Numerical modeling of unsteady compressible gas flow around a projectile

Valery Ivanovich Ponyavin

University of Nevada, Las Vegas

Follow this and additional works at: https://digitalscholarship.unlv.edu/rtds

Repository Citation
https://digitalscholarship.unlv.edu/rtds/1694

This Thesis is brought to you for free and open access by Digital Scholarship@UNLV. It has been accepted for inclusion in UNLV Retrospective Theses & Dissertations by an authorized administrator of Digital Scholarship@UNLV. For more information, please contact digitalscholarship@unlv.edu.
NUMERICAL MODELING OF UNSTEADY COMPRESSIBLE GAS FLOW AROUND A PROJECTILE

by

Valery Ivanovich Ponyavin

Bachelor of Science
Kazan State Technical University, Kazan, Russia
1994

A thesis submitted in partial fulfillment of the requirements for the

Master of Science Degree in Mechanical Engineering
Department of Mechanical Engineering
Howard R. Hughes College of Engineering

Graduate College
University of Nevada, Las Vegas
August 2004
INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.
The Thesis prepared by

Valery Ivanovich Ponyavin

Entitled


is approved in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

Examination Committee Chair

Examination Committee Chair

Dean of the Graduate College

Examination Committee Member

Examination Committee Member

Graduate College Faculty Representative
ABSTRACT

Numerical Modeling of Unsteady Compressible Gas Flow Around a Projectile

by

Valery Ivanovich Ponyavin

Dr. Darrell W. Pepper, Examination Committee Co-Chair
Director, Nevada Center for Advanced Computational Methods
Professor of Department of Mechanical Engineering
University of Nevada, Las Vegas

Dr. Yitung Chen, Examination Committee Co-Chair
Associate Director, Nevada Center for Advanced Computational Methods
Associate Professor of Department of Mechanical Engineering
University of Nevada, Las Vegas

The calculation of compressible gas flow during the motion of a projectile in the gun barrel is a complicated computational task due to the presence of numerous factors, such as nonisothermicity, turbulence, changes in the shape of the computational domain with time, etc.

In this project, an attempt to calculate the characteristics of compressible gas flow around a projectile during the motion of the projectile in the gun barrel is undertaken. The flow is considered axisymmetrical, nonstationary, nonisothermal, compressible, and turbulent. For calculating the compressible gas flow around a projectile, the finite volume method was employed. An $h$-adaptive mesh refinement scheme based on elemental flow feature gradients is utilized for greater solution accuracy. For modeling flow around the
moving projectile both sliding and dynamic meshes were used.

The application of the calculations is in support of the Joint Actinide Shock Physics Experimental Research (JASPER). The JASPER facility utilizes a two-stage light gas gun to conduct equation of state experiments. The gun has a launch tube bore diameter of 28 mm, and is capable of launching projectiles at a velocity of 7.4 km/s using compressed hydrogen as a propellant. A numerical study is conducted to determine what effects, if any, launch tube exit geometry changes have on attitude of the projectile in flight. A comparison of two launch tube exit geometries is considered. The first case is standard muzzle geometry where the wall of the bore and the outer surface of the launch tube form a 90 degree angle. The second case includes a 26.6 degree bevel transition from the wall of the bore to the outer surface of the launch tube. For both cases, solutions are calculated for several positions downstream of the launch tube exit. The effect of beveled muzzle geometry on flight attitude of projectile is studied by using numerical modeling and results are compared with standard design, which is 90° of exit angle.
TABLE OF CONTENTS

ABSTRACT ................................................................................................................................. iii

TABLE OF CONTENTS .................................................................................................................. v

LIST OF FIGURES ...................................................................................................................... vii

ACKNOWLEDGEMENTS .............................................................................................................. x

CHAPTER 1 INTRODUCTION ....................................................................................................... 1

CHAPTER 2 LITERATURE SURVEY .......................................................................................... 3

CHAPTER 3 GOVERNING EQUATIONS FOR COMPRESSIBLE FLOW AND
METHODS OF SOLUTION ............................................................................................................... 6

3.1. Basic equations for compressible flows............................................................................. 8
3.2. Modeling basic fluid flow in FLUENT ............................................................................. 13
  3.2.1. Boundary conditions for compressible flow ............................................................. 13
  3.2.2. Solution strategies for compressible flows ................................................................. 14

CHAPTER 4 USING DEFORMING MESHES ........................................................................ 16

4.1. Sliding meshes .................................................................................................................... 16
  4.1.1. Sliding Mesh Theory ................................................................................................. 17
  4.1.2. Setup and Solution of a Sliding Mesh Problem. Grid Requirements ......................... 19
4.2. Dynamic Meshes ............................................................................................................... 20
  4.2.1. Dynamic Mesh Conservation Equations ...................................................................... 21
  4.2.2. Spring-based smoothing method .............................................................................. 22
  4.2.3. Dynamic layering method ....................................................................................... 25
  4.2.4. Local remeshing method ......................................................................................... 28
4.3. Solid-body kinematics ...................................................................................................... 33

CHAPTER 5 MESH ADAPTATION ..................................................................................... 37

5.1. Grid adaptation in FLUENT ............................................................................................. 37
5.2. Adaptation guidelines ........................................................................................................ 38
5.3. The static adaptation process ............................................................................................ 40
  5.3.1. Adaptation and mask registers .................................................................................. 40
  5.3.2. Hanging node adaptation ......................................................................................... 43
5.4. Gradient adaptation ........................................................................................................... 45
5.5. Dynamic gradient adaptation ........................................................................................... 48

CHAPTER 6 VALIDATION .................................................................................................. 51

6.1. Two-dimensional flat flow ................................................................................................. 51
CHAPTER 7  CALCULATION OF CHARACTERISTICS OF THE UNSTEADY
COMPRESSIBLE GAS FLOW AROUND A PROJECTILE ............................. 75
  7.1. Motionless projectile ................................................................. 75
  7.2. Moving projectile ..................................................................... 78
  7.3. Moving projectile to containment ............................................. 83

CHAPTER 8  APPLICATION TO LIGHT GAS GUN DESIGN .................. 88
  8.1. Calculations of quasi-steady state flow ................................. 91
  8.2. Unsteady flow ........................................................................ 97
    8.2.1. Example of calculation processes within launch tube .......... 98
    8.2.2. Example of calculation processes within target chamber ....... 100

CHAPTER 9  CONCLUSIONS ................................................................. 103

APPENDIX A Using User-Defined Functions (UDFs) in FLUENT .......... 105

NOMENCLATURE .................................................................................. 108

REFERENCES ....................................................................................... 110

VITA ......................................................................................................... 113

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
<table>
<thead>
<tr>
<th>Figure 4-1. Two Passing Trains in a Tunnel</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 4-2. Zones Created by Non-Periodic Interface Intersection</td>
<td>18</td>
</tr>
<tr>
<td>Figure 4-3. Two-Dimensional Grid Interface</td>
<td>18</td>
</tr>
<tr>
<td>Figure 4-4. Spring-Based Smoothing on Interior Nodes: Start</td>
<td>24</td>
</tr>
<tr>
<td>Figure 4-5. Spring-Based Smoothing on Interior Nodes: End</td>
<td>24</td>
</tr>
<tr>
<td>Figure 4-6. Dynamic Layering</td>
<td>25</td>
</tr>
<tr>
<td>Figure 4-7. Results of Splitting Layer By Constant Height</td>
<td>26</td>
</tr>
<tr>
<td>Figure 4-8. Results of Splitting Layer By Constant Ratio</td>
<td>27</td>
</tr>
<tr>
<td>Figure 4-9. Use of Sliding Interfaces to Transition Between Adjacent Cell Zones and the Dynamic Layering Cell Zone</td>
<td>29</td>
</tr>
<tr>
<td>Figure 4-10. Remeshing at a Deforming Boundary</td>
<td>30</td>
</tr>
<tr>
<td>Figure 4-11. Expanding Cylinder Before Local Face Remeshing</td>
<td>32</td>
</tr>
<tr>
<td>Figure 4-12. Expanding Cylinder After Local Face Remeshing</td>
<td>32</td>
</tr>
<tr>
<td>Figure 4-13. Solid Body Rotation Coordinates</td>
<td>35</td>
</tr>
<tr>
<td>Figure 5-1. Adaptation Register with Shaded Cells Representing Marked Cells</td>
<td>42</td>
</tr>
<tr>
<td>Figure 5-2. Mask Register with Shaded Cells Representing Active Cells</td>
<td>42</td>
</tr>
<tr>
<td>Figure 5-3. New Adaptation Register Created from Application of Mask</td>
<td>42</td>
</tr>
<tr>
<td>Figure 5-4. Example of a Hanging Node</td>
<td>43</td>
</tr>
<tr>
<td>Figure 5-5. Hanging Node Adaptation of 2-D Cell Types</td>
<td>44</td>
</tr>
<tr>
<td>Figure 5-6. Hanging Node Adaptation of 3D Cell Types</td>
<td>45</td>
</tr>
<tr>
<td>Figure 6-1. Computational domain</td>
<td>51</td>
</tr>
<tr>
<td>Figure 6-2. Initial mesh (7175 nodes, 6960 cells)</td>
<td>52</td>
</tr>
<tr>
<td>Figure 6-3. Adapted mesh (15954 nodes, 13668 cells)</td>
<td>52</td>
</tr>
<tr>
<td>Figure 6-4. Zooming of adapted mesh</td>
<td>53</td>
</tr>
<tr>
<td>Figure 6-5. Mach number</td>
<td>53</td>
</tr>
<tr>
<td>Figure 6-6. Static pressure (Pa)</td>
<td>54</td>
</tr>
<tr>
<td>Figure 6-7. Static temperature (K)</td>
<td>54</td>
</tr>
<tr>
<td>Figure 6-8. Density (kg/m³)</td>
<td>54</td>
</tr>
<tr>
<td>Figure 6-9. Molecular viscosity (kg/m-sec)</td>
<td>54</td>
</tr>
<tr>
<td>Figure 6-10. Schematic representation of a Mach reflection</td>
<td>55</td>
</tr>
<tr>
<td>Figure 6-11. Location of line for comparison</td>
<td>56</td>
</tr>
<tr>
<td>Figure 6-12. Mach number distribution along the line for comparison</td>
<td>56</td>
</tr>
<tr>
<td>Figure 6-13. Static pressure distribution along the line for comparison</td>
<td>57</td>
</tr>
<tr>
<td>Figure 6-14. Static temperature distribution along the line for comparison</td>
<td>58</td>
</tr>
<tr>
<td>Figure 6-15. Density distribution along the line for comparison</td>
<td>59</td>
</tr>
<tr>
<td>Figure 6-16. Triangles of velocities for oblique shock wave</td>
<td>59</td>
</tr>
<tr>
<td>Figure 6-17. Corner for oblique shock</td>
<td>60</td>
</tr>
<tr>
<td>Figure 6-18. Location of line for comparison</td>
<td>61</td>
</tr>
<tr>
<td>Figure 6-19. Static pressure distribution along the line for comparison</td>
<td>62</td>
</tr>
</tbody>
</table>
Figure 7.27. Contours of velocity magnitude, m/s (a-intermediate position, b-final position) .......................................................... 87
Figure 7.28. Contours of static temperature, K (a-intermediate position, b-final position) 87
Figure 8-1. Diagram of JASPER light gas gun ................................................................. 89
Figure 8-2. Cross-section of muzzle exit showing attached protectors ......................... 90
Figure 8-3. Axisymmetric representation of launch tube exit geometries ...................... 92
Figure 8-4. Example of axisymmetric case 1 mesh with two levels of h-adaptation .... 93
Figure 8-5. Example of axisymmetric case 2 mesh with two levels of h-adaptation .... 94
Figure 8-6. Pressure contours around projectile for case 1 at x = 4 mm ...................... 94
Figure 8-7. Pressure contours around projectile for case 2 at x = 4 mm ...................... 94
Figure 8-8. Pressure contours around projectile for case 1 at x = 16 mm ................. 95
Figure 8-9. Pressure contours around projectile for case 2 at x = 16 mm ................... 95
Figure 8-10. Pressure contours around projectile for case 1 at x = 32 mm ............... 95
Figure 8-11. Pressure contours around projectile for case 2 at x = 32 mm ............... 96
Figure 8-12. Pressure contours around projectile for case 1 at x = 48 mm ............... 96
Figure 8-13. Pressure contours around projectile for case 2 at x = 48 mm ............... 96
Figure 8-14. Pressure contours around projectile for case 1 at x = 64 mm ............... 97
Figure 8-15. Pressure contours around projectile for case 2 at x = 64 mm ............... 97
Figure 8-16. Stages for calculation .............................................................................. 98
Figure 8-17. Mesh for initial part of launch tube ......................................................... 99
Figure 8-18. Contours of density (intermediate position) ........................................... 99
Figure 8-19. Contours of static temperature (intermediate position) ......................... 100
Figure 8-20. Mesh for target chamber ...................................................................... 101
Figure 8-21. Contours of static pressure, Pa (intermediate position) ....................... 101
Figure 8-22. Contours of velocity magnitude, m/s (intermediate position) .......... 102
ACKNOWLEDGEMENTS

I would like to express my profound gratitude and sincere appreciation to Dr. Darrell W. Pepper and to Dr. Yitung Chen for their inspiration, indispensable guidance and financial support throughout the course of this study. With their initial support, this great dream of mine could finally turn to reality, and with their unending guidance throughout the study, I was able to navigate through the arduous and blinding path down to the very end.

