Master's internship report

Framework implementation for medical simulation with a posteriori and dynamic mesh adaptivity

L. Felipe RESTREPO B.

Department of Informatics
Master 2 - Specialty Image
Professional Program
SAARA - Simulation, Analysis, Animation for Augmented Reality
Université Claude Bernard Lyon 1
Lyon, France 2015

Framework implementation for medical simulation
with a posteriori and dynamic mesh adaptivity
L. Felipe RESTREPO B.

Supervisor: Fabrice JAILLET, LIRIS CNRS-UMR5205
Examiner: Jean Claude IEHL, UFR Informatique, UCBL

Master's internship report 2015
Department of Informatics
Master 2 - Specialty Image
SAARA - Simulation, Analysis, Animation for Augmented Reality
Université Claude Bernard Lyon 1

Typeset in LaTeX
Lyon, France 2015

Abstract

In the field of bio-mechanical simulation, the complexity of the 3D models are usually high and require extensive calculations in order to apply the physics over the simulated objects, but also it requires a minimum error in the mesh displacements calculations. This makes the simulation of soft-tissue a delicate process where there is a trade off between speed and precision depending on the desired application. A Work-flow was implemented with the support of various existing tools. An error criteria was then defined and implemented for the process of mesh comparison, this helped to validate a tetrahedral mesh simulation with coarse resolution against a standard mesh, it was explored the convergence of the model and the validity of the error criteria and the adaptive meshing process.

Keywords: Simulation, Adatative remeshing, error estimation, soft tissue, FEM, 3D Surfaces, Tetrahedra, tetgen, camitk, sofa.

Contents

1
Introduction

The anatomical modeling and simulation towards high-fidelity remains a major challenge in computer graphics. Bio-mechanics focuses its efforts on achieving organ simulations that not only are anatomically correct, but that reproduces the accurate behavior of a particular organ. Depending on the organ considered, it can be modeled in a simplified way as rigid body or it can be like in the case of soft tissue, modeled by finite element where the simulation deformations are crucial in the analysis of organ behavior. With the last decade of technology advances it has been possible to simulate models that each time come closer to the reality, in visual and mathematical terms, often with the help of robotics where the interaction with real-time haptics feedback has helped to train clinicians around all over the world.

The main goal in the internship is to define a workflow in which it is possible to realize simulations using different applications available in the market, this will allow to test and verify results developed by the research team and by other researchers as well. Having a common framework it's a necessity when it comes to methods and algorithms comparison, where being able to reproduce the conditions on which a particular simulation was done is a critical step to validate the results.

This document shows the work developed in the six months internship as part of the Master 2 in computer graphics, professional program. The internship was done at the LIRIS laboratory within the SAARA (Simulation, Analysis, Animation for Augmented Reality) research group.

1.1 Context

The use of computer graphics in the field of medicine is now a common tool to help doctors and patients in different ways; Like flying simulators have been for training pilots, the medical simulators are used mainly to train surgeons on difficult or complex procedures. They also offer an alternative to the planification of new techniques of surgery [1], where it is possible to visualize the effect of an operation before proceeding into the real operation room [2], where different scenarios can be tested and verified against the desired result. Simulations can also help on the surgery process while is taking place, giving additional information to the doctors during the process. Example of this is the deformation undergone by an organ, regarding an pre-surgery taken image.

In general any of this type of simulators require a precise modeling from the

[1] http://www.insimo.com/helpmesee/
[2] http://www.simsurgery.com/

anatomical structures, in order to have faithful results in a medical context. In the case of soft tissue they must represent the characteristics and behavior of the group of organs in particular being modeled. This is a complex task, due to the fact that bio-mechanic simulations may follow a non-linear, anisotropic and inhomogeneous behavior.

In the process of simulation multiple challenges are faced, some are bound to the model itself like non-linear or anisotropic behavior and can be approached from different approximative and analytical methods, others are constrained not only by the chosen method but also trough hardware and software implementations, like precision or speed, where both will be a factor of major impact when a simulation is intended to be executed during the process of surgery, or even before and time is a critical factor.

In the context of the internship the increment of speed without penalizing precision is one challenge that need to be tackled in order to be able to lay the base to simulations without a 3D mesh references.

1.2 Proposed Objectives

Given a physiological model it will be degraded to move towards more interactive simulations. This is reflected for example by the use of lower resolution meshes, using laws of simpler behavior (linear, nonlinear). The work will go through a phase of confrontation with the full model (taken as 'ground truth'). The challenge will be to control the error while ensuring the mechanical validity.

The main objectives of the internship are as follows:

- Literature review leading to the definition of a relevant error criteria, a posteriori.
- Software implementation (C++) of the adopted solution in the CamiTK platform.
- Framework integration with a simulation software.

This document proposes a workflow that gives a path to follow in the process of simulation of a given 3D mesh, from it's representation in NURBS, to it's final tetrahedralized simulated mesh. It's important to note that some are guidelines on how to proceed in various parts of the process, while other issues involve the coding and extension of various existing tools, necessary to semi-automate parts of the process.

1.3 Working Team

The internship is developed at the LIRIS (Laboratoire d'InfoRmatique en Image et Systèmes d'information) in the DOUA campus, it's composed by more than 320 members from different universities, among the CNRS, INSA de Lyon, Université

Claude Bernard Lyon 1, Lyon 2 University and Central School Lyon. The focus of the laboratory is the research in computer science and information technology.

The scientific activities from its 14 research groups are structured in six fields of expertise :

(a) LIRIS organization

1.3.1 SAARA Team

The research works of the SAARA team is concerned by virtual human modelling. These may involve external body movements (arms and joints, thoracic and abdominal, facial expressions, etc.) or internal movements (respiratory system, parturient pelvic organs interacting with the foetus, etc.).

Three areas are specially studied:

- Motion capture and movement analysis from
- Reconstruction, animation and simulation of virtual physiological human
- Multiphysics realistic simulation and shape modelling of virtual physiological human

The internship will be developed under the direction of Fabrice Jaillet, member of SAARA team.

2
Problematic

One of the main goals of the internship was to integrate a set of existing tools (2.2) that would allow the work on simulations in an more efficient and seamless way. This corresponds to the technical aspect of the internship, where I was focused in the exploration of the tools and development of code for the communication among them. Being this a technical task it required a great amount of time to explore each tool individually in order to find the correct approach at each piece of code due to the size of the applications and, the different design patterns.

In order to create a workflow that allow the easy integration of different libraries, a 'proxy' application is needed, one that permits to plug-in the different 'know-how', embedded in existing libraries and applications. It's important to mention that in order to being able to develop for said proxy application, it is ideal that the applications themselves are Open-Source, while this is not an obligatory condition of the project, it would allow the development (without paid licensing) of the requirements of the internship and, other requirements that might arise due to multiple researchers working on similar fields, either bio-mechanical simulation, medical prototyping, image segmentation, etc. A tool with this characteristics would be considered the main axis of the workflow, from where most of the code could be developed, without the necessity of extend multiple applications at once.

Part of the work done in the internship involved the definition of these tools, taking into account multiple variables like: licensing, portability, stability or maturity of the application, open community, documentation and maintainability among other. This task led to the exploration of a wide variety of tools and the study of its advantages amid other similar applications. The selected ones and the reasons behind the decisions are presented in this chapter (see 2.2).

2.1 *a posteriori error estimation*

Having computed a FEM solution, it is possible to obtain *a posteriori error estimates* which give a more quantitative information about the accuracy of the solution where some properties are desired[16].

- Accurate measure of the discretization error for a range of mesh spacings an polynomial degrees.
- The procedure should be inexpensive relative to the cost of obtaining the FEM solution.
- A technique that uses multiple estimate of pointwise errors which can subsequently be used to calculate error measures in several norms, against a specific

norm.

There are four categories for *a posteriori error estimates*[16].

1. ***Residual error estimates***: Local finite elment problems are created on either an element or a subdomain and solved for the error estimate. The data depends on the residual of the FEM solution

2. ***Flux-projection error estimates***: A new flux (or flow) is calculated by post processing the FEM solution. This flux is smoother than the original finite element flux and an error estimate is obtained from the difference of the two fluxes.

3. ***Extrapolation error estimates***: Two finite element solution having different order or different meshes are compared and their differences used to provide an error estimate.

4. ***Interpolation error estimates***: Interpolation errors bounds are used with estimates of the unknown constants.

a posteriori error estimates provide accuracy evaluations that are necessary to terminate an adaptive procedure. However, optimal strategies for deciding where and how to refine or move a mesh or to change the basis are rare. *a posteriori error estimates* in a particular norm are computed by the sum of their elemental contributions as:

$$\|E\| = \sum \|E\|_e^2 \tag{2.1}$$

For each element in the mesh $\|E\|_e^2$ of the error estimate $\|E\|^2$ to element e. It's assumed that large errors come from regions where the local error estimate $\|E\|_e$ is large ad this is where it should refine the mesh.

