [85] extended their work to a three-dimensional model of the aortic valve.

Tezduyar's research team at University of Minnesota is another active group. In their works [74-77, 86-88], the fluid domain was solved with finite element method, and the moving structure was calculated with either finite element method for complex cases, or with dynamic equilibrium equation for simple cases. Refreshing of computational mesh was achieved by a combination of the deforming mesh method and the mesh regeneration method. At George Mason University, Professor Löhner's group is another active group in the research of FSI problems. The general idea of FSI calculation is the same as those described earlier. Besides the FSI calculation, they also considered the thermal effect in their calculations [89]. In their work, they combined commercial codes in their calculations. For CFD, they often used their FEFLO98 code for calculations, and for the structure they used COSMIC-NASTRAN for linear computational structure dynamics and DYNA3D for non-linear computational structure dynamics. Several examples are given in Professor Löhner's homepage [90], including generic weapon fragmentation, supersonic flow over a deforming panel.

Many other researchers also did much work in FSI, such as Elat *et al.* [91], Farhat *et al.* [92], Freitas *et al.* [93], Liang *et al.* [94], Piperno [95], Soria *et al.* [96], and Heil [79]. Their methods are almost the same as Tezduyar's method, except for using

the finite volume method for computing fluid domain, and using the deforming mesh method for mesh refreshment.

2.2.4 Parallel Computation

Parallelization has been identified as the basis for future high performance computing. In any of the areas concerned, particularly in mathematics, informatics, physics and all related disciplines up to the applications, researchers have heavily invested in parallelism. Computing speed and memory capacity have increased exponentially over the last several decades, have led naturally to work on parallel computational fluid dynamics.

Many techniques and tools for implementing parallel numerical methods are available to researchers. These include compiler directives (OpenMP [97]), message passing (MPI [98]), parallel languages (High Performance Fortran, HPF [99]), and parallel libraries (PETSc [100]). On machines with shared memory architectures, the usual parallelism model is Multiple Instruction Multiple Data (MIMD). The MIMD model is usually implemented at the loop level with compiler directives, such as the OpenMP set of directives. Although the MIMD model is easy to implement, scalable performance is usually limited to tens of processors [101]. Success with more processors is more difficult to produce [102]. On machines with distributed memory architectures, the popular parallelism model is Single Program Multiple Data (SPMD). The SPMD model is usually implemented by message passing, using libraries of message passing tools such as the MPI library [98] or PVM library [103]. The SPMD model has been shown to scale to hundreds and even to thousands of processors [101]. One important reason for this scalability is that good data locality can be maintained

with the SPMD model. However, message-passing implementations generally require extensive changes to a baseline code because the domain decomposition and message passing must be explicitly programmed. It needs the programmer to change all array declarations and all loop bounds, and explicitly code the transfer of data between processors. This is a large and difficult hurdle most of the programmers encounter.

McManus [104] examined the method of parallelization by geometric domain decomposition using the single program multiple data programming paradigm with explicit message passing in his Ph.D. thesis. In the thesis some techniques and strategies to parallelize unstructured mesh codes were developed and tested using an integrated CFD and solid mechanics code to explore the computational advantages offered by distributed memory parallel processors. The performance of the parallel code was examined and some work was carried out to achieve automatic parallelization. Yao *et al.* [105] presented the development and validation of an unsteady, three-dimensional, multi-block, parallel turbo machinery flow solver, TFLO. The Unsteady Reynolds Averaged Navier-Stokes (Unsteady RANS) equations are solved using a cell-centered discretization on arbitrary multi-block meshes. The solution procedure is based on efficient explicit Runge-Kutta methods with several convergence acceleration techniques such as multigrid, residual averaging, and local time stepping. The solver is parallelized using domain decomposition, an SPMD strategy, and the MPI Standard. The dual-time stepping technique is applied to advance unsteady computations in time. TFLO is routinely run on the CRAY T3E, SGI Origin 2000 and IBM-SP systems. Das *et al.* [106] implemented a three-dimensional unstructured grid Euler solver on massively parallel distributed-memory computer architectures. Their goal is to minimize solution time by achieving high computational rates with a numerically efficient

algorithm. An unstructured MG algorithm with an edge-based data structure has been adopted, and a number of optimizations have been devised and implemented to accelerate the parallel computational rates. Large practical unstructured grid problems are solved on the Intel iPSC/860 hypercube and Intel Touchstone Delta machine. The quantitative effects of the various optimizations are demonstrated and the combined effect of these optimizations leads to roughly a factor-of-three performance improvement. The overall solution efficiency is compared with that obtained on the Cray Y-MP vector supercomputer.

Professor Tezduyar's team in University of Minnesota has been using parallel computing extensively. With the support of powerful hardware facilities, most of their work was concerned with complex fluid-structure interaction problems in which mesh adaptation or regeneration took much of computing time. In 1995, they calculated 3D flow past a rectangular wing at Reynolds number 1,000, 2,500 and 10,000,000 in which the finite element mesh consisted of 753,168 nodes and 727,552 hexahedral elements, and flow past a flapping wing in which the finite element mesh consisted of 145,402 nodes and 137,280 elements [77]. These calculations were executed on a 512-node CM-5 supercomputer and good efficiency was achieved. In 1997 they calculated 3D flow past a sphere at Reynolds number of 400 [107]. The computational mesh consisted of 43,282 nodes and 258,569 elements and a mesh partition program, METIS, was used for domain decomposition. The calculation was carried out on a 256-PN CM-5 supercomputer. In 1997 they calculated flow past an automobile and 5 spheres falling in a liquid-filled tube on a Thinking Machine CM-5 and a Cray T3D supercomputer [74]. Another problem of parallel finite element simulation of large ram-air parachutes was also carried out on the CM-5 supercomputer with a data-parallel programming paradigm

and on a Cray T3D supercomputer with a message-passing paradigm [76]. The most difficult of their work might be the calculation of 1,000 spheres falling in a liquid-filled tube at Reynolds number 10 [75]. The mesh size during the simulation reached 1.2 million tetrahedral elements in this calculation. The simulation involved calculation of spheres colliding with each other, forming groups, bumping on the tube wall, as well as the fluid-sphere interaction, mesh regeneration and mesh partition. In the simulation, the mesh partition, flow computations and mesh movements were performed on a 512-node Thinking Machines CM-5, automatic mesh regeneration and projection of the solution were accomplished on a 2-processor SGI ONXY2 workstation.

2.3 COMPUTATIONAL STRUCTURAL DYNAMICS

This section reviews some computational structural dynamics works on fluid-structure interaction phenomena using the finite volume scheme. Some solid mechanics works about the shell model for thin-walled structures are also reviewed here.

2.3.1 Finite Volume Method for Structural Analysis

Computational structural dynamics (CSD) has been dominated by the finite element method (FEM) for the past three decades. However, there is now an active interest in finite volume (FV) discretization approach. Rather than simply attempting to displace the FE method from its position of prominence, this development is motivated by the desire to produce new CSD procedures that are compatible with existing baseline computational fluid dynamics solvers. In general, these procedures all convert surface or volume integrals representing the internal strain energy associated with each element or cell into line or surface integrals which are evaluated around the cell boundaries and

ensure that the tractions applied to these boundaries maintain each cell in a state of equilibrium. The tractions terms within the integrals can be related to strains and hence to the displacement derivatives by employing an appropriate constitutive relationship, like the Hooke's Law. The new CSD procedures have been integrated into CFD facilities and this has enabled complex problems involving the interaction between moving fluids and deforming structures such as pressure surging in pipelines [108] and flow induced vibrations in flexible structures [109] to be analyzed successfully. Earlier works [110-113] focused on developing FV based structural analysis procedures capable of predicting displacements, strains and stresses in loaded two and three dimensional linear elastic structures undergoing small deformations, and more complex problems incorporating material nonlinearities were presented by Taylor *et al.* [114]. Recent development of FV on structural analysis has been extended to problems involving structures undergoing large displacements as reported in these works [115, 116]. Slone *et al.* [117] studied the dynamic fluid-structure interaction problems by using a consistent set of FV methods on a single unstructured mesh. A vertex-based finite volume scheme was implemented for both the dynamic solid mechanics and fluid mechanics. Time accurate solutions could be obtained with the help of predictor-corrector Newmark scheme, which were used to analyze the onset of flutter in aircraft wings [118]. Fallah [119] developed cell-vertex finite volume (CV-FV) formulations for thin and thick plate analysis. In his work, the control volumes in the cell-vertex finite volume formulations were constructed over the mid-plane of the plate, and shape functions were used to describe the variation of a variable within an element.

2.3.2 Triangular Shell Element

Triangular shell elements are very useful for the solutions of large scale problems in engineering fields, like shell roofs under static and dynamic loads, sheet stamping process, vehicle dynamics, crash-worthiness situations and analysis of thin-walled structures. Most of these problems have high geometrical and material non-linearities. These difficulties are aggravated by the need to discretize the complex geometrical shapes. Here the use of triangular shell elements and unstructured meshes are a critical necessity. Several researchers have tried to derive triangular plate and shell finite elements with displacements as the only nodal variables [120-124]. In essence, all methods try to approximate the curvatures over an element in terms of the deflection of the nodes in a surrounding patch of elements. The formulations of displacement-based shell elements have been implemented into sheet forming analysis using explicit dynamic codes in the works of Yang *et al.* [125], Brunet *et al.* [126] and Rio *et al.* [127]. Oñate *et al.* [128, 129] have developed rotation-free thin (Kirchhoff-Love) plate and shell triangles by combining the standard finite element (FE) interpolation with finite volume (FV) concepts. In his works, the curvatures are expressed over a control domain in terms of the displacement gradients along the domain edges. These gradients are in turn written in terms of deflections of the nodes belonging to an element patch surrounding the control domain and this leads to a relationship between curvatures and nodal displacements. A mixed Hu-Washizu functional provides a framework for the derivation of the discretized equations for the rotation-free plate and shell triangles combining FE and FV approaches [130, 131]. An enhanced version of the shell element formulations is reported in [132] by using a quadratic interpolation of geometry in terms of the six-patch nodes, which computes an assumed linear membrane strain field, which improves in-plane behavior of the original elements.

2.4 REMARKS

From above reviews, it can be seen that extensive efforts have been put into numerical simulations of steady and unsteady flows in heart valves. Most of these works did not pay much attention to the moving boundary and the fluid-structure interaction in the calculations. For those considered both the moving boundary and the fluid-structure interaction, there were several problems encountered in their work: either the scheme for moving boundary was valid only for small mesh deformation [41, 42, 49, 50], or the implementation process was not implemented properly, and with no convincing results to support the derivation [26, 41, 42, 51-54]. There is still much work to do in studying pulsating flows with blood-leaflet interaction in heart valves.

In the moving boundary, the mesh undergoes a large or small displacement with the development of system dynamics. When this happens, the mesh needs some form of adjustment or mesh regeneration, and all these methods is computational expensive. In order to avoid mesh regeneration, Eulerian methods remain attractive measures for the moving boundary problems. However, the immersed boundary method and fictitious domain method smooth the boundary conditions on the fluid-structure interfaces, which are not suitable for high Reynolds numbers problems or problems with strong emphasis on maintaining a sharp fluid-structure interface. The ghost fluid cell finite difference method can track a sharp interface, but it can not handle complex geometries due to the structured mesh used. The ghost fluid method uses a level set function, and defines two separate fluid domains (one for real fluid, another for ghost fluid). The two sets of fluid domain is a burden for computing resources when it comes to large scale fluid-structure interaction problems. And this method is not suitable for implementation on unstructured meshes, hence it can not handle complex geometries, either.

The important parts of parallel computation are mesh partitioning and domain decomposition. Decomposition of a mesh into a set of S sub-domains that may be allocated to a set of P processors involves finding partitions of the mesh so that the amount of computational time on each processor is almost equal. A good mesh partition is one which divides the computational load equally amongst the sub-domains (i.e. each sub-domain having the same number of nodes) and minimizes the amount of communication required between sub-domains. Single Program Multiple Data (SPMD) diagram can be conveniently implemented together with the domain decomposition technique.

The finite volume method popular with computational fluid dynamics (CFD) is now widely used in computational structural dynamics (CSD). It provides an important tool to embed the structural analysis procedures with existing finite-volume based CFD solvers. The triangular shell element is the simplest element, which is useful for large deformation structure analysis and complex geometries. And triangular shell elements with only translational degrees of freedom have been successfully implemented in a wide range of structural analysis.

The above remarks serve as a guide for the implementations of the parallel computing, immersed membrane method and structural dynamics into a three-dimensional incompressible Navier-Stokes unstructured grid solver for the numerical simulation of bio-prosthetic heart valves under physiological condition with blood-leaflet interaction.

CHAPTER THREE

COMPUTATIONAL FLUID DYNAMICS

This chapter describes the mathematical formulation for the three-dimensional unsteady incompressible viscous flow solver developed in this work. The finite-volume Navier-Stokes solver is based on the artificial compressibility method (ACM) with a high-resolution method of characteristics-based scheme on unstructured grids. The unsteady flow is calculated with a matrix-free implicit dual time stepping scheme. A five-stage Runge-Kutta time integration algorithm is used between each physical time step to iterate the numerical solution in pseudo time until convergence is reached. The methods used to accelerate the convergence rate to steady state in pseudo time, which are local time stepping and implicit residual smoothing, are briefly described.

3.1 GOVERNING EQUATIONS

For an incompressible flow, the density does not vary with time; therefore, the derivative of the density term in the conservation of mass equation is zero. And the energy equation is decoupled from the momentum equation; therefore, they can be solved independently. The three-dimensional governing incompressible unsteady Navier-Stokes equations are as follows,

$$\nabla \cdot \mathbf{U} = 0 \quad (3.1)$$

$$\frac{\partial u}{\partial t} + \nabla \cdot (u\mathbf{U}) = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} \quad (3.2)$$

$$\frac{\partial v}{\partial t} + \nabla \cdot (v\mathbf{U}) = -\frac{1}{\rho} \frac{\partial p}{\partial y} + \frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} \quad (3.3)$$

$$\frac{\partial w}{\partial t} + \nabla \cdot (w\mathbf{U}) = -\frac{1}{\rho} \frac{\partial p}{\partial z} + \frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z} \quad (3.4)$$

where $\mathbf{U} = u \cdot \vec{i} + v \cdot \vec{j} + w \cdot \vec{k}$ and for Newtonian fluid, the shear stress is,

$$\tau_{xx} = \lambda(\nabla \cdot \mathbf{U}) + 2\mu \frac{\partial u}{\partial x}, \quad \tau_{xy} = \tau_{yx} = \mu \left(\frac{\partial v}{\partial x} + \frac{\partial u}{\partial y} \right),$$

$$\tau_{yy} = \lambda(\nabla \cdot \mathbf{U}) + 2\mu \frac{\partial v}{\partial y}, \quad \tau_{xz} = \tau_{zx} = \mu \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right),$$

$$\tau_{zz} = \lambda(\nabla \cdot \mathbf{U}) + 2\mu \frac{\partial w}{\partial z}, \quad \tau_{yz} = \tau_{zy} = \mu \left(\frac{\partial w}{\partial y} + \frac{\partial v}{\partial z} \right),$$

where μ represents the molecular viscosity coefficient, and λ represents the second viscosity coefficient. Stokes made the hypothesis that $\lambda = -\frac{2}{3} \cdot \mu$ [133]. Substituting the shear stresses defined above into Equations (3.2) to (3.4), the equations become,

$$\frac{\partial u}{\partial t} + \nabla \cdot (u\mathbf{U}) = -\frac{1}{\rho} \frac{\partial p}{\partial x} + \mu \cdot \nabla^2 \cdot u \quad (3.5)$$

$$\frac{\partial v}{\partial t} + \nabla \cdot (v\mathbf{U}) = -\frac{1}{\rho} \frac{\partial p}{\partial y} + \mu \cdot \nabla^2 \cdot v \quad (3.6)$$

$$\frac{\partial w}{\partial t} + \nabla \cdot (w\mathbf{U}) = -\frac{1}{\rho} \frac{\partial p}{\partial z} + \mu \cdot \nabla^2 \cdot w \quad (3.7)$$

The position vector is non-dimensionalized by a characteristic length L^* , the velocities are non-dimensionalized by a free-stream velocity, U_∞^* , the time is non-dimensionalized by L^*/U_∞^* , and the pressure is non-dimensionalized by $\rho_\infty (U_\infty^*)^2$ where ρ_∞ is the free-stream density. The superscript * represents dimensional quantities and ∞ indicates free-stream conditions. The non-dimensional form can be summarized as follows,

$$(x, y, z) = \left(\frac{x^*}{L^*}, \frac{y^*}{L^*}, \frac{z^*}{L^*} \right); \quad (u, v, w) = \left(\frac{u^*}{U_\infty^*}, \frac{v^*}{U_\infty^*}, \frac{w^*}{U_\infty^*} \right);$$

$$t = \frac{t^*}{L^*/U_\infty^*}; \quad p = \frac{p^* - p_o}{\rho_\infty (U_\infty^*)^2}; \quad \text{Re} = \frac{\rho_\infty U_\infty^* L^*}{\mu_\infty^*}$$

where Re represents the Reynolds number. Re-formulating the Navier-Stokes equations into non-dimensional form,

$$\nabla \cdot \mathbf{U} = 0 \quad (3.8)$$

$$\frac{\partial u}{\partial t} + \nabla \cdot (u\mathbf{U}) = -\frac{\partial p}{\partial x} + \frac{1}{\text{Re}} \cdot \nabla^2 \cdot u \quad (3.9)$$

$$\frac{\partial v}{\partial t} + \nabla \cdot (v\mathbf{U}) = -\frac{\partial p}{\partial y} + \frac{1}{\text{Re}} \cdot \nabla^2 \cdot v \quad (3.10)$$

$$\frac{\partial w}{\partial t} + \nabla \cdot (w\mathbf{U}) = -\frac{\partial p}{\partial z} + \frac{1}{\text{Re}} \cdot \nabla^2 \cdot w \quad (3.11)$$

Equation (3.8) shows that conservation of mass is an elliptic equation, which makes this a system of mixed-type equations. The algorithms developed for unsteady compressible flow solver can not be used efficiently for incompressible flow, since it is designed to solve system for hyperbolic equations. Therefore, the conservation of mass needs to be reformulated into hyperbolic form. An artificial time derivative of pressure is added to the continuity equation, which was first proposed by Chorin [18] for steady flows problem. It is possible to develop a time-accurate time-marching scheme for unsteady flows by using an implicit real-time discretization and a dual-time approach [17]. The main purpose of this artificial term is to directly couple the pressure and

velocity fields at the same time level, thus producing a hyperbolic system of equations (for the inviscid parts only), as shown in the following equations with the addition of pseudo-time, τ .

$$\frac{1}{\beta} \frac{\partial p}{\partial \tau} + \nabla \cdot \mathbf{U} = 0 \quad (3.12)$$

$$\frac{\partial u}{\partial \tau} + \frac{\partial u}{\partial t} + \nabla \cdot (u\mathbf{U}) = -\frac{\partial p}{\partial x} + \frac{1}{\text{Re}} \cdot \nabla^2 \cdot u \quad (3.13)$$

$$\frac{\partial v}{\partial \tau} + \frac{\partial v}{\partial t} + \nabla \cdot (v\mathbf{U}) = -\frac{\partial p}{\partial y} + \frac{1}{\text{Re}} \cdot \nabla^2 \cdot v \quad (3.14)$$

$$\frac{\partial w}{\partial \tau} + \frac{\partial w}{\partial t} + \nabla \cdot (w\mathbf{U}) = -\frac{\partial p}{\partial z} + \frac{1}{\text{Re}} \cdot \nabla^2 \cdot w \quad (3.15)$$

where β is an artificial compressibility parameter and is to be chosen to ensure the fastest convergence to the steady state. It has the same analogy as the equation of state for compressible flow as shown below,

$$p = \beta \rho \quad (3.16)$$

Equations (3.12) to (3.15) can be expressed in non-dimensional vector form as follows,

$$\mathbf{C} \frac{\partial \mathbf{W}}{\partial \tau} + \mathbf{K} \frac{\partial \mathbf{W}}{\partial t} + \nabla \cdot \vec{\mathbf{F}}_c = \nabla \cdot \vec{\mathbf{F}}_v \quad (3.17)$$

where

$$\mathbf{W} = \begin{bmatrix} p \\ u \\ v \\ w \end{bmatrix} \quad \bar{\mathbf{F}}_c = \begin{bmatrix} \mathbf{U} \\ u\mathbf{U} + p\delta_{ij} \\ v\mathbf{U} + p\delta_{ij} \\ w\mathbf{U} + p\delta_{ij} \end{bmatrix} \quad \bar{\mathbf{F}}_v = \begin{bmatrix} 0 \\ \frac{1}{\text{Re}} \cdot \nabla \cdot u \\ \frac{1}{\text{Re}} \cdot \nabla \cdot v \\ \frac{1}{\text{Re}} \cdot \nabla \cdot w \end{bmatrix}$$

$$\mathbf{K} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} \frac{1}{\beta} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where $\mathbf{W}$ is the vector of dependent variables and $\mathbf{U} = \mathbf{U}_f - \mathbf{U}_g$ is the velocity vector, $\mathbf{U}_f$ and $\mathbf{U}_g$ are the fluid velocity and grid velocity respectively, and $\bar{\mathbf{F}}_c$ and $\bar{\mathbf{F}}_v$ are the convective flux and viscous flux vector, respectively. $\mathbf{K}$ is the unit matrix, except its first element is zero and $\mathbf{C}$ is a preconditioning matrix.

Equation (3.17) can be recast in an integral form as follows,

$$\mathbf{C} \frac{\partial}{\partial \tau} \iiint_v \mathbf{W} dV + \mathbf{K} \frac{\partial}{\partial t} \iiint_v \mathbf{W} dV + \iiint_v \nabla \cdot (\bar{\mathbf{F}}_c - \bar{\mathbf{F}}_v) dV = 0 \quad (3.18)$$

Equation (3.18) is equivalent to the following equation,

$$\mathbf{C} \frac{\partial}{\partial \tau} \iiint_V \mathbf{W} dV + \mathbf{K} \frac{\partial}{\partial t} \iiint_V \mathbf{W} dV + \oint_{S_{cv}} (\bar{\mathbf{F}}_c - \bar{\mathbf{F}}_v) \cdot d\mathbf{S} = 0$$

Once the artificial steady state is reached, those derivative terms with respect to τ become zero and the above equation reduces to the following equation,

$$\mathbf{K} \frac{\partial}{\partial t} \iiint_V \mathbf{W} dV + \oint_{S_{cv}} (\bar{\mathbf{F}}_c - \bar{\mathbf{F}}_v) \cdot d\mathbf{S} = 0 \quad (3.19)$$

Equation (3.19) shows that the preconditioning matrix does not affect the solution and the original unsteady incompressible Navier-Stokes equations, Equations (3.8) - (3.11) are fully recovered.

3.2 ARTIFICIAL COMPRESSIBILITY METHOD

In the present work, the method of artificial compressibility, proposed by Chorin [18] is adopted for the solution of the incompressible Navier-Stokes equations. This method directly couples the pressure and velocity fields at the same time level, thus producing a hyperbolic system of equations. The artificial waves act as a mechanism to propagate the information throughout the domain, and driving the divergence of velocity toward zero. The main disadvantage of introducing artificial compressibility into the system of incompressible Navier-Stokes equations is that it is not suitable for time-dependent solutions and as a result, only steady state solutions can be obtained. However, this undesirable effect has been overcome by implementing implicit dual time-stepping, which is actually the same as the implicit dual time-stepping technique implemented in compressible flow solver [17]. The time-dependent solution is viewed as the solution of

a steady state problem at each time step and this pseudo-time is usually set to a very large value, which leads to a fast convergence. The working range of global value of β is between 0.1 and 10. The control of this parameter affects the relative speed of propagation of pressure wave and the spread of vorticity. Since $\Delta\tau$ is an artificial time-step, the speed of propagation of pressure wave can be affected by setting a larger value as compared to the time-step in the momentum equation.

The effects of incompressibility will increase if the value of β in Equation (3.12) is increased, however, large β makes the equations very stiff numerically. The major difference between compressible and incompressible flows is that there is no time derivative term in the incompressible continuity equation and whereas for compressible flow, the time derivative of density is in the continuity equation. Incompressible Navier-Stokes equations have a mixed parabolic-elliptic character. The solution of the elliptic equation consumes much of the computational effort and inhibits the development of methods with second order temporal accuracy [134] due to the need to perform global iterations for every time step. The most straightforward means of strongly coupling the continuity and momentum equations is to append a time derivative to the continuity equation. Since density is constant, adding $\partial\rho/\partial t$ is not possible. Time derivatives of velocity components appear in the momentum equations so they are not logical choices. That leaves the time derivative of the pressure as the clear choice.

3.3 FINITE VOLUME FORMULATION

There are two techniques in finite volume spatial discretization for a given grid, namely the cell-vertex and the cell-centred approaches. In the cell-vertex approach, the flow

field variables are stored at the vertices of the grid, whereas in the cell-centred approach, the variables are stored at the centroid of the cells. There are mainly two advantages for the cell-vertex approach. Firstly, the cell-vertex approximation needs considerably less computational effort and memory requirements for a given grid. For instance in the case of tetrahedral grids, the flux computations are performed by looping over the edges in the cell-vertex scheme, whereas in a cell-centred scheme they are looped over the triangular faces. Secondly, a cell-vertex scheme is better suited to compute the viscous fluxes, especially when the triangulation becomes highly irregular [135]. One major drawback of a cell-vertex scheme is that the solution quality is poorer than that of a cell-centred scheme for a given grid.

Equation (3.17) is discretized on an unstructured tetrahedral grid and a cell-vertex scheme is adopted here, i.e., all computed variables in vector $\mathbf{W}$ are stored at vertices of the tetrahedral cells. For every vertex, as shown in Figure 3.1, a control volume is constructed using the median dual of the tetrahedral grid [17]. In Figure 3.1, nodes A, P, B and C form the vertex of the tetrahedral cell and O is the centre of the element APBC. a, b and c are the median duals of the edges AP, BP and CP, respectively. 1, 2 and 3 are the centre points of triangles APC, CBP and ABP, respectively. In the cell-vertex scheme, the computed variables are stored at vertices A, P, B and C. Triangles O1a and O3a form the control volume surface for the convective term of the governing equation on edge AP. Likewise, the rest of the convective term for different edges are computed based on this way. Triangles APC, CBP, ABC and ABP form the corresponding control volume surfaces for the calculation of viscous term.

Figure 3.1 Construction of control volume within a tetrahedron for a node P .

Finite volume numerical discretization are based on the solution of the governing equations in integral form and spatial discretization is performed by using the integral form of the conservation equations over the control volume surrounding node P , as shown in the following equation,

$$\mathbf{C} \frac{\partial}{\partial \tau} \iiint_{cv} \mathbf{W}_p dV + \mathbf{K} \frac{\partial}{\partial t} \iiint_{cv} \mathbf{W}_p dV + \iiint_{cv} \nabla \cdot \bar{\mathbf{F}}_c dS = \iiint_{cv} \nabla \cdot \bar{\mathbf{F}}_v dV$$

To introduce the upwind scheme using an edge-based procedure, the convective term is transformed into the following equation,

$$\iiint_{cv} \nabla \cdot \bar{\mathbf{F}}_c dV = \oint_{S_{cv}} \bar{\mathbf{F}}_c \cdot \bar{\mathbf{n}} dS = \sum_{n=1}^{nbseg} \left[(\bar{\mathbf{F}}_c)_{ij} \cdot \bar{\mathbf{n}} \Delta S \right]_n \quad (3.20)$$

where $nbseg$ is the number of the edges associated with node P , $(\vec{\mathbf{F}}_c)_{ij}$ is the convective flux through the part of control volume surface. ΔS_n is part of the control volume surface associated with edge n . Therefore, all the fluxes are calculated for the edges and then collected at the two end of each edge for updating of flow variables in time marching. The inviscid flux at the control volume surface associated with an edge ij and $(\mathbf{F}_c)_{ij} = (\vec{\mathbf{F}}_c)_{ij} \cdot \vec{\mathbf{n}}_{ij}$ is evaluated based on Roe's approximate Riemann solver [136]:

$$(\mathbf{F}_c)_{ij} = \frac{1}{2} [\mathbf{F}_c^L + \mathbf{F}_c^R - |\overline{\mathbf{A}}|(\mathbf{W}^R - \mathbf{W}^L)] \quad (3.21)$$

where $|\overline{\mathbf{A}}|$ is defined as Roe-averaged Jacobian matrix.

The viscous term is calculated based on the cell-based method as follows,

$$\iiint_{cv} \nabla \cdot \vec{\mathbf{F}}_v dV = \oint_{S_{cv}} \vec{\mathbf{F}}_v dS = \sum_{i=1}^{ncell} [\vec{\mathbf{F}}_v \cdot \Delta S_c]_i \quad (3.22)$$

where $ncell$ is the number of elements associated with node P and ΔS_{ci} is the part of control volume surface in cell i . By using the following relation,

$$\oint_s dS = 0$$

The total vector surface of the control volume in a cell i becomes,

$$\Delta S_{ci} = \frac{1}{3} \Delta S_{pi}$$

Thus, the calculation of viscous terms can be simplified as,

$$\oint_{S_{cv}} \vec{F}_v \cdot d\vec{S} = \sum_{i=1}^{ncell} \left[\vec{F}_v \cdot \Delta S_c \right]_i = \frac{1}{3} \sum_{i=1}^{ncell} \left(\vec{F}_v \cdot \Delta S_p \right)_i \quad (3.23)$$

where ΔS_{pi} is the surface vector of the face opposite node P of the tetrahedron under consideration. Here the $(\vec{F}_v)_i$ is calculated at the center of the tetrahedron with a node P, and can be obtained by using the Green's Theorem based on the variables at the four vertices of the tetrahedron. Similar to the Galerkin type of formulation, the gradient of a flow variable ϕ at the center of a tetrahedron is evaluated as follow,

$$grad \phi_c = -\frac{\sum_{i=1}^4 \phi_i 9S_i}{27V} = -\frac{1}{3} \frac{\sum_{i=1}^4 \phi_i S_i}{V} \quad (3.24)$$

where ϕ_i is the flow variable at a vertex i of the tetrahedron and S_i is the surface vector that is opposite to node i , V is the volume of the tetrahedron. Gradients at the vertices are obtained by a volume averaging of the gradients at the center of cells associated with the vertex under consideration.

3.4 UPWIND-BIASED INTERPOLATION

Here an edge-based method of calculating the total inviscid flux is adopted by calculating and storing the flux integrals based on the edges [137]. Such a treatment leads to higher efficiency in computation and reduced data storage requirement.

The left and right state vectors $\mathbf{W}_L$ and $\mathbf{W}_R$ at a control volume surface are evaluated using a nominally third-order upwind-biased interpolation scheme. If the left and right state vectors are set to $\mathbf{W}_i$ and $\mathbf{W}_j$ (i and j corresponding to the two end nodes of an edge), it is a first-order upwind scheme, which are shown as follows,

$$\mathbf{W}_L = \mathbf{W}_i + \frac{1}{4} [(1 - \kappa) \Delta_i^- + (1 + \kappa) \Delta_i^+] \quad (3.25a)$$

$$\mathbf{W}_R = \mathbf{W}_j - \frac{1}{4} [(1 - \kappa) \Delta_j^+ + (1 + \kappa) \Delta_j^-] \quad (3.25b)$$

where

$$\Delta_i^+ = \Delta_j^- = \mathbf{W}_j - \mathbf{W}_i$$

$$\Delta_i^- = \mathbf{W}_i - \mathbf{W}_{i-1} = 2\vec{ij} \cdot \nabla \mathbf{W}_i - (\mathbf{W}_j - \mathbf{W}_i) = 2\vec{ij} \cdot \nabla \mathbf{W}_i - \Delta_i^+$$

$$\Delta_j^+ = \mathbf{W}_{j+1} - \mathbf{W}_j = 2\vec{ij} \cdot \nabla \mathbf{W}_j - (\mathbf{W}_j - \mathbf{W}_i) = 2\vec{ij} \cdot \nabla \mathbf{W}_j - \Delta_j^-$$

Therefore, substituting the above equations into Equations (3.25a) and (3.25b), and obtain the final equations based on upwind-biased interpolation scheme is,

$$\mathbf{W}_L = \mathbf{W}_i + \frac{1}{2} [(1-\kappa) \vec{ij} \cdot \nabla \mathbf{W}_i + \kappa \Delta_i^+] \quad (3.26a)$$

$$\mathbf{W}_R = \mathbf{W}_j - \frac{1}{2} [(1-\kappa) \vec{ij} \cdot \nabla \mathbf{W}_j + \kappa \Delta_j^-] \quad (3.26b)$$

where κ is set to 1/3, which corresponds to a nominally third-order accuracy.