In addition, I would also like to acknowledge Dr. Randy Clarksean for providing valuable suggestions along the way. Mr. Roald Akberov assisted me in various critiques and provided additional suggestions during the turbulent days of my studies. Moreover, thanks go to Dr. Hsuan-Tsung (Sean) Hsieh for teaching me invaluable knowledge, reviewing my thesis, and fruitful discussions about the project.
CHAPTER 1

INTRODUCTION

This thesis deals with the calculation of compressible fluid flow around a moving and motionless projectile. A basic road map of this work is presented in this chapter.

Chapter 2 represents the literature survey of experimental and theoretical works about motion of a projectile in a gun barrel.

Chapter 3 focuses on the governing equations for compressible flow and methods of solution of the equations.

Chapter 4 centers on descriptions of creating deforming meshes which are necessary for calculating compressible fluid flow in deforming zones.

One of the primary disadvantages of attacking complex problems relative to calculation of compressible fluid flow is that the number of nodes (or elements) required to resolve certain flow phenomena increases substantially. Finer mesh density is especially important in accurately capturing various flow features, such as the precise locations of shocks. Rather than using a finer mesh throughout the entire solution domain, mesh adaptation is employed. Further discussion on the mesh adaptation methods and strategies concerning this work will be handled in Chapter 5.

Results of several benchmark test cases will be presented in Chapter 6. Benchmarking is an important part of numerical model development. If results of well documented
experimental data or theoretical data can be duplicated with a numerical model, then greater trust can be given to that model's results as it is applied to new problems.

Chapter 7 focuses on characteristics of the unsteady compressible gas flow around a projectile in a gun barrel for several cases: motionless projectile, moving projectile, moving projectile to containment.

In Chapter 8, results for the flow field around the muzzle of a light gas gun are presented. Two different muzzle configurations are considered. The first case is standard muzzle geometry where the wall of the bore and the outer surface of the launch tube form a 90 degree angle. The second case includes a 26.6 degree bevel transition from the wall of the bore to the outer surface of the launch tube. Due to the extreme nature of the problem, numerical simulation is the only feasible way of examining what effects, if any, launch tube exit geometry changes have on the attitude of the projectile in flight.

Finally, in Chapter 9, conclusions will be drawn based on the results obtained, and recommendations will be made regarding future research.
CHAPTER 2

LITERATURE SURVEY

The main principles of the motion of a projectile in a gun barrel were proposed by Prof. Hertzberg’s and his team at the University of Washington, where most of the pioneering theoretical and experimental research was done. The research is based on the ramjet principle. A sharp-nosed projectile, which resembles the centerbody of a conventional ramjet runs inside a tube. The tube acts as the outer cowling of the ramjet and the energy release by combustion produces high pressure at the base of the projectile providing thrust. Several modes of operation have been suggested with great potential for applications in surface-to-orbit launching of inert payloads by Bogdanoff (1992) and in ground-based testing of hypersonic propulsive cycles by Bruckner, Knowlen and Hertzberg (1992). The reason for investigating such phenomena was for obtaining reliable information on the processes taking place around a fast moving projectile inside the ram accelerator.

The numerical simulations of unsteady motion of the projectile in a gun barrel were performed by several investigators. M. J. Nusca (1997) used CFD solutions of the full Navier-Stokes equations along with finite-rate chemical kinetics to numerically simulate the reacting in-bore flowfield for 120 mm ram accelerator projectile propulsion system. He investigated various unsteady compressible flow phenomena, including projectile starting, obturator discard and high-velocity unstarting. The simulations illustrate the
importance of obturator discard dynamics in achieving a successful starting of the ram acceleration process. However, once started, high projectile velocities can induce an unstart as the combustion wave precedes the projectile under certain conditions.

Steady and unsteady numerical simulations are conducted for the experiments performed to investigate the ram accelerator flow field by using the expansion tube facility in Stanford University by Choi, Jeung and Yoon (1997). Navier-Stokes equations for chemically reacting flows are analyzed by fully implicit time accurate numerical method with Jachimowski’s detailed chemistry mechanism for hydrogen-air combustion involving 9 species and 19 reaction steps (1988). Although the steady state assumption shows a good agreement with the experimental schlieren and OH PLIF images for the case of $2H_2+O_2+17N_2$, it fails in reproducing the combustion region behind the shock intersection point shown in the case of $2H_2+O_2+12N_2$ mixture. Therefore, an unsteady numerical simulation was conducted for this case and the result showed all the detailed flow stabilization process. The experimental results were revealed to be an instantaneous result during the flow stabilization process. The combustion behind the shock intersection point was the result of a normal detonation formed by the intersection of strong oblique shocks that exist at early stage of the stabilization process. At final stage, the combustion region behind the shock intersection point disappeared and the steady state result was retained. The time required for stabilization of the reacting flow in the model ram accelerator was found to be very long in comparison with the experimental test time.

In article of Henner and others (1997) the numerical simulation of the flow around the body was conducted with the Navier-Stokes code TASCflow, used in a non-reactive, steady and three-dimensional version. The aim of the computational work was to
contribute to the projectile shape optimization under pure aerodynamical conditions, and therefore, to localize shock waves and their interactions, recirculation zone and high-temperature areas. Results presented in this paper show the effects of the fins on the flow configuration around the body and at the base (recirculation zone). Parameters such as profile of leading edges and the number and size of fins were taken into account to compare the different flow fields. The results of computation, related to experiments performed in inert gas with different projectiles, were used to highlight the influence of these parameters on the efficiency of the diffuser formed by the projectile afterbody. The pressure and temperature in the compressible fluid flow around a projectile when entering the ram tube and just before the initiation of the combustion were given. The presented work demonstrated that the computer code TASCflow could provide a valuable tool for the optimization of the projectile geometry.
CHAPTER 3

GOVERNING EQUATIONS FOR COMPRESSIBLE FLOW
AND METHODS OF SOLUTION

Compressibility effects are encountered in gas flows at high velocity and/or in which
there are large pressure variations. When the flow velocity approaches or exceeds the
speed of sound of the gas or when the pressure change (Δp) in the system is large, the
variation of the gas density with pressure has a significant impact on the flow velocity,
pressure, and temperature.

Compressible flows can be characterized by the value of the Mach number:

\[ M = \frac{u}{a} \]

Here, \( a \) is the speed of sound in the gas \( (a = \sqrt{\gamma RT}) \)
u is the gas flow velocity,
and \( \gamma \) is the ratio of specific heats \((c_p/c_v)\).

When the Mach number is less than 1.0, the flow is termed subsonic. At Mach
numbers much less than 1.0 \((M < 0.3 \text{ or so})\), compressibility effects are negligible and the
variation of the gas density with pressure can safely be ignored in the flow modeling. As
the Mach number approaches 1.0 (which is referred to as the transonic flow regime),
compressibility effects become important. When the Mach number exceeds 1.0, the flow
is termed supersonic, and may contain shocks and expansion fans which can impact the
flow pattern significantly. Commercial software FLUENT provides a wide range of
compressible flow modeling capabilities for subsonic, transonic, and supersonic flows.

Compressible flows are typically characterized by the total pressure $P$ and total
temperature $T_{tot}$ of the flow. For an ideal gas, these quantities can be related to the static
pressure and temperature by the following:

$$\frac{P}{p} = \left(1 + \frac{\gamma - 1}{2} M^2\right)^{\frac{\gamma}{\gamma - 1}} \quad (3.1)$$

$$\frac{T_{tot}}{T} = 1 + \frac{\gamma - 1}{2} M^2 \quad (3.2)$$

where $p$ is the static pressure and $T$ is the static temperature.

These relationships describe the variation of the static pressure and temperature in the
flow as the velocity (Mach number) changes under isentropic conditions. For example,
given a pressure ratio from inlet to exit (total to static), Eq. 3.1 can be used to estimate
the exit Mach number which would exist in a one-dimensional isentropic flow. For air,
Eq. 3.1 predicts a choked flow (Mach number of 1.0) at an isentropic pressure ratio, $P/p$,
of 0.5283. This choked flow condition will be established at the point of minimum flow
area (e.g., in the throat of a nozzle). In the subsequent area expansion the flow may either
accelerate to a supersonic flow in which the pressure will continue to drop, or return to
subsonic flow conditions, decelerating with a pressure rise. If a supersonic flow is
exposed to an imposed pressure increase, a shock will occur, with a sudden pressure rise
and deceleration accomplished across the shock.
3.1. Basic equations for compressible flows

The problem of interest is a 2-D axisymmetric compressible turbulent flow. The continuity equation is given by

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x} (\rho u) + \frac{1}{r} \frac{\partial}{\partial r} (\rho v) + \frac{\rho v}{r} = 0$$

where $x$ is the axial coordinate, $r$ is the radial coordinate, $u$ is the axial velocity, and $v$ is the radial velocity.

The axial and radial momentum conservation equations are given by

$$\frac{\partial \rho u}{\partial t} + u \frac{\partial (\rho u)}{\partial x} + \frac{1}{r} v \frac{\partial (\rho v)}{\partial r} - \frac{\partial}{\partial x} \left( (v + v_T) \frac{\partial \rho u}{\partial x} \right)$$

$$- \frac{1}{r} \frac{\partial}{\partial r} \left( (v + v_T) r \frac{\partial \rho u}{\partial r} \right)$$

$$= \frac{1}{r} \left\{ \frac{\partial}{\partial x} \left( v_T r \frac{\partial \rho u}{\partial x} \right) + \frac{\partial}{\partial r} \left( v_T r \frac{\partial \rho u}{\partial r} \right) \right\} - \frac{\partial}{\partial x} \left( p + \frac{2}{3} k \right)$$

and

$$\frac{\partial \rho v}{\partial t} + u \frac{\partial (\rho v)}{\partial x} + \frac{1}{r} v \frac{\partial (\rho v)}{\partial r} - \frac{\partial}{\partial x} \left( (v + v_T) \frac{\partial \rho v}{\partial x} \right)$$

$$- \frac{1}{r} \frac{\partial}{\partial r} \left( (v + v_T) r \frac{\partial \rho v}{\partial r} \right)$$

$$= \frac{1}{r} \left\{ \frac{\partial}{\partial x} \left( v_T r \frac{\partial \rho v}{\partial x} \right) + \frac{\partial}{\partial r} \left( v_T r \frac{\partial \rho v}{\partial r} \right) \right\}$$

$$- \frac{\partial}{\partial r} \left( p + \frac{2}{3} k \right) - (v + 2 v_T) \frac{\partial v}{\partial r}$$

where $v_T$ is the turbulent kinematic viscosity, $k$ - turbulent kinetic energy (TKE), $p$ - static pressure.
Turbulent heat transport is modeled using the concept of Reynolds' analogy to turbulent momentum transfer. The "modeled" energy equation is thus given by the following:

\[
\frac{\partial}{\partial t} (\rho E) + \frac{\partial}{\partial x} \left[ u (\rho E + p) \right] + \frac{1}{\tau} \frac{\partial}{\partial \tau} \left[ r v (\rho E + p) \right] = \frac{\partial}{\partial x} \left[ \left( k + \frac{c_p \mu_T}{\text{Pr}_T} \right) \frac{\partial T}{\partial x} + u \tau_{ij,\text{eff}} \right] \\
+ \frac{1}{\tau} \frac{\partial}{\partial \tau} \left[ \left( k + \frac{c_p \mu_T}{\text{Pr}_T} \right) \frac{\partial T}{\partial x} + ru \tau_{ij,\text{eff}} \right] 
\] (3.6)

where \( E \) is the total energy, \( \mu_T \) is the turbulent kinematic viscosity and \( \tau_{ij,\text{eff}} \) is the deviatoric stress tensor, defined as

\[
\tau_{ij,\text{eff}} = \mu_{\text{eff}} \left( \frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} \right) - \frac{2}{3} \mu_{\text{eff}} \frac{\partial u_i}{\partial x_i} \delta_{ij} 
\] (3.7)

The term involving \( \tau_{ij,\text{eff}} \) represents the viscous heating, and is always computed in the coupled solvers. The value of the turbulent Prandtl number is 0.85.

