UNIVERSITY OF CALIFORNIA,
IRVINE

A Bio-Medical Model of the Human Head for Impact Simulation

THESIS

submitted in partial satisfaction of the requirements
for the degree of

MASTER OF SCIENCE

in Electrical Engineering

by

Yang Zhang

Thesis Committee:
Professor Frithjof Kruggel, Chair
Professor Ender Ayanoglu
Associate Professor Gultekin Gulsen

2015

Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my gratitude to my supervisor Dr. Frithjof Kruggel for the useful comments, remarks and engagement through the learning process of this master thesis. His guidance helped me in all the time of research and writing of this thesis. Furthermore I would like to thank my thesis comittee: Dr. Ayanoglu and Dr. Gulsen for their encouragement. I also appreciate my fellow lab mates: Fumitaro Masaki and Jerod Rasmussen, for the stimulating discussions.

Last but not least, I would like to thank my family for encouraging me to pursue my academic career to study here in the US.

Abstract of The Thesis

A Bio-Medical Model of the Human Head for Impact Simulation

By

Yang Zhang

Master of Engineering in Electrical Engineering

University of California, Irvine, 2015

Professor Frithjof Kruggel, Chair

The finite element method (FEM) is a way of solving partial differential equation on a spatially (and temporarily) discretized model. It is widely used in medical applications. In this research, we develop a tool to calculate the response to an impact on human head and brain. We use the finite element method to model the solid tissue as linear elastic materials and the fluid-filled (1) as impressible viscous fluid. The geometric (2) model is derived from Magnetic Resonance Imaging (MRI) data of a real subject. To solve the nonlinear dynamic problem, we use the Newton-Raphson method, and employ parallelization to speed up the computation. Example simulations demonstrate the validation of the model.

1 Introduction

1.1 Motivation

Traumatic brain injury, also known as intracranial injury, occurs when an external force injures the brain. Traumatic brain injury can be classified based on severity, mechanism (closed or penetrating head injury), or other features (e.g., occurring in a specific location or over a widespread area). Head injury usually refers to traumatic brain injury, but is a broader category because it can involve damage to structures other than the brain, such as the scalp and skull.

Traumatic brain injury is a major cause of death and disability worldwide, especially in children and young adults. Males sustain traumatic brain injuries more frequently than do females. Causes include falls, vehicle accidents, and violence. Prevention measures include use of technology to protect those suffering from automobile accidents, such as seat belts and sports or motorcycle helmets, as well as efforts to reduce the number of automobile accidents, such as safety education programs and enforcement of traffic laws.

Brain trauma can occur as a consequence of a focal impact upon the head, by a sudden acceleration/deceleration within the cranium or by a complex combination of both movement and sudden impact. In addition to the damage caused at the moment of injury, brain trauma causes secondary injury, a variety of events that take place in the minutes and days following the injury. These processes, which include alterations in cerebral blood flow and the pressure within the skull, contribute substantially to the damage from the initial injury.

Traumatic brain injury can cause a host of physical, cognitive, social, emotional, and behavioral effects, and outcome can range from complete recovery to permanent disability or death. The 20th century saw critical developments in diagnosis and treatment that decreased death rates and improved outcome. Some of the current imaging techniques used for diagnosis and treatment include CT scans computed tomography and MRIs magnetic resonance imaging. Depending on the injury, treatment required may be minimal or may include interventions such as medications, emergency surgery or surgery years later. Physical therapy, speech therapy, recreation therapy, occupational therapy and vision therapy may be employed for rehabilitation.

Considerable effort is made to minimize the consequences of impacts using passive (e.g., helmets, air bags) and active devices (e.g., collision warning systems). Comparatively, little is known how external mechanical forces actually impact the brain. A likely reason for this lack of knowledge is that controlled scientific impact experiments in humans are unethical, and results of feasible animal experiments are difficult to extrapolate to humans due to considerable differences in head and brain geometry.

This project focuses on the development of a bio-mechanical simulation system that allows the consequence of a mechanical impact onto human head. Given a large set of forward solutions, the forces leading to a specific frame pattern can be modeled. The model described in this work is based on individual data from Computation Tomography (CT) or Magnetic Resonance Imaging (MRI). Due to high computation cost, parallel computing is employed in order to achieve a scalable tool.

The key of the simulation is finite element (FE) analysis, which is a numerical technique for finding the approximate solution to partial differential equations that describe properties of a biomechanical system. The human head is a complicated system which is made up of several components, including skull, scalp, white matter, gray matter and cerebrospinal fluid (CSF). In contrast to many other generic FE simulation for human head, we develop a multi-physics model here: we use fluid mechanics to simulate the CSF, and structural mechanics simulation is used for solid components. We link these portions by a fluid structural interface.

1.2 Finite Element Method(FEM)

1.2.1 Introduction

The finite element method is a way of solving partial differential equation on a spatially (temporarily) discretized model. As we all know, partial differential equations are widely applied in application to describe the physical phenomena. However, solving partial differential equations is not a straight forward task in the engineering analysis. By discretization, the finite element method can give an approximate solution to the partial differential equations. In spatial domain, the object will be downsampled into small uniform cells which will show simple quantitative and physical properties and will be considered easily. In temporal domain, an act dependent on time will be sample by some discrete time points.

For solid mechanics, Hook's law is a powerful choice to establish partial derivative equations. And for fluid mechanics, Navier-Stokes equation can be modeled. Once PDE has been found, we need to divide the continuum region into elements, where a variety of elements shapes will be employed. Then the interpolation functions will be applied into these nodes. Usually, although not always, polynomial are selected as interpolation functions to make the field variables (displacement for solid, or velocity for fluid) easy to integrate or differentiate. According to the interpolation functions, the system equations can be constructed. Then, the partial differential equations become linear equations. There exist a lot of numerical method solving these equations.

This thesis introduces a biomechanical simulation (3) for the human head. More details of the finite element method will be given in the next chapters. Chapter 3 will give detailed derivations for solid FEM. Chapter 4 will give derivations for fluid FEM. In Chapter 5, these two techniques will be combined by fluid-structure interaction. Some numerical methods in this work will be introduced in Chapter 6. Finally, the simulation results will be shown in Chapter 7.

2 Preprocessing

Image acquisition is an important part of this project but not included in this thesis. For the completeness, the approach of image acquisition and segmentation will be briefly introduced.

An image representative of a human head can be obtained by Magnetic Resonance Imaging (MRI) or Computed Tomography CT. MRI is a proper choice for the intracranial tissue, and CT works better for the osseous tissue. Ideally both imaging methods combined to generate a dataset. However, CT implicates a relatively high radiation does make CT inapplicable for research purpose. So a proton density weighted MR dataset is applied. In proton density MR images, bones has a low signal intensity due to its water content. Therefore, bone can be easily segmented from soft tissues. And for segmentation of soft tissue, T1 weighted MR is used.

To obtain high image quality, several image processing methods have been developed. We will combine T1 weighted MR, T2 weighted MR, proton density weighted MR and CT to process the images.

For segmentation, some widely used strategies are shown in [3]. To segment this dataset, an

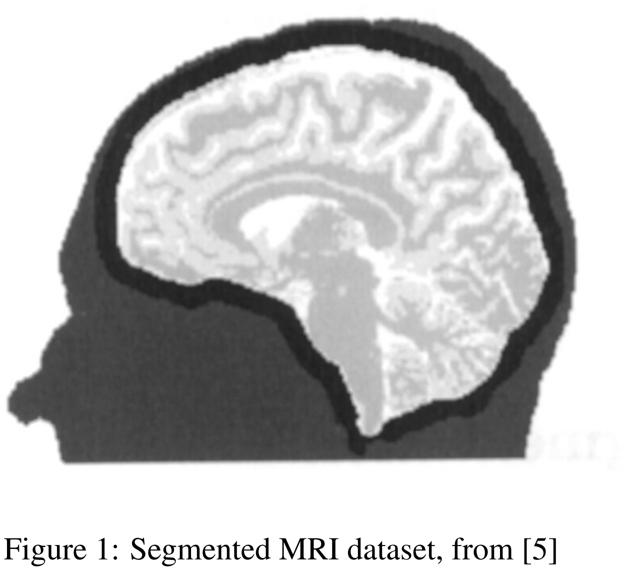
Figure 1: Segmented MRI dataset, from [5]

intensity based classification algorithm is applied. According to the number of tissue classes to be labeled, the center positions of classes are iteratively found to minimize the distance between centers [5]. As this process yields many objects which do not correspond to an anatomical structure, a connected component analysis is used.

Another segmentation approach is based on registration. There are two methods for registration [5]. One optimized for rigid registration of data from the same subject, including the ability to register images from different modalities. Another method is a full nonlinear registration algorithm for registering images from the same modality where one of the images is the reference images.

A successful segmentation will give a dataset with labeled objects, such as skull, white matter, gray matter, CSF and extracranial tissue, as shown in Fig. 1. In this dataset, the resolution of the segmented dataset is 1 mm isotropic. Note that this high spatial resolution is still not detailed enough to represent ().

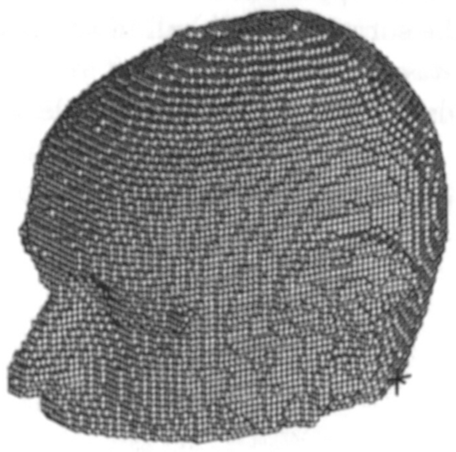

Figure 2: Isotropic mesh of a head with an edge length of 3mm, from [5]

Then a finite element description of the labeled objects has to be introduced by a mesh generation procedure. In this procedure, the graph is discretized. Each voxel of this graph can be converted to on of two kinds of elements: hexahedra and tetrahedra. The geometry determines which kind of elements will be applied. Fig. 2 shows the result after this procedure. One element has four nodes, if it is hexahedra, or eight nodes, if it is tetrahedra. And each node is shared by one or more elements.

However, the resolution of this graph usually results a mesh with large number of elements, and with a large number of nodes as well. In the Chapter 3 and Chapter 4, large number of nodes will lead to a large size of structure matrices. Then this will give the computation cost. Therefore, in this project, we need to downsample the image by combining several voxels in one element, and obtain a mesh with a lower resolution. By this way, the computation cost decreases and we can save a lot of time in simulation. But we will lost some detailed information by using this method. At the same time, all nodes and elements are numbered. In the rest part of this thesis, the finite element equation system is determined based on elements. Each element will form a matrix, and then the global structure matrix will be assembled according to the node number. The details will be given soon.

3 Solid FEM

As mentioned before, FEM is a numerical analysis technique for obtaining approximately solutions to a wide variety of engineering problems. In this project, the human head model consists of two parts, the solid part which contains skull, scalp, gray matter, white matter and other solid tissue, and the fluid part which has CSF. In this section, we will briefly review the mathematical derivation of the FEM of structure. And all of the structural finite element analysis is based on displacement. More details can be found in [1].