3.5 UPWIND CHARACTERISTICS-BASED METHOD

The Euler equation modified by the Chorin's method [18] are rewritten in partial differential form in Cartesian co-ordinate system for the derivation of the method of characteristics:

$$\frac{\partial p}{\partial \tau} + \beta \frac{\partial u_i}{\partial x_i} = 0 \quad (3.27)$$

$$\frac{\partial u_i}{\partial \tau} + u_j \frac{\partial u_i}{\partial x_j} + u_i \frac{\partial u_j}{\partial x_j} + \frac{\partial p}{\partial x_i} = 0 \quad (3.28)$$

where subscripts i and j equal 1, 2 and 3 representing three co-ordinates. Suppose that ξ is a new co-ordinate outward normal to the surface of a control volume that surrounds a particular edge. In order to extend the method of characteristics to the three-dimensional unstructured grid solver, it is assumed that flow in the ξ direction is approximately one-dimensional and the preceding equations can then be transformed into,

$$\frac{\partial p}{\partial \tau} + \beta \frac{\partial u_j}{\partial \xi} \xi_{x_j} = 0 \quad (3.29)$$

$$\frac{\partial u_i}{\partial \tau} + u_j \frac{\partial u_i}{\partial \xi} \xi_{x_j} + u_i \frac{\partial u_j}{\partial \xi} \xi_{x_j} + \frac{\partial p}{\partial \xi} \xi_{x_i} = 0 \quad (3.30)$$

where $\xi_{x_i} = \frac{\partial \xi}{\partial x_i}$ and $\xi_{x_j} = \frac{\partial \xi}{\partial x_j}$.

In the $\tau - \xi$ space as shown in Figure 3.2, flow variable $\mathbf{W}$ at pseudo time level $m + 1$ can be calculated along a characteristics, k , using a Taylor series expansion and the initial value at time level m ($\mathbf{W}^k$),

$$\mathbf{W} = \mathbf{W}^k + \mathbf{W}_\xi \xi_\tau \Delta \tau + \mathbf{W}_\tau \Delta \tau \quad (3.31)$$

and

$$\mathbf{W}_\tau = \frac{\mathbf{W} - \mathbf{W}^k}{\Delta \tau} - \mathbf{W}_\xi \xi_\tau \quad (3.32)$$

A wave speed, λ^k , is introduced

$$\xi_\tau = \lambda^k \sqrt{\xi_{x_i} \xi_{x_i}}$$

and the unit normal vector components are

$$n_{x_j} = \frac{\xi_{x_j}}{\sqrt{\xi_{x_i} \xi_{x_i}}}$$

Figure 3.2 τ - ξ Co-ordinate.

Substitute components of $\mathbf{W}_\tau$ into Equations (3.29) and (3.30), we have

$$\frac{1}{\sqrt{\xi_{x_i} \xi_{x_i}}} \frac{(p - p^k)}{\Delta \tau} - p_\xi \lambda^k + \beta(u_\xi n_x + v_\xi n_y + w_\xi n_z) = 0 \quad (3.33)$$

$$\frac{1}{\sqrt{\xi_{x_i} \xi_{x_i}}} \frac{(u - u^k)}{\Delta \tau} - u_\xi (\lambda^0 - \lambda^k) + u(u_\xi n_x + v_\xi n_y + w_\xi n_z) + p_\xi n_x = 0 \quad (3.34)$$

$$\frac{1}{\sqrt{\xi_{x_j} \xi_{x_j}}} \frac{(v - v^k)}{\Delta \tau} - v_\xi (\lambda^0 - \lambda^k) + v(u_\xi n_x + v_\xi n_y + w_\xi n_z) + p_\xi n_y = 0 \quad (3.35)$$

$$\frac{1}{\sqrt{\xi_{x_j} \xi_{x_j}}} \frac{(w - w^k)}{\Delta \tau} - w_\xi (\lambda^0 - \lambda^k) + w(u_\xi n_x + v_\xi n_y + w_\xi n_z) + p_\xi n_z = 0 \quad (3.36)$$

where λ^0 is the contra-variant velocity,

$$\lambda^0 = un_x + vn_y + wn_z$$

In order to derive the compatibility equations, the spatial derivatives, such as u_ξ , v_ξ , w_ξ and p_ξ have to be eliminated from the preceding equations. Following the approach of Eberle [138] for compressible flow equations, each of the preceding four equations is multiplied by an arbitrary variable and all the resulting equations are summed to form a new equation as follows,

$$\frac{1}{\Delta\tau\sqrt{\xi_{x_j}\xi_{x_j}}}A - p_\xi B + u_\xi C + v_\xi D + w_\xi E = 0 \quad (3.37)$$

where

$$\begin{aligned} A &= a(p - p^k) + b(u - u^k) + c(v - v^k) + d(w - w^k) \\ B &= -a\lambda^k + bn_x + cn_y + dn_z \\ C &= an_x\beta + b(\lambda^0 - \lambda^k + un_x) + cvn_x + dwn_x \\ D &= an_y\beta + bun_y + c(\lambda^0 - \lambda^k + vn_y) + dwn_y \\ E &= an_z\beta + bun_z + cvn_z + d(\lambda^0 - \lambda^k + wn_z) \end{aligned} \quad (3.38)$$

a , b , c and d are the arbitrary variables used to multiply the equations. We define the coefficients of the partial space derivatives to be zero, i.e., A , B , C , D and E are zero,

$$A = a(p - p^k) + b(u - u^k) + c(v - v^k) + d(w - w^k) = 0 \quad (3.39)$$

$$B = 0 \quad (3.40)$$

$$C = 0 \quad (3.41)$$

$$D = 0 \quad (3.42)$$

$$E = 0 \quad (3.43)$$

Equations (3.39) to (3.43) constitute a linear system $\Phi X = 0$ with $X = \{a, b, c, d\}$. Variables a , b , c and d are generally non-zero. Thus the system of equations has non-trivial solution. This means that $\det(\Phi) = 0$, and the following eigenvalues can be derived as:

$$\begin{aligned} \lambda^0 &= un_x + vn_y + wn_z \\ \lambda^1 &= \lambda^0 + \sqrt{(\lambda^0)^2 + \beta} = \lambda^0 + C \\ \lambda^2 &= \lambda^0 - \sqrt{(\lambda^0)^2 + \beta} = \lambda^0 - C \end{aligned}$$

For each eigenvalue or characteristics speed, characteristic equations can be derived from Equations (3.39) to (3.43). For example, for $\lambda^k = \lambda^0$, we have,

$$a = \frac{bn_x + cn_y + dn_z}{\lambda^0}$$

Substitute this into Equation (3.38), we obtain,

$$\frac{bn_x + cn_y + dn_z}{\lambda^0} (p - p^0) + b(u - u^0) + c(v - v^0) + d(w - w^0) = 0$$

that is,

$$b[n_x(p - p^0) + \lambda^0(u - u^0)] + c[n_y(p - p^0) + \lambda^0(v - v^0)] + d[n_z(p - p^0) + \lambda^0(w - w^0)] = 0$$

For any b , c and d , the preceding equations are always satisfied. Therefore, all the terms in square brackets are zero. As a result, we have,

$$(u - u^0)n_y - (v - v^0)n_x = 0 \quad (3.45)$$

$$(v - v^0)n_z - (w - w^0)n_y = 0 \quad (3.46)$$

$$(u - u^0)n_z - (w - w^0)n_x = 0 \quad (3.47)$$

For $\lambda = \lambda^1$,

$$p - p^1 = -\lambda^1[(u - u^1)n_x + (v - v^1)n_y + (w - w^1)n_z] \quad (3.48)$$

For $\lambda = \lambda^2$,

$$p - p^2 = -\lambda^2[(u - u^2)n_x + (v - v^2)n_y + (w - w^2)n_z] \quad (3.49)$$

Finally, u , v , w and p are determined using the preceding characteristics Equations (3.45) to (3.49):

$$u = fn_x + u^0(n_y^2 + n_z^2) - v^0n_xn_y - w^0n_xn_z$$

$$v = fn_y + v^0(n_x^2 + n_z^2) - w^0n_y n_z - u^0n_y n_x$$

$$w = fn_z + w^0(n_x^2 + n_y^2) - u^0n_z n_x - v^0n_z n_y$$

$$p = p^1 - \lambda^1 [(u - u^1)n_x + (v - v^1)n_y + (w - w^1)n_z]$$

or

$$p = p^2 - \lambda^2 [(u - u^2)n_x + (v - v^2)n_y + (w - w^2)n_z]$$

where

$$C = \sqrt{(\lambda^0)^2 + \beta}$$

$$f = \frac{1}{2C} [(p^1 - p^2) + n_x(\lambda^1 u^1 - \lambda^2 u^2) + n_y(\lambda^1 v^1 - \lambda^2 v^2) + n_z(\lambda^1 w^1 - \lambda^2 w^2)]$$

Flow quantities at $m+1$ pseudo time level obtained from the preceding equations on the characteristics are then used to calculate convection fluxes at the control volume interface. Those on different characteristics at m time level are approximately evaluated by an upwind scheme using the signs of the characteristics as suggested by Drikakis *et al.* [139].

$$\mathbf{W}^j = \frac{1}{2} [(1 + \text{sign}(\lambda^j))\mathbf{W}_L + (1 - \text{sign}(\lambda^j))\mathbf{W}_R]$$

where $\mathbf{W}_L$ and $\mathbf{W}_R$ are obtained by the upwind-biased interpolation.

The advantages of the characteristics scheme are: (i) Stable solution with adding artificial viscosity; (ii) Less sensitive to grid orientation because flow signals are propagated along characteristics.

3.6 MATRIX-FREE IMPLICIT DUAL TIME-STEPPING ALGORITHM

To enhance computational efficiency, implicit schemes can be used to permit largest possible time-step sizes, taking into considerations of accuracy. However, most implicit schemes require a large amount of computing effort per time step compared with explicit schemes, making them less attractive for many unsteady flow computations. In this work, a matrix-free implicit scheme is derived, which is found to be efficient in terms of memory requirement and computing effort per time step. This is due to the fact that no matrix manipulation is required by the scheme.

By re-writing equation (3.17) for a given node P , the spatially discretized equations form a system of coupled ordinary differential equations, which can be reformulated as,

$$\begin{aligned} \mathbf{C} \frac{\partial}{\partial \tau} (\Delta V_{cv} \mathbf{W}_p) + \mathbf{K} \frac{\partial}{\partial t} (\Delta V_{cv} \mathbf{W}_p) &= - \left\{ \sum_{n=1}^{nbseg} [(\vec{\mathbf{F}}_e)_{ij} \cdot \vec{\mathbf{n}} \Delta S]_n - \frac{1}{3} \sum_{i=1}^{ncell} (\vec{\mathbf{F}}_v \cdot \vec{\mathbf{n}} \Delta S_p)_i \right\} \\ &= -R(\mathbf{W}_p) \end{aligned} \quad (3.50)$$

where $R(\mathbf{W}_p)$ represents the residual error, or deviation from steady state, which includes the convective and diffusive fluxes and ΔV_{cv} is the control volume of node P .

For convenience, the pseudo time derivative term is ignored at the moment. An implicit scheme is adopted to approximate Equation (3.50) and the semi-discrete equation is shown as follows,

$$\mathbf{K} \frac{\partial}{\partial t} (\mathbf{V}_{cv}^{n+1} \mathbf{W}_p^{n+1}) = -\mathbf{R}(\mathbf{W}_p^{n+1}) \quad (3.51)$$

The superscript $(n+1)$ denotes the time level $(n+1)\Delta t$ and all the variables are evaluated at this time level. In this work, $\frac{\partial}{\partial t}$ is discretized as a second-order accurate backward difference, so that Equation (3.51) is re-formulated as follows,

$$\mathbf{K} \left(\frac{1.5 \Delta \mathbf{V}_{cv}^{n+1} \mathbf{W}_p^{n+1} - 2.0 \Delta \mathbf{V}_{cv}^n \mathbf{W}_p^n + 0.5 \Delta \mathbf{V}_{cv}^{n-1} \mathbf{W}_p^{n-1}}{\Delta t} \right) + \mathbf{R}(\mathbf{W}_p^{n+1}) = \tilde{\mathbf{R}}(\mathbf{W}_p^{n+1}) = 0 \quad (3.52)$$

where $\tilde{\mathbf{R}}(\mathbf{W}_p^{n+1})$ is the new modified residual, which contains both the time derivative and flux vectors. The advantage of the preceding implicit scheme is that the physical time step size is not restricted by numerical stability, but only by numerical accuracy. This is especially useful in unsteady flow simulation where the maximum time-step size is much smaller than the size permitted by accuracy consideration. From Equation (3.52), it is worthwhile to note that there is no second-order accurate backward difference term in the conservation of mass and this is because Equation (3.8) does not have a time derivative term. And a pseudo-unsteady pressure term is added for the purpose of transforming the conservation of mass into a hyperbolic equation.

Following Jameson's work [23], the derivative with respect to a fictitious pseudo time τ , is added to the Equation (3.52) and the unsteady Navier-Stokes equations can be re-formulated as,

$$\mathbf{C}\Delta V_{cv}^{n+1} \frac{d\mathbf{W}_p^{n+1,m}}{d\tau} = -\tilde{\mathbf{R}}(\mathbf{W}_p^{n+1,m}) \quad (3.53)$$

whose solution is sought by marching to a pseudo steady state in τ . Here m denotes the pseudo time level $m\Delta\tau$. Once the artificial steady state is reached, the derivative of $\mathbf{W}_p$ with respect to τ becomes zero, and the solution satisfies $\tilde{\mathbf{R}}(\mathbf{W}_p^{n+1}) = 0$. This is actually the solution of Equation (3.52). Hence, the original unsteady Navier-Stokes equations are fully recovered. Therefore, instead of solving each time step in the physical time domain (t), the problem is transformed into a sequence of steady-state computations in the artificial time domain (τ). This can be performed using a pseudo time explicit five-stage Runge-Kutta scheme [140,141]. However, the pseudo time step size may be severely restricted if the physical time step size is very small. In view of this, an implicit dual time stepping scheme is adopted in this work.

Equation (3.53) can be re-formulated as,

$$\mathbf{C}\Delta V_{cv}^{n+1} \frac{\mathbf{W}_p^{n+1,m+1} - \mathbf{W}_p^{n+1,m}}{d\tau} = -\tilde{\mathbf{R}}(\mathbf{W}_p^{n+1,m}) \quad (3.54)$$

An approximate flux function is introduced here to simplify the implicit time stepping calculation. Following [142], the total flux (including both inviscid and viscous fluxes) across a control volume surface associated with a certain edge ij can be approximated as,

$$\mathbf{F}_{ij} \approx \frac{1}{2} [\bar{\mathbf{F}}_i^c \cdot \bar{\mathbf{n}} + \bar{\mathbf{F}}_j^c \cdot \bar{\mathbf{n}} - |\lambda_{ij}| (\mathbf{W}_i - \mathbf{W}_j)]$$

where λ_{ij} is the spectral radius associated with edge ij , is given as,

$$\lambda_{ij} = \vec{\mathbf{U}} \cdot \vec{\mathbf{n}}_{ij} + \sqrt{(\vec{\mathbf{U}} \cdot \vec{\mathbf{n}}_{ij})^2 + \beta^2}$$

A Taylor series expansion is performed for the residual in Equation (3.54) with respect to the pseudo time for node i ,

$$\tilde{\mathbf{R}}(\mathbf{W}_i^{m+1}) = \tilde{\mathbf{R}}(\mathbf{W}_i^m) + \frac{\partial \tilde{\mathbf{R}}_i}{\partial \mathbf{W}_i} + \sum_{j=1}^m \frac{\partial \tilde{\mathbf{R}}_i}{\partial \mathbf{W}_j} \Delta \mathbf{W}_j$$

And in all the $\mathbf{R}$ terms, the Taylor series expansions of the fluxes are,

$$\frac{\partial \mathbf{F}_{ij}}{\partial \mathbf{W}_i} = \frac{1}{2} \left[\frac{\partial \mathbf{F}_{ij}^c}{\partial \mathbf{W}_i} - |\lambda_{ij}| \right] \quad \text{and} \quad \frac{\partial \mathbf{F}_{ij}}{\partial \mathbf{W}_j} = \frac{1}{2} \left[\frac{\partial \mathbf{F}_{ij}^c}{\partial \mathbf{W}_j} + |\lambda_{ij}| \right]$$

And for the physical time-dependent terms, we have

$$\mathbf{W}_i^{m+1} = \mathbf{W}_i^m + \Delta \mathbf{W}_i$$

After combining all the residuals at every node in the flow field into a vector, we have

$$\mathbf{R}(\mathbf{W}^{n+1,m+1}) = \mathbf{R}(\mathbf{W}^{n+1,m}) + \mathbf{A} \Delta \mathbf{W}$$

where $\mathbf{A} = \left\{ \frac{\partial \mathbf{R}_i}{\partial \mathbf{W}_j} \right\}$.

And the whole-field equivalent Equation (3.54) can then be re-written as,

$$\mathbf{C}\Delta V_{cv}^{n+1} \left(\frac{\Delta t + 1.5\Delta\tau}{\Delta t} - \frac{A\Delta\tau}{\Delta V_{cv}^{n+1}} \right) \frac{\Delta \mathbf{W}}{\Delta\tau} = -\mathbf{R}^{n+1,m} - \mathbf{K} \left(\frac{1.5\Delta V_{cv}^{n+1} \mathbf{W}^{n+1,m} - 2.0\Delta V_{cv}^n \mathbf{W}^n + 0.5\Delta V_{cv}^{n-1} \mathbf{W}^{n-1}}{\Delta t} \right)$$

that is,
$$\mathbf{C}\Delta V_{cv}^{n+1} \tilde{\mathbf{A}} \frac{\Delta \mathbf{W}}{\Delta\tau} = \tilde{\mathbf{R}}^{n+1,m} \quad (3.55)$$

thus,
$$\mathbf{C}\Delta V_{cv}^{n+1} \frac{\Delta \mathbf{W}}{\Delta\tau} = \tilde{\tilde{\mathbf{R}}}^{n,m+1} \quad (3.56)$$

where
$$\tilde{\tilde{\mathbf{R}}}^{n,m+1} = \tilde{\mathbf{A}}^{-1} \tilde{\mathbf{R}}^{n+1,m} \quad \text{and} \quad \tilde{\mathbf{A}} = \frac{\Delta t + 1.5\Delta\tau}{\Delta t} - \frac{A\Delta\tau}{\Delta V_{cv}}$$

Therefore,
$$\tilde{\mathbf{R}}^{n+1,m} = -\mathbf{R}^{n+1,m} - \mathbf{K} \left(\frac{1.5\Delta V_{cv}^{n+1} \mathbf{W}^{n+1,m} - 2.0\Delta V_{cv}^n \mathbf{W}^n + 0.5\Delta V_{cv}^{n-1} \mathbf{W}^{n-1}}{\Delta t} \right)$$

Further approximation can be introduced in order to achieve matrix-free computation. If we employ point implicit treatment to the preceding equations, then only the diagonal terms in $\tilde{\mathbf{A}}$ are used in the pseudo time stepping. As a result, the equation for every node can now be written as,

$$\mathbf{C}\Delta V_{cv_i}^{n+1} \frac{\Delta \mathbf{W}_i}{\Delta\tau} = \tilde{\tilde{\mathbf{R}}}_i^{n,m+1} \quad (3.57)$$

where
$$\tilde{\tilde{\mathbf{R}}}_i^{n,m+1} = \tilde{\mathbf{A}}_{ii}^{-1} \tilde{\mathbf{R}}_i^{n,m+1} \quad \text{and} \quad \tilde{\mathbf{A}}_{ii}^{-1} = \left(\frac{\Delta t + 1.5\Delta\tau}{\Delta t} - \frac{A_{ii}\Delta\tau}{\Delta V_{cv_i}} \right)^{-1}$$

Pseudo time stepping is then performed on Equation (3.57).

3.7 TIME INTEGRATION

In this work, a five-stage Runge-Kutta time integration algorithm is used between each physical time step to iterate the numerical solution in an artificial time τ until convergence is reached [17,23]. Therefore, the converged solution from the artificial steady-state equations becomes the time accurate solution at current physical time, t . To advance the solution in time from m to $m+1$, the formulation of a five-stage Runge-Kutta scheme is as follow,

$$\begin{aligned}
 \mathbf{W}_p^{(0)} &= \mathbf{W}_p^m \\
 \mathbf{W}_p^{(1)} &= \mathbf{W}_p^{(0)} - \alpha_1 \mathbf{C} \frac{\Delta \tau}{\Delta V_{cv}} \tilde{\tilde{\mathbf{R}}}(\mathbf{W}_p^{(0)}) \\
 \mathbf{W}_p^{(2)} &= \mathbf{W}_p^{(0)} - \alpha_2 \mathbf{C} \frac{\Delta \tau}{\Delta V_{cv}} \tilde{\tilde{\mathbf{R}}}(\mathbf{W}_p^{(1)}) \\
 \mathbf{W}_p^{(3)} &= \mathbf{W}_p^{(0)} - \alpha_3 \mathbf{C} \frac{\Delta \tau}{\Delta V_{cv}} \tilde{\tilde{\mathbf{R}}}(\mathbf{W}_p^{(2)}) \\
 \mathbf{W}_p^{(4)} &= \mathbf{W}_p^{(0)} - \alpha_4 \mathbf{C} \frac{\Delta \tau}{\Delta V_{cv}} \tilde{\tilde{\mathbf{R}}}(\mathbf{W}_p^{(3)}) \\
 \mathbf{W}_p^{(5)} &= \mathbf{W}_p^{(0)} - \alpha_5 \mathbf{C} \frac{\Delta \tau}{\Delta V_{cv}} \tilde{\tilde{\mathbf{R}}}(\mathbf{W}_p^{(4)}) \\
 \mathbf{W}_p^{(m+1)} &= \mathbf{W}_p^{(5)}
 \end{aligned} \tag{3.58}$$

where the stage coefficients for a five-stage Runge-Kutta time integration is as follows,

$$\alpha_1 = \frac{1}{4}, \quad \alpha_2 = \frac{1}{6}, \quad \alpha_3 = \frac{3}{8}, \quad \alpha_4 = \frac{1}{2}, \quad \alpha_5 = 1$$

3.8 CONVERGENCE ACCELERATION TECHNIQUES

In the present code, time-dependent calculations require the convergence of the Navier-Stokes equations to the steady state in pseudo-time for each real time step. The methods used to accelerate convergence to steady state in pseudo-time are local time stepping, implicit residual smoothing, and multigrid method. Since these convergence acceleration techniques destroy the time accuracy in pseudo-time and they are only used to accelerate convergence in pseudo-time, they will not affect the accuracy in physical time. In the dual time-stepping algorithm, global time stepping is used to advance the solution in the real time whereas local time stepping is used to advance the solution in the pseudo-time.

3.8.1 Local Time Stepping in Pseudo Time

The maximum permissible time step for an explicit time stepping scheme is determined by a stability criterion of the numerical method. The Courant-Friedrichs-Lewy (CFL) condition [133,143] for hyperbolic equations can be used to derive such a condition by analysing a linearised model equation. Due to the disparity in grid cell sizes for viscous calculations, the maximum time step for the entire mesh will be the minimum of the time steps of all the control volumes for time accurate calculations. In this work, the large variation in grid size for the unstructured mesh will restrict the time step used and the smallest control volume dictates the maximum time step. To overcome the above problems, each control volume can be advanced in time by its own maximum local time step, which greatly enhanced the convergence rate.

In this study, the local time step size is estimated via CFL stability condition as,

$$\Delta\tau = \text{CFL} \cdot \frac{\Delta l}{|\vec{U}| + c} \quad (3.59)$$

where Δl is the length scale associated with a node under consideration. Normally, it is taken as the smallest height of all the tetrahedral cells sharing the node. For global time stepping, Δl is taken as the smallest value of all the nodes on the domain. And c is the speed of sound.

3.8.2 Implicit Residual Smoothing

To speed up the convergence rate, an implicit residual smoothing scheme developed for unstructured grids was employed. The idea behind this is to replace the residual at one point of flow field with a smoothed or weighted average of the residuals at the neighbouring points [143]. The averaged residuals are calculated implicitly in order to increase the maximum CFL number, thus increasing the convergence rate. Normally this procedure allows the CFL number to be increased by a factor of 2 or 3. The smoothing equation for a vertex k can be expressed as follows,

$$\bar{R}_k = R_k + \varepsilon \nabla^2 \bar{R}_k \quad (3.60)$$

where $\bar{R}$ is the smoothed residual, R is the original residual, and ε is the smoothing coefficient, which can be defined as,

$$\varepsilon = \max \left\{ \frac{1}{4} \left[\left(\frac{\text{CFL}}{\text{CFL}^*} \right)^2 - 1 \right], 0 \right\} \quad (3.61)$$

where CFL^* is the maximum CFL number of the basic scheme. The solution to the preceding equations can be obtained on an unstructured grid by using the Jacobi iterative method as follows,

$$\begin{aligned} \overline{R}_k^{(m)} &= R_k^{(0)} + \varepsilon \sum_{i=l}^{numnod(k)} [\overline{R}_i^{(m)} - \overline{R}_k^{(m)}] \\ \text{i.e.} \quad \overline{R}_k^{(m)} &= \frac{R_k^{(0)} + \varepsilon \sum_{i=l}^{numnod(k)} \overline{R}_i^{(m-1,m)}}{1 + \varepsilon \cdot numnod(k)} \end{aligned} \quad (3.62)$$

where $numnod(k)$ is the number of neighboring nodes of vertex k .

3.9 BOUNDARY CONDITIONS

At the solid wall, slip (for inviscid flows) or no-slip (for viscous flows) and no-injection boundary conditions are imposed, that is, the zero normal fluxes of mass, momentum, and energy are imposed. In addition, the solid surface is assumed to be adiabatic, and the pressure gradient normal to the wall at the surface is considered to be zero.

For a far-field or upstream boundary, the flow velocity is given directly, and gradients of variables are assumed to be zero. The pressure at the upstream was calculated while downstream pressure is fixed at a constant value.

CHAPTER FOUR

COMPUTATIONAL STRUCTURAL DYNAMICS

The chapter presents a structural analysis for thin structures immersed in unsteady viscous flows and the numerical algorithms developed to solve the structural equations. The governing equation is the Newton's Second Law of Motion for structural elements under the influence of fluid forces. The equation is discretized over beam elements (2D structures) or triangular shell elements (3D structures). Discrete equations are solved by the dual-time stepping scheme using the five-stage Runge-Kutta method for pseudo time stepping. In section one, the numerical methods are derived for the two-dimensional immersed membranes based on beam elements. In section two, the structural governing equation is outlined for a bi-leaflet mechanical heart valve. Lastly, the structural dynamics equation and numerical methods are derived for bioprosthetic heart valves based on 3D triangular shell elements in section three.

4.1 TWO-DIMENSIONAL IMMERSED MEMBRANE

4.1.1 Governing Equations

For 2D fluid-structure interaction problems, beam elements as used by Cai and Luo [144] are adopted to discretize membrane-like structures immersed in flow fields. Considering the motion of one element of the immersed membrane, the forces acting on it are sketched in Figure 4.2. T_{right} and T_{left} are the tension forces acting on the left and right ends of the element. p_1 and p_2 are fluid pressures on both sides of the immersed structure. σ_t is viscous shear stress and σ_n the fluid normal stress. M_1 and M_2 are bending moments acting on both ends of the element. With these forces and moments, the governing equations for one element are given as:

$$\Delta m \frac{d^2 \mathbf{r}}{dt^2} = \Sigma \mathbf{F} \quad (4.1)$$

$$\Delta I \frac{d^2 \theta}{dt^2} = \Sigma \mathbf{M} \quad (4.2)$$

and,

$$\Sigma \mathbf{M} = \mathbf{M}_1 - \mathbf{M}_2 + \Sigma \mathbf{F} \times \mathbf{r} \quad (4.3)$$

$$\Sigma \mathbf{F} = (p_1 \Delta l - p_2 \Delta l) \mathbf{n} + (\sigma_{t1} \Delta l - \sigma_{t2} \Delta l) \mathbf{t} + (\sigma_{n1} \Delta l - \sigma_{n2} \Delta l) \mathbf{n} + (T_{right} \mathbf{t}_r - T_{left} \mathbf{t}_l) d \quad (4.4)$$

where $\mathbf{r}$ is the position vector of the membrane, $\mathbf{n}$ is the unit vector in normal direction originating in the center of mass of the element, $\mathbf{t}$ is the unit vector in tangential direction. $\mathbf{t}_l$ and $\mathbf{t}_r$ are the unit vectors in the left and right directions of the membrane.

Δm is the mass of the membrane element and can be written as $\Delta m = \rho_m w d \Delta l$, where ρ_m is the density of the membrane. And w is the unit width, d the unit thickness

Figure 4.1 Forces and moments acting on an element of the immersed membrane

and Δl the length of the membrane element. ΔI is the inertia of moment with respect to the center of mass of the element. It is given as,

$$\Delta I = \frac{\Delta m \Delta l^2}{12} \quad (4.5)$$

The structure equations are non-dimensionalized in the same manner as the fluid equations as follows,

$$\begin{aligned}
 T'_i &= \frac{T_i}{\rho U_\infty^2 L} & \sigma'_n &= \frac{\sigma_n}{\rho U_\infty^2} & \sigma'_t &= \frac{\sigma_t}{\rho U_\infty^2 L} & \Delta l' &= \frac{\Delta l}{L} \\
 p'_i &= \frac{p_i}{\rho U_\infty^2} & \mathbf{r}' &= \frac{\mathbf{r}}{L} & \Delta m' &= \frac{\Delta m}{\rho L L} = \frac{\rho_m w d \Delta l}{\rho L L} \equiv m' \\
 \mathbf{M}' &= \frac{\mathbf{M}}{\rho U_\infty^2 L^3} & \Delta I' &= \frac{\Delta I}{\rho L^3 L^2} = \frac{\Delta I}{\rho L^5} \equiv I'
 \end{aligned}$$

After dropping the primes, the structure equations can be given as,

$$m \frac{d^2 \mathbf{r}}{dt^2} = (p_1 \Delta l - p_2 \Delta l) \mathbf{n} + (\sigma_{t1} \Delta l - \sigma_{t2} \Delta l) \mathbf{t} + (\sigma_{n1} \Delta l - \sigma_{n2} \Delta l) \mathbf{n} + (T_{right} \mathbf{t}_r - T_{left} \mathbf{t}_l) d \quad (4.6)$$

$$I \frac{d^2 \theta}{dt^2} = \mathbf{M}_1 - \mathbf{M}_2 + \Sigma \mathbf{F} \times \mathbf{r} \quad (4.7)$$

Assuming the immersed structure is linearly elastic, the tension forces can be obtained by,

$$T = E \varepsilon w = E w \left(\frac{dl}{\Delta l_0} \right) = E w \left(\frac{\Delta l - \Delta l_0}{\Delta l_0} \right)$$

For unit width w ,

$$T = E \left(\frac{\Delta l - \Delta l_0}{\Delta l_0} \right) \quad (4.8)$$

where Δl_0 is the initial length of the element and Δl its current length, and E is the Young's Modulus of the structure. It is non-dimensionalized by

$$E' = \frac{E}{\rho U_{\infty}^2}$$

Using the Kirchhoff Laws for a thin wall structure, the bending moments can be calculated by,

$$\mathbf{M} = -\frac{EJ}{1-\nu^2} \mathbf{k} \quad (4.9)$$

Figure 4.2 Computation of curvature for a structural element

where $\mathbf{k}$ is the curvature of the element, and ν is the Poisson ratio, J the geometric moment of inertia which is given as,

$$J = \frac{wh^3}{12} \quad (4.10)$$

where w is the unit width, and h the thickness of the membrane. Curvature $\mathbf{k}$ of the element is equal to the reciprocal of radius of curvature R as sketched in Figure 4.2.

i.e., $|k| = \frac{1}{R}$

and, $R = \frac{\Delta l}{\theta}$

$$\theta = \arccos(|-\mathbf{t}_l \bullet \mathbf{t}_r|)$$

so the curvature k can be given as,

$$|k| = \frac{\arccos(|-\mathbf{t}_l \bullet \mathbf{t}_r|)}{\Delta l} \quad (4.11)$$

With the above derivations, all the possible forces and moments have been considered to define the movement of the structural element.