Wall boundary conditions for scalar transport are handled analogously to momentum, using the appropriate "law-of-the-wall".

The state equation:

\[
p = \frac{\rho}{M^2 \gamma} \left[ E - M^2 \gamma (\gamma - 1) \left( k + \frac{u^2 + v^2}{2} \right) \right] 
\] (3.8)

In turbulence models that employ the Boussinesq approach, the central issue is how the eddy viscosity is computed. The model proposed by Spalart and Allmaras (1992) solves a transport equation for a quantity that is a modified form of the turbulent kinematic viscosity.
The transport equation for the $v_T$ in the Spalart-Allmaras model is

$$
\rho \frac{Dv_T}{Dt} = G_v + \frac{1}{\sigma_{v_T}} \left[ \frac{\partial}{\partial x} \left( \mu + \rho v_T \right) \frac{\partial v_T}{\partial x} \right] + \frac{1}{\rho \sigma_{v_T}} \left[ \frac{\partial}{\partial r} \left( \mu + \rho v_T \right) \frac{\partial v_T}{\partial r} \right] + C_{b_2} \rho \left( \frac{\partial v_T}{\partial r} \right) + Y_v
$$

(3.9)

where $G_v$ is the production of turbulent viscosity and $Y_v$ is the destruction of turbulent viscosity that occurs in the near-wall region due to wall blocking and viscous damping. $\sigma_{v_T}$ and $C_{b_2}$ are constants and $\nu$ is the molecular kinematic viscosity.

The turbulent viscosity, $\mu_T$, is computed from

$$
\mu_T = \rho v_T f_{u_1}
$$

(3.10)

where the viscous damping function, $f_{u_1}$, is given by

$$
f_{u_1} = \frac{\chi^3}{\chi^3 + C_{ol}^3}
$$

(3.11)

and

$$
\chi = \frac{v_T}{\nu}
$$

(3.12)

The production term, $G_v$, is modeled as

$$
G_v = C_{u_1} \rho \tilde{S} v_T
$$

(3.13)

where

$$
\tilde{S} = S + \frac{v_T}{K^2 \delta^2} f_{u_2}
$$

(3.14)
\[ f_{u2} = 1 - \frac{\chi}{1 + \chi f_{v1}} \]  

(3.15)

\[ C_{b1} \text{ and } \kappa \text{ are constants, } d \text{ is the distance from the wall, and } S \text{ is a scalar measure of the deformation tensor. By default in FLUENT, as in the original model proposed by Spalart and Allmaras, } S \text{ is based on the magnitude of the vorticity:} \]

\[ S = \sqrt{2 \Omega_{yj} \Omega_{yi}} \]  

(3.16)

where \( \Omega_{yj} \) is the mean rate-of-rotation tensor and is defined by

\[ \Omega_{yj} = \frac{1}{2} \left( \frac{\partial u_j}{\partial x_i} - \frac{\partial u_i}{\partial x_j} \right) \]  

(3.17)

The justification for the default expression for \( S \) is that, for the wall-bounded flows turbulence is found only where vorticity is generated near walls. However, it has since been acknowledged that one should also take into account the effect of mean strain on the turbulence production, and a modification to the model has been proposed and used here.

This modification combines measures of both rotation and strain tensors in the definition of \( S \):

\[ S = |\Omega_{yj}| + C_{\text{prod}} \min(0, |S_{yj}| - |\Omega_{yj}|) \]  

(3.18)

where

\[ C_{\text{prod}} = 2.0, \quad |\Omega_{yj}| \equiv \sqrt{2 \Omega_{yj} \Omega_{yi}}, \quad |S_{yj}| \equiv \sqrt{2 S_{yj} S_{ji}} \]  

(3.19)

with the mean strain rate, \( S_{yj} \), defined as

\[ S_{yj} = \frac{1}{2} \left( \frac{\partial u_j}{\partial x_i} + \frac{\partial u_i}{\partial x_j} \right) \]  

(3.20)

The destruction term is modeled as

\[ \frac{1}{1 + \chi f_{v1}} \]  

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.
\[ Y_v = C_{w1} \rho f_w \left( \frac{v_T}{d} \right)^2 \] (3.21)

where

\[ f_w = g \left[ \frac{1 + C_{w3}^6}{g^6 + C_{w3}^6} \right]^{1/6} \] (3.22)

\[ g = r + C_{w2}(r^6 - r) \] (3.23)

\[ r \equiv \frac{v_T}{S \kappa^2 d^2} \] (3.24)

\( C_{w1}, C_{w2}, \) and \( C_{w3} \) are constants.

The model constants \( C_{b1}, C_{b2}, \sigma_y, C_{u1}, C_{w1}, C_{w2}, \) \( C_{w3} \) and \( \kappa \) have the following default values:

\[ C_{b1} = 0.1335, \quad C_{b2} = 0.622, \quad \sigma_y = 2/3, \]

\[ C_{u1} = 7.1, \quad C_{w1} = \frac{C_{b1}}{\kappa^2} + \frac{1 + C_{b2}}{\sigma_y}, \]

\[ C_{w2} = 0.3, \]

\[ C_{w3} = 2.0, \]

\[ \kappa = 0.41 \] (3.25)

At walls, the modified turbulent kinematic viscosity, \( v_T \), is set to zero.

When the mesh is fine enough to resolve the laminar sublayer, the wall shear stress is obtained from the laminar stress-strain relationship:

\[ \frac{u}{u_\tau} = \frac{\rho u_\tau y}{\mu} \] (3.26)

If the mesh is too coarse to resolve the laminar sublayer, it is assumed that the centroid of the wall-adjacent cell falls within the logarithmic region of the boundary layer, and the law-of-the-wall is employed:
where $\kappa = 0.419$ and $E_i = 9.793$.

3.2. Modeling basic fluid flow in FLUENT

For calculating compressible flows we do not need to activate any special physical models (other than the compressible treatment of density as detailed below). The energy equation solved by FLUENT correctly incorporates the coupling between the flow velocity and the static temperature, and should be activated whenever we are solving a compressible flow. In addition, if we are using the segregated solver, we should activate the viscous dissipation terms in Eq. 3.6, which become important in high-Mach-number flows.

In FLUENT for compressible flows, the ideal gas law is written in the following form:

$$
\rho = \frac{p_{op} + p}{R \frac{M_w T}{}}
$$

where $p_{op}$ is the operating pressure defined in the Operating Conditions panel in FLUENT, $p$ is the local static pressure relative to the operating pressure, $R$ is the universal gas constant, and $M_w$ is the molecular weight. The temperature, $T$, will be computed from the energy equation.

3.2.1. Boundary conditions for compressible flow

Well-posed inlet and exit boundary conditions for compressible flow are listed below:

For flow inlets:
Pressure inlet: Inlet total temperature and total pressure and, for supersonic inlets, static pressure.

Mass flow inlet: Inlet mass flow and total temperature

For flow exits:

Pressure outlet: Exit static pressure (ignored if flow is supersonic at the exit).

It is important to note that our boundary condition inputs for pressure (either total pressure or static pressure) must be in terms of gauge pressure, i.e., pressure relative to the operating pressure.

All temperature inputs at inlets should be total (stagnation) temperatures, not static temperatures.

3.2.2. Solution strategies for compressible flows

The difficulties associated with solving compressible flows are a result of the high degree of coupling between the flow velocity, density, pressure, and energy. This coupling may lead to instabilities in the solution process and, therefore, may require special solution techniques in order to obtain a converged solution. In addition, the presence of shocks (discontinuities) in the flow introduces an additional stability problem during the calculation.

Solution techniques that may be beneficial in compressible flow calculations include the following:

1) Use conservative under-relaxation parameters on the velocities, perhaps values of 0.2 or 0.3.

2) Set the under-relaxation on pressure to a value of 0.1 or so and use the SIMPLE algorithm.
3) Set reasonable limits for the temperature and pressure to avoid solution divergence, especially at the start of the calculation.

4) If required, begin the calculations using a reduced pressure ratio at the boundaries, increasing the pressure ratio gradually in order to reach the final desired operating condition. We can also consider starting the compressible flow calculation from an incompressible flow solution (although the incompressible flow solution can in some cases be a rather poor initial guess for the compressible calculation).

In some cases, computing an inviscid solution as a starting point may be helpful.
4.1. Sliding meshes

The sliding mesh model is the most accurate method for simulating flows in multiple moving reference frames, but also the most computationally demanding. Most often, the unsteady solution that is sought in a sliding mesh simulation is time periodic. That is, the unsteady solution repeats with a period related to the speeds of the moving domains. However, we can model other types of transients, including translating sliding mesh zones (e.g., two cars or trains passing in a tunnel, as shown in Figure 4-1).

![Figure 4-1. Two Passing Trains in a Tunnel](image)

In the sliding mesh technique two or more cell zones are used. Each cell zone is bounded by at least one “interface zone” where it meets the opposing cell zone. The interface zones of adjacent cell zones are associated with one another to form a “grid interface.” The two cell zones will move relative to each other along the grid interface. The grid interface must be positioned so that it has fluid cells on both sides. During the
calculation, the cell zones slide relative to one another along the grid interface in discrete steps.

The grid interface and the associated interface zones can be any shape, provided that the two interface boundaries are based on the same geometry.

4.1.1. Sliding Mesh Theory

The sliding mesh model allows adjacent grids to slide relative to one another. In doing so, the grid faces do not need to be aligned on the grid interface. This situation requires a means of computing the flux across the two non-conformal interface zones of each grid interface.

To compute the interface flux, the intersection between the interface zones is determined at each new time step. The resulting intersection produces one interior zone (a zone with fluid cells on both sides) and one or more periodic zones. If the problem is not periodic, the intersection produces one interior zone and a pair of wall zones (which will be empty if the two interface zones intersect entirely), as shown in Figure 4-2. The resultant interior zone corresponds to where the two interface zones overlap; the resultant periodic zone corresponds to where they do not. The number of faces in these intersection zones will vary as the interface zones move relative to one another. Principally, fluxes across the grid interface are computed using the faces resulting from the intersection of the two interface zones, rather than from the interface zone faces themselves.
In the example shown in Figure 4.3 the interface zones are composed of faces A-B and B-C, and faces D-E and E-F. The intersection of these zones produces the faces a-d, d-b, b-e, etc. Faces produced in the region where the two cell zones overlap (d-b, b-e, and
e-c) are grouped to form an interior zone, while the remaining faces (a-d and c-f) are paired up to form a periodic zone. To compute the \( u_\text{x} \) across the interface into cell IV, for example, face D-E is ignored and faces d-b and b-e are used instead, bringing information into cell IV from cells I and III, respectively.

4.1.2. Setup and Solution of a Sliding Mesh Problem. Grid Requirements

Before beginning the problem setup in FLUENT, we need to be sure that the grid we have created meets the following requirements:

1) A different cell zone exists for each portion of the domain that is sliding at a different speed.

2) The grid interface must be situated such that there is no motion normal to it.

3) The grid interface can be any shape (including a non-planar surface, in 3D), provided that the two interface boundaries are based on the same geometry. If there are sharp features in the mesh (e.g., 90-degree angles), it is especially important that both sides of the interface closely follow that feature.

4) If we create a single grid with multiple cell zones, we must be sure that each cell zone has a distinct face zone on the sliding boundary. The face zones for two adjacent cell zones will have the same position and shape, but one will correspond to one cell zone and one to the other.
4.2. Dynamic Meshes

The dynamic mesh model in FLUENT can be used to model flows where the shape of the domain is changing with time due to motion on the domain boundaries. The motion can be a prescribed motion (e.g., we can specify the linear and angular velocities about the center of gravity of a solid body with time) or an unprescribed motion where the subsequent motion is determined based on the solution at the current time (e.g., the linear and angular velocities are calculated from the force balance on a solid body). The update of the volume mesh is handled automatically by FLUENT at each time step based on the new positions of the boundaries. To use the dynamic mesh model, we need to provide a starting volume mesh and the description of the motion of any moving zones in the model. FLUENT allows us to describe the motion using either boundary profiles or user-defined functions (UDFs).