3.1 Model

For any object with mass m, when adding a force $r(t)$, according to Newton's law, we get

$$m\,\ddot{u}(t) = r(t) \tag{1}$$

here, $u(t)$ is the displacement of this object. If this object was linear elastic mechanical properties, usually considering damping force (F_d) which is proportional to the velocity , and the restoring force (F_s) which is proportional to the displacement, we get

$$m\,\ddot{u}(t) + F_d + F_s = r(t) \tag{2}$$

that is

$$m\,\ddot{u}(t) + c\,\dot{u}(t) + k\,u(t) = r(t) \tag{3}$$

Here, c is damping coefficient and k is called stiffness of this object. Eq. (3) is called the dynamic model of this object. As described in the previous section, if this object is spatially discretized, Eq. (3) becomes

$$\mathbf{M}\ddot{\mathbf{U}}(t) + \mathbf{D}\dot{\mathbf{U}}(t) + \mathbf{K}\mathbf{U}(t) = \mathbf{R}(t) \tag{4}$$

.

Eq. (4) is called a dynamic finite element equation ,and is widely used in finite element analysis. $\mathbf{M}$ is called the mass matrix, $\mathbf{D}$ is called the damping matrix, and $\mathbf{K}$ is called the stiffness matrix. Sometimes the static model needs to be considered, that means $R(t)$ is a static force which is independent of time, so Eq. (4) becomes

$$\mathbf{KU} = \mathbf{R} \tag{5}$$

which is called the static finite element equation. The second order derivative term and the first order derivative term will be removed. Please note that (4) is a nonlinear differential equation, which can be solved iteratively in temporal domain. So we can convert Eq. (4) from continuous-time system to discrete-time system. Then instead of solving differential equation Eq. (4), what we need to do is solve the linear equation Eq.(5). Therefore, formulating the stiffness matrix $\mathbf{K}$, mass matrix $\mathbf{M}$ damping matrix $\mathbf{D}$ and the right hand side $\mathbf{R}$ plays a key role. Next the derivation of this will be given.

3.2 Derivation

3.2.1 Principle of Virtual Work

For a displacement-based finite element formulation, the principle of virtual work can be employed. As shown in Fig.3, a three-dimensional body located in a fixed coordinate system X, Y, Z. Considering the body surface area, the body is supported in the area S_u with precribed displacement $\mathbf{U}^{S_u}$, and it is subjected to surface tractions $\mathbf{f}^{S_f}$ on the surface area S_f. At the same time, there exist externally applied body forces $\mathbf{f}^B$, and concentrated loads $\mathbf{F}^i$ (i denotes the point of load application). In this X, Y, Z coordinates,

$$\mathbf{f}^{S_f} = \begin{bmatrix} f_X^{S_f} \\ f_Y^{S_f} \\ f_Z^{S_f} \end{bmatrix} \tag{6}$$

$$\mathbf{f}^{B} = \begin{bmatrix} f_X^{B} \\ f_Y^{B} \\ f_Z^{B} \end{bmatrix} \tag{7}$$

$$\mathbf{F}^{i} = \begin{bmatrix} f_X^{i} \\ f_Y^{i} \\ f_Z^{i} \end{bmatrix} \tag{8}$$

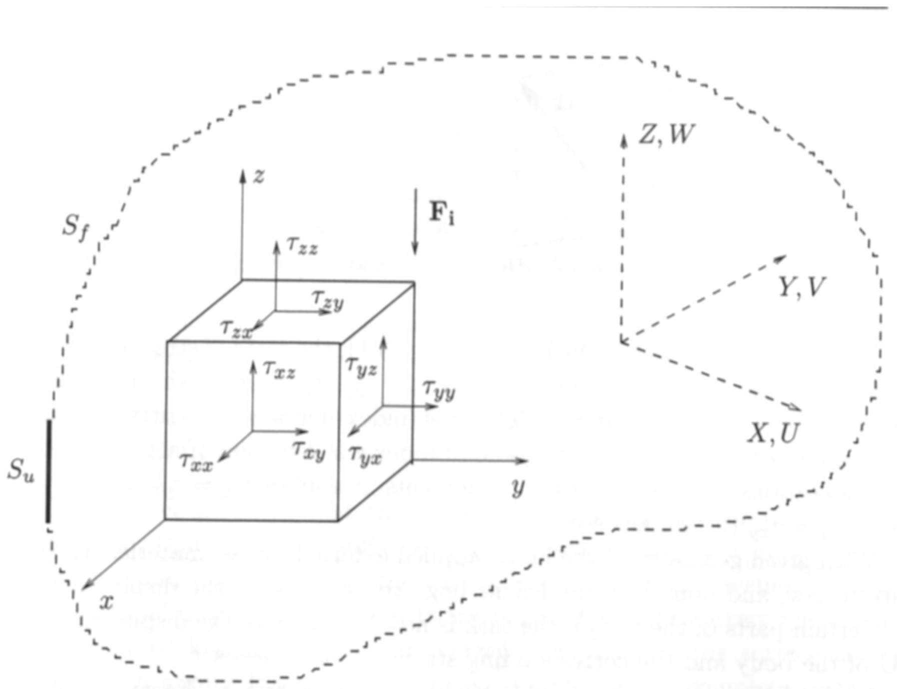

Figure 3: 3-dimensional body with a cubic element, from [5]

In the unloaded configuration, the displacement of the body can be viewed as

$$\mathbf{U}(X, Y, Z) = \begin{bmatrix} U \\ V \\ W \end{bmatrix} \tag{9}$$

and $\mathbf{U} = \mathbf{U}^{S_u}$ on the surface S_u.

If adjacent points in the point are displacement inhomogeneously, we need to consider the strains. And the entries of the strain tensor include partial derivatives of the displacements $\mathbf{U}$. It is

$$\epsilon = \begin{bmatrix} \epsilon_{XX} \\ \epsilon_{YY} \\ \epsilon_{ZZ} \\ \epsilon_{XY} \\ \epsilon_{YZ} \\ \epsilon_{ZX} \end{bmatrix} \tag{10}$$

where $\epsilon_{XX} = \frac{\partial \mathbf{U}}{\partial \mathbf{X}}$, $\epsilon_{YY} = \frac{\partial \mathbf{V}}{\partial \mathbf{Y}}$, $\epsilon_{ZZ} = \frac{\partial \mathbf{W}}{\partial \mathbf{Z}}$, $\epsilon_{XY} = \frac{\partial \mathbf{U}}{\partial \mathbf{X}} + \frac{\partial \mathbf{V}}{\partial \mathbf{Y}}$, $\epsilon_{XZ} = \frac{\partial \mathbf{U}}{\partial \mathbf{X}} + \frac{\partial \mathbf{W}}{\partial \mathbf{Z}}$, $\epsilon_{YZ} = \frac{\partial \mathbf{V}}{\partial \mathbf{Y}} + \frac{\partial \mathbf{W}}{\partial \mathbf{Z}}$. Here, ϵ_{XX}, ϵ_{YY}, and ϵ_{ZZ} are the extension in the direction of x, y, z of one element, and ϵ_{XY}, ϵ_{XZ} and ϵ_{YZ} are the strains in the shears, that is in the (x, y) plate, (x, z) plate and (y, z) plate.

The stresses corresponding to ϵ are

$$\tau = \begin{bmatrix} \tau_{XX} \\ \tau_{YY} \\ \tau_{ZZ} \\ \tau_{XY} \\ \tau_{XZ} \\ \tau_{YZ} \end{bmatrix} \tag{11}$$

There is a relationship between stresses and strains:

$$\tau = \mathbf{C}\epsilon \tag{12}$$

where **C** is the material matrix. This will be discussed in next sections. This is known as material stress-strain law.

Given geometry of this body, applied external forces, material stress-strain law, and boundary conditions, such as the constraints in certain parts of the body, we can calculate the displacements **U** as well as strains ϵ and stresses τ.

In this part, there are two assumptions required:

- The displacement is small enough and the equilibrium of the body can be established with respect to its unloaded configuration.
- The material matrix is constant, ie. , is dependent of the strain.

After this, we can employ the principle of virtual work or the principle of virtual displacement. This principle states that the equilibrium of the body requires that for any compatible small virtual displacements imposed on the body in its state of equilibrium, the total internal virtual work is equal to the total external virtual work.

$$\int_{\mathbf{V}} \bar{\epsilon}\tau d\mathbf{V} = \int_{\mathbf{V}} \bar{\mathbf{U}}^T \mathbf{f}^B d\mathbf{V} + \int_{S_f} \bar{\mathbf{U}}^{S_f^T} \mathbf{f}^{S_f} dS + \sum_i \bar{\mathbf{U}}^{i^T} \mathbf{F}^i \tag{13}$$

where $\bar{\mathbf{U}}$ is the virtual displacements and $\bar{\epsilon}$ is the corresponding strains. $\mathbf{f}^B$, $\mathbf{f}^{S_f}$ and $\mathbf{F}_i$ are the applied loads. The left hand side represents the inner virtual work, and the right hand side represents the outer virtual work. Here,

$$\bar{\mathbf{U}} = \begin{bmatrix} \bar{U} \\ \bar{V} \\ \bar{W} \end{bmatrix} \tag{14}$$

3.2.2 Static Finite Element Equations

In Eq. (14), the virtual displacements and virtual strains are totally independent from the actual displacement and actual strains. And we can use Eq. (14) to derive the governing finite element equations. After discretization, the body is made up of elements so that we can consider each element. For each element a local coordinate x, y, z (to be chosen conveniently) can be established, and the displacement at the N finite element nodal points can be modeled by this coordinate. Considering the element m, we have

$$\mathbf{u}^{(m)}(x, y, z) = \mathbf{H}^{(m)}(x, y, z)\,\hat{\mathbf{U}} \tag{15}$$

where $\mathbf{H}^{(m)}$ represents the displacement interpolation matrix, and $\hat{\mathbf{U}}$ is a vector of the three global displacement component U_i, V_i and U_i. So the dimension of $\hat{\mathbf{U}}$ is $3\,N$ such that

$$\hat{\mathbf{U}}^T = [U_1V_1W_1 \quad U_2V_2W_2 \quad \ldots \quad U_NV_NW_N] \tag{16}$$

Usually, in the linear elastic structural problem, the displacement interpolation matrix with size of 3 by $3\,N$ can be formulated as

$$\mathbf{H} = \begin{bmatrix} h_1 & 0 & 0 & h_2 & 0 & 0 & \ldots & h_N & 0 & 0 \\ 0 & h_1 & 0 & 0 & h_2 & 0 & \ldots & 0 & h_N & 0 \\ 0 & 0 & h_1 & 0 & 0 & h_2 & \ldots & 0 & 0 & h_N \end{bmatrix} \tag{17}$$

In Eq. (15), $\mathbf{u}^{(m)^T} = [u \quad v \quad w]$. For each entries,

$$u = \sum_{i=1}^{N} h_i U_i, \quad v = \sum_{i=1}^{N} h_i V_i, \quad w = \sum_{i=1}^{N} h_i W_i$$

