Université de Lille 1 SCIENCES ET TECHNOLOGIES

Thesis submitted to obtain the title of
Doctor of Philosophy of
the University of Lille 1

Physically-based 6-DoF Nodes Deformable Models: Application to Connective Tissues Simulation and Soft-Robots Control

Prepared at **Inria** by

JULIEN BOSMAN

Defended on November 27th 2015

Jury members:

Advisor:	Christian Duriez	Research director, INRIA
Reviewers:	Stéphane Bordas	Professor, University of Luxembourg
	Maud Marchal	Associate Professor, IRISA-INSA
Examiners:	Benjamin Gilles	Researcher, LIRMM
	Hongbin Liu	Associate Professor, King's College London
	Adrien Theetten	Research Engineer, Dassault Systemes
President:	Mathias Brieu	Professor, École Centrale de Lille

Résumé

En simulation médicale, la majorité des travaux était, jusqu'à présent, consacrée à la simulation d'organes, ces derniers étant généralement simulés seuls. Cette situation pose un réel problème car l'influence qu'ont les organes environnants sur les conditions aux limites est négligée. Ces interactions peuvent être complexes, impliquant des contacts mais aussi des liaisons mécaniques dues à des couches de tissus connus sous le nom de tissus conjonctifs ou fasciae. Pour cette raison, les influences mutuelles entre les structures anatomiques sont généralement simplifiées, diminuant le réalisme des simulations.

Cette thèse vise à étudier l'importance des tissus conjonctifs, et d'une bonne modélisation des conditions aux limites. Dans ce but, le rôle des ligaments lors d'une intervention chirurgicale sur la foie par laparoscopie a été étudié. Afin d'améliorer le réalisme des simulations, un modèle mécanique dédié aux tissus conjonctifs a été mis au point. Ainsi, une méthode basée sur la mécanique des milieux continus et un ensemble de nœuds à 6 degrés de liberté a été développée. L'objectif de ce modèle étant de permettre la simulation simultanée de plusieurs organes liés par des interaction complexes.

En outre, les travaux sur les tissus conjonctifs ont donné lieu à la mise au point d'une méthode de modélisation utilisée dans le cadre des robots déformables. Cette méthode permet un contrôle précis, et temps-réel, d'un bras robotisé déformable. En effet, l'utilisation de nœuds orientables a permis de développer un modèle a nombre de degrés de liberté réduit, qui permet de reproduire le comportement de structures plus complexes.

Mots-clés : Simulation médicale, modèles basés sur la physique, tissus conjonctifs, mécanique, nœuds à 6 degrés de liberté, objets déformables, réduction de modèles, robots déformables.

Abstract

So far, most of the research work done in medical simulation has been dedicated to organs simulation. These are generally simulated alone. This raises a real problem as the role of the surrounding organs in boundary conditions is neglected. However, these interactions can be complex, involving contacts but also mechanical links provided by layers of soft tissues. The latter are known as connective tissues or fasciae. As a consequence, the mutual influences between the anatomical structures are generally simplified, weakening the realism of the simulations.

This thesis aims at studying the importance of the connective tissues, and especially of a proper modeling of the boundary conditions. To this end, the role of the ligaments during laparoscopic liver surgery has been investigated. In order to enhance the simulations' realism, a mechanical model dedicated to the connective tissues has been worked out. This has led to the development of a physically-based method relying on material points that can, not only translate, but also rotate themselves. The goal of this model is to enable the simulation of multiple organs linked by complex interactions.

In addition, the work on the connective tissues' model has been derived to be used in soft robotics. Indeed, the principle of relying on orientable material points has been used to develop a reduced model that can reproduce the behavior of more complex structures. The objective of this work is to provide the means to control – in real-time – a soft robot, more specifically, a deformable arm.

Keywords : Medical simulation, physically-based simulation, continuum mechanics, deformable objects, connective tissues, 6-DoF nodes, reduced models, soft-robots.

Acknowledgement

First of all, I would like to thank Stéphane Cotin for giving me the opportunity to start this thesis in his team (Shaman at that time). Thanks to you, I've been able to make a wish come true. I also want to warmly thank my advisor Christian Duriez, for his implication, its help when I experienced difficulties and its kindness. Christian, be assured that you have all my gratitude.

Thank you to all the people I met in the Shaman/Shacra/Defrost teams. This was a great experience, and I particularly appreciated to work in such a good atmosphere. I'm especially thinking to Hugo, for his constant cheerfulness he diffused in the team. I could not forget to mention the person with whom I shared my office during three years: thank you Nazim, It has been a pleasure to work with you, even if your songs and your phone ringing still resonate in my head. Thank you Thor, It has been really interesting to work with you on a topic that was new to me; I wish you the best. Thank you all, Jérémie, Eulalie, Mario, Bruno, Damien, Fred, Alex, Anne…I hope I do not forget anyone.

I also want to thank my family a lot. Thank you for your support, for those family dinners that were essential to balance the stress of this thesis. What I've managed to do is the result of the trust you placed in me since the beginning. Thank you to my unique but nonetheless preferred sister, Justine, who take part in this achievement, by proof-reading this manuscript.

Thank you to all my friends, and especially Arnaud and Vanessa, for all the good time we have had. Thank you for those memorable breaks in Toulouse. I hope to join you very soon.

Finally, the most important of all, I want to thank the woman who shares my life for eleven years now. Your love really propelled me along those years. Thank you my love for always believing in me, thank you for your support, through the good but also through the hard times. A new life full of great projects starts right now.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Why medical simulation ?

Computers capacities and efficiency have been continuously increasing over the past decades. These improvements have led to the omnipresence of computing devices in everyday life. Computer assisted tool are nowadays employed in numerous fields such as entertainment, transports, engineering, *etc.* Among them, the medical field also largely benefited from this evolution, especially thanks to medical imaging.

In this context, medical simulation has been an active research field over the past years. The results already obtained are promising and allow to envision a wide range of applications that are presented in the subsequent paragraphs.

Training

Despite continuously improving techniques and technologies, practitioner skills remain the main factor of success or failure. However, a recent study conducted by the World Health Organization (WHO) estimates that medical errors are responsible for more than 400,000 deaths per year [Jam13]. These facts highlight the importance of a proper training. Indeed, the life expectancy increase contributes to an increase of medical care needs, especially from aging population which requires more physician to be trained. In addition, surgical techniques become more and more complex, increasing the demand for the training of students and surgeons. Traditionally, students are instructed by means of fellowship. During this period, a skilled surgeon progressively teach them medical gestures they will have to perform all along their life. Yet, the main drawback of this kind of training is that it is limited to the availability of the teacher which makes the students training period longer. Moreover, some types of surgery are rare and may be highly risky. This further limits the learning possibilities of new surgeons, and at the same time, hinders already trained surgeons to maintain their skills.

In this context, medical simulation associated to haptic devices could help to answer the practitioner learning and training needs without involving risks of serious injuries or deaths.

Pre-operative planning

In the case of certain pathologies, surgical interventions to be done can be extremely complex and expose the patient to a high risk level. Furthermore, some patients may exhibit atypical anatomy of physiology making the surgery more difficult. Manual planning of medical interventions can take up to several hours, which is difficult for physicians to manage as the number of patients grows. Here, medical simulation used along with patient specific data (issued from medical imaging for instance), may be employed for planning purpose. Simulation thus may help surgeons to decide on the best way to perform the intervention. It can prevent the complications that may occur during surgery, enhancing patient safety.

Per-operative guidance

Another application made possible by medical simulation is the per-operative guidance which consists in assisting practitioner during the surgery. These techniques mixing video processing, image registration and mechanical simulation are used to provide visual cues to surgeons. By means of augmented reality or imaging, it helps surgeons to locate a target such as a tumor in the context of the liver resection for instance. At the same time, it helps the surgeon to avoid dangerous areas like blood vessels and arteries.

Predictive simulation

Sometimes, for a given pathology and a given patient, different types of surgery are available. For instance, this may happen in the case of pelvic floor pathologies. If so, the challenge is to determine which kind of surgery has the best success probability on the long term. Similarly, the ability to predict the aesthetic outcome of a surgical intervention is crucial in the case of maxillofacial surgery. Medical simulation, associated with patient specific data, can simulate anatomical structures and predict their behavior after the intervention. Thus, it may constitute an helpful tool enabling to choose the best kind of surgery.

Medical research and biomechanical engineering

Medical simulation can also be used to study some pathologies like the ones of the pelvic floor [LgDC+13], [VRW+11], [RRD+11]. As a result, simulation could provide a better understanding of the cause of these diseases. Another possible use of medical simulation may be found in biomechanical engineering where it can help to design prostheses [LZJC04]. By doing so, it becomes possible to check how it integrates among the surrounding anatomical structures.

1.2 Requirements for medical simulation

In order to suit the previously mentioned purposes, medical simulation has to exhibit different properties. If certain of these properties are common to all use cases, some other are specific to a particular context:

Realism: this is the main goal of medical simulation since it is the main factor that participates to the user's immersion in the case of training simulation. Hence the need for mechanical models able to precisely reproduce the behavior of soft tissues. This encompasses methods for handling large deformations and for modeling special properties of biological materials.

Fast computations: computational speed is an important feature. However, depending on the application, the allowed timespan for computations ranges from milliseconds to minutes. Indeed, in the case of training simulations, it is essential to maintain the interactivity. Hence,the computations must be done as fast as possible, which implies trade-offs between precision and efficiency. When dealing with pre- or post-operative planning, the computation time constraint is still crucial but lighten as the accuracy is emphasized over speed. In this case, the goal is to provide surgeons with acceptable delays for the planning.

Precision and reliability: in some case, such as pre- and post-operative planning, the computational efficiency is not the main objective. For such an application, there is almost no need for interactivity and the emphasis is put on the precision of the results.

Patient specific data: in the case of planning applications, simulation's ability to incorporate anatomical specificities of the patients is a key point. This enables the integration of realistic and detailed models of the anatomical structures into the simulation.

The main challenge posed by medical simulation is to conciliate these requirements. Indeed, many of them are often conflicting, especially the realism and efficiency needs.

1.3 Thesis outline

Despite the promising advances done in medical simulation, the complete virtual patient's model is yet to come. There are still many avenues for improvements, especially concerning the mechanical modeling of boundary conditions. So far, most of the work has been dedicated to organs simulation, which are generally simulated alone. This raises a real problem as the role of the surrounding organs in boundary conditions is neglected. However, these interactions can be complex, involving contacts but also mechanical links provided by layers of soft tissues. The latter are known as *connective tissues* or *fasciae*. As a consequence, the mutual influences between the anatomical structures are generally simplified, weakening realism of simulations.

This thesis aims at studying the importance of the connective tissues, and especially of a proper modeling of the boundary conditions. To this end, the role of the ligaments during laparoscopic liver surgery has been investigated. In order to enhance the simulations' realism, a mechanical model dedicated to the connective tissues has been worked out. This has led to the development of a method based on material points that can, not only translate, but also rotate themselves. The goal of this model is to enable the simulation of multiple organs linked by complex interactions.

In addition, the work on the connective tissues model has been derived to be used in soft robotics. Indeed, the principle of relying on orientable material points has been used to developed a reduced model that can reproduce the behavior of more complex structures. The objective of this work is to provide the means to control – in real-time – a soft robot made of a deformable arm.

This manuscript is organized in the following way:

Chapter 2 introduces the work related to physically and non-physically based deformable bodies simulation. Mesh-based techniques are presented as well as the improvement proposed to handle large deformations. This chapter also presents work dedicated to particle-based or mesh-free methods and their use. In addition, techniques based on a peculiar types of nodes, such as beams and shells, are detailed as well.

Chapter 3 presents a study dedicated to the influence of the ligaments in the context of liver surgery. This study shows that they play an important role in the organs behavior. This is followed by an introduction to the continuum mechanics and more specifically to the elasticity problems. Then, this chapter shows how the continuum mechanics equations can be derived to be applied to deformable bodies using orientable nodes.

Chapter 4 demonstrates how the mechanical model presented in the previous chapter can be used to simulate the connective tissues. This chapter also shows that this model can be used in a mesh-free or mesh-based scheme without changing the core principle of the method. Then, the time integration method is discussed and the obtained results are presented. Finally, the details about the GPU implementation of the mechanical model are given.

Chapter 5 presents a method based on orientable node aiming at the guidance and the control of soft robots. This chapter demonstrates how structural analysis and domain decomposition can be used to model a deformable arm. It also shows how a reduced model enables for real-time piloting of a soft robot.

Chapter 2

Related work

Surgical simulation emphasizes the need for realism. It thus requires methods for an accurate reproduction of the deformations undergone by volumetric objects. The term “accuracy” refers here to the ability of a model to faithfully reproduce a physical phenomenon. The simulation of deformable objects is also needed for many applications like computer animation, video games and films. For this reason, the topic has been largely studied for decades.

A way to reach this goal is to solve the *partial differential equation* (PDE) which governs elasticity problems given by continuum mechanics. Continuum mechanics is a branch of mechanics that studies the behavior of a continuous volume of matter (fluid or solid) at a macroscopic scale, allowing to ignore intermolecular voids. Solving continuum mechanics equations generally consists in approximating an unknown function. This function lays in an infinite dimensional *Sobolev* space. This makes it impossible to approximate the unknown function, because an infinite number of functions could satisfy the problem. Hence, the infinite dimensional space is discretized by a set of finite dimension, generally a set of polynomial functions. This restricts the number of usable functions used for the approximation, making the approximation possible.

Concretely, this requires the discretization of the domain on which the problem has to be solved, in our case, the deformable object being simulated. Once done, the approximation of the unknown function can be performed on the obtained discrete elements. The type of approximation done depends on the method chosen for the discretization of the domain.

In the following sections, we will review related work about methods of simulation of deformable bodies relying on two different discretization schemes: mesh-based and meshless methods.

2.1 Mesh-based simulation

Non-physically based methods

One of the simplest ways to animate deformable objects is to use the *Mass Spring Method* (MSM). To use this technique, the volume of the animated objects has to be discretized via a meshing procedure. Once this mesh is obtained, each of its edges is considered as a linear spring with a rest length and a stiffness coefficient that are used to compute a reaction force according to the *Hooke's* law. Medical simulation has used MSMs, and early works by Nedel *et al.* in [NT98] presents a simulation of muscle deformation using linear springs and introduce angular springs to handle bending and torsion. Other MSM approaches for surgical simulation may be found in the literature like [MC00] or [MBB$^+$02]. More recently, in [ZYJ$^+$10], Zhu *et al.* use this technique to provide high refresh rates for haptic rendering.

Adopting MSMs for the animation of deformable bodies can be convenient as they are quite easy to implement and more particularly because their fast computation time [LBOK13], allows real-time performance. By real-time, we meant refresh rates that provide users with a seamless simulation (*i.e.* fluidity). This is an important feature for training simulators. On the other end, this method exhibits severe drawbacks making them unsuitable for precise simulations. The main problem of MSMs is that the stiffness coefficient does not represent a physical quantity and, as a consequence, fine tuning of a spring network can be a tedious task. Moreover, MSMs' behavior largely depends on the topology and the resolution of the mesh. For instance, additional springs needs to be added to include shear resistance and bending deformation. Additionally, this formulation does not takes into account the object's volume, which is an important point in the computation of the elastic forces on a volumetric object.

Physically based methods

A commonly encountered way to solve elasticity problems is the *Finite Element Method* (FEM), which discretizes objects using polyhedral elements (mainly tetrahedra and hexahedra). The FEM has been extensively used in many engineering applications (aeronautic, automotive, civil engineering…) for decades. This is a generic engineer tool that can be used to solve, not only elasticity problem, but also, electromagnetic or thermodynamic problem for instance.

Still, engineering FEM applications differ, in some ways, from medical simulation. Despite computational performance is still a concern for engineering applications, these latter generally prefer accuracy over computation times. Indeed, medical training simulation is, as other interactive applications, one of the rare application that requires real-time performance. However, because of its ability to realistically simulate deformable volume objects, FEM is extensively used in medical simulation. As a consequence, medical simulation relies on coarser meshes to maintain adequate computational rate. Another particularity of medical simulation comes from the fact that it deals with biological tissues. Hence, it must rely on computational models able to reproduce the specific behavior of such materials.

The initial work on using FEM in surgical simulation is owed to Bro-Nielsen and Cotin in [BNC96] with a linear elasticity model. To deliver higher computation rate, their method relies on a so-called *condensation* technique. Surface and volume nodes are rearranged and separated so that both sets appear grouped in the final system of equation. This enables the computation of the *Schur complement* of the stiffness matrix giving the relationship between the displacement of surface nodes and volume nodes. Using this method reduces the number of degrees of freedom and thus the size of the system. More recently, Torres *et al.* in [TECO14] adopted the use of condensation technique for the deformation of regular tetrahedral mesh issued from medical imaging. Their method uses a multi-resolution volumetric mesh. In a similar way, the authors merge a coarse and a fine resolution mesh in the same system of equation. By the mean of the condensation, nonlinear shape functions between coarse and fine meshes can be computed, allowing them to interpolate the displacements of the fine mesh from the ones of the coarse mesh. These approach can be qualified as *model reduction* techniques since only a reduced set of degrees of freedom are used to compute the whole object's behavior. Such a method is also used by Kerfriden *et al.* in [KGRB12]. In this work, the authors use a domain decomposition approach and select a reduced set containing the most representative deformation modes off-line. These are then used to accelerate the on-line simulation of deformable object.

Handling large displacements

Despite the FEM effectiveness when dealing with elasticity problems, certain situations are likely to produce abnormal results. Indeed, linear elasticity FEM simulations may fail at handling large displacements, resulting in a poor accuracy and/or unrealistic behaviors. Yet, training medical simulations often face large deformations of anatomical structures. These are interactively induced by users through virtual surgical devices. As a consequence, numerous works have been conducted to address these issues.

When objects are subject to large deformation, linear elasticity finite element simulation may result in artefacts such as volume increase. These anomalies are due to geometrical nonlinearities induced by the rotation of the elements. These rotations are mistakenly considered as displacements by the linear model, leading to unrealistic results. To address this issue, many solutions have been proposed. In [MDM$^+$00], Müller *et al.* introduce a method called *Stiffness Warping* dedicated to the linear elasticity model. The core idea of this work is to estimate the rotation of each element of the mesh and to then rotate their stiffness matrix which is, in such a case, constant. This procedure can be seen as an equivalent of the rotation of the displacement vector in the element frame to perform nodal forces computation.

Another method has been proposed by Felippa in [Fel00] with the *Corotational* method. The principle of this method is to decompose the displacement of each vertices of an element into a pure rotation and a linear displacement. In this way the displacements are evaluated in the local frame of the element, making it possible to discard its rotation in the computation and avoids the aforementioned artifacts. In [NPF05], Nesme *et al.* explore rotational invariance of FEM based simulations by the mean of QR decomposition for rotation estimation. Similarly, [GW08], Georgii and Westermann improve the stability of the corotational formulation for large deformation by using an energy minimization approach to extract rotations.

Element inversion is another problem that can be encountered when working with FEM simulation. It occurs when a vertex of a tetrahedron is pushed strong enough to make it go through its opposite face. This type of problem is generally induced by collisions or user interaction. In [ITF04], Irving *et al.* address this problem by diagonalizing tetrahedra deformation gradient as well as providing an *ad-hoc* modified version of the *neo-Hookean* constitutive law.

Another issue that may arise when using the FEM is the non-conservation of elements volume when they are deformed. This problem is quite crucial in the field of medical simulation since a lot of anatomical structures like organs and muscles are nearly incompressible. It results from the high quantity of water these tissues contain. Irving *et al.* offer a solution to this problem in [ISF07] by adding a pressure term an adopting an approach similar to the one used in fluid dynamics to obtain a divergence-free velocity field.

Handling topological changes

The ability to remove matter or to cut objects is capital topic in medical simulation, especially for incision and tumor resection. In [VVW04] Vigneron *et al.* propose to use the *eXtended Finite Element Method* (XFEM) usually used in fracture mechanics [BMUP01]. This method allows to use finite element meshes without explicitly modeling the discontinuities (*i.e.* the cuts), avoiding mesh refinement of remeshing. This can be done by locally enriching the polynomials used for finite element approximation. A new approach is proposed by Kaufmann *et al.* in [KMBG08] who use *discontinuous Galerkin* FEM. In this method, discontinuous shape functions can be used as well as arbitrary polyhedral elements which are not restricted to tetrahedron and hexahedron. This make the cutting as it does not involve complex remeshing operations around the cutting plane. In [CKD13], Courtecuisse *et al.* demonstrate a method for simulating cuts in soft tissues discretized using tetrahedral elements. Another methods that may ease the handling of topological changes are the meshfree methods that will be discussed in a later section.

Hyper-elasticity

Generally, organs and other soft tissues exhibit complex properties such as anisotropy, nonlinear elasticity, porosity or viscoelasticity. In order to faithfully reproduce the mechanical behavior of anatomical structures, several authors have developed methods to overcome the limitations of linear elasticity by using more complex constitutive laws. A constitutive law is a function that relates strain and stress and characterizes the behavior of a given material.

A way to reach this objective is to rely on nonlinear hyperelastic materials which should not be confused with geometrical nonlinearities. In [PDA03], Picinbono *et al.* propose a real-time nonlinear elasticity model including anisotropic behavior based upon Mass Tensor Model. Their method is suitable for large displacements by enforcing rotational invariance. In this work, real-time performance is achieved by mixing linear and nonlinear elasticity depending on displacement magnitude. As a result, nonlinear elasticity is only computed on nodes undergoing a large displacement

Some works propose to use more complex constitutive laws, more adapted to soft tissues simulation. This is the case in [JWM10] where Joldes *et al.* proposed to use a neo-Hookean material on an hexahedral mesh or in [ODYK12] where Ogata *et al.* adopt a *Mooney-Rivlin* material on tetrahedral mesh.

In [ZWP05], Zhong *et al.* propose a method for probing simulation on nonlinear elastic models. Their approach relies on a relationship function between surface and volume nodes which is used to perform polynomial interpolation between the displacements of the surface nodes in contact with the probe and the interior nodes.

Some materials have their properties evolving with time when they are subject to displacements. In [TCC$^+$08], Taylor *et al.* present a method for simulating anisotropic visco-hyperelastic materials. Furthermore, the authors introduce a time dependent stress relaxation term in their model allowing to simulate the stiffness decrease over the time.

In the approach proposed by Marchesseau *et al.* in [MHC$^+$10b], the formulation of the strain energy function is rewritten to facilitate the computation of the nodal forces, independently of the chosen constitutive law. This method leads to a faster computation of the stiffness matrix, even for more complex constitutive laws, making real-time simulation of hyperelastic material on coarse meshes possible. In addition, their work models visco-elastic and porous materials.

Although the use of hyperelastic models enables more accurate simulations, it may entail some problems. Complex constitutive laws may depend on several material parameters which are difficult to estimate, complicating the setup of the simulations harder. Those parameters may be difficult to estimate for different reasons. First, soft tissues sampling is strictly regulated by the law and by ethics and requires volunteer donors. Hence, its is difficult to obtain biological tissues samples to perform mechanical experiments. Second, soft tissues properties rapidly change after the death [MNjFS14], hindering an accurate estimation of living tissues parameters. To alleviate this problem, methods have been developed to perform *in vivo* material parameters measurement. In [SCB$^+$09], Schiavone *et al.* use an aspiration device to measure brain's material parameters. Such a type of device is presented by Promayon *et al.* in [PLC$^+$12].

An alternative way to estimate these parameters is to rely on FEM *inverse simulation*, whose principle is to reformulate the elasticity problem by considering the material parameters as the unknowns, supposing that the displacements are known beforehand. Among these techniques, *elastography* is employed by Eskandri *et al.* in [ESRB11] to estimate the parameters of viscoelastic soft tissues. In this work, displacements are generated thanks to an external stimulus and recorded using medical imaging (MRI) and the problem is solved using a *quadratic programming* (QP) approach. Similarly, Coevoet *et al.* in [CRL$^+$14], propose a real-time inverse simulation method dedicated to radiotherapy planning. Real-time performance is achieved by projecting the mechanical model into a restrained constraint space, enabling for faster solving of the QP problem.

Yet complex constitutive laws have been carried out in order to reproduce soft tissues behavior, the validity of the results of the simulations that used them needs to be assessed. In [KCO$^+$03], Kerdok *et al.* describes an experimental device and method used to validate the results of a FEM simulation of soft tissues. The authors propose to create a silicone rubber cube, whose material parameters are known, which embeds a regular grid of markers. MR images of the cube are then recorded while it undergoes a compression test with precise load values. The study then compares the deformations experienced by the silicone cube with the result of a FEM simulation using the same parameters.

Real-time simulation

The downside of the FEM's accuracy is its high computational load that may hinder real-time applications. To alleviate this problem, different techniques reducing of the computational cost of the simulations have been developed. For instance, in [BNC96], Bro-Nielsen and Cotin are able to reduce computation times by taking advantage of the linear elasticity model, allowing them to precompute the inverse of the displacement matrix. As a result, a substantial part of the computational load is deferred at initialization.

Another method from Debunne *et al.*,in [DDCB01], proposes to use an *adaptive* level of detail technique in which a volumetric mesh of the object embeds a set of "non-nested" meshes. Depending on a quality criterion, the method automatically switches from one mesh to another to perform force computations.

However, when dealing with hyperelastic materials, the stiffness matrix is no longer constant and thus, the precomputation approach does not hold. In order to speed up computations when using more elaborated constitutive laws, Barbič and James rely on *reduced coordinates* models to perform simulations at higher frequency for haptic feedback. The principle of their technique is to use modal analysis, allowing them to select the most significant deformation modes. The latter are then combined to simulate the deformations of the objects.

A similar approach is employed by Choi *et al.* in [CWK07] with the so-called *modal warping* for the simulation of thin shells.

When the elasticity is solved using an iterative method, like the *conjugate gradient* (CG) as described in [She94], preconditioning techniques can be used to improve the convergence rate. Its principle is to multiply both sides of the linear system by a matrix which is close to the inverse of the system matrix. Such a method is used in [CADC10] by Courtecuisse *et al.*, who propose an asynchronous preconditioner, recomputed periodically in a background thread, based on the stiffness warping technique. With this approach, the performance considerably improves, enabling real-time simulation of large deformations.

Another method improving the convergence rate is to rely on multi-grid methods as done by Georgii and Westermann in [GW08] on hexahedral lattices. This approach is also an iterative process based on the use of grids of different resolutions. The problem is first solved on coarse grids enabling fast computations, leaving a correction factor that is transfered to a finer grid. Since the problem is partially solved on coarser grids, only a few iterations are carried out at fine levels, leading to shorter computation times.

In [NKJF09], Nesme *et al.* propose a method embedding highly detailed model into coarse hexahedral lattice enabling interactive frame rates. The particularity of this method is that it is able to take partially void cells into account when computing the lattice stiffness. In addition, their approach enables to preserve distinct behavior of disconnected structures, even if they fall inside the same cell.

Time integration

To provide the simulation with a dynamic behavior, the problem must be integrated over the time. To do so, the equation stated by the *Newton's 2nd law* must be solved. This is done thanks to the internal forces obtained from the elasticity problem solving. The temporal integration scheme can be split into two categories −*explicit* time integration and *implicit* time integration – both being used in the simulation of deformable bodies.

Explicit time integration, also known as *forward* Euler integration computes the state, in this case the positions, of the system at the next time-step based on the state at the beginning of the current time-step. MSMs generally rely on this type of integration scheme.

Among the methods for simulating deformable bodies is the Mass Tensor Model (MTM), which is a technique mixing the FEM and MSM that has been introduced by Cotin *et al.* in [CDA99]. A stiffness matrix (tensor) is computed for each element as in FEM, but instead of being incorporated into the global system matrix, it is used to add internal force to the element's nodes in an iterative manner. Time integration is performed explicitly and, as a consequence, the method offers a good compromise between accuracy, derived from FEM and speed issued from MSM. The MTM approach has also been used in the field of maxillofacial surgery by Mollemans *et al.* in [MSNS05] to predict soft tissues deformation after an intervention.

In [MJLW07], Miller *et al.* propose to rely on a *Total Lagrangian Explicit Dynamics* (TLED) FEM simulation of soft tissues using hexahedral elements along with a hyperelastic constitutive law. The total Lagrangian approach refers to the fact that nodes displacements are, at any time in the simulation, evaluated with respect to the object's rest, *i.e.* undeformed, configuration. A similar method is employed by Taylor *et al.* in [TCC$^+$08] for the simulation of anisotropic viscoelastic soft tissues or by Joldes *et al.* [JWM10] in the context of brain surgery.

The explicit time integration is advantageous in the sense that, because it does not involve additional derivations of the internal forces, fast computations are possible. In return, this kind of scheme is conditionally stable and requires small time-steps to maintain the stability of the simulation. For instance, if we consider a mass-spring system, the new position obtained through an explicit integration with too large a time-step may overshoot the rest position of the spring. In other words, an additional energy is introduced into the system and at the next time step, this phenomenon will occur again and will finally lead to the explosion of the system.

In parallel, implicit time integration, among which the *backward* Euler method, bases the computation of the new positions upon the state of the mechanical system at the end current time step. It results in an algebraic system of equations that has to be solved at each time step. Depending on the nature of the force to be applied to the nodes, this system is either linear or nonlinear.

Implicit time integration has been introduced in computer graphics by Baraff and Witkin in [BW98] for cloth simulation purposes. This integration scheme has also been used in a context of soft tissues simulation as in [MHC$^+$10b] where Marchesseau *et al.* use implicit integration along with hyperelastic material. Similarly, Allard *et al.* in [ACF11] propose an implicit FEM solver for surgical simulation. In addition, Dequidt *et al.* use implicit integration in the context of interventional radiology simulation.

Unlike explicit integration, implicit integration is unconditionally stable, even for large time steps. In addition, it enable equilibrium error control as discussed in [CAK$^+$14]. Such a property is useful, especially when user interaction is included in the simulation. However, implicit integration comes at the price of a higher computational cost. Indeed, internal forces at the end of the time step must be determined. This amounts to solve a nonlinear problem, using for example, the *Newton-Raphson* method. To do so, internal forces at the end of the time-step are approximated using a *Taylor expansion*, requiring the linearization of their expression. Hence, internal forces expression must be differentiated with respect to the degrees of freedom of the deformable object. Then, the problem is solved iteratively to obtain the new accelerations. This is costly, especially when dealing with nonlinear material because this procedure as the be repeated at each time-step.

2.2 Particle-based simulation

The result of a FEM simulation is largely influenced by the quality of the volumetric mesh. Yet, some objects with irregular topology and/or thin parts and details are prone to produce *ill-shaped* elements, *i.e.* very flat tetrahedra or deformed hexahedra. These elements are numerically less stable and degrade the conditioning of the stiffness matrix, resulting in a loss of accuracy. One way to deal with this problem may be to increase the resolution of the mesh. This will however lead to a higher number of element and negatively affect the performance.

Meshless methods, also referred to as Point Base Animation (PBA) in computer graphics, are used concurrently with FEM. PBA is another discretization scheme that is not affected by the problem mentioned above. While in a mesh-based approach the object's nodes have a structured connectivity determined by the edges of the elements, the meshless approach relies on an unstructured connectivity between nodes. Generally this connectivity is given by a *shape function* which encodes, for a given point, the influence of its neighbors. In the case of deformable objects simulations, these shape functions, allow the interpolation of the nodes displacements as well as the subsequent internal force computation.

Non physically-based methods

Over the past years, authors have proposed several methods for the animation of meshless objects. In [MH05] Müller *et al.* propose to use a method named *shape matching* to animate deformable object. The main idea of this method is to decompose the simulated object into a set of overlapping clusters of particles. When the object undergoes a deformation, an optimal rotation is estimated for each cluster using a *polar decomposition*, with respect to its rest configuration. Particles are then pulled toward their matched rest position using a mass-spring like technique. A similar method is shown by Müller and Chentanez in [MC11] the difference being that it computes rotations per particle instead of computing them per cluster as done in [MH05].

In [CBP05], Clavet *et al.* demonstrate a particle-based method for simulating viscoelastic fluids. Their method is based on the ability to dynamically add, remove or adjust springs acting between particles so as to reproduce convincingly various effects such as plasticity, elasticity or adherence.

However, regardless of their relative simplicity and computational efficiency, these methods are not well suited to surgical simulation. Indeed, they are not physically-based and, as mass-spring systems, these methods rely on parameters which have no physical meaning. This makes them difficult to use for the reproduction of soft tissues' behavior

Physically-based methods

Smoothed Particle Hydrodynamics

Among the meshless methods, the *Smoothed Particle Hydrodynamics* (SPH), introduced independently by Gingold and Monaghan [GM77] and by Lucy [Luc77] is one of the oldest technique. The SPH were initially intended for astrophysical problem such as the simulation of the formation stars and other celestial bodies. This method finds its origins in *kernel* approximation techniques developed in statistics. Its principle is based on the use of compactly supported radial basis functions and provides a rather simple way to approximate values and spatial derivatives of a scalar or vector field. In this approach, forces are computed from the inter-particles interactions with the neighborhood.

Years later, Cani and Desbrun, in [DC96], brought the SPH to the computer graphics community for the simulation of highly deformable and inelastic objects like fluids. The SPH formulation is adapted so that it includes pressure/cohesion and viscosity force through the use of better suited kernels. In [SP07], Solenthaler *et al.* propose a unified simulation of fluids and solids relying on SPH. In their approach, solid objects are also modeled using particles, allowing them to achieve various effects like melting and solidification by modeling heat transfers. They also provide a method for the simulation of deformable objects using SPH based on the linear elasticity Hookean constitutive law.

