Mechanical compliance: from soft robot modeling theory to finite element method computation

Christian Duriez

Mechanical compliance: from soft robot modeling theory to finite element method computation

Christian Duriez

TECHNICAL REPORT
N° 0521
February 2024
Project-Team Defrost

ISSN 0249-0803 ISRN INRIA/RT--0521--FR+ENG

Mechanical compliance: from soft robot modeling theory to finite element method computation

Christian Duriez*

Project-Team Defrost

Technical Report n° 0521 — February 2024 — 34 pages

Abstract: This technical report proposes a generic modeling method for soft robots. This method is based on the notion of mechanical compliance, which provides a compact description of the behavior of a deformable robot. In particular, the kinematics of a soft manipulator arm can be derived from this notion. The chapter then shows how this compliance can be calculated from an analysis of the robot's materials and mechanical structure. Starting from continuum mechanics, we describe how the equations can be integrated using the finite element method (FEM). This method is known to be computationally time-consuming, so the chapter briefly introduces the notion of model reduction by projection. Then, using a Lagrangian formulation, actuation, end-effectors, sensors and contacts are introduced which are essential for robot modeling and not often found in FEM formulations. In particular, models for fluidic, tendon-based and articulation-based actuatiors are detailed. Finally, this chapter describes how to obtain inverse models on soft robots and how to use them to invert kinematics, measure external forces without any force sensor or take contacts into account in the robot model-based control.

Key-words: robotics, soft robotics, deformation modeling, finite element method, real-time, statics, kinematics, kinetics and dynamics of a soft robot

* Univ. Lille, Inria, CNRS, Centrale Lille, UMR 9189 CRIStAL, F-59000 Lille, France

RESEARCH CENTRE
LILLE – NORD EUROPE

Parc scientifique de la Haute-Borne
40 avenue Halley - Bât A - Park Plaza
59650 Villeneuve d'Ascq

Compliance Mécanique: d'une théorie de la modélisation pour les robots souples au calcul par la méthode des éléments finis

Résumé : Ce rapport technique propose une méthode générique de modélisation des robots souples. Cette méthode est basée sur la notion de conformité mécanique, qui fournit une description compacte du comportement d'un robot déformable. En particulier, la cinématique d'un bras manipulateur souple peut être dérivée de cette notion. Le chapitre montre ensuite comment cette compliance peut être calculée à partir d'une analyse des matériaux et de la structure mécanique du robot. En partant de la mécanique des milieux continus, nous décrivons comment les équations peuvent être intégrées à l'aide de la méthode des éléments finis (FEM). Cette méthode est connue pour être co ûteuse en temps de calcul, c'est pourquoi le chapitre introduit brièvement la notion de réduction de modèle par projection. Ensuite, à l'aide d'une formulation lagrangienne, l'actionnement, les effecteurs, les capteurs et les contacts sont introduits, ce qui est essentiel pour la modélisation des robots et n'apparaît pas souvent dans les formulations FEM. En particulier, les mod èles pour les actionneurs fluidiques, à base de tendons et à base d'articulations sont détaillés. Enfin, ce chapitre décrit comment obtenir des modèles inverses sur les robots souples et comment les utiliser pour inverser la cinématique, mesurer les forces externes sans capteur de force ou prendre en compte les contacts dans la commande basée sur le modèle du robot.

Mots-clés : robotique, robotique souple, modélisation des déformations, méthodes des éléments finis, temps-réel, statique, cinématique et dynamique d'un robot souple

Contents

1 Introduction **4**

2 Modeling soft robot through mechanical projected compliance **5**

3 Notions of continuum mechanics **7**
3.1 Motion tracking . . . 7
3.2 Strain tensors . . . 8
3.3 Stress tensors . . . 10
3.4 Constitutive laws . . . 11
3.5 Variational formulation . . . 12

4 Computing soft robot configuration and compliance from Finite Element Modeling **13**
4.1 Discussion about dimensionalities . . . 13
4.2 Elements: Geometry, topology, interpolation and nodes . . . 13
4.3 Internal forces computation and assembling . . . 14

5 Equations of motion **15**
5.1 Quasi-static motion . . . 15
5.2 Dynamic motion . . . 16

6 Lagrangian mechanics formulation **18**
6.1 DOF reduction and stiffness projection . . . 19
6.2 Compliance projection . . . 21

7 Actuator models **23**
7.1 Pressure based Actuation . . . 23
7.2 Tendon based Actuation . . . 23
7.3 Direct actuation with stiffness projection . . . 24