One of the goals of the research team SAARA, is to work towards a simulation process where the *a posteriori error estimation* can be discarded thanks to the knowledge acquired on an specific simulation process i.e, childbirth simulation, moreover when the desire is to realize a real-time simulation where the mesh can be coarse but medically and mathematically accurate. That's why the *a posteriori error estimation* will be taking into account 'Extrapolation error estimates' due to the advantage that offers to work on simplified meshes.

2.1.1 Refinement Criteria

The refinement criteria for the model re-meshing can be chosen from the *a posteriori error estimator*, regardless of the magnitude the simplest approach is to refine a fixed percentage of elements having the largest error indicator refine all elements e satisfying:

$$\epsilon_e \geq \lambda \max_{1 \leq j \leq n} \epsilon_j \tag{2.2}$$

A typical choice of the parameter $\lambda \in [0,1]$ is 0.8[16]. We can dynamically change this percentage to try a more aggressive criteria when the method is not producing enough new elements.

We will consider that the process of refinement is over when no new elements are added to the mesh, having this mesh fewer elements than the reference.

2.2 Used Applications

On the internship objectives CamiTK (2.3) was proposed as the one of the applications of the workflow, the described need was to integrate it to a simulator, some simulators where discussed, among them: Abaqus[1], libMesh[2], Vega[3] and SOFA (see 2.4). This was a technical task that required a detailed evaluation of the capabilities on each application, similarly was the case with other software such as the meshing libraries. It was very important to choose from the beginning a fit application to integrate, that could be extended and portable in order to pull together all the pieces of the workflow. This section will present the applications selected in the process, the capabilities of each one, the limitations considered and the role that each application or framework played in the workflow of the proposed solution.

2.3 CamiTK

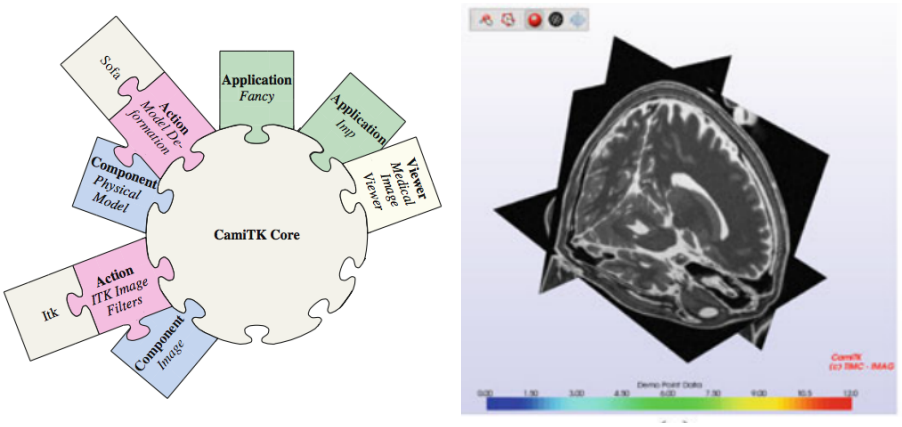
CamiTK, is a specific modular framework that helps researchers and clinicians to collaborate in order to prototype Computer Assisted Medical Intervention (CAMI) applications by using their knowledge and know-how during all the required steps. CamiTK is an open-source, cross-platform generic tool, written in C++, which can handle medical images, surgical navigations and bio-mechanical simulations[5]. CamiTK is developed with the idea to be a core connector between end-user (usually medical expert) and image based developers, where the 'know how' of each area can be easily integrated. Its general design is inspired by component-based software engineering (CBSE).

CamiTK like other applications aimed on biomechanics [6, 7, 5] has a basic set of tools already available out of the box like rigid body transformations, mesh clipping, gradient calculations, etc. but, added to these basic functionalities, it provides very intuitive interfaces that allow a fast prototyping of medical models. This application was chosen to be the core of the workflow, due to its functional architecture. It also has the advantage of an on-line community, which develops and contribute for the application, and that offer fast solutions to code issues through the forum.

2.3.1 Cmake

In order to have a multi-platform application, a build system that is compiler independent is required, Cmake[4] has this advantage and was also present as default build system for SOFA and CamiTK. Cmake played a big role in the integration

[1] http://www.3ds.com/products-services/simulia/
[2] http://libmesh.github.io/index.html
[3] http://run.usc.edu/vega/index.html
[4] http://www.cmake.org/



[5]

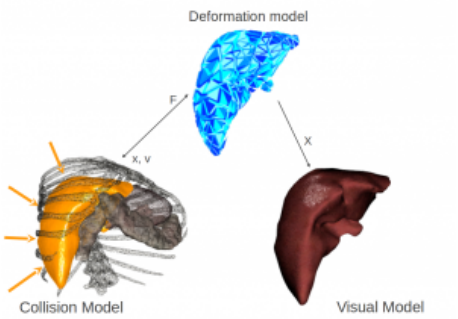
(a) CamiTK architecture Overview

(b) Full featured VTK 2D/3D renderer

process due to its capability to include external libraries which was crucial for the implementation of the workflow. It is also an application that runs on multiple platforms like Windows, Mac OSX and linux distributions.

2.4 Sofa Framework

SOFA (Simulation Open Framework Architecture) is an open-source C++ library primarily targeted at interactive computational medical simulation. It was developed with the idea to facilitate the collaborations between specialists from various domains, by decomposing complex simulators into components designed independently and organized in a scene-graph data structure. Each component encapsulates one of the aspects of a simulation, such as the degrees of freedom, the forces and constraints, the differential equations, the main loop algorithms, the linear solvers, the collision detection algorithms or the interaction devices[8]. The simulated objects can be represented using several models, each of them optimized for a different task such as the computation of internal forces, collision detection, haptics or visual display. These models are synchronized during the simulation using a mapping mechanism.



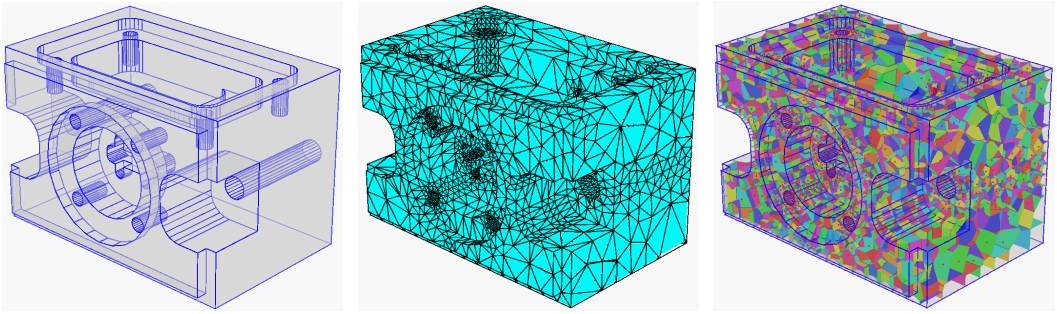
(a) Sofa Multi-mapping mechanism

Sofa uses for its simulations the visitors coding pattern which traverse the scene top-down and bottom-up, and call the corresponding virtual functions at each graph node traversal. Algorithmic operations on the simulated objects are implemented by deriving the Visitor class and overloading its virtual functions processNodeTopDown() and processNodeBottomUp().
CamiTK was previously integrated with SOFA[5], through an extension called MML (Markup Monitoring Language. See 2.6), this integration provided the communication Sofa and CamiTK, where the only exchange was the coordinates of the mesh being simulated.

2.5 TetGen

TetGen is a software designed to generate tetrahedral meshes for 3D (weighted) points, it can generate the Delaunay and weighted Delaunay as well as the Voronoi diagram power diagram. Additional to the mentioned methods For a 3D polyhedral domain, TetGen generates the constrained Delaunay tetrahedralization and an isotropic adaptive tetrahedral mesh of it. Domain boundaries (edges and faces) are respected and can be preserved in the resulting mesh. Written in C++ it uses only C standard library, this allows an easy multi-platform integration.

Besides of it's computational efficiency, TetGen, doesn't have special requirements to compile, it's easy and fast to build and an be called from outside trough the main method when compiled as an dynamic library.