In [BIT09], Becker *et al.* propose to enhance the simulation of deformable using SPH by adapting the corotational formulation issued from the FEM. This method reuses the idea of the previously mentioned shape matching to estimate rotations. The force to be applied on the particles is calculated using linear elasticity and, thanks to the corotational formulation, the method can handle large deformations. It is interesting to note that, because compactly supported kernel are used, this method can handle collinear and coplanar particles distribution. In parallel, in [GBB09], Gerszewski *et al.* adapt the SPH to the simulation of elastic and plastic deformable objects. This work relies on the approach proposed by Bargteil *et al.* in [BWHT07] for modeling plasticity. However, the use of a meshless approach avoids complex and costly remeshing operations.

In the context of the SPH, only a few works deal with nonlinear dynamics. Hosseini *et al.* in [HMH07], and later Paiva *et al.* in [PPLT09] propose a way to simulate *non-Newtonian fluids* whose viscosity is not constant. Although not related to nonlinear elasticity, their respective works model the nonlinear relationship between the shear strain experienced by the particles and their shear rate.

Despite its popularity, the SPH method exposes several shortcomings. Classical SPH suffer from a limited accuracy caused by the lack of *consistency* also known as *completeness* or *reproducibility*, which is the capacity of a method to exactly reproduce a polynomial function up to a given order. Besides, in its classical form, the SPH is not able to reproduce constant functions [NRBD08]. As a consequence, the SPH fails at properly interpolating scalar or vector fields.

This problem was addressed in many works, leading to enhancement like the *Corrected SPH* [KB00] or similar techniques like the *Reproducing Kernel Particle Method* (RKPM) [LJ95]. These methods introduce a correction term in the SPH equations enabling first order consistency.

Another problem that may arise when using SPH in the context of deformable object simulation is *tensile instability* [SHA95], [MSH04]. This phenomenon may appear when particles are slightly disrupted during large deformations, and especially elongation. When this occurs, negative pressures appear, leading to particles clumping as they try to recover their rest density.

Another shortcoming encountered when using SPH is the handling of the boundary conditions. Indeed, the SPH were originally designed for unbounded problems in astrophysics and. In this type of formulation, the density of a particle is computed from the distance to its neighbors. Yet, the deficiency of particles near the boundaries leads to an underestimation of the density at the frontiers of the object. In [SB12], Schechter and Bridson propose a solution to this problem. They use a technique known as *ghost particles* which consists in the addition of a layer of fake particles near the boundaries of the domain.

The SPH also suffers from a compressibility problem making the simulation of highly incompressible materials difficult. To address this issue, Solenthaler and Parajola propose in [SP09] an iterative method to update the position of the particles in order to maintain their rest density. In [BLS12] Bodin *et al.* propose another approach that models the incompressibility as a constraint on the inter-particle distance. The incompressibility is then enforced through the use of *Lagrange multipliers* obtained by solving the constraints using a *Non Linear Complementary Problem* (NLCP) solver.

Meshless Galerkin methods

Most of meshless methods developed to simulate the behavior of deformable bodies, come from the computational mechanics community. Nayroles *et al.* were pioneers in the study of meshless methods. They introduced the *Diffuse Elements Method* (DEM) [NTV92], presented as a generalization of the FEM. It replaces the interpolation on an element by a *least-squares* procedure on a small domain in order to provide a more continuous approximation. This idea is reused and enhanced by Belytschko *et al.* in [BL94], by carrying out a new meshless method named *Element Free Galerkin* (EFG). The EFG also relies on a *Moving Least Squares* interpolation to build test and trial functions for the weak form of the problem. However, the authors present a more complete way to compute the derivative of the interpolants, and propose to enforce boundary conditions using Lagrange multipliers. Later, Organ *et al.* propose in [OFTB96] to model discontinuities like cracks in the EFG framework using the *diffraction method* inspired by the way the light diffracts. Its principle is to modify the shape functions near the discontinuities so that the mutual influence of particles on both of the discontinuity sides vanishes.

In [MKN$^+$04], Müller *et al.* propose a method in which displacements and their derivatives are approximated using *Taylor series* and linear basis MLS. The internal force computation is physically-based because of the use of strain energy function, analog to the one encountered in FEM. In the same reference, the authors also include methods for simulating plasticity effects and for handling topological changes which is made easier by the meshless approach. In [RBX04], Rabczuk *et al.* introduce a particle-based method for nonlinear materials based on *Lagragian kernels* to overcome accuracy and stability problems similar to the ones encountered in a SPH framework. Choosing a Lagrangian particle method implies that the RBF used in the approximation are computed in material coordinates instead of spatial coordinates and cancels the tensile instability.

Many works have investigated the use of meshless methods in the context of medical simulation and biomechanical engineering [DCC+05]. A real-time simulation of nonlinear soft tissues for surgical simulation has been proposed by Lim and De in [LD07] using the *Method of Finite Spheres* (MFS). This method is, in many ways, similar to an EFG method using spheres as integration support. The work conducted by Horton *et al.* in [HWJM10] proposes to use a meshless approach for surgical simulation by using a method analogous to the EFG relying on a neo-Hookean model, and compares the results to FE analysis. In [HK08], Hieber and Koumoutskos propose a similar meshless approach based on a Lagragian particle method for the simulation of nonlinear elasticity dedicated to the modeling of soft tissues.

One advantage of the meshless methods is that by removing the need for complex and costly remeshing operations, they can handle topological changes. This feature is used by Steinelann *et al.* in [SOG09] for the cutting of deformable objects simulated using the method proposed by Müller in [MKN+04]. Their approach uses a connectivity graph between particles that is dynamically updated by removing the edges crossed by the slicing plane. Jung *et al.* in [JL12] also use the connectivity graph approach, where they propose a real-time cutting simulation of meshless object. The use of *Bounding Volume Hierarchy* (BVH) tree speeds up the simulation. One interesting characteristic of this method is its ability to detect totally disconnected parts of the simulated objects. In [LHLW11], Liu *et al.* propose a meshless method for the animation of brittle fractures relying on the *Meshless Local Petrov Galerkin* (MLPG) method allowing the authors to simulate cracks without visual artifacts. The simulation of fractures can also be done using the enrichment of the shape functions as done in the XFEM. This enrichment allows to model the discontinuities of the displacement field in the fractured material. Such a type of approach has been proposed by Duflot in [Duf06] and by Barbieri *et al.* in [BPMT11] using weight function enrichment. These weights are added to the *intrinsic basis* which is the polynomial basis used for the construction of the interpolants. As a consequence these methods do not require the introduction of additional degrees of freedom. In [BZR07], Bordas *et al.* use an extended EFG method for simulating fracture in nonlinear solids. However, unlike the works previously mentioned, this is done by enriching the MLS approximation using an *extrinsic basis* thus needing the introduction of additional DoF.

Other techniques for handling object splitting use mixed mesh-based/mesh-free approaches. For instance, in [SS07], Sifakis *et al.* propose to embed a point cloud into the mesh of the simulated object in ease the treatment of fractures and collisions. For the same purpose, a similar approach is adopted by Yang *et al.* in [YLW$^+$14] where the objects are modeled as hexahedral lattices and the particles are simulated using EFG method. In this work, the meshless approach also allows the authors to simulate heterogeneous materials.

Another method that can be used to simulate objects subject to large deformations or topological changes is the *Material Point Method* (MPM). Such an approach is an extension of the Particle-In-Cell (PIC) method, used for fluids, to highly deformable solids. This type of method belongs to the *Arbitrary Lagrangian-Eulerian* methods which rely on both Lagrangian and Eulerian representation of the continuum. More precisely, the particles (Lagrangian description) move through a grid (Eulerian description). The physical properties of the particles (mass, strain, stress, *etc.*) are mapped to the grid which is use to solve the governing equations. The results are then transfered back to the particles to update their state. The MPM has been used by Bardenhagen *et al.* in [BBS00] for the simulation of granular materials. More recently, Stomakhin *et al.* in [SSC$^+$13] used the MPM for snow simulation with an elasto-plastic constitutive model allowing for dislocation and cohesion effect.

Meshless and implicit time integration

All the meshless approaches previously mentioned using SPH, RKPM, EFG, *etc.* rely on an explicit time integration scheme and whereas implicit time integration of particle methods are rather rare.

However, a few examples can be found in the literature.n In [GQ05], Guo *et al.* propose a meshless approach to simulate deformable objects based on continuum mechanics and implicit time integration. Similarly, the method demonstrated by Gilles *et al.* in [GBFP11] proposes to simulate sparsely sampled deformable objects using this integration scheme. In [ZLKW13], Zhou *et al.* propose a meshless approach for the simulation of elastic objects including plasticity effects. Their work relies on an implicit integration to provide a stable simulation, even for very stiff objects.

Boundary Element Method

Another method that may alleviate the problem of meshing volume objects with complex topologies is the *Boundary Element Method* (BEM). In this approach, only the boundary of the domain need to be meshed. The problem is then formulated and solved onto the boundaries of the domain. As a consequence the dimensionality of the problem is reduced by one by solving the problem on a surface in 3D or on a curve in 2D. The solution at any interior point of the domain, can be then approximated from the solution on the boundaries. Hence, the BEM is often presented as more efficient, in terms of computational resources, than the FEM because there is no storage or computations on the interior point. The BEM has been used in the context of the surgical simulation by Koppel *et al.* in [KC08] for organ simulation. This approach has also been used by Wang *et al.* in [WBJ$^+$09] to simulate the behavior soft-tissues containing a tumor.

2.3 6-DoF nodes mechanical models

The vast majority of the methods cited previously aims at simulating volume deformable objects. However, for some types of objects, methods based on volume elements are not the most efficient approach. Indeed, in some cases, 3-dimensional objects are much more elongated in one or two direction. For instance, objects like wires can be considered as 1-dimensional objects and thin-walled objects like membranes or organs like stomach can be considered as 2-dimensional objects. In such cases, these can be hardly modeled using polyhedral elements without discretizing them finely to obtain an accurate representation. As a consequence, the element count is largely increased, which severely degrades the performances.

However, 3 degrees of freedom per node is not sufficient to capture the behavior of these structures. To address this issue, some authors have proposed to use 6-DoF nodes to model deformable bodies. With such an approach, nodes are not only able to move linearly, but can also rotate themselves. The main idea behind this concept is to bring the ability to twist and bend to objects that have virtually no thickness. This motivates the addition of a rotational part to the node's DoF.

However, to accurately reproduce the behavior of this type of objects, physical parameters are needed, meaning that their animation must rely on physically-based techniques. They can be represented by a special class of finite element models known as *beam* and *shell* elements, which are based upon 6-DoF nodes. In addition, the *Cosserat theory* [CC09] provides an alternative representation of an elastic continuum that sees the medium as made up of oriented particles.

Thus it constitutes a mechanical background for objects discretized by a set of 6-DoF nodes.

In the field of surgical simulation, different kind of entities can be seen as 1-dimensional objects. It is the case, for instance, of surgical threads for suturing or catheters in the case of interventional radiology [WPL$^+$05]. In [Pai02] Pai presents a method for the simulation of thin 1-dimensional objects based on 6-DoF nodes and the Cosserat theory.

A similar approach is developed by Duriez *et al.* in [DCLN06] where a catheter is simulated as a succession of beam elements. In this work, the authors use 6-DoF nodes and a domain decomposition approach inspired by structural analysis [Prz85] for the assembly of the catheter. An analogous methods is described by Lenoir *et al.* in [LCDN06] for guidewire and stent simulation.

Although 1-dimensional object models are generally used to simulate the instruments that interact with the virtual patient (catheter, endoscopes, *etc.*), they have also been used to simulate vascular network by [PDC12] *et al.*

Within the human body, many structures can be abstracted as membranes like the stomach, the uterine, glisson capsule, *etc.* Different approaches have been proposed to simulate them using 6-DoF nodes. A method for real-time simulation of membrane like objects is developed by Choi *et al.* in [CWK07]. It consists in extending the stiffness warping proposed Müller *et al.* ([MDM$^+$00]) to thin shell models. For medical simulation purposes, Comas *et al.* propose an approach based on a corotational formulation of the *shell theory* for the simulation of intra-ocular implant deployment [CCD10].

Still, the use of 6-DoF nodes is not solely dedicated to model the mechanical behavior of curves and surfaces. This approach can also be used for the simulation of volumetric objects. The method developed by Martin *et al.* in [MKB$^+$10] aims at unifying the simulation of 1-, 2- and 3-dimensional objects using a physically-based meshless approach. They start with a 3-dimensional representation of the object, and then lower its dimensionality by linearizing position and displacements in the directions in which it has the smaller extent. Although the fact that the authors do not use 6-DoF nodes, they introduce the concept of *elaston* which a strain measure with the ability to encode twisting and bending. In their work, the authors use a *Generalized Moving Least Squares* (GMLS) allowing them to compute shape functions even when the node distribution is collinear or coplanar.

The simulation of meshless volumetric deformable bodies using 6-DoF nodes is proposed by Gilles *et al.* in [GBFP11]. Their approach relies on a sparse sampling of the objects using *dual-quaternions*, which are a mathematical objects representing both position and orientation. Increasingly used in skeletal animation [KCOZ06, KCŽO07], dual quaternions include a part built using *dual* *numbers* and allow the encoding of complex motions such as screw motion. The authors also provide a way to precompute stiffness matrices by using a MSM in order to increase the computational rate. Being designed for sparsely sampled objects, this approach could be used to simulate deformable object without consuming too much computational resources.

This work is then extended by Faure *et al.* in [FGBP11] who propose to study the influence of different shape functions, and more specifically, material aware shape functions.

Another approach form Müller and Chentanez propose to simulate deformable bodies using oriented particles [MC11]. However, this method does not rely, strictly speaking, on 6-DoF particles. Rather, it uses an orientation computed from a generalized shape matching technique. Indeed, in this work, the rotation matrix of a particle is computed from the displaced position of its neighbors. In addition, this work is more related to a MSM approach and thus, does not use material parameters.

2.4 GPU based simulation

Physically-based simulation of deformable bodies is computationally intensive. It is especially true when high resolution models and complex material behaviors are used. Yet, often, computations have to be done as fast as possible to remain compatible with the medical procedure. Depending on the latter the timespan allowed for the computations ranges from minutes, in the case of pre-operative planning, to milliseconds, in the case of per-operative assistance or training simulation.

The advent of new parallel architectures especially the *Graphic Processing Units* (GPU) enables considerable acceleration factors in diverse fields. The CPU clock rate has stalled over the past few years and the current trend goes toward the multiplication of computation units. Although present day CPU can embed up to 16 cores, the acceleration factors obtained are not comparable to those obtained by GPU. GPU performances are, indeed still increasing in both clock rate and computation units count. Still, some problems are difficult to parallelize. In addition dedicated hardware programming is subject to technical constraints that have to be met to maintain satisfying performances. Indeed, the way GPUs work is different from CPUs. For instance, in contrast with CPUs, all the threads running on a GPU execute strictly the same instructions. Hence, the difficulty comes from the fact that one must find a way to solve a given problem while assigning the same task to all the threads.

GPU programing (GPGPU) has been increasingly used for physically-based simulation, and has quickly generated a lot of interest from the medical simulation community [SM06]. Linear elasticity models are interesting because a non-negligible part of the computation can be done off-line. Conversely, in the case of nonlinear elasticity models, precomputations are not possible, except by using reduced models as done by Barbič *et al.* [BJ05]. In this context, Comas *et al.* propose, in [CTA+08], a GPU implementation of nonlinear FEM simulation that takes into account material anisotropy and viscoelasticity. To the same end, Joldes *et al.* propose a TLED simulation of nonlinear finite element on GPU dedicated to brain surgery. In [CA09], Courtecuisse and Allard worked out a parallel implementation of a Gauss-Seidel solver for dense system that can be used for solving constraints. Later, the authors propose a GPU based FEM solver relying on implicit temporal integration [ACF11]. Similarly, Dick *et al.* in [DGW11] develop a GPU version of a multigrid FEM solver for hexahedral meshes.

GPU computing can be particularly interesting in certain particle-based simulations where forces can be computed independently from other particles. SPH for instance is one of those cases, and many work has been carried out for their implementation on GPU, like Gowsami *et al.* in [GSSP10]. In [SE10] Shahingohar *et al.* propose a simulation of needle insertion relying on a GPU accelerated meshless method which is itself based on a generalized shape matching.

2.5 Medical simulation

The goal of medical simulation is to develop virtual clones of the human body for training or planning purposes. To this end, deformable models are use to simulate the biomechanic behavior of organs. However, in order to be truly efficient, simulation's models must faithfully reproduce the human anatomy. This is made possible thanks to the use of patient data issued from medical imaging.

In this section, we will treat about the use of deformable model and their applications to organs simulation. The role of patient specific data in planning procedures, as well as a source for anatomical models is also discussed. Finally, we will introduce a category of soft tissues, known as *connective tissues.* These are often neglected in simulation even though they play an important role in organs behavior.

Organ specific simulation

Many works in medical simulation are dedicated to one specific organ which is simulated alone. The main reason for this is to provide a framework for a particular surgical intervention. The simulation of the liver, for instance, has received a lot of attention. In [MHC+10a], Marchesseau *et al.* propose a new approach, based on FEM, and its application to liver surgery. Similarly, Peterlik *et al.* in [PDC12] propose a real-time simulation that integrates the effect of the vascular network on the behavior of the liver. The brain has also been well studied in medical simulation. In [HWM07], Horton *et al.* present a meshless method for the simulation of brain deformation. Wittek *et al.* propose in [WKWM05], a nonlinear FEM simulation of the brain shift effect. Similarly, in [BDDC11], Bilger *et al.* propose a FEM simulation for electrodes insertion planning.

However, methods offering the ability to simultaneously simulate multiple organs are rather rare. In [CAK$^+$14], Courtecuisse *et al.* provide and example of such a method that specifically aims at modeling the interaction between the different organs of the abdominal cavity.

Use of patient-specific data

One of the key aspects of the medical simulation is its ability to use biomedical data. Indeed, the evolution of medical technologies, especially in imaging, combined with image processing techniques and numerical simulation opens avenues for surgical planning. Data coming from medical imaging like CT scans, MRI *etc.* are processed and segmented to produce 3-dimensional images usable to generate meshes for simulation. In [DDCK09], for instance, Dequidt *et al.* use patient-specific vascular network to simulate the coil embolization of an aneurysm. In [HNR$^+$10], Hostettler *et al.* propose a method to predict the position of abdominal viscera from the analysis of the patient's breathing motion. A FEM approach is proposed by Luboz *et al.* in [LCSP05] for the prediction of post-operative results in the context of maxillofacial facial surgery. In [NRBD08], Nguyen *et al.* use a patient-specific liver vascular network for per-operative surgical assistance. Mansi *et al.* [MVM$^+$11], in the context of mitral valve disease treatment, propose to integrate patient-specific anatomy and boundary conditions in a FEM simulation to predict the efficiency of the surgery.

Connective Tissues

Connective tissues are soft tissues that link anatomical structures together, like ligaments, tendons, fat *etc.* They can be seen as an heterogeneous volumes of soft tissues that occupies the space between certain organs. Therefore, these tissues do not have uniform geometrical properties. Indeed, they may expose different topologies in the same anatomical structure. The can have volume, very thin, and sometimes linear parts at the same time. Modeling these structures using volume elements would be inefficient. Indeed, it would require the use of very small element to accurately reproduce the thinnest parts of the tissues. Thus, by drastically increasing the element count, it would consume the computational resource needed for the simulation of the organs. Furthermore, modeling such structures using a mix of 1-, 2- or 3-dimensional elements (beams, shells and polyhedra) is also difficult as the nature of the interface between these elements is not known. In addition, they can hardly be seen on medical images. For these reasons, they are rarely modeled in medical simulations. Still, they are suspected to have a considerable influence on surrounding organs. Thus, neglecting them implies possible large errors on the biomechanical models used in simulations.

In [Sis12] Sismanidis *et al.* briefly treat of the connective tissues but only from a collision detection point of view. The work developed by Plantefève *et al.* in [PPC$^+$14] propose to create an atlas containing connective tissues around the liver from segmented medical images. Thank to registration techniques, the content of the atlas is then used to impose boundary conditions of a virtual liver model. However, this work emphasizes on the correct location of connective tissues rather than on their modeling. Still, we will show in a subsequent section that the method used to simulate their influence has a non negligible impact on simulation's results.

2.6 Contribution

The work done during this Ph.D. aims at the modeling of the boundary conditions on the organs in the context of medical simulation. More precisely, we aim at modeling the role of connective tissues. This work encompasses two different aspects: the simulation of connective tissues by themselves as well as their behavior, but also the interplay they provide between anatomical structures. In this thesis we aim for an accurate simulation of the complex mechanical coupling that occurs between organs owed to connective tissues. An additional objective of this work is to propose a model computationally light enough to enable the simulation of other organs in real-time.

Connective tissues

The goal of this work is to provide a method that enables the simulation of deformable objects exhibiting mixed topology, with, for instance, both quasi-surfacic and volumetric parts. We will show that, for this purpose, we have developed a dedicated physically-based mechanical model based upon 6-DoF nodes. This model makes coupling between anatomical structures easier.

The proposed approach is based on nonlinear FEM and can be used in mesh-based or meshfree schemes, depending on the topology of the soft tissues to simulate. Since it is founded on continuum mechanics, this model is not restricted to the use of a particular constitutive law. To make the discretization of the connective and the mechanical coupling even easier, this method is usable in both mesh-based and meshfree schemes. In addition, we will show in a following chapter that it allows to use fewer degrees of freedom than classical for a 3-DoF nodes FEM simulation while resulting in higher accuracy. Thus, it enables for a sparse sampling of connective tissues. To provide the user with an interactive experience as required in training simulations, a parallel implementation running on GPU has been developed.

Finally, the formulation of the force on the 6-DoF nodes has been derived to support implicit time integration and thus stable simulation. This once again aims at providing a method suitable for user interaction.

FEM based guidance of soft robots

Along with the physically-based 6-DoF mechanical model, an inverse simulation method also dedicated to 6-DoF nodes has been developed. The objective here is to determine the force to be applied on a given set of nodes to deform an object so that it takes a particular shape. The particularity of this method comes from the transfer of the behavior of a complex FE model onto a reduced set of 6-DoF nodes.

Such a kind of method can be applied to robotics, especially for soft robot guidance. Thus, we have proposed an inverse simulation approach to control a deformable robot, based on a reduced model that takes the robot's geometry into account.

Our approach relies on structural analysis techniques allowing to decompose a soft robot arm in several parts. The dynamics of each of these parts is computed using continuum mechanics and a method similar to condensation techniques, in order to provide a realistic behavior even in the case of complex shapes.

The model reduction techniques enables to considerably lower the number DoF of the final system, and the use of optimized resolution methods allows us to solve the inverse problem in real-time while keeping a good precision.

List of publications

- Julien Bosman, Christian Duriez, and Stéphane Cotin. Connective Tissues Simulation on GPU. In *10th Workshop on Virtual Reality Interaction and Physical Simulation*, pages 41–50, Lille, 2013

- Julien Bosman and Christian Duriez. Modèle pour la simulation de tissus connectifs. *Revue Electronique Francophone d'Informatique Graphique,* *REFIG*, 8:1–13, 2014

- Julien Bosman, Nazim Haouchine, Igor Peterlik, and Christian Duriez. The Role of Ligaments: Patient-Specific or Scenario-Specific?, 2014
- Julien Bosman, Thor Morales Bieze, Othman Lakhal, Mario Sanz, Rochdi Merzouki, and Christian Duriez. Domain decomposition approach for FEM quasistatic modeling and control of Continuum Robots with Rigid Vertebras. In *International Conference on Robotics and Automation (ICRA)*, 2015

Chapter 3

6-DOF model for continuum mechanics

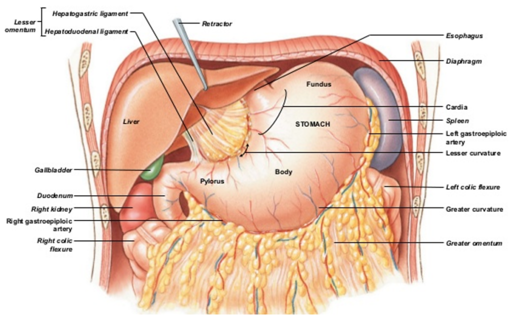
3.1 Introduction

In the field of medical simulation, a lot of work and attention has been dedicated to constitutive laws and to the estimation of their parameters to faithfully reproduce the behavior of the organ [DPP12], [PLC$^+$12], [Pay12]. Yet, modeling the mechanical behavior of the organs is only half of the problem that needs to be solved to obtain accurate simulations. Indeed, complex constitutive laws are of limited use without a precise modeling of the boundary conditions that occur on the organs.

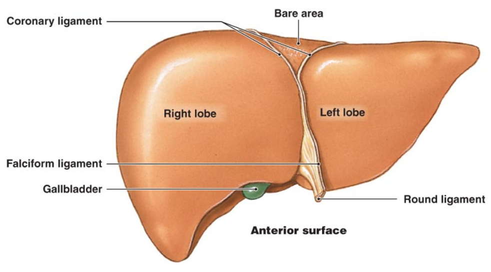
The liver for instance, because of its position inside the abdominal cavity, is subject to two different kinds of interactions influencing its boundary conditions. The first type of interaction is due to the direct influence of the surrounding organs, which can be seen as a unilateral constraint. In this case, interactions can be modeled as frictional contact as done by Courtecuisse *et al.* in [CAK$^+$14]. Though, such interactions do not reflect reality since organs are rarely in direct contact.

Another kind of interaction is provided by surrounding anatomical structures by means of connective tissues. Here, an indirect influence comes from the mechanical coupling with other organs. For instance, the liver is linked to the stomach by the *hepatogastric* ligament as depicted on figure 3.1(a). Moreover, it is also influenced by the respiratory motion as it is linked to the diaphragm by the *falciform* ligament as shown on figure 3.1(b).

Yet, connective tissues are often ignored in medical simulations or are described using very simplified model such as bilateral constraints or springs. Some reasons that may explain this situation is that they have complex shape,



(a)



(b)

Figure 3.1: Hepatogastric and falciform ligaments

and, in addition, they can be hard to locate by medical imaging. Still, connective tissues are responsible for a non negligible part of organ's boundary condition, but there are no dedicated technique to model their behavior.

In this chapter, we will first study the influence of the boundary conditions and the impact of the model chosen to simulate them. This study takes the liver as an example, but the approach could be extended to other soft organs surrounded by other organs, ligaments and connective tissues. We will show that, depending on the type of effort that liver is subject to, the modeling of the ligaments plays a role at least as important as a correct estimation of the organ's material parameters.

Then, in a second part, we will introduce the continuum mechanics principles and their application to deformable bodies dynamics.

Finally, in the third part, we will derive the continuum mechanics formulation to 6-DoF nodes mechanical models to provide a dedicated method for the modeling of connective tissues.

3.2 Influence of the boundary conditions

In this part, we introduce a study treating about the importance of correctly modeling of connective tissues and more specifically the ligaments. Our objective is to highlight the need for a mechanical model dedicated to these tissues. The approach followed here is to quantify the influence of ligament's biomechanical model on simulation's outcome. We will show that the chosen model for ligaments has sensible impact on liver's behavior. We will also demonstrate that depending on the type of forces/tractions the organ is subject to, ligaments model may have more influence than liver's material parameters. Multiple methods of simulation these soft tissues' behavior are used and compared. The results of this study are demonstrated numerically with scenarios typically performed in the liver surgery described below.

Context

This study is performed in the context of laparoscopic liver surgery. Laparoscopic surgery is a type of minimally invasive surgery that consists in doing small incisions to introduce surgical instruments. It is beneficial for patients as it reduces scarring and shortens recovery duration. The laparoscopic scene represents the right and left lobes separated on the anterior side by the falciform ligament (Fig. 3.2). The 3D model of the lobes is obtained from pre-operative CT-scans and includes the vascular network [PDC12]. The falciform ligament is more complicated to extract from the pre-operative images. However, recent work [PPC+14] shows promising results in the transfer of the ligament positions using atlas-based techniques. Thus, we assume that the ligament position can be determined. The underlying fat supporting the liver is also modeled to simulate the surrounding connective tissues.

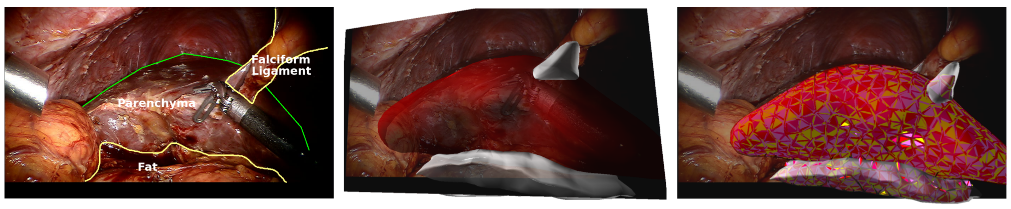


Figure 3.2: Laparoscopic view of the liver with in (a) the lobes, the falciform ligament and the fat, (b) the meshes computed from CT-scans and (c) the underlying finite element model.

Method

The study presented here is purely numeric: its aim is to observe the influence of boundary conditions on the liver's behavior. Hence, it involves several simulations in which we combine different methods to model the ligaments. The results of these simulations are then compared.

For this sensitivity study, three experiments are performed: the first one is the deformation of the liver by a surgical instrument whose position is supposed to be known. The second one is the application of a force similar to the one exerted by the instrument. The last one is the modeling of the deformation of the liver during the pneumoperitoneum, a procedure consisting in the insufflating gas into the abdominal cavity to enlarge the surgeon's workspace. Both of these three experiments result in a considerable deformation of the liver.

It should be emphasized that the important difference between these experiments is the type of load that creates the deformation. In the first case, a displacement is imposed on a part of the mesh (in the region of the liver that is being grasped). In the second and third cases a force and a pressure (tractions) are imposed on the surface of the model. In both cases, the goal is to obtain the displacement field of the liver model nodes.

In each test, the liver parenchyma is simulated with the corotational formulation of the finite element method using the linear constitutive law.

With regard to the ligaments, the comparative simulations are based on two hypotheses. In the first hypothesis, the ligament is considered as a full anatomical structure and simulated using linear elasticity. In the second, the ligaments are considered highly stiff. Hence the points of the liver that are supposed to be linked to ligaments are fixed in space.

In all the simulations performed, the volume meshes are made up of linear tetrahedral elements. Time integration is done using the backward Euler method and the linearized equation system is solved using a preconditioned conjugate gradients. In the reference simulation, both liver and ligaments are simulated using the finite element corotational formulation.

In each of the three experiments of this sensitivity analysis, three tests are conducted and their results are compared to the reference simulation. The error evaluation on the displacement is done after the equilibrium is achieved using *Root-Mean-Square Error* (RMSE).

The tests are organized as follows: in the first test, the role of the ligaments is compared when modeled as fixed points and when simulated using corotational formulation with a Young's modulus of $E_{lig} = 150\,\text{kPa}$. Then, in the second and third tests, ligaments are simulated with FEM, but the Young's modulus of the parenchyma $E_{ref} = 27\,\text{kPa}$ is under- and over-estimated by a factor two with respect to the reference simulation.

The grasping is simulated by prescribing a displacement on the lobe of the liver. The motion of the displaced nodes is extracted from a video of a laparoscopic surgery. The maximum displacement applied by the instrument is 75 mm when the equilibrium is reached.

To ensure that both prescribed displacements/efforts tests result in a similar motion, the prescribed force value is computed from the prescribed displacements scenario. To do so, average elastic forces of the displaced nodes of the liver are measured after the equilibrium. From the measurement, a traction force of 15 N has to be applied on the nodes to be equivalent to the prescribed displacements. Displacements and efforts are prescribed on the same nodes and the direction of the force vector is chosen to be the same as displacement vector one.

In the simulations, the pneumoperitoneum is modeled by applying a pressure of 12 mmHg (1600 Pa) on the liver surface, which is the average pressure used by surgeons during abdominal laparoscopic surgeries [BHN+12].

Results

The results presented in Tab. 3.1 show two specific scenarios: simulations where the deformation is induced by prescribing a displacement and those where applied forces are prescribed, each of them having their own requirements.

Considering prescribed displacements, results show that a large difference in the elasticity parameter of the liver has a relatively small impact on the error when compared to the influence of the modeling of the ligaments. The influence of the model used for ligaments is more than twice as important as the elasticity parameter used for the parenchyma. The results show that in such a scenario, the use of a physically-based deformable model is crucial.

As for the prescribed forces, the alteration of Young's modulus in liver has much more influence than in the case of prescribed displacements. The results emphasize the importance of using patient-specific data in such scenario. However, in this case, they also show that the influence of the model used for ligaments is comparable to the impact of the elasticity parameters.

Going further, the results of show that the difference between errors due to the ligament model and those due to the elasticity parameter is important when simulating the same action. In this case, force instance, a liver lobe is pulled, using two different load methods (prescribed displacement/prescribed forces).

Experiments	***Grasping***	***Traction***	***Pressure***
Fixed nodes	2.75	7.19	3.53
$E = 0.5 \times E_{ref}$	1.12	7.76	8.40
$E = 2 \times E_{ref}$	1.77	4.85	3.93

Table 3.1: Evaluation of the Root-Mean-Square Error (RMSE) (in mm) for prescribed displacement (grasping) and prescribed forces (traction and pressure), compared to the reference simulation.

3.3 Continuum mechanics

In rigid bodies, all the material points move in a similar manner. It means that their motion can be reduced to that of a single frame, and can be computed using rigid body dynamics. However, a deformable body is made up of an infinity of material points that can move much more independently. In this case, the motion of the object must rely on more evolved concepts.

Continuum mechanics is a branch of classical mechanics that encompasses multiple fields such as fluid dynamics and solid mechanics. It is based on the assumption that objects are studied at a macroscopic scale, allowing us to see them as perfectly continuous volumes of matter.