4.1.2 Discretization of The Structural Equations

The immersed membrane is evenly discretized into a finite number of elements, and the structural equations are spatially discretized over every element of the immersed membrane. The implicit dual time stepping scheme described in section 2.2 is applied here to solve the structural equations.

Considering the left hand sides of equations (4.1) and (4.2), if $\mathbf{u}$ is the translation velocity vector and $\boldsymbol{\omega}$ is the rotation velocity vector, where $\mathbf{u} = d\mathbf{r}/dt$ and $\boldsymbol{\omega} = d\boldsymbol{\theta}/dt$, then

$$\frac{d^2 \mathbf{r}}{dt^2} = \frac{d\mathbf{u}}{dt} = \frac{\sum \mathbf{F}}{m}$$

$$\frac{d^2\theta}{dt^2} = \frac{d\omega}{dt} = \frac{\sum M}{I} \quad (4.12)$$

Introducing the pseudo time τ , the two equations can be transformed into the followings:

$$\begin{aligned} \frac{d\mathbf{u}}{d\tau} &= \frac{\sum \mathbf{F}}{m} - \frac{d\mathbf{u}}{dt} = \mathbf{RHS} \\ \frac{d\omega}{d\tau} &= \frac{\sum M}{I} - \frac{d\omega}{dt} = \mathbf{RHS} \end{aligned} \quad (4.13)$$

The second order backward difference scheme is used for the time differences in the above equation as:

$$\begin{aligned} \frac{d\mathbf{u}}{dt} &= \frac{1.5\mathbf{u}^{n+1} - 2.0\mathbf{u}^n + 0.5\mathbf{u}^{n-1}}{\Delta t} \\ \frac{d\omega}{dt} &= \frac{1.5\omega^{n+1} - 2.0\omega^n + 0.5\omega^{n-1}}{\Delta t} \end{aligned} \quad (4.14)$$

Finally, equations (4.13) can be written as:

$$\begin{aligned} \frac{\mathbf{u}^{n+1,m+1} - \mathbf{u}^{n+1,m}}{\Delta \tau} &= \mathbf{RHS}^{n+1,m} \\ \frac{\omega^{n+1,m+1} - \omega^{n+1,m}}{\Delta \tau} &= \mathbf{RHS}^{n+1,m} \end{aligned} \quad (4.15)$$

where $n+1$ is assigned to the physical time level t^{n+1} , $m+1$ to the pseudo time level τ^{m+1} and m to the pseudo time level τ^m . The above equations are integrated in pseudo time by a five stage Runge-Kutta method in sub-iterations in order to approach $\mathbf{RHS}=0$, which will recover the original structure equations (4.1) and (4.2).

The solution of equations (4.15) gives two velocities of the structure element: the translation velocity $\mathbf{u}$ and rotation velocity $\boldsymbol{\omega}$, so the total velocity of the structure node $\mathbf{P}$ at the end of the element is given by,

$$\mathbf{u}^P = \mathbf{u} + \boldsymbol{\omega} \Delta l / 2.0 \quad (4.16)$$

In order to keep the continuity of the membrane nodes, a smoothing algorithm is implemented after obtaining the velocities of all the membrane points in each pseudo time step. The velocity of each membrane point $\mathbf{P}$ is recalculated as follows,

$$\mathbf{u}_{i+1}^P = \frac{\mathbf{u}_{i+1}^{P-1} + \mathbf{u}_i^P + \mathbf{u}_i^{P+1}}{3} \quad i = 1, 5 \quad (4.17)$$

4.2 BI-LEAFLET MECHANICAL HEART VALVE

A bi-leaflet MHV has two symmetric leaflets rotating around their hinges. The leaflets are considered rigid bodies, so they only have one degree of freedom, i.e., the angular deviation θ of the leaflets from an arbitrary initial angular position θ_0 . Under the influence of hemodynamic forces, the governing equation for the rotation of the leaflets can be written as follows,

$$\mathbf{I} \cdot \frac{d^2\theta}{dt^2} = \mathbf{M}_p + \mathbf{M}_\tau \quad (4.18)$$

where $\mathbf{I}$ is the moment of inertia of the leaflet, θ is the rotating angle of the leaflet, $\mathbf{M}_p$ and $\mathbf{M}_\tau$ are the moments acting on the leaflet about the pivotal axis generated by the pressure and the shear stress of the ambient fluids respectively, $\frac{d^2\theta}{dt^2}$ is the angular acceleration of the leaflet. The terms in Equation (4.18) are non-dimensionalized as follows,

$$\tilde{I} = \frac{I}{\rho_{\infty} L^5}; \quad \tilde{M} = \frac{M}{\rho_{\infty} U_{\infty}^2 L^3} \quad (4.19)$$

For convenience, the hat on top of the non-dimensional term is drop for all equations. Considering the forces acting on the surface of the leaflet as shown in Figure 4.5, the leaflet will rotate about the pivot axis AA' under the action of the moments generated by both the pressure and shear stress, which are given as follows,

$$M_p = \iint_{\Gamma} \mathbf{r} \times (p \cdot \mathbf{n}) \cdot dA \quad (4.20)$$

$$M_{\tau} = \iint_{\Gamma} \mathbf{r} \times (\bar{\tau} \cdot \mathbf{n}) \cdot dA \quad (4.21)$$

where Γ denotes the surface area of the leaflet, $\mathbf{r}$ is the distance vector which points from the pivot axis to the center of the control area and is perpendicular to the pivot axis, p is the pressure on the control area, $\bar{\tau}$ is the shear stress tensor on the small control area, $\mathbf{n}$ is the normal vector of the control area, and dA denotes the area of the small control area.

Figure 4.3 Hemodynamic forces acting on the surface of the valve leaflet [45].

The second order backward difference scheme is adopted for the temporal discretization of governing equation (4.18) as follows,

$$\frac{d^2\theta}{dt^2} = \frac{d\omega}{dt} = \frac{1.5\omega^{n+1} - 2.0\omega^n + 0.5\omega^{n-1}}{\Delta t} \quad (4.22)$$

And the dual time stepping scheme as in the previous section is used to solve the discretized governing equations.

4.3 BIOPROSTHETIC AORTIC HEART VALVE

4.3.1 Governing Equations

The universal governing equation for any continuum undergoing motion is given by the Cauchy's equation,

$$\mathbf{b} + \nabla \cdot \boldsymbol{\sigma}_{ij} = \rho \mathbf{a} \quad (4.23)$$

Where $\mathbf{b}$ is body force, $\boldsymbol{\sigma}_{ij}$ is the stress tensor defined for any continuum, ρ is the material density and $\mathbf{a}$ is the acceleration. This equation is variously described as the stress equation of small motion, the equation of equilibrium or the equation of motion in the theory of elasticity. In this work, the term equation of dynamic equilibrium is adopted to distinguish it from static structural problems. Discarding the body forces for heart valves and adding the influence of fluid forces, equation (4.23) is transformed into the follows:

$$\frac{\partial \mathbf{W}}{\partial t} = \nabla \cdot \boldsymbol{\sigma}_{ij} + \mathbf{f}_{ext} \quad (4.24)$$

and

$$\mathbf{W} = \begin{bmatrix} \rho u \\ \rho v \\ \rho w \end{bmatrix} \quad \boldsymbol{\sigma}_{ij} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} \quad (4.25)$$

where $\mathbf{f}_{ext}$ is the external forces per unit volume acting on the structure. $\mathbf{W}$ is the vector of dependent variables. The constitutive equation for stress and strain is the generalized Hook's laws, which states that the stress applied to any solid is proportional to the strain it produces within the elastic limit for that solid. The aortic valve model used in this work is assumed to be isotropic, so elasticity is uniform throughout the valve leaflets. The above equations are non-dimensionalized as followings:

$$\begin{aligned} \left(x', y', z' \right) &= \left(\frac{x}{D}, \frac{y}{D}, \frac{z}{D} \right); & \left(u', v', w' \right) &= \left(\frac{u}{U_{ref}}, \frac{v}{U_{ref}}, \frac{w}{U_{ref}} \right); \\ t' &= \frac{t}{D/U_{ref}}; & \mathbf{f}' &= \frac{\mathbf{f}_{ext}}{\rho_{ref}(U_{ref})^2}; & \sigma'_{ij} &= \frac{\sigma_{ij}}{\rho_{ref}(U_{ref})^2}; \end{aligned}$$

where D is the characteristic length of the model. U_{ref} is reference velocity. ρ_{ref} is reference material density. To maintain consistency for both the fluid domain and structure domain, the two governing equations (NS equation for fluids and equation of dynamic equilibrium for structures) are non-dimensionalized by the same set of parameters, i.e., using the same characteristic length, reference velocity and material density. The non-dimensional form of the governing equation can be written as:

$$\frac{\partial \mathbf{W}'}{\partial t'} = \nabla' \bullet \boldsymbol{\sigma}'_{ij} + \mathbf{f}' \quad (4.26)$$

Dropping the primes for simplicity, the non-dimensional governing equation recovers its original form of equation (4.24) as:

$$\frac{\partial \mathbf{W}}{\partial t} = \nabla \bullet \boldsymbol{\sigma}_{ij} + \mathbf{f}_{ext} \quad (4.27)$$

To adopt the finite volume scheme for structural analysis, the governing equation is transformed into intergral form as:

$$\frac{\partial}{\partial t} \iiint_V \mathbf{W} \, dV = \iiint_V \nabla \cdot \boldsymbol{\sigma}_{ij} \, dV + \oint_{S_{cv}} \mathbf{f}_{ext} \, dS \quad (4.28)$$

According to the Green's theorem:

$$\iiint_V \nabla \cdot \boldsymbol{\sigma}_{ij} \, dV = \oint_{S_{cv}} \boldsymbol{\sigma}_{ij} \cdot \mathbf{n} \, dS \quad (4.29)$$

The integral governing equation can be written as:

$$\frac{\partial}{\partial t} \iiint_V \mathbf{W} \, dV = \oint_{S_{cv}} \boldsymbol{\sigma}_{ij} \cdot \mathbf{n} \, dS + \oint_{S_{cv}} \mathbf{f}_{ext} \, dS \quad (4.30)$$

4.3.2 Triangular Shell Elements

The bioprosthetic aortic heart valve is a tri-leaflet structure. And the valve leaflets are very thin membranes, which can be represented by their mid-planes when considering the behavior of structural dynamics. The mid-planes can be treated as shell structures which can be discretized into an assembly of shell elements. In this work, the triangular shell elements are adopted to discretize the mid-planes, which are piecewise triangular flat facets. It is assumed that the behavior of a continuously curved surface can be adequately represented by the behavior of a surface built up of sufficiently dense flat facets [145], as shown in Figure 4.4.

Figure 4.4 Mesh for one leaflet of a bioprosthetic heart valve

4.3.3 Finite-Volume Scheme

The cell-vertex finite volume scheme is adopted to construct control volumes around the structure mesh vertices as shown in Figure 4.5. The control volume around vertex $\mathbf{O}$ is constructed by connecting the medians of all edges and the centroids of all triangular shell elements associated with vertex $\mathbf{O}$. Structural dynamics for the valve leaflets is considered based on the control volumes. Taking one control volume as example, it is under the influence of external fluid forces and internal tension forces as shown Figure 4.6. T_1-T_5 are tension forces acting on vertex $\mathbf{O}$ along mesh edges connected to it. p_L and p_R are fluid pressures on both sides of the valve leaflet. σ_t is fluid shear stress and σ_n the corresponding normal stress on the surfaces of the valve leaflets. And ΔS is the area of the control volume.

The volume-averaged fluid forces are stored at the vertices, thus fluid forces are computed within one element and distributed to the three vertices of the triangular element. Considering element one of the control volume O as shown in Figure 4.6, the fluid forces acting on it are calculated as follows:

Figure 4.5 Cell-vertex control volume around vertex O

Figure 4.6 Forces acting on one control volume

Figure 4.7 Computation of tension forces

$$\mathbf{F}_1 = (p_{L1} \Delta S_1 - p_{R1} \Delta S_1) \mathbf{n}_1 + (\sigma_{tL1} \Delta S_1 - \sigma_{tR1} \Delta S_1) \mathbf{t}_1 + (\sigma_{nL1} \Delta S_1 - \sigma_{nR1} \Delta S_1) \mathbf{n}_1 \quad (4.31)$$

where $\mathbf{n}_1$ is the unit normal vector of element one and $\mathbf{t}_1$ is the unit tangential vector of element one. L represents the forces acting on the left side and R the right side. The shaded region belongs to the control volume $\mathbf{O}$, which occupies 1/3 of the total area of the control volume, so the contribution of fluid forces to the control volume from element one is 1/3 of the fluid forces $\mathbf{F}_1$, i.e.:

$$\mathbf{F}_{o1} = \frac{\mathbf{F}_1}{3} \quad (4.32)$$

where $\mathbf{F}_{o1}$ represents the fluid forces acting on the control volume $\mathbf{O}$ contributing from structural element one. And the contribution of mass from element one to the control volume is calculated in the same manner:

$$m_{o1} = \frac{m_1}{3} = \frac{\rho h \Delta S_1}{3} \quad (4.33)$$

where ρ is the density of valve leaflet, and h is the thickness. Summing fluid forces from all the elements around node O , the total fluid forces acting on the control volume can be obtained:

$$\mathbf{F}_O = \sum \mathbf{F}_{oi} = \frac{\sum \mathbf{F}_i}{3} \quad i = 1, n \quad (4.34)$$

where n is the number of elements associated with vertex O . And the mass of the control volume is:

$$m_O = \sum m_{oi} = \frac{\sum m_i}{3} \quad i = 1, n \quad (4.35)$$

The tension forces acting on the control volume O are calculated based on mesh edges around node O . Taking edge OA as example, it is assumed that the tension force is uniform within the quadrilateral region around edge OA . So the tension T_I can be calculated by measuring the change of length of edge OA .

$$\mathbf{T}_I = EA_I \varepsilon_I \mathbf{t}_I = EA_I \left(\frac{dl_I}{l_I} \right) \mathbf{t}_I = EA_I \left(\frac{l_I - l_I^o}{l_I} \right) \mathbf{t}_I \quad (4.36)$$

Where E is the Young's modulus of the valve leaflet. l_I is the length of edge OA . $\mathbf{t}_I$ is the unit direction vector of edge OA pointing from O to A . A_I is the average transversal area of the quadrilateral region around edge OA . In this work, this area is approximated as:

$$A_I = \frac{l_I h}{3} \quad (4.37)$$

Where h is the thickness the valve leaflet. The total tension force can be obtained by summing the tension forces along all the edges associated with vertex O as:

$$\mathbf{T}_o = \sum_i \mathbf{T}_i = \sum_i EA_i \left(\frac{l_i - l_i^o}{l_i} \right) \mathbf{t}_i \quad i = 1, n \quad (4.38)$$

All the above variables are non-dimensionalized as the followings:

$$\begin{aligned} T'_i &= \frac{T_i}{\rho U_\infty^2 L} & \sigma'_n &= \frac{\sigma_n}{\rho U_\infty^2} & \sigma'_t &= \frac{\sigma_t}{\rho U_\infty^2 L} \\ E' &= \frac{E}{\rho U_\infty^2} & p'_i &= \frac{p_i}{\rho U_\infty^2} & \Delta l' &= \frac{\Delta l}{L} \\ \mathbf{r}' &= \frac{\mathbf{r}}{L} & \Delta m' &= \frac{\Delta m}{\rho L L L} = \frac{\rho_m h \Delta s}{\rho L^3} \equiv m' \end{aligned}$$

Dropping primes of the non-dimensional variables and applying the Newton's Second Law of motion to vertex $\mathbf{O}$, the governing equation for this control volume can be given as:

$$m_o \frac{d^2 \mathbf{r}}{dt^2} = \sum \mathbf{F} = \mathbf{F}_o + \mathbf{T}_o = \sum_i \frac{\mathbf{F}_i}{3} + \sum_i EA_i \left(\frac{l_i - l_i^o}{l_i} \right) \mathbf{t}_i \quad i = 1, n \quad (4.39)$$

4.3.4 Discretization of Structural Equations

The governing equation for the valve leaflet is spatially discretized over the triangular shell elements representing the mid-plane of the valve leaflet. Equation (4.39) can be transformed into the following equation:

$$\frac{d^2 \mathbf{r}}{dt^2} = \frac{d\mathbf{u}}{dt} = \frac{\sum \mathbf{F}}{m} \quad (4.40)$$

where $\mathbf{u} = d\mathbf{r}/dt$. Then introducing the pseudo time τ , the equation (4.40) can be given as:

$$\frac{d\mathbf{u}}{d\tau} = \frac{\sum \mathbf{F}}{m} - \frac{d\mathbf{u}}{dt} = \mathbf{RHS} \quad (4.41)$$

The second order backward difference scheme is used for the time differences in the above equation as:

$$\frac{d\mathbf{u}}{dt} = \frac{1.5\mathbf{u}^{n+1} - 2.0\mathbf{u}^n + 0.5\mathbf{u}^{n-1}}{\Delta t} \quad (4.42)$$

Finally, equations (4.41) can be written as:

$$\frac{\mathbf{u}^{n+1,m+1} - \mathbf{u}^{n+1,m}}{\Delta \tau} = \mathbf{RHS}^{n+1,m} \quad (4.43)$$

where $n+1$ is assigned to the physical time level t^{n+1} , $m+1$ to the pseudo time level τ^{m+1} and m to the pseudo time level τ^m . The above equations are integrated in pseudo time by a five stage Runge-Kutta method in sub-iterations in order to approach $\mathbf{RHS}=0$, which will recover the original structure equation (4.39).

4.3.5 Boundary Conditions

The aortic heart valve leaflet is mounted at the beginning of the aorta. It has one free edge, and the other boundaries are connected to the aorta root along the commissure line. Therefore the fully clamped boundary conditions are applied for these edges. i.e.,

$$w = 0 \quad \text{and} \quad \nabla w = 0$$

where w is the displacement of the boundary nodes. ∇w is the gradient of the displacement. For the free edge, it moves with the ambient fluids.

CHAPTER FIVE

IMMERSED MEMBRANE METHOD AND FLUID-STRUCTURE INTERACTION

This chapter presents the immersed membrane method (IMM) and the fluid-structure solution strategy. The core elements of this method are multiple ghost nodes and one-sided extrapolations for fluid variables to these ghost fluid nodes on unstructured grids. With the ghost fluid nodes, the mesh update for fluid domain is avoided and the fluid-structure interfacial boundary conditions are imposed. Hence, the numerical cost for mesh treatment is greatly reduced. By using the one-sided extrapolated ghost fluid variables, sharp fluid-structure interfaces can be maintained and “jump” fluid conditions across the immersed structure can be captured without smoothing, thus more accurate fluid solution can be achieved. Section one presents the concept of ghost fluid nodes and one-sided extrapolations, while section two describes the 2D and 3D immersed membrane methods. Finally, section three details the fluid-structure interaction solution procedure based on the IMM.

5.1 MULTIPLE GHOST FLUID NODES AND ONE-SIDED EXTRAPOLATIONS

When a thin structure is immersed in a flow field, it causes discontinuous flow conditions across the structure. Although velocity is continuous, the gradient of velocity, pressure, gradient of pressure and fluid stresses are quite different, even having sharp changes over both sides of the thin structure. Therefore, one-sided extrapolations of velocities and pressures are necessary for the investigation of the movement of the structure as well as the investigation of stresses and strains inside the structure. For example, the aortic heart valve is a thin and flexible structure subject to pulsatile blood flow. During the diastolic phase the valve has to sustain a large transvalvular pressure gradient. It is obvious that the one-sided extrapolation of pressure is important in such a case, while a pure Eulerian method is not able to handle this situation. These phenomena are especially important to high Reynolds-number flows when accurate descriptions of the thin boundary layers on immersed interfaces are needed.

In order to account for the discontinuous fluid conditions and to adopt the one-sided extrapolations on unstructured grids, a set of multiple ghost fluid nodes are introduced to replace every original fluid node when computations involve fluid nodes on both sides of the immersed structure. Fluid variables on these different ghost fluid nodes at the same location will be extrapolated from their related fluid nodes on the other side of the structure along different extrapolation paths, i.e., the edges of the cells in this case. And these extrapolated fluid variables will be used for the fluid computations across the structure.

5.2 IMMERSED MEMBRANE METHOD (IMM)

In the IMM, a stationary Eulerian mesh of triangular elements (2D) or tetrahedron elements (3D) is used for the fluid domain. The immersed thin structures are treated as membranes moving on top of the background Eulerian mesh, without changing the topology and geometry of the mesh. (The structure is defined as “thin” when the thickness of the structure is less than $1/20$ of the characteristic length of the fluid domain.) The structure is discretized into a mesh of line elements (2D) or a mesh of triangular shell elements (3D). Once the fluid mesh and the structure mesh are determined, their inter-relationship can be established. The inter-relationship is defined by which structural mesh node falls in which fluid mesh cell. Based on this inter-relationship, intersection points between the two meshes are found and the ghost fluid nodes are introduced. At last, one-sided extrapolations are carried out to obtain ghost fluid variables on the ghost fluid nodes.

5.2.1 Inter-relationship between Fluid Mesh and Structural Mesh

The inter-relationship between the fluid mesh and structure mesh is established through a searching process, which is illustrated in this section by an example of a 2D fluid mesh and a 2D structure mesh. The fluid mesh consists of unstructured triangular elements and the structural mesh line elements. Geometries of the fluid domain and the immersed structure are shown in Figure 5.1. The extension of the 2D immersed membrane method to 3D is also outlined in this chapter.

Figure 5.1 Geometry of a 2D fluid domain and an immersed structure with its bounding box for search process.

At the pre-processing stage, it is beneficial to narrow down the searched region of the fluid mesh that might contain a structure mesh node. A bounding box is defined around the immersed structure by the extrema of the structural coordinates to limit the searching region as depicted in Figure 5.1. If any of the three nodes of a fluid element falls into this region, this element is considered as a suspected element and stored into an array called “suspected array”. In the following process, searching will be only carried out between the structure nodes and this array of suspected fluid elements. This pre-processing reduces the search work greatly. For example, only 500 suspected fluid elements of a mesh of 30,000 elements fall into the bounding region in this work.

The concept of bounding box is also applied in the following search process to improve the efficiency of the algorithm. The structure node is first tested against this box before the full search algorithm is employed. The inter-relationship between the fluid mesh and structure mesh is determined by the value of dot products between the normal vectors of fluid edges and the vectors pointing from the median points of the

fluid edges to the structure nodes. As illustrated in Figure 5.2(a), $\vec{n}_1$, $\vec{n}_2$ and $\vec{n}_3$ are the unit normal vectors oriented inward from the fluid edges. While $\vec{p}_1$, $\vec{p}_2$ and $\vec{p}_3$ are vectors pointing from median points of the fluid edges to the structural node p . The dot products between these two sets of vectors are,

$$(p_n)^{nedge} = \{\vec{p}\} \cdot \{\vec{n}\} = \begin{Bmatrix} p_x & p_y \end{Bmatrix} \cdot \begin{Bmatrix} n_x \\ n_y \end{Bmatrix} = (p_x n_x + p_y n_y)^{nedge}, nedge=1 \text{ to } 3 \quad (5.1)$$

where the superscript *nedge* is the edge number of the triangular element. The value of the dot products is larger than or equal to zero if the angle, θ , subtended between them is less than 90° and less than zero if the angle is greater than 90° . If the structural node p is to fall within the triangular element, the three angles θ_1 , θ_2 and θ_3 must be less or equal to 90° , i.e., the dot products between these two sets of vectors must be positive for all the three edges. If one of the dot products is less than zero, the node being tested is not within the cell, as depicted in Figure 5.2(b). The criterion for the structural node to be within the triangular element is,

$$\{(p_n)^1 \text{ .and. } (p_n)^2 \text{ .and. } (p_n)^3\} \geq 0 \quad (5.2)$$

Figure 5.2(a) Structure node within a triangular fluid element.

Figure 5.2(b) Structure node not within a triangular fluid element

For 3D cases, the fluid mesh is of tetrahedron elements and the structure is of triangular elements, however, the basic idea for identifying inter-relationship between the two meshes is the same as for 2D cases. The same identifying criterion is used as,

$$(p_n)^{nface} = \{\vec{p}\} \cdot \{\vec{n}\} = \{p_x \ p_y \ p_z\} \cdot \begin{Bmatrix} n_x \\ n_y \\ n_z \end{Bmatrix} = (p_x n_x + p_y n_y + p_z n_z)^{nface}, nface=1 \text{ to } 4 \quad (5.3)$$

where $nface$ is the number of triangular faces of a tetrahedron element. $\vec{n}_1, \vec{n}_2, \vec{n}_3$ and $\vec{n}_4$ are the inward unit normal vectors of the four faces. $\vec{p}_1, \vec{p}_2, \vec{p}_3$ and $\vec{p}_4$ are vectors pointing from centroids of the four triangular faces to the structural node $\mathbf{p}$. For the node $\mathbf{P}$ to be within a tetrahedron fluid element, $(p_n)^{nface}$ must be no less than 0. Otherwise, the node falls out of the tested fluid element. The above searching process is done for all the structural nodes against the suspect fluid element array. After this procedure, the inter-relationship between the fluid mesh and structural mesh is established as a preparation for the next step of finding intersection points between the two meshes.

5.2.2 Intersection Points between Fluid Mesh and Structural Mesh

In this section, the intersection points between the fluid mesh and structural mesh are searched through as a preparation for defining ghost fluid nodes and extrapolations of ghost fluid variables.

Based on the established inter-relationship between the fluid mesh and structural mesh, all the structural nodes are now associated with their surrounding fluid elements. If two neighboring structural nodes are associated with two different fluid elements, i.e. the two nodes are in different fluid elements, the line elements between these two structural nodes will intersect the fluid element. As shown in Figure 5.3, structural nodes **M-2**, **M-1** and **M** are associated with fluid element **N**, while nodes **M+1** and **M+2** are associated with fluid element **N+1**. Checking through the neighboring structural nodes from node **M-2** to node **M+2**, it is found out that the neighboring pair of nodes **M** and **M+1** is in different fluid elements, thus the line element between nodes **M** and **M+1** will intersect with fluid elements **N** and **N+1**. The coordinates of the intersection point **I** are calculated out, and the distances between **I** and fluid nodes **2**, **3** are also calculated out for the extrapolation of ghost fluid variables.

In 3D cases, the searching for intersection points is a little more complicated. But the underlying algorithm is similar to the 2D cases. If the three nodes of a structural element are associated with different fluid elements, then this structural element is intersected by the fluid edges of the tetrahedron fluid elements.

Figure 5.3 Intersection point between the immersed membrane and fluid elements

5.2.3 Multiple Ghost Fluid Nodes and One-Sided Extrapolations

In this work, the fluid domain is represented by a stationary Eulerian background mesh. When a thin membrane is immersed in the fluid flow, it will intersect with the Eulerian mesh and causes discontinuous flow conditions. Taking a 2D fluid element N for example, it is cut by the membrane as shown in Figure 5.4. Fluid conditions on the nodes 1, 2 and 3 are discontinuous across the membrane. To consider this discontinuity, a set of multiple ghost fluid nodes is introduced here to replace the original fluid nodes when they are on different sides of the immersed membrane. Considering node 1 on the left side of the membrane, node 3 is a related node on the other side of the membrane. And a ghost fluid node g_{13} is introduced to replace node 3 when the extrapolation from node 1 to node 3 along edge 13 is performed. In such a case, node 1 is called the real fluid node and g_{13} is one of the multiple ghost nodes at 3, which is linked to node 1 through edge 13. As shown in Figure 5.4, node 3 has two ghost nodes g_{23} and g_{13}

because it is linked to 1 and 3 by edges 13 and 23, which are cut by the membrane. The naming convention for the ghost node is that it is prefixed with a *g* for ghost and the first number denotes the linked real fluid node and the second number the corresponding ghost fluid node. Under this convention, all ghost nodes for fluid element *N* are listed in Figure 5.4. For 3D cases, the real fluid nodes and multiple ghost fluid nodes of a tetrahedron fluid element intersected by the immersed membrane are listed in Figure 5.5.

Figure 5.4 Real fluid nodes and ghost fluid nodes of a 2D fluid element intersected by the immersed membrane

As described in Chapter 3, convective fluxes are computed based on mesh edges in the finite volume scheme. In the computation of the convective flux along edge 23, it involves the flow conditions of node 2 and node 3, (see Figure 5.4). When the convective flux is computed for node 2, ghost node g23 and its extrapolated flow variables are introduced to replace node 3 and its values in the computation. Likewise, when the convective flux is calculated for node 3, ghost node g32 and the extrapolated ghost variables are introduced to replace node 2 and its values in the computation. The

viscous fluxes and gradients are computed based on mesh elements. In element N , when the viscous flux is computed for node 1, it involves flow conditions at node 1, node 2 and node 3. The ghost node $g13$ and ghost variables are introduced to replace node 3 in the computation. The computations of viscous fluxes and gradients for node 1, 2 and 3 are listed in the Table 5.1.

Figure 5.5 Real fluid nodes and ghost fluid nodes of a 3D tetrahedron element intersected by the immersed membrane

Computation for	Involving flow conditions of
Node 1	Node 1, Node 2, Ghost fluid nodes $g13$ or $g23$
Node 2	Node 1, Node 2, Ghost fluid node $g13$ or $g23$
Node 3	Node 3, Ghost fluid node $g32$ and ghost fluid node $g31$

Table 5.1 Real fluid nodes and ghost fluid nodes for the computations of viscous fluxes and gradients of a 2D fluid element intersected by the immersed membrane

Fluid variables on the ghost fluid nodes are extrapolated linearly from their corresponding real fluid nodes along the mesh edges which are intersected by the immersed membrane. The extrapolated flow variables are called ghost node variables. Extrapolations of ghost node velocities are illustrated in Figure 5.6 (a). Taking node 1 as a real node, the velocity at node g12 is calculated as follows,

$$\frac{V_{g12} - V_I}{V_1 - V_I} = -\frac{|r_{2I}|}{|r_{1I}|} \quad (5.4)$$

therefore,

$$V_{g12} = V_I + \frac{|r_{2I}|}{|r_{1I}|} \cdot (V_1 - V_I) \quad (5.5)$$

where V_I is the fluid-structure interfacial velocity at intersection point I , V_1 is the velocity at node 1 and V_{g12} the velocity at node g12. Along the fluid-structure interface, the fluid velocity is equal to the structure velocity according to the no-slip boundary condition at the structure surfaces, i.e.,

$$V_I = V_s \quad (5.6)$$

where V_s is the velocity of the immersed structure. In fluid-structure interaction phenomena, the two domains are coupled by the velocity continuity along their interface as stated in equation (5.6). In the IMM, this velocity continuity is enforced by the extrapolations for ghost node velocities as shown in equation (5.5). The extrapolation of ghost node pressure is illustrated in Figure 5.6(b). Taking node 1 as real node, the ghost node pressure of node g12 is calculated as,

$$p_{g12} = p_1 + r_{12} \cdot \nabla p_1 \quad (5.7)$$

where p_{g12} is the pressure at node g12, ∇p_I is the pressure gradient at node 1, r_{12} is the distance vector pointing from node 1 to node 2.