(a) PLC Mesh **(b)** Tetrahedral Mesh **(c)** Dual - Voronoi

2.5.1 Mesh Adaptation, Mesh Sizing Functions

During the first weeks of the internship I worked mostly in the literature review of adaptive re-meshing methods, in order to define an error criteria for mesh comparison, this is one of the research objectives proposed in the internship. After a exploring some articles on the subject it was clear that a fair amount of literature related to adaptive meshing existed, mostly focused to improve the accuracy of finite element methods. An example of the vast literature is Oden and Demkowicz that worked on a numerical survey[11], while Jones and Plassmann did a geometric survey of hierarchical mesh refinement[12]. Moving mesh methods where presented by Budd, Huang, and Russell [14]. In general adaptive methods are a specialized form

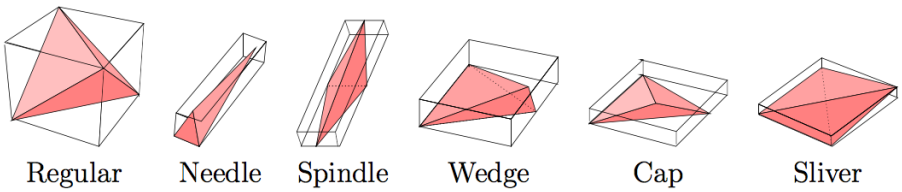
of dynamic meshing, and they are effective for adding detail only where needed, but they have the drawback that the deformation field is always anchored to the original coarse mesh[17].
Common procedures for mesh adaptation are the classified as follows:

- local refinement and/or coarsening of a mesh (h-refinement),
- relocating or moving a mesh (r-refinement),
- locally varying the polynomial degree of the basis (p-refinement).

The principal idea in the internship is to achieve a good model of mesh approximation by mesh refinement, for this strategy a coarse mesh will the starting point and it will be refined gradually, yielding ideally, the accuracy needed against a fine mesh. This strategy might not be optimal due to local meshing, where is possible to left untouched regions in some cases, this hypotesis will be tested with tetgen.

2.5.2 Mesh Quality, Tetrahedron Shape Measures

There are many definitions for 'mesh quality' and they are usually bounded to the application and the employed method see [10]. As a general rule, elements with very small and large angles (and dihedral angles) should be avoided since they downgrade the accuracy and performance of numerical methods. A tetrahedron shape measure is a continuous function which evaluates the shape of a tetrahedron by a real number. The shapes can be as follows:



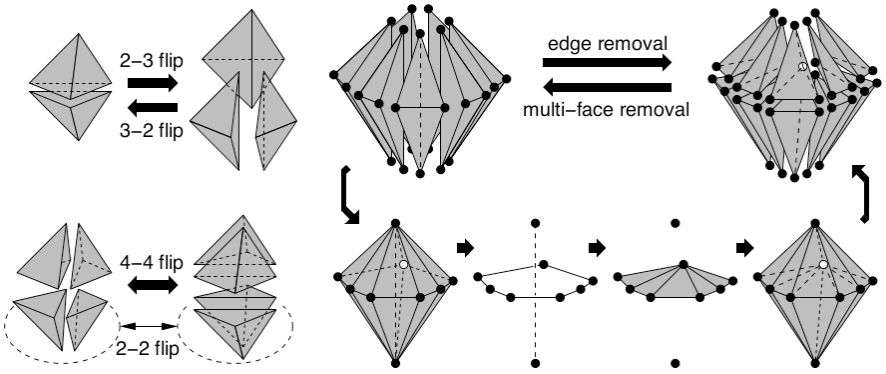
(a) Tetrahedra of different shapes.[17].

The most used shape measure for a simplex is the aspect ratio, for a tetrahedron its represented by $\eta(\tau)$ where τ is the ratio between the longest edge length l_{max} and the shortest hight h_{min} : $\eta(\tau) = l_{max}/h_{min}$. It measures the 'roundness' of a tetrahedron in terms of a value between $\sqrt{2}/\sqrt{3}$ and $+\infty$. A low aspect ratio implies a better shape. Other possible definitions of aspect ratio exist, such as the ratio between the circumradius and inradius.

Choosing a good quality measure for the current work represents a challenge, because a small number in the quality measure can change the response over the simulation, this is important to note because it will deviate the model from its ideal solution, but also because in some cases will render the solution unstable. Our model will only explore isotropic meshes generated with Delaunay triangulation methods, this means that the elements should have a low aspect ratio. Aditionally, three quality measures will be tested (*1.2*), ($\sqrt{2}$) and (*2*) [16, 9], in order to test the sensibility of the quality in the meshing method.

The quality of the produced mesh will undergo multiple possible ways of topological transformations. This transformations are going to be necessarily bounded

to the quality chosen for the meshing process. In many cases topological transformations will occur, changing the mesh topology by removing elements from a mesh and replacing them with a different set of elements occupying the same space. Figure 2.1a shows some of these transformations 2-3 flips, 3-2 flips, 4-4 flips, and 2-2 flips. Where the first digit denotes the number of tetrahedra removed and the second the number that are created.



(a) Examples of topological transformations[17]

2.6 XML Languages

One of the main issues when creating a mechanical model, can be its physical representation. This happens because these physical representations are usually bounded to the application and technologies that can process them. This makes difficult to share or to verify results of particular methods due to the lack of a generic representation of these models. Moreover when the models store different parameters relevant to its field of study, it's cumbersome to pass from one format to another, even if the models have similar characteristics, like being discrete representations of a figure, or a more elaborated object.

XML[5] 'Extensible Markup Language' is a markup language designed to encode the data in a human-readable as well as machine-readable way. Like its name says it, XML is a language which is mainly concerned with its extensibility, this means that it is a set of rules to represent data in a structured way more than a static language. This makes easy to generate subsets of rules that can represent a very particular database of information. It's extensibility and flexibility is an advantage when representing unstructured data, like the case of HTML.

2.6.1 PML and LML

During the internship multiple formats where considered for the representation of the models, once CamiTK was chosen to be the medium between the model and the simulator, PML 'Physical Model Markup' and LML 'Loads Markup Language' where a natural selection for the representation of the models. Chabanas and Promayon

[5]https://www.w3.org/XML

[13] proposed these generic languages that allowed to represent both continuum and discrete models, and also a library in C++ that allows to manipulate the formats.

The **PML** format regroups concepts that are used on different physical models, briefly those are :

- **Atoms**: this is the structure that holds the 3D coordinates, because this structure allows properties, it can represent also the mass of a spring system, for example, or a node in the finite elements method.

- **Cells** : The cells are the structure that regroups multiple atoms, to this structure it is also possible to associate properties, which means that they can represent a spring in a discrete model or an element from a continuum model with defined Poisson coefficient for each element for example. The cells are described by its geometry this being: triangle, hexahedron, tetrahedron, quad, etc.

The advantage of this generic approach is that gives a granular control over the model, where it can be a standard model or a topologically mixed one i.e Tetrahedra paired whit hexahedra, with different physical or mechanical parameters for each atom or group of atoms.

LML is also a generic description based in XML, that describes the loads that can be applied to the model, these loads can be External forces, pressure, displacements, etc.

- **Target**: The target is any structure defined in the PML file, being this Atoms and Cells or any other structure created by the user, like groups of cells

- **Type**: Defines the kind of force to be applied like pressure, displacement, rotation or just a force.
- **Direction**: this is a 3D vector that defines in which direction the force is going to be applied

- **Value**: This defines the time in which the force is going to be applied, there can be multiple values. It comes in pairs, one for start value and the next one for the stop.

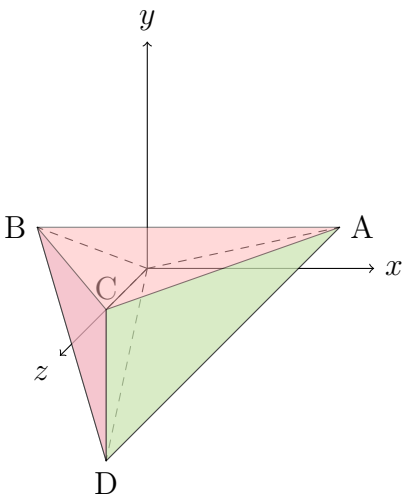
  s

2.6.2 MML

The Monitoring Markup Language 'MML'[1] developed by Deram came as a necessity to follow the data being simulated, it unifies both PML and LML, and adds a set of monitoring tools for the simulations. It is divided in two parts MML_in and MML_out. MML_in describes the input data, that ranges from the simulator used in the process, SOFA in this case, a link to the PML file that contains the geometry to the monitors being registered, such as position, geometric deviation,

```
<?xml version='1.0' encoding='UTF-8'?>
<physicalModel name='Tetrahedron' nrOfAtoms='4'
  nrOfExclusiveComponents='2'
  nrOfInformativeComponents='2'
  nrOfCells='5'
>
<!-- list of atoms: -->
<atoms>
<structuralComponent name='DOF' mode='WIREFRAME_AND_SURFACE'>
<nrOfStructures value='4'/>
<atom><atomProperties index='0' x='0' y='0' z='0'/></atom>
<atom><atomProperties index='1' x='10' y='50' z='0'/></atom>
<atom><atomProperties index='2' x='5' y='8.66025' z='20'/></atom>
<atom><atomProperties index='3' x='5' y='2.88675' z='20'/></atom>
</structuralComponent>
</atoms>
<!-- list of exclusive components : -->
<exclusiveComponents>
<multiComponent name='Exclusive Components'>
<multiComponent name='Elements'>
<structuralComponent name='Tetras-Tetra' mode='WIREFRAME_AND_SURFACE
  '>
<nrOfStructures value='1'/>
<cell>
<cellProperties index='0' type='TETRAHEDRON'/>
<nrOfStructures value='4'/>
<atomRef index='0' />
<atomRef index='1' />
<atomRef index='2' />
<atomRef index='3' />
</cell>
</structuralComponent>
</multiComponent>
</multiComponent>
</exclusiveComponents>
```

(a) PML example code



(b) Simple PML Tetrahedra representation

stability criteria, etc. A complete explanation of the monitors is given in 2.6.2.1. MML_out it's a file that contains the results of the simulation. this can be used as a data base for comparison, due to its capacity to store the different monitors at each simulation step.