Solid mechanics includes, among others, the elasticity model which describes the motion of deformable objects. In the following, we will consider elasticity which, unlike plasticity, refers to objects that recover their initial shape when the cause creating the deformation stops. However, it must be noted that in reality, perfect elasticity only exist for a limited number of material, subject to small deformations.

In this section, we introduce the basic concepts of continuum mechanics and more precisely the elasticity model that are used in the next section to develop a physically-based model for the simulation of connective tissues.. We will show how the internal forces that allow a deformable object to recover its rest shape can be computed thanks to this model.

Lagrangian representation

The principle of the Lagrangian description is to follow the evolution of the position and properties of a material point over the time. The position of the material point is always given with respect to its rest position $\bar{\mathbf{x}}$. Thus, the state of a material point at time t is given by :

$$\mathbf{x} = \mathcal{L}(\bar{\mathbf{x}}, t)$$

and the rest position is defined as :

$$\bar{\mathbf{x}} = \mathcal{L}(\bar{\mathbf{x}}, 0)$$

Such formalism allows us to follow the motion of a material point experiencing a displacement. For this reason the Lagrangian description provides an interesting tool to describe the motion of a deformable body that will be used in the following sections.

Updated/Total Lagrangian dynamics

The Lagrangian representation offers two ways to measure the deformation of an object. The first one is called the *updated Lagrangian formulation.* It considers the current deformations with respect to the state of the object at the previous configuration:

$$\mathbf{x} = \mathcal{L}(\mathbf{x}, t_n), \quad \bar{\mathbf{x}} = \mathcal{L}(\mathbf{x}, t_{n-1})$$

where t_n and t_{n-1} are respectively the current and the previous time-step.

Alternatively, the *total Lagrangian formulation* considers the current displacements with respect to the object's rest – or initial – configuration;

$$\mathbf{x} = \mathcal{L}(\bar{\mathbf{x}}, t_n), \quad \bar{\mathbf{x}} = \mathcal{L}(\bar{\mathbf{x}}, 0)$$

The latter is more suitable for the simulation of perfectly elastic objects while the updated Lagrangian formulation suits the simulation of plastic deformations better. Thus, all the work presented in this thesis relies on a total Lagrangian formulation. The advantages of this choice will be described in a later section.

Measuring deformations

When an object undergoes a deformation, it stores energy into its volume. This potential energy, from which the internal force arise, is dependent on the intensity of the deformations. Thus, to compute the internal forces, one must find a way to quantify these deformations.

Using the total Lagrangian description, the displacement of a material point can be formulated as:

$$\mathbf{u}_i = \mathcal{L}(\bar{\mathbf{x}}_i, t) - \mathcal{L}(\bar{\mathbf{x}}_i, 0) = \mathbf{x}_i - \bar{\mathbf{x}}_i \tag{3.1}$$

Now, if we consider the displacements of all the material points of the object, it results in a vector field, called *displacement field.* It fully describe the motion of the object between the two configurations.

However, displacements alone are not enough to characterize the deformations undergone by an object. For instance, let's consider the case of a rigid object in motion (translation and/or rotation). Between two consecutive times, the position of its material points has changed, resulting in a non-zero displacement field. Yet, this object has not undergone any deformation.

Thus, in order to quantify the deformations, one must use a more reliable measurement method.

Deformation gradient

The deformation gradient is a second order tensor that corresponds to the spatial derivative of the current configuration with respect to the reference configuration:

$$\mathbf{F} = \frac{\partial \mathbf{x}_i}{\partial \bar{\mathbf{x}}_j} = \begin{bmatrix} \frac{\partial \mathbf{x}_0}{\partial \bar{\mathbf{x}}_0} & \frac{\partial \mathbf{x}_0}{\partial \bar{\mathbf{x}}_1} & \frac{\partial \mathbf{x}_0}{\partial \bar{\mathbf{x}}_2} \\ \frac{\partial \mathbf{x}_1}{\partial \bar{\mathbf{x}}_0} & \frac{\partial \mathbf{x}_1}{\partial \bar{\mathbf{x}}_1} & \frac{\partial \mathbf{x}_1}{\partial \bar{\mathbf{x}}_2} \\ \frac{\partial \mathbf{x}_2}{\partial \bar{\mathbf{x}}_0} & \frac{\partial \mathbf{x}_2}{\partial \bar{\mathbf{x}}_1} & \frac{\partial \mathbf{x}_2}{\partial \bar{\mathbf{x}}_2} \end{bmatrix} \tag{3.2}$$

Another way to express the deformation gradient is:

$$\mathbf{F} = \frac{\partial \mathbf{u}_i}{\partial \bar{\mathbf{x}}} + \mathbf{I} = \mathbf{G} + \mathbf{I} \tag{3.3}$$

where $\mathbf{G}$ is the *displacement gradient.*

However, deformed and rest configuration may first seem unrelated, hindering the computation of the deformation gradient. In fact, those two configuration are tightly linked by the function that is the Lagrangian description. Hence, the deformation gradient can be computed as follows:

$$\mathbf{F} = \frac{\partial \mathcal{L}(\bar{\mathbf{x}}, t)}{\partial \bar{\mathbf{x}}} \tag{3.4}$$

The deformation gradient gives a description of local deformation in the vicinity of a material point. It can be seen as a “mapping” between the rest and deformed configurations. For example, considering a vector $d\bar{\mathbf{x}}$ between two infinitesimally close material points in the rest configuration, this vector in the deformed configuration becomes:

$$d\mathbf{x} = \mathbf{F}\, d\bar{\mathbf{x}} \tag{3.5}$$

However, if the object does not sustain any deformation, the deformation gradient is not null, but the identity. Furthermore, if the whole deformable object is rotated, the deformation gradient only contains the rotation matrix that encodes this transformation. Generally, the deformation gradient is not used alone, but serves as a basis for more evolved deformation measurements that are detailed in the following.

Strain tensor

Alternatively, one may want to quantify the deformation by measuring the squared distance between two points relatively to the deformed configuration.

Applying this to Eq (3.5) one obtains:

$$\begin{aligned} \|\mathbf{v}\|^2 &= (\mathbf{F}\,\bar{\mathbf{v}}) \cdot (\mathbf{F}\,\bar{\mathbf{v}}) \\ &= (\mathbf{F}\,\bar{\mathbf{v}})^\top (\mathbf{F}\,\bar{\mathbf{v}}) \\ &= \bar{\mathbf{v}}^\top\,\mathbf{F}^\top\,\mathbf{F}\,\bar{\mathbf{v}} \\ &= \bar{\mathbf{v}}^\top\,\mathbf{C}\,\bar{\mathbf{v}} \end{aligned} \tag{3.6}$$

where $\mathbf{C} = \mathbf{F}^\top\,\mathbf{F}$ is the *right Cauchy-Green* strain tensor. One may notice that while this strain tensor is invariant under rigid body rotation and equals the identity if no deformation occurs.

Another way to measure the deformation is to express the difference of the squared distance between two points in both deformed and undeformed configuration. Computing this distance by reusing Eq. (3.6) leaves:

$$\begin{aligned} \|\mathbf{v}\|^2 - \|\bar{\mathbf{v}}\|^2 &= \bar{\mathbf{v}}^\top \cdot (\mathbf{C}\,\bar{\mathbf{v}}) - \bar{\mathbf{v}} \cdot \bar{\mathbf{v}} \\ &= \bar{\mathbf{v}} \cdot (\mathbf{C} - \mathbf{I})\bar{\mathbf{v}} \\ &= 2\,\bar{\mathbf{v}} \cdot \left(\frac{1}{2}\,(\mathbf{C} - \mathbf{I})\,\bar{\mathbf{v}}\right) \\ &= 2\,\bar{\mathbf{v}}^\top\,\mathbf{E}\,\bar{\mathbf{v}} \end{aligned} \tag{3.7}$$

where $\mathbf{E} = \frac{1}{2}\left(\mathbf{F}^\top\mathbf{F} - \mathbf{I}\right)$ is the *Green-Lagrange* strain tensor. This tensor is also rotationally invariant, and zero if the object has not undergone any deformation.

To measure deformation, other strain tensors exist, though the two introduced above are the most commonly encountered in simulations. The choice of strain tensor type depends on the assumptions made when modeling the problem. The main criterion is the magnitude of the displacements that the object is supposed to experience.

If the object is prone to large displacements, the Green-Lagrange strain tensor is better suited, because it can handle geometrical nonlinearities such as local rotations. Indeed, if the Green-Lagrange strain tensor is rewritten using Eq. (3.3) it yields:

$$\begin{aligned} \mathbf{E} &= \frac{1}{2}\left[(\mathbf{G} + \mathbf{I})^\top\,(\mathbf{G} + \mathbf{I}) - \mathbf{I}\right] \\ &= \frac{1}{2}\left(\mathbf{G}^\top\mathbf{G}\right) + \frac{1}{2}\left(\mathbf{G}^\top + \mathbf{G}\right) \end{aligned} \tag{3.8}$$

where the left term encodes the nonlinear deformation while the right term describes the linear component of the deformation.

If assumptions are made on a small strain (*i.e.* $\|\mathbf{u}\| \ll 1$ and $\|\mathbf{G}\| \ll 1$), the left term is negligible and it is possible to keep only the right term:

$$\varepsilon = \frac{1}{2}\left(\mathbf{G}^\top + \mathbf{G}\right) \tag{3.9}$$

which is also known as the *linearized* Green-Lagrange strain tensor or *Cauchy's infinitesimal* strain tensor. Its main advantage is that it enables faster computations, but it loses its accuracy as the displacements become larger.

Stress

In the case of a deformable object, the restoring forces that try to counteract the deformation come from the stress induced by the strain. The strain itself is produced by displacements, as seen in the previous section. Stress is equivalent to a pressure, *i.e.* a force occurring on a small surface of the object's cross section. This force can be broken down into three components, one normal to the cross section plane, and the two others parallel to the plane. When the force is only directed along the normal, it is called *normal stress*, and its sign defines whether the effort is an elongation (positive), or a compression (negative).

If the whole cross section of the object is taken into account, it only results in an average measure of the force acting on the surface. Hence, to precisely evaluate the stress in the object's continuum, one has to consider an infinitesimal surface inside the object's cross section. By doing so, the stress measure over a surface tends toward a measure of stress around a material point. However, because the surface has been shrunk to a point, there is now an infinity of planes that goes through it, making the stress hard to define. Yet, it can be shown that an equivalent measure can be done by using three mutually orthogonal planes passing through the point.

The force is measured in each of these planes and, for each plane. Then the force is separated in three orthogonal components, one normal and two parallel to the plane, as shown on Fig. (3.3). Thus, the stress at a given point is measured using 9 components and can be expressed as a second rank tensor. The 3 normal stress components are on the diagonal while the off-diagonal entries are related to shear stress components. As a result, a stress tensor is written as:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{00} & \sigma_{01} & \sigma_{02} \\ \sigma_{10} & \sigma_{11} & \sigma_{12} \\ \sigma_{20} & \sigma_{21} & \sigma_{22} \end{bmatrix} \tag{3.10}$$

where the first index designates the plane in which the stress is expressed and the second index is related to the stress component in the plane.

When modeling elastic bodies, different types of stress tensor are available, depending on the application (small/large displacements, updated/total Lagrangian *etc.*) and on the reference configuration (deformed/undeformed) used to express stress.

In the context of total Lagrangian formulation, the *2nd Piola-Kirchhoff* stress tensor must to be used. This tensor, also called *PK2* ha no physical interpreta-

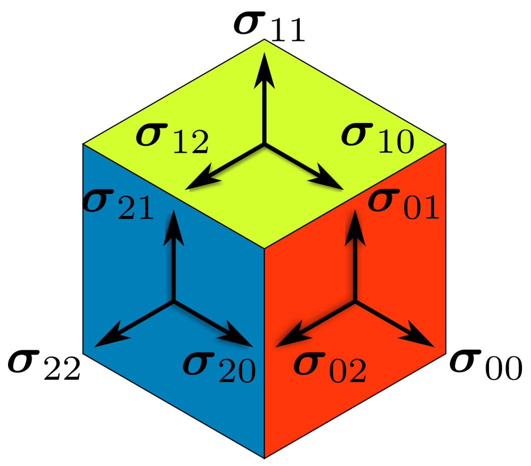


Figure 3.3: Stress planes

tion. However, it has the advantage of being totally defined by the rest configuration of the deformable object. Indeed, it relates the force in the undeformed configuration with respect to the undeformed surfaces, which makes it easier to compute. For this reason, it is often used in total Lagrangian formulations. Additionally, as it is described in material coordinates, unlike the Cauchy stress, it is rotationally independent and thus suitable for simulations implying large deformations.

Constitutive laws

Strain tensors measure the deformation of an object around a material point. On the other hand stress tensor describes how internal forces are acting to put the object back in its rest configuration. In fact, strain and stress are two tightly related notions. Indeed, given the strain (local deformations) at one point of the domain, it is impossible to know the associated internal forces without knowing the properties and the behavior of the material the object is made of.

Thus, to establish the relationship between stress and strain, one must rely on a *constitutive law* that describes how a material reacts to the deformation it undergoes. Such a relationship is determined by performing mechanical tests on a material sample. For instance, tests may consist in the traction and compression of a sample. By imposing a displacement, whose value is known, and measuring the force generated (or vice versa), it is possible to characterize the behavior of a material.

As a result, a constitutive law may be defined as follows:

$$\sigma = f(\varepsilon) \tag{3.11}$$

and conversely:

$$\varepsilon = f^{-1}(\sigma) \tag{3.12}$$

Linear elasticity

Linear elasticity, also known as *Hooke's law*, is the simplest constitutive law. It linearly relates strain and stress. In other words, there is a linear relationship between the displacement and the resulting internal force, which results in a linear strain-stress curve. Hooke's Law is given by:

$$\sigma = 2\mu\varepsilon + tr(\varepsilon)\lambda\mathbf{I} \tag{3.13}$$

where μ and λ are the Lamé coefficients which can be computed from *Poisson's ratio* ν and *Young's modulus* E:

$$\mu = \frac{E}{2\,(1+\nu)}$$

and

$$\lambda = \frac{E\nu}{(1+\nu)\,(1-2\nu)}$$

Poisson's ratio is related to the compressibility and ranges from 0 when the material is perfectly compressible (*i.e.* does not expand when compressed) to 0.5 when perfectly incompressible (*i.e.* no volume change). However, in the latter case, it results in λ going to infinity, leading to infinite stress which cannot be handled by simulations. As for Young's modulus, it describes the rigidity of the material. In other words, the higher E is, the stiffer the object will be.

The main advantage of the linear elasticity model is its simplicity. Because a major part of it can be precomputed, faster computations are possible. To put it another way, the rigidity of the material does not change with the deformation. Yet, its simplicity also hinders an accurate simulation of biological soft tissues which requires more evolved constitutive laws.

Hyperelastic materials

Hyperelastic constitutive laws materials for which the stress derives from the strain-energy density function:

$$\mathbf{S} = \frac{\partial W(\mathbf{E})}{\partial \mathbf{E}} \tag{3.14}$$

where $W(\cdot)$ is the strain-energy density function and $\mathbf{E}$ is the strain tensor.

Physically speaking, they may result in materials that becomes stiffer (hardening), or softer (softening), the more they are deformed. Nonetheless, material nonlinearity (that comes from the properties of the material) should not be confused with geometrical nonlinearity, that comes from the type of deformation.

Among hyperelastic material, the *Saint Venant-Kirchhoff* material is an extension of the Hooke's Law to large deformations that use the full Green-Lagrange strain tensor instead of the linearized one.

Thanks to hyperelastic materials more complex materials such as polymer and soft tissues can be described, but at the cost of heavier computations.

3.4 Solving elasticity problems

Governing equations for elasticity

Strain measure, constitutive laws and stress are mathematical tools that allow to formulate the governing equation of elasticity problems. The *Navier* equation is an equivalent of Newton's second law of motion for deformable bodies. It relates the stress that arises in the material to the restoring forces that help the object to maintain its rest shape:

$$\nabla \cdot \boldsymbol{\sigma} + f_{\text{ext}} = \rho \ddot{\mathbf{u}} \tag{3.15}$$

where ρ is the material density, $\ddot{\mathbf{u}} = \frac{\partial^2 \mathbf{u}}{\partial t^2}$ and $\nabla \cdot$ is the divergence operator.

This formulation of the problem is called *strong form* and states the stress and internal force relationship for every material point of the domain. However, working directly with the strong form of the problem may be inefficient. Indeed, strong forms often involve higher order derivatives, which may be unavailable of difficult to estimate. In addition, in Eq. (3.15), boundary conditions are not explicitly specified, making them difficult to include in the solving process.

Method of weighted residuals

In order to obtain the internal forces acting on a deformable body, one must solve a *Partial Differential Equation* (PDE) of the following form:

$$\mathcal{D} x = f \tag{3.16}$$

where $\mathcal{D}$ is a differential operator and x is a property of the object (here, the displacements). Generally, such properties are known only at some locations in the object's domain, namely the nodes of the object. These are the discrete value of the observed scalar of vector field. Thus, to define this property at each material point of the domain, the known values can be interpolated and displacements can now be considered as a function:

$$\tilde{x} = \sum_i \phi_i x_i = \boldsymbol{\Phi}^\top \mathbf{x} \tag{3.17}$$

where $\mathbf{\Phi}$ is a *trial function*, commonly called *shape function*, that will be discussed in a further section and x_i are the known values.

However, the shape function cannot always reproduce perfectly the considered field. Thus, Eq. (3.16) is no longer satisfied and a residual error must be added. Substituting Eq. (3.17) into Eq. (3.16) leaves:

$$\mathcal{D}\tilde{x} - f = \epsilon \tag{3.18}$$

Moreover, as mentioned previously, directly solving the strong form of the problem may bring difficulties. Hence, to make the solving process easier, one can rely on the *weighted residuals*.

This process is carried out by multiplying the equation by an arbitrary *test function* u and integrating it over the object's domain. The test function must be chosen so that it minimizes the residual:

$$\int_\Omega u(\mathcal{D}\tilde{x} - f)d\Omega = 0 \tag{3.19}$$

This equation is also known as the *weak formulation* of the problem and is equivalent to the strong form. The weak formulation has several advantages over the strong form. To be more specific, the order of the PDE to solve can be lowered and, as shown bellow, boundary conditions can be explicitly included.

The method of weighted residuals is now applied to elasticity governing equations. To do so, Eq. (3.15) is multiplied by a test function δu, and then integrated over the object's domain:

$$\int_\Omega \delta\mathbf{u}\left[\nabla\cdot\boldsymbol{\sigma} + \mathbf{f}_{\text{ext}} - \rho\ddot{\mathbf{u}}\right]d\Omega \tag{3.20}$$

which, using indicial notation along with Einstein summation convention, can be rewritten:

$$\int_\Omega \delta u_i \frac{\partial\sigma_{ij}}{\bar{x}_j}d\Omega + \int_\Omega \delta u_i f_{\text{ext}}d\Omega - \int_\Omega \delta u_i \rho\ddot{u}_i d\Omega \tag{3.21}$$

However, the leftmost term in Eq. (3.21) still contains the divergence of the stress tensor which requires the second derivative of displacement field. Thus, to lower this requirement, one may apply the *divergence* theorem:

$$\begin{aligned}\int_\Omega \delta u_i \frac{\partial\sigma_{ij}}{\bar{x}_j}d\Omega &= \int_\Omega \frac{\partial(\delta u_i)}{\partial\bar{x}_j}\sigma_{ij}d\Omega + \int_\Omega \delta u_i \frac{\partial\sigma_{ij}}{\partial\bar{x}_j}d\Omega \\ &= \int_\Omega \frac{\partial(\delta u_i)}{\partial\bar{x}_j}\sigma_{ij} + \int_{\partial\Omega} \delta u_j\sigma_{ij}n_j dS\end{aligned} \tag{3.22}$$

Note that the volume integral has been converted to surface integral.

Substituting Eq. (3.22) into Eq. (3.21) yields the decomposition forces acting on the deformable body. This leaves the *weak form* of the problem:

$$\underbrace{\int_\Omega \frac{\partial(\delta u)}{\partial \bar{x}_i}\sigma_{ij} d\Omega}_{\text{elastic forces}} + \underbrace{\int_{\partial\Omega} \delta u_j \sigma_{ij} n_j dS}_{\text{boundary conditions}} + \underbrace{\int_\Omega \delta u_i f_{\text{ext}} d\Omega}_{\text{body forces}} - \underbrace{\int_\Omega \delta u_i \rho \ddot{u}_i d\Omega}_{\text{inertial force}} = 0 \quad (3.23)$$

or expressed in matrix form:

$$\int_\Omega \nabla(\delta\mathbf{u})^\top : \boldsymbol{\sigma} d\Omega + \int_{\partial\Omega} \delta\mathbf{u}^\top \boldsymbol{\sigma}\mathbf{n} dS + \int_\Omega \delta\mathbf{u}^\top \mathbf{f} d\Omega - \int_\Omega \delta\mathbf{u}^\top \rho\ddot{\mathbf{u}} d\Omega \quad (3.24)$$

where the operator : is a tensor product contracted twice.

Virtual work principle

Virtual work is a particular type of weak form in which the arbitrary test function is said to be a *virtual* displacement. Those virtual displacements are not related to the real displacements that occur in the deformable object. To be more precise, virtual work is the work done by an actual force through virtual displacements:

$$\delta W = \int_{x_a}^{x_b} \mathbf{f} \cdot d\mathbf{x} = \mathbf{u}^\top \mathbf{f} \quad (3.25)$$

Virtual work on a deformable body can be decomposed in three terms – internal, external and kinetic virtual work [BLM00](§4.8) – which correspond to each force term found in Eq. (3.23):

$$\delta W^{\text{int}} = \int_\Omega \nabla(\delta\mathbf{u})^\top : \boldsymbol{\sigma} d\Omega \quad (3.26)$$

$$\delta W^{\text{ext}} = \int_{\partial\Omega} \delta\mathbf{u}^\top \boldsymbol{\sigma}\mathbf{n} dS + \int_\Omega \delta\mathbf{u}^\top \mathbf{f}^{\text{ext}} d\Omega \quad (3.27)$$

$$\delta W^{\text{kin}} = \int_\Omega \delta\mathbf{u}^\top \rho\ddot{\mathbf{u}} d\Omega \quad (3.28)$$

It can be noticed that this formulation is similar to the *potential energy principle.* Considering the virtual work done by elastic forces and combining Eq. (3.25) and Eq. (3.26), it follows that:

$$\delta\mathbf{u}^\top \mathbf{f}^{\text{int}} = \int_\Omega \nabla(\delta\mathbf{u})^\top : \boldsymbol{\sigma} d\Omega \quad (3.29)$$

In addition, thanks to the symmetry of the strain tensor, it can be shown that:

$$\delta \mathbf{u}^\top \mathbf{f}^{\text{int}} = \int_\Omega \delta \boldsymbol{\varepsilon} : \boldsymbol{\sigma} d\Omega \tag{3.30}$$

or, using *Voigt notation*:

$$\delta \mathbf{u}^\top \mathbf{f}^{\text{int}} = \int_\Omega \{\delta \boldsymbol{\varepsilon}\}^\top \{\boldsymbol{\sigma}\} \, d\Omega \tag{3.31}$$

where $\delta\boldsymbol{\varepsilon}$ is the *virtual strain* induced by virtual displacements.

3.5 Application to 6-DoF nodes models

In this section, the methods mentioned above are derived to develop a mechanical model based upon 6-DoF nodes. In a first part, we will describe the modeling and the discretization of deformable objects. Then, in a second part, we will discuss the kinematics of the discretization primitives. Finally, the last section will detail how the mechanical model is derived from the kinematics to provide a continuum mechanics based approach for internal forces computation.

Objects modeling

The deformable objects are discretized using 2 types of primitives: a set of frames and a set of samples, as shown on figure 3.4. The frames are the object's 6-DoF nodes, consisting of both a translation part $\mathbf{x}$ and an orientation part $\mathbf{r}$:

$$\mathbf{q}_i = [\mathbf{x}_i \; \mathbf{r}_i]^\top \tag{3.32}$$

In practice, the orientation part is described using a quaternion. Frames play the role of mechanical points, meaning that they are the entities on which displacements are evaluated and elastic forces applied.

On the contrary, samples are used as integration points and only have only 3 degrees of freedom in translation. Their role is to discretize the object's volume between frames by carrying a fraction of the object's total volume. As such, they serve as a support for computations of volume related quantities such as strains and stresses.

Kinematics

Even though samples have 3-DoF, their position is entirely determined by the positions and orientations of their surrounding frames. Indeed, their kinematics is based upon an approach similar to skinning [Blo02]. Moreover, as this methods relies on a total Lagrangian formulation, the position of a sample is

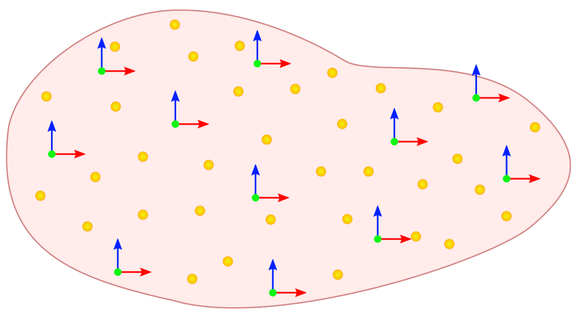


Figure 3.4: Discretization of an object with our method.

expressed with respect to its local position in each neighboring frame coordinates system in the object's rest (undeformed) configuration. Hence, the local position of a sample s in the frame i coordinates system is given by:

$$\bar{\mathbf{l}}_s^i = \bar{\mathbf{R}}_i^\top (\bar{\mathbf{p}}_s - \bar{\mathbf{x}}_i) \tag{3.33}$$

where $\bar{\mathbf{x}}_i$ and $\bar{\mathbf{R}}_i$ are respectively the position and the rotation matrix associated with the orientation of frame i and $\bar{\mathbf{p}}_s$ is the global position of sample s, all given in the rest configuration. Throughout this manuscript, the overbar notation designates quantities expressed in the rest configuration.

Going further, the position of sample s in global coordinates is computed as the combination of its local positions in its neighboring frames:

$$\mathbf{p}_s = \sum_i^n w_i(\bar{\mathbf{p}}_s) \left(\mathbf{x}_i + \mathbf{R}_i \bar{\mathbf{l}}_s^i\right) \tag{3.34}$$

where $\mathbf{x}_i$ and $\mathbf{R}_i$ are the position and the rotation matrix corresponding to the orientation of the frame i in the current (deformed) configuration. $w_i(\bar{\mathbf{p}}_s)$ is the value of the shape function between frame i and sample s. The shape function is used to interpolate the displacements – which are only known at frames – at samples location.

It should be noted that this method is similar to Lagrangian kernel techniques. Indeed, the current position of sample s ($\mathbf{p}_s$) is expressed using shape function values computed in the rest configuration. This means that the support of the shape function deforms accordingly with object's local deformations.

Internal forces computation

The goal here is to provide a realistic simulation of the behavior of a deformable body modeled using 6-DoF nodes. Thus continuum mechanics are used to compute internal forces acting on frames. To do so, virtual work and weighted residual methods presented earlier are used to establish a relationship between displacements on frames (*i.e.* linear and angular deformations) and the associated elastic forces. The latter are computed according to the following scheme:

$$\underbrace{\mathbf{u}_i}_{\text{displacement}} \rightarrow \underbrace{\mathbf{F}_s}_{\substack{\text{deformation}\\ \text{gradient}}} \rightarrow \underbrace{\mathbf{L}_s}_{\substack{\text{strain}\\ \text{tensor}}} \rightarrow \underbrace{\mathbf{S}_s}_{\substack{\text{stress}\\ \text{tensor}}} \rightarrow \underbrace{\mathbf{f}_i}_{\substack{\text{internal}\\ \text{force}}}$$

where subscripts i and s denote respectively quantities computed on frames and samples.

The evaluation of internal forces relies on the principle of virtual work described in subsection 3.4. Therefore, internal virtual work Eq. (3.31) is taken as the starting point of these computations:

$$\delta\mathbf{q}^\top\mathbf{f}^{\text{int}} = \int_\Omega \{\delta\boldsymbol{\varepsilon}\}^\top \{\boldsymbol{\sigma}\}\, d\Omega$$

Then, by reformulating virtual strain as follows:

$$\delta\boldsymbol{\varepsilon} = \frac{\partial\boldsymbol{\varepsilon}}{\partial\mathbf{q}}\delta\mathbf{q} \tag{3.35}$$

and simplifying by $\delta\mathbf{q}$ one obtains:

$$\mathbf{f}^{\text{int}} = \int_\Omega \frac{\partial\boldsymbol{\varepsilon}}{\partial\mathbf{q}}^\top \boldsymbol{\sigma} d\Omega \tag{3.36}$$

Since the method presented here aims at simulating connective tissues, it should handle large deformations and geometrical nonlinearities. To this end, the Green-Lagrange strain tensor $\mathbf{L}$ is used. As this method is based upon total Lagrangian formulation the 2nd Piola-Kirchhoff stress tensor $\mathbf{S}$ needs to be used. As a result, Eq. (3.36) becomes:

$$\mathbf{f}^{\text{int}} = \int_\Omega \frac{\partial\mathbf{L}}{\partial\mathbf{q}}^\top \{\mathbf{S}\}\, d\Omega \tag{3.37}$$

where $\mathbf{L} = \frac{1}{2}\left(\mathbf{F}^\top\mathbf{F} - \mathbf{I}\right)$. Thus, the application of product differentiation rule yields:

$$\frac{\partial\mathbf{L}}{\partial\mathbf{q}} = \frac{1}{2}\left(\frac{\partial\mathbf{F}^\top}{\partial\mathbf{q}}\mathbf{F} + \mathbf{F}^\top\frac{\partial\mathbf{F}}{\partial\mathbf{q}}\right) \tag{3.38}$$

The next step is the computation of the deformation gradient as required by Eq. (3.38). As described in Eq. (3.2), deformation is the spatial derivative of the current position with respect to the rest configuration. Hence, $\mathbf{F} = \nabla_{\bar{\mathbf{x}}}\mathbf{p}_s$.

In parallel, by looking at Eq. (3.33), one can notice that:

$$\frac{\partial \bar{\mathbf{l}}_s^i}{\partial \bar{\mathbf{x}}} = \frac{\partial(\bar{\mathbf{R}}_i^\top \, (\bar{\mathbf{p}}_s - \bar{\mathbf{x}}_i))}{\partial \bar{\mathbf{x}}} = \bar{\mathbf{R}}_i^\top \tag{3.39}$$

Thus, the deformation gradient at a sample s is obtained by differentiating Eq. (3.34) with respect to the undeformed configuration and by using the result of Eq. (3.39):

$$\mathbf{F}^s = \frac{\partial \mathbf{p}_s}{\partial \bar{\mathbf{x}}} = \sum_i^n \left[\left(\mathbf{x}_i + \mathbf{R}_i \, \bar{\mathbf{l}}_s^i \right) \nabla w_{is}^\top + w_{is} \mathbf{R}_i \bar{\mathbf{R}}_i^\top \right] \tag{3.40}$$

where $w_{is} = w_i(\bar{\mathbf{p}}_s)$.

However, the Green-Lagrange strain tensor derivative of Eq. (3.38) also involves the derivative of the deformation gradient with respect to the frames' degrees of freedom $\mathbf{q}$. By doing so, one can obtain the variation of the deformation gradient:

$$\delta \mathbf{F}^s = \frac{\partial \mathbf{F}^s}{\partial \mathbf{q}} \delta \mathbf{q} \tag{3.41}$$

Those 9 values are expressed with respect to an infinitesimal variation of both the position and the orientation of the frames $\delta \mathbf{q}$, *i.e.* with respect to $\delta \mathbf{x}$ and $\delta \mathbf{r}$ ($\delta \mathbf{q}^t = [\delta \mathbf{x} \; \delta \mathbf{r}]^t$). Thus, for each coefficient i, j of $\delta \mathbf{F}^s$ one obtains:

$$\delta \mathbf{F}_{ij}^s = \sum_n \frac{\partial w_{ns}}{\bar{\mathbf{x}}_{,j}} \delta \mathbf{x}_n^i - \left(\frac{\partial w_{ns}}{\bar{\mathbf{x}}_{,j}} \left[\mathbf{R}_n \, \bar{\mathbf{l}}_{s\wedge}^n \right]^{i,} + w_{ns} \left[\mathbf{R}_n \left[\bar{\mathbf{R}}_n^t \right]^{,j}{}_{\wedge} \right]^{i,} \right) \cdot \delta \mathbf{r_n} \tag{3.42}$$

where lowercase indices correspond to a spatial coordinate and n identifies a frame.

$\mathbf{R}_n \bar{\mathbf{l}}_s^n$ corresponds to $\bar{\mathbf{l}}_s^n$ rotated by $\mathbf{R}_n$, $\mathbf{v}_\wedge$ is the antisymmetric-matrix representation of the cross product (see appendix A.2). The vector corresponding to the i^{th} row and the j^{th} column of matrix $\mathbf{A}$ are respectively noted $[\mathbf{A}]^{i,}$ and $[\mathbf{A}]^{,j}$.