It is important to note that the displacements on one node only affect the displacements and strains in this element. According to Eq. (15), the strains

$$\epsilon^{(m)} = \mathbf{B}^{(m)}\hat{\mathbf{U}} \tag{18}$$

where $B^{(m)}$ is called the strain-displacement matrix which is obtained by the differentiating of $H^{(m)}$. For the linear elastic problem,

$$\mathbf{B}^{(m)} = [\mathbf{b}_1 \quad \mathbf{b}_2 \quad \ldots \quad \mathbf{b}_N] \tag{19}$$

where

$$\mathbf{b}_i = \begin{bmatrix} \frac{\partial h_i}{\partial x} & 0 & 0 \\ 0 & \frac{\partial h_i}{\partial y} & 0 \\ 0 & 0 & \frac{\partial h_i}{\partial z} \\ \frac{\partial h_i}{\partial y} & \frac{\partial h_i}{\partial x} & 0 \\ 0 & \frac{\partial h_i}{\partial z} & \frac{\partial h_i}{\partial y} \\ \frac{\partial h_i}{\partial z} & 0 & \frac{\partial h_i}{\partial x} \end{bmatrix} \tag{20}$$

Please note that that coordinate in Eq. (15) is (u, v, m). However, in Eq. (18), the derivative is about (x, y, z). In order to calculate the derivative of the interpolation function s with with the local coordinates, we make use of the chain rule.

$$\begin{bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \\ \frac{\partial}{\partial w} \end{bmatrix} = \begin{bmatrix} \frac{\partial x}{\partial u} & \frac{\partial y}{\partial u} & \frac{\partial z}{\partial u} \\ \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} & \frac{\partial z}{\partial v} \\ \frac{\partial x}{\partial w} & \frac{\partial y}{\partial w} & \frac{\partial z}{\partial w} \end{bmatrix} \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix} \tag{21}$$

Note

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial u} & \frac{\partial y}{\partial u} & \frac{\partial z}{\partial u} \\ \frac{\partial x}{\partial v} & \frac{\partial y}{\partial v} & \frac{\partial z}{\partial v} \\ \frac{\partial x}{\partial w} & \frac{\partial y}{\partial w} & \frac{\partial z}{\partial w} \end{bmatrix}$$

so we can get

$$\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix} = \mathbf{J}^{-1} \begin{bmatrix} \frac{\partial}{\partial u} \\ \frac{\partial}{\partial v} \\ \frac{\partial}{\partial w} \end{bmatrix} \tag{22}$$

which is solvable. J is Jacobian matrix and its size is 3 by 3. So $\mathbf{J}^{-1}$ is simple to calculate. With Eq. (19), Eq. (20) and Eq. (22), B can be determined.

The stresses in the finite element m are related to the element stains using

$$\tau^{(m)} = \mathbf{C}^{(m)} \epsilon^{(m)} \tag{23}$$

as mentioned before, $\mathbf{C}^{(m)}$ is material matrix or stress-strain matrix, of element m. $\mathbf{C}^{(m)}$ is determined by the material properties of element m.

In general, when working with a 3 dimensional stress state, a forth order tensor containing 81 elastic coefficients must be defined to link the stress tensor and the strain tensor. Due to the symmetry of the stress tensor, strain tensor and elastic tensor, only 21 elastic components are independent. If the material is assumed to be isotropic (properties are independent of direction of space), its elasticity can be characterized by only two parameters E and ν;

$$\mathbf{C} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1 & \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ \frac{\nu}{1-\nu} & 1 & \frac{\nu}{1-\nu} & 0 & 0 & 0 \\ \frac{\nu}{1-\nu} & \frac{\nu}{1-\nu} & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1-2\nu}{2(1-\nu)} \end{bmatrix} \tag{24}$$

and E is Young's modulus, and ν is Poisson's ratio. They describe the material properties of this element. So $\mathbf{C}$ is a constant matrix within one element. Young's modulus is the ratio of stress, which has units of pressure. E can be determined from the slope of a stress-strain curve created during tensile tests conducted on a sample of the material. When a material is compressed in on direction, it usually tends to expand in the other two directions perpendicular to the direction of compression. The Poisson ratio is the ratio of the fraction of the expansion divided by the fraction of compression. Conversely, if the material is stretched rather than compressed, it usually tends to contract in the directions transverse to the direction of stretching. Poisson ratio will be ratio of relative contraction to relative stretching, and will have the same value as above. Most materials have ν ranging from 0 to 0.5. A perfectly incompressible material deformed elastically at small strains would have $\nu = 0.5$ exactly.

Now we complete one derivation within one element. At the same time, the principle of the virtual work can be expressed as the summation of the integrations.

$$\sum_m \int_{V^{(M)}} \bar{\epsilon}^T \tau d\mathbf{V}^{(m)} \\ = \sum_m \int_{V^{(M)}} \bar{\mathbf{U}}^{(m)^T} \mathbf{f}^{B^{(m)}} d\mathbf{V}^{(m)} + \sum_m \int_{S_f^{(m)}} \bar{\mathbf{U}}^{S_f^{(m)^T}} \mathbf{f}^{S_f^{(m)}} dS_f^{(m)} + \sum_i \bar{\mathbf{U}}^{i^T} \mathbf{F}^{\mathbf{i}} \quad (25)$$

if np is the total number of the elements, $m = 1, 2, \ldots, np$. i is the index of the points with concentrated loads $\mathbf{F}^i$. This is based on our assumption that concentrated loads act at nodal points.

We can substitute Eq. (25) with Eq. (18) and Eq. (23) and obtain

$$\bar{\hat{\mathbf{U}}}^T \left[\sum_m \int_{V^{(m)}} \mathbf{B}^{(m)^T} \mathbf{C}^{(m)} \mathbf{B}^{(m)} d\mathbf{V}^{(m)} \right] \hat{\mathbf{U}} \\ = \bar{\hat{\mathbf{U}}}^T \left[\sum_m \int_{V^{(m)}} \mathbf{H}^{(m)^T} \mathbf{f}^{B^{(m)}} d\mathbf{V}^{(m)} + \sum_m \int_{S_f^{(m)}} \mathbf{H}^{S_f^{(m)^T}} \mathbf{f}^{S_f^{(m)}} dS_f^{(m)} + \mathbf{F} \right] \quad (26)$$

$\mathbf{H}^{S^{(m)f}}$ is the surface displacement interpolation matrices, which can be obtained from $\mathbf{H}^{(m)}$ in Eq. (15) after replacing the corresponding element surface coordinates.

Next, we can define the element body forces

$$\mathbf{R}_B = \sum_m \int_{V^{(m)}} \mathbf{H}^{(m)^T} \mathbf{f}^{B^{(m)}} d\mathbf{V}^{(m)} \quad (27)$$

and the element surface forces

$$\mathbf{R}_{S_f} = \sum_m \int_{S_f^{(m)}} \mathbf{H}^{S_f^{(m)^T}} \mathbf{f}^{S_f^{(m)}} dS_f^{(m)} \tag{28}$$

and the nodal concentrated loads

$$\mathbf{R}_E = \mathbf{F} \tag{29}$$

we can add them together

$$\mathbf{R} = \mathbf{R}_B + \mathbf{R}_{S_f} + \mathbf{R}_E \tag{30}$$

$\mathbf{R}$ works as the right hand side of the static finite element equation. For the left hand side, we need to calculate the stiffness matrix $\mathbf{K}$

$$\mathbf{K} = \sum_m \mathbf{K}^{(m)} \tag{31}$$

here

$$\mathbf{K}^{(m)} = \int_{V^{(m)}} \mathbf{B}^{(m)^T} \mathbf{C}^{(m)} \mathbf{B}^{(m)} dV^{(m)} \tag{32}$$

From Eq. (26) to 32, we can obtain the static finite element equation

$$\mathbf{K}\hat{\mathbf{U}} = \mathbf{R}$$

after substituting $\hat{\mathbf{U}}$ to $\mathbf{U}$, we can obtain Eq. (5). However, for Eq. (31), assemblying of the stiffness matrix $\mathbf{K}$ is not as simple as adding the element stiffness matrix $\mathbf{K}^{(m)}$. For most objects, not all nodes are connected to each other, so $\mathbf{K}$ is sparse. In practice, the element stiffness matrix is first calculated as a dense matrix. Only the nonzero rows and columns of an element matrix $\mathbf{K}^{(m)}$ are obtained according to the degree of freedom. So for K, the position of the nonzero elements contains the information about the geometry of the structure and the value of them is determined by the material property. Next an example will be given.

Assuming that a body has nv nodes after meshing and $nv = 3$, and just considering 1 degree of freedom, the $\mathbf{K}$ has size of 3 by 3. And this body has $np = 2$ elements. Element 1 contains node 1, and 2. And Element 2 contains node 2 and 3.

For element 1, the stiffness matrix

$$\mathbf{K}^{(1)} = \begin{bmatrix} k^1_{1,1} & k^1_{1,2} \\ k^1_{2,1} & k^1_{2,2} \end{bmatrix}$$

For element 2, the stiffness matrix

$$\mathbf{K}^{(2)} = \begin{bmatrix} k^2_{1,1} & k^2_{1,2} \\ k^2_{2,1} & k^2_{2,2} \end{bmatrix}$$

In $\mathbf{K}^{(1)}$, $k^1_{1,1}$, $k^1_{1,2}$, $k^1_{2,1}$ and $k^1_{2,2}$ represent the connectivity between node 1 and 1, between node 1 and 2, between node 2 and 1, between node 2 and 2 respectively. What is more, $k^1_{1,2} = k^1_{2,1}$, because the connection between the same nodes is the same in the same element. In $\mathbf{K}^{(2)}$, $k^2_{1,1}$, $k^2_{1,2}$, $k^2_{2,1}$ and $k^2_{2,2}$ represent the connectivity between node 2 and 2, between node 2 and 3, between node 3 and 2, between node 3 and 3 respectively. So the global stiffness matrix is:

$$\mathbf{K} = \begin{bmatrix} k^1_{1,1} & k^2_{1,2} & 0 \\ k^2_{1,2} & k^1_{2,2} + k^2_{1,1} & k^2_{1,2} \\ 0 & k^2_{1,2} & k^2_{2,2} \end{bmatrix}$$

From this example, a simple way of assembly of stiffness matrix is given. Note that the global matrix $\mathbf{K}$ is symmetric. In the same way, the assemblage body force vectors $\mathbf{R}_B$ and $\mathbf{R}_{S_f}$ can be assembled.

3.2.3 Dynamic Finite Element Equations

Usually the load and the displacement is dependent of time, as described in $Eq.$(4). The displacements and stresses caused by applied forces that vary in time, and inertia forces is required, which will be regarded as an additional body forces. Interpolating the element accelerations, the body load vector $\mathbf{R}_B$ becomes

$$\mathbf{R}_B = \sum_m \int_{V^{(m)}} \mathbf{B}^{(m)^T} \left[\mathbf{f}^{B^{(m)}} - \rho^{(m)} \mathbf{H}^{(m)} \ddot{U} \right] d\mathbf{V}^{(m)} \tag{33}$$

where $\rho^{(m)}$ is the density of element m which is another material property. Considering this, Eq. (4) becomes

$$\mathbf{M}\ddot{\mathbf{U}}(t) + \mathbf{K}\mathbf{U}(t) = \mathbf{R}(t) \tag{34}$$

where

$$\mathbf{M} = \sum_m \mathbf{M}^{(m)} \tag{35}$$

and

$$\mathbf{M}^{(m)} = \int_{V^{(m)}} \rho^{(m)} \mathbf{H}^{(m)^T} \mathbf{H}^{(m)} d\mathbf{V}^{(m)} \tag{36}$$

here, $\mathbf{M}$ is called mass matrix. The assembly of the global mass matrix follows same way as the assembly of the stiffness matrix.