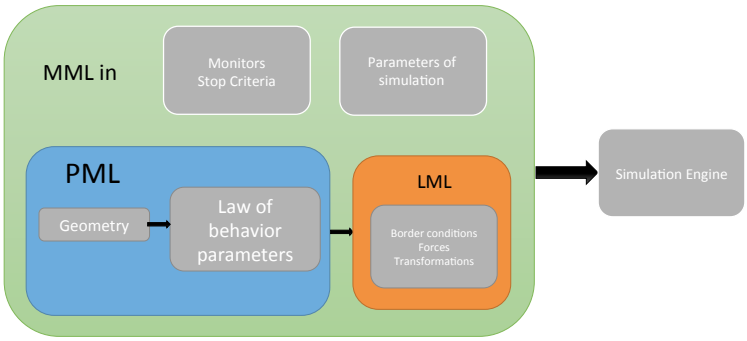


Figure 2.1: Description of PML, LML and MML_in relationships

2.6.2.1 MML Monitors

The MML component include many practical metrics (see 2.2). The component for CamiTK developed by Deram[1] include metrics captured through the simulation process and they serve to multiple proposes, this task is done with the capability included in the MML_in file, the monitors.

A monitor follows the the information of a metric during the simulation. This information can be taken directly from the simulator i.e, positions or can be a calculated metric like the geometric deviation. The monitors have the following control tags:

- Target: it makes reference to the PML atoms or Cells. PML allows to use tags on each structure, so it's possible to target a group of structures by tag.
- Time condition: Like the LML control sequence 'Value', Time allows to choose when the monitored structures are going to start and stop being recorded.
- Monitor or Metric: This is the monitor we want to follow in the simulation(see 2.2). The metrics can be calculated in base a reference mesh, this means that we can monitor the value of a specific structure against an analytical solution for example, or just against different simulations.

One major drawback of the monitoring system is that is topology dependent, this means that if a model follows a dynamic meshing, the monitors would not be able to trace the metric. The monitors are a point-to-point based system, which is only functional when the reference has the exact same topology.

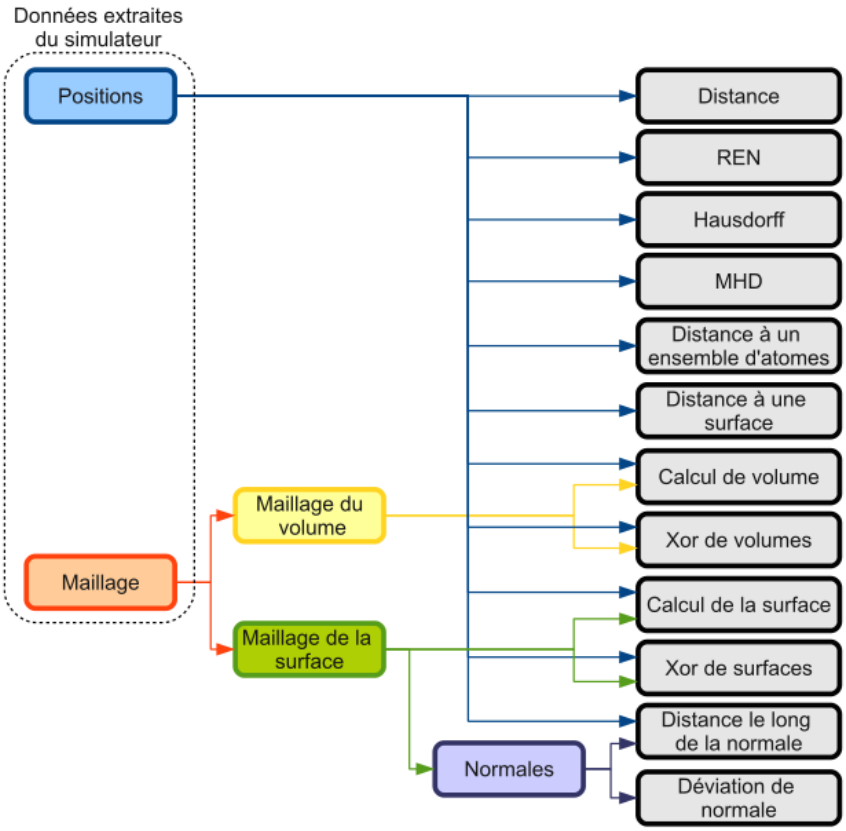


Figure 2.2: Monitors available in the MML [1]

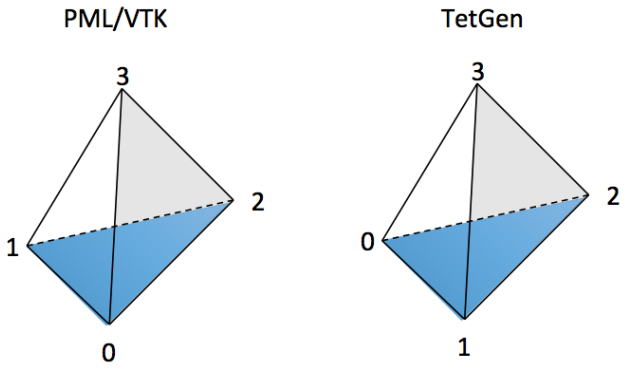
To face this problem of point-to-point monitor, a point-to-element error was defined (see Results), where an *a posteriori error estimate* is presented.

3
Methods

The work done in the intenship was directed to define a workflow to treat bio-simulations in general, this section of the document describes the problems encountered and the proposed solution to those problems.

3.1 Implementation

For the internship all the implementation code was done in C++ and QtCreator. The first reviewed application was Tetgen, which was partially integrated as a CEP 'CamiTK Extension Project'. It allowed to do a tetrahedralization based on a surface mesh, this meshing process was done with the help of VTK[1] functions. In order to be able to re-mesh a volumetric mesh, an extension of the class Tetrahedralize was done, where the VTK_CELL where passed though the library and fed back to tetgen in a VTK object. It is important to note that most file extensions (i.e, .pml, .vtk, .msh, .node etc.) use different conventions to express its atomic elements. Tetgen used an inverted index like shown in figure 3.1a. Not acknowledging this for example would drastically change the topology of the object being processed.



(a) Vtk vs Tetgen indexation

Once the tretrahedralization base process was implemented, the PML extension was included and compiled. CamiTK included an additional set of tools for the pml extension, this tools where no longer compatible with the changes done with the folder structure in the main application and library name changes, this problem was solved trough cmake. Some changes in the cmake files where proposed in the

[1]https://www.vtk.org

forum of CamiTK[2], and where later pushed to the main svn of the application. A similar issue arose with SOFA, where the constant developments of both applications didn't kept backwards compatibility with some of its extensions. This task although technical in essence, was time consuming, moreover the lack of documentation available on the simulation integration obliged a careful study of the complete Sofa and CamiTK source code, where through debugging a solution was found. The solution was also posted to the forum and late in the internship pushed to the SVN main project by the developers of the application along with a revamp of some PML features.

Having a functional integration with the simulator and the PML framework, the passage from PML to VTK was done in the library, this was needed because the simulations only work with the MML extension and not over a simple mesh (i.e vtk, msh). This allowed to remesh PML files and to work with both VTK and PML in the simulator and in the remeshing process. Moreover the large calculations where done over the VTK data structure where its speed and complete set of algorithms gave an advantage to work over the PML data structure, that is based on XML libraries.

3.1.1 Mechanical parameters

For the current work the mechanical parameters used in the simulation carry little to none weight this is in the sense that they do not follow a particular object or set of constraints, meaning that regardless of the behavior of the mesh, our interest is to verify that due to the use of a coarse mesh with little number of elements the behavior is still close to the one projected by the 'golden standard' or the reference, this is to say that regardless of the material, in predefined conditions the mesh should follow similar displacements.