The deformation gradient derivative with respect to $\mathbf{q}$ is a rank 3 tensor that can be expressed in Jacobian matrix form by rewriting the 9 values of $\delta \mathbf{F}^s$ as a column vector. Hence, for each interacting frame/sample pair, *i.e.* for each frame i and sample s so that $w_{is} > 0$, Eq. (3.42) can be written as:

$$\begin{bmatrix} \delta \mathbf{F}_{00}^s \\ \delta \mathbf{F}_{10}^s \\ \vdots \\ \delta \mathbf{F}_{22}^s \end{bmatrix} = \sum_i \mathbf{B}^{i,s} \, \delta \mathbf{q}_i \tag{3.43}$$

with $\delta\mathbf{q}_i^t = [\delta\mathbf{x}_i \; \delta\mathbf{r}_i]^t$. In addition, the 9×6 matrix $\mathbf{B}^{i,s}$ which is the deformation gradient derivative is given by:

$$\mathbf{B}^{i,s} = \begin{bmatrix} \frac{\partial w_{is}}{\partial x}\mathbf{I} & -\left[\frac{\partial w_{is}}{\partial x}[\mathbf{R}_i\bar{\mathbf{l}}^i_{s\wedge}] + [\mathbf{R}_i[\bar{\mathbf{R}}_i^t]^{,0}{}_\wedge]\, w_{is}\right] \\ \frac{\partial w_{is}}{\partial y}\mathbf{I} & -\left[\frac{\partial w_{is}}{\partial y}[\mathbf{R}_i\bar{\mathbf{l}}^i_{s\wedge}] + [\mathbf{R}_i[\bar{\mathbf{R}}_i^t]^{,1}{}_\wedge]\, w_{is}\right] \\ \frac{\partial w_{is}}{\partial z}\mathbf{I} & -\left[\frac{\partial w_{is}}{\partial z}[\mathbf{R}_i\bar{\mathbf{l}}^i_{s\wedge}] + [\mathbf{R}_i[\bar{\mathbf{R}}_i^t]^{,2}{}_\wedge]\, w_{is}\right] \end{bmatrix} \tag{3.44}$$

Having the derivative of the deformation gradient, the strain tensor derivative $\partial\mathbf{L}^{i,s}/\partial\mathbf{q}$, can be calculated for each frame/sample pair. Adopting an approach similar to the one used for $\delta\mathbf{F}^s$, one can combine Eq. (3.38) and (3.43) to express $\delta\mathbf{L}^{i,s}$ under a matrix vector product form:

$$\delta\mathbf{L}^{i,s} = \frac{\partial\mathbf{L}^{i,s}}{\partial\mathbf{q}}\delta\mathbf{q} \tag{3.45}$$

Since $\delta\mathbf{L}^{i,s}$ is a 3×3 symmetric matrix, it can be expressed as a 6-dimensional vector thanks to Voigt notation. For instance, for $\delta\mathbf{L}^{i,s}_{jk}$ with $j = 0$ and $k = 1$:

$$\begin{aligned} \delta\mathbf{L}^{i,s}_{01} &= \frac{1}{2}\left([\delta\mathbf{F}^t]^{0,} \cdot [\mathbf{F}]^{,1} + [\mathbf{F}^t]^{0,} \cdot [\delta\mathbf{F}]^{,1}\right) \\ &= \frac{1}{2}\left([\delta\mathbf{F}_{00}\; \delta\mathbf{F}_{10}\; \delta\mathbf{F}_{20}] \cdot [\mathbf{F}]^{,1} + [\mathbf{F}^t]^{0,} \cdot [\delta\mathbf{F}_{01}\; \delta\mathbf{F}_{11}\; \delta\mathbf{F}_{21}]\right) \end{aligned} \tag{3.46}$$

Furthermore, from Eq. (3.43):

$$[\delta\mathbf{F}_{00}\; \delta\mathbf{F}_{10}\; \delta\mathbf{F}_{20}] = \left[[\mathbf{B}]^{0,} \cdot \delta\mathbf{q}\; [\mathbf{B}]^{1,} \cdot \delta\mathbf{q}\; [\mathbf{B}]^{2,} \cdot \delta\mathbf{q}\right] \tag{3.47}$$

and

$$[\delta\mathbf{F}_{01}\; \delta\mathbf{F}_{11}\; \delta\mathbf{F}_{21}] = \left[[\mathbf{B}]^{3,} \cdot \delta\mathbf{q}\; [\mathbf{B}]^{4,} \cdot \delta\mathbf{q}\; [\mathbf{B}]^{5,} \cdot \delta\mathbf{q}\right] \tag{3.48}$$

substituting Eq. (3.47) and (3.48) into Eq. (3.46) and factoring by $\delta\mathbf{q}_i$ one can obtain:

$$\begin{aligned} \delta\mathbf{L}^{i,s}_{01} = \frac{1}{2}\big[&\mathbf{F}_{01}[\mathbf{B}]^{0,} + \mathbf{F}_{11}[\mathbf{B}]^{1,} + \mathbf{F}_{21}[\mathbf{B}]^{2,} \\ &+ \mathbf{F}_{00}[\mathbf{B}]^{4,} + \mathbf{F}_{10}[\mathbf{B}]^{4,} + \mathbf{F}_{20}[\mathbf{B}]^{5,}\big] \cdot \delta\mathbf{q} \end{aligned} \tag{3.49}$$

As a result, by rewriting the 6 values of $\delta\mathbf{L}^{i,s}$ in Voigt notation, one finally has:

$$\left\{ \begin{array}{c} \delta\mathbf{L}^{i,s}_{00} \\ \delta\mathbf{L}^{i,s}_{11} \\ \vdots \\ 2\delta\mathbf{L}^{i,s}_{01} \end{array} \right\} = \sum_i \mathbf{M}^{i,s}\, \delta\mathbf{q}_i \tag{3.50}$$

where each line of $\mathbf{M}^{i,s}$ has the same form as the equation (3.49).

Having Eq. (3.50) for Green-Lagrange strain tensor derivative, it can be substituted into Eq. (3.37) to obtain internal forces on frames:

$$\mathbf{f}^{\text{int}} = \int_{\Omega} \mathbf{M}^{\top} \{\mathbf{S}\} \, d\Omega \tag{3.51}$$

Eq. (3.51) is here given for the continuous case, implying that the integral has to be evaluated on the whole domain covered by the deformable object. However, it is not possible to compute this integral directly for arbitrary shapes as its expression becomes very complex or does not exists.

Thus, to enable internal force computations, Eq. (3.51) has to be converted into a discrete form. To this end, samples are used because they discretize the object's volume.

The scheme of frame internal force computation consists in accumulating the stress carried by the frame's neighboring samples and to multiplying it by the frame's strain tensor. Thus, the discrete expression of internal force to be applied on a frame i is given by:

$$\mathbf{f}_i = -\sum_{s} \left[\mathbf{M}^{i,s}\right]^{\top} \{\mathbf{S}^s\} v_s \tag{3.52}$$

where $\mathbf{S}^s$ and v_s are respectively the stress tensor and the volume associated to sample s. To ensure the proper integration of internal forces, the sample volume must be chosen so that the sum of all samples volume equals the total volume of the object.

The model used for stress computation is a hyperelastic constitutive law known as *Saint Venant - Kirchhoff* (SVK) model. It generalizes the *Hooke's law* to large deformations by relying on the Green-Lagrange strain tensor. Thus, the expression for the PK2 stress tensor is given by:

$$\mathbf{S}^s = \lambda \, tr(\mathbf{L}^s)\mathbf{I} + 2\mu\mathbf{L}^s \tag{3.53}$$

where λ and μ are the Lamé coefficients which are material dependent, and $\mathbf{L}^s$ is the Green-Lagrange strain tensor associated to sample s which is given by:

$$\mathbf{L}^s = \frac{1}{2}\left(\mathbf{F}^{s\top}\mathbf{F}^s - \mathbf{I}\right) \tag{3.54}$$

Being defined on a per sample basis, each sample can use different material parameters or even constitutive laws. The advantage brought by this approach is that it enables for heterogeneous material modeling.

In addition, this approach is not limited to the use of the SVK model. Indeed, the advantage of using the internal force expression of Eq. (3.52) is that it does not put constraints on the stress tensor to be used, except that it should be a PK2 stress tensor, which is required by the total Lagrangian formulation. As a result, this method makes the use of more advanced constitutive laws possible, but at a higher computational cost.

3.6 Temporal integration and elasticity problem solving

After computing the internal force exerted on a deformable body, another step is necessary to solve the PDE governing elasticity problems. However, at this point, two different approaches must be distinguished: static and dynamic systems.

Static approach

In the case of a static system, the primary interest is the equilibrium state, that is, how the object is deformed when the sum of both internal and external forces is zero:

$$\mathbf{f}^{\text{ext}} - \mathbf{f}^{\text{int}}(\mathbf{q}) = 0 \tag{3.55}$$

In such a case, the effect of inertia is negligible (small accelerations). This type of approach is commonly encountered in engineering for structure analysis.

This problem can be solved by applying a first order *Taylor expansion* on Eq. (3.55):

$$\mathbf{f}^{\text{ext}} - \mathbf{f}^{\text{int}}(\mathbf{q} + \partial\mathbf{q}) = \mathbf{f}^{\text{ext}} - \mathbf{f}^{\text{int}}(\mathbf{q}) + \frac{\partial \mathbf{f}^{\text{int}}(\mathbf{q})}{\partial \mathbf{q}} \partial\mathbf{q} = 0 \tag{3.56}$$

Using an iterative algorithm like the *Newton-Raphson* method one may find the displacements that satisfy Eq. (3.55). Starting with a displacement vector $\mathbf{q}_0$, one may evaluate:

$$\frac{\partial \mathbf{f}^{\text{int}}(\mathbf{q}_0)}{\partial \mathbf{q}} \partial\mathbf{q} = \mathbf{f}^{\text{int}}(\mathbf{q}_0) - \mathbf{f}^{\text{ext}} \tag{3.57}$$

to figure out the error with respect to equilibrium configuration. By successively refining the displacement vector $\mathbf{q}_{n+1} = \mathbf{q}_n + \partial\mathbf{q}$, the algorithm converges toward equilibrium. Since the positions change at each iteration, internal forces must be recomputed.

Dynamic approach

In this case, one wants to obtain all the transient states between the object's initial configuration and its equilibrium state, giving an impression of dynamic behavior to the simulation. This is done through time integration of Newton's 2nd Law of motion:

$$\mathbf{Ma} = \mathbf{f}^{\text{ext}} - \mathbf{f}^{\text{int}}(\mathbf{q}, \mathbf{v}) \tag{3.58}$$

where $\mathbf{M}$ is the mass matrix, $\mathbf{a}$ is the acceleration vector, $\mathbf{q}$ is the degrees of freedom vector of the deformable body and $\mathbf{v}$ the associated velocity vector. The internal force term has been rewritten as a general function of both degrees of freedom and velocity.

Whatever the method used, time integration cannot be done in a continuous manner. Thus, time has to be discretized into time-steps which are time intervals and forces have to be evaluated for each of them, until the deformable body reaches equilibrium state.

Explicit time integration

The simplest method for integrating Eq. (3.58) over time is known as explicit integration or *forward Euler method.* Its principle is to compute the acceleration using the force at the beginning of the time-step:

$$\begin{aligned} \mathbf{Ma}_t &= \mathbf{f}^{\text{int}}(\mathbf{q}_t, \mathbf{v}_t) \\ \mathbf{v}_{t+\Delta t} &= \mathbf{v}_t + \Delta t \mathbf{a}_t \\ \mathbf{q}_{t+\Delta t} &= \mathbf{q}_t + \Delta t \mathbf{v}_t \end{aligned} \tag{3.59}$$

where Δt is the length of the time step.

The advantage of explicit time integration is that its computational cost is relatively limited. Once the forces are obtained the computation only relies on the mass matrix which can be diagonal when using *mass lumping* techniques, resulting in a fast solving. On the other hand, the problem must be solved as fast as the mechanical system evolves. Because of that, this type of integration scheme faces hard constraints on computation time. Indeed, let's consider a node on which a force is applied at a given time-step. The position and reaction force on that node will be computed during the same time-step. However, all the adjacent nodes will be updated at the next time-step and so on. Thus, one may observe the propagation of a numerical wave trough the object. Problems may arise if the numerical wave propagates slower than the mechanical wave, in other words, if the system is solved more slowly than the object becomes deformed. In this case, at each time-step, additional energy will be introduced into the mechanical system leading the simulation to diverge because of infinite reaction forces. Hence, when using explicit integration, the time-step is generally needs to be short and must be carefully chosen. Furthermore, the stiffer the deformable object is, the faster the mechanical wave spreads, making this scheme more suitable for the simulation of very soft objects. In [HWJM10], Horton *et al.* propose an experimental formula for choosing the time-step's length which takes into account the material parameters.

Finally, as seen above, the explicit integration scheme relies on the acceleration at the begin of the time-step. This means that it neglects potentially large accelerations that may occur during the time-step, leading to the same consequences as described previously. One of the goals of the connective tissues mechanical model is to be usable in the context of training simulation where users interacts with objects, using haptic devices for example. However, this kind of interaction may produce sudden change in acceleration, causing stability problems. In addition, connective tissues are likely to be significantly stiff, as is the case for tendons. For these reasons, the use of an explicit integration scheme is not well suited for simulating connective tissues.

Implicit time integration

One way to overcome the problems posed by explicit integration is to rely on other integration schemes, such as implicit temporal integration. Initially proposed by Baraff *et al.* in [BW98] for cloth simulation, its principle is to update the position and the velocity through to the acceleration at the end of the current time-step:

$$\begin{aligned}\mathbf{M}\mathbf{a}_{t+\varDelta t} &= \mathbf{f}^{\text{int}}(\mathbf{q}_{t+\varDelta t}, \mathbf{v}_{t+\varDelta t})\\ \mathbf{v}_{t+\varDelta t} &= \mathbf{v}_t + \varDelta t\mathbf{a}_{t+\varDelta t}\\ \mathbf{q}_{t+\varDelta t} &= \mathbf{q}_t + \varDelta t\mathbf{v}_{t+\varDelta t}\end{aligned} \tag{3.60}$$

Yet, the force at the end of the time-step is not known, leaving a nonlinear problem to solve. To do so, the expression of the force has to be linearized using a 1st order Taylor expansion:

$$\mathbf{f}^{\text{int}}(\mathbf{q}_{t+\varDelta t}, \mathbf{v}_{t+\varDelta t}) \approx \mathbf{f}^{\text{int}}(\mathbf{q}_t, \mathbf{v}_t) + \frac{\partial \mathbf{f}^{\text{int}}}{\partial \mathbf{q}}\delta\mathbf{q} + \frac{\partial \mathbf{f}^{\text{int}}}{\partial \mathbf{v}}\delta\mathbf{v} \tag{3.61}$$

where $\frac{\partial \mathbf{f}^{\text{int}}}{\partial \mathbf{q}} = \mathbf{K}$ and $\frac{\partial \mathbf{f}^{\text{int}}}{\partial \mathbf{v}} = \mathbf{B}$ are respectively the system's stiffness and damping matrices. Substituting Eq. (3.61) int 1st line of Eq. (3.60) yields:

$$\begin{aligned}\mathbf{M}\mathbf{a}_{t+\varDelta t} &\approx \mathbf{f}^{\text{int}}(\mathbf{q}_t, \mathbf{v}_t) + \mathbf{K}\delta\mathbf{q} + \mathbf{B}\delta\mathbf{v}\\ &\approx \mathbf{f}^{\text{int}}(\mathbf{q}_t, \mathbf{v}_t) + \mathbf{K}(\mathbf{q}_{t+\varDelta t} - \mathbf{q}_t) + \mathbf{B}(\mathbf{v}_{t+\varDelta t} - \mathbf{v}_t)\\ &\approx \mathbf{f}^{\text{int}}(\mathbf{q}_t, \mathbf{v}_t) + \mathbf{K}(\mathbf{q}_t + \varDelta t\mathbf{v}_{t+\varDelta t} - \mathbf{q}_t) + \mathbf{B}(\mathbf{v}_t + \varDelta t\mathbf{a}_{t+\varDelta t} - \mathbf{v}_t)\\ &\approx \mathbf{f}^{\text{int}}(\mathbf{q}_t, \mathbf{v}_t) + \mathbf{K}(\varDelta t\mathbf{v}_t + \varDelta t^2\mathbf{a}_{t+\varDelta t}) + \mathbf{B}(\varDelta t\mathbf{a}_{t+\varDelta t})\end{aligned} \tag{3.62}$$

To solve the nonlinear problem of Eq. (3.60), both sides of Eq. (3.61) must be equated using an iterative algorithm such as the Newton-Raphson method to find proper acceleration. Hence, one can write:

$$\left(\mathbf{M} - \Delta t\mathbf{D} - \Delta t^2\mathbf{K}\right)\mathbf{a}_{t+\Delta t} = \mathbf{f}^{\text{int}}(\mathbf{q}_t, \mathbf{v}_t) + \mathbf{K}\Delta t\mathbf{v}_t \tag{3.63}$$

However, in practice, this system is generally expressed using velocity. By setting $\delta\mathbf{v} = \Delta t\mathbf{a}$ one obtains the following system:

$$\underbrace{\left(\mathbf{M} - \Delta t\mathbf{D} - \Delta t^2\mathbf{K}\right)}_{\mathbf{A}}\underbrace{\delta\mathbf{v}}_{\mathbf{x}} = \underbrace{\Delta t\mathbf{f}^{\text{int}}(\mathbf{q}_t, \mathbf{v}_t) + \Delta t^2\mathbf{K}\mathbf{v}_t}_{\mathbf{B}} \tag{3.64}$$

Implicit time integration enables stable simulations using larger time-steps which is interesting for simulations involving user interactions. Nonetheless, this temporal integration scheme is significantly more costly than explicit time integration, because the system of Eq. (3.64) must be constructed and solved at each time-step. Furthermore, when dealing with nonlinear elasticity, the stiffness matrix is not constant over the simulation and must also be recomputed at each time-step, introducing an additional cost.

Differentiating force expression

Implicit time integration requires to linearize the force equation. Depending on the type of solver used in simulations, it may be expressed in different ways. On the one hand, direct solvers will need it to be expressed in matrix form. On the other hand, when using iterative solvers like the conjugate gradient, it will be expressed as a variation of internal force with respect to a variation of degrees of freedom.

In both cases, internal forces have to be differentiated with respect to degrees of freedom. From [BLM00](§6.4.2), the internal force derivative can be separated into two parts:

$$\delta\mathbf{f}^{\text{int}} = \delta\mathbf{f}^{\text{mat}} + \delta\mathbf{f}^{\text{geo}} \tag{3.65}$$

where $\delta\mathbf{f}^{\text{mat}}$ is the *material tangent stiffness* and $\delta\mathbf{f}^{\text{geo}}$ is the *geometric stiffness*. Hence, differentiating Eq. (3.52) leaves:

$$\frac{\partial\mathbf{f}}{\partial\mathbf{q}} = \sum_s \left(\underbrace{\frac{\partial\left[\mathbf{M}^{i,s}\right]^\top}{\partial\mathbf{q}}\{\mathbf{S}^s\}}_{\text{geometric stiffness}} + \underbrace{\left[\mathbf{M}^{i,s}\right]^\top\frac{\partial\{\mathbf{S}^s\}}{\partial\mathbf{q}}}_{\text{material stiffness}} \right) \tag{3.66}$$

thus, the expression for the material tangent stiffness is given by:

$$\delta\mathbf{f}_i^{\text{mat}} = \sum_s \left[\mathbf{M}^{i,s}\right]^\top\{\delta\mathbf{S}^s\}\,v_s \tag{3.67}$$

with $\{\delta \mathbf{S}^s\}$ being the stress tensor derivative in Voigt notation:

$$\{\delta \mathbf{S}^s\} = \sum_j \mathbf{C}\left\{\delta \mathbf{L}^{j,s}\right\} \tag{3.68}$$

where $\mathbf{C}$ is a 6×6 matrix associated to Hooke's law whose coefficients depend on material parameters. Substituting Eq. (3.50) into Eq. (3.68) yields:

$$\{\delta \mathbf{S}^s\} = \mathbf{C}\sum_j \mathbf{M}^{j,s}\delta \mathbf{q}_j \tag{3.69}$$

Similarly, the expression for the material tangent stiffness matrix between frames i and j, which is a 6×6 matrix, is given by:

$$\mathbf{K}_{ij}^{\text{mat}} = \sum_s \left[\mathbf{M}^{i,s}\right]^\top \mathbf{C}\left[\mathbf{M}^{j,s}\right] v_s \tag{3.70}$$

As for the geometric stiffness which encodes the variation of internal forces due to the effects of geometric deformation (rotation, stretching, *etc.*), its expression is given by:

$$\delta \mathbf{f}_i^{geo} = -\sum_s \frac{\partial \left[\mathbf{M}^{i,s}\right]^\top}{\partial \mathbf{q}}\{\mathbf{S}^s\} v_s \tag{3.71}$$

By looking at equation (3.71), one can notice that the matrix $\mathbf{M}$ has to be differentiated with respect to $\mathbf{q}$. To better illustrate the process of this differentiation, we can examine the last component of the geometric stiffness (*i.e.* $\delta \mathbf{f}^{geo}$) as an example:

$$\mathbf{f}_{i_5}^{geo} = -\sum_s \frac{\partial \left[\delta \mathbf{M}^{i,s}\right]^{,5}}{\partial \mathbf{q}} \cdot \{\mathbf{S}^s\} v_s \tag{3.72}$$

In other words, the 6th component of the geometric stiffness is the dot product of the 6th column of $\delta\mathbf{M}$ by $\mathbf{S}$. Without loss of generality, the i^{th} component of the geometric stiffness is the dot product of the i^{th} column of $\delta\mathbf{M}$ and $\mathbf{S}$. Developing equation (3.72) leads to:

$$\begin{aligned}\mathbf{f}_{i_5}^{geo} = -\sum_s &\left[\mathbf{S}_0^s(\delta\mathbf{F}_{00}\mathbf{B}_{05} + \mathbf{F}_{00}\delta\mathbf{B}_{05} + \cdots\right.\\ &\left.+ \mathbf{S}_5^s(\cdots + \delta\mathbf{F}_{20}\mathbf{B}_{55} + \mathbf{F}_{20}\delta\mathbf{B}_{55})\right] v_s \cdot \delta \mathbf{q}_i\end{aligned} \tag{3.73}$$

with $\delta\mathbf{B}_{ij} = \partial\mathbf{B}_{ij}/\partial\mathbf{q}$. One can notice that equation (3.73) involves the deformation gradient derivatives, already known by evaluating equation (3.43), but also the derivatives of the $\mathbf{B}$ matrix coefficients.

For instance, let $\delta\mathbf{B}_{05} = [\delta\mathbf{B}^{\mathbf{x}}_{05}\, \delta\mathbf{B}^{\mathbf{r}}_{05}]$, the concatenation of two 3-dimensional vectors. Since the derivative of all the coefficients of the right half of $\mathbf{B}$ have the same form, we can differentiate $\delta\mathbf{B}^{\mathbf{r}}_{05}$ as an example:

$$\delta\mathbf{B}^{\mathbf{r}}_{05} = -\frac{\partial w_{is}}{\partial x}\frac{\partial\left[\mathbf{R}_i\,\bar{\mathbf{l}}^i_{s\wedge}\right]^{0,2}}{\partial\mathbf{q}} - w_{is}\frac{\partial\left[\mathbf{R}_i[\bar{\mathbf{R}}^t_i]^{,0}_{\wedge}\right]^{0,2}}{\partial\mathbf{q}} \tag{3.74}$$

where $\left[\mathbf{R}_i\,\bar{\mathbf{l}}^i_{s\wedge}\right]^{0,2}$ is the component at line 0 and column 2 of the matrix $\mathbf{R}_i\,\bar{\mathbf{l}}^i_{s\wedge}$. Proceeding with the differentiation results in:

$$\begin{aligned}\delta\mathbf{B}^{\mathbf{r}}_{05} &= \left(-\frac{\partial w_{is}}{\partial x}\left[-\mathbf{R}_i\,[\bar{\mathbf{l}}^i_{s\wedge}]^{,2}_{\wedge}\right]^{0,} - w_{is}\left[-\mathbf{R}_i\left[[\bar{\mathbf{R}}^t_i]^{,0}_{\wedge}\right]^{,2}_{\wedge}\right]^{0,}\right)\delta\mathbf{r}\\ &= \left(\frac{\partial w_{is}}{\partial x}\left[\mathbf{R}_i\,[\bar{\mathbf{l}}^i_{s\wedge}]^{,2}_{\wedge}\right]^{0,} + w_{is}\left[\mathbf{R}_i\left[[\bar{\mathbf{R}}^t_i]^{,0}_{\wedge}\right]^{,2}_{\wedge}\right]^{0,}\right)\delta\mathbf{r}\end{aligned} \tag{3.75}$$

and $\delta\mathbf{B}^{\mathbf{x}}_{16}$ is a null 3-dimensional vector and the differentiation all the coefficients of the left half of the $\mathbf{B}$ matrix results in a null vector.

The expression for the geometric stiffness matrix is much more complex than its material tangent stiffness counterpart and will be given in appendix B. Geometric stiffness can be split into two parts. The first one mainly encodes the influence of frame's self rotation, as it contains the $\mathbf{B}$ matrix coefficients derivatives which, as shown in Eq. (3.75), are only related to rotations. As for the second part, it describes the influence of both linear and angular displacements.

Discussion

In order to provide stable simulations, the 6-DoF model for connective tissues has been adapted to implicit temporal integration (see section 3.6). However, some limitations remain for using it along with an implicit integration, especially in the case of a dynamic system. Indeed, in such a case, building the global system matrix requires the construction of the mass matrix, which has not been studied. However, since it does not involve the mass matrix, it can be used in a quasi-static approach. This still makes sense since large accelerations are rarely encountered in surgical simulations. In addition, using an implicit integration scheme is still useful for providing a stable simulation of stiffer structures like tendons.

3.7 Summary

In this chapter we presented a sensitivity study investigating the influence of different boundary condition models. Its results show that, depending on the biomechanical model chosen, simulation's outcomes exhibit sensible differences. In the human body, connective tissues are mainly responsible for the boundary conditions on the organs they surround. Hence, this study has highlighted the need for dedicated biomechanical models, in order to properly simulate boundary condition.

Core principles of continuum mechanics and their application to deformable bodies dynamics have been presented. Then, we have derived these principles to develop a mechanical model based on 6-DoF nodes. Thanks to our approach, this model is not restricted to any constitutive law. As a consequence, it enables the simulation of any type of material.

In the next chapter, we will discuss the application of this model in the context of medical simulation. We will show that it exhibits interesting properties making it a good fit for connective tissues simulation. We will also show that thanks to the physically-based approach, it provides accurate results.

Chapter 4

Connective tissues simulation

In this chapter, the 6-DoF mechanical model is further developed and applied to the simulation connective tissues. In a first part, we address the ability of the method to be used in both mesh-based or mesh-free schemes, as well as the reasons to use of each, and their impact on objects modeling.

Then, we discuss the need to propose suitable interactions and particularly the issue of stability. This is important in that the method will be used for surgical simulators, including those dedicated to training. To this end, implicit temporal integration scheme is applied to the mechanical model presented in the previous chapter.

In a later section, we present the results of both the tests conducted for the validation of the method and the experiments on anatomical structures coupling.

Computational efficiency is a topic of major importance, especially in the field of surgical training simulators, because it provides users with a better experience. Thus, in a last section, we describe a GPU implementation of this method that makes it usable for real-time applications.

4.1 Shape functions

Connective tissues fill the volume between certain organs and may exhibit a wide range of topologies. For instance, soft tissues like fat are generally observed as volumetric structures (*i.e.* with non-negligible extent in the 3 dimensions). On the other hand, other anatomical structures, like ligaments and tendons, are much more likely to have curve of surface-like geometries. In some cases, these bodies may expose different types of geometries in the same structure, by having both volume and surface parts.

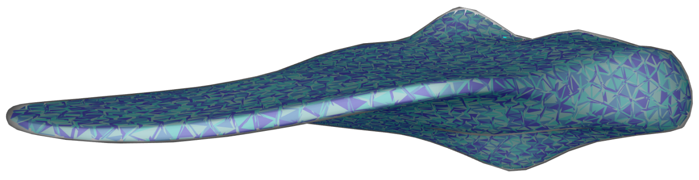


Figure 4.1: An accurate discretization of connective tissue's thinest part requires a large number of elements (here 26665 tetrahedra).

In the former case it is possible to discretize rather easily the soft tissues using a volumetric mesh with good quality elements. In the latter case volumetric meshing is still feasible, but comes with serious disadvantages. Indeed, ligaments and tendons may have very thin parts making meshing procedure a tedious task when trying to find optimal parameters. An accurate discretization of those tissues would require using small elements which would greatly increase the element count for the whole object as shown on figure 4.1. This may hinder real-time simulations required for interactive applications like training simulators. On the other hand, if the number of elements is limited by the user, bad shaped elements are likely to appear, involving ill-conditioning problems. In both cases, simulation's performances are negatively impacted. Thus, modeling of such soft tissues may be easier with mesh-free methods.

To solve these problems, the idea is to propose a method that can be used with a mesh-based or mesh-free scheme. As a result, the mechanical model based upon 6-DoF nodes can simulate connective tissues without prior assumption of their geometry.

The core principle of *Galerkin* mesh-free methods only differs from mesh-based approaches by the type of connectivity. The mesh-free methods rely on unstructured connectivity, where the latter use structured connectivity, resulting from the topology of the object's mesh. Still, mesh-free methods are often misleadingly described as having no connectivity between nodes.

The Galerkin method is a technique for solving PDE similar to the weighted residual method. More precisely, the 6-DoF mechanical model is based on the *Bubnov-Galerkin* method. Its principle states that the weak form of elasticity problem must be formulated using a test function which is the same as the trial function. For this reason, the shape functions defined over the object have to be used for all the computations (deformation gradient, strain/stress tensor *etc.*).

Finally, the method for simulating connective tissues abstracts the role of the shape functions. Because of this, the choice of a mesh-based, mesh-free or mixed scheme only depends on the choice of the shape function to be used. In the following sections, mesh-based and mesh-free shape functions are described.

Mesh-based simulation scheme

The use of the mesh-based scheme is similar to the 3-DoF nodes FEM approach and relies on the elements that constitute the object's volumetric mesh. These elements can be tetrahedra, hexahedra, prisms, *etc.* whose vertices are the frames of the object's model.

Weights between frames and samples are computed from the barycentric coordinates of the samples within the elements they belong to. The simplest form of shape function that can be used on a volumetric mesh is the linear interpolation which requires tetrahedral elements. Hence, displacement $\mathbf{u}$ at a point $\mathbf{x}$ that belongs to a tetrahedron defined by vertices $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3$ is given by:

$$\mathbf{u} = w_0\mathbf{u}_0 + w_1\mathbf{u}_1 + w_2\mathbf{u}_2, +w_3\mathbf{u}_3 \tag{4.1}$$

where the $\mathbf{u}_i$ are the displacements associated to tetrahedron's vertices.

Thus, finding the weights amounts to finding coefficients w_i so that the linear combination of tetrahedron's vertices equals the position of sample s:

$$\mathbf{p}_s = w_0\mathbf{x}_0 + w_1\mathbf{x}_1 + w_2\mathbf{x}_2, +w_3\mathbf{x}_3 \tag{4.2}$$

or in matrix form:

$$\mathbf{p}_s = \mathbf{P}\begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix}$$

where

$$\mathbf{P} = \begin{bmatrix} x_0 & x_1 & x_2 & x_3 \\ y_0 & y_1 & y_2 & y_3 \\ z_0 & z_1 & z_2 & z_3 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Hence, weights are given by:

$$\begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} = \mathbf{P}^{-1}\mathbf{p}_s \tag{4.3}$$

It is interesting to note that Eq. (4.3) reveals the problem that may arise with badly shaped tetrahedra. For instance, if a tetrahedron is almost flat, the $\mathbf{P}$ matrix is almost rank deficient, making its inverse less precise and leading to a poorer interpolation.

The same scheme applies to the trilinear interpolation used on meshes discretized using hexahedral elements. However, in this case, 8 weights have to be found and the $\mathbf{P}$ matrix becomes:

$$\mathbf{P} = \begin{bmatrix} x_0 & \cdots & x_7 \\ y_0 & \cdots & y_7 \\ z_0 & \cdots & z_7 \\ x_0y_0 & \cdots & x_7y_7 \\ y_0z_0 & \cdots & y_7z_7 \\ x_0z_0 & \cdots & x_7z_7 \\ x_0y_0z_0 & \cdots & x_7y_7z_7 \\ 1 & \cdots & 1 \end{bmatrix} \tag{4.4}$$

Here, the term “linear” refers to the ability of the shape function to reproduce perfectly a function of degree 1. However, it possible to use higher order shape function to reproduce complex functions more faithfully. In this case, the elements needs to be made up of more nodes (*e.g.* 10 nodes for a quadratic tetrahedra). This way a system of equation can be build to find weights, implying heavier computations. More generally, shape functions in conventional mesh-based methods are polynomial.

When the 6-DoF mechanical model is used in a mesh-based scheme, the samples are placed inside tetrahedra by directly specifying their barycentric coordinates. Hence, weights between frames and samples are already computed. As for numerical integration of internal forces, the volume of an element is equally distributed over the samples it contains. It should be noted that this approach is not optimal. Indeed, optimal quadrature are computed using the *Gaussian quadrature rule* [ZT00](§9.8.2) and may be found in numerous abacus. However, the simplification done here is not relevant for the application's targeted accuracy.

Meshless simulation scheme

Meshless methods are an alternative to FEM for solving PDEs. Unlike mesh-based methods, mesh-free schemes rely on unstructured connectivity. Indeed, the neighborhood of a node is defined by the set of points that lay within a finite region around it. As a result, a node may have a varying number of neighbors.

In mesh-free methods, numerous types of shape functions are available. The simplest form of shape function is the *Radial Basis Function* (RBF) sometimes called *kernels*, as used in the SPH. They are often encountered as Gaussian-like functions and are generally based on inverse distance functions. These kernels may be bounded and, in this case, are described as compactly supported and come with an influence radius. Points outside the radius of a node are not subject to its influence. Otherwise, they are described as globally supported, meaning that each node is influenced by every other node in the simulation, which may considerably increase the computational cost. On the other hand, globally supported kernels exhibits high convergence rates [DCNP04].