Figure 5.6(a) Velocity extrapolation for ghost node g23 (b) Pressure extrapolation for ghost node g23

After all the extrapolations for ghost nodes have been performed, the fluid elements intersected by the immersed membrane will have two sets of fluid variables: the real values and the ghost values. Which one should be used depends on whether the fluid node in the calculation is a real node or ghost node according to Table 5.1. Equation (5.5) and equation (5.7) can be transformed into Taylor series expansions as follows,

$$V_{g12} = V_I + \dot{V}_I \cdot |r_{2I}| + O(|r_{2I}|)^2 \quad (5.8)$$

$$p_{g12} = p_1 + r_{12} \cdot \nabla p_I + O(r_{12})^2 \quad (5.9)$$

where

$$\dot{V}_I \approx \frac{V_I - V_1}{|r_{1I}|} \quad (5.10)$$

From the above analysis, the linear extrapolations for ghost flow variables in this work are second order accurate. It would, of course, be possible to implement high-order extrapolation schemes like MUSCL etc. [17] to achieve higher accuracy in future works.

5.3 SOLUTION PROCEDURE FOR FLUID-STRUCTURE INTERACTION

The fluid-structure interaction phenomenon involves two domains: the fluid domain and the structure domain. Generally, the two domains are coupled by two types of boundary conditions. One type of boundary condition is the velocity continuity at the fluid-structure interface. This boundary condition implies that the structure movement follows the movement of ambient fluids. It is widely implemented in various techniques to tackle the FSI problems. In the IMM, the velocity continuity is enforced through the velocity extrapolations for ghost nodes, thus representing the influence of the immersed structure on ambient fluids and providing a boundary condition for the computation of the fluid domain. Another type of boundary condition is the fluid force exerted on the immersed structure by fluids. After the computation of fluid domain, fluid forces acting on the structure is calculated along the FSI interface and they are applied to advance the movement of the structure. In the IMM, the fluid domain and the structure domain are loosely coupled by these two boundary conditions. Their relation is

depicted in Figure 5.7. In the fluid domain, the governing equations are the Navier-Stokes equations, which are solved by various computational fluid dynamics techniques using the velocity continuity at the FSI interface as a boundary condition. In the structure domain, the governing equation is the Newton's second law of motion. It can be solved by computational structure dynamics with the fluid forces exerted by ambient fluids as boundary conditions.

Figure 5.7 Relation between the structure domain and the fluid domain in fluid-structure interaction.

5.3.1 Fluid Forces on The Fluid-Structure Interface

With the implementation of the IMM, the computation for fluid domain can be carried out on a Eulerian mesh with appropriate boundary conditions. After the computation of the fluid domain, fluid forces are needed for the update of the structure domain. The fluid forces include pressure p , shear stress $\boldsymbol{\sigma}_t$ and normal stress $\boldsymbol{\sigma}_n$. Since the fluid mesh and structural mesh are non-confirmatory, the fluid pressure needs to be extrapolated to the fluid-structure interface for the computation of fluid forces. The pressure is linearly extrapolated to the interface from fluid mesh nodes nearest to the immersed structure as shown in Figure 5.8. The fluid mesh edge **12** intersects with a structural mesh element

at point I . The fluid pressures at both sides of the structure element are extrapolated respectively from vertex 1 and vertex 2 as the followings,

$$p_{1I} = p_1 + \mathbf{r}_{1I} \cdot \nabla p_1 \quad (5.11)$$

$$p_{2I} = p_2 + \mathbf{r}_{2I} \cdot \nabla p_2 \quad (5.12)$$

where p_{1I} is the fluid pressure on the fluid-structure interface on the side of vertex 1 , ∇p_1 is the pressure gradient at vertex 1 . $\mathbf{r}_{1I}$ is the distance vector pointing from vertex 1 to the intersection point I . p_{2I} is the fluid pressure on the side of vertex 2 , ∇p_2 is the pressure gradient at vertex 2 . $\mathbf{r}_{2I}$ is the distance vector pointing from vertex 2 to the intersection point I . For Newtonian fluids, the fluid stress tensor is given as follows,

$$\mathbf{t} = \begin{bmatrix} \sigma_{xx} & \sigma_{xy} & \sigma_{xz} \\ \sigma_{yx} & \sigma_{yy} & \sigma_{yz} \\ \sigma_{zx} & \sigma_{zy} & \sigma_{zz} \end{bmatrix} \quad (5.13)$$

Figure 5.8 Extrapolation of fluid pressure to FSI interface.

$$\text{and} \quad \begin{aligned} \sigma_{xx} &= \lambda(\nabla \cdot \mathbf{V}) + 2\mu \frac{\partial v_x}{\partial x}, & \sigma_{xy} = \sigma_{yx} &= \mu \left(\frac{\partial v_y}{\partial x} + \frac{\partial v_x}{\partial y} \right), \\ \sigma_{yy} &= \lambda(\nabla \cdot \mathbf{V}) + 2\mu \frac{\partial v_y}{\partial y}, & \sigma_{yz} = \sigma_{zy} &= \mu \left(\frac{\partial v_z}{\partial y} + \frac{\partial v_y}{\partial z} \right), \\ \sigma_{zz} &= \lambda(\nabla \cdot \mathbf{V}) + 2\mu \frac{\partial v_z}{\partial z}, & \sigma_{xz} = \sigma_{zx} &= \mu \left(\frac{\partial v_x}{\partial z} + \frac{\partial v_z}{\partial x} \right) \end{aligned}$$

where μ is the molecular viscosity and λ is the second viscosity coefficient. Stokes

[133] hypothesized that $\lambda = -\frac{2}{3}\mu$, and for incompressible flows $\nabla \cdot \mathbf{V} = 0$. With the

above equations, the stress tensor $\mathbf{t}$ for incompressible flows is given as,

$$\mathbf{t} = \begin{bmatrix} 2\mu \frac{\partial v_x}{\partial x} & \mu \left(\frac{\partial v_y}{\partial x} + \frac{\partial v_x}{\partial y} \right) & \mu \left(\frac{\partial v_z}{\partial x} + \frac{\partial v_x}{\partial z} \right) \\ \mu \left(\frac{\partial v_y}{\partial x} + \frac{\partial v_x}{\partial y} \right) & 2\mu \frac{\partial v_y}{\partial y} & \mu \left(\frac{\partial v_z}{\partial y} + \frac{\partial v_y}{\partial z} \right) \\ \mu \left(\frac{\partial v_z}{\partial x} + \frac{\partial v_x}{\partial z} \right) & \mu \left(\frac{\partial v_z}{\partial y} + \frac{\partial v_y}{\partial z} \right) & 2\mu \frac{\partial v_z}{\partial z} \end{bmatrix} \quad (5.13)$$

so the fluid stress $\boldsymbol{\sigma}$ can be calculated as the following,

$$\boldsymbol{\sigma} = \mathbf{t} \cdot \mathbf{n} = \begin{bmatrix} 2\mu \frac{\partial v_x}{\partial x} & \mu \left(\frac{\partial v_y}{\partial x} + \frac{\partial v_x}{\partial y} \right) & \mu \left(\frac{\partial v_z}{\partial x} + \frac{\partial v_x}{\partial z} \right) \\ \mu \left(\frac{\partial v_y}{\partial x} + \frac{\partial v_x}{\partial y} \right) & 2\mu \frac{\partial v_y}{\partial y} & \mu \left(\frac{\partial v_z}{\partial y} + \frac{\partial v_y}{\partial z} \right) \\ \mu \left(\frac{\partial v_z}{\partial x} + \frac{\partial v_x}{\partial z} \right) & \mu \left(\frac{\partial v_z}{\partial y} + \frac{\partial v_y}{\partial z} \right) & 2\mu \frac{\partial v_z}{\partial z} \end{bmatrix} \cdot \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \quad (5.14)$$

where $\mathbf{n}$ is the unit normal vector of the structure element. The fluid normal stress σ_n

and shear stress σ_t are computed as,

$$\sigma_n = \boldsymbol{\sigma} \cdot \mathbf{n} \quad (5.15)$$

$$\sigma_t = \boldsymbol{\sigma} - \sigma_n \quad (5.16)$$

In this work, the fluid flow is computed using the finite volume scheme. In finite volume scheme, the gradients of velocity are computed based on mesh elements as described in chapter 3, so all the gradients of velocity are uniform within one fluid mesh

element. It means that on the fluid-structure interface the gradients of fluid velocities are equal to those at the element centroids. But for an element intersected by the immersed structure, the gradients are only uniform for centroids on the same side of the structure. So the computations of fluid stresses on both sides of the structure should use their respective gradients of velocities. For example, at the interface on the side of vertex **1**, fluid stresses are computed using the gradients of vertex **1**, i.e.,

$$\sigma_1 = \bar{t}_1 \bullet n = \begin{bmatrix} 2\mu \frac{\partial v_{1x}}{\partial x} & \mu(\frac{\partial v_{1y}}{\partial x} + \frac{\partial v_{1x}}{\partial y}) & \mu(\frac{\partial v_{1z}}{\partial x} + \frac{\partial v_{1x}}{\partial z}) \\ \mu(\frac{\partial v_{1y}}{\partial x} + \frac{\partial v_{1x}}{\partial y}) & 2\mu \frac{\partial v_{1y}}{\partial y} & \mu(\frac{\partial v_{1z}}{\partial y} + \frac{\partial v_{1y}}{\partial z}) \\ \mu(\frac{\partial v_{1z}}{\partial x} + \frac{\partial v_{1x}}{\partial z}) & \mu(\frac{\partial v_{1z}}{\partial y} + \frac{\partial v_{1y}}{\partial z}) & 2\mu \frac{\partial v_{1z}}{\partial z} \end{bmatrix} \bullet \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix} \quad (5.17)$$

While for the interface on the side of vertex **2**, the stress is computed using gradients of vertex **2** or vertex **3** or **4**.

5.3.2 Solution Procedure

The FSI is solved in an iterative manner as follows:

- (1) Use IMM to extrapolate velocity and pressure for ghost nodes enforcing velocity continuity at the FSI interface. The extrapolated ghost node variables are stored at ghost nodes for the computation of fluid domain when a ghost node is involved in the computation. This process imposes a boundary condition to the fluid domain due to the presence of the immersed structure.
- (2) With appropriate boundary conditions and the extrapolated ghost node variables, the Navier-Stokes equations are solved by the finite volume scheme as detailed

in Chapter 3. If the solution is converged, fluid forces are calculated on the FSI interface.

- (3) Under the influence of the fluid forces, velocity and displacement of the structure are updated by solving the Newton's Second Law of Motion.
- (4) At the new spatial position, the immersed structure intersects with the background Eulerian mesh. Then inter-relationship between the structure and the background mesh is re-established. Intersection points are searched for the newly updated structure with background mesh and distance vectors being calculated for the velocity and pressure extrapolations in IMM.

Repeating the above steps for other time steps until the desired time is reached. The flow chart of the solver is shown in the following diagram.

Figure 5.9 Flowchart for solution of fluid-structure interaction

5.4 EXTENSION OF IMM TO ARBITRARY OBJECT SHAPES

The IMM is not only limited to the studies of immersed membrane structures. It can be extended to study the fluid-structure interaction for arbitrary object shapes. For an arbitrary object immersed in a fluid as shown in Figure 5.10, it occupies part of the fluid domain. And some fluid nodes will fall within the immersed objects, which are represented by hole points in Figure 5.10. For these fluid nodes, they will not take part in the fluid calculation since there are no fluids inside the immersed object. So before the application of IMM, a pre-process is carried out first to identify those fluid nodes which fall in the domain of the object. The IMM is then carried out as described in previous sections, treating the surface of the immersed object as the immersed membrane. All extrapolations are done with respect to the object surface. Since the fluid nodes inside the object will not be involved in the fluid computation, only ghost variables for those ghost nodes outside the immersed objects are needed in the implementation of IMM.

Figure 5.10 IMM for arbitrary object shapes

CHAPTER SIX

PARALLEL COMPUTING

This chapter covers the implementations of parallel computing in this work. Numerical simulations of fluid-structure interaction with complex geometries are highly computational resource-consuming research tasks. The motivation to implement parallel computing is to develop algorithms and strategies that would enable the 3D Navier-Stokes solver based on unstructured grids to exploit the computational advantages offered by Symmetric Multiprocessing (SMP) and Distributed Memory (DM) machines. The serial program is parallelized by the Single Program Multiple Data (SPMD) programming paradigm combining Domain Decomposition (DD) and Message Passing Interface (MPI). The SPMD parallel strategy is detailed in section one, describing the MPI implementations and the geometric domain decomposition technique for data parallelism, which includes the algorithm used to identify the ghost nodes and overlapping elements, and the data structure that defines which node variables will be sent to and received from, as well as the associated processors. In section two, data communication for FSI is described. Finally, section three defines the speedup and efficiency used to measure the performance of the 3D parallel program.

6.1 SIMPLE PROGRAM MULTIPLE DATA PARALLEL PROGRAMMING

Researchers have developed several methods for parallel programming. They are shared-memory parallel, distributed-memory parallel and hybrid methods. The SPMD paradigm is chosen in this work in order to maintain consistency with the original serial solver, minimizing modifications to the original serial solvers when parallelizing them. It employs the MPI parallel strategy using distributed memory. It decomposes the computational domain into a set of several partitions (sub-domains) that may be distributed over several processors on a parallel machine. Each processor runs the same program and operates only on its own part of the problem with the boundary node variables and gradients being communicated with other processors using the Message-Passing Interface (MPI) [98]. MPI is based on independent processes which do not share any memory. An overlapping mesh-partitioning technique [19] is employed in this work so as to build the control volumes for the finite-volume formulation. Typically, a fixed number of MPI processes are used, corresponding to the number of sub-domains. The flowchart presented in Figure 6.1 depicts the SPDM parallelization strategy used in this work.

Figure 6.1 Flowchart depicting geometric domain decomposition parallelization strategy using MPI

6.1.1 Message Passing Interface (MPI)

MPI [98] is a specification of message passing libraries designed to be a standard for message communication in parallel computing. The interface attempts to establish a practical, portable, efficient, and flexible standard for message passing. Message passing involves the transfer of data from one process (send) to another process (receive). It requires the cooperation of both the sending and receiving processes. Send operations usually require the sending process to specify the data's location, size, type and the destination. Receive operations should match a corresponding send operation. In MPI parallel programming, the programmer is responsible for identifying parallelism and implementing the resulting algorithm with MPI constructs. From a programming perspective, message passing implementations commonly comprise a library of routines that are imbedded in source code. The MPI library has more 100 routines, but a general MPI program structure can be implemented by several MPI routines as depicted in Figure 6.2.

'mpi.h' is a header file required for all programs which make MPI library calls. **MPI_INIT** initializes the MPI execution environment. It must be called in every MPI program before any other MPI routines and must be called only once in an MPI program. **MPI_COMM_SIZE** determines the number of processes associated with a communicator, which is **MPI_COMM_WORLD** by default. **MPI_COMM_RANK** determines the rank of the calling process within the communicator. Calling this function will assign each process a unique integer number between 0 and (number of processes-1) which can be treated as the process ID. The major role of MPI is performed by the communication routines: **MPI_BCAST**, **MPI_SEND** and

MPI_RECV etc. **MPI_BCAST** is a collective communication routine. It broadcasts (copies) messages from the root (rank 0) process to all the other processes. **MPI_SEND** and **MPI_RECV** are two typical point-to-point communication routines, they send and receive message between processes. Finally, the parallel program is terminated by calling the routine **MPI_FINALIZE** to exit the MPI execution environment.

Figure 6.2 General structure for an MPI parallel program

6.1.2 Domain Decomposition

The SPMD paradigm is based on the division of computational data into sub data sets and assigning the same or different tasks to operate on a partition of the computational data. The tasks work collectively on the same set of computational data. Each task

works on a different partition of the same data structure. Before a parallel program runs, the computational domain should be pre-processed to be partitioned into smaller computational sub-domains. Domain decomposition (DD) is such a process to partition a certain domain into several smaller sub-domains that do not overlap with each other in an optimal way. It is a generic name given to a variety of computational activities, which involve the division of a problem space into two or more parts that can be operated on separately to some advantage. Domain decomposition of a mesh into a set of S sub-domains that may be allocated to a set of P processors involves finding a partition of the mesh so that the amount of computational time on each processor is almost equal. This method has the advantage of flexibility to allow variations in the decomposition strategy to be used to minimize communication and maximize processor utilization. This must be done with care so that the amount of computation and/or message passing associated with sub-domain boundary interfaces does not grow rapidly with the number of processors. For structured meshes, the mesh partitioning is quite straightforward. However, for unstructured meshes, the mesh partitioning becomes a non-trivial problem. Following the 2D approach presented in [19], domain decomposition contributes the most important part of this parallel implementation and new algorithms are adopted for this 3D study [147], which are outlined in the following sub-sections.

6.1.2.1 METIS

METIS [148] is a software package developed by University of Minnesota to produce the partitions used in this work. It provides two programs *partnmesh* and *partdmesh* for partitioning meshes into k equal-size parts. These programs take the element node array of the mesh as input and compute a partitioning for both its element and its nodes. The

difference between these two programs is that *partnmesh* converts the mesh into a nodal graph (*i.e.*, each node of the mesh becomes a vertex of the graph), whereas *partdmesh* converts the mesh into a dual graph (*i.e.*, each element becomes a vertex of the graph). In the case of *partnmesh*, the partitioning of the nodal graph is used to derive partitions of the elements. In the case of *partdmesh*, the partitioning of the dual graph is used to derive a partitioning of the nodes. *Partnmesh* is chosen over *partdmesh* to be used in this work as numerical experiments show that the run-time for simulation is much faster for partitioning file obtained from *partnmesh*. This maybe due to different type of algorithms used to partition the mesh for both *partnmesh* and *partdmesh*. Since *partnmesh* takes the element node array of the mesh as the input, the unstructured tetrahedral grid file generated in this work is yet to be used as the input for *partnmesh*. In view of this, a program named “*prepmesh3D*” is developed to output a new graph file that is written according to the element node array of the mesh. A script file, “*prepgrid3D*”, is written to integrate both the programs *prepmesh3D* and *partnmesh*. Upon successful execution of *prepgrid3D*, the actual partitioning is stored in two files names as *MeshFile.npart.Nparts* which stores the partitioning of the *nodes*, and *MeshFile.epart.Nparts* which stores the partitioning of the *elements*, where *MeshFile* is the name of the file that stores the mesh and *Nparts* is the number of required partitions. *MeshFile.npart.Nparts* is the only file required for the identification of “*ghost nodes*” and overlapping elements in this work. In this context, a core node is defined as the real and physical node of the partitioned mesh, whereas a *ghost* node is defined as the non-physical node of the partitioned mesh that belongs to the neighboring sub-domains and no computation is performed on it. Overlapping elements in this context is defined as those elements that overlap each other along the inter-processor boundaries and the nodes that formed these elements are a mixture of core and *ghost* nodes.

The developed algorithm is written in a program named “*Partition*” and is fully integrated into the 3D Navier-Stokes solver so as to have little intervention from the user. Basically, it consists of four essential steps:

- (1) Identify *ghost* nodes and overlapping elements between the neighboring sub-domains.
- (2) Generate individual grid files with local numbering for every partition.
- (3) Establish data structure for communication between core and *ghost* nodes.
- (4) Generate a script file, “*combine*”, to integrate all the result files into one after the parallel simulation has been performed.

The flowchart shown in Figure 6.3 depicts the above-described algorithm.

Figure 6.3 Flowchart depicting algorithm developed in program “*Partition*”

6.1.2.2 Identification of Ghost Nodes and Overlapping Elements

After partitioning the mesh into P parts using METIS, new algorithms are developed to identify the “*ghost nodes*” and overlapping elements and to write the individual grid files with local numbering for each partition. The program also includes establishing the data structure for communication and generating a script file, “*combine*”, to integrate all the result files into one as each processor writes their own result based on local numbering. The nodes and elements that are allocated uniquely to a processor are referred to as core mesh components in this work and each processor calculates the flow field variables and nodal gradients for it. Each sub-domain is enclosed with a layer of *ghost* nodes and overlapping elements, which overlap the sub-domains along the inter-processor boundaries as depicted in Figure 6.4. For illustration purposes, Figure 6.4 illustrates a 2D mesh being partitioned. The developed algorithms for 3D mesh are much more complex than the illustrated 2D mesh. In Figure 6.4, it shows that the irregular *partition lines* bisect the 2D mesh whereas irregular *partition faces* bisect the 3D mesh and the mesh elements are tetrahedral instead of triangles. In general, the ideas behind the algorithms for both 2D and 3D model are the same. Figure 6.5 shows a 3D cavity mesh being partitioned into various sub-domains generated by this partitioning algorithm.

These *ghost* nodes contain flow field variables and nodal gradients from neighboring sub-domains that are required for the solution of variables within the sub-domain so as to maintain a solution consistent with the original serial code and no computation is being performed on it. Communication between these core and *ghost* nodes is based on MPI and the data flow direction is always from core nodes to *ghost* nodes. The main concept of this algorithm is that those elements along the inter-

processor boundaries and with nodes having different partition numbers are considered as overlapping elements which are cut through by partition lines. And those nodes that form these elements are a mixture of core and ghost nodes. Basically, the ghost node of this partition is the mirror image of the corresponding core node of the neighboring partition. For example, as shown in Figure 6.4, the *ghost* node number 11 of partition 3 is the mirror image of core node number 10 of partition 1. Therefore, making use of this correlation between the core and *ghost* nodes, the rest of the *ghost* nodes can be identified so as to build the control volume for the core nodes.

6.1.2.3 Grid Files with Local Numbering

Since the SPMD programming paradigm is employed in this work for geometric domain decomposition, which distributes the problem domain over P processors and each processor runs the same program individually. It is a wise strategy to re-write the global grid file into the respective local partitioned grid files with local numbering. In this way, each processor will only read and write their respective grid file and output files individually, without having any *master* processor to control the I/O operations. This avoids any synchronization in the event of reading and writing and it behaves like a serial program with each performing its own computation. The only communication is done during the passing of nodal gradients and flow variables. Thus, a substantial amount of computational time and memory allocation is saved. Another advantage is that the solver only reads the respective partitioned grid file for the next simulation run without both accessing the global grid file and generating the mesh decomposition algorithm again.

Figure 6.4 Mesh decomposed into 4 partitions showing each sub-domain extended with a layer of *ghost* nodes and overlapping elements.

Figure 6.5 Cavity mesh decomposed into various partitions (a) original mesh; (b) 2 partitions; (c) 4 partitions; (d) 8 partitions; (e) 16 partitions; (f) 32 partitions.

6.1.2.4 Data Structure for the Exchange of *Ghost* Nodes Variables and Gradients

The communication between the core and *ghost* nodes requires a data structure for each sub-domain, which holds the node and processor numbers to be sent and received. Each processor regards its re-numbered sub-domain as a complete mesh, consisting of n_e elements and n_p nodes where n_e and n_p are the local number of elements and nodes

respectively. In this work, the communication and writing of variables to files are based on local numbering rather than global numbering and therefore, no translation back from local to global numbering is necessary.

With reference to the grid decomposed in Figure 6.4, the data structure established for communication between the core and *ghost* nodes is shown in Table 6.1 and it only shows the data structure for partition 1 and the data structure for the rest of the partitions will follow exactly the same fashion. The data structure is appended to the end of each individual grid file and the respective processor will only read its grid file. Part of the code for reading the data structure is shown as follows for a particular processor, *myid*,

```

read (nunit,*) nсенrev

do id = 1,nsenrev
  read(nunit,*) igver,nprtn,icore,npart,ighst
  if (nprtn.eq.myid) then
    nsend(npart) = nsend(npart) + 1
    sendvert(npart,nsend(npart)) = icore
  else
    nrecv(nprtn) = nrecv(nprtn) + 1
    recevert(nprtn,nrecv(nprtn)) = ighst
  endif
enddo

```

where **nсенrev** is the total number of nodes to be sent and received, **sendvert** is a two-dimensional array that contains the locally numbered core nodes, **icore** is the core node to be sent to the partition number, **npart**, and **recevert** is similar to the array **sendvert** except that it contains the locally numbered ghost nodes, **ighst** is the ghost node that will be received from the partition number, **nprtn**.

On most parallel computers, moving data from one processor to another takes more time than moving or manipulating data within a single process. To keep a program

from being slowed down, many parallel computers allow users to start sending or receiving several messages and to proceed with other operations. Message passing is

Global node number	<i>Send</i>		<i>Receive</i>	
	Partition number	Local node number (core node)	Partition number	Local Node number (ghost node)
igver	nprtn	icore	npart	ighst
11	1	5	2	17
31		8		16
27		9		15
26		10		14
26		10	3	11
23		7		10
6		3		9
26		10	4	12
10	2	5	1	17
32		9		16
29		10		15
24	3	8		13
22		7		12
7		3		11
28	4	8		14

Table 6.1 Communication between core and *ghost* nodes for partition 1 as depicted in Figure 6.4

performed asynchronously using the buffered-send **mpi_bsend()** routine in this work. This allows the messages to be placed in the buffer until it is delivered. Once all the flow field variables or nodal gradients or forcing terms have been sent, the processes are ready to receive all these from the respective processes. Messages are received by the standard blocking-receive **mpi_recv()** routine. To reduce the communication time and the latency overheads by creating fewer larger messages, the flow field variables (ua()),

nodal gradients and forcing terms are packed into a single array (`sendsol( )` for flow field variables only) before sending them to the respective partitions, as depicted in the code shown below. The flow field variables, nodal gradients or forcing terms are unpacked from the single array (`recvsol( )` for flow field variables only) after receiving from the respective partitions.

```

do npart=0,numprocs-1
  .....
  iq = 0
  do iv=1,nsend(npart)
    is = sendvert(npart,iv)
    do ivar=1,nvar
      iq = iq + 1
      sendsol(iq) = ua(ivar,is)
    enddo
  enddo
  call mpi_bsend(sendsol,(nsend(npart)*nvar),
&  mpi_real,npart,myid,mpi_comm_world,ierr)

  call mpi_recv(recvsol,(nrecv(npart)*nvar),
&  mpi_real,npart,npart,mpi_comm_world,istatus,ierr)
  id = 0
  do iv=1,nrecv(npart)
    is = recevert(npart,iv)
    do ivar=1,nvar
      id = id + 1
      ua(ivar,is) = recvsol(id)
    enddo
  enddo
  .....
enddo

```

The first argument in **mpi_bsend()** is the address of the variables to be sent, the second argument is the number of variables to be sent, the third argument is the MPI derived data type, **mpi_real**, the fourth argument is the destination where the variables are to be sent, the fifth argument is called the *tag*, which is an arbitrary non-negative integer to restrict receipt of the message. Finally, a communicator, **mpi_comm_world**, is specified and *ierr*, in which to return an error code. The **mpi_recv()** routine is a

blocking receive, in which the control is not returned to the program until the message is received. The first argument is the address of the buffer into which the received message is to be put, the second argument is the number of variables to be received, the third argument is the same as **mpi_bsend**, the fourth argument is the source of the variables to be received from, the fifth argument is the *tag*.

6.2 DATA COMMUNICATION FOR FLUID-STRUCTURE INTERACTION AND IMMERSED MEMBRANE METHOD

In this work, there are two domains involving in the FSI: the structure domain and fluid domain. In the previous section, the fluid domain is divided into several sub-domains which have their own local numbering data set for parallel computing. While for structure domain, it will not be parallelized since the mesh size is far smaller compared with the fluid mesh. For example, a mesh size used for fluid domain in this work is of about 180,000 nodes and 900,000 elements. It is divided into 8 partitions, and each partition has about 22,000 nodes and 110,000 elements. The mesh for structure domain is of about 5,000 nodes and 15,000 elements. The structure mesh is even smaller than one partition of the fluid mesh. So the structure mesh is treated as a single mesh without partition, and computation of the structural domain is executed on the root process. Since computation for the fluid domain is parallelized and operated on several processes, data communications are needed to pass the messages between local partitioned fluid meshes and the root process with the structure mesh for computation of fluid-structure interaction. This section will describe the data communication at different stages of the computation for FSI based on the immersed membrane method.

6.2.1 Sending Information of Ghost Nodes and Variables

Since the configuration and position of the structure mesh is not known apriori and it may span through several partitions of the fluid mesh during its motion, the inter-relationship is established between the global fluid mesh and structure mesh for identifying ghost fluid nodes in the implementation of the immersed membrane method. A copy of global fluid mesh is stored at the root process together with the structure mesh. The identifying process for ghost nodes and the one-sided extrapolations for ghost variables as described in section 5.2 are performed at the root process. After this stage, the information of ghost nodes is sent to other processes from the root process as well as the information of ghost variables associated with these ghost nodes. By doing so, all processes obtain the information of ghost nodes and ghost variables. Thus no matter which partitioned meshes the structure may span through, the IMM can be carried out on the local data with the ghost nodes and ghost variables. The ghost variables are packed into a single array and sent out to all processes from the root process as depicted in the following code,

```

num_send = 0
if (myid.eq.0)then
  do is = 1, ns_glb
    if (mflag(is).ne.0) then
      mf = mf + 1
      mfc(mf) = is
      num_send=num_send+1
      send_g2l(num_send) = uag(1,is)
      ... ..

```

Ghost variables include velocity and pressure $uag(4, is)$ and gradients $dxg(4, is)$, $dyg(4, is)$, $dzg(4, is)$, totally 16 variables for each ghost node.

```

... ..
    send_g2l(num_send) = dzg(4,is)
endif
enddo

do npart=1, numprocs-1
    call mpi_bsend(mf,1,mpi_integer,npart,0,mpi_comm_world,ierr)
    call mpi_bsend(mfc,mf,mpi_integer,npart,0,mpi_comm_world,ierr)
    call mpi_bsend(send_g2l,16*mf,mpi_real,npart,0,mpi_comm_world,ierr)
enddo
endif

```

Where *ns_glb* is the number of nodes of the global fluid mesh. *mflag* is an integer array of flags assigned to all fluid nodes. All ghost nodes have non-zero flags, while all normal fluid nodes have flags of zero values. *uag* is the array of ghost variables, *dxg*, *dyg* and *dzg* are arrays of ghost gradients of fluid variables. During the receiving process, the received data are unpacked and stored on the copy of global mesh. The receiving process is depicted in following code,

```

if (myid.ne.0) then
    call mpi_recv(mf,1, mpi_integer,0,0,mpi_comm_world,istatus,ierr)
    call mpi_recv(mfc,mf, mpi_integer,0,0,mpi_comm_world,istatus,ierr)
    call mpi_recv(recv_g2l,16*mf,mpi_real,0,0,mpi_comm_world,istatus,ierr)
    num_recv = 0
    do is=1,mf
        id=mfc(is)
        num_recv=num_recv+1
        uag(1,id) = recv_g2l(num_recv)
        ... ..
        ... ..
        num_recv=num_recv+1
    enddo
endif

```

```

        dzg(4,id) = recv_g2l(num_recv)
    enddo
endif

```

6.2.2 Data Communication for Computation of Fluid Forces

The fluid-structure interaction is solved in an iterative manner as described in section 5.3. In serial computing, the iterations are straightforward. However in parallel computing, the computation for fluid domains is carried out at different processes with local partitioned data, but the computation for the structure domain is carried out only at the root process. So after the computation of the fluid domains, the information of fluid nodes nearest to the immersed structure should be sent back to the root process for computation of fluid forces to update the structure domain. This data communication is a one-way communication from other processes to the root process, which sends information of only those fluid nodes having ghost fluid nodes. First the number of such fluid nodes $nl2g$ is counted in each partition. If the number is non-zero, this partition has ghost fluid nodes nearest to the immersed structure whose information will be sent to the root process. The information includes the velocity and pressure, gradients of velocity and pressure. These data will be packed a single array $send_l2g()$. At the root process, the fluid information is received in a single array $recv_l2g()$. The data sending is carried out as the following code,

```

do is=1, ns
    id = locglb(is)
    if (mflag(id)  $\neq$  0 )then
        nl2g = nl2g + 1
        l2gvert(nl2g) = id
    endif
enddo

```

```

    num_send = num_send+1
    send_l2g(num_send) = ua(1,is)
    ... ..
    ... ..
    num_send = num_send+1
    send_l2g(num_send) = dz(4,is)
endif
enddo

call mpi_bsend(nl2g, 1, mpi_integer, 0, myid, mpi_comm_world, ierr)
if(nl2g > 0)then
    call mpi_bsend(l2gvert, nl2g, mpi_integer, 0, myid, mpi_comm_world, ierr)
    call mpi_bsend(send_l2g, 16*nl2g, mpi_real, 0, myid, mpi_comm_world, ierr)
endif

```

Where *locglb* is an array storing information of the local number of a fluid node and its corresponding global number in the global fluid mesh. At the root process, the receiving data is unpacked and assigned to their corresponding global fluid nodes. The receiving is depicted as follows,

```

do npart=0,numprocs-1
    call mpi_recv(nl2g,1, mpi_integer, npart, npart, mpi_comm_world, istatus, ierr)
    if(nl2g > 0)then
        call mpi_recv(nd_gbs,nl2g,mpi_integer,npart,npart,mpi_comm_world,istatus,ierr)
        call mpi_recv(recv_l2g,16*nl2g,mpi_real,npart,npart,
&                mpi_comm_world,istatus,ierr)
        num_recv = 0
        do is=1,nl2g
            id=nd_gbs(is)
            num_recv=num_recv+1
            ua_glb(1,id)=recv_l2g(num_recv)

```

```

    ... ..
    ... ..
    num_recv=num_recv+1
    dz_glb(4,id)=recv_l2g(num_recv)
enddo
endif
enddo

```

6.3 MEASURING PERFORMANCE

Both speed-up and efficiency are commonly used to measure the performance of a parallel code. The run time of the original serial code is used as a measure of the run time on one processor. In this study, the run time or total simulation time starts from the moment before mesh partitioning, identifying *ghost* nodes and writing local grid files for different partitions to the time after writing all the results to the respective files.