3.1.2 Algorithms

The general algorithm implementation of the main process is described below. The process is simple in general, the main difficulties lay in the file format changes (i.e. mml, pml, vtk, msh) and the mesh processing parameters. Because the work was mainly done with unstructured meshes, not all file formats can be processed the same way, this obliges to do a robust filtering process before being able to pass the data to the tetgen library.

```
Mesh <- load Mesh structure
//process mesh quality parameters and filetype.
process_mesh();
Reference <- load Mesh reference structure
if (plc) {
        extract_surface();
        tetrahedralize();
}
```

[2]https://forge.imag.fr/forum/forum.php

```
else {
  //map of nodes against the reference mesh 'interpolation'
  create_node_map();
  for node=0:lenght(Mesh.nodes) {
        Mesh.get_monitored_data(); //from MML component
        Reference.calculate_ren(Mesh);
  }
  set_volume_constraints();
  tetrahedralize();
}
```

3.2 Point-to-Element Metric

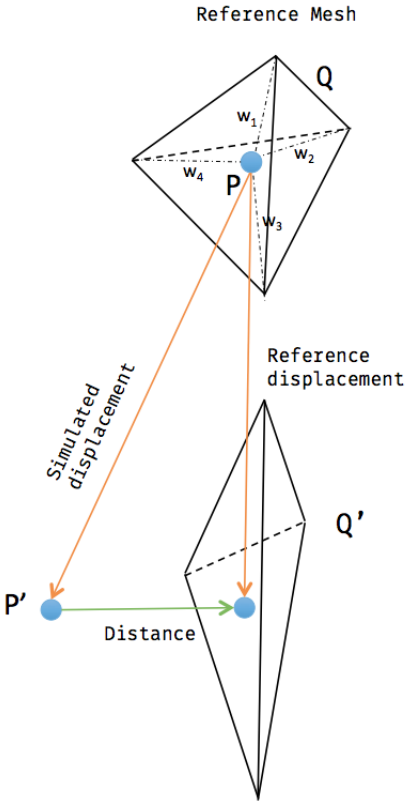
The standard REN 'Relative Error Norm' is a local metric that represents the distance between the simulated positions and the reference positions normalized over the displacement of the reference. If P_0 is the initial position of a point P will be its position at stability after the simulation, Q' will be the position of the point in the reference mesh. The norm is expressed as:

$$N = \frac{\|P - Q'\|}{\|P_0 - Q'\|} \tag{3.1}$$

Because this metric is point-to-point at is definition, the implementation made a variation that allowed to determine the reference position Q' under the assumption that the model is lineal, so it behaves in a predictable way for the forces applied and, that both models (reference and coarse mesh) have the same envelope or surface.

In order to calculate the error a search of correspondence had to be done in the reference mesh, this search was first done with a KD-tree search for the coordinates, but later on changed for more robust VTK search function on unstructured meshes. These VTK functions allowed to do a fast search to locate the closest point in the reference mesh, having this information the tetrahedra containing the point was deduced. A limitation of the method is that if the surface mesh or envelope of the mesh has a different coordinate system or a different surface that the one being simulated, the search won't yield valid distance values, and the enveloping tetrahedron won't be found.

Once all the nodes have a correspondence with a tetrahedron on the reference mesh, its barycentric coordinates are calculated, and stored in a std::map structure, so later we can decide which tetrahedrons of the simulated mesh have the major contribution of error over the simulation. This calculation replaces the value Q' in the equation 3.1.



(a) REN - Point to element interpolation

General Algorithm for the proposed REN calculation.

```
Y : Simulated Mesh
X : Reference Mesh
X0 : Reference Mesh at rest postion
for i:lenght(X0){
        // retrive interpolated values and caluculate the
        // expected simulated position.
        posRef = find_reference_pos(node_map(X0.at(i)))
    if ( distance(posReff, X.at(i) ) == 0 ||
        distance(posRef, Y.at(i) == 0 ) {
        Error = 0.0;
    }
    else {
       REN_point = (distance(posRef, Y.at(i)) /
                           distance(posRef, X0.at(i)) );
    }
        }
}
```

Reference position and map building algorithm.

```
X : simulated mesh
X0: simulated mesh at rest position
Y0: reference mesh at rest position
for i=1:lenght(X) {
        point = X0.at(i) -> find_closest_point(Y0)
        tetra_id = point -> find_enclosing_tetra(Y0);
        point -> calculate_interpolation_weights(tetra);
        node_map -> add(point, weights, tetra_id);
}
```

The calculated error for a tetrahedron was based on the sum of each one of its nodes $\sum \|E_e\|$ where the top 20% of tetrahedra (See Refining Criteria) with the biggest error where meshed. This meshing involved two criteria, a constrained volume reduction and a quality factor, both criteria where passed to Tetgen as parameters.

3.3 Workflow

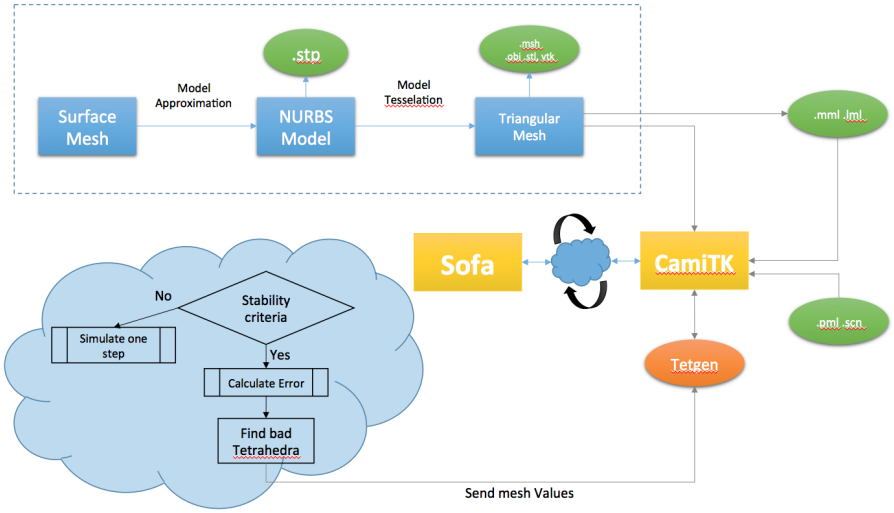
The implemented workflow had as an challenge to group different applications throughout an adaptive meshing process. The exploration of the tools that exist is a long task, where many are specialized over specific domains or don't have an integration mechanism which other tools could benefit from.

The workflow was divided in two parts: the first consisted in the specific-patient data acquisition, this is done generally with scanning technologies that will render a model usually this model is a NURBS representation, from that point to the tessellation of the model into different study cases, require advanced applications that are not often publicly available or require licensing. this part of the project was outside the scope of the internship, but nevertheless some investigation on the possible applications was made. For the reading and modification of the model a 3D modeling software like Rhino3D[3] or OPEN Cascade[4] can be used, this will produce models in BREP format, at least for the study cases, later a tessellation of the surfaces is needed in order to transform the curves in a finite elements mesh. OPEN Cascade provides this functionality. Once the model is tessellated it can be rendered into any available file format that exists in CamiTK, being those (stl, obj, msh, vtk, off).

Once the mesh is in any of the formats mentioned before the second part of the workflow (Process outside of the box see **??**) is done. In order to simulate, the file must be tetrahedralized. This is done trough TetGen, it's important that the meshing process defines various quality measures for the tetrahedra building, later on the behavior of the model can be analyzed based on the resulted meshing. The mesh can be then simulated through the use of a MML file, this file will only contain the simulator, and the metrics we want to simulate. This job is usually

[3] http://www.rhino3d.com/
[4] http://www.opencascade.com/

done but the medical expert, it's his task to point the sensible areas that need to be followed or the constraints or forces that need to be applied in order to do the required medical analysis. This forces are defined in an LML file as explained before. CamiTK will generate a simulation scene (.scn file) for SOFA. This scene will have the required nodes, including physical simulation parameters like, mass, Poisson coefficient, Young Modulus, FEM technique (co-rotational, St. Venant-Kirchhoff, etc.) to the kind of solver to be used. Some made where done to the automatic file generation in order to update the needs of the project. but it can also be manually edited trough the Modeler application of SOFA. Assuming a first simulation of a model exists, it can be then produced a coarse mesh with TetGen based on the tessellated surface, and launch the adaptive process of re-meshing.



(a) Workflow interaction diagram

4

Results

4.1 Validation of the Model

In order to validate the error criteria proposed and the pertinence of the meshing algorithms of Tetgen, some validations where made. A validation was done for an uniform model, where the amount of elements of the mesh was uniformly created, all having the same geometric quality.

4.1.1 Validation of Uniform Model

Test done in a horizontal beam for the uniform model:

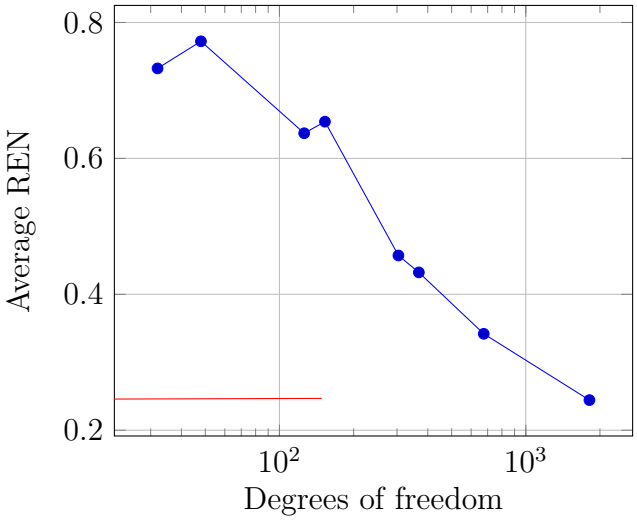


Figure 4.1: Convergence Plot for Uniform Model

As seen in the plot **??** as the mesh have more elements, the calculated REN (point-to-element) showed in the section 3.2, diminish. The table **??** shows the average error values for the points.