The *Moving Least Squares* (MLS) is a popular way of building shape functions in mesh-free methods. For instance, it has been used in the EFG [BL94], the *MLPG* [AZ98] and the MFS [DB00]. Initially introduced by Lancaster and Salkauskas in [LS81] for interpolating data, their principle is to perform a local approximation through a least square approximation on a support moving all over the domain. Without going deeply into mathematical grounding, the MLS approximation of an unknown function u can be written:

$$\tilde{u}(\mathbf{x}) = \mathbf{p}^\top(\mathbf{x})\left[\mathbf{M}(\mathbf{x})\right]^{-1}\mathbf{B}(\mathbf{x})\mathbf{u} \tag{4.5}$$

with $\mathbf{u}$ vector of nodal values and

$$\mathbf{M}(\mathbf{x}) = \sum_i^n w(\mathbf{x}-\mathbf{x}_i)\mathbf{p}(\mathbf{x})\mathbf{p}^\top(\mathbf{x}) \tag{4.6}$$

and

$$\mathbf{B}(\mathbf{x}) = \left[w(\mathbf{x}-\mathbf{x}_0)\mathbf{p}(\mathbf{x_0})\cdots w(\mathbf{x}-\mathbf{x}_{n-1})\mathbf{p}(\mathbf{x_{n-1}})\right] \tag{4.7}$$

where $w(\cdot)$ is a compactly supported kernel and $\mathbf{p}(\mathbf{x})$ is the *complete polynomial basis* vector, whose dimension depends on the order of the approximation and the dimension of the problem. For a second order approximation, this vector contains all the monomials of degree inferior or equal to 2. For instance, for a second order approximation in 2 dimensions the complete basis is:

$$\mathbf{p}(\mathbf{x}) = \left[1\ x\ y\ x^2\ xy\ y^2\right]^\top$$

A special case of MLS approximation is *Shepard's method* also known as *Inverse Distance Weighting* (IDW) which is a degree 0 MLS approximation:

$$\mathbf{u}(\mathbf{x}) = \frac{\sum_i^N w_i(\mathbf{x},\mathbf{x}_i)\mathbf{u}_i}{\sum_i^N w_i(\mathbf{x},\mathbf{x}_i)} \tag{4.8}$$

with

$$w_i(\mathbf{x},\mathbf{x}_i) = \frac{1}{d(\mathbf{x},\mathbf{x}_i)^p} \tag{4.9}$$

where d is a distance function and p a real positive number. A particularity of Shepard's method is that the value of $w_i(\mathbf{x}, \mathbf{x}_i)$ tends to infinity as $\mathbf{x}$ gets closer to $\mathbf{x}_i$. Furthermore, there is no restriction on the distribution of neighboring nodes. As a result, nodes can be collinear or coplanar.

The advantage of the MLS is that they allow the building of high order approximation (*i.e.* using high order polynomials) provided that enough points are available in the support region. However, it should be noted that, to obtain weights, as the order of the approximation grows, so does the size of the system to solve for each integration point, leading to heavy computations. Another one of the main problems of the MLS is that they fail at handling some spatial distributions of points, especially coplanar and collinear arrangements. In this case, the *moment matrix* $\mathbf{M}$ of Eq. (4.6) is rank deficient and thus no longer invertible, hindering shape function computation. To solve this problem, one may rely on *Generalized Moving Least Squares* (GMLS). It is a derivation of the MLS that takes into account derivative values at samples location for the shape function computation. This technique has been used by Atluri *et al.* in [ACK99] for thin beam analysis and by Martin *et al.* in [MKB$^+$10]. It also worth mentioning that, as presented in [LS81], MLS shape functions are only approximating, which means that they lack the *Kronecker delta property*:

$$\Phi_i(\mathbf{x}_j) \neq \delta_{ij} \tag{4.10}$$

in other words, the value of the shape function at an object's node is not 1. However, this issue has been tackled by Netuzhylov and Zilian in [NZ09], where the authors propose to construct interpolating shape functions using MLS.

Other methods are available to build shape functions that are truly interpolating. One of them is the *Natural Neighbor Interpolation* (NNI) or *Sibson Interpolation* whose principle is to build an interpolation between a node and its direct neighbors [SM99], [SMSB01]. Yet, it is difficult to classify it as a mesh-free interpolation because it uses a Voronoï diagram, which is the dual of the Delaunay triangulation used to generate volumetric meshes. Another interpolating technique can be obtained by the used of *maximum entropy* functions. In [Suk04], Sukumar describes the construction of polygonal interpolants which enables the computation of the shape functions for a point lying in a generalized polygon.

Table 4.1 summarizes the principal properties of these shape functions.

Domain sampling

When used in a mesh-free manner, the object's volume is sampled using *Poisson distribution* [Bri07], illustrated on figure 4.2. The principle is to generate a set of positions so that the average distance between each point is equal to a

Method	***Type***	***Interp.***	***colinear/coplanar***	***consistency***
RBF	Gaussian	✗	✓	0
Shepard	inv. dist.	✗	✓	0
MLS[1]	polynomial	✗	✗	up to n
IMLS	polynomial	✓	✗	up to n
MaxEnt	polynomial	✓	✗	1
NNI	polynomial	✓	✗	1

Table 4.1: Summary of meshless shape function's properties.

given length. In this case, a boundary surface mesh is provided to define the geometrical domain of connective tissues. Then, this domain is voxelized and two different samplings, based on inter-particle distance, are generated. The first one serves as a frames coordinates set, whereas the second one, generated using a shorter inter-particle distance, is used as samples coordinates. To enable the numerical integration of internal forces, the object's total volume is uniformly distributed over the whole samples set.

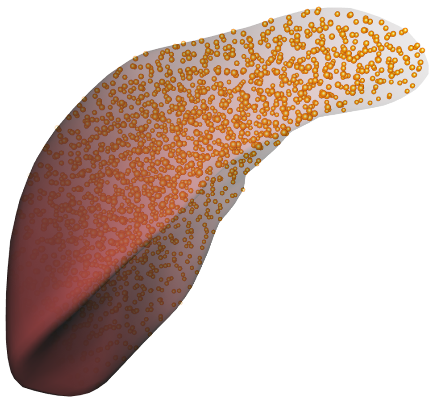


Figure 4.2: A liver model sampled using Poisson distribution.

Once the object's domain has been sampled, the chosen shape function is applied to compute weights between frames and samples. The use of total Lagrangian approach has several advantages. Firstly, both the force and the shape function computations are done with respect to the rest configuration, meaning that their values can be computed during the initialization step. In turn, this means that computationally heavy shape functions can be used without slowing down the simulation. Secondly, thanks to the total Lagrangian formulation, this method is similar to Lagrangian kernels methods which provide better stability. Indeed, it has been shown in [BGLX00] that Lagrangian kernels allows to avoid tensile instability phenomenon which is likely to appear when using mesh-free methods for simulating large deformations.

Comparison with other meshless methods

Finally, in order to characterize this method, some comparison elements with other techniques are provided. Since in this method the test function and the shape function are the same (Bubnov-Galerkin approach), one may consider it as an extension of the EFG method to 6-DoF nodes. However, our approach for the internal forces integration over the object's domain is different. Indeed, EFG is not totally mesh-free since it relies on a background mesh (generally a regular grid) filled with quadrature points to perform numerical integration [BL94]. When used in a meshless scheme, our method does not require background mesh.

In our approach, the integration of a frame's internal force on is done by using the samples within its support radius. Hence this method could be assimilated as a form of Meshless Local Petrov-Galerkin method (MLPG) which uses the node's local support as integration domain [AZ98]. However, the original MLPG method requires the test the shape functions to be different which is not the case in our approach. Still, Yet, a variant of MLPG, called MLPG6, is a particular case of Bubnov-Galerkin method similar to EFG without background integration mesh. Moreover, our approach shares some similarities with the Method of Finite Spheres (MFS) because the integration is performed on frame's support, which can be considered as a sphere.

4.2 Mechanical coupling

In this section, the application of the 6-DoF model for the mechanical coupling between organs is presented. Depending on their shape and their properties, anatomical structures can be simulated using different models, sometimes implying different types of degrees of freedom. In the following, we detail the method for coupling between thin walled organs, volume organs and rigid structures.

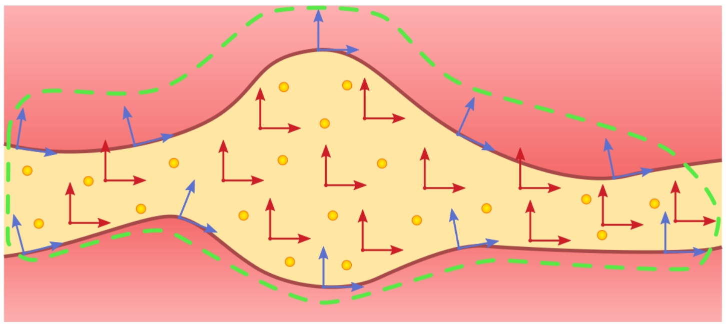


Figure 4.3: Mechanical coupling between two organs. The frames of the lower and the upper surface (blue) inside the green area are shared with the frames of the connective tissues (red).

Thin walled organs

Within the human body, many organs can be modeled as thin walled structures like the stomach, the bladder, the uterus (see section 4.3), *etc.* These organs can be modeled using shell elements [CCD10], that also relies on 6-DoF nodes. Our method takes advantage of this fact to enable the mechanical coupling between such a type of organs.

The chosen approach consists in sharing the frames of both the organs and the connective tissues as shown on figure 4.3. In other words, the DoF of all the structures are merged into the same mechanical system. The procedure for coupling these organs can be decomposed in two steps. First, the domain corresponding to connective tissues has to be defined by filling the region with frames and samples. Then, in a second step, the frames of the organs surfaces that should be linked to the connective tissues must be defined. Figure (4.3) illustrates this mechanism: connective tissues (yellow) have been sampled with frames (red) and samples. The surfaces of the top and bottom organs (pink) have been modeled with frames (blue). This results in a mechanical object containing organs and connectives tissues frames as well as the samples of the latter. It is then possible to compute the shape functions between frames and samples. In this case, the samples near the interface between the organs and the connective tissues will accumulate stress as explained in the previous chapter. This stress, as depicted on figure 4.4, will then participate to the force to be applied on the frames of both organs surfaces and connective tissues. This enables the mechanical coupling and the propagation of the boundary conditions.

The coupling between organs is made easier when using meshless shape functions as it does not requires connectivity information. Thanks to the unstructured connectivity of these shape functions, samples are "automatically"

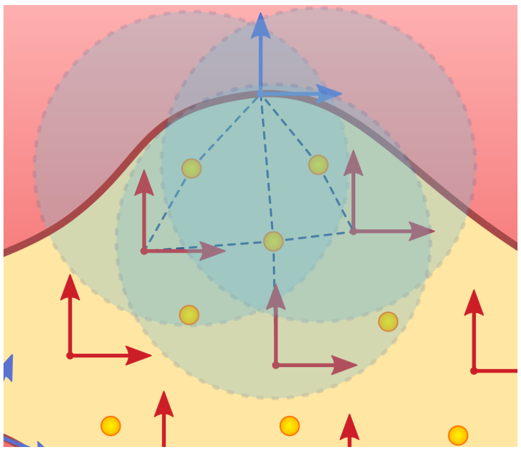


Figure 4.4: The samples near the interface enables the coupling of organs and connective tissues frames within their support radius.

linked to the frames lying in their support radius. As a consequence, the connective tissues and organs frames are implicitly linked without requiring the user to provide connectivity informations. Going further, the meshless approach enables an easier coupling of multiple organs at the same time. However, when dealing with thin walled organs, the mesh-based approach is slightly more complex. Indeed, it requires the merged object to be meshed which, as detailed previously, can be inefficient for some connective tissues topologies.

From a computational point of view, the application of forces depends on the type of solver used. In the case of an iterative solver, the organs' frames accumulate the forces issued from their own model (*i.e.* the shells), and the ones issued from the connective tissues. A similar scheme is used when using direct solver. In this case, the stiffness matrix of the shell elements and connective tissues will be added to the global stiffness matrix at rows and columns corresponding to the organs frames as illustrated on figure 4.5. Then the stiffness matrices of the connective tissues are added to the frames at the interface.

Volume organs

Many anatomical structures are volume objects like liver, muscles *etc.* These are naturally modeled using volume elements based on 3-DoF nodes and simulated using FEM. However, in this case, the merging approach described previously cannot apply because the connective tissues and the organs do not share the same type of DoF. Thus, the FE nodes on the organs' surface no longer undergo the influence from the samples of the connective tissues.

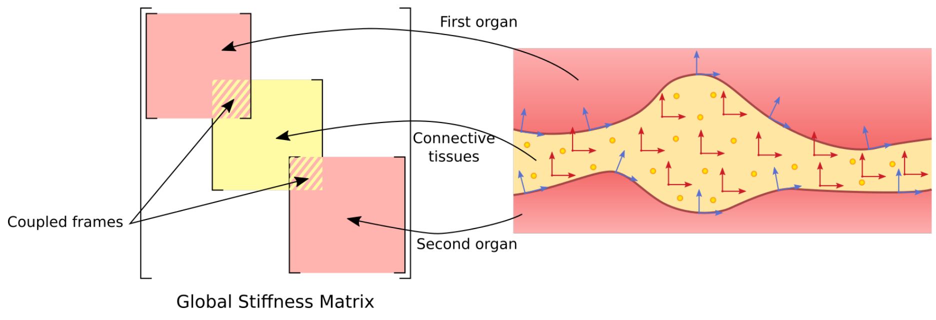


Figure 4.5: Global stiffness matrix assembly with DoF sharing approach.

The idea developed to alleviate this problem is to kinematically link the connective tissues frames near the interface to the boundary FE nodes. In this case, connective tissues are still modeled as previously described with their volume filled by samples and frames. Then, organs are partitioned in two sets: one containing the boundary nodes to be coupled and the other containing the bulk nodes. Then, the boundary nodes are linked to the boundary frames thanks to a *mapping*.

A mapping $\mathcal{M}$ is a function that computes the position of a set of nodes from the position of another set of nodes:

$$\mathbf{p} = \mathcal{M}(\mathbf{q}) \tag{4.11}$$

where $\mathbf{p}$ and $\mathbf{q}$ are two vectors of DoF.

In our case $\mathcal{M}(\cdot)$ corresponds to the function given by Eq. (3.34), $\mathbf{p}$ is the position of the boundary FE nodes and $\mathbf{q} = [\mathbf{x}\ \mathbf{r}]$ is the position and orientation of the boundary frames. This mapping relies on shape functions. Hence, by using, shape functions with compact support, the deformation keeps local character. This link is reciprocal: the FE boundary nodes transfer force $\mathbf{f}$ and torque $\boldsymbol{\tau}$ to the frames they are linked to. Hence, for a boundary frame i:

$$\mathbf{f}_i = \sum_n w_{in}\mathbf{f}_n \tag{4.12}$$

$$\boldsymbol{\tau}_i = \sum_n w_{in}(\mathbf{p}_n - \mathbf{x}_i) \times \mathbf{f}_n \tag{4.13}$$

where $\mathbf{f}_n$ is the force applied on FE boundary nodes and w_{in} is the value of the shape function between the frame i and the FE node n. This mechanism is depicted on figure 4.6.

Despite being mapped, boundary FE nodes are influenced by the FE elements they belong to. The boundary frames are also influenced by the other frames of

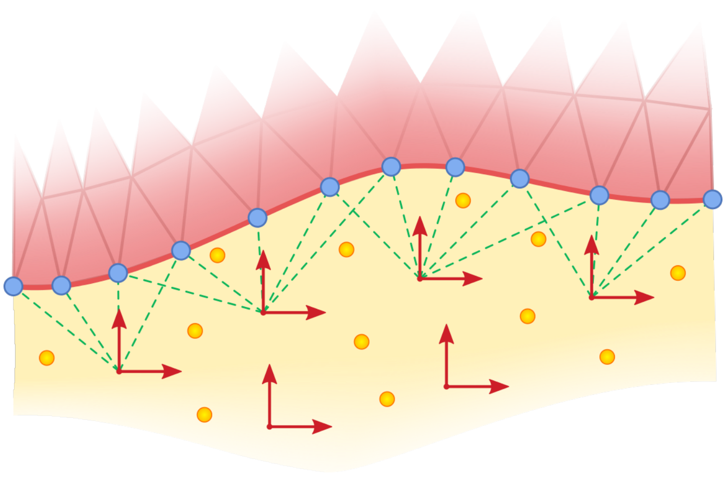


Figure 4.6: Coupling between volume object and connective tissues. The boundary FE nodes (blue) are linked to boundary frames (green lines).

the connective tissues according to the 6-DoF model presented in the previous chapter.

Thus, when a boundary FE node undergoes a displacement, this motion is transmitted to the connective tissues frames. Frames in turn transmit the force to the other organ. In addition, the product of the force by the displacements (see virtual work principle section 3.4) is the same on both sides of the interface. As a consequence, boundary conditions are propagated from one organ to another thanks to the connective tissues.

Rigid structures

Rigid bodies can be modeled using one frame as they are not subject to any deformation. This frame can be located at the center of mass of the object or at its center of rotation in the case of a bone that compose a joint. However, rigid bodies do not have frames on their surface. Hence, here again, the DoF sharing approach cannot be used.

In that case, the frames at the interface (*i.e.* which lie on the object surface) have a rigid motion determined by the one of the rigid body. Thus, to link rigid bodies, these frames are attached to the latter using bilateral constraints as shown on figure 4.7. The free motions (*i.e.* without constraints) of both rigid bodies and connective tissues are computed. Then, the constraint solver enforce the application of these constraints and links all the structures.

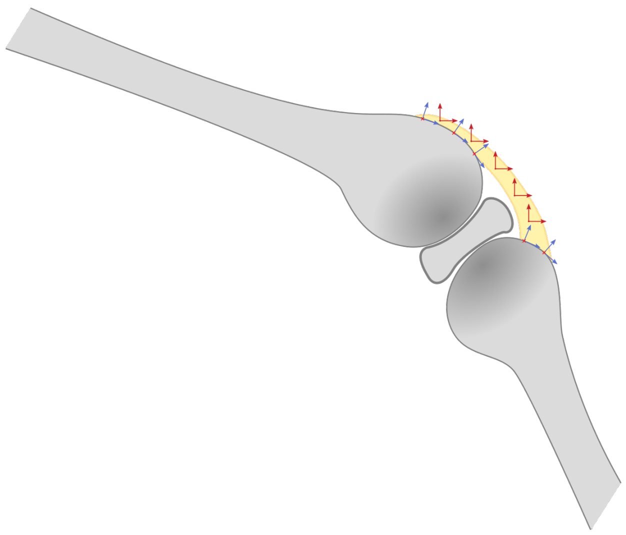


Figure 4.7: Coupling between rigid structures using bilateral constraints. Red crosses indicate the frames attached to the rigid bodies.

4.3 Results

In this section the results of the experiments and validation tests are presented. In a first part, experiments are conducted to demonstrate the method's ability to handle objects with various geometries such as linear, surface and volumetric models. Then, to assess its accuracy, the method is compared to the standard finite elements method, showing an increased convergence rate while using less degrees of freedom. Finally, the method is shown in its main role, namely the mechanical coupling of anatomical structures.

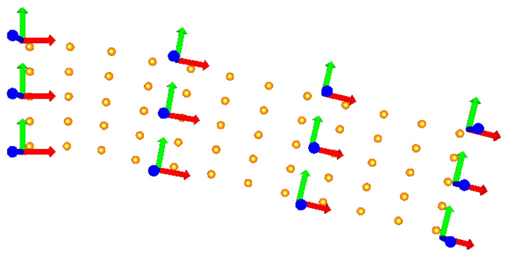


Figure 4.8: Slice of a beam modeled using frames and samples (orange dots).

Experimentation

The first test consists in the simulation of volumetric objects by modeling a beam. The cross section of the latter is modeled using 9 frames and 36 samples which are located between the frames. The beam is anchored at one of its ends and the only external force experienced is due to its own weight. Figure 4.8 shows the beam modeling and figure 4.9 shows the beam simulated with 3 different Young's modulus values and a constant Poisson's ratio ($\nu = 0.3$).

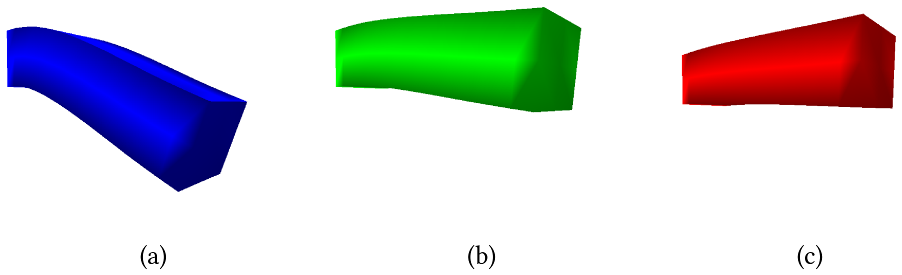


Figure 4.9: Beams simulated with different Young's modulus: E = (a) 3000 Pa, (b) 9000 Pa, (c) 18000 Pa.

The second test deals with the simulation of surface like objects. The plane depicted on figure 4.10 is modeled using 91 frames and 1024 samples. Both left and right sides frames are fixed (no motion in translation or rotation). The deformation shown on this figure is the result of a simulation where a 300 N constant force, perpendicular to the plane and oriented downward, has been applied on the frames highlighted by the orange dots. This simulation has been performed using a meshless approach and Shepard's functions were used as shape functions. As for material parameters, Young's modulus for the surface is 25000 Pa and its Poisson's ratio is 0.3. Despite the coplanar arrangement of both frames and samples, the simulation remains stable.

Finally, the third experiment aims at evaluating the usability of the method for the simulation of linear objects. Figure 4.11 shows an example of such a structure made of 10 frames and 50 integration points. The leftmost frame is fixed in rotation and translation. At the beginning of the simulation, all the frames have the same orientation, which is identical to the leftmost one. During the simulation, torsion is applied on the other end, such that the rightmost frame rotates 180 degrees. For this experiment, no gravity is applied and Shepard's functions are also used as shape functions which make handling the colinear distribution of points possible. One may observe that the torsion spreads along the structure. The method enables stable simulations of linear structures, even with a small Young's modulus.

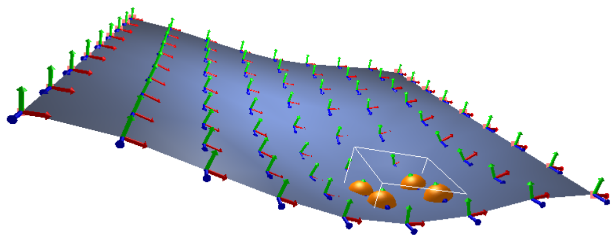


Figure 4.10: A surface under traction modeled using 6-DoF.

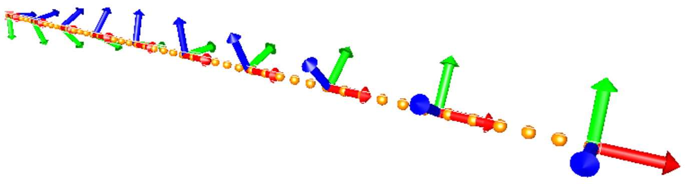
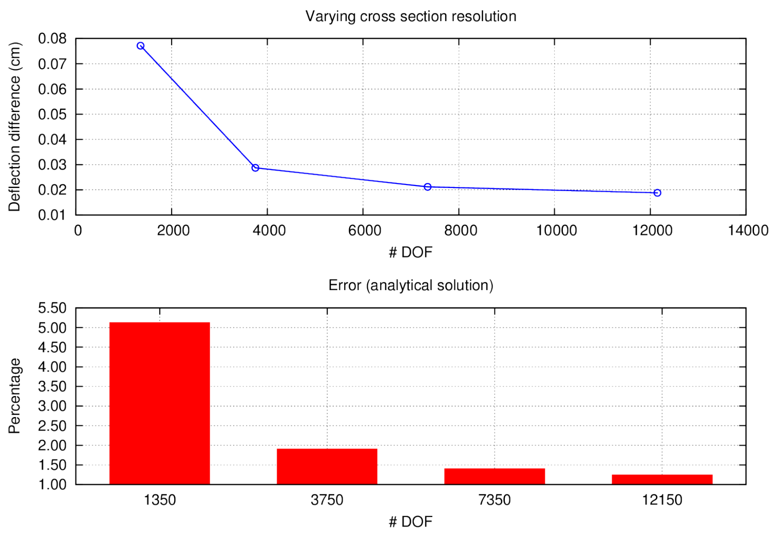


Figure 4.11: A linear structure under torsion.

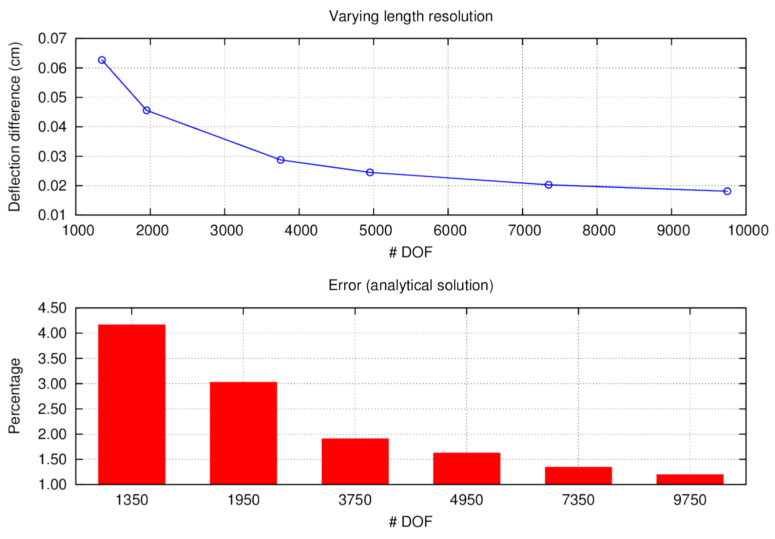
Numerical verification

In order to assess the accuracy of the model, multiple tests have been conducted. The first one consists in verifying the model's convergence in the case of small displacements. To this end, a beam, only subject to its own weight, whose measurements are $16 \times 0.5 \times 0.5$ meters, is simulated with material parameters similar to the ones of steel. The beam's Young's modulus is 200 GPa, its Poisson's ratio is 0.3 and its mass 31200 kg. The large dimensions and heavy weight of this beam is not a major concern as this test is purely numerical.

The aim of this test is to study the behavior of the beam, more precisely its maximum deflection, regarding the number of degrees of freedom used to model it. For this experiment, each integration point has 8 neighboring frames and the shape function used is a trilinear interpolation like the one used for hexahedral elements. For each simulation, the obtained maximum deflection of the end of the beam is compared to the analytical solution to the *cantilever beam* problem (uniformly loaded beam). The solver used here is a *conjugate gradient* with a precision set at 10^{-10}. The results of these tests are presented on figure 4.12. Figure 4.12(a) shows the difference between the simulation results



(a)



(b)

Figure 4.12: Simulated maximum deflection in function of DoF count and error with the cantilever beam analytical solution.

and the analytical solution, as well as the error percentage when the beam's cross section resolution varies and its length resolution is fixed. In this case, 25 frames are used to discretize the beam's length. Similarly, figure 4.12(b) shows the same results, this time when the length resolution of the beam varies with a cross section resolution fixed at 5×5 frames.

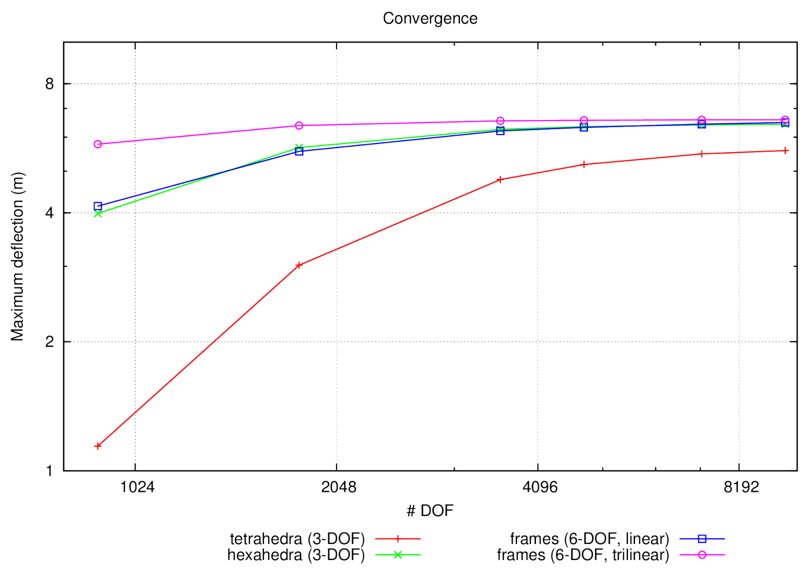


Figure 4.13: 6-DoF method convergence compared to corotational FEM.

In the second test, highly deformable beams have been voluntarily chosen in order to study the convergence of the method in the case of large deformations. It consists in the comparison of 2 beams simulated using the 6-DoF model and 2 beams simulated using FEM with corotational formulation included in the SOFA framework. Each of these 4 beams is only subject to their own weight and their degrees of freedom are arranged identically. The first beam simulated using the 6-DoF model uses 4 neighboring frames per sample and linear interpolation whereas the second uses 8 neighboring frames per sample and a trilinear interpolation. In this case, the 6-DoF model is used with a mesh-based approach. As for the FEM beams, they are discretized using tetrahedra for the first one and hexahedra for the second one. Their dimensions are $16 \times 1 \times 1$ meters, with a weight of 5 kg and their Poisson's ratio and Young's modulus are respectively 0.3 and 40 kPa.

The results of this study are presented on figure 4.13. An asymptotic limit may be observed, demonstrating the convergence of the method. One may also observe that the frame-based method requires much fewer degrees of freedom than corotational FEM, whether using tetrahedra or hexahedra, to converge toward the same solution. An artificial rigidity induced by a reduced number of DoF may also be noticed.

Finally, the last verification experiment is inspired from the "*Validation des Progiciels de Calcul de Structures*" (VPCS) guide published by the AFNOR [AFN90]. This test aims at assessing the simulated beam's behavior in traction. The beam

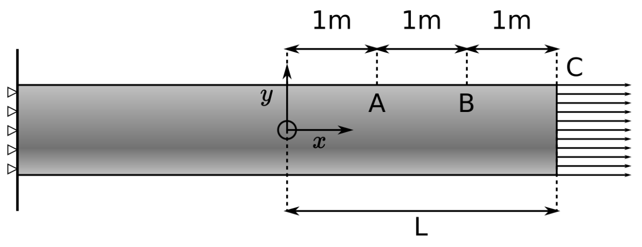


Figure 4.14: VPCS traction test.

Test	$u_y(A)$	$u_y(B)$	$u_y(C)$
VPCS reference	$0,015$	$0,015$	$0,015$
Simulation	$0,0151$	$0,0151$	$0,0151$

Table 4.2: Results of the VPCS test.

used for this test is 6 meters long and has circular section with a diameter of 1 meter. The test depicted on figure 4.14 consists in performing a traction directed along the cylinder to obtain a 3 mm elongation on the x axis. Then the displacement of points A, B and C on the y axis is measured. To reproduce this test, one face of the cylinder is anchored while a displacement is imposed on the other. The motion of the displaced face's frames is only constrained on the x axis. The beam is discretized using frames evenly spaced. Each integration point has 8 neighboring frames and the shape functions used are trilinear interpolations. The results of this test are presented in table 4.2. The difference between the results obtained by simulation and the theoretical reference is below 0.7%.

Connecting anatomical structures

In this section, we present several tests done on the coupling between anatomical structures. The first tests concerns the coupling of thin walled organs. To do so, the coupling is done using the DoF sharing approach discussed in section 4.2. The figure 4.15 shows the application of our method for simultaneous coupling of multiple organs. The 3 surfaces are modeled using shell elements. The middle one is linked to both upper and lower connective tissues. In this simulation, the frames of the front and rear edges of the topmost surface are fixed, all the other frames are able to move freely. The coupling is eased thanks to the use of meshless shape functions – here Shepard's functions – that "automatically" connect frames of all surfaces and nearby connective tissues.

When using a meshless shape function, the method makes the coupling of

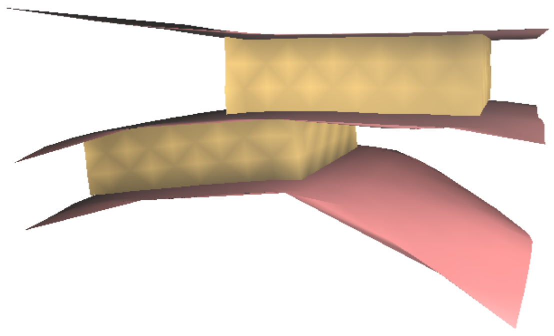


Figure 4.15: Coupling between multiple objects.

objects of different resolutions easier, thanks to the unstructured property of meshless shape functions. Indeed, the number of a sample's neighboring frames, in contrast with a mesh based method, is not limited and is generally influenced by the radius of influence of the shape function. Moreover, a meshless scheme is interesting in the case of connective tissues with thin parts in that it avoids meshing.

Simulations of mechanical coupling have also been carried out using more complex models. Figure (4.16) shows the result of a simulation in which a uterus is linked to a bladder using connective tissues. The top of the uterus surface is fixed. The bladder and the uterus are simulated using shell elements. The connective tissues are discretized with 22 frames and 827 samples.