FLUENT expects the description of the motion to be specified on either face or cell zones.

If the model contains moving and non-moving regions, we need to identify these regions by grouping them into their respective face or cell zones in the starting volume mesh that we generate. Furthermore, regions that are deforming due to motion on their adjacent regions must also be grouped into separate zones in the starting volume mesh. The boundary between the various regions need not be conformal. We can use the nonconformal or sliding interface capability in FLUENT to connect the various zones in the final model.
4.2.1. Dynamic Mesh Conservation Equations

The integral form of the conservation equation for a general scalar, \( \phi \), on an arbitrary control volume, \( V \), whose boundary is moving can be written as

\[
\frac{d}{dt} \int_V \rho \phi \, dV + \int_{\partial V} \rho \phi (\vec{u} - \vec{u}_g) \cdot dA = \int_V \nabla \phi \cdot dA + \int_V S_\phi \, dV
\]

where \( \rho \) is the fluid density

\( \vec{u} \) is the flow velocity vector

\( \vec{u}_g \) is the grid velocity of the moving mesh

\( \Gamma \) is the diffusion coefficient

\( S_\phi \) is the source term of \( \phi \)

Here \( \partial V \) is used to represent the boundary of the control volume \( V \).

The time derivative term in Equation 4.1 can be written, using a first-order backward difference formula, as

\[
\frac{d}{dt} \int_V \rho \phi \, dV = \frac{(\rho \phi V)^{n+1} - (\rho \phi V)^n}{\Delta t}
\]

where \( n \) and \( n+1 \) denote the respective quantity at the current and next time level. The \((n+1)\)th time level volume \( V^{n+1} \) is computed from

\[
V^{n+1} = V^n + \frac{dV}{dt} \Delta t
\]

where \( \frac{dV}{dt} \) is the volume time derivative of the control volume. In order to satisfy the grid conservation law, the volume time derivative of the control volume is computed from
\[
\frac{\text{d}V}{\text{d}t} = \int_{\text{d}V} \vec{u}_{\text{g},j} \cdot \text{d}\vec{A}_j = \sum_{j} \vec{u}_{\text{g},j} \cdot \vec{A}_j
\]  

(4.4)

where \( n_f \) is the number of faces on the control volume and \( \vec{A}_j \) is the \( j \) face area vector. The dot product \( \vec{u}_{\text{g},j} \cdot \vec{A}_j \) on each on each control volume face is calculated from

\[
\vec{u}_{\text{g},j} \cdot \vec{A}_j = \frac{\delta V}{\Delta t}
\]  

(4.5)

where \( \delta V_j \) is the volume swept out by the control volume face \( j \) over the time step \( \Delta t \).

Dynamic mesh update methods

Three mesh motion methods are available in FLUENT to update the volume mesh in the deforming regions subject to the motion defined at the boundaries:

- spring-based smoothing
- dynamic layering
- local remeshing

4.2.2. Spring-based smoothing method

In the spring-based smoothing method, the edges between any two mesh nodes are idealized as a network of interconnected springs. The initial spacings of the edges before any boundary motion constitute the equilibrium state of the mesh. A displacement at a given boundary node will generate a force proportional to the displacement along all the springs connected to the node. Using Hook's Law, the force on a mesh node can be written as

\[
\vec{F}_i = \sum_{j} k_{ij} \left( \Delta \vec{x}_j - \Delta \vec{x}_i \right)
\]  

(4.6)
where $\Delta \bar{x}_i$ and $\Delta \bar{x}_j$ are the displacements of node $i$ and its neighbor $j$, $n_i$ is the number of neighboring nodes connected to node $i$, and $k_{ij}$ is the spring constant (or stiffness) between node $i$ and its neighbor $j$. The spring constant for the edge connecting nodes $i$ and $j$ is defined as

$$k_{ij} = \frac{1}{\sqrt{|\bar{x}_i - \bar{x}_j|}}$$

At equilibrium, the net force on a node due to all the springs connected to the node must be zero. This condition results in an iterative equation such that

$$\sum_{j}^{n} k_{ij} \Delta \bar{x}_{ji}^{m+1} = \sum_{j}^{n} k_{ij} \Delta \bar{x}_{ji}^{m}$$

where $m$ is the number of the current iteration.

Since displacements are known at the boundaries (after boundary node positions have been updated), Eq. 4.8 is solved using a Jacobi sweep on all interior nodes. At convergence, the positions are updated such that

$$\bar{x}_i^{n+1} = \bar{x}_i^{n} + \Delta \bar{x}_i^{m,\text{converged}}$$

where $n+1$ and $n$ are used to denote the positions at the next time step and the current time step, respectively. The spring-based smoothing is shown in Figures 4.4 and 4.5 for a cylindrical cell zone where one end of the cylinder is moving.
Applicability of the spring-based smoothing method

We can use the spring-based smoothing method to update any cell or face zone whose boundary is moving or deforming.

For non-tetrahedral cell zones (non-triangular in 2-D), the spring-based method is recommended when the following conditions are met:
- The boundary of the cell zone moves predominantly in one direction (i.e., no excessive anisotropic stretching or compression of the cell zone).
- The motion is predominantly normal to the boundary zone.

If these conditions are not met, the resulting cells may have high skewness values, since not all possible combinations of node pairs in non-tetrahedral cells (or non-triangular in 2D) are idealized as springs.

4.2.3. Dynamic layering method

In prismatic (hexahedral and/or wedge) mesh zones, we can use dynamic layering to add or remove layers of cells adjacent to a moving boundary, based on the height of the layer adjacent to the moving surface. The dynamic mesh model in FLUENT allows us to specify an ideal layer height on each moving boundary. The layer of cells adjacent to the moving boundary (layer j in Figure 4.6) is split or merged with the layer of cells next to it (layer i in Figure 4.6) based on the height (h) of the cells in layer j.

![Diagram of dynamic layering](image)

Figure 4-6. Dynamic Layering

If the cells in layer j are expanding, the cell heights are allowed to increase until
\[ h_{\text{min}} > (1 + \alpha_s)h_{\text{ideal}} \]  \hspace{1cm} (4.10)

where \( h_{\text{min}} \) is the minimum cell height of cell layer \( j \), \( h_{\text{ideal}} \) is the ideal cell height, and \( \alpha_s \) is the layer split factor. When this condition is met, the cells are split based on the specified layering option: constant height or constant ratio.

With the constant height option, the cells are split to create a layer of cells with constant height \( h_{\text{ideal}} \) and a layer of cells of height \( h - h_{\text{ideal}} \). With the constant ratio option, the cells are split such that locally, the ratio of the new cell heights is exactly \( \alpha_s \) everywhere.

Figures 4.7 and 4.8 show the result of splitting a layer of cells above a valve geometry using the constant height and constant ratio option.

Figure 4-7. Results of Splitting Layer By Constant Height
If the cells in layer $j$ are being compressed, they can be compressed until

$$h_{\text{min}} < \alpha \cdot h_{\text{ideal}}$$  \hfill (4.11)

where $\alpha$ is the layer collapse factor. When this condition is met, the compressed layer of cells is merged into the layer of cells above the compressed layer; i.e., the cells in layer $j$ are merged with those in layer $i$.

**Applicability of the dynamic layering method**

We can use the dynamic layering method to split or merge cells adjacent to any moving boundary provided the following conditions are met:

- All cells adjacent to the moving face zone are either wedges or hexahedra (quadrilaterals in 2-D) even though the cell zone may contain mixed cell shapes. The cell layers must be completely bounded by one-sided face zones, except when sliding interfaces are used.
• If the bounding face zones are two-sided walls, we must split the wall and wallshadow pair and use the coupled sliding interface option to couple the two adjacent cell zones.

• If our model contains periodic face zones in the cell zone where dynamic layering is used, we can only use the serial version of the solver. However, if we model the periodic zones as periodic non-conformal interfaces, then we can use the parallel solver for dynamic layering.

• If the moving boundary is an internal zone, cells on both sides (possibly with different ideal cell layer heights) of the internal zone are considered for dynamic layering.

• If we want to use dynamic layering on cells adjacent to a moving wall that do not span from boundary to boundary, we must separate those cells which are involved in the dynamic layering and use the sliding interfaces capability in FLUENT to transition from the deforming cells to the adjacent non-deforming cells (see Figure 4.9).

4.2.4. Local remeshing method

On zones with a triangular or tetrahedral mesh, the spring-based smoothing method is normally used. When the boundary displacement is large compared to the local cell sizes, the cell quality can deteriorate or the cells can become degenerate. This will invalidate the mesh (e.g., result in negative cell volumes) and consequently, will lead to convergence problems when the solution is updated to the next time step.
To circumvent this problem, FLUENT agglomerates cells that violate the skewness or size criteria and locally remeshes the agglomerated cells. If the new cells satisfy the skewness and the size criteria, the mesh is locally updated with the new cells (with the solution interpolated from the old cells). Otherwise, the new cells are discarded.

FLUENT evaluates each cell and marks it for remeshing if it meets one or more of the following criteria:

- It is smaller than a specified minimum size.
- It is larger than a specified maximum size.
- It has a skewness that is greater than a specified maximum skewness.

In addition to remeshing the volume mesh, FLUENT also allows triangular and linear faces on a deforming boundary to be remeshed. FLUENT marks deforming boundary faces for remeshing based on moving and deforming loops of faces. FLUENT requires that these loops are closed.

FLUENT automatically extracts loops on the boundary of the face zone whose nodes are moving or deforming. Consider a simple tetrahedral mesh of a cylinder whose bottom wall is moving (see Figure 4.10). On the deforming boundary, a single loop is generated at the bottom end of the cylinder (where the nodes are moving). In a similar approach as in the dynamic layering technique, FLUENT analyzes the height of the faces connected to the nodes on the loop and subsequently, splits or merges the faces depending on the specified ideal face height and split/merge factor.

If the faces in layer $j$ are expanding, they are allowed to expand until

![Figure 4-10. Remeshing at a Deforming Boundary](image)
\[ h > (1 + \alpha_h) h_{\text{ideal}} \] (4.12)

where \( h_{\text{ideal}} \) is the ideal face height, and \( \alpha_h \) is a height factor. When this condition is met, the faces are split according to the predefined face height such that the new faces on layer \( i \) have exactly the face height \( h_{\text{ideal}} \). Conversely, if the layer is contracting, they are allowed to contract until

\[ h < \alpha_h h_{\text{ideal}} \] (4.13)

When this condition is met, the compressed layer of faces is merged into the layer of faces above it. The face remeshing is illustrated in Figure 4.11.

**Applicability of the Local Remeshing Method**

We can use the local remeshing method only in cell zones that contain tetrahedral or triangular cells.

If we define deforming face zones in our model and we use local remeshing in the adjacent cell zone, the faces on the deforming face zone can be remeshed only if the following conditions are met:

- The faces are triangular (or linear in 2-D).
- The faces to be remeshed are all adjacent to moving loops (i.e., moving nodes).
- The faces are on the same face zone, and form an annular (i.e., closed loop).
- The faces are not part of a symmetry or conformal periodic boundary.
Volume mesh update procedure

The volume mesh is updated automatically. FLUENT decides which method to use for a particular zone based on the shape of the cells in the zone. For example, if the
boundaries of a tetrahedral cell zone are moving, the spring-based smoothing and local remeshing methods will be used to update the volume mesh in this zone. If the zone consists of prismatic (hexahedral and/or wedge) cells, then the dynamic layer method will be used to determine where and when to insert and remove cell layers.

FLUENT automatically determines which method to use by visiting the adjacent cell zones and setting appropriate flags for the volume mesh update methods to be used. If we specify the motion for a cell zone, FLUENT will visit all of the neighboring cell zones and set the flags appropriately. If we specify the motion of a boundary zone, FLUENT will analyze only the adjacent cell zones. If a cell zone does not have any moving boundaries, then no volume mesh update method will be applied to the zone.

Note that as a result of the local remeshing procedures, updated meshes may be slightly different when dynamic meshes are used in parallel FLUENT, and local remeshing is selected, and therefore very small differences may arise in the solutions.

4.3. Solid-body kinematics

FLUENT uses solid-body kinematics if the motion is prescribed based on the position and orientation of the center of gravity of a moving object. This is applicable to both cell and face zones.