Points	Tetras	Avg. REN
32	42	0.732431
48	66	0.772317
126	312	0.637018
153	384	0.654031
304	972	0.457136
368	1188	0.432246
675	2496	0.341864
1813	7776	0.244312
2120	12348	0.000042

Table 4.1: Uniform Meshing Convergence Error

From the previous table it can be deduced that the error calculation trough the proposed REN is valid for an uniform meshing process. A small error is introduced by numerical precision when the reference mesh is compared against itself, as shown in the last element of the table

4.1.2 Validation of Adaptive Model

For the adaptive model three criteria for tetrahedra quality where tested $1, 2 - sqrt(2) - 2$, although any quality measure > 2 is considered bad because it can cause 'silver' tetrahedra, it was tested to verify the speed of convergence and the sensibility of the simulation with different quality parameters. This simulation was done on a Vertical Beam fixed on top that had 3824 vertices and 12267 tetrahedra.

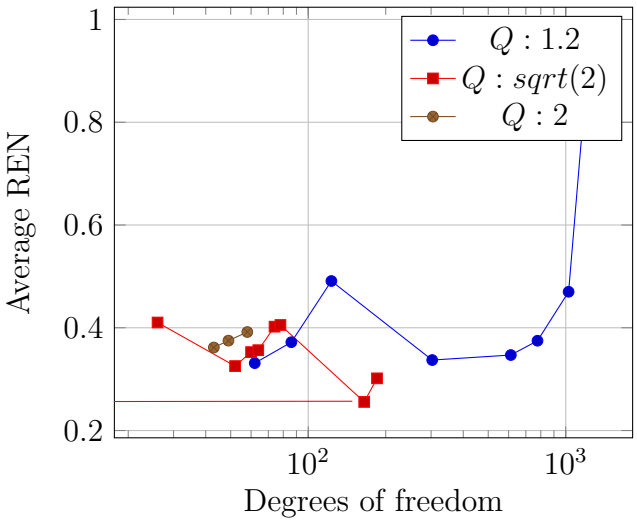


Figure 4.2: Convergence Plot for Adaptive Model

The meshing process was considered finish when no more tetrahedrons were added after a given iteration. For the different parameters the behavior wasn't similar.

The quality measure 1.2 was the one that produced on each iteration by far the biggest amount of subdivisions, unexpectedly this divisions did not contributed

to the convergence of the model, on the contrary each subdivision of the elements incremented the error. For the second quality measure $sqrt(2)$, it does in fact reduce the error, but is not necessarily a steady reduction. Even though the meshing method created the most accurate meshes its solution does not converge, at the end no more tetrahedra are generated to to volume constraints and quality factors. When the volume of an tetra can not be reduced without generating elements with bad quality, tetgen simply ignores the volume constraints imposed by the REN error calculation, and does not modify the mesh.

Points	Tetras	Avg. REN	Points	Tetras	Avg. REN
26	44	0.410276	26	44	0.410276
62	142	0.331183	52	112	0.325443
86	244	0.371915	60	138	0.352635
123	396	0.49101	64	156	0.356394
303	1167	0.337192	74	189	0.402189
612	2503	0.34695	78	206	0.40539
777	3290	0.374771	165	525	0.255699
1026	4566	0.470078	182	575	2.29452
1236	5719	0.953979	185	587	0.301611

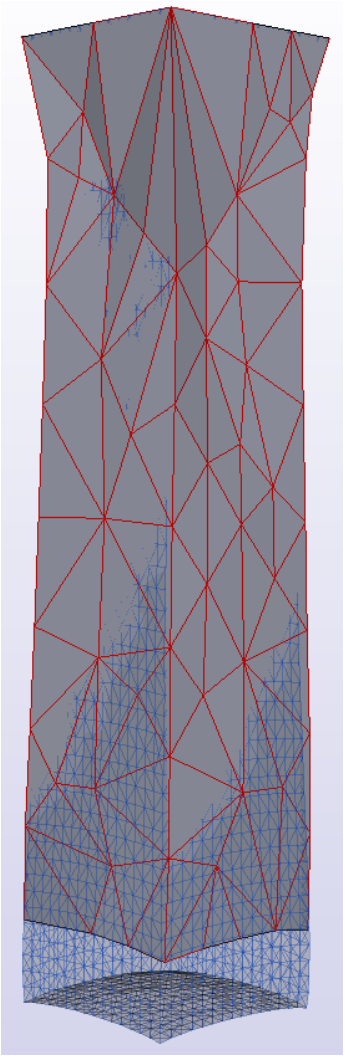
Table 4.2: REN table for meshes with Quality Measure 1.2 and $sqrt(2)$ respectively

As discussed before in the section *2.5.2* a quality superior to > 2 is not considered a good quality. Nevertheless the test was done and it was found that the library stop generating elements very fast due to the inability to respect the quality against the demanded volume reductions imposed by the REN calculation.

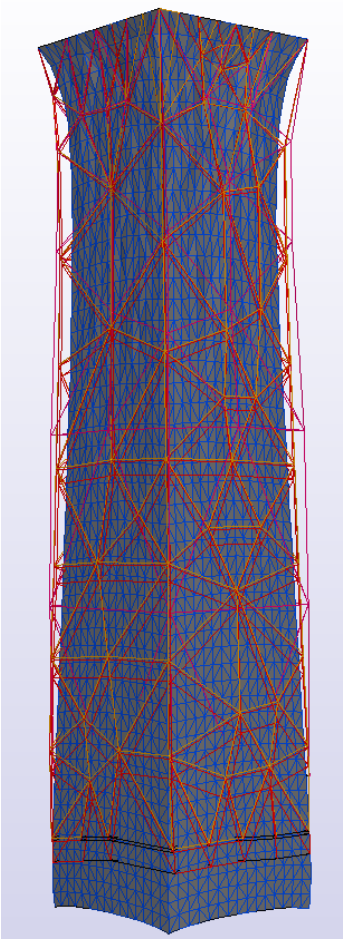
Points	Tetras	Avg. REN
26	44	0.410276
43	92	0.361613
49	111	0.374963
58	143	0.391735

Table 4.3: REN for mesh generated with quality criteria 2

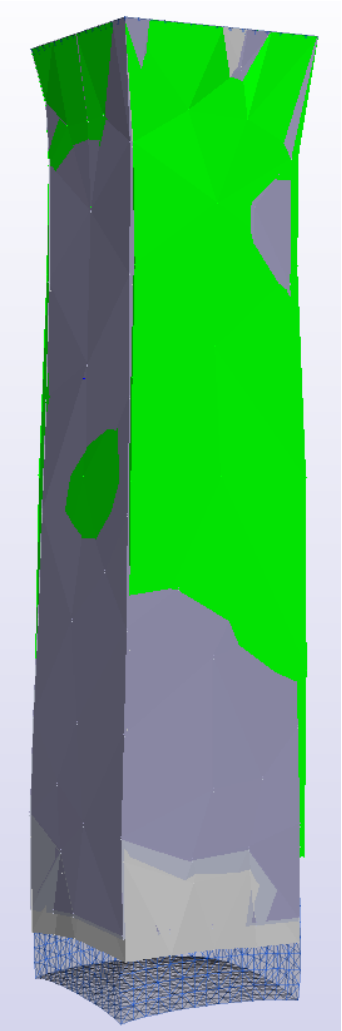
From all the measures tested the one that yielded best results was $sqrt(2)$ but it was still far from the desired solution. the images 4.3d, 4.3b, 4.3c show the results visually. The Blue mesh in the images is the reference mesh against which the models where compared.



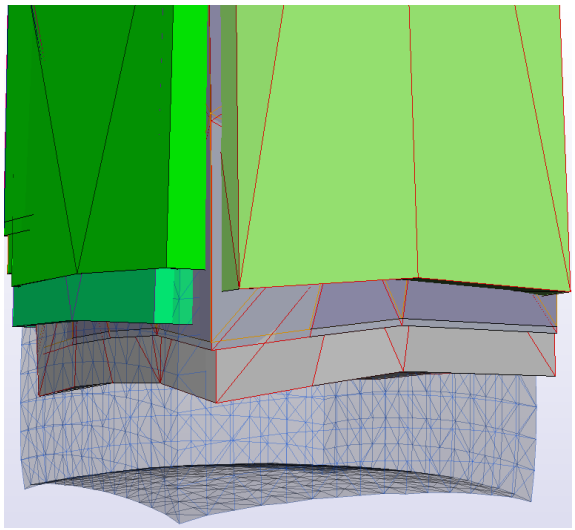
(a) Beam Best appraximation.