During vibration, the energy of the displacing body is dissipated. To consider this, the velocity-dependent damping forces need to be added into the body forces again. So Eq. (33) becomes

$$\mathbf{R}_B = \sum_m \int_{V^{(m)}} \mathbf{H}^{(m)^T} \left[\mathbf{f}^{B^{(m)}} - \rho^{(m)} \mathbf{H}^{(m)} \ddot{\mathbf{U}} - \kappa^{(m)} \mathbf{H}^{(m)} \dot{\mathbf{U}} \right] d\mathbf{V}^{(m)} \tag{37}$$

where $\kappa^{(m)}$ is the damping property parameter of the element m. Considering this, Eq. (34) becomes

$$\mathbf{M}\ddot{\mathbf{U}}(t) + \mathbf{D}\dot{\mathbf{U}}(t) + \mathbf{K}\mathbf{U}(t) = \mathbf{R}(t) \tag{38}$$

where

$$\mathbf{D} = \sum_m \mathbf{D}^{(m)} \tag{39}$$

and

$$\mathbf{D}^{(m)} = \int_{V^{(m)}} \kappa^{(m)} \mathbf{H}^{(m)^T} \mathbf{H}^{(m)} d\mathbf{V}^{(m)} \tag{40}$$

here, $\mathbf{D}$ is called damping matrix which can be obtained by the same way as the stiffness matrix and mass matrix. Sometimes, $\mathbf{D}$ is treated as linear combination of $\mathbf{M}$ and $\mathbf{K}$.

Now we have completed all derivation of structural finite element equations. And in this part, these equations are based on linear analysis. So we can consider $\mathbf{U}$ as a linear function of $\mathbf{R}$. Once large displacements occur, the previous assumptions does not hold, and the deformation of the finite elements have to be accounted. In Section 4, we will use Newton-Rapson method to process this problem.

3.3 Boundary Conditions

From the theory of differential equation, the solution are not uniquely determined by equation itself. To make a solution determinate it is necessary to specify certain supplemental conditions. For instance, to require that the unknown functions, and often also some of its derivatives, or certain combinations of the function and its derivatives.

Eq. (4) is a nonlinear dynamic equation. Two conditions are required to be specified. One is called initial condition which is that $\mathbf{U}(0)$ equals zeros. Another is called boundary condition which specifies some parts of the model. As mentioned before, for some specific part of the model, there are three kinds [23]:

1. Dirichlet boundary conditions give the value of the function.
2. Neumann boundary conditions give the normal derivative.
3. Robin boundary conditions are linearly combine Dirichlet boundary conditions and Robin boundary conditions.

With initial conditions and boundary conditions, Eq. (4) can be solved. In this project, we use Dirichlet boundary conditions. First, some nodes in the mesh will be fixed, and outer forces will be applied onto some nodes. For a fixed point, the corresponding rows and columns are filled with zeros except the diagonal entries which are set to be 1. For the outer forces, the corresponding elements of the load vector $\mathbf{R}$ (right hand side vector) are determined by the directions and amount of the forces. One thing should be noted is that boundary conditions should be determined properly. Wrong configuration will give unstable analysis results.

4 Fluid FEM

In this section, we will discuss the finite element analysis for the fluid part of human head. And the finite element method of fluid is based on velocity u and pressure p. Because fluid is not Hookean, we cannot use Hook's Law to model it. For given material of CSF, we will consider the fluid as the viscous flow, also called Stokes slow. In this thesis, we will establish a model by using Navier-Stokes equation. And it is important to note that the FEM of fluid is based on velocity and pressure, and is not based on displacement.

In the very beginning of this chapter, some notations should be given. First, $\mathbf{U}$ in previous chapter means the displacements of structure, but will be the velocity of fluid in this chapter. What's more, (u, v, w) represents the natural coordinate in Ch. 3. In this chapter, u, v, w are the velocity components in x, y, z directions respectively. More details about the derivation in this Chapter can be found in [4].

4.1 Navier-Stokes Equation

A fluid is a substance that continuous deforms under the action of applied surface stresses. Usually the fluid is classified as either inviscid or viscous. Invicid are frictionless flows characterized by zero viscosity. But no real flow is invicid. And viscous flow may further be classified as either Newtonian or Non-Newtonian, or either compressible or incompressible. A flow is Newtonian if it fits Navier-Stokes equation (which will be given soon). Whether the flow is compressible or incompressible depends on whether density variations are large or relatively unimportant. In this work, the flow is considered as Newtonian and incompressible so that it can be modeled by Navier-Stokes equation.

A flow field is characterized by a velocity of space and time. In this thesis, the flow field is based on three-dimensional velocity and one-dimensional pressure. Assume the three velocity components in x, y, z directions are u, v, w, respectively. p is the pressure.

In conservation of mass,

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial x} = 0 \tag{41}$$

In conservation of momentum,

$$\begin{aligned} \rho\frac{Du}{Dt} &= \rho B_x + \frac{\partial}{\partial x}\sigma_{xx} + \frac{\partial}{\partial y}\tau_{xy} + \frac{\partial}{\partial z}\tau_{xz} \\ \rho\frac{Dv}{Dt} &= \rho B_y + \frac{\partial}{\partial y}\sigma_{yy} + \frac{\partial}{\partial z}\tau_{yz} + \frac{\partial}{\partial x}\tau_{zy} \\ \rho\frac{Dw}{Dt} &= \rho B_z + \frac{\partial}{\partial z}\sigma_{zz} + \frac{\partial}{\partial x}\tau_{xz} + \frac{\partial}{\partial y}\tau_{zy} \end{aligned} \tag{42}$$

where ρ is the density, B_x, B_y and B_z are the body forces in x, y, z directions, σ_x, σ_y and σ_z are the normal stresses in x, y, z directions, and τ_{xy}, τ_{xz}, τ_{yz} are stress in shears. At the same time,

$$\begin{aligned} \frac{Du}{Dt} &= \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial t} + v\frac{\partial v}{\partial t} + w\frac{\partial w}{\partial t} \\ \frac{Dv}{Dt} &= \frac{\partial v}{\partial t} + u\frac{\partial u}{\partial t} + v\frac{\partial v}{\partial t} + w\frac{\partial w}{\partial t} \\ \frac{Dw}{Dt} &= \frac{\partial w}{\partial t} + u\frac{\partial u}{\partial t} + v\frac{\partial v}{\partial t} + w\frac{\partial w}{\partial t} \end{aligned} \tag{43}$$

$$\begin{aligned} \sigma_x &= 2\,\mu\frac{\partial u}{\partial x} \\ \sigma_y &= 2\,\mu\frac{\partial v}{\partial y} \\ \sigma_z &= 2\,\mu\frac{\partial w}{\partial z} \end{aligned} \tag{44}$$

$$\begin{aligned} \sigma_{xx} &= \sigma_x - p \\ \sigma_{yy} &= \sigma_y - p \\ \sigma_{zz} &= \sigma_z - p \end{aligned} \tag{45}$$

$$\begin{aligned}\tau_{xy} &= \mu\left(\frac{\partial u}{\partial y}+\frac{\partial v}{\partial x}\right)\\ \tau_{xz} &= \mu\left(\frac{\partial u}{\partial z}+\frac{\partial w}{\partial x}\right)\\ \tau_{yz} &= \mu\left(\frac{\partial w}{\partial y}+\frac{\partial v}{\partial z}\right)\end{aligned} \tag{46}$$

Eq. (41) and Eq. (42) are Navier-Stokes equations. In Eq. (42), the term in the left hand side is called the inertia force term, the first term in the right hand side is called the body force term, the second term is called pressure force term, and the rest forms the viscous terms.

4.1.1 Incompressible Viscous Flow without Inertia

If Eq. (41) and Eq. (42) are made dimensionless, the result is a dimensionless group known as the Reynolds number, Re, which represents the ratio of inertia forces to viscous forces in a fluid motion. When Re is small enough, such as $Re < 1$, the inertia force can be ignored from the governing momentum equations. Small Reynolds number characterize slow-moving fluids or viscous fluids.

If omitting the inertia term, Eq. (42) becomes

$$\begin{aligned}\frac{\partial}{\partial x}\sigma_{xx}+\frac{\partial}{\partial y}\tau_{xy}+\frac{\partial}{\partial z}\tau_{xz}&=0\\ \frac{\partial}{\partial x}\tau_{xy}+\frac{\partial}{\partial y}\sigma_{yy}+\frac{\partial}{\partial z}\tau_{yz}&=0\\ \frac{\partial}{\partial x}\tau_{xz}+\frac{\partial}{\partial y}\tau_{yz}+\frac{\partial}{\partial z}\tau_{zz}&=0\end{aligned} \tag{47}$$

Body forces have not been written in these equations because they may be grouped with pressure terms when the body can be expressed as the gradient of a potential function.

4.1.2 Incompressible Viscous Flow with Inertia

In contrast to the previous discussion, if Re is large, the problem becomes nonlinear because of the presence of the inertia terms. So Eq. (42) becomes

$$
\begin{aligned}
\rho\frac{Du}{Dt} &= \frac{\partial}{\partial x}\sigma_{xx} + \frac{\partial}{\partial y}\tau_{xy} + \frac{\partial}{\partial z}\tau_{xz} \\
\rho\frac{Dv}{Dt} &= \frac{\partial}{\partial y}\sigma_{yy} + \frac{\partial}{\partial z}\tau_{yz} + \frac{\partial}{\partial x}\tau_{zy} \\
\rho\frac{Dw}{Dt} &= \frac{\partial}{\partial z}\sigma_{zz} + \frac{\partial}{\partial x}\tau_{xz} + \frac{\partial}{\partial y}\tau_{zy}
\end{aligned}
\tag{48}
$$

Considering same reason as before, the body force terms have been removed. And the inertia terms in the left hand side are nonlinear which will be processed in the discretization.