The motion of the solid-body can be specified by the linear and angular velocity of the center of gravity. FLUENT allows the velocities to be specified either as profiles or user defined functions (UDF). FLUENT assumes that the motion is specified in the inertial coordinate system.
If the motion is specified using a profile, the components of the velocities must be described using the following profile fields:

- linear velocity \((v_x, v_y, v_z)\) as a function of time
- angular velocity \((\omega_x, \omega_y, \omega_z)\) as a function of time

In addition to the motion description, we must also specify the starting location of the center of gravity and orientation of the solid body. FLUENT automatically updates the center of gravity position and orientation at every time step such that

\[
\bar{x}_{c.g.}^{n+1} = \bar{x}_{c.g.}^n + \bar{\omega}_{c.g.} \Delta t
\]

\[
\bar{\theta}_{c.g.}^{n+1} = \bar{\theta}_{c.g.}^n + G\bar{\Omega}_{c.g.} \Delta t
\]

where \(\bar{x}_{c.g.}\) and \(\bar{\theta}_{c.g.}\) are the position and orientation of the center of gravity, \(\bar{\omega}_{c.g.}\) and \(\bar{\Omega}_{c.g.}\) are the linear and angular velocities of the center of gravity, and \(G\) is the transformation matrix that defines the choice of \(\bar{\theta}\). By default, \(G\) is taken to be the identity matrix.

Typically, \(\bar{\theta}\) is chosen to be an appropriate set of Euler angles. In this case, the solid-body motion must be specified using a user defined function (DEFINE CG MOTION) where the appropriate form of \(G\) can be specified.
The position vectors on the solid body are updated based on rotation about the instantaneous angular velocity vector $\hat{\Omega}_{c.g.}$. For a finite rotation angle $\Delta \theta = |\hat{\Omega}_{c.g.|} \Delta t$, the final position of a vector $\bar{x}_r$ on the solid body with respect to $\bar{x}_{c.g.}$ can be expressed as (See Figure 4.13).

$$\bar{x}_{r}^{n+1} = \bar{x}_{r}^{n} + \Delta \bar{x}$$

(4.16)

where $\Delta \bar{x}$ can be shown to be

$$\Delta \bar{x} = ||\bar{x}_{r}^{n}|| \left[ \sin(\Delta \theta) \hat{e}_{\theta} + (\cos(\Delta \theta) - 1) \hat{e}_{r} \right]$$

(4.17)

The unit vectors $\hat{e}_{\theta}$ and $\hat{e}_{r}$ are defined as

Figure 4-13. Solid Body Rotation Coordinates
\[ \hat{e}_\theta = \frac{\hat{\Omega}_{c_g} \times \vec{x}_r}{|\hat{\Omega}_{c_g} \times \vec{x}_r|} \]  
\[ \hat{e}_r = \frac{\hat{e}_\theta \times \hat{\Omega}_{c_g}}{|\hat{e}_\theta \times \hat{\Omega}_{c_g}|} \]  

If the solid body is also translating with \( \bar{\nu}_{c_g} \), the \( n+1 \) position vector on the solid body can be expressed as

\[ \vec{x}^{n+1} = \vec{x}^n_{c_g} + \bar{\nu}_{c_g} \Delta t + \vec{x}_r^{n+1} \]  

where \( \vec{x}_r^{n+1} \) is given by Equation 4.16.
CHAPTER 5

MESH ADAPTATION

5.1. Grid adaptation in FLUENT

The solution-adaptive mesh refinement feature of FLUENT allows us to refine and/or coarsen our grid based on geometric and numerical solution data. In addition, FLUENT provides tools for creating and viewing adaptation fields customized to particular applications.

Two significant advantages of the unstructured mesh capability in FLUENT are:

- The reduced setup time compared to structured grids
- The ability to incorporate solution-adaptive refinement of the mesh

By using solution-adaptive refinement, we can add cells where they are needed in the mesh, thus enabling the features of the flow field to be better resolved. When adaptation is used properly, the resulting mesh is optimal for the flow solution because the solution is used to determine where more cells are added. In other words, computational resources are not wasted by the inclusion of unnecessary cells, as typically occurs in the structured grid approach. Furthermore, the effect of mesh refinement on the solution can be studied without completely regenerating the mesh.

Any time we can perform mesh adaptation in a parallel computation, a load balancing step will be performed by FLUENT by default. We can turn off the automatic load
balancing by issuing the following command:

(disable-load-balance-after-adaptation)

To return to the default behavior, we can use the following command:

(enable-load-balance-after-adaptation)

But automatic load balancing will not occur in conjunction with dynamic adaptation.

5.2. Adaptation guidelines

The advantages of solution-adaptive refinement are significant. However, the capability must be used carefully to avoid certain pitfalls. Some guidelines for proper usage of solution-adaptive refinement are as follows:

- The surface mesh must be fine enough to adequately represent the important features of the geometry. For example, it would be bad practice to place too few nodes on the surface of a highly-curved airfoil, and then use solution refinement to add nodes on the surface. Clearly, the surface will always contain the facets contained in the initial mesh, regardless of the additional nodes introduced by refinement.

- The initial mesh should contain sufficient cells to capture the essential features of the flow field. Suppose, for example, that our intention is to predict the shock forming around a bluff body in supersonic flow. In addition to having sufficient surface resolution to represent the shape of the body, the initial mesh should also contain enough cells so that a reasonable first solution can be obtained. Subsequent gradient adaptation can be used to sharpen the shock and establish a grid-independent solution.

- A reasonably well-converged solution should be obtained before we perform an adaptation. If we adapt to an incorrect solution, cells will be added in the wrong region of
the flow. However, we must use careful judgment in deciding how well to converge the solution before adapting, because there is a trade-off between adapting too early to an unconverged solution and wasting time by continuing to iterate when the solution is not changing significantly. Note that this does not directly apply to dynamic adaptation, because here the solution is adapted either at every iteration or at every time-step, depending on which solver is being used.

- In general, we should write a case and data file before starting the adaptation process. Then, if we generate an undesirable mesh, we can restart the process with the saved files. Note that this does not directly apply to dynamic adaptation, because here the solution is adapted either at every iteration or at every time-step, depending on which solver is being used.

- When performing gradient adaptation, we must select suitable variables. For some flows, the choice is clear. For instance, adapting on gradients of pressure is a good criterion for refining in the region of shock waves. In most incompressible flows, however, it makes little sense to refine on pressure gradients. A more suitable parameter in an incompressible flow might be mean velocity gradients. If the flow feature of interest is a turbulent shear flow, it will be important to resolve the gradients of turbulent kinetic energy and turbulent energy dissipation, so these might be appropriate refinement variables. In reacting flows, temperature or concentration (or mole or mass fraction) of reacting species might be appropriate.

- Poor adaptation practice can have adverse effects. One of the most common mistakes is to over-refine a particular region of the solution domain, causing very large gradients in cell volume. This can adversely affect the accuracy of the solution.
5.3. The static adaptation process

The adaptation process has been separated into two distinct tasks. First, the individual cells are marked for refinement or coarsening based on the adaptation function, which is created from geometric and/or solution data. Next, the cell is refined or considered for coarsening based on these adaptation marks. The primary advantages of this modularized approach are the abilities to create sophisticated adaptation functions and to experiment with various adaptation functions without modifying the existing mesh.

It is highly recommended that we write a case and data file before starting the adaptation process. Then, if we generate an undesirable grid, our can restart the process with the saved files.

Two different types of adaptation are available in FLUENT: "conformal" and "hanging node" adaptation.

5.3.1. Adaptation and mask registers

Invoking the Mark command creates an adaptation register. It is called a register because it is used in a manner similar to the way memory registers are used in calculators. For example, one adaptation register holds the result of an operation, another register holds the results of a second operation, and these registers can be used to produce a third register. An adaptation register is basically a list of identifiers for each cell in the domain. The identifiers designate whether a cell is neutral (not marked), marked for refinement, or marked for coarsening. The adaptation function is used to set the appropriate identifier. For example, to refine the cells based on pressure gradient, the solver computes the gradient adaptation function for each cell. The cell value is compared
to the refining and coarsening threshold values and assigned the appropriate identifier, specifically for this example:

- cell value < coarsen threshold: mark for coarsening
- coarsen threshold < cell value < refine threshold: don't mark, neutral
- cell value > refine threshold: mark for refinement

The GUI and text interface commands generate adaptation registers that designate the cells marked for refinement or coarsening. These registers can be converted to mask registers. Masks, unlike the adaptation registers, maintain only two states: ACTIVE and INACTIVE. If the adaptation register is converted to a mask, cells marked for refinement become ACTIVE cells, while those that are unmarked or marked for coarsening become INACTIVE. We can use a mask register to limit adaptation to cells within a certain region. This process is illustrated below.

Figure 5.1 shows a cloud of cells representing an adaptation register (shaded cells are marked cells). Figure 5.2 illustrates the active cells associated with a mask register. If the mask is applied to (combined with) the adaptation register, the new adaptation register formed from the combination has the marked cells shown in Figure 5.3. (Note that this example does not differentiate between refinement or coarsening marks because the mask is applied to both types of marks.)
Figure 5-1. Adaptation Register with Shaded Cells Representing Marked Cells

Figure 5-2. Mask Register with Shaded Cells Representing Active Cells

Figure 5-3. New Adaptation Register Created from Application of Mask
In summary, adaptation registers can be created using geometric data, physical features of the flow field, and combinations of this information. Once created, adaptation registers can be listed, displayed, deleted, combined, exchanged, inverted, and changed to mask registers.

5.3.2. Hanging node adaptation

Grids produced by the hanging node adaptation procedure are characterized by nodes on edges and faces that are not vertices of all the cells sharing those edges or faces, as shown in Figure 5.4. Hanging node grid adaptation provides the ability to operate on grids with a variety of cell shapes, including hybrid grids. However, although the hanging node scheme provides significant grid flexibility, it does require additional memory to maintain the grid hierarchy which is used by the rendering and grid adaptation operations.

Figure 5-4. Example of a Hanging Node
5.3.2.1. Hanging node refinement

The cells are refined by isotropically subdividing each cell marked for refinement. Figures 5.5 and 5.6 illustrate the division of the supported cell shapes described below:

- A triangle is split into 4 triangles.
- A quadrilateral is split into 4 quadrilaterals.
- A tetrahedron is split into eight tetrahedra. The subdivision consists of trimming each corner of the tetrahedron, and then subdividing the enclosed octahedron by introducing the shortest diagonal.
- A hexahedron is split into 8 hexahedra.
- A wedge (prism) is split into 8 wedges.
- A pyramid is split into 6 pyramids and 4 tetrahedra.

To maintain accuracy, neighboring cells are not allowed to differ by more than one level of refinement. This prevents the adaptation from producing excessive cell volume variations (reducing truncation error) and ensures that the positions of the "parent" (original) and "child" (refined) cell centroids are similar (reducing errors in the flux evaluations).

Figure 5-5. Hanging Node Adaptation of 2-D Cell Types
5.3.2.2. Hanging node coarsening

The mesh is coarsened by reintroducing inactive parent cells, i.e., coalescing the child cells to reclaim the previously subdivided parent cell. An inactive parent cell is reactivated if all its children are marked for coarsening. We will eventually reclaim the original grid with repeated application of the hanging node coarsening. We cannot coarsen the grid any further than the original grid using the hanging node adaptation process. Conformal coarsening, however, allows us to remove original grid points to reduce the density of the grid.

5.4. Gradient adaptation

The gradient adaptation function allows us to mark cells or adapt the grid based on the gradient (or curvature) of the selected field variables.
The primary goal of solution-adaptive grid refinement is to efficiently reduce the numerical error in the digital solution, with minimal numerical cost. Unfortunately, direct error estimation for point-insertion adaptation schemes is difficult because of the complexity of accurately estimating and modeling the error in the adapted grids. In fact, no comprehensive mathematically rigorous theory for error estimation and convergence is available yet for CFD simulations. Assuming the greatest error occurs in high-gradient regions, the readily available physical features of the evolving flow field may be used to drive the grid adaptation process. Two approaches for using this information for grid adaptation are available in FLUENT:

- The first gradient approach is recommended for problems with strong shocks, e.g. supersonic inviscid flows. In this approach, FLUENT multiplies the undivided Euclidean norm of the gradient of the selected solution variable by a characteristic length scale. For example, the gradient function in two dimensions has the following form:

$$|e_{ij}| = \left( A_{cell} \right)^{r/2} |\nabla f|$$

(5.5)

where $e_{ij}$ is the error indicator, $A_{cell}$ is the cell area, $r$ is the gradient volume weight, and $\nabla f$ is the undivided Euclidean norm of the gradient of the desired field variable, $f$ (the Euclidean norm of $f$ is the intuitive notion of length of the vector $f=(f_1, f_2, ..., f_n)$ which is captured by the formula $\| f \| = \sqrt{|f_1|^2 + |f_2|^2 + ... + |f_n|^2}$). The default value of the gradient volume weight is unity, which corresponds to full volume weighting; a value of zero will eliminate the volume weighting, and values between 0 and 1 will use proportional weighting of the volume.