The other test consists in the simulation of two volume organs linked by connective tissues. In this case, the coupling is simulated using the method detailed in section 4.2. Here, the boundary FE nodes at the interface are mapped to the boundary frames of the connective tissues. Both organs are discretized with a total number of 3114 tetrahedra. As for the connective tissues, they are discretized using 83 frames and 309 samples. The figure 4.17 shows two volume organs linked by an irregular layer of soft tissues. This latter exhibits both thin and volume parts.

4.4 Implementation

In this section details about the implementation of the 6-DoF method are provided as well as the performance of the CPU implementation. Then, a short introduction on GPU computing is given and the approach followed for the GPU implementation is described. Finally, timings of both implementations are compared.

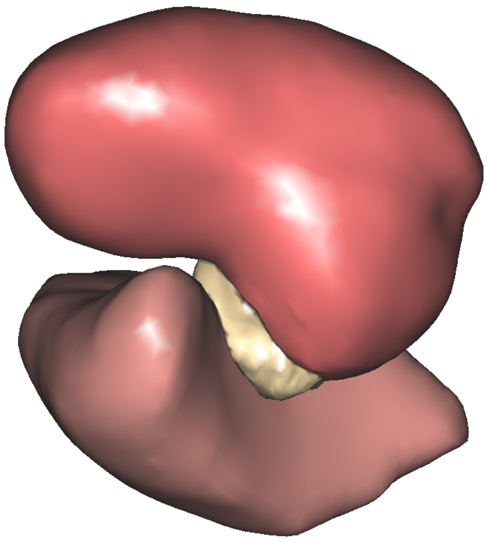


Figure 4.16: Simulation of the pelvic system. The uterus (top) is linked to the bladder (bottom) by a layer of fasciae.

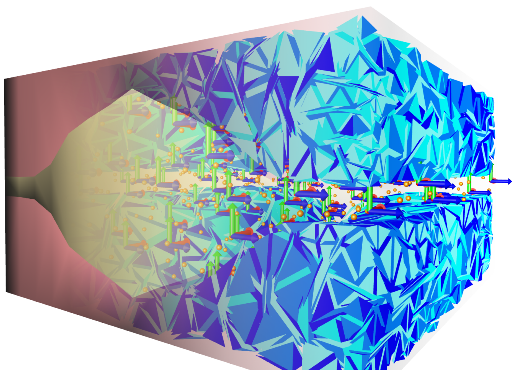


Figure 4.17: Simulation of two volume organs linked by an irregular layer of connective tissues.

Performances

The 6-DoF method has been implemented in the SOFA framework[2]. The computation of internal forces and their derivative with respect to the degrees of freedom given by algorithms 1 and 2.

The performance of the CPU implementation was measured with iterative

[2]www.sofa-framework.org

Algorithm 1 Frames internal forces computations

```
for all sample $s$ do
    for all neighboring frames $i$ do
        Compute deformation gradient $\mathbf{F}^s$ using Eq. (3.40)
        Compute deformation gradient linearization $\mathbf{B}^{is}$ using Eq. (3.42)
    end for
end for

for all sample $s$ do
    for all neighboring frames $i$ do
        Compute strain linearization $\mathbf{M}^{is}$ using Eq. (3.49)
    end for
end for

for all sample $s$ do
    Compute Green-Lagrange strain tensor $\mathbf{L}^{\mathrm{s}}$ using Eq. (3.54)
    Compute PK2 tensor $\mathbf{S}^{\mathrm{s}}$ using Eq. (3.53)
    for all neighboring frames $i$ do
        $\mathbf{f}_i^{\mathrm{int}} \leftarrow \mathbf{f}_i^{\mathrm{int}} - \left[\mathbf{M}^{is}\right]^\top \mathbf{S}^s v_s$
    end for
end for
```

Algorithm 2 Frame internal force derivative computations

```
Input: $\forall$ frames $i$ and samples $s$ $\mathbf{F}^s$, $\mathbf{B}^{is}$, $\mathbf{M}^{is}$
for all sample $s$ do
    for all neighboring frame $i$ do
        Compute $\delta\mathbf{S}^s$ using Eq. (3.68)
        Compute $\delta\mathbf{B}^{is}$ using Eq. (3.75)
    end for
end for

for all sample $s$ do
    for all neighboring frame $i$ do
        Compute $\delta\mathbf{f}_i^{\mathrm{int}}$ using Eqs. (3.67) and (3.71)
    end for
end for
```

Test	***# Frames***	***samples***	***neighbors***	***Time (ms).***
Beam	72	400	4	32,65
			8	45,42
			6	55,15
			10	69,56
		500	4	39,18
		864		43,42
		1470		114,3
		2340		160,53
	144	750		54,49
	288			56,64
	500			58,5
	792			69,12
Coupling	160	400	4	53,22

Table 4.3: CPU performances using iterative solver (conjugate gradient).

and direct solvers. The results presented in table 4.3 were observed for simulations using an implicit time integration and a conjugate gradient solver. This latter was running 50 iterations per time-step. The platform used for this test was equipped with an 2.67 Ghz Intel Xeon W3520 with 8 Gb RAM. The following parameters have been observed: the number of frames, the number of samples, as well as the number of neighboring frames per sample. It highlights that samples and frames counts are the parameters that affect the performance the most. More generally, the average complexity of algorithms 1 and 2 is $\mathcal{O}(s\tilde{n})$, with being s the number of samples and $\tilde{n}$ being the average number of neighboring frames per sample. These results also show that the method rapidly becomes unusable for real-time applications when the number of samples increase.

The second test concerns simulations using direct solvers, here a LDL$^\top$ solver optimized for sparse matrices. The table 4.4 presents the performance obtained on a computer equipped with a 3.4 Ghz Intel Core i7 with 8 Gb RAM. For this test, the 6-DoF method was used in a mesh-based scheme using tetrahedral interpolation (4 neighboring frames per sample). The following parameters have been recorded: the number of frames and samples, the time elapsed for the newton iteration (see section 3.6, Eq. (3.63)). It also contains the time elapsed for building and solving the system matrix. Rendering time is not taken into account.

Frames	*Samples*	*Newton it.*	*Sys. build*	*Sys. solving*	*Total*
48	648	3.53	22.41	1.42	27.36
216	3000	16.49	143.24	17.77	177.5
512	8232	46.24	600.91	105.99	753.144
1728	31944	183.64	5654.16	1572.54	7410.34
2744	52768	1036.46	99723.8	4926.15	105686.41

Table 4.4: Timings (in ms) for CPU implementation using direct solver.

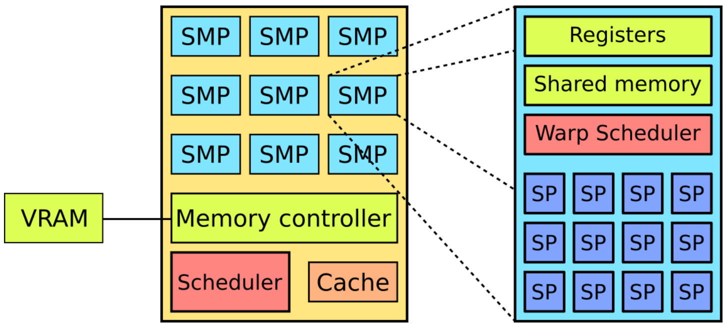


Figure 4.18: Schematic view of GPUs architecture.

GPU computing principles

Increasingly used over the past decade, GPU computing results in interesting performance gains through the massively parallel implementation enabled by the hundreds of compute units available on these devices. GPU computing not only shares the constraints of more traditional parallel computing (*e.g.* synchronization or concurrent writes problems) but also faces specific constraints related to the GPU architecture. In the following part, the particularities of this computing model, and key aspects of GPU implementations are briefly discussed.

GPU architecture

The key to GPUs massive parallelism resides in the hundreds of *stream processors* (ALUs) they embed. In fact, these stream processors are grouped into a stream multiprocessor (SMP) which also includes instruction fetching and scheduling units as well as a small amount of memory. These SMP are then laid on the GPU's chip which contains a SMP scheduling unit and memory controller. A simplified GPU architecture diagram is shown on figure 4.18.

Execution model

In *Flynn's taxonomy*, the GPU belongs to a class of architecture known as *Single Instruction Multiple Data* (SIMD). In this class all the computational units of the chip follow the same instruction stream. In other words, at a given time, all the units execute the same instruction, but with different data. They are said to run in *lockstep mode*. A function that is executed on GPU is called a *kernel*.

Although GPUs rely on SIMD, conditional instructions are still available but branch divergence should be avoided. Indeed, when a conditional statement is reached, divergent execution paths are serialized, dividing performances by factors. For instance, if threads reach an "if/then/else" statement, threads satisfying the "if" clause will execute the "then" instruction block first and. Only when the execution of the latter is completed, the other threads will run the "else" instruction block. When this is done, all the threads converge to the instruction right after the conditional statement and start again in lockstep mode.

When running a kernel, threads are grouped to form a thread block whose size is defined by the user. The purpose of thread groups is to enable the threads belonging to the same group to cooperate by means of a special memory area. All the thread groups that execute a kernel are restricted to have the same size.

In addition, it is interesting to introduce the notion of *warp*. In GPU computing a warp is the number of threads that effectively run the kernel in lockstep mode on a SMP. Thus, it is strongly advised to choose group sizes that are multiples of the warp size to maximize the efficiency of the execution.

Memory model

GPUs embed different types of memory whose use depends on speed and storage capacity requirements:

Registers are embedded into SMP, which makes them the fastest memory available, but they come in a small amount (32 kb to 128 kb). In addition, registers are private to each thread, which means that a thread cannot access another thread's registers.

Shared memory is also embedded into SMP but unlike registers, shared memory is accessible by all the threads belonging to a same thread group enabling thread cooperation. Its availability is also limited.

Constant memory is a read-only memory for storing data that will not change during kernel execution. It is located outside the chip and is accessible by all threads in any thread group.

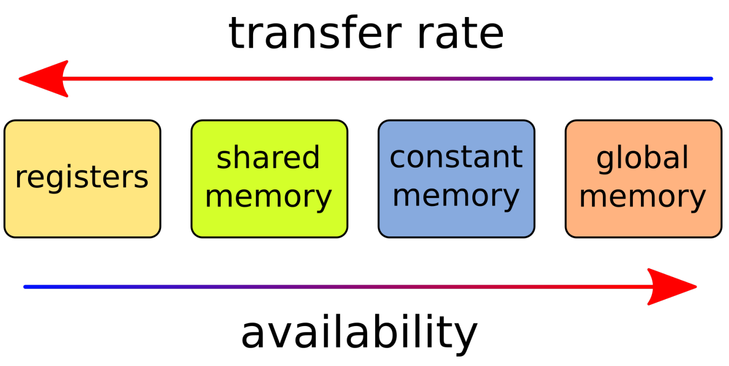


Figure 4.19: Availability and latency of the different memory families.

Global memory also known as VRAM, is the memory that is available in the largest amount. It is accessible by any thread from any group. However, because it is located outside the chip, it is significantly slower than other types of memory.

The availability compared to access times of these different types of memory is summed-up on figure 4.19.

In order to provide efficient implementations great attention must be paid to memory management. As memory transfers between the host and the device global memory (RAM to VRAM) are costly, data must be kept in global memory as much as possible once it has been transferred. In the same way, global memory access from the kernel is much slower than from registers or shared memory. Thus, it is recommended to limit the number of global memory accesses. If the data stored in the global memory has to be accessed multiple times, it is preferable to store it in registers or in shared memory. Finally, to make up for the slow global memory, which access can take up to thousands of cycles, GPU accesses multiple contiguous memory location at the same time. Thus, to perform efficient memory transfers, it is advised to use a technique known as coalesced memory access which principle is to fetch memory segments aligned with 32, 64 or 128 bytes boundaries. If done in such a way, multiple data location can be transferred in one memory transaction.

GPU implementation

The main problem with a massively parallel implementation of this method is that computations have to be done on 2 distinct entities: samples and frames. Indeed, the first steps involve computations of both deformation gradient and the stress/strain tensors, which are related to the samples. Yet, the stress accumulated on a given sample contributes to the force computation of multiple frames making it impossible to parallelize force computation at sample level. Indeed, if several samples added their contribution to the force of same frame, it would lead to concurrent writes conflicts. Using atomic operations is not a viable option since writes operations are serialized, reducing significantly the efficiency of the implementation. Besides, atomic operations on floats are not supported on older hardware. Furthermore, other quantities like deformation gradient derivative, strain tensor derivative, force and force derivative need to be computed for each interacting frame/sample pair. In order to overcome these problems, the chosen approach consists in splitting the computation in 2 distinct passes.

In a first kernel, one thread is assigned to each sample. Each thread sequentially computes deformation gradient, strain/stress tensors and its derivative as well as the sample's contributions to the internal force of its neighboring frames. Since deformation gradients and SVK stress tensors are heavily used throughout the computations, they are stored in registers so as to obtain a better memory access performance. Then for each frame the sample interacts with, a partial force is computed by multiplying the transpose of the $\mathbf{M}$ matrix of Eq. (3.50) with the stress tensor. However, the implementation relies on a matrix-free approach: neither the $\mathbf{B}$ nor the $\mathbf{M}$ matrix is explicitly built in the device global memory (VRAM). Indeed, storing all these matrices would require a substantial amount of memory, and, since global memory access is one of the main bottlenecks in GPU computing, performances would be severely affected. Thus, a matrix-free approach makes it possible to reduce the memory consumption and the memory access overhead. With this goal in mind, the result of the multiplication is evaluated row by row so that only one matrix row must be stored at a time. Thus, due to its small size, this matrix row can be stored in registers. The resulting partial force vector is stored in shared memory and then transferred to a temporary buffer contained in the device's global memory.

Additionally, parallelizing the computation at samples level is advantageous because for a given sample, previously mentioned computations are independent from other samples. This way, synchronization barriers are not needed and the performances are not affected.

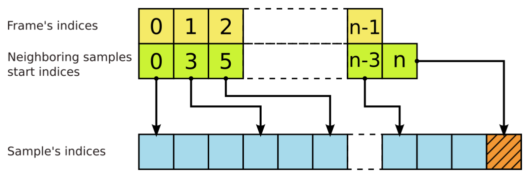


Figure 4.20: Reverse indirection table.

In contrast, the second pass is parallelized at the frame level and 6 threads are

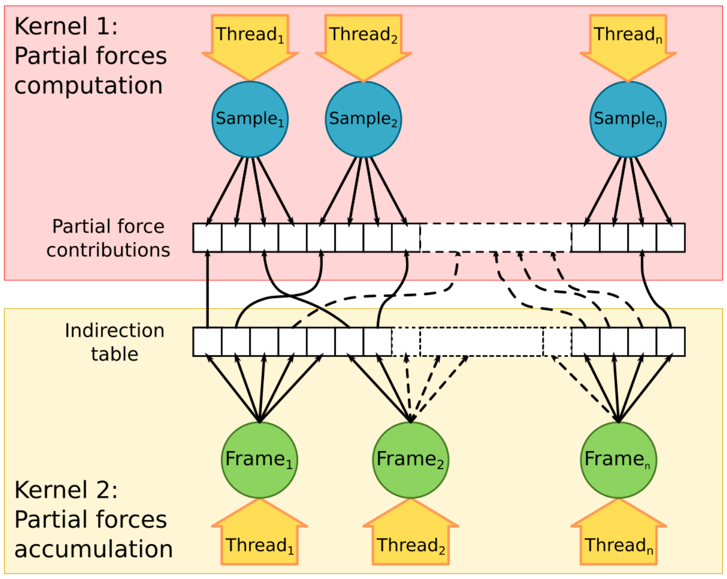


Figure 4.21: GPU implementation of the 6-DoF method.

assigned to each frame. This kernel takes the partial force contribution buffer of the previous kernel as an input. In addition, it also needs the reverse indirection table shown on figure 4.20, which gives connectivity information between frames and samples. More precisely, this table stores the count and the indices of each frame's neighboring sample. Thanks to this structure, each one of the 6 threads assigned to a frame gathers the partial force contributions and stores them into the on-device shared memory. Then, in a last step, these partial contributions are summed using a segmented inclusive scan [SHZO07] to output only one internal force per frame. Threads are run in groups of 96 because it is a multiple of warp size (32 threads), which allows to maximize the number of active threads. By doing so, only the last thread group may contain inactive threads.

This splitting approach maximizes the amount of work done in parallel, especially in the first kernel, since the discretized models contain many more samples than frames. As a result, the occupancy of the GPU is increased, as well as the overall efficiency of the implementation. Moreover, because there are only 2 kernels for internal force computations, the effect of kernel launching overhead is limited. The whole algorithm is described on figure 4.21.

Because of the limited amount of shared memory available on GPUs, the implementation is restricted to use at most 16 neighboring frames per sample. This limitation is however not a major concern since 16 neighbors per samples are enough for most applications. Indeed, when used with a mesh-based scheme,

Samples	***CPU time***	***GPU time***	***Accel.***
800	78,42	15,25	5,14
2600	251,77	17,11	14,72
6070	595,77	23,39	25,47
12700	1228,8	32,29	38,07
16210	1561,8	39,29	39,75
19990	1990,06	43,12	46,15
26750	2675,47	53,09	50,39
34610	3468,54	66,1	52,47

Table 4.5: Comparison between CPU and GPU performance. Timings are in ms.

this method requires either 4 neighbors (for tetrahedral linear interpolation) or 8 neighbors (hexahedral trilinear interpolation). With a mesh-free scheme, this restriction is not really a limiting factor since this method is designed for sparse models.

The GPU implementation was done using CUDA but the approach adopted still holds for other GPU computing APIs such as OpenCL or OpenGL (compute shaders). Performances were evaluated using the platform previously mentioned, equipped with an NVidia GTX 570 GPU with 1270 Mb VRAM.

Performances of the GPU implementation have been measured and compared to the CPU ones. The table 4.5 shows the timings obtained with GPU implementation when using an iterative solver. For this test, a conjugate gradient solver performing exactly 60 iterations per time-step has been used. However, it should be emphasized that CG converge with a number of iterations that depends linearly on the number of degrees of freedom. The 6-DoF method is used along with Shepard's shape function with 4 neighboring frames per samples. In this test, only the number of samples varies, the number of frames being constant. The data gathered in this table are: the number of samples, the time elapsed for a single time step with the CPU and the GPU implementations, and the acceleration factor. The timings are given for a full time step which includes internal/external forces computations and system solving, but not visual rendering.

An interesting property of the GPU implementation that can be seen in table 4.5 is that it scales well with the number of samples. This table shows that the GPU implementation reaches significant acceleration factor allowing for more complex scene to be simulated in real-time.

The second test consist in measuring the performances of the GPU implementation depending on the number of frames. Hence, the results presented in table 4.6 are obtained with 1000 samples. Here again the solver used is a

Frames	*CPU time*	*GPU time*	*Accel.*
27	8.11	0.77	10.53
216	8.25	0.8	10.31
1000	8.32	0.84	9.90
1728	8.81	0.87	10.13
3375	8.89	0.99	8.98
6859	9.44	1.58	5.97
10648	10.3	1.36	7.57
13824	10.70	2.3	4.65

Table 4.6: Comparison between CPU and GPU timings for a constant number of samples (1000). Timings are in ms.

conjugate gradient performing exactly 100 iterations per time-step. The shape function used is also a Shepard's function with 4 neighboring frames per samples. The data observed are the same as previously an visual rendering is not considered.

The results show an interesting acceleration factor with a number of frames up to 3000. Still, an acceleration factor decrease can be observed beyond this number. This is probably owed to the fact that, in this case, there are much more frames than samples. Thus, the addresses of one frame's neighboring samples contained in the reverse table are less contiguous. As a consequence, the second kernel performs much less coalesced memory accesses, negatively impacting the performances.

4.5 Summary

In this chapter, we discussed the use of the 6-DoF mechanical model in both meshless of mesh-based scheme. While the mesh-based approach can be used to simulate connective tissues with volume parts, the meshless approach is usefull to handle much thinner parts.

We also presented the use of this mechanical model for the coupling of anatomical structures. We detailed the different approaches used to link thin walled and volume organs, as well as rigid structures. Thanks to these methods we are able to model complex boundary conditions on a wide range of anatomical structures.

Then we presented the results of the tests performed to assess the accuracy of the method. We have shown that, for the studied cases, the physically-based approach offers a good accuracy. Among these results, the fact that this methods requires less degrees of freedom than FEM is particularly interesting. However, the validation tests proposed in this chapter are still preliminary. Indeed, these results have not been faced with a ground truth, especially in the case of large deformations. To address this issue, the results of the simulation could be compared, for instance, to an object made of silicon rubber which material parameters are known.

The performance tests have shown that the CPU implementation of this methods rapidly becomes unusable for real-time applications. Hence, to alleviate this problem, a GPU implementation has been developed, offering real-time performance for more complex simulations.

Chapter 5

FEM based soft robot guidance

Soft robots are a special class of robots that have been increasingly studied over the past few years. The main characteristic they exhibit is their high compliance owed to the materials they are made of, as, for instance, silicon rubber. In addition, their deformation capacities are also due to the fact that they have no, or a very few rigid parts. In parallel, continuum robots are another type of flexible robots which are designed as a linear structure.

The continuum robots design is generally inspired by biological structures like elephant trunks [NJ07], or octopus tentacles as in [CI10]. Such a kind of robot may be actuated, to name a few techniques, using concentric tubes able to move and rotate relatively to each others. They may also rely on pneumatic actuators, inflating deformable cavities, or tendon (cable) actuators that are pulled to deform the structure.

Their compliance makes them a good fit for a wide range of application: search and rescue, exploration of chaotic/hostile environment, underwater and space operations or surgical interventions. Another advantage of soft robots comes from the fact that they can be operated safely within their environment. Indeed, the risk of hurting humans or damaging delicate objects is greatly reduced. In return, this peculiar kind of robots is generally more difficult to control than their rigid link counterpart.

In this chapter, a physically-based modeling of a particular type of soft robot is detailed. This approach is designed for soft robots constituted by a deformable arm with rigid vertebras. It is based upon a 6-DoF nodes mechanical model. The use of such a type of DoF is important as it enables to control the end effector's orientation which is crucial for precise maneuvers. The proposed method aims at providing an accurate and automatic control of the effector's position. It relies on the simulation for determining the adequate values to be sent to the actuators. To this end, a reduced mechanical model taking the robot's specificity (*i.e.* rigid vertebras) into account has been developed to be used along with inverse simulation techniques.

5.1 Motivation

Soft robots are more difficult to control than their articulated counterpart because, unlike the latter, they do not have discrete parts. Instead, they are made of a deformable continuum and, as a result, they have – theoretically speaking – an infinite number of degrees. Among them, a large part is unactuated and moves only thanks to robot's natural compliance.

A way to tackle this problem is to model the robot and its actuators to predict and control its behavior. This requires a model that relates the external forces applied to the robot and the resultant posture of the continuum backbone. In, [GRWM03], Gravagne *et al.* propose a continuum mechanics based approach to compute the dynamics of flexible robot made of a highly deformable planar backbone. Furthermore, the potential of simulation for soft robot guidance purpose has already been proven by Duriez in [Dur13]. In this work, the author propose to control a cable actuated by deformable robot using an FEM inverse simulation. However, precise control requires an accurate simulation, implying the use of models with a high number of elements. As a result, it is difficult to reach performances that are compatible with real-time applications. In this case, GPU implementation is necessary but still delivers quite low refresh rates. With this approach real-time performances are possible up to 2000 FE nodes. In addition, some robots models with complex shapes need many more FE nodes to provide geometric and mechanical accuracy. Still, fast refresh rates are required to provide the robot with a smooth motion. This highlights the need for a real-time soft robots model to enable their control.

Soft robots deformable arms with rigid vertebra share the use of 6-DoF joints with articulated robots. Indeed, a rigid vertebra can be modeled as a rigid joint able to move and rotate in every direction. Yet, the deformable part between two rigid vertebras still has to be simulated by volumetric elements, raising again the problem of simulation's inadequate performances.

To alleviate this problem, the solution proposed is to build a reduced model of the vertebras by condensing the behavior of vertebra elements into an equivalent nonlinear 6-DoF spring between two frames. This results in an FEM based approach allowing to significantly reduce the number of degrees of freedom of the whole continuum robot's model. However, continuum robots are generally designed so that the cross section diameter varies along the arm to enhance its flexibility. Hence, in the presented approach, the robot's model can be broken down into multiple subparts, and simulated using techniques inherited from structural analysis. As a result, it provides a modular way of modeling soft robots arm while taking non-constant cross section into account.

5.2 Compliance computation

Domain decomposition approach

The developed approach considers a deformable arm that includes rigid parts, the vertebras, as depicted on figure 5.1. It can be observed that such an arm is made of almost the same repeated subpart, with only slight variations of the diameter, since the arm is thinner the closer to the effector. This variation of diameter implies differences in the mechanical behavior of each subpart. To handle these diameter variations, and without loss of generality shape variations of the arm, the adopted method consists in splitting the arm into multiple subparts. Each one is then isolated and its respective mechanical properties are computed. As a result, each subpart is composed of two halves of rigid vertebras (at each end), and an intervertebra section, modeled using FEM with corotational formulation. All the nodes used for the FE modeling of a subpart are 3-DoF nodes.

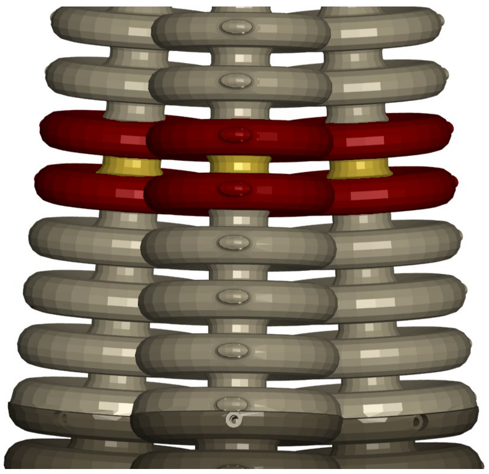


Figure 5.1: Design of the arm. Red and yellow parts respectively highlight vertebras and an intervertebra

Since a vertebra has a fully rigid motion, so do all the nodes of its cross section. Hence, the nodes of both ends of a subpart can be seen a rigid object. It follows that, to model this rigid behavior, kinematics of these nodes can be driven by a frame placed at both ends of the subparts, as shown on figure 5.2

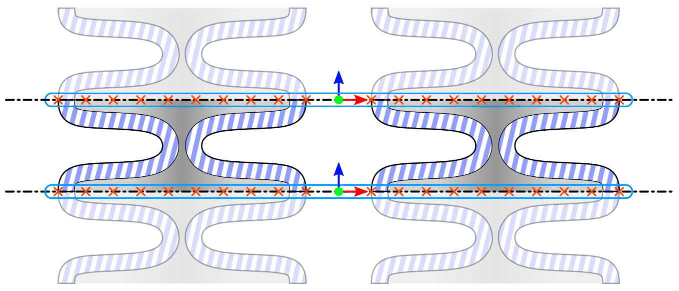


Figure 5.2: Substructure used for the computations. The red crosses indicate the FE nodes that are kinematically linked to the frames.

Finally, each subpart is made of volume elements defined using 3-DoF nodes, and two frames (6-DoF nodes) for modeling the rigid behavior of vertebras. The nodes that are not kinematically linked to frames only behave according to the FE modeling and will be henceforth referred as "independent nodes". To compute subparts mechanical properties, each of them must be represented by a mechanical system where 3-DoF nodes and frames are mixed. As a result, the vector containing the degrees of freedom of the whole subpart can be represented by:

$$\left[\ \overbrace{\mathbf{p}_0\ \mathbf{p}_1\ \cdots\ \mathbf{p}_{n_i-1}}^{\text{independent nodes}}\ \underbrace{\mathbf{q}_0\ \mathbf{q}_1}_{\text{frames}}\ \right]^\top \tag{5.1}$$

where the linked nodes have been replaced by two frames and n_i is the number of independent 3-DoF nodes. In addition, a frame $\mathbf{q}_i = [\mathbf{x}_i\ \mathbf{r}_i]$ is the concatenation of two 3-dimensional vectors representing translational and orientation parts.

Mixed node modeling

In order to perform simulations, a stiffness matrix must be built for each subpart. Since a subpart is modeled using volume elements, its stiffness matrix $\mathbf{K}_v$ is already provided, but frames are not included in it. Thus, to take into account the rigid ends in FE computations, a new stiffness matrix that incorporates the frames' influence must be built.

To this end, the new stiffness matrix must encode the influence of the independent nodes on themselves $\mathbf{K}_{II}$, the linked nodes on themselves $\mathbf{K}_{RR}$, the frames on the independent nodes $\mathbf{K}_{RI}$ and conversely $\mathbf{K}_{IR}$. Merged into a global stiffness matrix, it yields

$$\mathbf{K}^{mix} = \begin{bmatrix} \mathbf{K}_{II} & \mathbf{K}_{RI} \\ \mathbf{K}_{IR} & \mathbf{K}_{RR} \end{bmatrix} \tag{5.2}$$

Since $\mathbf{K}_{II}$ encodes the stiffness that has to be applied only on the independent nodes, it can be expressed as follows:

$$\mathbf{K}_{II} = \mathbf{J}_0^\top \mathbf{K}_v \mathbf{J}_0 \tag{5.3}$$

where

$$\mathbf{J}_0 = \left[\mathbf{I}^{3n_i \times 3n_i} \; \mathbf{0}^{3n_i \times 3n_l}\right]^\top \tag{5.4}$$

is a matrix which role is to "select" only the independent nodes and n_l is the number of linked nodes.

Similarly, $\mathbf{K}_{RR}$ applies the influence of the frames on the linked nodes and is given by:

$$\mathbf{K}_{RR} = \mathbf{J}_r^\top \mathbf{J}_1^\top \mathbf{K}_v \mathbf{J}_1 \mathbf{J}_r \tag{5.5}$$

where $\mathbf{J}_r$, a 3×6 matrix, encodes the rigid link between a frames and the linked 3-DoF nodes. For instance, for a frame $\mathbf{q}_i$ and a node $\mathbf{p}_j$, $\mathbf{J}_r$ is given by:

$$\mathbf{J}_r = \left[\mathbf{I}^{3\times 3} \; (\mathbf{p}_j - \mathbf{x}_i)_\wedge\right] \tag{5.6}$$

and $\mathbf{J}_1$, is analogous to the $\mathbf{J}_0$ matrix, except that its purpose is to "select" only the linked nodes. Hence:

$$\mathbf{J}_1 = \left[\mathbf{0}^{3n_l \times 3n_i} \; \mathbf{I}^{3n_l \times 3n_l}\right]^\top \tag{5.7}$$

In order give additional precision, the ideas behind the $\mathbf{K}_{RR}$ construction are detailed. First, the $\mathbf{J}_r$ matrix maps the frame rigid motion on the linked nodes. Then, the $\mathbf{J}_1$ matrix isolate the latter on which the stiffness $\mathbf{K}_v$ is applied yielding a force term. In the next step, the force component on linked nodes is isolated again thanks to $\mathbf{J}_1^\top$ which is finally mapped back to a rigid motion thanks to $\mathbf{J}_r^\top$. A similar scheme is involved in the construction of the other blocks of $\mathbf{K}^{mix}$.

As for the cross terms, which describe the influence of the linked nodes on the independent ones and conversely, they can be expressed as:

$$\mathbf{K}_{RI} = \mathbf{J}_0^\top \mathbf{K}_v \mathbf{J}_1 \mathbf{J}_r \tag{5.8}$$

and

$$\mathbf{K}_{IR} = \mathbf{J}_r^\top \mathbf{J}_1^\top \mathbf{K}_v \mathbf{J}_0 \tag{5.9}$$

Once the $\mathbf{K}^{mix}$ matrix is computed, it can be factorized and inverted to obtain the compliance for both FE nodes and frames. However, the goal here, is to condensate the behavior of the FE nodes on the frame to build a reduced model. Hence, it requires an additional step described in the following section.

Reduced model

Compliance computation

Having the $\mathbf{K}^{mix}$ matrix of Eq. (5.2), the next step for building the reduced model consists in the extraction of the frame related part of the compliance matrix. This compliance matrix is needed to build an equivalent stiffness on the frames. To do so, the compliance matrix is computed using a constraint based approach: the first frame is fixed while a unit force and torque is applied in each direction on the second frame. This results in 6 constraints on the second frame (one constraint per DoF). By doing so, one obtains a deformation measure of the subpart under a unit load in each direction, which is equivalent to the compliance matrix:

$$\mathbf{C} = \mathbf{J}\mathbf{K}^{mix\ -1}\mathbf{J}^{\top} \tag{5.10}$$

where

$$\mathbf{J} = \begin{bmatrix}\mathbf{0}^{6\times n_i} \; \mathbf{I}^{6\times 6}\end{bmatrix} \tag{5.11}$$

note that the 6×6 identity matrix of Eq. (5.11) corresponds to the constraints on each DoF of the second frame.

By doing so, the model is condensed on the rigid DoF (frames). This yields the compliance between two vertebras, which is equivalent to the one of the intervertebra FE model.

Equivalent stiffness and force computation

The $\mathbf{C}$ compliance matrix enables for both frames force and equivalent stiffness matrix computations. Let $\mathbf{q}_0 = [\mathbf{x}_0\,\mathbf{r}_0]$ and $\mathbf{q}_1 = [\mathbf{x}_1\,\mathbf{r}_1]$ the lower and upper frames with their respective position and orientation (represented by quaternions in practice). In order to compute a reaction force, the displacements must be evaluated. The latter are expressed in the $\mathbf{q}_0$ coordinate system as shown on figure 5.3.