(b) Beam Multiple Iterations.



(c) Beam Multiple Iterations overlapped.



(d) Close Up of solutions against the reference (Blue Mesh)

5
Conclusion

An implementation of a workflow was made with the intention of generate coarse model of a given mesh and gradually adapt it to an ideal solution by mesh refinement. Multiple criteria were used in order to make the gradual refinements, one of the proposed criteria was a residual error estimate (point-to-mesh REN) where the local estimation of the error gave the indications of the places where the refinement had to be made, although the criteria was tested with an uniform model, it performed poorly for the adaptive model. The reasons of this performance could be caused by a decisions of refinement made by the remeshing library (Tetgen) or they could also obey to the criteria itself. The analysis of the algorithms of the TetGen library where out of the scope of the internship, that's why is not possible to pinpoint the real origin of the issues encountered in the adaptive meshing process.
Several modifications and suggestions where made trough the CamiTK forum in order to re-establish the tools and plug-ins that where not compatible with the last version of the application, this problems where later solved by the developer group and introduced into the code repository.
In order to have a more efficient communication between formats some tools where introduced as form of plug-ins to the CamiTK application. This included the transformation from file formats like PML and VTK. Many techniques of code development seen in some of the applications were unknown to me, during this internship I had the opportunity to learn from applications like SOFA that used the Visitors Pattern, by CamiTK that used the Signals and Slots and from cmake, which I only used before as an end user. This process of search, research and coding, contributed greatly to my professional and academic formation.

Bibliography

[1] Deram, A. **Environnement Générique pour la Validation de Simulations Médicales**, Graphics. Université de Grenoble, 2012. French.

[2] Price, J. et al. **Lagrangian and Eulerian Representations of Fluid Flow: Kinematics and the Equations of Motion**, 2006 Woods Hole Oceanographic Institution, Woods Hole, MA, jprice@whoi.edu, http://www.whoi.edu/science/PO/people/jprice - Essay

[3] Reddy, J.N. **An introduction to continuum mechanics : with applications**. Cambridge Univ Pr, 2008.

[4] Huangfu, Z. et al. **An Improved Mass-spring Model for Simulation of Soft Tissue Deformation**. Department of Information Engineering, North China University of Water Resources and Electric Power, Zhengzhou 450011, China. 2013

[5] Foudard C, et al. **CamiTK: A Modular Framework Integrating Visualization, Image Processing and Biomechanical Modeling**. ID1., TIMC Laboratory, Joseph Fourier University, France. 2012

[6] Barrea A. et al. **Biomechanical ToolKit: Open-source framework to visualize and process biomechanical data.** Computer methods and programs in biomedicine. Elsevier, 2013

[7] Scott L. Delp. et al. **OpenSim: Open-source Software to Create and Analyze Dynamic Simulations of Movement.** Computer methods and programs in biomedicine. Elsevier, 2013

[8] Faure, F. et al. **SOFA: A Multi-Model Framework for Interactive Physical Simulation.** Soft Tissue Biomechanical Modeling for Computer Assisted Surgery, 11, Springer, pp.283-321, 2012, Studies in Mechanobiology, Tissue Engineering and Biomaterials.

[9] Hang Si. **TetGen, a Delaunay-Based Quality Tetrahedral Mesh Generator**. ACM Trans. on Mathematical Software. 41 (2), Article 11 (February 2015), 36 pages. DOI=10.1145/2629697 http://doi.acm.org/10.1145/2629697 2015

[10] Shewchuk, J. **What is a Good Linear Element? Interpolation, Conditioning, and Quality Measures** University of California at Berkeley, Berkeley, CA, U.S.A. jrs@cs.berkeley.edu

[11] Oden, J. T. and Emkowicz, D. **Advances in adaptive improvements: A survey of adaptive finite element methods in computational mechanics.** In State-of-the-Art Surveys on Computational Mechanics. The American Society of Mechanical Engineers, 441–467. 1998

[12] Jones, M. T., and Lassmann, P. E. **Adaptive refinement of unstructured finite-element meshes**. Finite Elements in Analysis and Design 25, 41–60. 1997.

[13] Chabanas, M. and Promayon, E. **Physical model language: Towards a unified representation for continuous and discrete models** Lecture notes in computer science, vol 3078, pages 256-266. 2004

[14] Budd , C. J., Huang , W., and Russell, R. D. 2009. **Adaptivity with moving grids**. In Acta Numerica 2009, vol. 18. 1–131

[15] Baraff, D. and Witkin, A. **Large Steps in Cloth Simulation** Robotics Institute Carnegie Mellon University COMPUTER GRAPHICS Proceedings, Annual Conference Series, 1998

[16] Flaherty, J. **Course Notes - Finite Element Analysis** Amos Eaton Professor, Rensselaer Polytechnic Institute.

[17] Wicke, M. et al. **Dynamic Local Remeshing for Elastoplastic Simulation** Computer Graphics Proceedings, Annual Conference Series, University of California, Berkeley and Graphwalking Associates. 2010

A
Mathematical Background

Depending on the chosen application, a medical simulation can be defined either by generic data, average data or patient specific data on the patient. For instance a simulator based on the training can use data that are not bound to a specific patient, then a diagnostic or planning simulator must ideally be based on the specific information data of the patient under study.

Generally the last kind of simulator require different phases such as: Acquisition of the patients organ geometry, segmentation of the input, 3D reconstruction, measurement of the mechanical parameters of the soft tissue, application of the deformation algorithm, interactions detection and management, haptic return force management and finally the analysis of the parameters during the simulation process.

In our case, some of the phases mentioned before are out of the scope of the internship and won't be explored through the document. We'll assume that the input data is a 3D surface model that can be given in any physical format, either STP, obj, msh, vtk etc. and for the propose of the research it will represent a generic case of study.

A.1 Soft tissue models

Mathematical models are often used in medical simulations to describe the behavior of soft tissue. This section is barely a scratch on the domain of those methods, but it will lay grounds on the mechanisms used by the simulator during the internship[1].

Generally the deformation models can be based over a physic logic or a descriptive one. The models physically based, describe the intern works of the modeled phenomenon. If the phenomenon at hand is well understood its precision will depend then on the resolution of the mesh. The descriptive models on the other hand reproduce directly the behavior of the phenomenon without any previous knowledge, making its precision depend solely on the skills of the person who made the model and the given parameters or external constraints in order to find the closest visual solution to the reality. In medical simulation the physic models are the most used, because it allows the users to make predictions over its behavior contrary to the descriptive models. There are two main representations in order to describe a deformable object, these are the **Lagrangian** representation that follows the particles over time, regardless of its position and the **Eulerian** representation that tracks a

[1] https://www.sofa-framework.org/

given position rather than the particle itself, this means that that the observation is fixed and we analyse all the particles that pass over our field over the time [2][1].

A.2 Model Types

In the field of soft tissue simulation there are two main physical model categories, those are: continuum and discrete models. The first are based on purely physical sense, that is to say that their mathematical model based on knowledge of the physical processes of deformation of continuous media. The passage of the equations of continuum mechanics to a digital model is through a method of resolution, the most famous being the finite element method. The discrete models are based on a discretization of the model from the mathematical formulation. These models, although always based on reproducible physical processes are more descriptive than continuous models, i.e. Mass-Spring systems.[1]

The next section will introduce the mathematical models in which the simulations are based, it intends to be more a mathematical tool presentation for the basis of most medical simulators, for a complete explanation on the subject see [3].

A.2.1 Continuum Model

Displacements
Upon deformation of a body, each point of which the initial position is given by the vector x_0 is found at a position x. The displacement $u(x)$ for this point corresponds to the vector between the deformed position and the rest position, $u(x) = x - x_0$
Deformation
A body which is just translated, that is to say, the displacement is the same at any point does not undergo a deformation. Deformations at a point are due to variations in the displacement field around this point; so we can characterize by using the gradient of the displacement field. The classic approach to characterize the deformation at a point is the Green-Lagrange tensor, defined by :

$$\mathbf{E} = \frac{1}{2}\left[(\nabla_{\mathbf{X}}\mathbf{u})^T + \nabla_{\mathbf{X}}\mathbf{u} + (\nabla_{\mathbf{X}}\mathbf{u})^T \cdot \nabla_{\mathbf{X}}\mathbf{u}\right] \tag{A.1}$$

Stress Tensor
The stress is an homogeneous variable which characterizes the mechanical actions exerted on the material for a given point and a given direction. This is a second order tensor, with nine components σ_{ij} that completely define the state of stress at a point inside a material in the deformed placement or configuration.