- The second gradient (curvature) approach is recommended for problems with smooth solutions. This is the equidistribution adaptation technique formerly used by
FLUENT, that multiplies the undivided Laplacian of the selected solution variable by a characteristic length scale. For example, the gradient function in two dimensions has the following form:

\[ |e_{i2}| = (A_{cell})^\frac{1}{2} |\nabla^2 f| \]  

(5.6)

where \( e_{i2} \) is the error indicator, \( A_{cell} \) is the cell area, \( r \) is the gradient volume weight, and \( \nabla^2 f \) is the undivided Laplacian of the desired field variable, \( f \). The default value of the gradient volume weight is unity, which corresponds to full volume weighting; a value of zero will eliminate the volume weighting, and values between 0 and 1 will use proportional weighting of the volume.

The length scale is the square (2-D) or cube (3-D) root of the cell volume. The introduction of this length scale permits resolution of both strong and weak disturbances, increasing the potential for more accurate solutions. We can, however, reduce or eliminate the volume weighting by changing the gradient Volume Weight in the Grid Adaptation Controls panel.

Any of the field variables available for contouring can be used in the gradient adaptation function. Interestingly, these scalar functions include both geometric and physical features of the numerical solution. Therefore, in addition to traditional adaptation to physical features, such as the velocity, we may choose to adapt to the cell volume field to reduce rapid variations in cell volume.

In addition to the Standard (no normalization) approach formerly used by FLUENT, two options are available for Normalization:

- Scale, which scales the values of \( e_1 \) or \( e_2 \) by their average value in the domain, i.e.:
when using the Scale option, suitable first-cut values for the Coarsen Threshold and the Refine Threshold are 0.3 to 0.5, and 0.7 to 0.9, respectively. Smaller values will result in larger adapted regions.

- Normalize, which scales the values of $e_{i1}$ or $e_{i2}$ by their maximum value in the domain, therefore always returning a problem-independent range of $[0, 1]$ for any variable used for adaptation, i.e.:

$$\frac{|e_i|}{\text{average}|e_i|}$$ (5.7)

when using the Normalize option, suitable first-cut values for the Coarsen Threshold and the Refine Threshold are 0.2 to 0.4, and 0.5 to 0.9, respectively. Smaller values will result in larger adapted regions.

5.5. Dynamic gradient adaptation

In contrast with the static gradient adaptation described before, where the adaptation of the mesh is performed "by hand", dynamic gradient adaptation is a fully automated process. For time dependent as well as for steady state problems, we can start the solution process without changing the initial settings, i.e. without stopping the iteration process, and setting new refine/coarsen threshold values, thus performing the mesh adaptation "by hand."

The dynamic gradient adaptation executes the gradient adaptation which was described before automatically. All options of gradient adaptation are also valid for the
dynamic gradient adaptation, but specific settings are recommended for the dynamic
gradient adaptation:

- The Refine as well as the Coarsen options should be switched on.
- Either the Scale or the Normalize option for Normalization should be used. The
  non-normalized values of the gradient or the curvature of a variable (obtained by
  selecting Standard for the Normalization) are generally strongly solution-dependent,
  and therefore would require re-adjustment of the Coarsen Threshold and Refine
  Threshold as the solution proceeds. For dynamic adaptation, scaling is usually preferred
  if we wish to resolve regions of minor values of the gradient (or curvature) accurately, in
  addition to the region of highest gradient (or curvature), because it does not take very
  high values of the gradient or curvature as much into account as does the normalization.
  Recommended refine and coarsen threshold values to start with are 0.4, and 0.9,
  respectively. In general the more refinement we want, the smaller these values should be.
- Hanging node adaptation only should be used.
- The Min # of Cells and Max # of Cells and the Max Level of Refine or the Min
  Cell Volume should be set. The limits for the Min # of Cells and Max # of Cells can
  aspect the Coarsen Threshold and Refine Threshold: if either the Min # of Cells or the
  Max # of Cells are violated, the Coarsen Threshold or the Refine Threshold are
  adjusted to fulfill the limits for the Min # of Cells or the Max # of Cells. For the Max
  Level of Refine, the default value of 2 is a good start for most problems. If this is not
  sufficient, we can increase this value, but keep in mind that even in a 2D problem, the
  default value of 2 can increase the number of cells by a factor of 16, in the adapted
regions. A value of zero leaves this parameter unbounded: in this case we should use a suitable limit for **Min Cell Volume**.

- The **Interval** between two consecutive automatic mesh adaptations must be specified. Depending on whether we are performing a steady-state or a time-dependent solution, the **Interval** is specified in iterations or time steps respectively. This value strongly depends on the problem solved and the time step used (where applicable): for (almost) steady state problems, values of 100 or higher are reasonable; for time dependent problems, values of 10 or lower are often required. Note that if we are using the coupled explicit solver with explicit unsteady formulation, our input here will be in number of iterations.

When the parallel solver is used, there is no automatic load balancing during the dynamic adaptation process. Thus we may want to use the execute command functionality to enforce repartitioning after a reasonable number of time steps or iterations. For example we could use the following commands:

- `para/part/method principal-axes`
- `para/part/use-stored`. 
CHAPTER 6

VALIDATION

Results from several test cases will now be presented. All computations were performed on a personal computer with an Intel Pentium 4 processor with 1GB of RAM.

6.1. Two-dimensional flat flow

Figure 6.1 shows the computational domain of 2-D flat case. Such geometry was chosen because of simplicity to compare calculation results with theoretical data (the most of theoretical data are presented in literature for flat (not axisymmetric) case).

Figure 6-1. Computational domain

Boundary conditions on inlet:

\( M_{in} = 2.28 \), then
\[ P_{in} = 101325 \left[ 1 + \frac{1.4 - 1}{2} \cdot 2.28^2 \right]^{\frac{1.4}{1+1}} = 122795 \text{ Pa} \]

\[ T_{in} = 300 \left[ 1 + \frac{1.4 - 1}{2} \cdot 2.28^2 \right] = 611.904 \text{ K} \]

Boundary conditions on the outlet are not important, since the flow is supersonic flow.

Figure 6-2 shows the initial mesh for the geometry. Figure 6-3 shows the adapted mesh, which was created using gradient adaptation method (zooming of the adapted mesh is showed on Figure 6-4).

![Initial mesh](image)

Figure 6-2. Initial mesh (7175 nodes, 6960 cells)

![Adapted mesh](image)

Figure 6-3. Adapted mesh (15954 nodes, 13668 cells)
Figures 6-5 – 6-9 show results of compressible flow calculation for the domain. We can see some subsonic domains close from upper wall (behind of first shock wave and upper wall interaction) and behind backward step (see Figure 6-5).
Figure 6-6. Static pressure (Pa)

Figure 6-7. Static temperature (K)

Figure 6-8. Density (kg/m$^3$)

Figure 6-9. Molecular viscosity (kg/m·sec)
6.1.1. Direct shock wave

On the top of the figures 6-5 - 6-9 we can see the direct shock wave. The wave takes place because a boundary layer near the wall develops where the velocity drops from its freestream value to zero. This means that the flow adjacent to the wall is subsonic and cannot sustain the pressure discontinuities associated with the shock wave. In this case, a so-called "Mach reflection" occurs, as shown in Figure 6-10. Here, a curved strong shock, behind which the flow is subsonic, forms near the wall. The flow behind the curved wall shock is divided from the flow behind the "reflected" oblique shock by the slipline across which there are changes in velocity, temperature and entropy.

![Figure 6-10. Schematic representation of a Mach reflection](image)

The solutions for the direct shock wave with theoretical data are compared. The line for comparison is located 0.2 m downstream from the wall (see Figure 6-11).
Figure 6-11. Location of line for comparison

Figure 6-12 shows the Mach number distribution along the line for comparison. We can see very strong direct shock wave (close from x=3 m) and two oblique shock waves (both of them are reflected shocks). The Mach number before the direct shock wave ($M_1$) equals 2.28. The Mach number behind the shock wave ($M_2$) equals 0.56. Theoretically (according to Oosthuizen and Carscallen (1997)) the Mach number behind the shock (for $M_1$=2.28) should be equal 0.537. Then the difference between theoretical $M_2$ and $M_2$, which is obtained in current calculation, is 4.28%.

Figure 6-12. Mach number distribution along the line for comparison.
Figure 6-13 shows static pressure distribution along the line for comparison. The theoretical ratio of the static pressure before the direct shock wave to the static pressure behind the direct shock wave \( \frac{p_2}{p_1} \) equals 5.898 (according to Oosthuizen and Carscallen (1997)). As result of current calculations the ratio is 6.02. Then the difference between theoretical \( \frac{p_2}{p_1} \) and \( \frac{p_2}{p_1} \), which obtained in current calculation, is 2.06%.

Figure 6-13. Static pressure distribution along the line for comparison.

Figure 6-14 shows the static temperature distribution along the line for comparison. The theoretical ratio of static temperature before the direct shock wave to the static temperature behind the direct shock wave \( \frac{T_2}{T_1} \) equals 1.929 (according to Oosthuizen and Carscallen (1997)). As result of current calculations the ratio is 1.93. Then the
difference between theoretical \(\frac{T_2}{T_1}\) and \(\frac{T_2}{T_1}\), which is obtained in current calculation, is 0.05%.

![Static Temperature Distribution](image)

Figure 6-14. Static temperature distribution along the line for comparison.

Figure 6-15 shows the density distribution along the line for comparison. The theoretical ratio of the density before the direct shock wave to the density behind the direct shock wave \(\frac{\rho_2}{\rho_1}\) equals 3.058 (according to Oosthuizen and Carscallen (1997)). As result of current calculations the ratio is 3.08. Then the difference between theoretical \(\frac{\rho_2}{\rho_1}\) and \(\frac{\rho_2}{\rho_1}\), which is obtained in current calculation, is 0.72%.
6.1.2. Oblique shock wave

The solutions for the oblique shock wave with theoretical data are compared.

Theoretically, the corner for an oblique shock wave (see Figure 6-16) are described by the following equations:
\[
\tan \alpha = \frac{w_{2n}}{w_{2m}} = \frac{w_{2n}}{w_{1n}} = \frac{w_{2n}}{w_{1n}} = \frac{1}{\lambda_{1m}^2}
\]

then

\[
\tan \beta = \frac{1}{\lambda_{1m}^2} \tan \alpha
\]

\[
\lambda^2 = \frac{k+1}{2} \left( \frac{1}{2} - \frac{k-1}{k+1} \right)
\]

\[
\tan \beta = \frac{k-1}{k+1} \left( 1 + \frac{2}{k-1} \frac{1}{M_1^2 \sin^2 \alpha} \right) \tan \alpha
\]

\[
\omega = \alpha - \beta
\]

The results, obtained from Eq. 6.1-6.5 are well known and appear as charts in various text books, for example in Oosthuizen and Carscallen (1997). Oosthuizen and Carscallen show an oblique shock wave chart for \( \gamma = 1.4 \) (air). The chart was created using NACA Report 1135 (1953). According Eq. 6.1-6.5 for our case \( \alpha_{tor} = 42.15° \). The result of the current calculations: \( \alpha = 42° \). Then the difference between theoretical angle and angle, which obtained in current calculation, is 0.36%.

Figure 6-17. Corner for oblique shock.
Figure 6-18. Location of line for comparison.

For other comparisons was used results along line locates on 1m down from wall (see Figure 6-18).

Figure 6-19 shows the static pressure distribution along the line for comparison. The line crosses the several oblique shock waves. The theoretical ratio of the static pressure before the first oblique shock wave to the static pressure behind the shock wave \( \frac{p_2}{p_1} \) equals 2.525 (according to Oosthuizen and Carscallen (1997)) (Eq. 6.6). As result of current calculations the ratio is 2.6. Then the difference between theoretical \( \frac{p_2}{p_1} \) and \( \frac{p_2}{p_1} \), which obtained in current calculation, is 2.97%.

\[
\frac{p_2}{p_1} = 1 + \frac{2k}{k+1} \left( M_i^2 \sin^2 \alpha - 1 \right)
\]  

(6.6)
The theoretical ratio of the static temperature before the first oblique shock wave to the static temperature behind the shock wave \( \frac{T_2}{T_1} \) equals 1.33 (according to Oosthuizen and Carscallen (1997)) (Eq. 6.7). As result of current calculations the ratio is 1.353. Then the difference between theoretical \( \frac{T_2}{T_1} \) and \( \frac{T_2}{T_1} \), which obtained in current calculation, is 1.73%.

\[
\frac{T_2}{T_1} = \frac{2 + (\gamma - 1)M_1^2 \sin^2 \alpha \left[ 2\gamma M_1^2 \sin^2 \alpha - (\gamma - 1) \right]}{(\gamma + 1)^2 M_1^2 \sin^2 \alpha}
\]  

(Eq. 6.7)
Figure 6-20. Static temperature distribution along the line for comparison.