4.2 Discretization

4.2.1 Incompressible Viscous Flow Without Inertia

Let h_i be the velocity interpolation function of node i within element m and h_i^p be the pressure interpolation function of node i within element m. We can get

$$u = \sum_i h_i U_i$$

$$v = \sum_i h_i V_i$$

$$w = \sum_i h_i W_i$$

$$p = \sum_i h_i^p P_i$$

Next we can apply Galerkin procedures to Eq. (47) and use the interpolation functions as wighting function,

$$
\begin{aligned}
&\sum_m \int_{V^{(m)}} \left[\left(2\,\mu \frac{\partial u}{\partial x} - p \right) \frac{\partial h_i}{\partial x} + \mu \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) \frac{\partial h_i}{\partial y} + \mu \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right) \frac{\partial h_i}{\partial z} \right] d\mathbf{V}^{(m)} \\
&\qquad = \sum_m \int_{S^{(m)}f} \bar{\sigma}_x h_i dS_f^{(m)} \\
&\sum_m \int_{V^{(m)}} \left[\mu \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) \frac{\partial h_i}{\partial x} + \left(2\,\mu \frac{\partial v}{\partial y} - p \right) \frac{\partial h_i}{\partial y} + \mu \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right) \frac{\partial h_i}{\partial z} \right] d\mathbf{V}^{(m)} \\
&\qquad = \sum_m \int_{S^{(m)}f} \bar{\sigma}_y h_i dS_f^{(m)} \\
&\sum_m \int_{V^{(m)}} \left[\mu \left(\frac{\partial u}{\partial z} + \frac{\partial w}{\partial x} \right) \frac{\partial h_i}{\partial x} + \mu \left(\frac{\partial v}{\partial z} + \frac{\partial w}{\partial y} \right) \frac{h_i}{\partial y} + \left(2\,\mu \frac{\partial w}{\partial z} - p \right) \frac{\partial h_i}{\partial z} \right] d\mathbf{V}^{(m)} \\
&\qquad = \sum_m \int_{S^{(m)}f} \bar{\sigma}_z h_i dS_f^{(m)}
\end{aligned}
\tag{49}
$$

Note that in the right hand side, σ becomes $\bar{\sigma}$, and the integration is not in the volume $\mathbf{V}^{(m)}$ but on the surface $S_f^{(m)}$. This means the natural boundary conditions, that is, the surface tractions, on the right hand side. The boundary conditions is

$$
\begin{aligned}
\bar{\sigma}_x &= (\sigma_x - p)\, n_x + \tau_{xy} n_y + \tau_{xz} n_z \\
\bar{\sigma}_y &= \tau_{xy} n_x + (\sigma_y - p)\, n_y + \tau_{yz} n_z \\
\bar{\sigma}_z &= \tau_{xz} n_x + \tau_{yz} n_y + (\sigma_z - p)\, n_z
\end{aligned}
\tag{50}
$$

where n_x, n_y, and n_z denote direction cosine of the unit outer normal to the surface.

Eq. (49) can be substituted by the interpolation functions of u, v, w and p. Then we can simply obtain a linear equation;

$$\mathbf{KU} = \mathbf{R}$$

which is exactly the same as Eq. (5) in the structure part. But the formulation of $\mathbf{K}$, $\mathbf{U}$ and $\mathbf{R}$ is different. Suppose that velocity components are interpolated at nv_1 nodes, while the pressure is interpolated at $nv2$ nodes. Then $\mathbf{K}$ with size of $3\,nv_1 + nv_2$ by $3\,nv_1 + nv_2$ will be splited into several submatrices.

$$\mathbf{K} = \begin{bmatrix} 2\mathbf{K}_{11} + \mathbf{K}_{22} + \mathbf{K}_{33} & \mathbf{K}_{12} & \mathbf{K}_{13} & \mathbf{L}_1 \\ \mathbf{K}_{12}^T & \mathbf{K}_{11} + 2\mathbf{K}_{22} + \mathbf{K}_{33} & \mathbf{K}_{23} & \mathbf{L}_2 \\ \mathbf{K}_{13}^T & \mathbf{K}_{23}^T & \mathbf{K}_{11} + \mathbf{K}_{12} + 2\mathbf{K}_{33} & \mathbf{L}_3 \\ \mathbf{L}_1^T & \mathbf{L}_2^T & \mathbf{L}_3^T & 0 \end{bmatrix} \tag{51}$$

For these submatrices in Eq. (51), the sizes of $\mathbf{K}_{11}$, $\mathbf{K}_{22}$, $\mathbf{K}_{33}$, $\mathbf{K}_{12}$, $\mathbf{K}_{13}$, and $\mathbf{K}_{23}$ are nv_1 by nv_1. Suppose that $k_{i,j}^{11}$, $k_{i,j}^{22}$, $k_{i,j}^{33}$, $k_{i,j}^{12}$, $k_{i,j}^{13}$, $k_{i,j}^{23}$ are their entries respectively. We can get

$$k_{i,j}^{11} = \int_{V^{(m)}} \mu \frac{\partial h_i}{\partial x}\frac{\partial h_j}{\partial x} d\mathbf{V}^{(m)} \tag{52}$$

$$k_{i,j}^{22} = \int_{V^{(m)}} \mu \frac{\partial h_i}{\partial y}\frac{\partial h_j}{\partial y} d\mathbf{V}^{(m)} \tag{53}$$

$$k_{i,j}^{33} = \int_{V^{(m)}} \mu \frac{\partial h_i}{\partial z}\frac{\partial h_j}{\partial z} d\mathbf{V}^{(m)} \tag{54}$$

$$k_{i,j}^{12} = \int_{V^{(m)}} \mu \frac{\partial h_i}{\partial y}\frac{\partial h_j}{\partial x} d\mathbf{V}^{(m)} \tag{55}$$

$$k_{i,j}^{13} = \int_{V^{(m)}} \mu \frac{\partial h_i}{\partial z}\frac{\partial h_j}{\partial x} d\mathbf{V}^{(m)} \tag{56}$$

$$k_{i,j}^{23} = \int_{V^{(m)}} \mu \frac{\partial h_i}{\partial z}\frac{\partial h_j}{\partial y} d\mathbf{V}^{(m)} \tag{57}$$

where the sizes of $\mathbf{L}_1$, $\mathbf{L}_2$ and $\mathbf{L}_3$ are nv_2 by nv_2. Suppose that $l^1_{i,j}$, $l^2_{i,j}$, $l^3_{i,j}$ are their entries respectively. We can get

$$l^1_{i,j} = -\int_{V^{(m)}} \frac{\partial h_j}{\partial x} h^p_i d\mathbf{V}^{(m)} \tag{58}$$

$$l^2_{i,j} = -\int_{V^{(m)}} \frac{\partial h_j}{\partial y} h^p_i d\mathbf{V}^{(m)} \tag{59}$$

$$l^3_{i,j} = -\int_{V^{(m)}} \frac{\partial h_j}{\partial z} h^p_i d\mathbf{V}^{(m)} \tag{60}$$

At the same time, we need to consider

$$\mathbf{U} = \begin{bmatrix} u_1 \\ \vdots \\ u_{nv1} \\ v_1 \\ \vdots \\ v_{nv1} \\ w_1 \\ \vdots \\ w_{nv1} \\ p_1 \\ \vdots \\ p_{nv2} \end{bmatrix} \tag{61}$$

and

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_u \\ \mathbf{R}_v \\ \mathbf{R}_w \\ \mathbf{0} \end{bmatrix} \tag{62}$$

Here we assume r_i^u, r_i^v, and r_i^w are the entries of $\mathbf{R}_u$, $\mathbf{R}_v$ and $\mathbf{R}_w$ respectively, them

$$r_i^u = \int_{S_f} \bar{\sigma_x} h_i dS_f \tag{63}$$

$$r_i^v = \int_{S_f} \bar{\sigma_y} h_i dS_f \tag{64}$$

$$r_i^w = \int_{S_f} \bar{\sigma_z} h_i dS_f \tag{65}$$

Once the element equations have been evaluated, they can be assembled in the usual manner to form the global equations. Boundary conditions work as the same as Sec. 3.3. However, one thing should be noticed again that these element finite equations are based on velocity. In Sec. 3, the finite element equations are based on displacement. For the fixed points, what we need to do is set the velocity rather than displacement as fixed values.

4.2.2 Incompressible Viscous Flow with Inertia

Once considering the inertia forces, the equations will depend on time. Compared with Eq. (47), Eq. (48) has inertia terms in the left hand side. According to Eq. (43), the inertia terms are nonlinear and require special consideration. To apply Galerkin approach, we need to use an approximate solution (u', v', w') to linearize this term. After linearizing, we get

$$\begin{aligned}
\rho\left(\frac{\partial u}{\partial t} + u'\frac{\partial u}{\partial x} + v'\frac{\partial v}{\partial y} + w'\frac{\partial w}{\partial z}\right) &= \frac{\partial}{\partial x}(\sigma_x - p) + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} \\
\rho\left(\frac{\partial v}{\partial t} + u'\frac{\partial u}{\partial x} + v'\frac{\partial v}{\partial y} + w'\frac{\partial w}{\partial z}\right) &= \frac{\partial \tau_{xy}}{\partial x} + \frac{\partial}{\partial y}(\sigma_y - p) + \frac{\partial \tau_{yz}}{\partial z} \\
\rho\left(\frac{\partial w}{\partial t} + u'\frac{\partial u}{\partial x} + v'\frac{\partial v}{\partial y} + w'\frac{\partial w}{\partial z}\right) &= \frac{\partial \tau_{xz}}{\partial x} + \frac{\partial \tau_{yz}}{\partial y} + \frac{\partial}{\partial z}(\sigma_z - p)
\end{aligned} \tag{66}$$

In this project, we set the solution in previous time points the previous (u', v', w') in Eq. (66). Similar to Section 4.2.1, the interpolation functions will be applied in Eq. (66). Following the same procedures, some new terms arise from the inertia term, such as

$$\mathbf{M}\dot{\mathbf{U}}(t) + \mathbf{N}\dot{\mathbf{U}}(t) + \mathbf{K}\dot{\mathbf{U}}(t) = \mathbf{R}(t) \tag{67}$$

For Eq. (67), the definitions of $\mathbf{K}$ and $\mathbf{R}(t)$ are the same as the derivation in Section 4.2.1. And

$$\mathbf{M} = \begin{bmatrix} \mathbf{M}_e & 0 & 0 & 0 \\ 0 & \mathbf{M}_e & 0 & 0 \\ 0 & 0 & \mathbf{M}_e & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{68}$$

$$\mathbf{N} = \begin{bmatrix} \mathbf{C}_{11} + \mathbf{C}_{22} + \mathbf{C}_{33} & 0 & 0 & 0 \\ 0 & \mathbf{C}_{11} + \mathbf{C}_{22} + \mathbf{C}_{33} & 0 & 0 \\ 0 & 0 & \mathbf{C}_{11} + \mathbf{C}_{22} + \mathbf{C}_{33} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{69}$$

where the sizes of $\mathbf{M}_e$,$\mathbf{C}_{11}$, $\mathbf{C}_{22}$, and $\mathbf{C}_{33}$ are nv_1 by nv_1. Suppose that $m^e_{i,j}$ and $c^{11}_{i,j}$, $c^{22}_{i,j}$,and $c^{33}_{i,j}$ are the entries of $\mathbf{M}_e$, $\mathbf{C}_{11}$, $\mathbf{C}_{22}$, and $\mathbf{C}_{33}$ respectively. And the entries can be calculated as

$$m^e_{i,j} = \rho \int_{V^{(m)}} h_j h_i d\mathbf{V}^{(m)} \tag{70}$$

$$c^{11}_{i,j} = \rho \int_{V^{(m)}} u'_j \frac{\partial h_i}{\partial x} dV^{(m)} \tag{71}$$

$$c^{22}_{i,j} = \rho \int_{V^{(m)}} v'_j \frac{\partial h_i}{\partial y} dV^{(m)} \tag{72}$$

$$c_{i,j}^{33} = \rho \int_{V^{(m)}} w_j' \frac{\partial h_i}{\partial z} dV^{(m)} \tag{73}$$

By now, we have completed all derivation for fluid finite element equations. From Eq. (51) to Eq. (73), we can consider the element matrices first and assemble the global matrices by the same way given in Sec. 3.2.3. And we can use the same approach as the way we use in Section 3.3 to apply the boundary conditions.