Laws of Behavior
The laws of behavior that govern an object, characterizes the response of the object to external forces. For elastic objects (such as organs) this law is defined by:

$$\sigma = f(\epsilon) \tag{A.2}$$

The laws as shown in [1] consist in 5 major behaviors briefly described below:

1. **Elasticity**:
   Elasticity is the ability of a body to resist a distorting influence or stress and to return to its original size and shape when the stress is removed. Solid objects will deform when forces are applied on them. If the material is elastic, the object will return to its initial shape and size when these forces are removed.
2. **Linear Elasticity** :
   For small deformations, most elastic materials such as springs exhibit linear elasticity and can be described by a linear relation between the stress and strain. This relationship is known as Hooke's law. This law can be stated as a relationship between force F and displacement x.

   $$F = -kx \tag{A.3}$$

   where k is a constant known as the rate or spring constant. It can also be stated as a relationship between stress σ and strain ε:

   $$\sigma = E\varepsilon \tag{A.4}$$

   where E Is known as the elastic modulus or Young's modulus.
   In order to be able to use this law the material must be isotropic (Same behavior in all directions). In terms of Young's modulus and Poisson's ratio, Hooke's law for isotropic materials can then be expressed as:

   $$\boldsymbol{\sigma} = 2\mu\boldsymbol{\varepsilon} + \lambda \operatorname{tr}(\boldsymbol{\varepsilon})I \tag{A.5}$$

   where σ is the stress, ε the strain tensor, I the identity matrix and $\operatorname{tr}(\cdot)$ the trace function.
3. **Hyper-elasticity** :
   For many materials, linear elastic models do not accurately describe the observed material behavior. The most common example of this kind of material is rubber, whose stress-strain relationship can be defined as non-linearly elastic, isotropic, incompressible and generally independent of strain rate. Hyperelasticity provides a means of modeling the stress-strain behavior of such materials. The simplest hyperelastic material model is the Saint Venant–Kirchhoff model which is just an extension of the linear elastic material model to the nonlinear regime. This model has the form:

   $$\boldsymbol{S} = \lambda \operatorname{tr}(\boldsymbol{E})\boldsymbol{1} + 2\mu\boldsymbol{E} \tag{A.6}$$

   where $\boldsymbol{S}$ is the second Piola–Kirchhoff stress and $\boldsymbol{E}$ is the Lagrangian Green strain, and λ and μ are the Lamé constants.
4. **Viscoelasticity** :
   Viscoelasticity is the property of materials that exhibit both viscous and elastic characteristics when undergoing deformation. Viscous materials, like honey, resist shear flow and strain linearly with time when a stress is applied. Elastic materials strain when stretched and quickly return to their original state once the stress is removed.

5. **Plasticity** :
   Plasticity describes the deformation of a material undergoing non-reversible changes of shape in response to applied forces. For example, a solid piece of metal being bent or pounded into a new shape displays plasticity as permanent changes occur within the material itself.

A.2.1.1 Mathematical Model

The mathematical model then that determines the displacements of a 'body' it's given by :

- Behavior Law : $\sigma = f(\epsilon)$
- Relation between deformations and displacements : $\varepsilon = g(\mathbf{u})$ corresponding to the selected tensor (Linear - non-linear)
- Equation of local equilibrium : $div([\sigma]) + \mathbf{f}_{ext_v} = \rho\ddot{u}$
  with $\mathbf{f}_{ext_v}$ being the external volumetric forces, et ρ the volumetric mass.

When this formulation has no analytical solution, Finite Elements Method is used.

A.2.1.2 Finite Elements Method - 'FEM'

The principle of FEM is to provide an approximate solution of partial differential equations, where the idea of the method is to discretize the object in a number of 'finite elements' on which the partial differential equations can be solved. This discretization is translated into a mesh, where the elements are simple geometric shapes (triangles, quads, tetras, hexagons, etc.), and the vertices are nodes. The physical properties are continually interpolated on each element according to the values of the nodes, so the precision of the solution depends directly on the mesh and the chosen interpolation methods. Once the the mesh and the interpolation functions are chosen, the solution follows these steps:

1. Approximation of displacement of the points of each element e as a function of displacement $\mathbf{u}^e$ of its nodes.
2. Calculation of the deformations as a function of the nodal variables.
3. Calculation of the constraints as a function of nodal variables using the behavior laws.
4. Use of the theorem of virtual works. For each element is obtained a relation on the form:
   $$\mathbf{F}^e = [K^e]\mathbf{u}^e \tag{A.7}$$
   where F^e represents the force over the nodes of the element and $[K^e]$ is the stiffness matrix of the element e
5. Sum of the contribution of forces of each element.
   Finally in the static case we obtain a system of the type:
   $$\mathbf{F} = [K]\mathbf{u} \tag{A.8}$$

With u being the unknown vector containing the displacement of all nodes, K the global stiffness matrix obtained by the sum of all element matrices and F the forces applied to all the nodes in the mesh. On the dynamic case we obtain a system of the type:

$$\mathbf{F} = [M]\ddot{\mathbf{u}} + [D]\dot{\mathbf{u}} + [K]\mathbf{u} \tag{A.9}$$

with M the mass matrix and D the damping matrix. The calculated solution on the nodes can be then interpolated over each point of the object.

In the static case if the matrix K don't depends on U the solution can be achieved simply by inversion of the K matrix, or by applying iterative methods like the conjugate gradient. If K matrix depends of U (non linearity), the system can be solved by the iterative Newton-Raphson method. In the dynamic case, the equations are solved by numerical integration, see [15].

A.2.2 Discrete Model

A.2.2.1 Mass Spring systems

This is one of the most known discrete models for deformable object representation. In this system, the mass of deformable object is discretized in n points of mass m_i that are attached to ideal springs (without mass). At each step of time t, each point i has a position x_i. the force F_i in each point es calculated in function of the springs attached and additional external forces. The equation that describes the movement for each point i is :

$$F = ma = m\frac{\mathrm{d}^2x}{\mathrm{d}t^2} = m\ddot{x} = -kx. \tag{A.10}$$

Where k is the stiffness coefficient of the spring (positive constant). F can be decomposed in :

$$F = F_{ext} + F_{int} - c\frac{\mathrm{d}x}{\mathrm{d}t} \tag{A.11}$$

Being $c\frac{\mathrm{d}x}{\mathrm{d}t}$ the damping on the system in relation to the preceding step of t. F_{ext} is for example the weight, and F_{int} are the forces applied over the point i by other particles, and because springs react proportionally to the displacement compared to the resting position, the internal forces are equivalent to :

$$F_{int} = \sum_{j=1}^{n} \frac{x_j - x_i}{\|x_j - x_i\|}(k_{ij}(\|x_j - x_i\| - l_{ij})) \tag{A.12}$$

k is the stiffness and l is the length of the spring, both between points i and j. In A.9 it was shown the particular case for one point, for multiple points the equation becomes a system of multiple equations:

$$F_{ext} = [M]\frac{\mathrm{d}^2x}{\mathrm{d}t^2} + [D]\frac{\mathrm{d}x}{\mathrm{d}t} + [K]x \tag{A.13}$$

with M, D and K being matrices $3x3$ representing the mass, damping and stiffness, and x a $3n$ vector with the positions of all points. The Mass-Spring system is widely used for soft tissue simulation due to it's simplicity of implementation and reduced speed of calculation, but it yields results less realistic. For an extended version of the mass-spring method on soft tissue [4] can be consulted.

A.3 Time Integration Schemes

In order to simulate soft tissue, it is necessary to have the coordinates of each point at each time step. The methods mentioned before allow to reach an equation where the positions of the points in each time step are not explicitly given but implicitly:

$$\ddot{x} = f(\dot{x}, x, t) \tag{A.14}$$

Where $x(t)$ is a vector holding the group of positions at the time t and f is one function that depends on the selected model. Because it can't be solved in an analytical way, we must use numerical integration.

Once we fix a time step dt the goal is to find the approximate value at different $x(\delta t)$. We wan rewrite the second order equation as follows:

$$\begin{cases} \dot{x} = v \\ \dot{v} = f(v, x, t) \end{cases} \tag{A.15}$$

To find the values of $x(t + dt)$ they are different methods of integration. If $x(t + dt)$ is dependent of the values in $t + dt$ then the method is implicit otherwise is explicit (A.15).

A.3.1 Explicit integration

The most used method is the Euler method :

$$\begin{cases} x(t + dt) = x(t) + dtv(t) \\ v(t + dt) = v(t) + dtf(v(t), x(t), t) \end{cases} \tag{A.16}$$

This method is generally stable, but it can present problems of stability when dt is bog

A.3.2 Implicit integration

The Backward Euler method or implicit method is given by:

$$\begin{cases} x(t + dt) = x(t) + dtv(t+dt) \\ v(t + dt) = v(t) + dtf(v(t+dt), x(t+dt), t) \end{cases} \tag{A.17}$$

This method is more costly to calculate, but unconditionally stable, which means we can use larger time steps to find a faster convergence of the system. This method was used in sofa to solve the equations system.