Figure 6-21 shows the density distribution along the line for comparison. The theoretical ratio of the density behind the first oblique shock wave to the density before the shock wave \( \left( \frac{\rho_1}{\rho_2} \right) \) equals 0.526 (according to Oosthuizen and Carscallen (1997)) (Eq. 6.8). As result of current calculations the ratio is 0.519. Then the difference between the theoretical \( \frac{\rho_2}{\rho_1} \) and \( \frac{\rho_2}{\rho_1} \), which obtained in current calculation, is 1.33%.

\[
\frac{\rho_1}{\rho_2} = \frac{\tan(\alpha - \omega)}{\tan \alpha} \tag{6.8}
\]
6.2. Numerical solution of a turbulent supersonic flow over a backward facing step

The flow over a backward facing step is a classic problem in applied aerodynamics. Among many other applications, backward facing steps are often used for ignition and stabilization of the flame in a scramjet engine.

The backward facing step compressible flow regime includes flow separation, reattachment and viscous-inviscid interactions as shown in Figure 6-22. A uniform viscous flow with turbulent boundary layer on the flat surface approaches the step corner. In experimental observations, it is found that the flow separates slightly below the corner, and a lip shock is formed. The free stream above the corner undergoes an expansion resulting in a sharp pressure drop behind the step. The boundary layer which is separated behind the step develops into a free shear layer in a region of essentially constant pressure.

Figure 6-21. Density distribution along the line for comparison.
Figure 6-22. Supersonic flow over backward facing step

INLET
\[ M = M_{in} \]
\[ P = P_{in} \]
\[ p = p_{in} \]
\[ T = T_{in} \]

EXIT
\[ \frac{\partial \phi}{\partial x} = 0 \]

\[ H = 9.08h \]

\[ L_1 = 4.6h \]
\[ L_2 = 9.05h \]
As this shear layer approaches the wall downstream of the step it is compressed, and the low-velocity part of the shear layer reverses into the slowly circulating fluid within the base cavity. In the base region the pressure is low and nearly constant. Downstream of the reattachment, the high-velocity part of the shear layer overcomes the pressure rise at reattachment and forms a new uniform stream with the re-developed boundary layer. The external flow which turns beyond the step corner through the expansion fan, turns back to a direction approximately parallel to the inflow by the oblique reattachment shock wave.

The present results are validated by the comparison between the experimental results of planar laser-induced iodine fluorescence (PLIIF) measurements presented in the work of Hartfield et al. (1993). These conditions are:

\[M_in=2;\]
\[P_in=273 \text{ kPa};\]
\[p_n=34.8 \text{ kPa};\]
\[T_in=301 \text{ K};\]
\[h=3.18 \text{ mm}.\]

Figure 6-24 shows the initial mesh for the geometry. Figure 6-25 shows the adapted mesh, which was created using gradient adaptation method (zooming of the adapted mesh is showed on Figure 6-26).

Figure 6-27 shows the Mach number distribution into the calculation domain. We can see the oblique reattachment shock wave behind backward step very legibly. Figures 6.27 and 6-28 show comparison between current calculations and PLIIF measurements.
(Hartfield et al. 1993). We can see that the calculations compare very well to experimental results. Figures 6-30 and 6-31 show the density and molecular viscosity distributions into the calculation domain.

Figure 6-24. Initial mesh (9827 nodes, 9626 cells)

Figure 6-25. Adapted mesh (23343 nodes, 21113 cells)
Figure 6-26. Zooming of the adapted mesh.

Figure 6-27. Mach number
Figure 6-28. Static pressure (a-FLUENT calculations (p in Pa), b- PLIIF measurements (p in kPa))
Figure 6-29. Static temperature (a-FLUENT calculations, b- PLIIF measurements), K
Figure 6-30. Density (kg/m$^3$)

Figure 6-31. Molecular viscosity (kg/m·sec)
Figures 6-32 – 6.35 show pressure, static temperature, u- and v-components of velocity behind 10 mm from the backward step. The current calculation results are compared with PLIIF measurements (Hartfield et al. 1993), PHOENICS calculations (Halupovich Y., Natan B., Rom J, 1999) and Navier-Stokes calculations (Hartfield et al. 1993).

Figure 6-32. Pressure profile behind 10 mm from the backward step.
Figure 6-33. Static temperature profile behind 10 mm from the backward step.

Figure 6-34. u-component of velocity profile behind 10 mm from the backward step.
Figure 6-35. v-component of velocity behind 10 mm from the backward step.

We can see that the current calculations compare very well to both theoretical and experimental results.
CHAPTER 7

CALCULATION OF CHARACTERISTICS OF THE UNSTEADY COMPRESSIBLE GAS FLOW AROUND A PROJECTILE

7.1. Motionless projectile

Since the domain is axisymmetric, only the upper part of the domain is used for solving flow around a projectile (see Figure 7-1).

The initial mesh is shown in Figure 7-2. The mesh is uniform. The total number of nodes is 3955. A solution-adaptive grid refinement was used. The primary goal of the solution-adaptive grid refinement is to efficiently reduce the numerical error in the digital solution. The method of the refinement is h-adaptation. The equidistribution adaption technique used the undivided Laplacian of the selected solution variable by a characteristic length scale. Static pressure was used as the gradient adaption function for the problem. Examples of mesh adaptation are shown in Figure 7-3 and Figure 7-4. The number of levels for adaptation is 2.

![Figure 7-1. Calculation domain](image)
Velocity of the projectile varied from 0 to 1000 m/s.

The results of the calculations are represented in Figures 7-5, 7-6 and 7-7 depicting the flow at different velocities for the projectile (U=100 m/s, 180 m/s, 800 m/s and 1000 m/s respectively). The figures show the fields of velocity, pressure and density, as well as the appearance of shock waves inside the gun barrel at subsonic and supersonic speeds. We see that if velocity of projectile equals 100 m/s, there is no shock waves inside the gun barrel. At 180 m/s, two straight compression shocks develop. At 800-1000 m/s, a systems of oblique shocks occur.
Figure 7-5. Contours of Mach numbers

Figure 7-6. Contours of density (kg/m$^3$)
7.2. Moving projectile

The investigation of moving the projectile into the barrel was performed. Figure 7.8 shows computation domain at initial time. The geometry was created according to article of Nusca M.J. (1997). The boundary conditions for the projectile and the barrel are shown in Figure 7.9. Figure 7.10 shows projectile velocity versus time of flight.
Figure 7-9. Boundary conditions for projectile and barrel.

Figure 7-10. Projectile velocity vs time of flight.

Figure 7-11 shows the initial mesh for the geometry (zooming of the initial mesh is showed on Figure 7-12). In time of calculations the mesh was modified using remeshing procedures.
Figure 7-11. Initial mesh (2892 nodes, 5272 cells)

Figure 7-12. Fragment of initial mesh

Figure 7-13 shows the final mesh for the geometry (zooming of the final mesh is showed on Figure 7-12). Figure 7-15 shows the contours of static pressure for intermediate and final positions of the projectile. We can see the systems of oblique shock waves.

Figure 7-13. Final mesh (14426 nodes, 28431 cells)

Figure 7-14. Fragment of final mesh
Figures 7-16 – 7-20 show the contours of density, Mach numbers, static temperature, molecular and turbulent viscosities for intermediate and final positions of the projectile.

Figure 7.15. Contours of static pressure, Pa (a-intermediate position, b-final position)

Figure 7.16. Contours of density, kg/m³ (a-intermediate position, b-final position)

Figure 7.17. Contours of Mach numbers (a-intermediate position, b-final position)
Figure 7.18. Contours of static temperature, K (a-intermediate position, b-final position)

Figure 7.19. Contours of molecular viscosity, kg/m·s (a-intermediate position, b-final position)

Figure 7.20. Contours of turbulent viscosity, kg/m·s (a-intermediate position, b-final position)
7.3. Moving projectile to containment

The investigation of moving the projectile to the containment was performed. The purpose of the calculations is demonstration of FLUENT ability to calculate velocity of projectile as function of pressure gradient behind the projectile. Figure 7.21 shows the computation domain at initial time. The initial and boundary conditions are shown in Figure 7.22.

Figure 7-21. Computational domain
Newton's second law of motion is used to calculate velocity of projectile based on pressure distribution behind the projectile.

\[ F = m \dot{v} \]

\[ F = p \cdot S \]

\[ \dot{v} = \frac{v_{n+1} - v_n}{\Delta t} \]

\[ v_{n+1} = \frac{S \cdot p \cdot \Delta t}{m} \bigg|_n + v_n \]
where $F$ is force, $m$ is mass of projectile, $\dot{v}$ is projectile acceleration, $p$ is static pressure behind the projectile, $S$ is cross-sectional area of projectile back, $n$ is current time step, $n+1$ is next time step and $v$ is velocity of projectile.

Figure 7-23 shows the initial mesh for the geometry (zooming of the initial mesh is showed on Figure 7-24). In time of calculations the mesh was modified using remeshing procedures.

Figure 7-23. Initial mesh (15885 nodes, 30860 cells)

Figure 7-24. Fragment of initial mesh
Figure 7-25 shows the final mesh for the geometry (zooming of the final mesh is showed on Figure 7-26).

Figure 7-25. Final mesh (14426 nodes, 28431 cells)

Figure 7-26. Fragment of final mesh

Figures 7-16 – 7-20 show the contours of velocity magnitude and static temperature for intermediate and final positions of the projectile.
Figure 7.27. Contours of velocity magnitude, m/s (a-intermediate position, b-final position)

Figure 7.28. Contours of static temperature, K (a-intermediate position, b-final position)
CHAPTER 8

APPLICATION TO LIGHT GAS GUN DESIGN

The first light gas gun was developed due to the need to achieve high projectile velocities. It was determined that high muzzle velocities could be achieved if the column of conventional powder gas driving the projectile was replaced with a light-weight gas such as hydrogen (Crozier and Hume, 1957). Since then, single-stage, two-stage, and three-stage light gas guns have been used for hypervelocity impact studies (Schonberg and Cooper, 1994) and equation of state experiments (Nellis, et al., 1991). When impacted by a high-velocity projectile, strong shock waves are generated in a target specimen. Equation of state data for the target material can then be obtained using a method based on the Rankine-Hugoniot equations (Mitchell and Nellis, 1981).

The Joint Actinide Shock Physics Experimental Research (JASPER) facility utilizes a two-stage light gas gun to conduct equation of state experiments (Braddy, et al., 2001). Figure 8-1 illustrates the major components of the JASPER light gas gun. The pump tube is 11.5 meters long with a bore diameter of 89 mm and a piston mass of 4.5 kg. The launch tube is 8.1 meters long with a bore diameter of 28 mm. Hydrogen is used to propel projectiles with a mass range of 16.5 g to 26.5 g to a velocity of 7.4 km/s. The projectiles are cylindrical in shape, with a diameter of approximately 28 mm and a length of 25.4 mm.
An explosive charge is loaded into the breech behind the piston. The remaining section of the pump tube in front of the piston is filled with hydrogen gas. When a shot is fired, the explosive charge is detonated, sending the piston down the pump tube, compressing the hydrogen. When the hydrogen reaches a pressure of approximately 400 bar (Mespoulet, 2001), a petal valve separating the pump tube and launch tube ruptures, allowing the compressed hydrogen to propel the projectile down the launch tube toward the target.

Due to the hazardous nature of the experiments, the target is placed within the primary target chamber. The primary target chamber is equipped with an explosively driven ultra fast closure valve to contain any debris resulting from the projectile impacting with the target. The primary target chamber is placed inside the secondary containment chamber, which is designed to contain hydrogen deflagration and provide containment should the primary target chamber fail. Before the shot, a vacuum is pulled on the secondary containment chamber and launch tube.