5 Fluid-Structure Interaction

To combine the structure and fluid analysis, the Fluid-Structure Interaction (FSI) problem has been employed and plays a key role in the simulation. Fluid structure interactions are a crucial consideration in the design of many engineering systems, e.g. aircraft and bridges. In this work, all boundary conditions applied at the boundary of CSF will impact the simulation results in the structural part as well as fluid part.

As mentioned before, Eq. (4) and Eq. (67), which are the dynamic equations for the solid and fluid model, need to be discretized in time domain. $\mathbf{U}$ is the displacements of the structure and is the velocities of the fluid. Now we can use $\mathbf{U}^t$ to represent the value of $\mathbf{U}$ at time point t, and use $\mathbf{U}^{t+\Delta t}$ to represent $\mathbf{U}$ at next time poit $t + \Delta t$. At the same time, the subscript of $\mathbf{U}_s$ and $\mathbf{U}_f$ is to distinguish $\mathbf{U}$ in structure and fluid. Details can be found in [6]

The structural predictor is applied to predict the initial velocities of fluid after getting the displacements of structures. In the boundary surface, we can assume

$$\mathbf{U}_f^t = \dot{\mathbf{U}}_s^t = \frac{\mathbf{U}_s^t - \mathbf{U}_s^{t-\Delta t}}{\Delta t} \tag{74}$$

which is the simplest scheme called backward Euler method.

The fluid interface force $\mathbf{h}$ describes the impact from the fluid to structure, and will be viewed as an additional boundary conditions for structure. Fluid interface force can be calculated as

$$\mathbf{h}^{\Delta t} = \left(\nabla \mathbf{U}_f^t - \nabla \mathbf{U}_f^{t^T}\right) \mathbf{n} - p\mathbf{n} \tag{75}$$

where $\mathbf{n}$ is the normal vector and ∇ is the gradient operator. It is important to indicate again that $\mathbf{U}_f$ and $\mathbf{U}_s$ only contains the information of nodes in the boundary.

Now the algorithm to solve this FSI problem can be summarized as

```
1: for time points $i = 0$ to n do
2:     Solve $\mathbf{U}_s^t$ from Eq. (4)
3:     Predict boundary velocity $\mathbf{U}_f^t$ using Eq. (74)
4:     Use $\mathbf{U}_f^t$ to solve Eq. (67)
5:     Calculate Eq. (75) by the velocity and pressure in the boundary nodes
6:     Add $\mathbf{h}$ to boundary conditions of structure
7:     $t = t + \Delta t$
8: end for
9: return $\mathbf{U}_f^t$ and $\mathbf{U}_s^t$
```

6 Numerical Methods

In Chapter 3, we derived the equation

$$\mathbf{KU} = \mathbf{R}$$

for the static analysis of solids, and

$$\mathbf{M}\ddot{\mathbf{U}}(t) + \mathbf{D}\dot{\mathbf{U}}(t) + \mathbf{KU} = \mathbf{R}(t)$$

for the dynamic analysis of solids.

In Ch. 4, we derived the equation

$$\mathbf{KU} = \mathbf{R}$$

for static analysis of fluids without inertia, and

$$\mathbf{M}\dot{\mathbf{U}}(t) + \mathbf{NU}(t) + \mathbf{KU} = \mathbf{R}(t)$$

for dynamic analysis of fluids with inertia. Here, the numerical concepts of solving this problem will be derived. To improve the efficiency, parallel computing methods are applied in this project.

6.1 Numerical Integration

To establish these finite element equation, we need to calculate the element matrices before compiling the global matrices. To process this, the volume integration is widely used. For example, in Eq. (32),

$$\mathbf{K}^{(m)} = \int_{V^{(m)}} \mathbf{B}^{(m)^T}\mathbf{C}^{(m)}\mathbf{B}^{(m)} dV^{(m)}$$

where $\mathbf{B}$ is function of coordinate (u, v, w). So the volume integration is based on coordinate (u, v, w). At the same time, column differential is

$$d\mathbf{V} = dx\,dy\,dz = du\,dv\,dw\,d\mathbf{J} \tag{76}$$

where $\mathbf{J}$ is the Jacobi matrix. In this project, Gaussian quadrature is applied in the integration. Gaussian quadrature gives an exact result for a polynomial by a suitable choice of sampling points, called Gaussian nodes and corresponding weights, called Gaussian weights. Each element will be assigned some Gaussian nodes, and each Gaussian node has a specific coefficient Gaussian weight w.

The numerical integration can be approximated by a summation

$$\int_{V^{(m)}} \mathbf{B}^{(m)^T}\mathbf{C}^{(m)}\mathbf{B}^{(m)} dV^{(m)} \approx \sum_{i,j,l} w_{i,j,k}\mathbf{B}^{(m)^T}\mathbf{C}^{(m)}\mathbf{B}^{(m)} det\mathbf{J} \tag{77}$$

By this way, the volume integral becomes a summation. And this can be used to calculate all the volume integration in Chapter 3 and 4.

6.2 Nonlinear and Dynamic Analysis

Eq. (4) and Eq. (67) are nonlinear dynamic differential equations. A widely used scheme is Newton-Raphson method, which is an iterative method to approximate a nonlinear function. A very simple example will given to show how the Newton-Raphson method works.

Suppose that $f(x)$ and $g(x)$ are nonlinear and time-varying functions and that $f(t) = g(x)$

to be solved. Then

$$\frac{\partial f}{\partial x}_{x_{i-1}^{t+\Delta t}} \Delta x_i = \left(g^{t+\Delta t} - f_{i-1}^{t+\Delta t}\right)$$

and

$$x_i^{t+\Delta t} = x_{i-1}^{t+\Delta t} + \Delta x_i$$

where i is the iteration number. The iteration continues until appropriate convergence criteria are satisfied. Compared with other numerical methods, Newton-Raphson method has better stability.

Eq. (4) is a second order differential equation, and Eq. (67) is a first order differential equation. The damping term in Eq. (4) can be obtained using similar way described below, and is not required in the simulation. So we can neglect that. It is obvious to know

$$\dot{\mathbf{U}}^{t+\Delta t} = \mathbf{U}^t + \frac{\Delta t}{2}\left(\dot{\mathbf{U}}^t + \dot{\mathbf{U}}^{t+\Delta t}\right) \tag{78}$$

$$\ddot{\mathbf{U}}^{t+\Delta t} = \dot{\mathbf{U}}^t + \frac{\Delta t}{2}\left(\ddot{\mathbf{U}}^t + \ddot{\mathbf{U}}^{t+\Delta t}\right) \tag{79}$$

We apply the Newton-Raphson method to Eq. (4). Then as above, we can get

$$\mathbf{M}\ddot{\mathbf{U}}_i^{t+\Delta t} + \mathbf{K}^t \Delta \mathbf{U}_i = \mathbf{R}^{t+\Delta t} \tag{80}$$

$$\mathbf{U}_i^{t+\Delta t} = \mathbf{U}_{i-1}^{t+\Delta t} + \Delta \mathbf{U}_i \tag{81}$$

and substitute Eq. (78) and Eq. (79) into Eq. (81), we can get

$$\ddot{\mathbf{U}}^{t+\Delta t} = \frac{4}{\Delta t^2}\left(\mathbf{U}_{i-1}^{t+\Delta t} - \mathbf{U}^t + \Delta \mathbf{U}_i\right) - \frac{4}{\Delta t}\dot{U}^t - \ddot{\mathbf{U}}^t \tag{82}$$

$$\dot{\mathbf{U}}^{t+\Delta t} = \frac{2}{\Delta t}\left(\mathbf{U}_{i-1}^{t+\Delta t} - \mathbf{U}^t + \Delta \mathbf{U}_i\right) - \dot{U}^t \tag{83}$$

Then we can take Eq. (82) into Eq. (80)

$$\left(\mathbf{K}^t + \frac{4}{\Delta t^2}\mathbf{M}\right) = \mathbf{R}_{i-1}^{t+\Delta t} - \mathbf{M}\left(\frac{4}{\Delta t^2}\left(\mathbf{U}_{i-1}^{t+\Delta t} - \mathbf{U}^t\right) - \frac{4}{\Delta t}\dot{\mathbf{U}}^t - \ddot{\mathbf{U}}^t\right) \tag{84}$$

Therefore, we can change the Newton-Raphson method to obtain an iterative algorithm to solve Eq. (4). The algorithm is:

```
1: for time points $i = 0$ to n do
2:     Solve Eq. (84)
3:     Use Eq. (82) and Eq. (13) to calculate $\ddot{\mathbf{U}}^{t+\Delta t}$ and $\dot{\mathbf{U}}^{t+\Delta t}$
4: end for
5: return $\mathbf{U}^t$
```

Similarly, we apply Newton-Raphson method in Eq. (67). we can obtain

$$\left(\mathbf{K}^t + \mathbf{N}^t + \frac{1}{\Delta t}\mathbf{M}\right) = \mathbf{R}_{i-1}^{t+\Delta t} - \mathbf{M}\left(\frac{2}{\Delta t}\left(\mathbf{U}_{i-1}^{t+\Delta t} - \mathbf{U}^t\right) - \dot{\mathbf{U}}^t\right) \tag{85}$$

Therefore, the algorithm to solve Eq. (67) is

```
1: for time points $i = 0$ to n do
2:     Solve Eq. (85)
3:     Use Eq. (82) and Eq. (13) to calculate $\ddot{\mathbf{U}}^{t+\Delta t}$ and $\dot{\mathbf{U}}^{t+\Delta t}$
4: end for
5: return $\mathbf{U}^t$
```

6.3 Matrix Format

The system matrix $\mathbf{K}$ in a FEM reflects the geometry of the modeled object. This matrix is sparse. Each row (or column) represents a degree of freedom of a vertex. Nonzero off-diagonal elements in this row (or column) represent the neighbors. Their values describe the interaction between neighbors. Since one vertex has limited number of neighbor vertices, most elements in this system matrix should be zeros.

We use compressed sparse row (CSR) to store a sparse matrix. The CSR includes three arrays. The first one is row index whose length corresponds to the number of rows, and of unsigned integer type. The second one is column index whose length is the number of nonzero elements, and of unsigned integer type. The third one is element value whose length is the same as column index and of real type. Considering the 3 by 3 matrix $\mathbf{A}$

$$\mathbf{A} = \begin{bmatrix} 1 & 0.5 & 0 \\ 100 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Here, the array of row indices is $\mathbf{A}.\mathbf{ri}$, which is

$$\mathbf{A}.\mathbf{ri} = \begin{bmatrix} 1 & 3 & 5 & 5 \end{bmatrix}$$

the array of column indices is $\mathbf{A}.\mathbf{ci}$, which is

$$\mathbf{A}.\mathbf{ci} = \begin{bmatrix} 1 & 2 & 1 & 2 & 3 \end{bmatrix}$$

and array of element values is $\mathbf{A}.\mathbf{nz}$, which is

$$\mathbf{A}.\mathbf{ci} = \begin{bmatrix} 1 & 0.5 & 100 & 1 & 1 \end{bmatrix}$$

The elements in $\mathbf{A}.\mathbf{ri}$ show the beginning of position of each rows. For example, the first element 1 means the first row begins at the first element in $\mathbf{A}.\mathbf{ci}$ and $\mathbf{A}.\mathbf{nz}$. And the second element 3 means the second row begins at the third element in $\mathbf{A}.\mathbf{ci}$ and $\mathbf{A}.\mathbf{nz}$. The elements in $\mathbf{A}.\mathbf{ci}$ show the column positions, and the elements in $\mathbf{A}.\mathbf{nz}$ shows the element values. For example, the third element in $\mathbf{A}.\mathbf{ci}$ is 1, and the third element in $\mathbf{A}.\mathbf{nz}$ is 100. Consider all the information from three arrays, the element in the second row and in the first column is 100.