The force acting on $\mathbf{q}_1$ in $\mathbf{q}_0$ local coordinates, noted $\mathbf{f}_{1/0}$, can be computed using the compliance matrix:

$$\mathbf{f}_{1/0} = \mathbf{C}_{0,1}^{-1}\,\partial\mathbf{q}_{1/0} \tag{5.12}$$

where $\partial\mathbf{q}_{1/0}$ is the $\mathbf{q}_1$ displacement in the $\mathbf{q}_0$ coordinates system and $\mathbf{C}_{0,1}$ is the compliance between frames 0 and 1. Then, using $\mathbf{f}_{1/0}$, force and torque can be applied on $\mathbf{q}_0$:

$$\mathbf{f}_{0/0} = \mathbf{H}_{0,1}\,\mathbf{f}_{1/0} \tag{5.13}$$

with

$$\mathbf{H}_{0,1} = -\begin{bmatrix} \mathbf{I}^{3\times 3} & \mathbf{0}^{3\times 3} \\ \left[\bar{\mathbf{R}}_0^{\top}(\bar{\mathbf{x}}_1 - \bar{\mathbf{x}}_0)\right]_{\wedge} & \mathbf{I}^{3\times 3} \end{bmatrix} \tag{5.14}$$

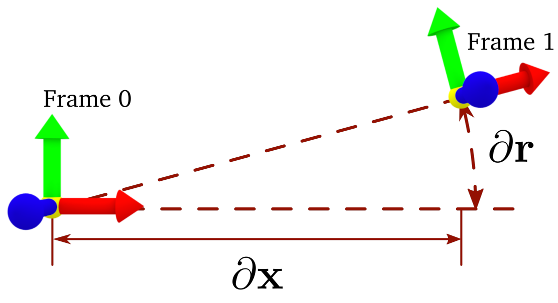


Figure 5.3: Evaluation of relative displacements between frames.

being responsible for the enforcement of momentum conservation. Here, the overbar notation describe a quantity in the rest configuration and $\mathbf{v}_\wedge$ is a skew-symmetric matrix obtained from the $\mathbf{v}$ vector.

However, Eqs. (5.12) and (5.14) give the internal forces on both rigid vertebras in the intervertebra local coordinates. Hence, to obtain the internal forces in global coordinates, local forces must be rotated:

$$\mathbf{f}_0 = \mathbf{R}_0\,\mathbf{f}_{0/0},\ \mathbf{f}_1 = \mathbf{R}_0\,\mathbf{f}_{1/0} \tag{5.15}$$

with $\mathbf{R}_i$ being the rotation matrix of the frame i. The equivalent stiffness matrix can be computed alike. For instance, given two consecutive frames i and j the equivalent stiffness matrix can be expressed as :

$$\mathbf{K}^{eq}_{i,j} = \mathbf{\Omega}_i \mathbf{\Gamma}_{i,j} \mathbf{\Omega}_i^\top \tag{5.16}$$

with

$$\mathbf{\Gamma}_{i,j} = \begin{bmatrix} \mathbf{H}_{i,j}\,\mathbf{C}^{-1}_{i,j}\,\mathbf{H}^\top_{i,j} & \mathbf{H}_{i,j}\,\mathbf{C}^{-1}_{i,j} \\ \mathbf{C}^{-1}_{i,j}\,\mathbf{H}^\top_{i,j} & \mathbf{C}^{-1}_{i,j} \end{bmatrix} \tag{5.17}$$

and

$$\mathbf{\Omega}_i = \begin{bmatrix} \mathbf{R}_i & \mathbf{0}^{3\times 3} \\ \mathbf{0}^{3\times 3} & \mathbf{R}_i \end{bmatrix} \tag{5.18}$$

One can note that the $\mathbf{\Gamma}_{i,j}$ matrix is held constant during the whole simulation Indeed, the compliance matrix $\mathbf{C}_{i,j}$ is precomputed off-line, and the $\mathbf{H}_{i,j}$ is computed in the rest position, which is also constant. Hence, the elasticity model between two successive vertebras is linear.

Such an approach is interesting since it enables fast computations. Indeed, there is no need for recomputing the whole mechanical model at each time-step.

The compliance matrix is computed off-line and only requires at most a few dozen of seconds for detailed FE meshes of the intervertebras. In addition, by integrating the rotation of the first vertebra into the equivalent stiffness matrix, the model is aware of the geometrical nonlinearities owed to large rotations.

On the other hand, such a method loose its accuracy if large deformations occur between two intervertebras because of the linear model.

In order to validate the reduced model, some comparisons have been made with FEM simulations using corotational formulation. Figure 5.4 shows the comparison between a FE cylinder and its frame-based reduced model counterpart. The difference between the deflections of the two cylinders is inferior to 0.72%.

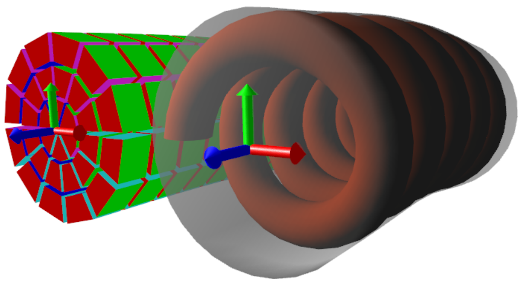


Figure 5.4: Comparison between a FE cylinder (left) and the reduced model using equivalent stiffness (right).

Modeling the robot

Using this approach, the FE model of a whole robotic arm can be replaced by a linear structure similar to a chain made of frames linked by nonlinear springs with an equivalent behavior. As a consequence, each frame, except the first and the last one are linked to exactly two adjacent frames. This underline an interesting property of the reduced model: thanks to this topology, the arm's global stiffness matrix is *Block Tridiagonal* (BTD) which enables the use of optimized resolution techniques [CB80].

Observing Eq. (5.16), one may note that each equivalent stiffness matrix between two consecutive frames can be computed using different compliance matrices. Thanks to this property, this method allows to reproduce a robotic arm made of different kind of subparts, as long as their compliance matrix is provided. The use of different compliance matrices for each subpart does not affect the performances since they are computed off-line from their FE model. However, they still have to be rotated, as shown in Eqs. (5.15) and (5.16).

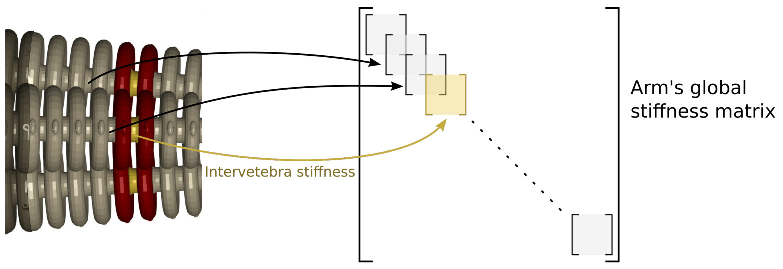


Figure 5.5: Domain decomposition strategy and global stiffness matrix assembly. The reduced model results in a Block Tridiagonal matrix.

5.3 Inverse simulation

The aim of this work is to control a deformable robot in order to make the effector reach a desired position. The reduced model detailed in the previous section enables the computation of the reaction forces implied by the deformations of the deformable arm. However, the goal here is to find the force that has to be applied by actuators to induce the desired displacement of the end effector. To do so, inverse simulation is used along with the reduced model to determine the adequate effort of the actuator so that the effector reaches his goal position.

In order to perform an inverse simulation, *Lagrange multipliers* are used. Indeed, since the actuator are able to apply a effort in a given direction, this can be seen as a constraint applied on some DoFs to make them move in a given direction.

Considering that the robot's moves at low velocities, a quasi-static approach (see section 3.6) can be used to solve the problem. Thus, at each time step, the mechanical system must reach equilibrium:

$$-\mathbf{K}(\mathbf{q}_{i-1})d\mathbf{q} = \mathbf{b} + \mathbf{f}(\mathbf{q}_{i-1}) + \mathbf{J}^\top \boldsymbol{\lambda} \tag{5.19}$$

In order to model a pressure actuated deformable arm, some cavities are placed between the vertebras as shown on figure 5.6. These can be inflated by change of pressure. The domain decomposition approach allows to account, at the frame level, for the internal forces created by the deformation of the cavities. However, the pressure exerted on the cavities will create additional forces on each node i:

$$\mathbf{f}_i^{\text{pressure}} = \frac{\sum_j A_j \mathbf{n}_j}{3} \tag{5.20}$$

where A_j and $\mathbf{n}_j$ are the surface and the normal of an triangle j adjacent to the node i.

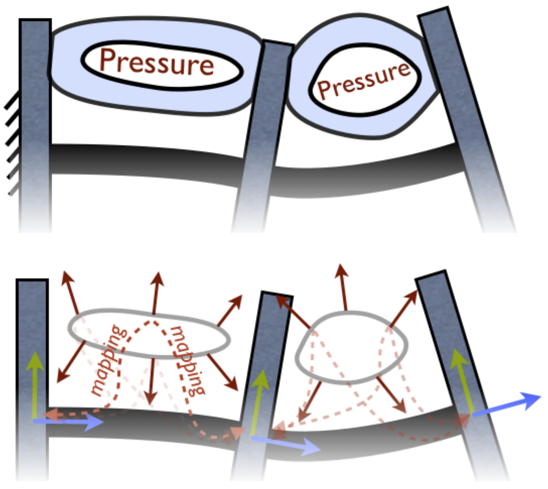


Figure 5.6: Pressure actuator. The constraint Jacobian is build using the forces on the points of the cavity created by unity of pressure ; these forces are then mapped back to the frame using skinning mapping.

By doing so, one can have a weighted direction of force that can be mapped at the frame level using the transposed *Jacobian* of a skinning mapping [WP02]. A mapping is a function $\mathcal{M}(\cdot)$ that gives the positions of a set of node from the positions of another one as described in section 4.2. Lets consider a vector of DoF $\mathbf{x}$ mapped on another set of DoF $\mathbf{q}$:

$$\mathbf{x} = \mathcal{M}(\mathbf{q}) \tag{5.21}$$

In order to express the displacements $\delta\mathbf{x}$ of the mapped nodes, one can use the following relation:

$$\delta\mathbf{x} = \frac{\partial\mathcal{M}(\mathbf{q})}{\partial\mathbf{q}}\delta\mathbf{q} \tag{5.22}$$

with $\frac{\partial\mathcal{M}(\mathbf{q})}{\partial\mathbf{q}}$ being the Jacobian matrix of the mapping function $\mathcal{M}(\cdot)$

In the resulting Jacobian $\mathbf{J}^T$, each column contains the weighted directions of the forces exerted on each frame by a unit pressure in a cavity. In that case λ_a represents the unknown pressure in the cavity, that have to be found by control.

Additional constraints are defined at the tip of the effector, using a Jacobian matrix $\mathbf{J}_e$ that consists in a matrix filled with 0 values except on the rows that corresponds to the last frame of the robot model (the end effector), which is filled with an identity matrix. The goal of this Jacobian is to obtain the mechanical coupling between effectors and actuators for the inverse model optimization.

At this point the method described in [Dur13] is used to compute the inverse model of the robot: how much effort needs to be applied on each actuator so that it creates a deformation that corresponds to the desired motion of the soft-robot's end effector. First, internal forces exerted on the frames are linearized at each time-step i of the computation:

$$\mathbf{f}(\mathbf{q_i}) \approx \mathbf{f}(\mathbf{q_{i-1}}) + \mathbf{K}(\mathbf{q_{i-1}})d\mathbf{q} \tag{5.23}$$

where $\mathbf{K}(\mathbf{q_{i-1}})$ is a Block Tridiagonal matrix, as described above.

Then, we project the mechanics into the constraint space using the Shur complement of the constraint problem:

$$\boldsymbol{\delta}_e = \underbrace{\left[\mathbf{J}_e\mathbf{K}^{-1}\mathbf{J}_a^T\right]}_{\mathbf{W}_{ea}} \boldsymbol{\lambda_a} + \boldsymbol{\delta}_e^{\text{free}} \tag{5.24}$$

$$\boldsymbol{\delta}_a = \underbrace{\left[\mathbf{J}_a\mathbf{K}^{-1}\mathbf{J}_a^T\right]}_{\mathbf{W}_{aa}} \boldsymbol{\lambda_a} + \boldsymbol{\delta}_a^{\text{free}} \tag{5.25}$$

Where $\boldsymbol{\delta}_e^{\text{free}}$ and $\boldsymbol{\delta}_a^{\text{free}}$ are, respectively, the gap between the effector position and the desired position p_{des}, and the displacement of the actuators when the effort of the actuator vanishes $\lambda_a = 0$. Matrices $\mathbf{W}_{ea}$ and $\mathbf{W}_{aa}$ are homogeneous to a compliance. Using $\mathbf{W}_{ea}$, we can get a measure of the mechanical coupling between effector and actuator, and with $\mathbf{W}_{aa}$, the coupling between actuators.

Obtaining these coupling matrices $\mathbf{W}_{ea}$ and $\mathbf{W}_{aa}$ is computationally expensive because it involves the inverse of the matrix $\mathbf{K}$ issued from the FEM model. However, in the particular case of this work, matrix $\mathbf{K}$ is condensed on the frame nodes that are sampled along the axis of the robot and this matrix is block tridiagonal. The complexity of the factorization of such matrix is linear with respect to the number of frames. The computation is particularly fast, even for a large number of nodes and actuators, making the scalability of the whole approach more appealing than what was proposed in [Dur13].

The problem now amounts to minimizing the $\boldsymbol{\delta}_e$ norm that measures the gap between the end effector and the goal position. Thus it is equivalent to solving a *quadratic minimization problem* (QP):

$$min\left(\frac{1}{2}\boldsymbol{\lambda_a}^\top\mathbf{W}_{ea}^\top\mathbf{W}_{ea}\boldsymbol{\lambda_a} + \boldsymbol{\lambda_a}^\top\mathbf{W}_{ea}^\top\boldsymbol{\delta}_e^{\text{free}}\right) \tag{5.26}$$

subject to the constraints defined by each actuator (limited pressure values for pneumatic actuators).

When multiple solutions are possible (which occurs when the number of actuator is greater than the number of DoF of the effector), the matrix $\mathbf{W}_{ea}^T\mathbf{W}_{ea}$ is positive but only semi-definite, which limits the kind of minimization algorithm that can be used. In this case, some QP algorithms are able to converge toward one solution among all the possible solutions. Still, it is possible to introduce a new minimization criterion based on deformation energy. This deformation energy is related to the work of the force applied by the actuators an is given by:

$$E_{\text{def}} = \boldsymbol{\lambda}_a^\top(\boldsymbol{\delta}_a - \boldsymbol{\delta}_a^{\text{free}}) = \boldsymbol{\lambda}_a^\top \mathbf{W}_{aa}\boldsymbol{\lambda}_a \tag{5.27}$$

with

$$\mathbf{W}_{aa} = \mathbf{J}_a\mathbf{K}^{-1}\mathbf{J}_a^\top \tag{5.28}$$

However, $\mathbf{W}_{aa}$ is definite positive if $\mathbf{J}_a$ is not rank deficient, in other words, if the actuators affects different nodes or have different directions. As a result, this energy can be added to the minimization process by replacing Eq. (5.26) by:

$$\min\left(\frac{1}{2}\boldsymbol{\lambda}_a^\top\left(\mathbf{W}_{ea}^\top\mathbf{W}_{ea} + \varepsilon\mathbf{W}_{aa}\right)\boldsymbol{\lambda}_a + \boldsymbol{\lambda_a}^\top\mathbf{W}_{ea}^\top\boldsymbol{\delta}_e^{\text{free}}\right) \tag{5.29}$$

Then by choosing ε small enough, the QP matrix becomes definite positive and the solver is able to find a unique solution that minimizes the deformation of the arm.

5.4 Results

The robot used for the tests is the Compact Bionic Handling Assistant (CBHA), presented on figure 5.7, which is inspired by the form and the functionality of an elephant trunk and manufactured by Festo Robotics. The CBHA is part of the didactic platform named RobotinoXT and is composed by an omni-directional mobile platform and the CBHA itself. The latter is composed by two serially connected sections of pneumatic artificial muscles: lower and upper section; a rotative wrist and a soft gripper. Each section is formed by three muscles arranged in parallel configuration. If differential (equal) pressures are applied to the muscles, the section can bend (extend) to change the position and orientation of its distal vertebra. The shape of the manipulator is determined by the net change in position and orientation of both sections that can be computed with the aid of string potentiometers attached to the sides of the muscles. Each muscle resembles a vertebral column or backbone, in which 8 vertebras are connected to each other serially by an inter-vertebras sections which are more flexible in comparison to the vertebras. The whole structure is composed by 16 vertebras.

Then, the inverse model of the robot has been tested in open-loop to control the movements of the CBHA: a desired trajectory of the tip of the manipulator

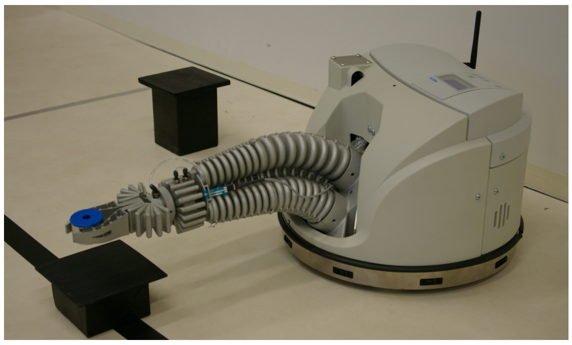


Figure 5.7: The RobotinoXT from Festo Robotics.

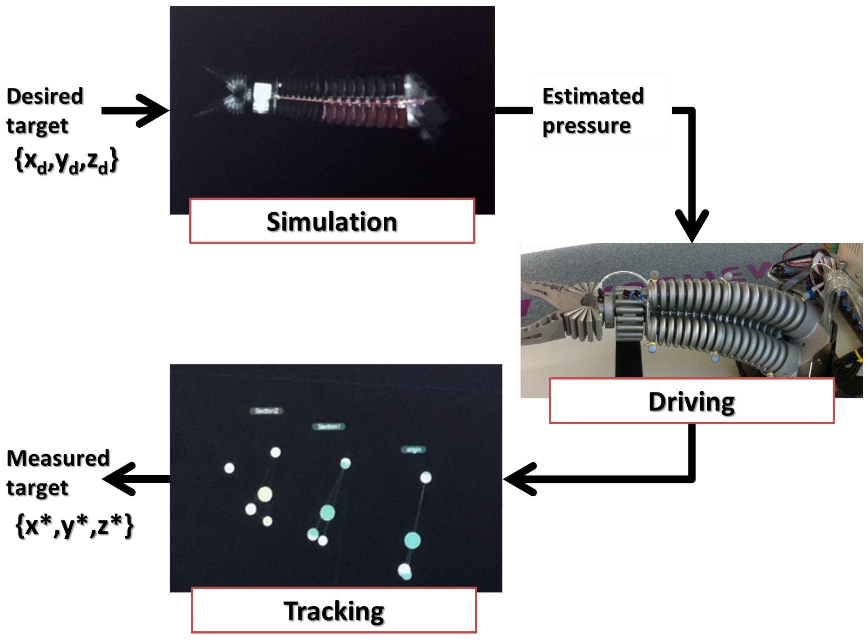
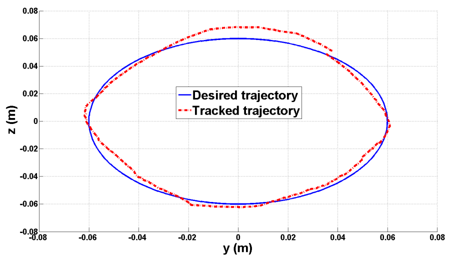


Figure 5.8: Experimental setup used to measure the position of the end effector.

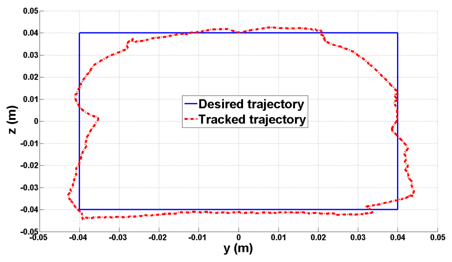
is selected and the simulation computes the pressures needed in each actuator to create the required deformation to reach the target.

A stereo-vision system was used to track the position of the tip of the arm to validate the accuracy of the model. Such setup is showed on figure 5.8. A circle and a square in the y-z plane were used as desired trajectories for the experiment.

Figures 5.9(a) and 5.9(b) show the comparison between the desired and tracked trajectories is presented. In the first trajectory, the circle is selected to have a radius of 60 mm, while a square of side lenght of 80mm is selected for the second



(a)



(b)

Figure 5.9: Results of the performed test on circle and square trajectories. Figure shows the difference between desired and tracked trajectories.

trajectory. The speed at which the square and circle trajectories are performed are 3.1 cm and 2.5 cm per second respectively, which is slow enough so that the dynamic effects can be neglected in the model.

During the experiment, the estimated pressures are sent to the robot at the end of each simulation step. This causes a latency of 54 ms between the model and the CBHA. Moreover, the pressure valves of the CBHA have a response time of 10 ms and the connection between the simulation and the robot has a latency of 3 ms. It is important to note that the results obtained during the experimentation rely only on the inverse model computation.

The experimental results presented on figure 5.9 demonstrate the applicability of reduced model for soft-robot control and guidance.

5.5 Summary

In this chapter we have presented a new approach combining FE models and 6-DoF nodes. The objective of this work is to enable the real-time guidance of a continuum robot made of a deformable arm. With this aim in mind, we developed a reduced 6-DoF mechanical model of a detailed FE model. Thanks to this, the behavior of a complex FE model is transfered onto a nonlinear spring between two frames. Our approach relies on partially precomputed model and inverse simulation to compute the forces to be applied by actuators. In our experiments, we showed that this reduced model provides a good accuracy while keeping real-time performances, even on consumer grade computer.

However, when large deformations occur, the accuracy of this method is decreased. In addition, the approach we developed is based on precomputed compliance matrices which are constant across the whole simulation. As a consequence, an arm made of complex material cannot be simulated for now. Indeed, in this case, this would require the use of hyperelastic materials, and the compliance matrices would be no longer constant.

To alleviate this problem, one could enhance the method by recomputing the compliances matrices at each time step. Since these are computed independently, this could still offer a performance gain. Indeed, inverting n systems of size k is still faster than inverting one large system of size nk.

Chapter 6

Conclusion

Initially geared toward industrial design and engineering, simulation has been used in the medical field with an increasing success over the two past decades. Research work has mainly focused on the simulation of a particular organ such as liver, heart, brain *etc.* Yet, these work are exclusive to one organ and rely on simplified boundary conditions, which affects the accuracy of the simulations. At the same time, realistic simulation of larger regions of the human body, including multiple organs like the abdominal cavity, is still difficult.

These difficulties are mainly due to complex interactions involving contacts but also the connective tissues which play the role of an interface between the organs. However, in this thesis, boundary conditions have been shown to have an non-negligible impact on the simulations' outcome. It has also been established that this influence is important, whatever the cause of the organs deformations is. The sensibility study presented in this thesis has highlighted that a proper modeling of the boundary conditions is a key factor for realistic simulations.

Thereby, this thesis partially addresses the modeling of the connective tissues and the mechanical coupling they provide. The method presented can be used in either mesh-based or mesh-free scheme for an easier object sampling. Connective tissues are simulated using a 6-DoF nodes mechanical model which present a better convergence rate than corotational FEM. This enables for accurate results while using more sparse models.

However there still remain avenues for improvement of the mechanical coupling between the organs and the boundary conditions modeling. First, a proper modeling of boundary condition could benefit from further study of the mechanical properties of the connective tissues. Indeed, these tissues exhibit complex properties. Mechanical experiments could help to determine the most appropriate constitutive law to use for simulating them. Moreover, the connective tissues are not homogeneous: their structure may contain small voids that surely affect their behavior. Still, they are currently modeled as a continuum. Thus, investigating the ways to model these discontinuities could provide interesting results. In addition, a correct modeling of boundary conditions requires to know the location of connective tissues. Hence, prospecting the means to extract them from medical imaging is certainly a key aspect for this. Finally, it should be noted that despite the increasing computational capacity of the hardware, the simulation of the boundary conditions adds supplementary cost to already complex simulations. Indeed, connective tissues simulation must leave enough computational resource for the organs simulation. Thus, with the aim in mind to evolve toward the "virtual patient" model, improving the efficiency of the implementation is essential. This could be done through higher level of parallelization or thanks to reduced models.

A part of this thesis has also been dedicated to soft robotics. Although first seeming unrelated to the connective tissues work, both are based on 6-DoF nodes. This contribution enables the guidance of a deformable arm based on an inverse FEM simulation. Currently, the inverse simulation of detailed FE models can hardly be done in real-time. This hinders smooth guidance of soft robots. To alleviate this problem, a reduced model relying on 6-DoF nodes has been developed. Thanks to domain decomposition and structural analysis approach, the behavior of the arm FE model is condensed on equivalent springs between successive frames. The arm is transformed into a linear structure which results in drastically reduced computation times. As a result, this method provides means for real-time smooth and precise guidance of a soft-robot deformable arm.

Numerical simulation opens new perspectives for soft-robotics. Although it enables for smooth and precise guidance, the method proposed in this thesis can be improved. First, with this approach, the simulation has no feedback on the actual position of the end effector (using sensor for instance). Thus, strictly speaking, this can be considered as an open-loop control method: the guidance is interactive but the operator is responsible for error correction. Hence, this method could be enhanced by adding end effector's position feedback. In addition, this approach relies on small strain theory but with large transformations. Indeed, linear FEM with corotational formulation is used when condensing the arm's behavior. Thus, stiffness matrices of the FE model are constant, enabling to compute them off-line. This is admissible for this particular robot since the arm is relatively rigid. However, this method could not be used for much more compliant materials, like silicon rubber which makes a good fit for building soft-robots. In such a case, the simulations must rely on hyperelastic materials. Yet, these materials are nonlinear implying that the stiffness matrices are function of the displacements. No longer being constant, they must be recomputed at each time-step, and so does the reduced model. Thus, method's computational cost would severely increase. As a consequence, an interesting research perspective would be to adapt the method to hyperelastic materials.

Appendix A

Notation

A.1 Voigt notation

The Voigt notation is a formalism used to represent a symmetric tensor in a compact form by reducing it's order. In the case of strain/stress tensors, it allows to write a 3×3 symmetric matrix as a 6-dimensional vector. However, its use slightly differs depending on the type of tensor it is applied to.

Let's first consider the case of the stress tensor:

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{00} & \sigma_{01} & \sigma_{02} \\ \sigma_{10} & \sigma_{11} & \sigma_{12} \\ \sigma_{20} & \sigma_{21} & \sigma_{22} \end{bmatrix} \rightarrow \begin{Bmatrix} \sigma_{00} \\ \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \\ \sigma_{02} \\ \sigma_{01} \end{Bmatrix} = \{\boldsymbol{\sigma}\} \tag{A.1}$$

Still, in the case of a strain tensor, off-diagonal indices – and more generally components with unequal indices – must be multiplied by 2:

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_{00} & \varepsilon_{01} & \varepsilon_{02} \\ \varepsilon_{10} & \varepsilon_{11} & \varepsilon_{12} \\ \varepsilon_{20} & \varepsilon_{21} & \varepsilon_{22} \end{bmatrix} \rightarrow \begin{Bmatrix} \varepsilon_{00} \\ \varepsilon_{11} \\ \varepsilon_{22} \\ 2\varepsilon_{12} \\ 2\varepsilon_{02} \\ 2\varepsilon_{01} \end{Bmatrix} = \{\boldsymbol{\varepsilon}\} \tag{A.2}$$

A.2 Antisymmetric cross product matrix

The skew-symmetric matrix representing the cross product used in section 3.5 is written:

$$[\mathbf{v}_\wedge] = \begin{bmatrix} 0 & -\mathbf{v}_2 & \mathbf{v}_1 \\ \mathbf{v}_2 & 0 & -\mathbf{v}_0 \\ -\mathbf{v}_1 & -\mathbf{v}_0 & 0 \end{bmatrix} \tag{A.3}$$

Thus, the cross product between two vectors $\mathbf{a}$ and $\mathbf{b}$ can be written as:

$$\mathbf{a} \times \mathbf{b} = [\mathbf{a}_\wedge]\,\mathbf{b} \tag{A.4}$$

Appendix B

Geometric stiffness matrix

This appendix details the computation of geometric stiffness matrix for the 6-DoF mechanical model of section 3.6. The geometric stiffness between two frames i and j can be decomposed in two matrices:

$$\mathbf{K}_{ij}^{\text{geo}} = \mathbf{K}_{ij}^{\text{R}} + \mathbf{K}_{ij}^{F} \tag{B.1}$$

As a matter of fact, the matrix $\mathbf{K}_{ij}^{\text{R}}$ only accounts for frames self angular displacement. Thus, for $i \neq j$, $\mathbf{K}_{ij}^{\text{R}} = \mathbf{0}$. Otherwise, $\mathbf{K}_{ij}^{\text{R}}$ has the following form:

$$\mathbf{K}_{ii}^{\text{R}} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Theta} \end{bmatrix} \tag{B.2}$$

where each line n of $\boldsymbol{\Theta}$, a 3×3 matrix, is given by:

$$\begin{aligned}
[\boldsymbol{\Theta}]^{n,} = \sum_s \; & \left(F_{00}^s \delta\mathbf{B}_{0n}^{\mathbf{r}} + F_{10}^s \delta\mathbf{B}_{1n}^{\mathbf{r}} + F_{20}^s \delta\mathbf{B}_{2n}^{\mathbf{r}}\right) \mathbf{S}_0^s \\
& + \left(F_{01}^s \delta\mathbf{B}_{3n}^{\mathbf{r}} + F_{11}^s \delta\mathbf{B}_{4n}^{\mathbf{r}} + F_{21}^s \delta\mathbf{B}_{5n}^{\mathbf{r}}\right) \mathbf{S}_1^s \\
& + \left(F_{02}^s \delta\mathbf{B}_{6n}^{\mathbf{r}} + F_{12}^s \delta\mathbf{B}_{7n}^{\mathbf{r}} + F_{22}^s \delta\mathbf{B}_{8n}^{\mathbf{r}}\right) \mathbf{S}_2^s \\
& + \left(F_{02}^s \delta\mathbf{B}_{3n}^{\mathbf{r}} + F_{12}^s \delta\mathbf{B}_{4n}^{\mathbf{r}} + F_{22}^s \delta\mathbf{B}_{5n}^{\mathbf{r}} + F_{01}^s \delta\mathbf{B}_{6n}^{\mathbf{r}} + F_{11}^s \delta\mathbf{B}_{7n}^{\mathbf{r}} + F_{21}^s \delta\mathbf{B}_{8n}^{\mathbf{r}}\right) \mathbf{S}_3^s \\
& + \left(F_{02}^s \delta\mathbf{B}_{0n}^{\mathbf{r}} + F_{12}^s \delta\mathbf{B}_{1n}^{\mathbf{r}} + F_{22}^s \delta\mathbf{B}_{2n}^{\mathbf{r}} + F_{00}^s \delta\mathbf{B}_{6n}^{\mathbf{r}} + F_{10}^s \delta\mathbf{B}_{7n}^{\mathbf{r}} + F_{20}^s \delta\mathbf{B}_{8n}^{\mathbf{r}}\right) \mathbf{S}_4^s \\
& + \left(F_{01}^s \delta\mathbf{B}_{0n}^{\mathbf{r}} + F_{11}^s \delta\mathbf{B}_{1n}^{\mathbf{r}} + F_{21}^s \delta\mathbf{B}_{2n}^{\mathbf{r}} + F_{00}^s \delta\mathbf{B}_{3n}^{\mathbf{r}} + F_{10}^s \delta\mathbf{B}_{4n}^{\mathbf{r}} + F_{20}^s \delta\mathbf{B}_{5n}^{\mathbf{r}}\right) \mathbf{S}_5^s
\end{aligned} \tag{B.3}$$

with $\delta\mathbf{B}_{mn}^{\mathbf{r}}$ being the 3-dimensional vector issued from the rotational part of the $\mathbf{B}$ matrix coefficients derivative (see Eq. (3.75)).