In this work, both CPU time and wall-clock time are used to record the total simulation time. The main difference is that CPU time is the recorded time when only a process performs a calculation whereas clock time is almost similar to CPU time but it includes idling time when the processor idles while waiting for other processors to communicate. The wall-clock time is used to represent the total simulation time in this work since it includes the idling time, computation time and communication time, which is the true representation of the total simulation time. Computation time is the time spent on computing and communication time is the total time spent on exchange of flow field variables, nodal gradients and forcing terms.

6.3.1 Speed-up

The parallel speed-up S_p is the ratio of the run-time on one processor t_1 to the run-time on P processors, t_p [149].

$$S_p = \frac{t_1}{t_p} \quad (5.1)$$

If the parallelization is 100% efficient, then $S_p = P$, but this is rarely the case as both communication and load balancing adversely affect the performance of parallel computation. In this work, a theoretical model has been formulated for the prediction of the parallel computational and communication times. The total simulation wall-clock time, t_p , is used as the run-time to compute the speedup and it is the maximum wall-clock time among all the processors defined as follows,

$$\text{Total simulation wall-clock time, } t_p = \max(t_{\text{cpu}}^j + t_{\text{idle}}^j); \quad \text{for all } j \quad (5.2)$$

where j is the processor number, t_{cpu}^j and t_{idle}^j is the CPU time and idling time for processor j , respectively. And the CPU time for processor j is defined as follows,

$$\text{CPU time, } t_{\text{CPU}}^j = t_{\text{comp}}^j + t_{\text{comm}}^j \quad (5.3)$$

where t_{comp}^j and t_{comm}^j is the computation and communication time for processor j , respectively. The overall communication time is defined as the maximum communication time among all the processors:

$$\text{Communication time, } t_{\text{comm}}^j = (t_{\text{comm}}^{\text{FFV}} + t_{\text{comm}}^{\text{NG}} + t_{\text{comm}}^{\text{FT}})_j \quad (5.4)$$

where $t_{\text{comm}}^{\text{FFV}}$, $t_{\text{comm}}^{\text{NG}}$ and $t_{\text{comm}}^{\text{FT}}$ are the communication time for exchanging of flow field variables, nodal gradients and forcing terms, respectively. And the representative computation time of a process j is defined as follows,

$$\text{Computation time, } t_{\text{comp}}^j = t_{\text{CPU}}^j - t_{\text{comm}}^j \quad (5.5)$$

6.3.2 Parallel Efficiency

The parallel efficiency is sometimes used as the performance parameter for a parallel code. Parallel efficiency E_p is simply the ratio of the parallel speed-up S_p to the number of processors P [149].

$$E_p = \frac{S_p}{P} \quad (5.6)$$

where $0 \leq E_p \leq 1$.

CHAPTER SEVEN

NUMERICAL RESULTS AND ANALYSIS

This chapter presents the numerical results of various fluid-structure interaction problems computed by the Immersed Membrane Method (IMM). First, the results of steady and unsteady flows in a 2D channel with immersed structures are presented to demonstrate the capacity of the proposed IMM method to tackle FSI problems. The motions of a rigid membrane and a flexible membrane in 2D viscous incompressible fluids are computed and analyzed by the new method. It is observed that the proposed method is able to simulate large displacement in fluid-structure interaction at relatively low costs. Second, a 29mm bi-leaflet mechanical heart valve is modeled by the IMM under physiological conditions. The results agree well with available numerical and experimental results. The IMM can be extended to model immersed structures with arbitrary shapes as demonstrated in the simulations of flows due to oscillating sphere and the flows due to rotating sphere. The results are in good agreements with analytical solutions and available numerical results. Analysis on the results of grid convergence study shows that the IMM is a second order accurate method. At last, the IMM is applied to model a stented aortic heart valve under physiological conditions, and the results are in good agreements with available experimental and numerical results.

7.1 TWO-DIMENSIONAL IMMERSED MEMBRANE COMPUTATIONS

The proposed immersed membrane method is firstly applied to simulate two-dimensional flows with an immersed thin structure. At the beginning, the grid convergence study is carried out, which is followed by a comparison of the 2D steady flow results with those obtained with the baseline solver to validate the newly proposed method. Results of moving rigid and flexible membranes are then presented.

7.1.1 Grid Convergence Study

The model used for the grid convergence study is a 2D channel with aspect ratio of 6:1, as depicted in Figure 7.1. The width of the channel is 1, and the length is 6. A membrane with length of 0.5 is fixed at the middle of the channel. The membrane is vertical to the wall of the channel. The left side of the channel is inlet and the right side outlet. Three grids of different sizes are used to simulate the steady flow in the channel. Grid (I) has 2,000 nodes, grid (II) has 18,000 nodes and grid (III) has 32,000 nodes. The Reynolds number of the flow is 100, and the inflow velocity is 1.0 m/s. All the fluid and geometrical parameters for the three simulations are kept the same. The grid convergence of the solver is verified by comparing the numerical results of the flow field obtained from the three grids. If the results change little when the computational grid gets finer and finer, then it can demonstrate that the simulation results are independent of the computational grid and the solution is grid convergent. Figure 7.2 shows the flow field obtained from grid (I), where only one primary vortex appears. Figures 7.3 and 7.4 show the results obtained from grids (II) and (III). In these two results, two vortices appear in

the flow fields. One is the primary vortex behind the membrane. The other is a small secondary vortex before it. This shows that grid (I) with 2,000 nodes is not fine enough to capture all the characteristics of the flow field. The comparison of the flow fields is shown in Table 7.1. The results of grids (II) and (III) are close to each other with the centers of vortices located at similar locations and of similar vorticities. The velocity profiles extracted at same locations agree well with each other as shown in Figure. 7.5(a), while the velocity profile of grid (I) at the same location is very different to those of the other two grids. The convergence history shown in Figure 7.5(b) demonstrates that the solver developed based on the immersed membrane method has fast convergent rate. From the above analysis, it is concluded that the grid of 18,000 grid nodes is the optimum grid to use.

Grid	Secondary Vortex			Primary Vortex		
	vortex center	vorticity	length of vortex	vortex center	vorticity	length of vortex
(I) 2,000 nodes	-	-	-	(0.75, 0.31)	-6.69	1.70
(II) 18,000 nodes	(-0.06, 0.06)	-0.23	0.17	(1.34, 0.31)	-7.44	2.57
(III) 36,000 nodes	(-0.07, 0.06)	-0.26	0.17	(1.50, 0.31)	-7.30	2.62

Table 7.1 Properties of the vortices of the three grids.

7.1.2 Validation

To validate the proposed scheme and test its performance, a stationary rigid membrane is employed for the immersed membrane method. The model examined in this section is a 2D channel with a sinus cavity in the middle of the bottom wall, as sketched in Figure 7.6. A rigid membrane is attached to the channel just before the sinus cavity. The channel has a length of $6L$ and a width of $1L$ (L is equal to 20mm). The radius of the sinus cavity is $0.5L$. The immersed membrane has a length of $0.5L$ and thickness of $0.5L/100$, and it is attached to the bottom wall at an angle of $\alpha = 42.5^\circ$. The 2D channel is meshed with triangular cells. In the region near the membrane, the mesh is further refined to catch detailed flow characteristics during the movement. The immersed membrane is discretized into 10 elements. Close-up views of the mesh and the membrane are shown in Figure 7.7 (a). Since the membrane is considered rigid, the Young's modulus is set to a very high value. The inflow velocity V is 1.0m/s and the Reynolds number is 100.

The results are compared with those of a channel with internal boundary representing a membrane which is calculated by a well-validated 2D Navier-Stokes solver **DELTINKE** [153]. The internal boundary has the same geometry as the immersed membrane which is shown in Figure 7.7 (b), and it is under the same flow conditions as the immersed membrane. Figure 7.8 shows the convergence history of the immersed membrane and the internal boundary. The new solver based on the immersed membrane scheme converges faster and better than the baseline solver. Figure 7.9 shows that the two flow fields have the same number of vortices with similar shapes, and the velocity profiles at the same locations agree well with each other as shown in Figure 7.10. Table 7.2 compares the two flow fields quantitatively, which also demonstrates that the results

obtained by the immersed membrane method agree well with those obtained by the standard Navier-Stokes solver using the internal boundary.

	Immersed Membrane		Internal Boundary	
	primary vortex	secondary vortex	primary vortex	secondary vortex
Center	(0.926L, 0.270L)	(0.077L, -0.035L)	(0.925L, 0.275L)	(0.046L, -0.026L)
Vorticity	-4.992	0.116	-4.995	0.117

Table 7.2 Comparison of flow field with immersed membrane and flow field with internal boundary (L=20mm).

7.1.3 Moving Rigid Membrane

The rigid membrane in the developed steady flow as shown in Figure 7.9(a) is then released and moved by fluid forces. The immersed membrane is hinged at the bottom wall at one end and it is allowed to rotate around this end. Figure 7.10 shows a sequence of computed flow fields with corresponding membrane positions at different non-dimensional time instants. Streamlines are shown to illustrate the evolution of the flow field. Starting from the steady position, the membrane is pushed forward around the joint point until it reaches its resting position almost parallel to the inflow velocity. The small vortex in the sinus cavity disappears soon after the membrane starts moving at $t=0.05$, while the primary vortex is pushed downstream along the bottom wall and its size decreases due to viscous dissipation until it disappears at approximately the end of the downstream field. The CPU time for this simulation is 216 minutes on a SGI Octane Workstation. The whole process is divided into 50 time steps. Within each time step, the

number of pseudo time sub-iterations is set to 100 in order for the computation to achieve the divergence free conditions

7.1.4 Moving Flexible Membrane

In this section, a flexible membrane is studied by the immersed membrane method. The computational model examined in this section is slightly different. The radius of the sinus cavity is $0.8L$, and two small rounds are added to where the sinus cavity joins the bottom wall. The membrane is fully clamped by the wall at the lower end, while the other one is a free end. The flexible membrane begins moving in the developed steady flow as shown in Figure 7.12, in which the membrane is stationary and the Young's modulus is set to a very high value to harden the membrane to obtain the steady flow. The inflow velocity V is 1.0 m/s and the Reynolds number is 100. Table 7.3 lists geometrical and material parameters of the flexible membrane and the flow field.

Fluid Parameters		Geometric Parameters of Membrane		Material Parameters of Membrane	
ρ_f (kg/m ³)	$1.0 \bullet 10^3$	Length (mm)	0.5L	ρ_m (kg/m ³)	$1.0 \bullet 10^3$
Re (-)	100	Thickness (mm)	0.5L/50	E (N/m ²)	$1.5 \bullet 10^6$
V (m/s)	1.0	α	45°	ν	0.49

Table 7.3. Parameters of a flexible membrane.

Figure 7.13 shows the motion of the immersed flexible membrane. Starting from the steady flow, the flexible membrane bends and moves towards the sinus cavity. It is found that the cause of the bending of membrane at its center is due to the fact that the pressure difference at the center is the greatest, which result in the most significant bending there. The vortex decreases in size and moves downstream due to the motion of the membrane. Finally it disappears at the corner where the sinus cavity joins the bottom wall. The membrane continues moving towards the sinus cavity until it rests in the sinus cavity slightly declining inwards the sinus. The CPU time for this simulation is 307 minutes on a SGI Octane Workstation.

7.1.5 Remarks

The above two examples demonstrate the capacity of the immersed membrane method to analyze large deformations, complex and time-dependent motions of immersed thin structures driven by viscous flow, without changing the stationary background mesh.

7.2 SERIAL COMPUTATION OF BI- LEAFLET MECHANICAL HEART VALVE

In this part, the proposed IMM method is applied to compute the flow fields around a St. Jude 29 bi-leaflet aortic mechanical heart valve for opening and closing phases under physiological conditions with a serial solver. Only one of the two valve leaflets is computed due to the symmetry of the two leaflets, and the results are mirrored along the plane x - z to obtain a full view of the flow field. The results of flow field are shown in the form of velocity vector plots, streamline distributions and pressure contours with corresponding valve positions. Detailed stress and pressure distributions on the leaflet

surfaces are presented for analysis of loading variations of the valve. Comparison of computed results with experimental results conforms that the method can accurately simulate the opening/closing phases of the MHVs.

7.2.1 Model Properties of St. Jude Bi-leaflet Mechanical Heart Valve

In this MHV model, the annulus diameter is 29mm, the diameter of the internal flow channel is 25mm, and the thickness of the valve leaflet is 0.8mm [154]. At fully open and fully closed positions, the valve leaflets are 5° and 60° respectively away from the x - z plane as illustrated in Figure 7.14. Thus the valve leaflet moves a total angle of 55° during its opening and closing phases. The density of the valve leaflets ranges from $1.5 \times 10^3 \text{ kg/m}^3$ to $2.0 \times 10^3 \text{ kg/m}^3$ [154]. In this work, the density is set to be $1.8 \times 10^3 \text{ kg/m}^3$.

The density of human blood is $1.06 \times 10^3 \text{ kg/m}^3$ and its dynamic viscosity is $\eta = 3.5 \times 10^{-3} \text{ N}\cdot\text{s/m}^2$. The pulsatile inflow velocity profile is specified according to the physiological mass flow rate of a normal human based on the requirements for pulsatile flow testing of heart valve substitutes published by British Standard organization [155]. In this study, a set of time-dependent inlet velocity equation for both the opening and closing phases are devised according to the profile shown in Figure 7.15 and is given as follows [44, 45],

$$U_{\text{in}} (\text{m/s}) = \begin{cases} 0.0 & \text{for } 0 < t \leq 0.075T \\ U_0 \cdot \sin(5.0\pi(t - 0.075T)/T) & \text{for } 0.075T < t \leq 0.175T \\ U_0 \cdot \cos(1.98\pi(t - 0.175T)/T) & \text{for } 0.175T < t \leq 0.44T \\ 0.0 & \text{for } 0.44T < t \leq T \end{cases} \quad (7.1)$$

where $U_o = \pm 0.5\text{m/s}$ (positive for opening phase and negative for closing phase), and T is the period for one cardiac cycle which is 0.83s and this corresponds to the normal heartbeat of 72 beats/min . Thus the Reynolds number, defined as $Re = \frac{\rho U_{\infty} D}{\eta}$, reaching a value around $4,000$ at peak mainstream velocity at the moment of maximum systolic flow, will result in turbulent flow. However, the application of this Reynolds number is not feasible due to the limited computing resources and the lack of a suitable model to simulate both the turbulent and laminar flows simultaneously. Under the pulsatile inflow condition, the presence of turbulence is a very brief time within a cardiac cycle. So in this work, the Reynolds number is set to a moderate value of $1,800$ to limit the flow condition to unsteady laminar flow.

The MHV is immersed in a circular vessel with the same internal diameter of 25mm . The length of the vessel is 100mm , and the valve is located at 35mm away from the inlet as shown in Figure 7.16. A grid convergence study was carried out and a mesh size of $50,096$ nodes and $251,547$ tetrahedral elements was chosen for the fluid domain. While the mesh size of the valve leaflet is of $4,200$ nodes and $8,151$ triangular elements. Initially, velocities at all nodes are set to be zero except at the inlet the pulsatile inflow velocity is imposed. And a fixed pressure of 10Kpa is assigned at the outlet. No-slip boundary condition is specified on the rigid vessel walls and the leaflet surfaces. Slip boundary condition is applied for the symmetry plane.

7.2.2 Results for Opening Phase

The opening phase occurs from $t=0.0\text{s}$ to $t=0.14\text{s}$ according to the pulsatile inflow velocity profile depicted in Figure 7.15. From $t=0.0\text{s}$ to $t=0.06\text{s}$ the velocity of the valve

leaflet is set to be zero since the inflow velocity is zero. Only a very small amount of fluid flows in to push the leaflets as shown in Figure 7.18(a). The valve leaflets start moving from $t=0.06s$ when the inlet velocity increases. A series of successive pressure contours of x - y plane at $z=0$ with the corresponding leaflet positions during valve opening phase are shown in Figure 7.17. At $t=0.105s$, the leaflets reach their fully open positions, and from $t=0.105s$ to $t=0.14s$ they remain fully open until the start of the leaflet closing phase. The opening of the valve leaflets takes about $0.045s$, which agrees well with the experimental and numerical results of Shi [156]. This opening process is a very brief period compared with the whole opening phase of $0.14s$. It agrees with the fact that during the opening phase the leaflets open very fast and leave much of the time of the opening phase for blood to go through aortic valve. Pressure difference between the upstream and downstream of the valve leaflets keeps decreasing as the valve opens. Finally pressures at upstream and downstream of the leaflets are almost the same when the valve leaflets are resting at their fully open positions ($\text{angle}=5^\circ$). Figure 7.18 shows six successive valve positions with corresponding velocity vector fields and volume streamlines around the valve leaflets of the opening phase. In Figure 7.18(b), three jets form when the leaflets start moving at $t=0.06s$. One jet is at the central orifice of the two leaflets. The other two are at the small gaps between the leaflets and the solid housing wall. In the vector plots, it can be observed that near the region of the leaflets, fluid velocity is vertical to the leaflet surface. It implies that the leaflets follow the motion of the fluid or vice versa, which is an intrinsic phenomenon of dynamic fluid-structure interaction, because surface velocity on the rotating leaflets is always perpendicular to the surfaces. Figure 7.19 shows the pressure and fluid shear stress distributions on two surfaces of the valve leaflet. Figures 7.19(a1)-(f1) are the pressure distributions on the leaflet surface of the upstream side, and 7.19(a2)-(f2) are the pressure distributions on the

leaflet surface of the downstream side. On the upstream leaflet surface, high pressures appear near the trailing edge of the leaflet. On the downstream leaflet surface, high pressures are at the central region. It is found that pressures on both surfaces across the leaflet have very different values and distribution patterns. This demonstrates that the immersed membrane method can consider the discontinuous fluid properties across the immersed structure. Figures 7.19(a3)-(f3) are the fluid shear stress distributions on the upstream surface and 7.19(a4)-(f4) are the fluid shear stress distributions on the downstream surface. The shear stress also has radical changes on both sides of the valve leaflet. It is observed the fluid shear stress is about 100 times smaller than the fluid pressure, so the pressure difference is the major cause of the valve motion. On both the upstream and downstream surfaces, the higher shear stress is near the trailing edge of the leaflet.

The physical time step size is set to 0.0005s to track the transient movement of the valve leaflet. In each time step, 400 sub-iterations are used to achieve the divergence free condition. The order of the residual is 10^{-6} . The CPU time of the simulation is 780 minutes on one node of a SGI O3400 server.

7.2.3 Results for Closing Phase

The closing phase of the 29mm bi-leaflet MHV is computed using the same model as that of the opening phase. During the closing phase, since the flow velocity direction is reversed, the inlet side is downstream to the valve while the previously downstream outlet is upstream to the valve now. To eliminate the downstream effects, the distance between inlet and the position where the valve is seated is elongated to 65mm, and the distance between the outlet and the valve is shortened to 35mm to maintain the total

length of the model to be 100mm. The pulsatile inflow velocity profile for closing phase from $t=0.14\text{s}$ to $t=0.83\text{s}$ is applied at the new inlet and a fixed pressure of 10Kpa is assigned to the new outlet. The flow field of closing phase is far more complex than that of the opening phase because of the reversal of the flow velocity.

Initially a steady flow shown in Figure 7.20(a) and Figure 7.21(a) is developed for the closing phase by fixing the leaflets at their fully open positions. The leaflets start moving in the developed steady flow. A series of pressure contours of x - y plane at $z=0$ with corresponding valve positions are shown in Figure 7.20. It shows that pressure of the upstream side of the leaflets is larger than the pressure of the downstream side. The pressure difference will push the leaflets towards their closing positions. At the end of the opening phase, pressures at both the leading and trailing edges of the leaflets are almost the same, so the leaflets remain at their fully open positions for about 0.035s from $t=0.105\text{s}$ to $t=0.14\text{s}$. When pulsatile inflow inverses its direction after $t=0.14\text{s}$, the pressure difference gradually increases. In Figure 7.20(b), it is observed that the pressure increases at the upstream side, and the leaflets are pushed to an angle of 11° . The leaflets continue to rotate towards the fully closed position with an increasing pressure difference across the valve as shown in Figures 7.20(c)-(f). The pressure difference is more prominent at the end of the closed position as observed in Figure 7.20(g). This huge transvalular pressure difference pushes the leaflets to close tightly so that no backflow of blood is allowed from the left ventricle into the aorta. A series of consecutive velocity vector plots and streamline plots with corresponding leaflet positions are shown in Figure 7.21. In the closing phase, the flow field still has three channels formed as in the opening phase. At early stages of the closing phase, the flow field is quite smooth and no vortex appears as shown in Figures 7.21(b)-(d). As the leaflets further approach fully closing positions, the

flow field becomes more and more obstructed by the leaflets as shown in Figure 7.21(e)-(g). The irregular velocity streamlines indicate that the flow field is disturbed by the closing leaflets. In Figure 7.21, it is observed that a jet is formed in the central channel due to the converging flow obstructed by the leaflets. Velocity of the jet increases as the leaflets approaching the fully closed positions. Near the fully closed positions, squeeze flow appears in the side channels. Figure 7.22 shows the pressure and fluid shear stress distributions on the two surfaces across valve leaflets. Both the pressure and shear stress have radical changes across the valve leaflets. At early stages of the closing phase, pressures on both upstream and downstream surfaces have small difference, and pressures at the leading edges are higher than pressures at the trailing edges as shown in Figure 7.22(a1)-(c1) and Figure 7.22(a2)-(c2). As the leaflets move towards their closed positions, the pressure difference on both the upstream and downstream surfaces increases as shown in Figure 7.22(d1)-(g1) and Figure 7.22(d2)-(g2). The pressure is much higher than the shear stress as in the opening phase.

The physical time step size is set to 0.0005s. And 300 sub-iterations are used to achieve the divergence free condition. The CPU time of the simulation is about 670 minutes on the SGI O3400 server.

7.2.4 Remarks

An immersed membrane method is proposed in this work to compute the fluid-structure interaction problem in an MHV. In this method a stationary Eulerian background mesh is used for the computation of the fluid domain, thus avoiding the update of background mesh. By introducing fluid ghost nodes in the computations involving mesh nodes across the structure, a set of ghost values are linearly extrapolated from their corresponding real

nodes to replace the real nodes in the cross-structure fluid computations. These one-sided extrapolations can account for radical changes in fluid conditions across the immersed thin structure and provides a promising way to improve the accuracy of the FSI simulations. After computation of the fluid domain, the structure is driven directly by the fluid forces according to the Newton's Second Law of Motion, thus inertial effects can be considered in the computation. The flow field in a real St. Jude 29mm was studied by the immersed membrane method using a 3D Navier-Stokes solver. The results of the opening and closing processes agree well with other available numerical results and experimental results.

7.3 VALIDATION OF PARALLEL SOLVER WITH IMM

7.3.1 Grid Convergence and Accuracy Study

This study is performed for a case where a solid oscillating sphere is immersed in a fluid enclosed within a domain with solid no-slip walls, similar to the cases studied by Gilmanov [157] and Udaykumar [158]. The flow condition is depicted in Figure 7.23. A rigid sphere of diameter $D = 1$ is initially placed at the center of the cubic box with edge $L = 2$ units within an incompressible and viscous fluid, which is initially at rest. Flow is induced by oscillating the immersed sphere back and forth horizontally along the x -axis. The oscillating velocity and location of the center of the sphere are prescribed as follows:

$$V_x = 0.25\pi \sin(2\pi t); \quad V_y = V_z = 0; \quad 0 \leq t \leq 1 \quad (7.2)$$

$$x_n(t) = x_n^0 + A[1 - \cos(2\pi t)]; \quad y_n(t) = y_n^0; \quad z_n(t) = z_n^0; \quad A = 0.25D \quad (7.3)$$

where x_n , y_n and z_n are the coordinates of the n -th node of the surface of the sphere. The motion is impulsively initiated at $t=0$ with an amplitude of $A=0.25D$ and a non-dimensional time period of 1. The Reynolds number for this flow is defined as $\frac{V_x D}{\nu}$ and it is set to be 50 for this study.

Four uniformly spacing and successively finer grids are used for this study, with 11^3 , 21^3 , 41^3 and 81^3 grid nodes respectively. The meshes are constructed by two grid generation programs: **FORGEN** and **ST2UNST** [159]. First in **FORGEN**, a structured grid is generated by specifying the length and the number of intervals for each edge. The grid nodes are placed along each edge with uniform spacing decided by the edge length and interval numbers. Taking the grid of 11^3 nodes for example, the length of the cubic edge is 2 and 10 intervals are specified for each edge, so the interval distance h_I is 0.2 and 11 nodes are placed in each direction along x , y , and z coordinates. With such spacing, a total of 11^3 nodes can be placed in the cube, and the structured grid is finally generated by connecting all the nodes into an assembly of hexahedral elements. In the construction of the next level of finer grid of 21^3 nodes, the current interval of $h_I=0.2$ is further divided into two smaller intervals of $h_{II}=0.1$, thus increasing the number of nodes in each direction to 21, and the total number of nodes to 21^3 . The next two grids are constructed in the same manner as intervals of $h_{II}=0.1$ are divided into $h_{III}=0.05$ with 41 nodes in each direction and the total number of nodes of 41^3 , and intervals of $h_{III}=0.05$ are divided into $h_{IV}=0.025$, resulting in 81 nodes in each direction and a total of 81^3 nodes. The grids are constructed in this way in order to identify the same nodes for calculating errors in the four grids. By consecutively dividing the intervals, only new nodes are added into the finer grids and old nodes are kept in the finer grids but with different numberings. A relation of the numberings can be established to identify the same nodes, which will be

used for the error analysis for the four grids. In the program **ST2UNST**, the structured grids are transformed into unstructured grids by splitting the hexahedron elements into tetrahedron elements by simply connecting the diagonal grid nodes of the structured hexahedral elements. The construction process for consecutively finer meshes is illustrated in Figure 7.24 with two simple 2D meshes. To perform SPMD parallel computation, the four grids are partitioned into sub-domains. Since eight processes are used in the parallel computing, all the grids are partitioned into eight sub-domains as shown in Figure 7.25. Although the first two grids of 11^3 and 21^3 have smaller number of nodes, they are still partitioned into eight sub-domains in order to keep consistency in the grid convergence study in the parallel computation. The surface of the rigid sphere is discretized into an unstructured mesh of 7,308 nodes and 14,612 triangular elements.

The simulations are run for one complete time period. A small time step size of $\Delta t = 0.005$ is chosen for all the simulations in order to minimize the effect of temporal errors on solutions. So the simulation for one oscillation period is divided into 200 time steps, and in each time step, 400 sub-iterations are used for pseudo-time stepping in order to ensure divergence-free conditions. At the end of the oscillation period, flow fields obtained on the four grids are recorded and compared for grid convergence study. Figure 7.26 shows the streamline and vector plots for the mid planes at $y = 0$ using the four grids. Because the sphere oscillates back and forth in an incompressible fluid confined in the cube enclosed by wall boundaries, at the end of the oscillation period, a recirculating flow is induced with streamlines starting from the leading side ($-x$ side) and ending at the trailing side ($+x$ side) of the sphere. It is observed that two primary vortices appear symmetrically about the equatorial planes in all the four flow fields at similar positions with similar vorticities. However, for the grid of 11^3 nodes, it is too coarse to capture the

small vortices near the corners of the cube. For the grid of 21^3 nodes, it shows some signs of the development of the small vortices, though not apparent. In the grid of 41^3 nodes, the two small vortices develop well near the corners, and their positions and vorticities are in good agreement with the results obtained on the grid of 81^3 nodes. So the grid of 41^3 nodes are fine enough to obtain accurate solutions, and the solutions are grid independent since the results change little when the grid is finer than that of 41^3 nodes.

7.3.2 Error Analysis

To demonstrate the second-order accuracy of the proposed immersed membrane method (IMM) for moving boundaries and fluid-structure interactions, an error analysis is carried out on the solutions obtained on the four consecutively fine meshes in the grid convergence study. Without analytical solution to this flow problem, the solution on the finest grid used in the study, the grid of 81^3 nodes, is taken to be the “exact” solution, which is used as a basis in calculating errors of the coarser grids. The errors are calculated for x-component velocities of all grid nodes under consideration at the end of the simulation period. For a grid of N^3 nodes, the k^{th} norm of x-component velocities are calculated as follows:

$$\mathcal{E}_N^k = \left[\frac{1}{N^3} \sum_j^{N^3} \left| (V_x)_j^{(N^3)} - (V_x)_j^{(81^3)} \right|^k \right]^{1/k} \quad 1 \leq k \leq N^3 \quad (7.4)$$

and the infinity error norm of x-component velocities is calculated as:

$$\varepsilon_N^\infty = \max_{j=1, N^3} \left| (V_x)_j^{(N^3)} - (V_x)_j^{(81^3)} \right| \quad 1 \leq t \leq N^3 \quad (7.5)$$

where $(V_x)_j^{(N^3)}$ is the x-component velocity of the j^{th} node of the grid with N^3 nodes, $(V_x)_j^{(81^3)}$ is the x-component velocity of the same node in the finest grid as the j^{th} node in the grid of N^3 nodes. The relationship between numberings of one node in different grids is established when constructing the grids as described in the previous section.

The convergence history of L_1 , L_2 and L_∞ error norms for the three coarser grids with respect to the finest grid are shown in Figure 7.27 against grid spacing h in a log-log coordinate system. Slope 2 and slope 1 are also shown in the plot for comparison. In the plot, both L_1 and L_2 are very close to slope 2, actually even steeper than slope 2, which suggests that the current method converges at a rate slightly higher than the second order. The overall slope for L_∞ is about 1.6, a small deviation from the second order convergence rate. This may be caused by the fact of treating the solution on the finest grid as “exact” solution while not using the exact analytical solution since it is not available for the governing equations. The results of error analysis confirm that the proposed immersed membrane method is second-order accurate, which is stated in the numerical formulation in Chapter 5.

7.4 VALIDATION OF PARALLEL SOLVER

The parallel solver with immersed membrane method for fluid-structure interaction problems is validated by performing simulations for the axially symmetrical steady flows induced by a sphere rotating at constant angular velocities about a diameter in an

incompressible fluid which is rest at large distance from it under moderate Reynolds numbers. In this study, radius of the sphere is R and the fluid domain is a cube with edge of $10R$. The sphere is placed at the center of the cube. It rotates about z -axis at a constant angular velocity of ω as shown in Figure 7.28. The Reynolds number is defined as $\frac{\omega R^2}{\nu}$.