6.4 Linear Solver and Preconditioner

In the previous sections, we have converted the nonlinear dynamic problems into linear problems. Since the resolution of the mesh is huge, the corresponding linear equation system are huge, typically, will beyond $500,000$ equations. Direct methods for solving such large systems are impossible. So we use iterative solvers and preconditioners to speed up the computations.

Some linear solvers will be applied in this project, such as stabilized bi-conjugate gradient (BiCG) solver [18], generalized minimum residual (GMRES) solver [19], and induced dimension reduction (IDR) solver [20]. All of these solvers have been applied in the simulation described in the Section 7. However, due to the large number of equations, it will takes very long time to meet the stopping criteria. So we need to use preconditioner to accelerate this process.

A preconditioner is a procedure to approximate the inverse of a matrix. For a proper choice, the number of the iterations of the solvers mentioned above might be reduced a lot. In this work, Jacobi preconditioner and algebraic multigrid (AMG) preconditioner [21] are used. Jacobi preconditioner [22] is to use the reciprocal of elements to replace the original elements in the diagonal of the matrix in the left hand side. AMG preconditioner can construct coarser grids in different levels. After solving the coarest level and correcting the result, we recover the original result of this system. Compared with Jacobi preconditioner, AMG is more complicated but more efficient. The performance of these preconditioner will be given in Chapter 7.

6.5 Parallel Computing

Since the size of the matrix is large, memory is a critical resource here. If memory requirements exceeds the amount installable on a single machine, a straight forward way to solve this problem is to distribute the computation to multiple processors. Each computer will have a part of the full problem, and will share the required information with others. In this project, the parallelization is based on the basic operations of matrices and vectors. For a matrix, each processor can has a subset of the rows. And for a vector, each processor has part of the elements.

For most operations, such as addition and subtraction, parallel computing uses the same scheme as the serial method. Here, some special operations will be described .

Vector multiplication is a common operation which is run several times in the simulation program. Each processor holds a part of the global vector, and calculates the corresponding result. Then all processors share the result and add all of them together.

Matrix vector multiplication is another common operation. According to the position of each element in the parallel matrix, each processor will request the corresponding vector element. And each processor will calculate the results and formulate the result vector.

Matrix matrix multiplication is the most complicated operation. For instance, $\mathbf{A}] \times \mathbf{B}$ will be calculated in parallel environment. We can obtain the indexes of rows of $\mathbf{B}$ which are required by other processors, according to the position of element of $\mathbf{A}$. Then the processors communicate the information to complete the computation. This operation costs much time in communication between processors, and is only used in the initialization of the AMG preconditioner.

Matrix transpose is another operation that needs communication between processors. For FEM, the elements are close to the diagonal. So the the transpose will cost less than the transpose of normal matrix. Each processor will obtain the elements that will be given to others, and send them to others. At same time, each processor will receive the element from other processor and allocate them.

These are basic operations used in the parallel environment. We use Message Passing Interface (MPI) standard [13] to implement the communication scheme in this work. MPI is a library specification for message-passing, proposed as a standard by a broadly based committee of vendors, implementors, and users. In this work, MPI works to transmit and receive the information between different processors, and to implement synchronizations between the processors.

An issue in parallelization a computational problem is to balance the load between processors. By this way, the communication cost will be minimized. In the very beginning of this project, METIS library is used to partition the mesh. METIS is a set of serial programs for partitioning graphs, partitioning finite element meshes, and producing fill reducing orderings for sparse matrices. The algorithms in METIS are based on multilevel graph partitioning described in [8] [9] [10] [11] and [15]. METIS uses approaches to successively reduce the size of the graph as well as to further refine the partition. After running optimization algorithm, vertices or primitives are assigned in different processors according to the geometry. The boundary surfaces of the mesh divided can be minimized, which means the portions in each processor has the least number of links with others. The left plot of Fig.4 shows the partitioned without using METIS, and the right one shows the partitioned mesh using METIS. Different colors show the portion assigned different processors. At the same time, METIS can balance the number of equations in each processors.

The mesh partitioning rountings of METIS take as input the element node array of the mesh

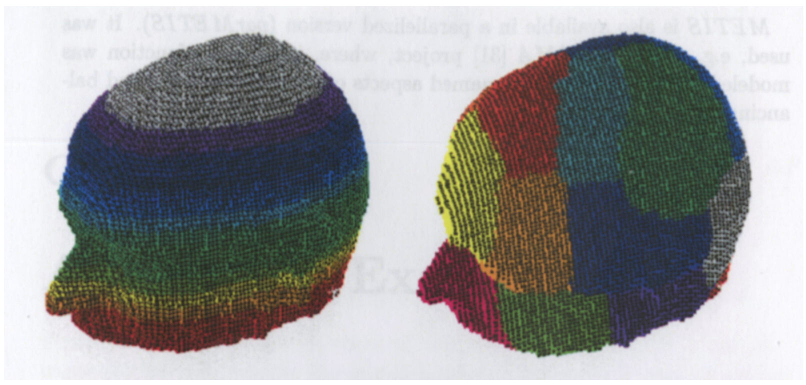
Figure 4: Partition using or not using METIS

and compute a partitioning for both its elements and its nodes. The built-in functions convert the mesh into a graph and calculate a partition of this graph as mention above. The mesh is converted either into a nodal graph which means each vertex of the graph represents a node of the mesh, or into a dual graph which mean each vertex represents an element of the mesh [16].

Here, it is important to note that the partition in this thesis is element-wise. This can reduce the communication cost in the process of formulating the structure matrices and solving linear equations. And other other operations are based on node-wise. In the simulation program, the mapping between partitions is a key part.

7 Experiments

In this section, the applibity of the composed chain of tools for clinical research shall be demonstrated. The intracranial pressure changes resulting from a forces applied ar the forehead are obtained under two assumptions. The first consideration is that CSF is solid in the head, another consideration is that CSF is fluid in the head. For the first consideration, [5] and [7] have performed similar simulations.

7.1 Materials

We use a simple linear elastic material model to complete this simulation. The material parameters is obtained from [5], listed in Table 1.

tissue	$E[MPa]$	ν	$\rho[kg/m^3]$
sclap / extracranial tissue	16.70	0.42	1200
skull	6500.00	0.22	1420
white matter	0.12	0.499	1040
gray matter	0.075	0.499	1040
CSF	0.075	0.49995	1045

Table 1: Material parameters

7.2 Choosing Solvers and Preconditioner

We need to find the best combination of linear solver and preconditioner. To minimize the computation cost, we test the run time for a the static analysis of the dataset. So, Eq. (5) will be calculated for different combinations of linear solvers and preconditioners, and run on 1, 5, 10 ,20 processors. We collect the number of iterations, total running time and running time in one iteration, is collected. In this work, we use the Generalized minimal residual method (GMRES), Stabilized Biconjugate gradient method (BiCGstab), and Induced Dimension Reduction method (IDRS) as linear solvers, and use Jacobi preconditiner and AMG preconditioner. Therefore, in this test, we will use the dataset to run the static analysis with different solvers, preconditioners and number of solvers. Once the results are obtained, which means the solver algorithm converges, the total number of iterations, the total time and the time per iteration will be recorded and analyzed. Results are collected.

Solver	Preconditioner	No. of iteration	running time (s)	Time per iteration (s)
GMRES	AMG	2	270	51
GMRES	Jacobi	3	70	9
BiCG	AMG	1010	2435	2.4
BICG	Jacobi	1991	582	0.27
IDRS	AMG	110	652	30
IDRS	Jacobi	136	1161	8

Table 2: Run time under serial environment

The GMRES solver needs the least number of iterations. Although the AMG preconditioner improves the speed of convergence more than a Jacobi preconditioner, the former one has a more higher computation cost. This is due to the complicated calculation in the initialization and calling of AMG. Considering the whole running time, a Jacobi preconditioner works better than AMG, and GMRES is the best choice for solver. Then, GMRES and AMG form the best combination.

Solver	Preconditioner	No. of iteration	Run time (s)	Time per iteration (s)
GMRES	AMG	2	$\sim$195	30
GMRES	Jacobi	3	$\sim$81	8
BiCG	AMG	990	$\sim$1709	1.5
BICG	Jacobi	1860	$\sim$352	0.16
IDRS	AMG	111	$\sim$2126	18.5
IDRS	Jacobi	147	$\sim$658	4.5

Table 3: Run time when using 5 processors

From Table 3, we can get same conclusion as above. In the test using 5 processors, the computation cost is distributed into 5 processors and each of them carries out $1/5$ amount of computation. However, the run times in Table 3 and Table 2 are similar. This is due to the communication cost between different processors. When more processors are working, the communication cost increases more. Many factors will influence the communication cost, such as the network topology or the performance of hardware.

No. of processors	No. of iterations	Run time (s)	Time per iteration (s)
1	3	70	9
5	3	~81	8
10	3	~51	5
20	3	~46	4.6

Table 4: Run time using GMRES and Jacobi preconditioner

No. of processors	No. of iteration	running time (s)	Time per iteration (s)
1	2	270	51
5	2	~195	38
10	2	~146	22
20	2	~152	22

Table 5: Run time using GMRES and AMG preconditioner

In Table 4, the deceasing trends of the total running time and the time in one iteration become slow when the number of processors increases. And this phenomena becomes more obvious in Table 5. This is because AMG precontioner is more complicated and needs more communication cost during calculation.

Based on this experiment a combination of a Jacobi perconditioner and GMRES will be used in the following simulations.

7.3 Simulation Results

As an example, we model the change of the intracranial pressure due to an impact that is directed approcximately to the center of mass of the head. The time course of the load is a half sinusoid with a period of $20ms$ and an offset of $2ms$. The simulation is performed from $0ms$ to $15ms$, with a sampling time Δt in Eq. 84 and Eq. 85 of $0.5ms$. A force of 1N is applied to a small area on the forehead (shown in Fig. 5).

There are two kinds of simulation in this section. First, CSF is considered as an elastic solid,

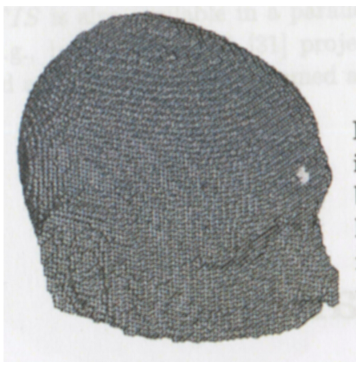

Figure 5: The impact forces are applied in the brighten area, from [5]

and a dynamic nonlinear solid analysis was preformed. For this condition, Eq. (4) will be solved directly by using the derivation in Chapter 3. Second, CSF is fluid with viscosity $\mu = 0.8\,(mPa * s)$ without inertia. To solve this, FSI problem will be involved.