As for $\mathbf{K}_{ij}^F$, the second part of the geometric stiffness between two frames i and j, it's expression is given by:

$$\begin{aligned}
\mathbf{K}_{ij}^F = \sum_s & \left(\left[\mathbf{B}^{i,s}\right]^{0,\top} \left[\mathbf{B}^{j,s}\right]^{0,} + \left[\mathbf{B}^{i,s}\right]^{1,\top} \left[\mathbf{B}^{j,s}\right]^{1,} + \left[\mathbf{B}^{i,s}\right]^{2,\top} \left[\mathbf{B}^{j,s}\right]^{2,} \right) S_0^s \\
& + \left(\left[\mathbf{B}^{i,s}\right]^{3,\top} \left[\mathbf{B}^{j,s}\right]^{3,} + \left[\mathbf{B}^{i,s}\right]^{4,\top} \left[\mathbf{B}^{j,s}\right]^{4,} + \left[\mathbf{B}^{i,s}\right]^{5,\top} \left[\mathbf{B}^{j,s}\right]^{5,} \right) S_1^s \\
& + \left(\left[\mathbf{B}^{i,s}\right]^{6,\top} \left[\mathbf{B}^{j,s}\right]^{6,} + \left[\mathbf{B}^{i,s}\right]^{7,\top} \left[\mathbf{B}^{j,s}\right]^{7,} + \left[\mathbf{B}^{i,s}\right]^{8,\top} \left[\mathbf{B}^{j,s}\right]^{8,} \right) S_2^s \\
& + \left(\left[\mathbf{B}^{i,s}\right]^{6,\top} \left[\mathbf{B}^{j,s}\right]^{3,} + \left[\mathbf{B}^{i,s}\right]^{7,\top} \left[\mathbf{B}^{j,s}\right]^{4,} + \left[\mathbf{B}^{i,s}\right]^{8,\top} \left[\mathbf{B}^{j,s}\right]^{5,} \right) S_3^s \\
& + \left(\left[\mathbf{B}^{i,s}\right]^{3,\top} \left[\mathbf{B}^{j,s}\right]^{6,} + \left[\mathbf{B}^{i,s}\right]^{4,\top} \left[\mathbf{B}^{j,s}\right]^{7,} + \left[\mathbf{B}^{i,s}\right]^{5,\top} \left[\mathbf{B}^{j,s}\right]^{8,} \right) S_3^s \\
& + \left(\left[\mathbf{B}^{i,s}\right]^{6,\top} \left[\mathbf{B}^{j,s}\right]^{0,} + \left[\mathbf{B}^{i,s}\right]^{7,\top} \left[\mathbf{B}^{j,s}\right]^{1,} + \left[\mathbf{B}^{i,s}\right]^{8,\top} \left[\mathbf{B}^{j,s}\right]^{2,} \right) S_4^s \\
& + \left(\left[\mathbf{B}^{i,s}\right]^{0,\top} \left[\mathbf{B}^{j,s}\right]^{6,} + \left[\mathbf{B}^{i,s}\right]^{1,\top} \left[\mathbf{B}^{j,s}\right]^{7,} + \left[\mathbf{B}^{i,s}\right]^{2,\top} \left[\mathbf{B}^{j,s}\right]^{8,} \right) S_4^s \\
& + \left(\left[\mathbf{B}^{i,s}\right]^{3,\top} \left[\mathbf{B}^{j,s}\right]^{0,} + \left[\mathbf{B}^{i,s}\right]^{4,\top} \left[\mathbf{B}^{j,s}\right]^{1,} + \left[\mathbf{B}^{i,s}\right]^{5,\top} \left[\mathbf{B}^{j,s}\right]^{2,} \right) S_5^s \\
& + \left(\left[\mathbf{B}^{i,s}\right]^{0,\top} \left[\mathbf{B}^{j,s}\right]^{3,} + \left[\mathbf{B}^{i,s}\right]^{1,\top} \left[\mathbf{B}^{j,s}\right]^{4,} + \left[\mathbf{B}^{i,s}\right]^{2,\top} \left[\mathbf{B}^{j,s}\right]^{5,} \right) S_5^s
\end{aligned} \tag{B.4}$$

where $[\mathbf{B}^{i,s}]^{n,\top}$ represents the transpose of the n^{th} line of the $\mathbf{B}^{i,s}$ matrix.

Bibliography

[ACF11] Jérémie Allard, Hadrien Courtecuisse, and François Faure. Implicit FEM and fluid coupling on GPU for interactive multiphysics simulation. In *ACM SIGGRAPH 2011 Talks*, SIGGRAPH '11, pages 52:1—-52:1, New York, NY, USA, 2011. ACM.

[ACK99] S. N. Atluri, J. Y. Cho, and H.-G. Kim. Analysis of thin beams, using the meshless local Petrov-Galerkin method, with generalized moving least squares interpolations. *Computational Mechanics*, 24(5):334–347, nov 1999.

[AFN90] AFNOR. *Guide de Validation des Progiciels de Calcul de Structures.* Paris-La Défense, 1990.

[AZ98] S. N. Atluri and T. Zhu. A new Meshless Local Petrov-Galerkin (MLPG) approach in computational mechanics. *Computational Mechanics*, 22(2):117–127, aug 1998.

[BBL$^+$15] Julien Bosman, Thor Morales Bieze, Othman Lakhal, Mario Sanz, Rochdi Merzouki, and Christian Duriez. Domain decomposition approach for FEM quasistatic modeling and control of Continuum Robots with Rigid Vertebras. In *International Conference on Robotics and Automation (ICRA)*, 2015.

[BBS00] S.G. Bardenhagen, J.U. Brackbill, and D Sulsky. The material-point method for granular materials. *Computer Methods in Applied Mechanics and Engineering*, 187(3-4):529–541, 2000.

[BD14] Julien Bosman and Christian Duriez. Modèle pour la simulation de tissus connectifs. *Revue Electronique Francophone d'Informatique Graphique, REFIG*, 8:1–13, 2014.

[BDC13] Julien Bosman, Christian Duriez, and Stéphane Cotin. Connective Tissues Simulation on GPU. In *10th Workshop on Virtual Reality Interaction and Physical Simulation*, pages 41–50, Lille, 2013.

[BDDC11] Alexandre Bilger, Jérémie Dequidt, Christian Duriez, and Stéphane Cotin. Biomechanical simulation of electrode migration for deep brain stimulation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6891 LNCS(PART 1):339–346, 2011.

[BGLX00] Ted Belytschko, Yong Guo, Wing Kam Liu, and Shao Ping Xiao. A unifieded stability analysis of meshless particle methods. *International Journal for Numerical Methods in Engineering*, 48(9):1359–1400, 2000.

[BHN$^+$12] J Bano, a Hostettler, S a Nicolau, S Cotin, C Doignon, H S Wu, M H Huang, L Soler, and J Marescaux. Simulation of pneumoperitoneum for laparoscopic surgery planning. *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, 15(Pt 1):91–8, 2012.

[BHPD14] Julien Bosman, Nazim Haouchine, Igor Peterlik, and Christian Duriez. The Role of Ligaments: Patient-Specific or Scenario-Specific?, 2014.

[BIT09] Markus Becker, Markus Ihmsen, and Matthias Teschner. Corotated SPH for deformable solids. In *Proc. Eurographics Workshop on Natural Phenomena*, pages 27–34, 2009.

[BJ05] Jernej Barbič and Doug L. James. Real-Time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Transactions on Graphics*, 24(3):982, 2005.

[BL94] T. Belytschko and YY Lu. Element-free Galerkin methods. *International Journal for Numerical Methods in Engineering*, 37(2):229–256, 1994.

[BLM00] Ted Belytschko, Wing Kam Liu, and Brian Moran. *Nonlinear Finite Elements for Continua and Structures*. Wiley, 2000.

[Blo02] Jules Bloomenthal. Medial-based vertex deformation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 147–151, 2002.

[BLS12] Kenneth Bodin, Claude Lacoursière, and Martin Servin. Constraint fluids. *IEEE transactions on visualization and computer graphics*, 18(3):516–26, mar 2012.

[BMUP01] T. Belytschko, N. Moës, S. Usui, and C. Parimi. Arbitrary discontinuities in finite elements. *International Journal for Numerical Methods in Engineering*, 50(August 2000):993–1013, 2001.

[BNC96] M Bro-Nielsen and S Cotin. Real-time volumetric deformable models for surgery simulation using finite elements and condensation. *In Proc. Eurographics'96*, 15(3):57–66, 1996.

[BPMT11] E Barbieri, N Petrinic, M Meo, and V L Tagarielli. A new weight-function enrichment in meshless methods for multiple cracks in linear elasticity. *International Journal for Numerical Methods in Engineering*, 90(2):177–195, 2011.

[Bri07] R Bridson. Fast Poisson disk sampling in arbitrary dimensions. *ACM SIGGRAPH*, page 2006, 2007.

[BW98] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '98, pages 43–54, New York, NY, USA, 1998. ACM.

[BWHT07] Adam W. Bargteil, Chris Wojtan, Jessica K. Hodgins, and Greg Turk. A finite element method for animating large viscoplastic flow. *ACM Transactions on Graphics*, 26(3):16, 2007.

[BZR07] S Bordas, Goangseup Zi, and Timon Rabczuk. Three-dimensional non-linear fracture mechanics by enriched meshfree methods without asymptotic enrichment. 5:21–36, 2007.

[CA09] Hadrien Courtecuisse and Jérémie Allard. Parallel dense gauss-seidel algorithm on many-core processors. *2009 11th IEEE International Conference on High Performance Computing and Communications, HPCC 2009*, (June):139–147, 2009.

[CADC10] Hadrien Courtecuisse, Jérémie Allard, Christian Duriez, and Stéphane Cotin. Asynchronous Preconditioners for Efficient Solving of Non-linear Deformations. In *Vriphys*, pages 59–68, 2010.

[CAK$^+$14] Hadrien Courtecuisse, Jérémie Allard, Pierre Kerfriden, Stéphane P A Bordas, Stéphane Cotin, and Christian Duriez. Real-time simulation of contact and cutting of heterogeneous soft-tissues. *Medical Image Analysis*, 18(2):394–410, 2014.

[CB80] Samuel Daniel Conte and Carl W De Boor. *Elementary Numerical Analysis: An Algorithmic Approach.* McGraw-Hill Higher Education, 3rd edition, 1980.

[CBP05] Simon Clavet, Philippe Beaudoin, and Pierre Poulin. Particle-based viscoelastic fluid simulation. *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 219—-228, 2005.

[CC09] Eugène Cosserat and François Cosserat. *Théorie des corps déformables.* 1909.

[CCD10] Olivier Comas, Stéphane Cotin, and Christian Duriez. A shell model for real-time simulation of intra-ocular implant deployment. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5958 LNCS:160–170, 2010.

[CDA99] Stéphane Cotin, Hervé Delingette, and Nicholas Ayache. Real-Time Elastic Deformations of Soft Tissues for Surgery Simulation. *IEEE Transactions on Visualization and Computer Graphics*, 5(1):62–73, 1999.

[CI10] Dorian Cojocaru and Mircea Ivanescu. Experiments with Tentacle Robots. In *Robotics (ISR), 2010 41st International Symposium on and 2010 6th German Conference on Robotics (ROBOTIK)*, pages 948–953, 2010.

[CKD13] Hadrien Courtecuisse, Pierre Kerfriden, and Christian Duriez. Real-time simulation of surgical cutting in biological tissues using a semi-implicit time integration scheme. *International Conference on Computational Mechanics*, (March):25–27, 2013.

[CRL$^+$14] Eulalie Coevoet, Nick Reynaert, Eric Lartigau, Luis Schiappacasse, and Christian Duriez. Introducing interactive inverse FEM simulation and its application for adaptive radiotherapy. In *MICCAI - 17th International Conference on Medical Image Computing and Computer-Assisted Intervention*, 2014.

[CTA$^+$08] Olivier Comas, Zeike A. Taylor, Jérémie Allard, Sébastien Ourselin, Stéphane Cotin, and Josh Passenger. Efficient nonlinear FEM for soft tissue modelling and its GPU implementation within the open source framework SOFA. In *Lecture Notes in Computer*

Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), volume 5104 LNCS, pages 28–39, 2008.

[CWK07] Min Gyu Choi, Seung Yong Woo, and Hyeong Seok Ko. Real-time simulation of thin shells. *Computer Graphics Forum*, 26(3):349–354, 2007.

[DB00] S. De and K. J. Bathe. The method of finite spheres. *Computational Mechanics*, 25(4):329–345, apr 2000.

[DC96] Mathieu Desbrun and Marie-Paule Cani. Smoothed Particles: A new paradigm for animating highly deformable bodies. In R Boulic and G Hegron, editors, *Eurographics Workshop on Computer Animation and Simulation (EGCAS)*, pages 61–76. Springer-Verlag, aug 1996.

[DCC$^+$05] M. Doblaré, E. Cueto, B. Calvo, M. a. Martínez, J. M. Garcia, and J. Cegoñino. On the employ of meshless methods in biomechanics. *Computer Methods in Applied Mechanics and Engineering*, 194(6-8):801–821, 2005.

[DCLN06] C Duriez, S Cotin, J Lenoir, and P Neumann. New approaches to catheter navigation for interventional radiology simulation. *Computer aided surgery : official journal of the International Society for Computer Aided Surgery*, 11(6):300–8, nov 2006.

[DCNP04] K Djidjeli, P P Chinchapatnam, P B Nair, and W G Price. Global and compact meshless schemes for the unsteady convection-diffusion equation, 2004.

[DDCB01] Gilles Debunne, Mathieu Desbrun, Marie-Paule Cani, and Alan H. Barr. Dynamic Real-time Deformations using Space & Time Adaptive Sampling. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 31–36, 2001.

[DDCK09] Jeremie Dequidt, Christian Duriez, Stephane Cotin, and Erwan Kerrien. Towards interactive planning of coil embolization in brain aneurysms. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5761 LNCS(PART 1):377–385, 2009.

[DGW11] Christian Dick, Joachim Georgii, and R Westermann. A real-time multigrid finite hexahedra method for elasticity simulation using

CUDA. *Simulation Modelling Practice and Theory*, 19(2):801–816, 2011.

[DPP12] A Deram, Y Payan, and E Promayon. Towards a Generic Framework for Evaluation and Comparison of Soft Tissue Modeling. In *Proceedings of the 19th Medicine Meets Virtual Reality Conference*, pages 116–122, 2012.

[Duf06] Marc Duflot. A meshless method with enriched weight functions for three-dimensional crack propagation. *International Journal for Numerical Methods in Engineering*, 65(12):1970–2006, 2006.

[Dur13] Christian Duriez. Control of elastic soft robots based on real-time finite element method. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3982–3987. IEEE, 2013.

[ESRB11] Hani Eskandari, Septimiu E Salcudean, Robert Rohling, and Ian Bell. Real-time solution of the finite element inverse problem of viscoelasticity. *Inverse Problems*, 27(8):085002, 2011.

[Fel00] Carlos A Felippa. A systematic approach to the element-independent corotational dynamics of finite elements. Technical Report CU-CAS-00-03, Center for aerospace structures, 2000.

[FGBP11] François Faure, Benjamin Gilles, Guillaume Bousquet, and Dinesh K. Pai. Sparse meshless models of complex deformable solids. *ACM Transactions on Graphics*, 30(4):1, 2011.

[GBB09] Dan Gerszewski, Haimasree Bhattacharya, and Adam W. Bargteil. A point-based method for animating elastoplastic solids. *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation - SCA '09*, page 133, 2009.

[GBFP11] Benjamin Gilles, Guillaume Bousquet, Francois Faure, and Dinesh K. Pai. Frame-based elastic models. *ACM Transactions on Graphics*, 30(2):1–12, 2011.

[GM77] R. a. Gingold and J.J. Monaghan. Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Mon. Not. R. astr. Soc.*, (181):375–389, 1977.

[GQ05] Xiaohu Guo and Hong Qin. Real-time meshless deformation. *Computer Animation and Virtual Worlds*, 16(3-4):189–200, 2005.

[GRWM03] Ian A Gravagne, Christopher D Rahn, Ian D Walker, and Senior Member. Planar Continuum Robots. 8(2):299–307, 2003.

[GSSP10] Prashant Goswami, Philipp Schlegel, Barbara Solenthaler, and Renato Pajarola. Interactive SPH simulation and rendering on the GPU. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '10, pages 55–64, Aire-la-Ville, Switzerland, Switzerland, 2010. Eurographics Association.

[GW08] Joachim Georgii and R Westermann. Corotated Finite Elements Made Fast and Stable. In *Proceedings of the 5th Workshop On Virtual Reality Interaction and Physical Simulation*, pages 11–19, 2008.

[HK08] Simone E. Hieber and Petros Koumoutsakos. A Lagrangian particle method for the simulation of linear and nonlinear elastic models of soft tissue. *Journal of Computational Physics*, 227(21):9195–9215, nov 2008.

[HMH07] S M Hosseini, M T Manzari, and S K Hannani. A fully explicit three-step SPH algorithm for simulation of non-Newtonian fluid flow. *International Journal of Numerical Methods for Heat & Fluid Flow*, 17(7):715–735, 2007.

[HNR$^+$10] A Hostettler, S A Nicolau, Y Rémond, J Marescaux, and L Soler. A real-time predictive simulation of abdominal viscera positions during quiet free breathing. *Progress in Biophysics and Molecular Biology*, 103(2-3):169–184, 2010.

[HWJM10] Ashley Horton, Adam Wittek, Grand Roman Joldes, and Karol Miller. A meshless Total Lagrangian explicit dynamics algorithm for surgical simulation. *Integration The Vlsi Journal*, 26(8):977–998, 2010.

[HWM07] Ashley Horton, Adam Wittek, and Karol Miller. Subject-specific biomechanical simulation of brain indentation using a meshless method. *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, 10(Pt 1):541–548, 2007.

[ISF07] Geoffrey Irving, Craig Schroeder, and Ronald Fedkiw. Volume conserving finite element simulations of deformable models. *ACM Transactions on Graphics*, 26(3):13, 2007.

[ITF04] G Irving, J Teran, and R Fedkiw. Invertible Finite Elements for Robust Simulation of Large Deformation. *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, pages 131–140, 2004.

[Jam13] John T James. A new, evidence-based estimate of patient harms associated with hospital care. *Journal of patient safety*, 9(3):122–8, sep 2013.

[JL12] Hoeryong Jung and DY Lee. Real-time cutting simulation of meshless deformable object using dynamic bounding volume hierarchy. *Computer Animation and Virtual Worlds*, 23(5):489–501, 2012.

[JWM10] Grand Roman Joldes, Adam Wittek, and Karol Miller. Real-time nonlinear finite element computations on GPU - Application to neurosurgical simulation. *Computer Methods in Applied Mechanics and Engineering*, 199(49-52):3305–3314, 2010.

[KB00] S Kulasegaram and J Bonet. Corrected Smooth Particle Hydrodynamics – A Meshless Method for Computational Mechanics. (September):11–14, 2000.

[KC08] Dan Koppel and Shiv Chandrasekaran. A new framework for behavior modeling of organs and soft tissue using the Boundary-Element Methods. *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pages 1–6, 2008.

[KCO$^+$03] A Kerdok, Stéphane Cotin, M Ottensmeyer, A Galea, R Howe, and Steve Dawson. Truth Cube: Establishing Physical Standards for Real-Time Soft Tissue Simulation. *Medical Image Analysis*, 7(3), 2003.

[KCOZ06] Ladislav Kavan, Steven Collins, C O'Sullivan, and Jiri Zara. Dual quaternions for rigid transformation blending. *Technical report*, 2006.

[KCŽO07] Ladislav Kavan, Steven Collins, Ji\vrí Žára, and Carol O'Sullivan. Skinning with dual quaternions. *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, pages 39–46, 2007.

[KGRB12] Pierre Kerfriden, Olivier Goury, Timon Rabczuk, and Stephane Pierre-Alain Bordas. A partitioned model order reduction approach to rationalise computational expenses in multiscale fracture mechanics. 44(0):1–35, 2012.

[KMBG08] Peter Kaufmann, Sebastian Martin, Mario Botsch, and Markus Gross. Flexible simulation of deformable models using discontinuous Galerkin FEM. *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 105–115, 2008.

[LBOK13] Tiantian Liu, Adam W. Bargteil, James F. O'Brien, and Ladislav Kavan. Fast simulation of mass-spring systems. *ACM Transactions on Graphics*, 32(6):1–7, 2013.

[LCDN06] Julien Lenoir, Stephane Cotin, Christian Duriez, and Paul Neumann. Physics-based models for catheter, guidewire and stent simulation. *Studies in health technology and informatics*, 119:305–310, 2006.

[LCSP05] Vincent Luboz, Matthieu Chabanas, Pascal Swider, and Yohan Payan. Orbital and maxillofacial computer aided surgery: patient-specific finite element models to predict surgical outcomes. *Computer methods in biomechanics and biomedical engineering*, 8(4):259–265, 2005.

[LD07] Yi-Je Lim and Suvranu De. Real time simulation of nonlinear tissue response in virtual surgery using the point collocation-based method of finite spheres. *Computer Methods in Applied Mechanics and Engineering*, 196(31-32):3011–3024, jun 2007.

[LgDC$^+$13] Pauline Lecomte-grosbras, Mouhamadou Nassirou Diallo, Michel Cosson, Christian Duriez, and Matthias Brieu. Towards a Better Understanding of Pelvic System Disorders Using Numerical Simulation. pages 307–314, 2013.

[LHLW11] Ning Liu, Xiaowei He, Sheng Li, and Guoping Wang. Meshless simulation of brittle fracture. *Comput. Animat. Virtual Worlds*, 22(2-3):115–124, 2011.

[LJ95] WK Liu and S Jun. Reproducing kernel particle methods. *International Journal for Numerical Methods in Fluids*, 20(8-8):1081–1106, 1995.

[LS81] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–141, 1981.

[Luc77] L. B. Lucy. A numerical approach to the testing of the fission hypothesis. *The Astronomical Journal*, 82:1013, 1977.

[LZJC04] Winson C C Lee, Ming Zhang, Xiaohong Jia, and Jason T M Cheung. Finite element modeling of the contact interface between trans-tibial residual limb and prosthetic socket. *Medical engineering & physics*, 26(8):655–62, oct 2004.

[MBB+02] Kevin Montgomery, Cynthia Bruyns, Joel Brown, Stephen Sorkin, Frederic Mazzella, Guillaume Thonier, Arnaud Tellier, Benjamin Lerman, and Anil Menon. Spring: A general framework for collaborative, real-time surgical simulation. *Studies in Health Technology and Informatics*, 85:296–303, 2002.

[MC00] By P Meseure and C Chaillou. A deformable body model for surgical simulation. *The Journal of Visualization and Computer Animation*, 11(4):197–208, 2000.

[MC11] M Müller and Nuttapong Chentanez. Solid simulation with oriented particles. *ACM Transactions on Graphics*, 30(4), 2011.

[MDM+00] Matthias Müller, Julie Dorsey, Leonard Mcmillan, Robert Jagnow, and Barbara Cutler. Stable Real-Time Deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 49–55, 2000.

[MH05] M Müller and Bruno Heidelberger. Meshless deformations based on shape matching. *ACM Transactions on Graphics*, 24(3):471–478, 2005.

[MHC+10a] Stéphanie Marchesseau, T Heimann, S Chatelin, R Willinger, and Hervé Delingette. Fast porous visco-hyperelastic soft tissue model for surgery simulation: Application to liver surgery. *Progress in Biophysics and Molecular Biology / Progress in Biophysics & Molecular Biology*, 103(2-3):185–196, 2010.

[MHC+10b] Stéphanie Marchesseau, Tobias Heimann, Simon Chatelin, Rémy Willinger, and Hervé Delingette. Multiplicative Jacobian energy decomposition method for fast porous visco-hyperelastic soft tissue model. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6361 LNCS(PART 1):235–242, 2010.

[MJLW07] Karol Miller, Grand Joldes, Dane Lance, and Adam Wittek. Total Lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. *Communications in Numerical Methods in Engineering*, 23(2):121–134, 2007.

[MKB$^+$10] Sebastian Martin, Peter Kaufmann, Mario Botsch, Eitan Grinspun, and Markus Gross. Unified simulation of elastic rods, shells, and solids. *ACM Transactions on Graphics*, 29(4):1, jul 2010.

[MKN$^+$04] M Müller, R Keiser, A Nealen, M Pauly, M Gross, and M Alexa. Point Based Animation of Elastic, Plastic and Melting Objects. *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 141–151, 2004.

[MNjFS14] Pedro S Martins, Renato M Natal-jorge, Francisca L Ferreira, and Agostinho Santos. Mechanical behaviour of soft biological tissues after death. (Wccm Xi):20–21, 2014.

[MSH04] Yasmin Meleán, L. D G Sigalotti, and Anwar Hasmy. On the SPH tensile instability in forming viscous liquid drops. *Computer Physics Communications*, 157(3):191–200, 2004.

[MSNS05] Wouter Mollemans, Filip Schutyser, Nasser Nadjmi, and Paul Suetens. Very fast soft tissue predictions with mass tensor model for maxillofacial surgery planning systems. *International Congress Series*, 1281:491–496, 2005.

[MVM$^+$11] T Mansi, I Voigt, E Assoumou Mengue, R Ionasec, and B Georgescu. Simulation of MitralClip Procedure. In Terry Fichtinger, Gabor and Martel, Anne and Peters, editor, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2011*, volume 6891, pages 452–459. Springer Berlin Heidelberg, 2011.

[NJ07] Srinivas Neppalli and Bryan A Jones. Design, construction, and analysis of a continuum robot. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1503–1507. Ieee, 2007.

[NKJF09] Matthieu Nesme, Paul G. Kry, Lenka Jeřábková, and François Faure. Preserving topology and elasticity for embedded deformable models. *ACM Transactions on Graphics*, 28(3):1, 2009.

[NPF05] Matthieu Nesme, Yohan Payan, and François Faure. Efficient, Physically Plausible Finite Elements. In John Dingliana and Fabio

Ganovelli, editors, *Eurographics 2005, Short papers, August, 2005*, Trinity College, Dublin, Irlande, 2005.

[NRBD08] Vinh Phu Nguyen, Timon Rabczuk, Stéphane Bordas, and Marc Duflot. Meshless methods: A review and computer implementation aspects. *Mathematics and Computers in Simulation*, 79(3):763–813, dec 2008.

[NT98] L.P. Nedel and D. Thalmann. Real time muscle deformations using mass-spring systems. *Proceedings. Computer Graphics International (Cat. No.98EX149)*, pages 156–165, 1998.

[NTV92] B. Nayroles, G. Touzot, and P. Villon. Generalizing the finite element method: Diffuse approximation and diffuse elements. *Computational Mechanics*, 10(5):307–318, 1992.

[NZ09] Hennadiy Netuzhylov and Andreas Zilian. Space–time meshfree collocation method: Methodology and application to initial-boundary value problems. *International Journal for Numerical Methods in Engineering*, 80(May):355–380, 2009.

[ODYK12] Masato Ogata, Yasunori Dohi, Takahiro Yamada, and Yoshinobu Kubota. Implementation and evaluation of hyperelastic model for surgical simulator and navigation. *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pages 6297–6300, 2012.

[OFTB96] D Organ, M Fleming, T Terry, and T. Belytschko. Continuous meshless approximations for nonconvex bodies by diffraction and transparency. *Computational Mechanics*, 18, 1996.

[Pai02] Dinesh K. Pai. STRANDS: Interactive simulation of thin solids using cosserat models. *Computer Graphics Forum*, 21(3):347–352, 2002.

[Pay12] Yohan Payan. *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*, volume 11. Springer-Verlag, Berlin, 2012.

[PDA03] Guillaume Picinbono, Hervé Delingette, and Nicholas Ayache. Non-linear anisotropic elasticity for real-time surgery simulation. *Graphical Models*, 65(5):305–321, 2003.

[PDC12] Igor Peterlík, Christian Duriez, and Stéphane Cotin. Modeling and real-time simulation of a vascularized liver tissue. *Medical image* *computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, 15(Pt 1):50–7, 2012.

[PLC+12] Emmanuel Promayon, Vincent Luboz, Grégory Chagnon, Thierry Alonso, Denis Favier, Christine Barthod, and Yohan Payan. Comparison of LASTIC (Light Aspiration device for in vivo Soft TIssue Characterization) with classic Tensile Tests. In Y Payan, editor, *Proceedings of the EUROMECH534 Colloquium.*, pages 75–76. 2012.

[PPC+14] Rosalie Plantefeve, Igor Peterlik, Hadrien Courtecuisse, Raffaella Trivisonne, and Jean-pierre Radoux. Atlas-based Transfer of Boundary Conditions for Biomechanical Simulation. In *MICCAI - 17th International Conference on Medical Image Computing and Computer Assisted Intervention*, 2014.

[PPLT09] A Paiva, F Petronetto, T Lewiner, and G Tavares. Particle-based viscoplastic fluid/solid simulation. *Computer-Aided Design*, 41(4):306–314, 2009.

[Prz85] J. Przemieniecki. *Theory of matrix structural analysis*. McGraw-Hill, 1985.

[RBX04] T. Rabczuk, T. Belytschko, and S.P. Xiao. Stable particle methods based on Lagrangian kernels. *Computer Methods in Applied Mechanics and Engineering*, 193(12-14):1035–1063, mar 2004.

[RRD+11] G. Rivaux, C. Rubod, B. Dedet, M. Brieu, B. Gabriel, L. Landscheere, P. Devos, V. Delmas, and M. Cosson. Biomechanical characterisation of uterine ligaments. Implications for the pelvic floor. *Pelvi-périnéologie*, 6(2):67–74, 2011.

[SB12] Hagit Schechter and Robert Bridson. Ghost SPH for animating water. *ACM Transactions on Graphics*, 31(4):1–8, 2012.

[SCB+09] P. Schiavone, F. Chassat, T. Boudou, E. Promayon, F. Valdivia, and Y. Payan. In vivo measurement of human brain elasticity using a light aspiration device. *Medical Image Analysis*, 13(4):673–678, 2009.

[SE10] Aria Shahingohar and Roy Eagleson. A framework for GPU accelerated needle insertion simulation using meshfree methods. In *Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2010)*, 2010.

[SHA95] J.W. Swegle, D.L. Hicks, and S.W. Attaway. Smoothed Particle Hydrodynamics Stability Analysis *. *Journal of Computational Physics*, 134:123–134, 1995.

[She94] JR Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. 1994.

[SHZO07] Shubhabrata Sengupta, Mark Harris, Yao Zhang, and John D Owens. Scan Primitives for GPU Computing. In *GRAPHICS HARDWARE 2007*, pages 97–106. Association for Computing Machinery, 2007.

[Sis12] E Sismanidis. In \cite. In *Virtual Reality Short Papers and Posters (VRW), 2012 IEEE*, pages 109–110, 2012.

[SM99] N Sukumar and B Moran. C 1 natural neighbor interpolant for partial differential equations. *... Methods for Partial Differential Equations*, 1999.

[SM06] TS Sørensen and Jesper Mosegaard. An introduction to GPU accelerated surgical simulation. *Proceedings of the Third international conference on Biomedical Simulation*, pages 93–104, 2006.

[SMSB01] N Sukumar, B Moran, A Yu Semenov, and V V Belikov. Natural neighbour Galerkin methods. *International Journal for Numerical Methods in Engineering*, 50(March 2000):1–27, 2001.

[SOG09] Denis Steinemann, Miguel a. Otaduy, and Markus Gross. Splitting meshless deforming objects with explicit surface tracking. *Graphical Models*, 71(6):209–220, nov 2009.

[SP07] By Barbara Solenthaler and Renato Pajarola. A unified particle model for fluid – solid. 1:69–82, 2007.

[SP09] B Solenthaler and R Pajarola. Predictive-corrective incompressible SPH. *ACM Transactions on Graphics*, 28(3):40:1–40:6, jul 2009.

[SS07] Eftychios Sifakis and Tamar Shinar. Hybrid Simulation of Deformable Solids. *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 81–90, 2007.

[SSC$^+$13] Alexey Stomakhin, Craig Schroeder, Lawrence Chai, Joseph Teran, and Andrew Selle. A material point method for snow simulation. *ACM Transactions on Graphics*, 32(4):1, 2013.

[Suk04] N. Sukumar. Construction of polygonal interpolants: A maximum entropy approach. *International Journal for Numerical Methods in Engineering*, 61(12):2159–2181, 2004.

[TCC+08] Zeike a. Taylor, Olivier Comas, Mario Cheng, Josh Passenger, David J. Hawkes, David Atkinson, and Sébastien Ourselin. Modelling anisotropic viscoelasticity for real-time soft tissue simulation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5241 LNCS(PART 1):703–710, jan 2008.

[TECO14] Rosell Torres, Jose M Espadero, Felipe A Calvo, and Miguel A Otaduy. Interactive Deformation of Heterogeneous Volume Data, 2014.

[VRW+11] a. Vallet, C. Rubod, J. F. Witz, M. Brieu, and M. Cosson. Simulation of pelvic mobility: topology optimisation of ligamentous system. *Computer Methods in Biomechanics and Biomedical Engineering*, 14(sup1):161–162, 2011.

[VVW04] Lara M Vigneron, Jacques G Verly, and Simon K Warfield. Modelling Surgical Cuts, Retractions, and Resections via Extended Finite Element Method. *Proceedings of MICCAI*, pages 311–318, 2004.

[WBJ+09] P Wang, a a Becker, I a Jones, a T Glover, S D Benford, and M Vloeberghs. Real-time surgical simulation for deformable soft-tissue objects with a tumour using Boundary Element techniques. *Journal of Physics: Conference Series*, 181:012016, 2009.

[WKWM05] Adam Wittek, Ron Kikinis, Simon K Warfield, and Karol Miller. Brain shift computation using a fully nonlinear biomechanical model. *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, 8(Pt 2):583–590, 2005.

[WP02] Xiaohuan Corina Wang and Cary Phillips. Multi-weight enveloping: least-squares approximation techniques for skin animation. In *Proceedings of the 2002 Symposium on Computer Animation*, pages 129–138, New York, NY, USA, 2002. ACM.

[WPL+05] Xunlei Wu, Vincent Pegoraro, Vincent Luboz, Paul F Neumann, Ryan Bardsley, Steven Dawson, and Stephane Cotin. New approaches to computer-based interventional neuroradiology training. *Studies in health technology and informatics*, 111:602–607, 2005.

[YLW+14] Chen Yang, Shuai Li, Lili Wang, Aimin Hao, and Hong Qin. Real-time physical deformation and cutting of heterogeneous objects via hybrid coupling of meshless approach and finite element method. *Computer Animation and Virtual Worlds*, 25(3-4):421–433, may 2014.

[ZLKW13] Yahan Zhou, Zhaoliang Lun, Evangelos Kalogerakis, and Rui Wang. Implicit integration for particle-based simulation of elasto-plastic solids. *Computer Graphics Forum*, 32(7):215–223, 2013.

[ZT00] Olgierd Cecil Zienkiewicz and Robert Leroy Taylor. *Finite Element Method: Volume 1, Fifth Edition*, volume 1. Butterworth-Heinemann, 2000.

[ZWP05] Hualiang Zhong, Mark P Wachowiak, and Terry M Peters. A real time finite element based tissue simulation method incorporating nonlinear elastic behavior. *Computer methods in biomechanics and biomedical engineering*, 8(3):177–189, 2005.

[ZYJ+10] Ling Zhu, Xiufen Ye, Xi Ji'er, Yizhou Gu, and Shuxiang Guo. A real-time deformation modeling scheme of soft tissue for virtual surgical. *2010 IEEE International Conference on Information and Automation, ICIA 2010*, pages 771–775, 2010.