For Reynolds number between 1-100, Dennis *et al.* [160] have obtained analytical solutions by solving the governing Navier-Stokes equations with a vorticity-streamfunction formulation in spherical polar coordinate systems. And Gilmonov *et al.* [157] reported their numerical simulation results for this problem at the Reynolds numbers of 20, 50 and 100. Under moderate and low Reynolds numbers, the resultant flow fields are steady and axisymmetric. However, intermediate unsteady simulation results obtained in the process from the sphere impulsively starts rotating from rest to the final steady status are also presented in this study. The simulation of intermediate unsteady flows induced by the rotating sphere provides a sound validation for the accuracy of the parallel solver and infuses some confidence about the correct implementation of the numerical infrastructure developed in this work for solving fluid domains with immersed moving structures.

The computational domain is a closed cube of edge equal to $10R$ and with non-slip boundary conditions applied on all the six surfaces. Such a large fluid domain is chosen in order to minimize the boundary effects on fluid flows. The fluid domain is discretized into a uniformly spacing grid of 42^3 nodes, and for parallel computation it is partitioned into eight sub-domains as shown in Figure 7.29. Surface of the sphere is discretized into an unstructured mesh of 7,306 nodes and 14,612 triangular elements. It starts rotating impulsively at $t = 0$ around z -axis with a constant angular velocity of ω in

the clock-wise direction, so the velocities of all the structure nodes are prescribed as the follows,

$$\begin{aligned}
 V_x^i &= -\omega R \cos \phi^i \sin \theta^i, \quad V_y^i = \omega R \cos \phi^i \cos \theta^i, \quad V_z^i = 0 & \text{if } x_i \leq 0 \quad y_i \leq 0 \\
 V_x^i &= \omega R \cos \phi^i \sin \theta^i, \quad V_y^i = -\omega R \cos \phi^i \cos \theta^i, \quad V_z^i = 0 & \text{if } x_i \geq 0 \quad y_i \geq 0 \\
 V_x^i &= \omega R \cos \phi^i \sin \theta^i, \quad V_y^i = \omega R \cos \phi^i \cos \theta^i, \quad V_z^i = 0 & \text{if } x_i \leq 0 \quad y_i \geq 0 \\
 V_x^i &= -\omega R \cos \phi^i \sin \theta^i, \quad V_y^i = -\omega R \cos \phi^i \cos \theta^i, \quad V_z^i = 0 & \text{if } x_i \geq 0 \quad y_i \leq 0 \quad (7.6)
 \end{aligned}$$

where V_x^i , V_y^i and V_z^i are velocity components of the i^{th} structure node, θ^i is the angle between the structure node and the plane of $x = 0$, ϕ^i is the azimuth angle of the structure node. Simulations are carried out for Reynolds numbers of 10, 20, 50 and 100. In all the simulations, a physical time step of $\Delta t = 0.01T$ is used, where $T = \frac{2\pi}{\omega}$ is the rotation period of the sphere. Therefore, simulation for one rotation period is divided into 100 physical time steps, and in each time steps 200 sub-iterations are used for pseudo-time stepping in order to obtain fairly accurate intermediate unsteady results.

7.4.1 Unsteady Flows at Reynolds Number of 100

The intermediate unsteady flows of $Re=100$ are presented first to demonstrate capacity of the parallel solver to capture accurate transient results. The unsteady flows are recorded at the time instants of $t = 0.1T$, $t = 0.5T$, $t = 1.0T$, $t = 2.0T$, and the flow field reaches steady status around $t = 3.0T$. The instantaneous plots of streamlines with contour for x-component velocity and vector plots of the unsteady flows at $y = 0$ plane are shown in Figure 7.30, together with the steady flow recorded at the time instant of $t = 3.0T$. These

plots document the evolution history of the flow field due to a rotating sphere at the Reynolds number of 100. At early stage of the impulsive rotation, two small vortices form near both southern and northern poles of the sphere as shown in Figure 7.30(a). But subsequently the four symmetrical vortices strengthen themselves as they move towards the equatorial plane as shown in Figure 7.30(b) and (c). And near the equatorial plane, the vortices change their moving direction to the radial direction and move away the sphere as shown in Figure 7.30(c), 7.30(d) and (e). Finally the vortices stabilize near the equatorial plane with some distance from the sphere, and the flow field reaches its final axisymmetric steady state as shown in Figure 7.30(e). It is worth noting that the flow remains perfectly axisymmetrical during the whole evolution process. This flow phenomenon is caused by the imbalance between the component of the centrifugal force in the tangential direction to the sphere direction and the pressure gradient, which is directed in the direction normal to the sphere. The induced force imbalance will direct the fluids around the sphere flowing from the poles to equator. And the two converging fluids from both poles collide at the equatorial plane and form a jet flow which flows away from the sphere in radial direction. To satisfy the mass conservation, the streamlines have to close and form two toroidal vortices located symmetrically with respect to the equatorial plane. Numerical results of Gilmanov [157] are shown in Figure 7.31 for qualitative comparison, and the same flow patterns were also observed by the researcher.

7.4.2 Steady Flows Due to A Rotating Sphere

Steady flows at low and moderate Reynolds numbers are also simulated in this work for quantitative comparison. The steady flow patterns for Reynolds numbers of 10, 20, 50 and 100 are shown for the plane of $y = 0$ in Figure 7.32. The main feature of these steady

flows is that the polar inflow is balanced by the outflow at the equator. The inflow changes to an outflow at a critical angle which depends on Reynolds number. This inflow angle θ_{inflow} is the angle between z -axis and the radius that passes through vortex center as depicted in Figure 7.32(a). The inflow angles are 56.7° , 61.2° , 71.8° and 75.9° for Reynolds number 10, 20, 50 and 100, respectively. It is observed that the inflow angle increases as the Reynolds number increases, which indicates that the polar inflow gets slower and expands a larger region as the Reynolds number increases, and the equatorial jet flow gets faster and narrower. It agrees with the fact that with a larger Reynolds number the sphere rotates at a larger angular velocity and induces a greater influence on the surrounding fluids. The values of θ_{inflow} at different Reynolds numbers are tabulated in Table 7.4 with the analytical results obtained by Dennis [160] and the numerical results obtained by Gilmonov [157] for quantitative comparison. The Flow fields obtained by Gilmanov are also shown in Figure 7.33. The numerical results in this work agree excellently with the previous works. It is proved that the proposed immersed membrane method are capable to solve the dynamic fluid-structure interaction problems accurately.

Re	Dennis [160]	Gilmonov [157]	Current	ε % (compared with [160])
10	58.0°	-	56.7°	2.2
20	62.6°	61.6°	61.2°	2.2
50	69.4°	68.2°	71.8°	3.4
100	73.8°	72.1°	74.9°	1.5

Table 7.4 Polar inflow angles at different Reynolds numbers compared with [157] and [160]

7.5 PARALLEL COMPUTATION OF BIO-PROSTHETIC HEART VALVE

The bioprosthetic heart valves (BHVs) replace diseased natural aortic valves and serves their function to allow the blood flow only in one direction from the left ventricle to the aorta. A desirable BHV is expected to cause as little resistance to blood flow as possible, and it should seal firmly to prevent any regurgitation when the valve closes. The BHVs have better hemodynamic performance compared with mechanical heart valves (MHVs), but their long-term performance is disappointing. Most of the degeneration of BHVs can be attributed to calcification and tearing of leaflets. It is believed that the calcification and tearing of valve leaflets are related to the distribution of surrounding fluid stresses and internal structure stresses. Therefore it is important to understand the flow dynamics and stress distributions of the valve leaflet under physiological conditions. With the advent of high-speed computers and the development of CFD and CSD analysis algorithms, bio-fluid and structural dynamic simulation of valvular functions provides a promising alternative in the detailed analysis of complex flow dynamics during opening and closing phases. These numerical simulations can be used to analyze detailed fluid dynamic stress distributions in the region of interest. Researchers have identified four phases of the valve within one cardiac cycle, which are the opening phase, the fully opened phase, the closing and the fully closed phases. Among theses four phases, the fully open and fully closed phases are relatively simple due to their steady flow field structure, and many researchers have intensively and extensively studied these two phases. For the opening and closing phases, the flow phenomena are far more complex and the limitations of computing resources also contribute to the slow progress in their study. One of the important requirements of an accurate simulation of valve functions is the development of fluid-structure algorithm for the prediction of leaflet motion resulting from

hemodynamic forces acting on the leaflets. In present work, involving FSI, the computation is first carried out by solving the Navier-Stokes equations for the fluid flow field using the numerical method described in Chapter 3 and the IMM method with the valve leaflets immersed into the fluid computational domain. Then hemodynamic forces on the valve leaflets are computed for solving the structural dynamics equation. The governing equation for structure domain is solved by using the numerical methods developed in Chapter 4. New velocities and spatial configurations are available once the structural equation is solved. And with the new velocities and positions, fluid flow is calculated in a new time step. In this numerical simulation, attention is not focused on the determination of an optimal artificial valve, rather on investigation of the flow fields around the BHV and the response of the valve leaflets. And the study is also focused on the complex features and the fluid dynamics of the moving leaflets due to the blood-leaflet interaction. In this section, the motions of a stented BHV for both opening and closing phases and blood flow through it are computed. Results are presented in terms of fluid pressure contours, fluid stress distributions on leaflet surfaces, streamlines, velocity vectors and extracted velocity profiles.

7.5.1 Opening Phase

7.5.1.1 Computational Model

Although in reality the three leaflets of a bioprosthetic heart valve are not precisely identical, it is assumed in this work that they are similar enough that the BHV can be treated as an axisymmetrical structure. Hence due to limited computing resources, only one third of the vessel with one leaflet is considered in the model as depicted in Figure 7.34. The radius R of the model is 12.5mm and the total length L is 150mm. The inlet

side is left ventricle and the outlet side is aorta. The BHV is positioned right after the left ventricle in reality, in order to minimize the inlet and outlet effects on the flow field, the sinus and valve is located 50mm (4R) away from the inlet, and the outlet is 100mm (8R) away from the valve as show in Figure 7.34. The height of sinus is 20mm and the depth is 6mm. The computational domain is discretized into an unstructured mesh of tetrahedral elements. And in the region from 45mm to 80mm away from the inlet, where the valve is located, the mesh is further refined in order to capture detailed fluid-structure interaction phenomena. A grid convergence study was performed on the model and the grid size of 173,885 nodes and 974,752 elements was chosen for the simulation. For parallel computation, the mesh is partitioned into 8 sub-domains (see Figure 7.37).

7.5.1.2 Material and Geometrical Properties of Valve Leaflet

The aortic valve consists of three highly flexible leaflets, which are attached to the aortic root from one commissural point along the aortic ring towards a second commissural point. Behind each leaflet the aortic root bulbs into a sinus cavity to form the beginning of the ascending aorta. The shape of aortic heart valve leaflets in closed and unloaded phase is assumed to be an elliptic paraboloid [164] (see Figure 7.35). In his Ph.D. work, De Hart [3] used an isotropic aortic heart valve for simulations. Following De Hart, the BHV used in this work is modified based on his model. The valve has a length of 15mm and a uniform thickness of 0.04mm. It is assumed in this work that the valve is isotropic for simplicity, although the real valve is not the case. The Young's modulus of the valve leaflets is 100 KPa and the Poission's ratio is 0.3. The valve leaflets are assumed to be neutrally buoyant in the blood flow, so density of the leaflet is taken to be equal to the density of blood, $1.06 \times 10^3 \text{ kg/m}^3$. The properties are tabulated into Table 7.5. The valve leaflet is discretized into an unstructured mesh of 4,225 nodes and 8,216 triangular shell

elements (see Figure 7.36). The valve leaflet mesh is immersed in the fluid mesh as shown in Figure 7.37.

Fluid Parameters		Geometric Parameters of Valve		Material Parameters of Valve	
ρ_f (kg/m ³)	1.06×10^3	Height (mm)	15	ρ_m (kg/m ³)	1.06×10^3
Re	1,500	Thickness (mm)	0.04	E (KPa)	100
U_{in}	Pulsatile Inflow	Height of Sinus (mm)	20	ν	0.30

Table 7.5 Geometrical and material properties of the flow and valve leaflet

7.5.1.3 Initial and Boundary Conditions

As the non-Newtonian nature of human blood affects the flow field only when the diameter of the flow channel is comparable with the size of the blood cell [161-163], the human blood is treated as a Newtonian fluid in this computation, with a density of 1.06×10^3 kg/m³ and a dynamic viscosity of $\eta = 3.5 \times 10^{-3}$ N•s/m². The initial velocity in the flow field is set to zero everywhere except at the inlet. The time-dependent inlet velocity profile for both opening and closing phases is depicted in Figure 7.15, which is specified by equation (7.1) and repeated here:

$$U_{in} \text{ (m/s)} = \begin{cases} 0.0 & \text{for } 0 < t \leq 0.075T \\ U_0 \cdot \sin(5.0\pi(t - 0.075T)/T) & \text{for } 0.075T < t \leq 0.175T \\ U_0 \cdot \cos(1.98\pi(t - 0.175T)/T) & \text{for } 0.175T < t \leq 0.44T \\ 0.0 & \text{for } 0.44T < t \leq T \end{cases}$$

where $U_o = \pm 0.5\text{m/s}$ (positive for opening phase and negative for closing phase), and T is the period for one cardiac cycle which is 0.83s for a normal heartbeat of 72 beats/min . The maximum flow reaches 250ml/s or 0.5m/s , which is approximately half of the maximum physiological flow. A higher flow would instigate numerical instabilities [44]. Within a systolic interval as shown in Figure 7.15, the aortic heart valves undergo three phases of positions, as labeled in the figure as phase A, phase B and phase C. In phase A, the aortic valves are closed from 0.0s to 0.06s . In phase B, the aortic valves open from 0.06s to 0.14s and close from 0.14s to 0.35s . The maximum flow takes place at $t = 0.14\text{s}$. In phase C, the aortic valves remain closed from 0.35s to 0.8s . For incompressible flow in this work, the pressure is treated as relative to the pressure value at the outflow plane and a fixed pressure of 10KPa is assigned to the outflow plane in this study. The Reynolds number, defined as $\text{Re} = \frac{\rho U_{\infty} R}{\eta}$ based on the free-stream velocity, the radius of the flow channel and the density and dynamic viscosity of blood is set to 1500 in this study.

Flow in a typical BHV under the pulsatile flow condition in a cardiac cycle would normally exhibit transition from unsteady laminar to turbulent flow. The need for turbulence modeling arises from the local Reynolds number of the blood flow past the BHV that may easily exceed 7000 . However, under pulsatile flow conditions the turbulence is intermittent, peaking only during the deceleration phase following peak systole [84]. Currently there is still no suitable turbulence model that could successfully describe both the laminar and turbulent flows at the same time and also take into account of their transition with affordable computing resources [165]. Large eddy simulation (LES) could be a choice to overcome this problem, but it is very time-consuming for this type of problems. From the open literature, numerical study of pulsatile flows in prosthetic heart valves is currently limited to unsteady laminar flow only. As the initial

step towards the computation of pulsatile flow through bioprosthetic heart valves, the turbulence effect of the flow field is not taken into account and the entire flow field is considered as an unsteady laminar flow in all the computations.

The aorta walls are assumed to be rigid to mimic a stented BHV, and the valve leaflet is attached to the rigid walls using fully clamped boundary conditions imposed to suppress relative motion of leaflet on rigid wall. No-slip boundary condition is specified on the rigid aorta walls and blood-leaflet interface. Slip boundary condition is applied for the symmetry planes. At the inlet the pulsatile velocity profile is applied as described by equation (7.1) and at the outflow plane a fixed pressure of 10 KPa is assigned for the incompressible flow.

7.5.1.4 Results

The flow is initiated with the leaflet in its fully closed position. A steady flow solution is obtained and used as the initial condition for the moving leaflet computation. For the unsteady flow computation, the CFL is set to 1.0. And a small physical time step size of 5×10^{-4} s is used to obtain accurate unsteady results, so the opening phase with a duration of 0.14s is divided into 280 steps. In each time step, 400 sub-iterations are used for pseudo-time stepping in order to achieve a divergence free condition. All the simulations were run on a SGI Origin 3400 machine with 32 processors based on Silicon Graphics Scalable Distributed-Shared-Memory (DSM) Multiprocessing architecture. It has a system bandwidth of 44.8 GB/sec, 32 GB onboard memory, 800 MFlops/CPU, 8 nodes and each node has four MIPs 64 Bit R12000 400MHz/8MB RISC CPUs. The CPU time for the whole simulation process is about 800 minutes.

According to the pulsatile inflow velocity depicted in Figure 7.15, the opening phase for the valve leaflets occurs from 0.06s (see Figure 7.40(a)) to 0.14s (see Figure 7.40(g)). The results show that the opening phase can be sub-divided into two phases regarding the motion status of the leaflet. Starting moving from 0.06s, the valve leaflet opens quickly and around 0.12s it reaches its maximum opening position (see Figure 7.40(f)). The valve leaflet remains at its maximum opening position until the end of opening phase around $t = 0.14s$ (see Figure 7.40(g)). It is observed that the motion of the valve leaflet is very fast and little time is spent on actual opening, which takes only 0.06s out of the whole cardiac cycle of 0.8s. Most of the time within one cardiac cycle is spent on ejecting the blood to aorta when the valve is open and filling the heart when the valve is closed.

Figure 7.38 shows the fluid pressure and stress distributions on the surfaces of the valve leaflet. Figures 7.38 (i) are the fluid pressure distribution on the surface of the valve leaflet on the left ventricle (LV) side, and Figures 7.38(ii) are the fluid pressure distribution on the surface of the valve leaflet on the aorta side. Figures 7.38 (iii) are the fluid stress distribution on the surface of the valve leaflet on the LV side, and Figures 7.38(iv) are the fluid stress distribution on the surface of the valve leaflet on the aorta side. It is observed that the fluid pressure level is usually 100 times that of the fluid stresses, which indicates that fluid pressure is a major cause for valve motions. At $t = 0.06s$, the valve leaflet is at its initial fully closed configuration. An average transvalular pressure difference of 12KPa is observed at this moment. The pressure difference will push the valve leaflet to move towards its fully open position. The fluid stresses on the two sides of the leaflet are far lower than fluid pressure and they are of similar values because fluid velocity is still close to zero on both sides of it. As the inflow velocity

increases, the valve leaflet is pushed open by forward flow. At $t = 0.07\text{s}$, the valve opens a little and forms a small orifice for the blood flow to be ejected into the aorta. The transvalular pressure is similar to the pressure difference at its fully closed position. High fluid stress is observed on the leaflet surface near to the free edge as shown in Figure 7.38(b), because the forward blood flow passes through the free edge at relatively high speed due to the small valve orifice at this moment. The existence of high fluid stresses near the free edge of valve leaflet indicates that these areas are high-risk regions for leaflet tearing. At $t = 0.09\text{s}$, the valve opens with a larger orifice with an area of about 70% of the maximum orifice area when the valve achieves its fully opened position. At this moment, the transvalular pressure difference is very small compared with that at $t = 0.06\text{s}$. The high stress near the free edges disappears as shown in Figure 7.38(d), because the velocity of forward blood flow is now low over the valve leaflet surface due to a large valve orifice. From $t = 0.12\text{s}$ onwards, the valve reaches its fully open position, the pressure on the LV side equals the pressure on the aorta side. And the valve will no longer move towards the sinus cavity. It remains at this position and fluctuate till the end of the opening phase $t = 0.14\text{s}$.

Figure 7.39 shows a series of pressure contour plots on $y = 0$ plane in the vicinity of the immersed valve leaflet for the opening phase at different time instants. It presents the evolution of pressure distribution of the flow field during the opening phase with the motion of the valve leaflet. Figure 7.39(a) shows the pressure contour at $t = 0.06\text{s}$ when the valve leaflet is about to move. It can be seen that at this moment the average pressure on the left ventricle side is higher than the average pressure at the aorta side. A transvalular pressure difference of 12 KPa is measured across the leaflet at this time instant. This pressure difference on the two sides of the valve provides a driving force to

push the valve forward and toward the sinus cavity until it reaches its maximum opening position. From $t = 0.09\text{s}$ onwards, the pressure difference gradually decreases and finally reaches an equilibrium state at $t = 0.12\text{s}$ for the fully opened position. One consideration for an ideal prosthetic heart valve would be one across which the pressure drop is minimum. The effective orifice area of the device determines the value of this pressure drop. A smooth transition between left ventricle and the ascending aorta depends particularly on the geometry. The viscous effects, in this case, are secondary because the energy dissipated into the bulk fluid by viscosity is small compared with the work done by this pressure gradient.

The vector fields and the streamline distributions near the valve leaflet are given in Figure 7.40 for the opening phase at different time instants with the corresponding positions of the valve leaflet. Figures 7.40(i) show the vector plots on the plane at $y = 0$ and the corresponding configurations of the valve leaflet. Figures 7.40(ii) show the velocity vector plots of the flow field near the valve leaflet. Figures 7.40 (iii) shows the velocity vector plots extracted at $x = 50\text{mm}$ (Plane I), $x = 60\text{mm}$ (Plane II), $x = 70\text{mm}$ (Plane III), $x = 80\text{mm}$ (Plane IV) and $x = 90\text{mm}$ (Plane V). And Figures 7.40(iv) shows the streamline distribution of the flow field around the valve leaflet. From the figures, it is observed that the opening phase is characterized by an accelerating central flow in the mainstream, and little flow is observed in the sinus cavity. The valve opens rapidly, which takes only 0.06s out of the whole cardiac cycle of 0.8s . As the leaflet moves into the sinus cavity, vortices form between the leaflets and the sinus walls. And the leaflet projects slightly into the sinus cavity, see Figure 7.40(g).

At $t = 0.06\text{s}$ as shown in Figure 7.40(a), the valve leaflet starts to move from its initial configuration, corresponding to a fully closed position. Little flow is observed at this moment since velocity of the pulsatile inflow is zero. At $t = 0.07\text{s}$, the valve leaflet is pushed open by the transvalular pressure difference. A forward flow is observed at every node of the fluid domain as shown in Figure 7.40(b), including in the sinus cavity, i.e. the wash-out effect. At this moment, the valve opens with a small orifice of a triangular shape as shown in Figure 7.41(a). Five velocity profiles are extracted from the plane at $y = 0$ at $X = 50\text{mm}$ (plane I), 60mm (plane II), 70mm (plane III), 80mm (plane IV) and 90mm (plane V) as shown in illustrated 7.41. On plane I just in front of the valve leaflet, the velocity profile is fairly flat and the average velocity is about 0.2 m/s as indicated in Figure 7.43(a). At pane II, the average velocity inside the sinus cavity is about 0.1 m/s . And the average velocity within the mainstream is about 0.4m/s due to the small valve orifice at this moment. On plane III, the average velocity inside the sinus cavity is about 0.1 m/s , and the average velocity within the mainstream is about 0.6m/s . The velocity inside the sinus cavity is much lower than the velocity in the mainstream, whose maximum value is about 10-15% of the maximum velocity of the mainstream. Steenhoven *et al.* [169, 170] reported similar findings in their two-dimensional experiments using an aortic valve model where the velocities in the sinus are about 15% of those in the mainstream. It is also observed that in the sinus the velocities on plane II and plane III are of similar values, while in the mainstream the velocity on plane III is about 0.2 m/s higher than that of plane II, which indicates that the rate of change of velocity is higher in the mainstream than that in the sinus cavity. It results in a higher fluid stress on the valve leaflet surface on the mainstream side (LV side) than the fluid stress on the leaflet surface at the sinus side (aorta side), which are shown in frames (iii) and (iv) in Figure 7.38(b). In frame (iii), the average fluid stress on leaflet surface is

about 150 Pa and in frame (iv) the average fluid stress on the leaflet surface is about 50 Pa. On planes IV and V after the valve leaflet, the maximum velocity is 1.1 m/s and 0.9 m/s respectively. Both velocity profiles have parabolic shapes.

The valve opens with a narrow orifice area under the hemodynamic forces at the time instant of 0.08s as shown in Figure 7.41(b). At this moment, the orifice area is still quite narrow showing a strong tendency of blood rushing towards the aorta with an increase in the inlet velocity as shown in Figure 7.40(c). On plane I, the average velocity is 0.2 m/s and the velocity profile does not change much compared with that of the time instant of 0.07s in Figure 7.43(b). On planes II and III, the average velocities inside the sinus cavity are 0.1m/s, which indicates that little flow is observed in the sinus cavity. In the mainstream, the average velocity on plane is about 0.4 m/s on plane II and the velocity profile is flatter due to a larger opening orifice. The average velocity on plane III in the mainstream increases to 1.0 m/s. With higher velocity in the mainstream, the average fluid stress is about 120 Pa on the leaflet surface on the mainstream side as found in Figure 7.38(c). And fluid stresses are found mainly in the region near the free edge of the leaflet because the rate of change of velocity is higher there than that at the central region of the leaflet. On the sinus-cavity side, the average fluid stress is of a low value at around 10 Pa. On planes IV and V, the maximum velocities increase to 1.2m/s and 0.95m/s respectively due to the acceleration of the pulsatile inflow. In the region between $r/R0 = -0.6$ and $r/R0 = 0.0$, the velocity is larger than 0.2 m/s. While at $t = 0.06$ s, this region is between $r/R0 = -0.4$ and $r/R0 = 0.0$ as depicted in Figure 7.43(a).

At $t = 0.09$ s, the valve opens with a moderate valve orifice area as illustrated in Figure 7.41(c). The flow field remains accelerating till this time instant. On plane I before

the leaflet, the average velocity increases to 0.4 m/s. On planes IV and V after the leaflet, the maximum velocities increase to 1.4 m/s and 1.2 m/s respectively with the region of the central flow extending to $r/R0 = -0.75$ as presented in Figure 7.43(c). On plane II, the average velocity in the mainstream increases to 0.6 m/s. And in the region between $r/R0 = -0.60$ and $r/R0 = -1.0$, a negative velocity of -0.1 m/s is observed near the valve leaflet surface in the sinus cavity. Near the sinus wall at $r/R0 = -1.6$, a forward flow of 0.2 m/s is present. This suggests that the mainstream flow enters the sinus at the center of the free edge. It moves along the leaflet surface and along the sinus wall and finally leaves the sinus near the commissures. The same flow pattern was also observed by Peskin *et al.* [26] and Bellhouse *et al.* [168] in the three-dimensional flow field with sinus cavity. On plane III, the average velocity in the mainstream increases to 1.2 m/s. The change of velocity in the mainstream between planes II and III is 0.6 m/s, and in the sinus cavity the change of the velocity near leaflet is 0.3 m/s. Therefore the fluid stress on the leaflet surface at the mainstream side is about 130 Pa and on the sinus side it is about 80 Pa as found in Figure 38 (d).

The valve opens with a large orifice at $t = 0.10$ s as depicted in Figure 7.41(d), whose orifice area is about 80 percent of the maximum orifice area. At this moment, a large region of forward flow is observed in the mainstream as noted in Figure 7.40(e) due to the large valve orifice. The region of central flow in ascending aorta extends to $r/R0 = -0.85$ as shown in the velocity profiles on planes IV and of Figure 7.43(d). The maximum velocity on plane IV registers a value of 1.8m/s and on plane V the maximum velocity is 1.0 m/s. In the mainstream, the average velocities on planes III and II are 1.1 m/s and 0.8 m/s respectively, resulting a velocity difference of 0.3 m/s. In the sinus cavity, the average velocities on planes III and II are 0.15 m/s and 0.1 m/s respectively, with a

velocity difference of 0.05 m/s. The fluid stress on the leaflet surface on the mainstream side is about 80 Pa and on the other side it is about 20 Pa.

From $t = 0.12$ s till the end of the opening phase, the valve reaches its fully open position with a maximum valve orifice as shown in Figure 7.41(d) and (e). The flow is concentrated more in the center and little flow is observed in the sinus cavity as shown in Figures 7.40(f) and (g). The leaflet projects slightly into the sinus cavity. It remains at the fully open position and fluctuates as the blood flow passing through the valve. At $t = 0.12$ s as presented in Figure 7.43 (e), the average velocities on the planes I, IV and V are 0.2 m/s, 0.6m/s and 0.8m/s. On plane II, the mainstream average velocity is 0.6 m/s and on plane III it is 0.8 m/s. In the sinus cavity, the average velocities on both planes are about 0.1 m/s. This results in an average fluid stress of 110 Pa on the leaflet surface on the mainstream side and 10 Pa on the sinus cavity side as found in Figure 7.38(f). At $t = 0.14$ s, the average velocities on the planes I, IV and V are 0.1 m/s, 0.3m/s and 0.4m/s as presented in Figure 7.43(f). On planes II and III, the mainstream average velocities are 0.3 m/s and on plane III it is 0.7 m/s. By comparing the velocity profiles at $t = 0.12$ s and $t = 0.14$ s, it is observed that at the end of the opening phase the flow has started to decelerate already. Similar results were found by Steenhoven *et al.* [169, 170] for *in vivo* experiments and by De Hart [3] in the numerical simulation of a stented aortic heart valve.

7.5.2 Closing Phase

7.5.2.1 Computational Model

The closing phase of a stented BHV based on the parameters listed in Table 7.5 is computed under the physiological conditions specified by equation 7.1. The flow field configuration should not change over the pulsatile period for simulation, so the general flow field structure in the closing phase remains the same as the opening phase. However during the closing phase, the flow reverses its direction. In order to have consistency with the opening phase, the left ventricle side of the BHV is still treated as upstream, and the opposite aorta side downstream. But in the computation, the inlet boundary conditions are applied to the downstream plane, while the outlet boundary conditions are applied to the upstream plane as shown in Figure 7.44. The same mesh as shown in Figure 7.34 is used for the computation of closing phase. No-slip boundary conditions are applied to solid walls and blood-leaflet interface. Slip boundary conditions are applied to symmetry planes. And the valve leaflet is fully clamped into the solid walls.

The computation of the closing phase is far more difficult compared with the computation of the opening phase due to some factors. Firstly, the turbulent effects are more prominent in the closing phase [166, 167], because the closing leaflet blocks the flow and instigate much turbulence. Secondly, the historical development of the flow field affects its current status greatly. However in this work, the historical development of the flow field is not taken into account due to the swapping of the outlet and inlet positions. Without the consideration of the historical development, the current computation might deviate from the real situation. Despite all these difficulties faced, it is still worthy to carry out the computation of the closing phase as a first attempt to obtain

some knowledge of the flow field for the sake of possible future research works in this field.

7.5.2.2 Results

According to the pulsatile flow velocity profile depicted in Figure 7.15, the closing phase occurs from 0.14s to 0.35s. For the unsteady computation, the CFL is set to 1.0. And a small physical time step size of 5.0×10^{-4} s is used to obtain accurate unsteady results, so the opening phase with a duration of 0.21s is divided into 420 steps. In each time step, 400 sub-iterations are used for pseudo-time stepping in order to achieve a divergence free condition. The results for the closing phase are presented in Figures 7.45, 7.46, 7.47 and 7.48 in terms of pressure contours, fluid pressures and stress distributions on leaflet surfaces, velocity vectors and streamlines as well as velocity profiles extracted at several locations. From these figures, it is observed that the valve closure is initiated by the pressure reversal of the fluid domain. The negative transvalular pressure pushes the valve from the fully open position towards its fully closed position. This agrees with Steenhoven's theory [169] that the valve closure is induced mainly by the adverse pressure gradient.

Figure 7.45 presents the pressure contour on plane $y = 0$ with corresponding valve positions at different time instants during the closing phase. From the figures, it is observed that the pressure on the aorta side is higher than the pressure on the left ventricle side during the closing phase. At $t = 0.14$ s as shown in Figure 7.45(a), the pressure difference between the aorta and left ventricle is about 1 KPa. This pressure provides a driving force to push the valve away from its fully opened position towards the fully closed position and initiates the closing phase. This pressure difference across

the valve has increased gradually as the valve closes and causes a noticeable motion of the leaflet at $t = 0.19\text{s}$ (see Figure 7.45(b)). And the pressure difference is more prominent towards the end of the closed position as observed in Figures 7.45(c) and (d). Finally the valve is sealed off with a large pressure difference at $t = 0.34\text{s}$ (see Figure 7.45(e)).