As a consequence of a focal mechanical impact,the pressure in the blow area increases, and a

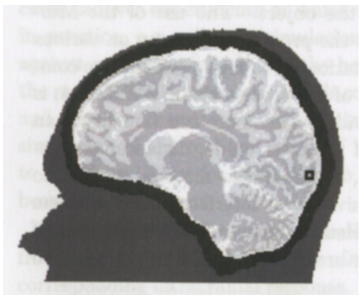

Figure 6: Contrecoup region, from [5]

pressure in the contralateral region decreases [5]. This is called coup-contrecoup phenomenon. The pressure is obtained from the contercoup region (area on the oppsed side of the impace, see Fig. 6) [2] and calculated as

$$\Delta\mathbf{P} = \frac{(\tau_{xx} + \tau_{yy} + \tau_{zz})}{3} \tag{86}$$

The simlation result is shown in Fig 7. The shock wave transferred via the skull and the acceleration of the head lead to intracainal pressure gradients which can lead to brain injuries. Injuries are usually more severe in the contrecoup region. According to the difference of the

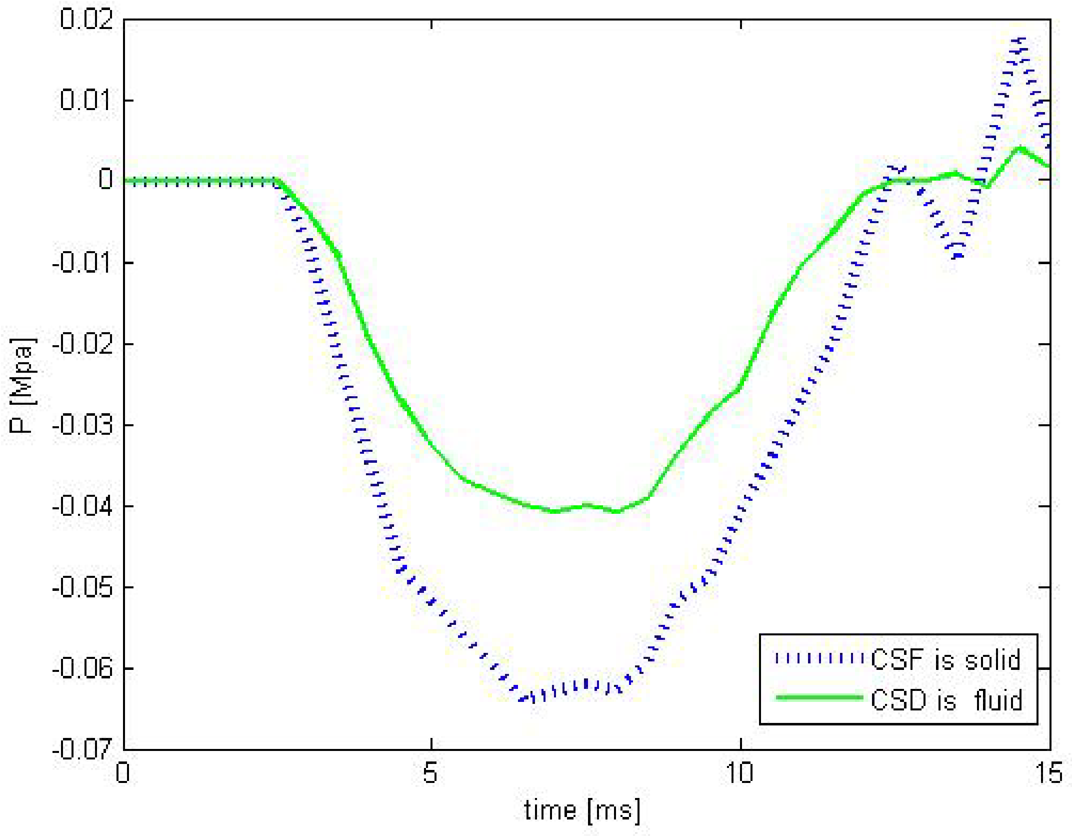

Figure 7: Simulation results

two curves, CSF acts as a cushion or buffer for the brain's cortex, providing a basic mechanical and immunological protection to the brain inside the skull [12].

8 Conclusion and Future work

In this thesis, a simulation tool of biomechanics has been presented. And s simulation of a frontal head impact has been given. A main feature of this simulation is that CSF is considered as fluid. In previous research, the solid or structure mechanics places an impart role when processing CSF. Therefore, one of contributions is increasing the realism of the simulation about human head impact. As a well-known fact, the damping effect of CSF works to protect brain from outer forces. The fluids properties of CSF allows much space for displacements and deformation when affected by applied forces. In this work, the fluids movements is determined by the displacements of coupled solid, at the same time the velocity and pressure forms the forces which act as boundary conditions in FEM to the coupled solid. The experiment results in Section 7.3 provide validation of this fact. In contrast, if considering CSF as solid, the different parts of human head will have tightly connection. As a result, the damping effect will be eliminated, and the brain injuries will occur easily.

Topics of possible future research in this area are manifold. One issue is about image resolution. Meninges mechanically decoupled sub-compartments of the skull. But they are below the resolution of the current image dataset. Because they are enclosed the CSF, form a boundary between fluid and solid. Once considering them, the velocity and pressure of the fluids will be changed and the boundary conditions will be modified as well. As a result, the simulation results will be more reasonable. Therefore, we need to make use of the progress in imaging techniques to enhance the accuracy and complexity of the graph.

Another problem is about concerns the resolution of the mesh. To obtain more accurate information, the resolution of the mesh should be increased. A higher resolution influences sparsity of the system matrices, which means the test in Section 7.2 is not valid and the new proper solvers and preconditioners need to be explored. At the same time, this will lead to larger computation cost and more computation time. There are several methods solving this.

Due to the high computation speed, we can use graphic processing unit (GPU) to implement the parallel computing. But this will be limited by the size of memory. So a reasonable way is to distribute the storage of data into several GPU, and use GPU to carry out the processing. This approach involves the hierarchy of the parallel computing.

In Section 7.3, the simulation is described about the front head impact. However, in practice, when a simulation is designed to replicate head impact against an object, for example, a steering wheel, the head neck junction should be considered in the model. This neck gives more constraints which will be designed as appropriate boundary conditions. Modern head-neck coils could be used to acquire a dataset of head simulation of the real trajectory of head movement with respect to much more than three degrees of freedoms.

This project uses a simplification of the material properties. Biomaterials have nonlinear and anisotropic properties. Nonlinear properties have to be included to update the system matrix. Knowledge about anisotropic materials can be obtained from other imaging methods such as diffusion-weighted imaging. As a result, all of the system matrices will be modified and the computation process is more involved.

In the simulation part, the inertia is not included in the fluid analysis. So, CSF is considered as still at each point. Obviously, this is not true. We will add body force terms and pressure force terms in Eq. (42) so as to make the fluids depend on time. Except the boundary conditions from solid, we need to consider the initial velocity and pressure from circulatory system.

All in all, a general goal of future work is to explore how practical and complicated of this model could be, in order to get more reasonable results. And the results can be applied in medical application. The current simulation has flexibility which can be used as basis in the future investigations.

References

[1] Bathe, Klaus-Jurgen
Finite Element Procedures
Prentice Hall, Inc. 1996.

[2] A. Sauren and M. Claessens
Finite Element Modeling of Head Impact: The second decade
Proceedings of the International Conference on Biomechanics of Impact (IRCOBI), pp. 241-254, 1993.

[3] M. Wu, O. Carmichael, P. Lopez-Garcia, C. Carter, and H. Aizenstein
Quantitative comparison of AIR, SPM, and the fully deformed model for atlas-based segmentation of functional and structural MR images
Human Brain Mapping, vol. 27, no. 9, pp. 747-754, 2006

[4] K. Huebner
The Finite Element Method for Engineers, second edition,
John Wiley & Sons, Inc, 1982

[5] A. Sprinkart
A Biomedical Finite Element Model of the Human Head Based on Individual Medical Images using Parallel Computing
Master Thesis, October, 2010

[6] G. Hou, J. Wang, and A. Layton
Numerical Methods for Fluid-Structure Interaction
Commun. Comput. Phys, vol. 12, no. 2, pp. 337-377, 2012

[7] G. Hou, J, Wang. and A. Layton
Intracranial Pressure Dynamics during Head Impact
Proceedings 21st Stapp Car Crash Conference, pp. 229-367, 1977

[8] G. Karypis and V. Kumar
Analysis of Multilevel Graph Partitioning
Proceedings 24th International Conference, pp. 113-122, 1995

[9] G. Karypis and V. Kumar
Multilevel k-way Partitioning Scheme for Irregular Graphs
Journal of Parallel and Distributed computing, pp. 96-129, 1998

[10] G. Karypis and V. Kumar
Multilevel Algorithms for Multi-Constraint Graph Partitioning
SC '98 Proceedings of the 1998 ACM/IEEE conference on Supercomputing, pp. 1-13 1998

[11] G. Karypis and V. Kumar
A fast and high quality multilevel scheme for partitioning irregular graphs
SIAM Journal on Scientific Computing, Vol. 20, No. 1, pp. 359 - 392, 1999

[12] Wikipedia
http://en.wikipedia.org/wiki/Cerebrospinal_fluid
(accessed: November 1, 2014).

[13] Message Passing Interface Forum
MPI: A Message-Passing Interface Standard, Version 3.0
https://computing.llnl.gov/tutorials/mpi , 2012

[14] K. Ueno, J.Melvin, L. Li, and J. Lighthall
Development of tissue level brain injury criteria by finite element analysis
Journal of Neurotrama, vol. 12, no, 4, pp. 695-706, 1995

[15] D. LaSalle and G. Karypis
Multi-Threaded Graph Partitioning
Proceeding IPDPS '13 Proceedings of the 2013 IEEE 27th International Symposium on Parallel and Distributed Processing, pp. 225-236, 2013

[16] G. Karypis
METIS: A Software Package for Partitioning Unstructured Graphs, Partitioning Meshes, and Computing Fill-Reducing Orderings of Sparse Matrices, Version 5.1.0
March 30, 2013

[17] George U. Hartmann, G. Lonsdale, G. Berti, J. Fingberg, A. Basermann, R. Grebe, P. Aimedieu, D. R. Hose, A. McCarthy, F. Kruggel, M. Tittgemeyer, C. Wolters, T. Hierl
SimBio: A generic Environment for Bio-numerical Simulation
Deliverables, 2002

[18] R. Barrett ,M. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van der Vorst
Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, 2nd Edition
SIAM, Philadelphia, PA, 1994

[19] D. Fong
Minimum-residual Methods for Sparse Least-squares Using Golub-Kahan Bidiagonalization
PhD Dissertation,Stanford University, 2011

[20] Peter Sonneveld and Martin B. van Gijzen
IDR(s): A Family of Simple and Fast Algorithms for Solving Large Nonsymmetric Systems of Linear Equations
SIAM Journal on Scientific Computing, Volume 31, No. 2, pp. 1035-1062, 2008

[21] R. Bridson and W. P. Tang
Multiresolution Approximate Inverse Preconditioners
SIAM Journal on Scientific Computing, Volume 23, no. 2, pp. 463-479, 2001

[22] Wikipedia
http://en.wikipedia.org/wiki/Preconditioner
(accessed: November 16, 2014)

[23] S. Sobolev
Partial Differential Equations of Mathematical Physics
Dove Publication, Inc. 1989