From the muzzle exit to the entrance of the primary target chamber the projectile encounters a free flight zone approximately 1 meter in length. What the projectile does in this free flight zone is of particular interest in this study. Ideally, the projectile should
impact the target with no tilt in the axial direction, ensuring that the shock propagates through the target as uniformly as possible. This work is focused on determining what effects, if any, launch tube exit geometry changes have on attitude of the projectile in flight. Similar investigations were performed by DeBues (2002) for inviscid flow.

At the muzzle exit, a muzzle protector is attached to guard against debris. Depending on the configuration of the muzzle protector, the geometry of the muzzle may be altered. Two different configurations of this muzzle protector are under consideration (see Figure 8-2). The first case is standard muzzle geometry where the wall of the bore and the outer surface of the launch tube form a 90 degree angle. The second case includes a 26.6 degree bevel transition from the wall of the bore to the outer surface of the launch tube. For both cases, solutions are calculated for several positions downstream of the launch tube exit.

![Figure 8-2. Cross-section of muzzle exit showing attached protectors.](image)
8.1. Calculations of quasi-steady state flow

Quasi-steady state means that at each location the projectile is held fixed while the flow field is calculated. In other words, when performing a simulation for a particular location of the projectile, results for prior locations are not taken into account. Hence, the current study is focused on indicating if changes in projectile attitude might occur, and not with quantifying the actual changes in projectile attitude.

**Boundary Conditions**

The computational domains for both cases are illustrated in Figure 8-3. The exit plane of the muzzle is defined as \( x=0 \). The nodes at the inlet boundary are set according to the following Dirichlet conditions:

\[
M_{in} = 2.8
\]

\[
\frac{P_{in}}{P_{in}} = \left(1 + \frac{k-1}{2} M_{in}^2 \right)^{\frac{k}{k-1}}
\]

\[
T_{tot in} = 1 + \frac{k-1}{2} M_{in}^2
\]

On the surfaces of the projectile and muzzle

\[
u_{wall} = v_{wall} = w_{wall} = 0
\]
CASE 1. 90 degree angle at end of muzzle.

CASE 2. 26.6 degree angle at end of muzzle.

Figure 8-3. Axisymmetric representation of launch tube exit geometries.
The initial coarse mesh used for all cases is comprised of elements that are 1 mm square, or close to 1 mm square, depending on geometry. All calculations were conducted on the coarse mesh with 2 levels of $h$-adaptation (see Figures 8-4 and 8-5).

Pressure contours around the projectile located at $x = 4, 16, 32, 48,$ and $64$ mm are presented for both cases in Figures 8-6 through 8-15 (for the current calculations the pressure contours are given in Pa). The difference between the two cases is more evident closer to the launch tube. When comparing the two cases further away from the muzzle, there is little difference in the pressure contours behind the projectile. It would appear that the effects of the fluid expansion out of the muzzle are most prominent within approximately 35 mm of exit.

When comparing the pressure contours at each location from case to case, it is evident that the flow fields are different. However, there is no information indicating that one flow field would have more influence than the other with regard to the attitude of the projectile.

Figure 8-4. Example of axisymmetric case 1 mesh with two levels of $h$-adaptation.
Figure 8-5. Example of axisymmetric case 2 mesh with two levels of $h$-adaptation.

![Diagram of axisymmetric mesh with pressure contours](image)

De Bues (2003) results

Current calculations using FLUENT

Figure 8-6. Pressure contours around projectile for case 1 at $x = 4$ mm.

![Diagram of pressure contours](image)

De Bues (2003) results

Current calculations using FLUENT

Figure 8-7. Pressure contours around projectile for case 2 at $x = 4$ mm.
Figure 8-8. Pressure contours around projectile for case 1 at \( x = 16 \) mm.

Figure 8-9. Pressure contours around projectile for case 2 at \( x = 16 \) mm.

Figure 8-10. Pressure contours around projectile for case 1 at \( x = 32 \) mm.
Figure 8-11. Pressure contours around projectile for case 2 at $x = 32$ mm.

Figure 8-12. Pressure contours around projectile for case 1 at $x = 48$ mm.

Figure 8-13. Pressure contours around projectile for case 2 at $x = 48$ mm.
Figure 8-14. Pressure contours around projectile for case 1 at x = 64 mm.

Figure 8-15. Pressure contours around projectile for case 2 at x = 64 mm.

We can summarize that pressure contours exhibit a good qualitative agreement with De Bues (2003) results for both cases. It is impossible to compare the current calculations results with De Bues (2003) results quantitatively because the De Bues (2003) have used nondimensional character for his results.

8.2. Unsteady flow

For calculating unsteady flow in JASPER, it is best to split thee process into 3 stages (Figure 8.16).
Stage 1 - moving piston B within pump tube.

Stage 2 - moving projectile D within launch tube.

Stage 3 - moving projectile D within containment chambers (G and F).

Stage 1 and stage 2 are calculated using a deforming mesh (dynamic layering method shown on Figure 4.6). A local remeshing method (as shown in Figure 4.12) is employed for stage 3.

8.2.1. Example of calculation processes within launch tube

To illustrate, flow within the launch tube is shown in Figures 8.17-8.22. Figure 8-17 shows the mesh for initial part of the geometry (zooming of the initial mesh is showed on Figure 7-24).
Figure 8-17. Mesh for initial part of launch tube.

Figures 8-18 – 8-19 show the contours of density and static temperature for intermediate position of the projectile.

Figure 8-18. Contours of density (intermediate position).
8.2.2. Example of calculation processes within target chamber

The investigation of moving the projectile within target chamber was performed. Figure 8.20 shows computation domain at initial time.

Figures 8-21 – 8-22 show the contours of static pressure and velocity magnitude for intermediate position of the projectile.
Figure 8-20. Mesh for target chamber.

Figure 8-21. Contours of static pressure, Pa (intermediate position).
Figure 8-22. Contours of velocity magnitude, m/s (intermediate position).
CHAPTER 9

CONCLUSIONS

Compressible flow simulations have been obtained using FLUENT. The calculations included the presence of nonisothermicity, turbulence, changes in the shape of the computational domain with time, etc.

Numerical solutions of several benchmark problems were presented, illustrating the model's ability to accurately capture shock waves and resolve viscous boundary layers. The calculations of flow behind the backward step show that the developed model is able to predict flow separation, reattachment of shocks, viscous-inviscid interactions, lip-shock and expansion fans. All of comparisons of the current calculations with theoretical and experimental results show good qualitative and quantitative agreement. The benchmarks results also illustrated the ability of the $h$-adaptive mesh refinement algorithm to increase solution accuracy.

The investigation of moving projectile into barrel was performed. It showed the good ability of FLUENT to calculate unsteady flow features while changing geometry. The mesh deforming procedures such as local remeshing method, spring-based smoothing method and dynamic layering method work very well.

The calculations of projectile velocity as function of pressure gradient behind the projectile were performed. The calculations demonstrate the good ability of developed
model to couple hydrodynamics calculations with solid mechanics calculations, using C++ subroutines.

The numerical modeling was used to simulate the flow field around a projectile as it exits the muzzle of the JASPER light-gas gun. Specifically it was used to investigate if a change in muzzle geometry would cause the projectile to tilt in the axial direction during free flight. A comparison between two launch tube exit geometries was made. The first case was a standard muzzle geometry, where the wall of the bore and the outer surface of the launch tube form a 90 degree angle. The second case included a 26.6 degree bevel transition from the wall of the bore to the outer surface of the launch tube.

Results showed that for both cases the flow field is irregular close to the muzzle exit and more uniform further downstream.
Using User-Defined Functions (UDFs) in FLUENT

The function is consigned for calculation of the projectile position into the barrel as
result of the pressure distribution behind the projectile.

```
#include <stdio.h>
#include "udf.h"

#if !RP_NODE
#define UDF_FILENAME "udf_loc_velo"

/* read current location and velocity from file */
static void
read_loc_velo_file (real *loc, real *velo)
{
    FILE *fp = fopen(UDF_FILENAME, "r");
    if (fp != NULL)
    {
        float read_loc, read_velo;
        fscanf(fp, "%e %e", &read_loc, &read_velo);
        fclose(fp);
        *loc = (real) read_loc;
        *velo = (real) read_velo;
    }
    else
    {
        *loc = 0.0;
        *velo = 0.0;
    }
}

/* write current location and velocity in file */
static void
```
write_loc_velo_file (real loc, real velo)
{
    FILE *fp = fopen(UDF_FILFNAMF, "w");

    if (fp != NULL)
    {
        fprintf(fp, "%e %e", loc, velo);
        fclose(fp);
    }
    else
        Message ("Warning: cannot write %s file", UDF_FILFNAMF);
}
#endif/* !RP_NODF */

DEFINE_ON_DEMAND(reset_velocity)
{
#if !RP_NODF
    real loc, velo;

    read_loc_velo_file (&loc, &velo);
    write_loc_velo_file (loc, 0.0);

    Message ("UDF reset_velocity called.");
#endif
}

DEFINE_CG_MOTION(valve, dt, cg_yel, cg_omega, time, dtime)
{
#if !RP_NODF
    Thread *t = DT_THREAD (dt);
    face_if;
    real force, loc;
#endif
    real velo;

    /* reset velocities */
    NV_S (cg_vel, =, 0.0);
    NV_S (cg_omega, =, 0.0);

    if (!Data_Valid_P ())
        return;

#if !RP_NODF
    /* compute force on projectile wall */
    force = 0.0;
    begin_f_loop (f, t)
real *AA;

AA = F_AREA_CACHE (f, t);
force += F_P (f, t) * AA[0];
end_f_loop (f, t)

#if RP_2D
if (rp_axi)
    force *= 2.0 * M_PI;
#endif

read_loc_velo_file (&loc, &velo);

/* add in spring force */
#define K_SPRING 150000

real init_disp = 0.4 * 0.0254;
real s_force = K_SPRING * (loc + init_disp);

force = force;

/* compute change in velocity */

real dv = dtime * force;

velo += dv;
loc += velo * dtime;

Message ("nUDF valve: time = %f, x_vel = %f, force = %f, loc(m) = %f\n", time, velo, force, loc);
write_loc_velo_file (loc, velo);
#endif /* !RP_NODE */

#if PARALLEL
host_to_node_real_l (velo);
#endif

cg_vel[0] = velo;
NOMENCLATURE

a  speed of sound
e  internal energy
E  total energy
F  force
k  turbulent kinetic energy
M_w molecular weight
P  total pressure
p  static pressure
Pr Prandtl number
R  universal gas constant
Re Reynolds number
S  Sutherland constant
t  time
T  static temperature
T_tot total temperature
u  x-component of velocity
u_i  x, y, z -components of velocity in tensor notation at i = 1,2,3
v  y-component of velocity
w  z-component of velocity
x_i  x, y, z-coordinates at i = 1,2,3
x  horizontal Cartesian coordinate
y  vertical Cartesian coordinate

Greek

z  lateral Cartesian coordinate
\mu  dynamic viscosity
\mu_t turbulent dynamic viscosity
\tau  viscous stress tensor
\nu  kinematic viscosity
\rho  density

Subscripts

e  effective
i, in  inlet
i,j,k  unit vectors in the x, y, and z directions
\begin{tabular}{|c|l|}
\hline
m & iteration number \\
\hline
n & time step \\
\hline
T & turbulent \\
\hline
tot & total \\
\hline
u,v,w & velocity components in the x,y,z directions \\
\hline
x,y,z & coordinate directions \\
\hline
1 & in front of shock \\
\hline
2 & behind shock \\
\hline
\end{tabular}
REFERENCES


Bruckner, A., Knowlen, C., and Hertzberg, A. Applications of the ram accelerator to hypervelocity aerothermodynamic testing, AIAA paper 92-3949, 1992.


Hartfield, R.J., Hollo, S.D., McDaniel, J.C. Planar measurement technique for compressible flows laser induced iodine fluorescence. AIAA J. 31 (3), 1993, 483-490


VITA

Graduate College
University of Nevada, Las Vegas

Valery Ivanovich Ponyavin

Local Address:
4224, Chatham Cir., Apt #4
Las Vegas, NV 89119

Degree:
Bachelor of Science
Aerospace Engineering, 1994
Kazan State Technical University, Kazan, Russia

Selected publications:


113

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.


Thesis Title:
Numerical Modeling of Unsteady Compressible Gas Flow Around a Projectile

Thesis Examination Committee:
Co-chairperson, Dr. Darrell W. Pepper, Ph. D.
Co-chairperson, Dr. Yitung Chen, Ph. D.
Committee Member, Dr. William Culbreth, Ph. D.
Committee Member, Dr. Hsuan-Tsung Hsieh, Ph. D.
Graduate College Representative, Dr. Jichun Li, Ph. D.