A series of velocity vector plots and streamline plots for the closing phase with corresponding valve configurations are presented in Figure 7.47. Velocity profiles for plane $y = 0$ are extracted at $X = 40\text{mm}$ (plane I), 50mm (plane II), 60mm (plane III), 70mm (plane IV) and 80mm (plane V) as shown in Figure 7.48. At $t = 0.14\text{s}$, as shown in Figure 7.47(a), the flow is quite smooth from the aorta to the left ventricle. The maximum velocities on planes I, II and V are -0.53 m/s , -0.52 m/s and -0.23 m/s as indicated in Figure 7.49(a). On plane III, the average velocity inside the sinus is about -0.01 m/s . It is much smaller than the velocity in the mainstream of -0.5 m/s . For plane IV just before the valve leaflet, the average velocity inside the sinus is about -0.02 m/s , and the average mainstream velocity is about -0.3 m/s . It can be seen that there is little flow in the sinus cavity at this moment. Both the velocity and the rate of change of the velocity in the mainstream is much larger than those in the sinus, thus resulting a larger fluid stress on the leaflet surface on the mainstream side than the other side, with the former being about 90 Pa and the latter about 20 Pa as found in Figure 7.46(a).

The motion of the valve leaflet is relatively small at the early stages of the closing phase due to the small pressure differences. The valve leaflet moves to a new position at $t = 0.19\text{s}$ under the influence of blood-leaflet interaction as shown in Figure 7.47(b). At this moment, the flow field is still smooth because the valve orifice is large for the

reverse flow. The maximum velocities on planes I, II and V are -0.68 m/s , -0.67 m/s and -0.24 m/s . On planes III and IV, the average velocities inside the sinus are 0.06 m/s and 0.08 m/s , while in the mainstream the average velocities are -0.5 m/s and -0.36 m/s . The maximum velocity on plane III is -0.72 m/s . This is expected because the presence of the leaflet limits the orifice area and increases the mainstream velocity. The larger difference of velocity in the mainstream results in a larger fluid stress of 60 Pa on the leaflet surface. While on the leaflet surface on the sinus side, the fluid stress is about 8 Pa as illustrated in Figure 7.46(b).

At $t = 0.27\text{s}$, the valve moves to a new position under the hemodynamic forces acting on it (Figure 7.47(c)). At this moment, the flow is obstructed a little by the closing valve leaflet. The maximum velocities on planes I, II and V are -0.5 m/s , -0.5 m/s and -0.11 m/s as depicted in Figure 7.49(c). The maximum velocities on planes I and II are larger than that on plane V, because the closing valve obstruct the blood flow and reduce the passage for flow after the valve leaflet, thus increasing the maximum velocity after the leaflet. On plane III, the maximum velocity in the mainstream is about -0.58 m/s due to the influence of the blood-leaflet interaction. While on plane IV, the maximum velocity is about -0.2 m/s . Inside the sinus cavity, the average velocity is 0.01 m/s , which is much lower than the mainstream velocity. The fluid stress on the leaflet surface facing the mainstream side is 150 Pa . And a high fluid stress of about 300 Pa is observed in the region near the free edge due to the small valve orifice area. On the surface facing the sinus side, the average fluid stress is about 30 Pa as shown in Figure 7.46(c).

The valve is about to close to at $t = 0.32\text{s}$ as observed in Figure 7.47(d). The blood flow is obstructed a lot by the valve leaflet. The maximum velocities on planes I, II

and V are -0.52 m/s, -0.38 m/s and -0.14 m/s as shown in Figure 7.49(d). These values increase because the valve orifice area is much smaller at this moment. On plane III, the maximum velocity in the mainstream is -0.57 m/s as the valve almost closes. On plane IV, the maximum velocity in the mainstream is -0.18 m/s as the inflow slows down. And in the sinus cavity, the average velocities on the two planes are near zero, which indicates that the flow inside the sinus cavity is almost stagnant at this moment. Thus the average fluid stress on the leaflet surface facing the sinus side is of a very low value of 5 Pa, while at the other side, the average fluid stress is about 150 Pa. The high stress region still exists near the free edge of the leaflet due the large velocity in the mainstream. The valve finally snaps to the fully closed position at $t = 0.34$ s and little flow is observed in the sinus and the mainstream at this moment as shown in Figure 7.47(e).

7.5.3 Remarks

The motions of a stented bioprosthetic heart valve for both opening phase and closing phase under physiological conditions and the blood flows through it are simulated by the developed parallel solver with the immersed membrane method for blood-leaflet interaction. And the results are compared with experimental and numerical results. Good agreements are observed between current numerical results and other researchers' results.

The general motions of the valve can be pictured as follows: the valve leaflet opens and closes very fast. For most of the time of one cardiac cycle the heart ejects the blood to the aorta when the valve is open, and it is filled with blood from left atrium when the valve is closed. Transvalvular pressure difference is the major cause for the motions of the valve leaflet. During the opening phase, the pressure on the left ventricle side is higher than the aorta side, and the positive pressure difference pushes the leaflet

towards its fully opened position. While during the closing phase, the pressure on the left ventricle side is lower than the pressure on the aorta side, and the adverse pressure difference pushes the leaflet towards its fully closed position. The opening phase starts from $t = 0.06s$, it reaches its maximum opening position around $t = 0.12s$. The valve leaflet remains at its maximum opening position and fluctuates until the end of opening phase around $t = 0.14s$. The central flow enters the sinus cavity at the leaflet free edge, curls back along the sinus wall, along the valve leaflet and then flows out into the mainstream. For both the opening and closing phases, the velocity inside the sinus cavity is much lower than the velocity in the mainstream, whose maximum value is about 10-15% of the velocity of the mainstream. The fluid stresses on the leaflet surfaces on the mainstream side are higher than those on the sinus side due to the high velocity in the mainstream. And high fluid stresses exist in the region near the free edge of the leaflet, which indicates that these areas are high-risk regions for leaflet tearing. At the moment of complete valve opening, the blood flow has started to decelerate already. The negative transvalvular pressure pushes the valve from its fully opening position towards its fully closed position. And the motion of the valve leaflet is relatively small in the early stages of the closing phase due to the small pressure differences. The valve leaflet fluctuate at its fully opened position for quite some time till $t = 0.19s$. At the end of the systolic phase, the valve snaps to the fully closed position.

Figure 7.1 Geometry of the 2D channel model with an immersed membrane.

Figure 7.2 Flow field of grid (I) with 2,000 nodes

Figure 7.3 Flow field of grid (II) with 18,000 nodes

Figure 7.4 Flow field of grid (III) with 32,000 nodes

(a)

(b)

Figure 7.5 (a) X-component velocity profiles at $X=1$. (b) Convergence history plot for three different mesh densities

Figure 7.6 Computational model for validation of steady flow ($L=20$ mm)

(a)

(b)

Figure 7.7 (a) Close-up view of immersed membrane and surrounding mesh (b) Close-up of internal boundary and surrounding mesh

Figure 7.8 Convergence history of immersed membrane and internal boundary.

(a)

(b)

Figure 7.9 (a) Flow field with an immersed membrane computed by the immersed membrane method (b) flow field with an internal boundary computed by solver DELTINKE.

(a) at $X = 0$ (b) at $X = 1$

Figure 7.10 (a) Comparison of X-component velocity profiles at $X=0$ (b) Comparison of X-component velocity profiles at $X=1$

(a) $t = 0.2$

(b) $t = 0.6$

(c) $t = 1.0$

(d) $t = 1.6$

Figure 7.11 Motion of a rigid membrane, thickness of $0.5L/100$, driven by a constant inflow at $Re=100$

Figure 7.12 Steady flow with an immersed flexible membrane

(b) $t = 0.4$

(c) $t = 0.6$

(d) $t = 1.0$

(e) $t = 1.6$

(f) $t = 2.0$

(g) $t = 2.6$

(h) $t = 3.0$

Figure 7.13 Motion of an immersed flexible membrane, driven by constant viscous inflow at $Re=100$.

(a)

(a)

Figure 7.14(a) Fully closed configuration of St. Jude Bi-leaflet MHV (b) Fully open configuration of St. Jude Bi-leaflet MHV.

Figure 7.15 Time-dependent inlet velocity profile for opening and closing phases
 (A: aortic valve closed, $0.0s < t \leq 0.06s$, B: aortic valve opening & closing, $0.06s < t \leq 0.35s$, C: aortic valve closed, $0.35s < t \leq 0.8s$) [23]

Figure 7.16(a) Geometry of the vessel for simulation of St. Jude 29mm Bi-leaflet MHV
(b) Geometry and position of the valve leaflet

(a) angle = 60° at $t=0.06s$

(b) angle = 50° at $t=0.093s$

(c) angle = 30° at $t=0.098s$

(d) angle = 22° at $t=0.100s$

(e) angle = 10° at $t=0.102s$

(f) angle = 5° at $t=0.105s$

Figure 7.17 Pressure contours of the x - y plane at $z=0$ during opening phase at different time instants

(a) angle = 60° at $t=0.06s$

(b) angle = 50° at $t=0.033s$

(c) angle = 30° at $t=0.038s$

(d) angle = 22° at $t=0.040s$

(e) angle = 10° at $t=0.042s$

(f) angle = 5° at $t=0.045s$ Figure 7.18 Velocity vectors of the x - y plane at $z=0$ and volume stream around the valve leaflet during opening phase at different time instants (a) angle = 60° at $t=0.06s$ (b) angle = 50° at $t=0.033s$

(c) angle = 30° at t=0.038s

(e) angle = 22° at t=0.040s

(e) angle = 10° at t=0.042s

(f) angle = 5° at $t=0.045s$

Figure 7.19 Pressure and fluid shear stress distributions on leaflet surface for opening phase. (1) pressure distribution on leaflet surface of the upstream side. (2) pressure distribution on leaflet surface of the downstream side. (1) fluid shear stress distribution on leaflet surface of the upstream side. (2) fluid shear stress distribution on leaflet surface of the downstream side

(a) $t=0.14s$ angle $=5^\circ$ (steady flow) (b) $t=0.251s$ angle $=11^\circ$ (c) $t=0.264s$ angle $=18^\circ$ (d) $t=0.301s$ angle $=34^\circ$

(e) $t=0.308$ angle= 43°

(f) $t=0.319$ s angle= 56°

(g) $t=0.325$ s angle= 60°

Figure 7.20 Pressure contours of the x-y plane at $z=0$ around valve leaflets at different time instants of closing phase

(a) $t=0.14$ s angle= 5° (steady flow)

(b) $t=0.251s$ angle= 11°

(c) $t=0.264s$ angle= 18°

(g) $t=0.301s$ angle= 34°

(e) $t=0.308$ angle= 43°

(f) $t=0.319s$ angle= 56°

(g) $t=0.325s$ angle= 60° Figure 7.21 Velocity vector of the x-y plane at $z=0$ and volume stream around the valve leaflet during closing phase at different time instants (a) $t=0.14s$ angle= 5° (steady flow) (b) $t=0.251s$ angle= 11°

(1)

(2)

(3)

(4)

(c) $t=0.264s$ $\text{angle}=18^\circ$

(1)

(2)

(3)

(4)

(d) $t=0.301s$ $\text{angle}=34^\circ$

(1)

(2)

(3)

(4)

(e) $t=0.308s$ $\text{angle}=43^\circ$

Figure 7.22 Pressure and fluid shear stress distributions on leaflet surface for closing phase. (1) pressure distribution on leaflet surface of the upstream side. (2) pressure distribution on leaflet surface of the downstream side. (1) fluid shear stress distribution on leaflet surface of the upstream side. (2) fluid shear stress distribution on leaflet surface of the downstream side

Figure 7.23 Computational domain for oscillating sphere

Figure 7.24 Grid generation for consecutively finer meshes

(a) Grid of 11^3 nodes

(b) Grid of 21^3 nodes

(c) Grid of 41^3 nodes

(d) Grid of 81^3 nodes

Figure 7.25 Mesh partition for grid convergence study

(a) Grid of 11^3 nodes

(b) Grid of 21^3 nodes

(c) Grid of 41^3 nodes

(d) Grid of 81^3 nodes

Figure 7.26 Flow fields at the moment of the end of one oscillation time period (Pictures on the left are plots of streamline and x-component velocity contour for $y = 0$ planes. Pictures on the right are vector plots for $y=0$ planes.)

Figure 7.27 Convergence history of L_1 , L_2 and L_∞ error norms of x-component velocity for the three coarser grids with respect to the finest grid

Figure 7.28 Computational domain for rotating sphere

Figure 7.29 Mesh partition for steady flows induced by a rotating sphere

(a) $t = 0.1T$

(b) $t = 0.5T$

(c) $t = 1.0T$

(d) $t = 2.0T$ (e) $t = 3.0T$ Figure 7.30 Evolution history for flow field due to a rotating sphere at $Re = 100$

Figure 7.31 Unsteady results reported by Gilmonov [157] for flow due to a rotating sphere at $Re = 100$

(a) $Re = 10$

(b) $Re = 20$

(c) $Re = 50$

(d) $Re = 100$

Figure 7.32 Steady flow due to rotating sphere at different Reynolds numbers

(a) $Re = 20$

(b) $Re = 50$

(c) $Re = 100$

Figure 7.33 Numerical results by Gilmonov [157] of steady flow due to rotating sphere at different Reynolds numbers

(a) Perspective view of the computational model

(a) Geometry of the computational model

Figure 7.34 Computational domain for opening phase

Figure 7.35 Configuration of a closed and unloaded porcine aortic valve leaflet

Figure 7.36 Mesh for a bioprosthetic aortic valve leaflet

(a)

(b)

Figure 7.37 (a) Fluid mesh, immersed leaflet mesh for opening phase (b) Fluid mesh extracted from the center of y – axis ($x - z$ plane) with the immersed leaflet mesh

(a) $t = 0.06$ s

(b) $t = 0.07$ s

(c) $t = 0.08$ s (d) $t = 0.09$ s

(e) $t = 0.10s$ (f) $t = 0.12s$

(g) $t = 0.14s$

Figure 7.38 Fluid pressure and stress distributions on a valve leaflet at different time instants for opening phase. (i) Pressure distribution on the LV (left ventricle) side; (ii) Pressure distribution on the aorta side; (iii) Fluid stress distribution on the LV side; (iv) Fluid stress distribution on the aorta side.

(a) $t = 0.06$ s

(b) $t = 0.07$ s

(c) $t = 0.08$ s

(d) $t = 0.09$ s

(e) $t = 0.12$ s

(f) $t = 0.14$ s

(g) $t = 0.14 \text{ s}$

Figure 7.39 Pressure contours of $y = 0$ plane at different time instants for opening phase

(a) $t = 0.06 \text{ s}$

(b) $t = 0.07s$

(c) $t = 0.08s$

(d) $t = 0.09s$

(e) $t = 0.10s$

(f) $t = 0.12$ s

(g) $t = 0.14s$

Figure 7.40 Velocity vector plots at different time instants for opening phase. (i) Velocity vector plot for $y = 0$ plane with valve leaflet. (ii) Velocity vector plot of the flow field near the valve leaflet. (iii) Velocity vector plots extracted at $x = 50\text{mm}$ (Plane I), $x = 60\text{mm}$ (Plane II), $x = 70\text{mm}$ (Plane III), $x = 80\text{mm}$ (Plane IV) and $x = 90\text{mm}$ (Plane V). (iv) Streamline distribution of the flow field around the valve leaflet.

Figure 7.41 Valve orifice area of opening phase. (View from aorta side to left ventricle)

Figure 7.42 Extracted positions of velocity profiles

(a) $t = 0.07s$

(b) $t = 0.08s$ (c) $t = 0.09s$ (d) $t = 0.10s$

(e) $t = 0.12s$ (f) $t = 0.14s$

Figure 7.43 Velocity profiles extracted for opening phase

Figure 7.44 Perspective view of the computational model for closing phase

(a) $t = 0.14$ s

(b) $t = 0.19$ s

(c) $t = 0.27$ s

(d) $t = 0.32$ s (e) $t = 0.34$ sFigure 7.45 Pressure contours of $y = 0$ plane at different time instants for closing phase

(a) $t = 0.14s$

(b) $t = 0.19s$

(c) $t = 0.27s$ (d) $t = 0.32s$

Figure 7.46 Fluid pressure and stress distribution on valve leaflet at different time instants for opening phase. (i) Pressure distribution on the LV (left ventricle) side; (ii) Pressure distribution on the aorta side; (iii) Fluid stress distribution on the LV side; (iv) Fluid stress distribution on the aorta side.

(a) $t = 0.14s$

(b) $t = 0.19s$

(c) $t = 0.27s$

(d) $t = 0.32s$

(e) $t = 0.34s$

Figure 7.47 Velocity vector plots at different time instants for opening phase. (i) Velocity vector plot for $y = 0$ plane with valve leaflet. (ii) Velocity vector plot of the flow field near the valve leaflet. (iii) Velocity vector plots extracted at $x = 40\text{mm}$ (Plane I), $x = 50\text{mm}$ (Plane II), $x = 60\text{mm}$ (Plane III), $x = 70\text{mm}$ (Plane IV) and $x = 80\text{mm}$ (Plane V). (iv) Streamline distribution of the flow field around the valve leaflet.

Figure 7.48 Positions for velocity extraction of plane $y = 0$

(a) $t = 0.14s$

(b) $t = 0.19s$ (c) $t = 0.27s$ (d) $t = 0.31s$

Figure 7.49 Velocity profiles extracted for opening phase

Figure 7.50 Valve orifice area for closing phase. (View from aorta side to left ventricle)

(i) Full view of the flow field

(ii) One third of the flow field

Figure 7.51 Numerical results excerpted from De Hart's thesis [3] for comparison

CHAPTER EIGHT

CONCLUSIONS AND FUTURE WORK

This chapter concludes the thesis with a summary of the achievements and contributions of the present research work, which aims to develop efficient and accurate numerical methods for moving boundary and fluid-structure interaction problems in biofluid dynamic applications. This chapter will summarize the works for two main parts: one is the development of the three-dimensional unsteady parallel Navier-Stokes incompressible flow solver with immersed membrane method for fluid-structure interaction phenomena, and the other is its application to compute the motions of a stented bioprosthetic aortic heart valve and the blood flows through it. Finally this chapter proposes the possible directions for future researches.

8.1 CONCLUSIONS

In this work, a three-dimensional immersed membrane method (IMM) is developed and applied to simulate the fluid-structure interaction in prosthetic aortic heart valves. This method is based on the imposition of the velocity continuity, by introducing multiple ghost nodes and one-sided extrapolations for ghost variables, which represent the no-slip boundary conditions along the fluid-structure interface. The implementation of this numerical technique is carried out in the framework of finite volume scheme encoded in an unstructured parallel Navier-Stokes solver for 3D unsteady incompressible flows. The main features of this numerical technique are avoidance of mesh update and allowance of capturing ‘jump’ conditions across the immersed structure, which result in the fact that the unstructured fluid mesh is not interrupted in any way by the presence of the immersed structure and more accurate cross-structure fluid conditions can be captured without any smoothing. And compared with traditional methods, this scheme does not need to perform structure-surface-normal extrapolations and interpolations which are difficult for 3D unstructured meshes. The solver is applied to analyze motions of a stented aortic heart valve under physiological conditions. The results show that the current numerical method can efficiently simulate the fluid-structure interaction problems with large deformation. The main achievements are summarized in the following sections.

8.1.1 Numerical Methods

The 3D parallel solver used in this work is developed from the finite volume scheme based on unstructured tetrahedral meshes for incompressible flows. Artificial

compressibility (ACM) is introduced to transform the elliptic governing Navier-Stokes equations into a set of hyperbolic equations [22]. The unsteady flow is calculated with a matrix-free implicit dual time stepping scheme. A five-stage Runge-Kutta time integration algorithm is used between each physical time step to iterate the numerical solution in pseudo time until convergence is reached [45].

The cell-centered finite volume scheme is adopted for the computation of the structural domain and triangular shell elements are used to discretized the thin-walled structures. Control volumes are constructed over the unstructured triangular mesh, and structural computation is carried out based on the control volumes. The finite volume method based on triangular shell elements has been successfully applied to model the behavior of a bioprosthetic heart valve under physiological conditions, which will be discussed in the next section.

The goal to develop a highly parallel solver so as to accelerate the convergence rate, and at the same time, to reduce the time for simulating fluid-structure interaction problems with complex geometries and avoid very large single processor memory requirements in sequential processing are successfully achieved in this work. The serial solver is parallized by using the Single Program Multiple Data (SPMD) programming paradigm with Domain Decomposition (DD) technique [45] and Message Passing Interface (MPI) [98]. The implementation of parallel computing improves the computational efficiency greatly, and the parallel solutions obtained remain the same as those obtained by the serial solvers.

An efficient immersed membrane method is developed and implemented in this work to tackle the fluid-structure interaction and moving boundary problems. In the IMM, the fluid and structural domains are discretized independently. The stationary Eulerian mesh is used for the fluid domain, whilst the immersed structure moves on top of the background Eulerian mesh, without changing the topology and geometry of the mesh. For those fluid nodes nearest to the immersed structure, the concept of ghost nodes is introduced to adopt the influence of the presence of the structure. The influence of the structure on ambient fluids is conveyed by enforcing the velocity continuity condition along the fluid-structure interface. And the enforcement of velocity continuity is implemented by the one-sided linear extrapolations of ghost variables for ghost nodes. With the ghost fluid nodes, the conventional mesh update strategies, such as remeshing or Arbitrary Lagrange-Euler (ALE) techniques, are consequently superfluous. Hence, the computational cost for mesh treatment is greatly reduced. By using the one-sided extrapolated ghost fluid variables, sharp fluid-structure interfaces can be maintained and “jump” fluid conditions across the immersed structure can be captured without smoothing, thus more accurate fluid solution can be achieved. The ghost nodes and ghost variables are treated as boundary conditions for the computation of the fluid domain. After the computation of the fluid domain, fluid forces are calculated on the structure surfaces, and the structure is updated by solving the Newton’s Second Law of Motion under the influence of the fluid forces. The fluid-structure interaction problem is solved in an iterative manner as above for each time step. This method is unique in two main factors compared with existing methods using ghost nodes: (i) Extrapolation of ghost values along fluid mesh edges instead of surface-normal directions of the structural domain. The surface-normals of the structural domain are difficult to find, besides you have to find out the intersection points between the surface-normals and the

3D unstructured fluid meshes; (ii) Multiple ghost nodes are used for the same fluid node. The use of a particular value depends on the edge being considered. The implementation of multiple ghost nodes increases the accuracy of the method.

This method has been successfully applied to various 2D and 3D fluid-structure interaction problems (see Section 7.1). For 2D cases, the motions of a rigid membrane and a flexible membrane are modeled by the IMM under the influence of 2D incompressible flows, which demonstrate the ability of the method to compute fluid-structure interaction problems with large deformation. A 29mm bi-leaflet mechanical heart valve is also modeled by the IMM under physiological conditions (see Section 7.2). The results agree well with available numerical and experimental results. The IMM is then extended to model immersed structures with arbitrary shapes as demonstrated in the simulations of flows due to an oscillating sphere and the flows induced by a rotating sphere (see Section 7.3 and Section 7.4). The results are in good agreements with analytical solutions and available numerical results. Analysis on the results of grid convergence study shows that the IMM is a second order accurate method. At last the IMM is applied to model a stented aortic heart valve under physiological conditions, which will be detailed in the next section.

8.1.2 Simulation of A Stented Bioprosthetic Heart Valve

The motions of a stented bioprosthetic aortic heart valve are modeled by the developed parallel solver using IMM. The simulations are performed under physiological conditions and blood-leaflet interaction is calculated for both the opening and closing phases. In this numerical simulation, attention is not focused on the determination of an

optimal artificial valve, rather on the investigation of the flow fields around the BHV and the response of the valve leaflets. And the study is also focused on the complex features and the fluid dynamics involving the moving leaflets due to the blood-leaflet interaction. The key features of this numerical model are three-dimensionality, fully coupled blood-leaflet interaction, physiologically realistic flow conditions, an Eulerian mesh strategy with the novel IMM and parallel computation.

During a cardiac cycle, one may picture the valve in action as follows: a sudden opening of the valve, then followed by only a little movement while the valve remains open, a sudden closure of the valve, and then almost no movement while the valve remains closed. The simulated pulsatile flow showed a very complicated, inherently three-dimensional flow behavior. The motions of the valve leaflets for both opening and closing phases are driven by the blood-leaflet interaction. It is found that the pressure force acting on the surface of the leaflets is the main driving force for leaflet motions for both the opening and closing phases. The fluid stress is much smaller than the fluid pressure. The opening phase starts from $t = 0.06s$ and reaches its maximum opening position around $t = 0.12s$. The valve leaflet remains at its maximum opening position and fluctuates until the end of opening phase around $t = 0.14s$. For both the opening and closing phases, the velocity inside the sinus cavity is much lower than the velocity in the main stream, whose maximum value is about 10-15% of the maximum velocity of the main stream. Therefore the fluid stresses on the leaflet surfaces on the mainstream side are higher than those on the sinus side. And high fluid stresses exist in the region near the free edge of the leaflet, which indicates that these areas are high-risk regions for leaflet tearing. At the moment of complete valve opening, the blood flow has started to decelerate already. The negative transvalular pressure pushes the valve from the fully

open position towards its fully closed position. And the motion of the valve leaflet is relatively small at the early stages of the closing phase due to the small pressure differences. The valve leaflet fluctuate at its fully opened position for quite some time till $t = 0.19s$. At the end of the systolic phase, the valve snaps to the fully closed position.

In summary, the current solver together with the proposed immersed membrane method is very efficient and robust in solving moving flexible body problems with fluid-structure interaction in 3D unsteady flows. In this work, it serves as a first step in an effort to develop a high performance computational tool that can be used in the rational design and test of new BHVs. In this research work, the numerical models are found to give a reasonably accurate representation of the dominant flow patterns of stented bioprosthetic heart valves. And the solver is found to be an effective computational tool to analyze and test different designs of prosthetic heart valves under physiological conditions and blood-leaflet interaction.

8.2 FUTURE WORKS

In the present work, a high performance computational tool based on the unsteady parallel-unstructured Navier-Stokes incompressible solver with the immersed membrane method for fluid-structure interaction problems has been developed to numerically simulate flow past a stented BHV. However, several problems regarding the developments of numerical methods and algorithms exist that deserve future attentions listed as follows:

- (1) Numerical extrapolation schemes with higher order accuracy can be explored to improve the overall accuracy of the currently second order accurate immersed membrane method.
- (2) A full three dimensional numerical model based on finite volume scheme for the structural dynamics are required to simulate the BHV with real geometries in order to obtain more accurate results. And the finite volume scheme based numerical model for the structural domain is more compatible with the current fluid solver developed within the framework of the FV method. This could result in a solver with a single FV scheme for both the structure and fluid domains.
- (3) The problem of laminar-to-turbulence transition at certain locations and time instants is quite an interesting phenomenon. Large Eddy Simulation (LES) can be employed as a versatile simulation tool so that all flow conditions, including laminar, turbulent flows and flows with laminar-to-turbulence transition, can be modeled in the calculation. This will greatly improve the accuracy for the simulation of the whole BHV system.
- (4) To compute the opening and closing phases together in a single cardiac cycle so that the historical data of the opening phase is available for the closing phase. In this way it will greatly enhance the accuracy of the closing phase results.
- (5) The effects of sinus cavity on the closing behavior of the aortic valve need further investigation. Therefore sinus cavities with different sizes could be used in the numerical models and comparisons of the results are desirable.

- (6) Flexible walls rather than rigid walls used in current work are more realistic representations of the ascending aorta walls where the valve situated. The flexible vessels themselves are complex moving boundaries problems, combined with the blood-leaflet interaction of the heart valves, which will greatly increase the burden on computing resources. But these simulations can provide more accurate and realistic results for the investigation of BHVs.
- (7) Fiber-reinforced BHVs are interesting developments for current BHV designs. Future works may be focused on the study the behavior of such BHVs under similiar physiological conditions.

REFERENCE

- [1] Thubrika M., The Aortic Valve, Virginia, CRC Press Inc, 1990.
- [2] The Heart Valve Lab, <http://www.heartlab.robarts.ca>, University of Western Ontario.
- [3] De Hart J., Fluid-Structure Interaction in the Aortic Heart Valve: A Three-Dimensional Computational Analysis, Ph.D. Thesis, Eindhoven University of Technology, 2003.
- [4] Fung Y. C., Biodynamics – Circulation, Springer – Verlag, New York, 1984.
- [5] Van Steenhoven A. A., et al., An In-vivo Cinematographic Analysis of the Behaviour of the Aortic Valve. *American Journal of Physiology*, **240**(2): H286-H292, 1981.
- [6] Nerem R. M., et al., An *in vivo* Study of Aortic Flow Disturbances, *Cardiovascular Research*, **6**(1): 1-14, 1972.
- [7] De Hart J., et al., A Three-Dimensional Analysis of A Fiber-Reinforced Aortic Valve Prosthesis, *Journal of Biomechanics*, **31**(7): 629-638, 1998.
- [8] Cacciola G., et al., A Three-Dimensional Mechanical Analysis of A Stentless Fiber-Reinforced Aortic Valve Prosthesis, *Journal of Biomechanics*, **33**(5): 521-530, 2000.
- [9] Hoerstrup S. P., et al., Functional Living Trileaflet Heart Valves Grown *in vitro*, *Circulation*, **102**(19): 44-49, 2000.
- [10] Sodian R., et al., Early *in vivo* Experience with Tissue-Engineered Trileaflet Heart Valves, *Circulation*, **102**(19): 22-29, 2000.

- [11] Jiang Y., Investigation of Mechanical Heart Valve under Physiological Condition, Ph.D. Thesis, Nanyang Technological University of Singapore, 1997.
- [12] Gresho P. M., Some Current CFD Issues Relevant to the Incompressible Navier-Stokes Equations, *Computer Methods in Applied Mechanics and Engineering*, **87**:201-252, 1991.
- [13] Justesen P., A Numerical Study of Oscillating Flow Around a Circular Cylinder, *Journal of Fluid Mechanics*, **222**, pp.157, 1991.
- [14] Osswald G. A., Ghia K. N., and Ghia U., A Direct Algorithm for Solution of Incompressible Three-Dimensional Unsteady Navier–Stokes Equations, *AIAA Paper* 87-1139-CP, 1987.
- [15] Tokunaga H., Yoyeda K. and Satofuka N., Direct Simulations of Three-Dimensional Flows Using Generalized Vector Potential Method, *AIAA Paper* 91-1610-CP, 1991
- [16] Rubel A., and Volpe G., Biharmonic Vector Stream Function Formulation and Multigrid Solution for a Three-Dimensional Driven-Cavity Stokes Flow, *AIAA Paper* 89-1968-CP, 1989.
- [17] Zhao Y., and Tai C. H., Higher-Order Characteristics-Based Methods for Incompressible Flow Computation on Unstructured Grids, *AIAA Journal*, **39**(7): 1280-1287, 2001.
- [18] Chorin A. J., A Numerical Method for Solving Incompressible Viscous Flow Problems, *Journal of Computational Physics*, **2**(1): 12 –26, 1967.
- [19] Tai C. H., and Zhao Y., Parallel Unsteady Incompressible Viscous Flow Simulation Using an Unstructured Multigrid Method, *Journal of Computational Physics*, **192**(1): 277-311, 2003.

- [20] Nithiarasu P., An Efficient Artificial Compressibility (AC) Scheme Based on the Characteristic Based Split (CBS) Method for Incompressible Flows, *International Journal for Numerical Methods in Engineering*, **56**,1815-1845, 2003.
- [21] Harlow F. H., and Welch J. E., Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface, *Physics of Fluids*, **8**(12): 2182, 1965.
- [22] Lin P. T., Implicit Time Dependent Calculations for Compressible and Incompressible Flows on Unstructured Meshes, MSc Thesis, Department of Mechanical and Aerospace Engineering, Princeton University, 1994.
- [23] Jameson A., Time Dependent Calculations Using Multigrid, with Applications to Unsteady flows Past Airfoils and Wings, *AIAA Paper* 91-1596, June 1991.
- [24] Pozrikidis C., Introduction to Theoretical and Computational Fluid Dynamics, Oxford University Press, 1997.
- [25] Tai C. H., Zhao Y., and Liew K. M., Parallel-Multigrid Computation of Unsteady Incompressible Viscous Flows Using a Matrix-Free Implicit Method and High-Resolution Characteristics-Based Scheme, *Computer Methods in Applied Mechanics and Engineering*, 2004 (In Press).
- [26] Peskin C. S., Flow Patterns Around Heart Valves: A Digital Computer Method for Solving the Equation of Motion, Ph.D. Thesis, Albert Einstein College of Medicine, Yeshiva University, 1972.
- [27] Fogelson A. L., A Mathematical Model and Numerical Method for Studying Platelet Adhesion and Aggregation During Blood Clotting, *Journal of Computational Physics*, **56**: 111-134, 1984.
- [28] Beyer R. P., A Computational Model of the Cochlea Using the Immersed Boundary Method, *Journal of Computational Physics*, **98**: 145-162, 1992.

- [29] Arthurs K. M., Moore L. C., Peskin S. C., Pitman E. B., and Layton H. E., Modeling Arteriolar Flow and Mass Transport Using the Immersed Boundary Method, *Journal of Computational Physics*, **147**: 402-440, 1998.
- [30] Stockie J. M. and Green S. I., Simulating the Motion of Flexible Pulp Fibres Using the Immersed Boundary Method, *Journal of Computational Physics*, **147**: 147-165, 1998.
- [31] Roma A. M., Peskin S. C., and Berger M. J., An Adaptive Version of the Immersed Boundary Method, *Journal of Computational Physics*, **153**, pp.509-534, 1999.
- [32] Lai M.-C., and Peskin S. C., An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity, *Journal of Computational Physics*, **160**: 705-719, 2000.
- [33] Underwood F. N. Jr, A Numerical Study of the Steady, Axisymmetric Flow Through a Disk Type Prosthetic Heart Valve, Ph.D. Thesis, Graduate School of University of Notre Dame, 1975.
- [34] Huang Z. J., Merkle C. L., Abdallah S., and Tabell J. M., Numerical Simulation of Unsteady Laminar Flow through a Tilting Disk Heart Valve: Prediction of Vortex Shedding, *Journal of Biomechanics*, **27**(4), pp.391-402, 1994.
- [35] Shim E. B., and Chang K. S., Numerical Analysis of Three-Dimensional Björk-Shiley Valvular Flow in an Aorta, *Journal of Biomechanical Engineering*, **119**, pp.45-51, 1997.
- [36] King M. J., David T., and Fisher J., Three-Dimensional Study of the Effect of Two Leaflet Opening Angles on the Time-Dependent Flow through a Bi-leaflet Mechanical Heart Valve, *Medical Engineering Physics*, **19**(3): 235-241, 1997.

- [37] Leefe S. E., and Gentle C. R., A Review of the In-Vitro Evaluation of Conduit-Mounted Cardiac Valve Prostheses, *Medical Engineering Physics*, **17**(7), pp.497-506, 1995.
- [38] Yoganathan A. P., Ellis J. T., Healy T. M., and Chatzimavroudis G. P., Fluid Dynamics Studies for the Year 2000, *Journal of Heart Valve Disease*, **7**, pp.133-139, 1998.
- [39] Lim W. L., Chew Y. T., Chew T. C., and Low H. T., Steady Flow Dynamics of Prosthetic Aortic Heart Valves: A Comparative Evaluation with PIV Techniques, *Journal of Biomechanics*, **31**, pp.411-421, 1998.
- [40] Bluestein D., Li Y. M., and Krukenkamp I. B., Free Emboli Formation in the Wake of Bi-leaflet Mechanical Heart Valves and the Effects of Implantation Techniques, *Journal of Biomechanics*, **35**, pp.1533-1540, 2002.
- [41] Chen D., and Zhang J. H., Transient Analysis of Artificial Mechanical Valve-Blood Interaction, *Applied Mathematics and Mechanics*, **16**(5): 437-441, 1995.
- [42] Chen D., and Zhang J. H., A Comparative Study of the Opening and Closing Process of Two Types of Mechanical Heart Valves using ALE Finite Element Method, *Applied Mathematics and Mechanics*, **17**(4): pp.299-307, 1996.
- [43] Zhang J. H., Chen D., and Zou S. Q., ALE Finite Element Analysis of the Opening and Closing Process of the Artificial Mechanical Valve, *Applied Mathematics and Mechanics*, **17**(5): 403-411, 1996.
- [44] Shi Y. B., Numerical and Experimental Study of Pulsatile Flow in Bi-leaflet Mechanical Heart Valves, Ph.D. Thesis, Nanyang Technological University, 2002.

- [45] Tai, C.H., Parallel-Multigrid Computation of Prosthetic Heart Valves under Physiological Conditions using an Immersed Object Method with Overlapping Grids, Ph.D. Thesis, Nanyang Technological University, 2005.
- [46] CFD Research Corp. (CFDRC) – Engineering Simulations Software, Services and R&D, <http://www.cfdrc.com>.
- [47] Makhijani Vinod B., Yang H. Q., Sighal Ashok K., and Hwang Ned H. C., An Experimental-Computational Analysis of MHV Cavitation: Effects of Leaflet Squeezing and Rebound, *Journal of Heart Valve Disease*, **3**(suppl. I): S35-S48, 1994.
- [48] Siegel John M. Jr, Makhijani Vinod B., Sinhgal Ashok K., Experimental-Numerical Assessment of the Effect of Squeeze Flow on Mechanical Heart Valve Fluid Dynamics and Cavitation Potential, BED Vol. 35, Bioengineering Conference ASME, 1997.
- [49] Makhijani Vinod B., Siegel John M. Jr, Hwang Ned H.C., Numerical Study of Squeeze-Flow in Tilting Disc Mechanical Heart Valves, *Journal of Heart valve Disease*, **5**: 97-103, 1996.
- [50] Makhijani Vinod B., Yang H.Q., Dionne Paul J., Thubrikar Mano J., Three Dimensional Coupled Fluid-Structure Simulation of Pericardial Bio-prosthetic Aortic Valve Function, *ASAIO Journal*, **48**: M387-M392, 1997.
- [51] Gao Zhi B., Hosein Nadeen, Dai Frank F., Hwang Ned H.C., Pressure and Flow Fields in the Hinge Region of Bileaflet Mechanical Heart Valves, *Journal of Heart Valve Disease*, **8**: 197-205, 1999.
- [52] Yun Xueyan, Numerical Simulation of Coupled Fluid and Solid Motions in a Leaflet Type Artificial Heart Valve, Ph.D. Thesis, University of Miami, 1999.

- [53] Gross J. M., Shu Mark C. S., Dai Frank F., Ellis Jeffery, Yoganathan Ajit P., Micro-Structural Flow Analysis within a Bileaflet Mechanical Heart Valve Hinge, *Journal of Heart Valve Disease*, **5**(6): 581-590, 1996.
- [54] Kiris C., Kwak D., Rogers S., and Chang I-D., Computational Approach for Probing the Flow Through Artificial Heart Devices, *Journal of Biomechanical Engineering*, **119**, pp.452-460, 1997.
- [55] Peskin S. C., and McQueen D. M., Modeling Prosthetic Heart Valves for Numerical Analysis of Blood Flow in the Heart, *Journal of Computational Physics*, **37**: 113-132, 1980.
- [56] Peskin S. C., and McQueen D. M., A Three-dimensional Computational Method for the Blood Flow in the Heart, *Journal of Computational Physics*, **81**: 372-405, 1989.
- [57] McCracken M. F., and Peskin S. C., A Vortex Method for Blood Flow Through Heart Valves, *Journal of Computational Physics*, **35**: 183-205, 1980.
- [58] Glowinski R. et al., A Fictitious Domain for Dirichlet Problems and Applications, *Computer Methods in Applied Mechanics and Engineering*, **111**: 283-303, 1994.
- [59] Glowinski R. et al., A Fictitious Domain for External Incompressible Viscous Flow Modeled by Navier-Stokes Equations, *Computer Methods in Applied Mechanics and Engineering*, **112**: 133-148, 1994.
- [60] Baaijens, F.P.T., A Fictitious Domain/Mortar Element Method for Fluid-Structure Interaction, *International Journal for Numerical Methods in Fluids* **35**(7): 743-761, 2001.
- [61] Majumdar S., Iaccarino G., Durbin P.A., RANS Solver with Adaptive Structured Boundary Non-Conforming Grids, *Annu. Res. Briefs 2001, Cent. Turbul. Res.* 353-364, 2001.

- [62] Ghias R., Mittal R., Lund T.S., A Non-Body Conformal Grid Method for Simulation of Compressible Flows with Complex Immersed Bondaries. *AIAA Paper*: 2004-0800.
- [63] Mittal R., Seshadri V., Udaykumar H.S., Flutter, Tumble and Vortex Induced Autorotation. *Theoretical Computational Fluid Dynamics*, **17**(3): 165-70.
- [64] Iaccarino G., Kalizin G., Elkins C.J., Numerical and Experimental Investigation of the Turbulent Flow in Ribbed Serpentine Passage, *Annu. Res. Briefs 2001, Cent. Turbul. Res.* 379-388, 2003.
- [65] Kalitzin G., Iaccarino G., Khalighi B., Towards An Immersed Boundary Solver for RANS Simulations. *AIAA Paper*: 2003-0770. Fung Y. C., *Biodynamics – Circulation*, Springer – Verlag, New York, 1984.
- [66] Fedkiw R., et al., A Non-Oscillatory Eulerian Approach to Interfaces in Multimaterial Flows (the Ghost Fluid Method). *Journal of Computational Physics* **152**, 457-492, 1999.
- [67] Donea J., Giuliani S., and Halleux J. P., An Arbitrary Lagrangian-Eulerian Finite Element Method for Transient Dynamic Fluid-Structure Interactions, *Computer Methods in Applied Mechanics and Engineering*, **33**, pp.689-723, 1982.
- [68] Margolin L. G., Introduction to An Arbitrary Lagrangian–Eulerian Computing Method for All Flow Speeds, *Journal of Computational Physics*, **135**, pp.198-202, 1997.
- [69] Hirt C. W., Amsden A. A., and Cook J. L., An Arbitrary Lagrangian–Eulerian Computing Method for All Flow Speeds, *Journal of Computational Physics*, **135**, pp.203-216, 1997.

- [70] Brackbill J U., and Pracht W. E., An Implicit, Almost-Lagrangian Algorithm for Magnetohydrodynamics, *Journal of Computational Physics*, **13**, pp.455, 1973.
- [71] Benson D. J., Computational Methods in Lagrangian and Eulerian Hydrocodes, *Computer Methods in Applied Mechanics and Engineering*, **99**, pp.235-394, 1992.
- [72] Farhat, C. et al., Load and Motion Transfer Algorithms for Fluid/Structure Interaction Problems with Non-matching Discrete Interfaces: Momentum and Energy Conservation, Optimal Discretization and Application to Aeroelastics, *Computer Methods in Applied Mechanics and Engineering* **157**, 95-114, 1998.
- [73] Horsten J. B. A. M., On the Analysis of Moving Heart Vales: A Numerical Fluid-Structure Interaction Model, Ph.D. Thesis, Eindhoven University of Technology, 1990.
- [74] Johnson A. A., and Tezduyar T. E., Parallel Computation of Incompressible Flows with Complex Geometries, *International Journal for Numerical Methods in Fluids*, **24**, pp.1321-1340, 1997.
- [75] Johnson A. A., and Tezduyar T. E., Advanced Mesh Generation and Update Methods for 3D Flow Simulations, Supercomputer Institute Research Report, University of Minnesota, 1998.
- [76] Kalro V., Aliabadi S., Garrard W., Tezduyar T. E., Mittal S., and Stein K., Parallel Finite Element Simulation of Large Ram-Air Parachutes, *International Journal for Numerical Methods in Fluids*, **24**, pp.1353-1369, 1997.
- [77] Mittal S., and Tezduyar T. E., Parallel Finite Element Simulation of 3D Incompressible Flows: Fluid-Structure Interaction, *International Journal for Numerical Methods in Fluids*, **21**, pp.933-953, 1995.

- [78] Hassan O., Probert E. J. and Morgan K., Unstructured Mesh Procedures for the Simulation of Three Dimensional Transient Compressible Inviscid Flows with Moving Boundary Components, *International Journal for Numerical Methods in Fluids*, **27**, 41-55, 1998.
- [79] Heil M., Stokes Flow in an Elastic Tube – A Large Displacement Fluid-Structure Interaction Problem, *International Journal for Numerical Methods in Fluids*, **28**, 243-265, 1998.
- [80] Farhat C., Michel L., and Nathan M., Mixed Explicit/Implicit Time Integration of Coupled Aero-elastic Problems: Three-field Formulation, Geometric Conservation and Distributed Solution, *International Journal for Numerical Methods in Fluids*, **21**, 807-835, 1995.
- [81] Schulz K. W., and Kallinderis Y., Unsteady Flow Structure Interaction for Incompressible Flows Using Deformable Hybrid Grids, *Journal of Computational Physics*, **143**, 569-597, 1998.
- [82] Farnell D. J. J., David T., and Barton D. C., Coupled States of Flapping Flags, *Journal of Fluids and Structures*, **19**, pp.29-36, 2004.
- [83] De Hart J., Peters G.W.M., Schreurs P.J.G., Baaijens F.P.T., A Two-Dimensional Fluid-Structure Interaction Model of the Aortic Valve, *Journal of Biomechanics*, **33**, pp.1079-1088, 2000.
- [84] Bluestein D., and Einav S., Techniques in the Analysis of Stability of Pulsatile Flow through Heart Valves, In: Leondes, C.T. (Ed.), *Biomechanics Systems Techniques and Applications: Cardiovascular Techniques*, Vol. II. CRC Press LLC, Boca Raton, FL, pp.4-1-4-39, 2001.
- [85] De Hart J., Peters G.W.M., Schreurs P.J.G., Baaijens F.P.T., A Three-Dimensional Computational Analysis of Fluid-Structure Interaction in the Aortic Valve, *Journal of Biomechanics*, **36**, pp.103-112, 2003.

- [86] Johnson A. A., and Tezduyar T. E., Three-dimensional of Fluid-Particle Interactions with the Number of Particles Reaching 100, *Computer Methods in Applied Mechanics and Engineering*, **145**, pp.301-321, 1997.
- [87] Wren G. P., Ray S. E., Aliabadi S. K., and Tezduyar T. E., Space-Time Finite Element Computation of Compressible Flows between Moving Components, *International Journal for Numerical Methods in Fluids*, **21**, pp.981-991, 1995.
- [88] Wren G. P., Ray S. E., Aliabadi S. K., and Tezduyar T. E., Simulation of Flow Problems with Moving Mechanical Components, Fluid-structure Interactions and Two-fluid Interfaces, *International Journal for Numerical Methods in Fluids*, **24**, pp1433-1448, 1997.
- [89] Löhner R., Chi Y., Cebal J. R., Baum J. D., Luo H., Mestreau E., Pelessone D., and Charman C., Fluid-Structure-Thermal Interaction Using a Loose Coupling Algorithm and Adaptive Unstructured Grids, Technical Report, Institute for Computational Sciences and Informatics, George Mason University, 1998.
- [90] Professor Rainald Löhner's Homepage – Computational Fluid Dynamics – GMU, <http://www.science.gmu.edu/~rlohner>.
- [91] Elad D., Katz D., Kimmel E., and Einav S., Numerical Schemes for Unsteady Fluid Flow through Collapsible Tubes, *Journal of Biomechanical Engineering*, **13**, pp.10-18, 1991.
- [92] Farhat C., Lesginne M., Maman N., Mixed Explicit/Implicit Time Integration of Coupled Aeroelastic Problems: Three-field Formulation, Geometric Conservation and Distributed Solution, *International Journal for Numerical Methods in Fluids*, **21**, pp.807-835, 1995.
- [93] Freitas F. B. Rachid, and Costa H.S. Mattos, Modelling of Pipeline Integrity Taking into Account the Fluid-Structure Interaction, *International Journal for Numerical Methods in Fluids*, **28**, pp.337-355, 1998.

- [94] Liang S. J., Neitzel G. P., and Aidun C. K., Finite Element Computations for Unsteady Fluid and Elastic Membrane Interaction Problems, *International Journal for Numerical Methods in Fluids*, **24**, pp.1091-1110, 1997.
- [95] Piperno S., Explicit/Implicit Fluid/Structure Staggered Procedures with a Structural Predictor and Fluid Subcycling for 2D Inviscid Aeroelastic Simulation, *International Journal for Numerical Methods in Fluids*, **25**, pp.1207-1226, 1997.
- [96] Soria A., and Casadei F., Arbitrary Lagrangian-Eulerian Multi-component Compressible Flow with Fluid-Structure Interaction, *International Journal for Numerical Methods in Fluids*, **25**, pp.1263-1284, 1997.
- [97] OpenMP home page, OpenMP: Simple, Portable, Scalable SMP Programming. <http://www.openmp.org>, 1999.
- [98] Gropp W., Lusk E., and Skjellum A., Using MPI: Portable Parallel Programming with the Message-Passing Interface, The MIT Press, Cambridge, Massachusetts, 1994.
- [99] Karypis G., and Kumar V., A Fast and High Quality Scheme for Partitioning Irregular Graphs, *SIAM Journal of Scientific Computing*, **20**, pp.359-392, 1999.
- [100] Balay S., Gropp W. D., McInnes L. C., and Smith B. F., The Portable Extensible Toolkit for Scientific computing (PETSc), <http://www.mcs.anl.gov/petsc.html>
- [101] Manke J. W., Parallel Computing in Aerospace, *Parallel Computing*, **27**, pp.329-336, 2001.
- [102] Hoeflinger J., Alavilli P., Jackson T., and Kuhn B., Producing Scalable Performance with OpenMP: Experiments with Two CFD Applications, *Parallel Computing*, **27**, pp.391-413, 2001.

- [103] Geist A., Beguelin A., Dongarra J., Jiang W. C., Manchek R., and Sunderam V., PVM : Parallel Virtual Machine: A Users' Guide and Tutorial for Networked Parallel Computing, Scientific and Engineering Computation, MIT Press, Cambridge, MA, USA, 1994.
- [104] McManus K., A Strategy for Mapping Unstructured Mesh Computational Mechanics Programs onto Distributed Memory Parallel Architectures, Ph.D. Thesis, School of Computing and Mathematical Science, University of Greenwich, 1996.
- [105] Yao J., Jameson A., Alonso J. J., and Liu F., Development and Validation of a Massively Parallel Flow Solver for Turbomachinery Flows, 38th Aerospace Sciences Meeting and Exhibit, *AIAA Paper* 00-0882, January 10-13, 2000.
- [106] Das R., Mavriplis D. J., Saltz J., Gupta S., and Ponnusamy R., The Design and Implementation of a Parallel Unstructured Euler Solver using Software Primitives, AIAA-92-0562, In Proceedings of the 30th Aerospace Sciences Meeting, January 1992.
- [107] Kalro V., and Tezduyar T. E., 3D Computation of Unsteady Flow Past a Sphere with a Parallel Finite Element Method, Army High Performance Computing Research Center Reprint 97-011, University of Minnesota, 1997.
- [108] Greenshields C.J., Weller H.G., Ivankovic A., The Finite Volume Method for Coupled Fluid Flow and Stress Analysis, *Computer Modelling and Simulation Engineering* **4**: 213-218, 1994.
- [109] Oldroyd A.B., Wheel M.A., Scanlon T.J., An Integrated Finite Volume Based Approach for Analysing Flow Induced Vibrations, *Proceedings European Conference on Computational Mechanics (ECCM)*, Munich, Germany, 1999.
- [110] Demirdzic I, Muzaferija S., Finite Volume Method for Stress Analysis in Complex Domains, *International Journal of Numerical Methods in Engineering* **37**: 3751-3766, 1994.

- [111] Bailey C., Cross M., A Finite Volume Procedure to Solve Elastic Solid Mechanics Problems in Three Dimensions on A Unstructured Mesh, *International Journal of Numerical Methods in Engineering* **38**: 1757-1776, 1995.
- [112] Wheel M.A., A Geometrically Versatile Finite Volume Formulation for Plane Elastostatic Stress Analysis, *Journal of Strain Analysis* **31**: 111-116, 1996.
- [113] Jasak H., Weller H.G., Application of the Finite Volume Method and Unstructured Meshes to Linear Elasticity, *International Journal of Numerical Methods in Engineering* **48**: 267-287, 2000.
- [114] Taylor G.A., Bailey C., Cross M., Solution of the Elastic Visco Plastic Constitutive Equations: A Finite Volume Approach, *Applied Mathematical Modelling* **19**: 747-760, 1995.
- [115] Wenke P., Wheel M.A., A Finite Volume Method for Predicting Finite Strain Deformations in Incompressible Materials, *Proceedings European Conference on Computational Mechanics (ECCM)*, Munich, Germany, 1999.
- [116] Maneeratana K., Ivankovic A., Finite Volume Method for Geometrically Nonlinear Stress Analysis Applications, *Proceedings of the Seventh Annual ACME Conference*, School of Engineering, University of Durham, 1999.
- [117] Slone A.K., Bailey C., Cross M., Dynamic Solid Mechanics Using Finite Volume Methods, *Applied Mathematical Modelling* **27**: 69-87, 2003.
- [118] Slone A.K., Pericleous K., Bailey C., Cross M., Bennett C., A Finite Volume Unstructured Mesh Approach to Dynamic Fluid-Structure Interaction: An Assessment of the Challenge of Predicting the Onset of Flutter, *Applied Mathematical Modelling* **28**: 211-239, 2004.

- [119] Fallah N., A Cell Vertex and Cell centred Finite Volume Method for Plate Bending Analysis, *Computer Methods in Applied Mechanics and Engineering*, **193**: 3457-3470, 2004.
- [120] Nay R.A., Utku S., An Alternative to the Finite Element Method. *Variational Methods in Engineering* **1**(3): 62-67, 1972.
- [121] Barnes M.R., From Finding and Analysis of Tension Space Structure by Dynamic Relaxation, Ph.D. Thesis, Department of Civil Engineering, The City University, London, 1977.
- [122] Hampshire J.K., Topping B.H.V., Chan H.C., Three Node Triangular Elements with One Degree of Freedom Per Node. *Engineering and Computation* **9**: 49-62, 1992.
- [123] Phaal R., Calladine C.R., A Simple Class of Finite Elements for Plate and Shell Problems I: Elements for Beams and Thin Plates, *International Journal for Numerical Methods in Engineering*, **35**: 955-977, 1992.
- [124] Phaal R., Calladine C.R., A Simple Class of Finite Elements for Plate and Shell Problems II: An Element for Thin Shells with Only Translational Degrees of Freedom, *International Journal for Numerical Methods in Engineering*, **35**: 979-996, 1992.
- [125] Yang D.Y., *et al.*, Comparative Investigation into Implicit, Explicit and Iterative Implicit/Explicit Schemes for simulation of Sheet Metal Forming Processes, *NUMISHEET'93*, Tokyo, 1993.
- [126] Brunet M., Sabourin F., Prediction of Necking and Wrinkles with A Simplified Shell Element in Sheet Forming, *International Conference of Metal Forming Simulation in Industry*, Volume **II**: 27-48, 1994.

- [127] Rio G., Tathi B., Laurent H., A New Efficient Finite Element Model of Shell with Only Three Degrees of Freedom Per Node, Applications to Industrial Deep Drawing Test. *18th IDDRG Biennial Congress*, Lisbon, 1994.
- [128] Oñate E., Cervera M., Derivation of Thin Plate Bending Elements with One Degree of Freedom Per Node, *Engineering and Computation* **10**: 543-561, 1993.
- [129] Oñate E., Cervera M., Zienkiewicz O.C., A Finite Volume Format for Structural Mechanics, *International Journal for Numerical Methods in Engineering*, **37**: 181-201, 1994.
- [130] Oñate E., Zarate F., Rotation-free Plate and Shell triangles, *International Journal for Numerical Methods in Engineering*, **47**: 557-603, 2000.
- [131] Fernando G.F., Oñate E., A Basic Thin Shell Triangle with Only Translational DOFs for Large Strain Plasticity, *International Journal for Numerical Methods in Engineering*, **51**: 57-83, 2001.
- [132] Oñate E., Fernando G.F., Advances in the Rotation-free Basic Shell Triangle, Computer Methods, *Journal of Computer Methods in Applied Mechanics and Engineering* (article in press), 2005.
- [133] John D. Anderson, JR., Computational Fluid Dynamics: The Basics with Applications, McGraw-Hill, Inc.
- [134] Feriger J. H., and Peric M., Numerical Methods for Fluid Dynamics, Oxford University Press, 1997.
- [135] Huang D., Computation of Compressible Flow using Euler Solver on Unstructured and Adaptive Meshes, MEng. Thesis, School of Mechanical and Production Engineering, Nanyang Technological University, 1997.

- [136] Roe P. L., Approximate Riemann Solvers, Parameter Vectors and Difference Schemes, *Journal of Computational Physics*, **43**: 357-372, 1981.
- [137] Bram van Leer, Towards the Ultimate Conservation Difference Scheme V, A Second-order Sequel to Godunov's Method, *Journal of Computational Physics*, **32**, pp.101-136, 1979.
- [138] Eberle A., Three-dimensional Euler Calculations using Characteristics Flux Extrapolation, *AIAA paper* 85-0119, 1985.
- [139] Drikakis D., Govatsos P. A., and Papantonis D.E., A Characteristic-Based Method for Incompressible Flows, *International Journal for Numerical Method in Fluids*, **19**(8): 667-685, 1994.
- [140] Rogers S. E., Kwak D., and Kiris C., Steady and Unsteady Solutions of the Incompressible Navier-Stokes Equations, *AIAA Journal*, **29**(4): 603-610, 1991.
- [141] Pedley T. J., and Stephanoff K. D., Flow Along a Channel with a Time-Dependent Indentation in One Wall: The Generation of Vorticity Waves, *Journal of Fluid Mechanics*, **160**: 337-367, 1985.
- [142] Lou H., Baum J. D., and Löhner R., An Accurate Fast, Matrix-Free Implicit Method for Computing Unsteady Flows on Unstructured Grids, *Computer & Fluids*, **30**: 137-159, 2001.
- [143] Hirsch C., Numerical Computation of Internal and External Flows, **2**, New York, A Wiley Interscience Publication, 1991.
- [144] Cai Z.X., Luo X.Y., A Fluid-Beam Model for Flow in A Collapsible Channel, *Journal of Fluids and Structures* **17**: 125-146, 2003.
- [145] Zienkiewicz O.C., Taylor, R.L., Finite Element Method (5th edition) Volume 2- Solid Mechanics, Elsevier, 2000.

- [146] Flynn M., Some Computer Organizations and Their Effectiveness, *IEEE Transaction Computer*, Volume **C-21**: 94, 1972.
- [147] Tai C. H., Zhao Y., and Liew K. M., Parallel Computation of Unsteady Three-Dimensional Incompressible Viscous Flow Using an Unstructured Multigrid Method, The Second M.I.T. Conference on Computational Fluid and Solid Mechanics, June 17-20, 2003, MIT, Cambridge, MA 02139, U.S.A.
- [148] Karypis G., and Kumar V., Metis: A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices, Version 4.0, University of Minnesota, Department of Computer Science, Sept 1998.
- [149] Peter S. Pacheco, Parallel Programming with MPI, Morgan Kaufmann Publishers, Inc., San Francisco, California, 1997.
- [150] Zhao Y. and Ahmed F., A General Method for Simulation of Fluid Flows with Moving and Compliant Boundaries Using Unstructured Grids, *Computer Methods in Applied Mechanics and Engineering*, **192**: 4439-4466, 2003.
- [151] Xia G.H., Zhao Y. and Yeo J. H., Numerical Simulation of 3D Fluid-Structure Interaction Using An Immersed Membrane Method, International Symposium on Physics of Fluids, Huangshan, China, 9-12 June, 2005.
- [152] Xia G.H., Zhao Y. and Yeo J. H., An Immersed Membrane Method For Simulation of Fluid-Structure Interaction in Bio-Fluid Flows, 1st International BioEngineering Conference (**IBEC 2004**), Singapore, 08-10, September, 2004.
- [153] Zhao Y., Zhang B. L., A High-Order Characteristics Upwind FV Method for Incompressible Flow and Heat Transfer Simulation on Unstructured Girds, *Journal of Computer Methods in Applied Mechanics and Engineering* **190**(5-7): 733-756, 2000.

- [154] St. Jude Medical Artificial Heart Valve, Cardiac Peacemakers and Other Medical Devices Home Page, <http://www.sjm.com>.
- [155] British Standard BS EN 12006-1:1999, Non-Active Surgical Implants – Particular Requirement for Cardiac and Vascular Implants, Part 1: Heart Valve Substitutes, 1999.
- [156] Shi Y.B., et al., Numerical Simulation of Opening Process In A Bi-leaflet Mechanical Heart Valve under Pulsatile Flow Condition, *The Journal of Heart Valve Disease*, Vol. 12 No.2, pp245-256, 2003.
- [157] Gilmanov A., et al., A Hybrid Cartesian/Immersed Boundary Method for Simulating Flows with 3D, Geometrically Complex, Moving Bodies, *Journal of Computational Physics*, **207**: 457-492, 2005.
- [158] Udaykumar H.S., et al., A Sharp Interface Cartesian Grid Method for Simulating Flows with Complex Moving Boundaries, *Journal of Computational Physics*, **174**: 345-380, 2001.
- [159] Zhao Y., Grid Generation Solver: **FORGEN** and **ST2UNST**, 1999.
- [160] Dennis, S.C.R., et al, The Steady Flow Due to A Rotating Sphere at Low and Moderate Reynolds Numbers, *Journal of Fluid Mechanics*, **101**: 257-259, 1980.
- [161] Fung Y. C., Biomechanics – Motion, Flow, Stress, and Growth, Springer-Verlag, 1990.
- [162] Nicholes Wilmwe W., and O'Rourke Michael F., McDonald's Blood Flow in Arteries, Theoretical, Experimental and Clinical Principles, 3rd edition, Lea & Feibiger, 1990.
- [163] Xu X. Y., and Collins M. W., Studies of Blood Flow in Arterial Bifurcations using Computational Fluid Dynamics, Proceedings of Institute of Mechanical

- Engineering, Part H: *Journal of Engineering in Medicine*, **208**, pp.163-175, 1994.
- [164] Hamid M.S., Sabbah H.N., Stein P.D., Influence of Stent Height Upon Stresses on The Cusps of Closed Bioprosthetic Valves, *Journal of Biomechanics* **19**: 759-769, 1986.
- [165] Tzabiras G., Calculation of Complex Turbulent Flows, WIT Press, 2000.
- [166] Bluestein D., and Einav S., Transition to Turbulence in Pulsatile Flow through Heart Valves – A Modified Stability Approach, *Journal of Biomechanical Engineering*, **116**, pp.477-487, 1994.
- [167] Liu J. S., Lu P. C., and Chu S. H., Turbulence Characteristics Downstream of Bileaflet Aortic Valve Prostheses, *Journal of Biomechanical Engineering*, **122**, pp.118-124, 2000.
- [168] Bellhouse B.J., and Talbot L., The Fluid Mechanics of The Aortic Valve, *Journal of Fluid Mechanics*, **35**, 721, 1969.
- [169] van Steenhoven A.A. and van Dongen M.E.H, Model Studies of The Closing Behaviour of The Aortic Valve, *Journal of Fluid Mechanics*, **90**, 21, 1979.
- [170] van Steenhoven A.A. and Veenstra P.C., The Effect of Some Hemodynamic Factors on The Behaviour of The Aortic Valve, *Journal of Biomechanics*, **15**, 941, 1982.
- [171] Xia G.H., Zhao Y. and Yeo J. H., Numerical Simulation of 3D Fluid-Structure Interaction Using An Immersed Membrane Method, (in Press), Modern Physics Letters, 2005.
- [172] X. Lv, Y. Zhao, X. Y. Huang, G. H. Xia and Z. J. Wang, An Efficient Parallel/Unstructured-Multigrid Preconditioned Implicit Method for

Simulating 3D Unsteady Compressible Flows with Arbitrarily Moving Objects, (in press), Journal of Computational Physics, 2005.