8 Compliance based models of soft robots **25**
8.1 Effector model and projected compliance . . . 26
8.2 Kinematics and inverse model by optimization . . . 26
8.3 Sensor model and external forces measurement . . . 27

9 Contact interaction with the environment **28**
9.1 Contact model . . . 28
9.2 Direct Solving process using compliance projection . . . 29
9.3 Inverse modeling with contact . . . 30

10 Conclusion **31**

References **32**

1 Introduction

This chapter summarizes the work carried out by the DEFROST team these last years on the mechanical modeling of soft robots using the finite element method, computed in real time. The work of the team is constantly implemented in the framework SOFA which is open-source, allowing the community to have an implementation of what is described in this chapter. The way in which this chapter has been written is original in relation to the team's previous publications. The emphasis is put on purpose on mechanical compliance, which, to the opinion of the author, is at the heart of soft robot modeling theory.

Soft robots have solid structures and create their motion by deformation. The behavior of the robot depends on the type of material used to build its body. From a mechanical point of view, we can also speak of deformable robots, just as we speak of deformable solids in mechanics. The kinematics at the scale of the soft robot as a whole (the relationship between the actuator motions and the robot body motions) depends on mechanics and in particular on the balance between internal forces, external forces, and actuation-related forces.

This is one of the main difference between rigid and deformable robots. At this macroscopic scale, the kinematic relations of the robot are therefore no longer only geometrical as for rigid robots, but derive from the mechanics of the structure, and in particular from its structural mechanical compliance. In this chapter, we will see how the mechanical compliance (the opposite of stiffness), gives the movement created on a structure at certain points, called effector or sensor spaces, by the action of forces exerted on it by actuators (actuator space) or contacts (contact space). This compliance, derived from the robot's deformable statics, enables to write the kinematics and inverse kinematics of these robots.

The computation of this compliance derives from solid body deformation laws of continuum mechanics. Continuum mechanics focus on deformations at a small scale: that of all particles that compose the robot's deformable body. It builds kinematic relations between particle motion and deformation tensors allow to derive physical models, based on the constitutive laws of material. As it includes the material properties, it is an excellent basis for accurate modeling of soft robots. We will very briefly describe the type of equations we obtain for 3D elastic solid bodies and we will try to draw some links with the equations used in robotics. Unfortunately, there is no general analytical solution for these PDE, except in very idealized (and unrealistic) cases.

In numerical mechanics, a variety of numerical methods have been developed to find convergent approximate solutions, and one of the most famous one is the Finite Element Method. Numerical methods are commonly used in engineering to integrate these equations and model deformable structures. The finite element family is one of these numerical methods and is probably the most popular for this purpose. In civil engineering for example, the finite element method is used to assess structure strength and stability. But in these types of applications, there is no special concern of having interactive simulations, running at high frame rate. One of the main challenge in modeling soft robots is to make numerical methods compatible with real-time simulation which is necessary for control. We will also briefly discuss this point, in particular by presenting model order reduction.

Once numerical methods explained for obtaining deformable robot models, we'll see how to model actuators as constraints, using Lagrange multipliers. We'll then extend the constraint-based approach to other spaces of interest in robotics: effectors, sensors and contacts in particular, and we'll see why and how to project mechanical compliance into these different spaces. This will enable us to derive direct and inverse models of deformable soft robots, including in contact situations.

We emphasize that this chapter is still relatively short, and that you should not hesitate to refer to other books, particularly on continuum mechanics and FEM. For more detailed informa-

tion on the modeling of the soft robots presented in this chapter, please refer to the publications and theses of the DEFROST team.

2 Modeling soft robot through mechanical projected compliance

In mechanics, compliance is defined as the inverse of stiffness. This concept is best known in mechanics for finding the equivalent spring for springs placed in series. In the 1D case of springs in series shown in the figure, we find the equivalent spring by applying the formula:

$$1/k_{eq} = 1/k_1 + 1/k_2 + ... + 1/k_N$$

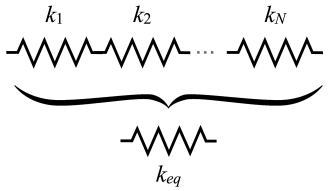


Figure 1: Use mechanical compliance to find the value of a spring equivalent k_{eq} to springs connected in series.

In this series alignment of springs, the interacting forces are the same at each link. But potentially, this force, applied to each end of the springs, creates a particular deformation, depending on the stiffness of the spring. So, in the broadest sense, compliance describes the movement created by a force. On a deformable structure such as a soft robot, compliance is obtained by taking into account the entire structure, integrating the stiffness of the materials in the domain, taking into account boundary conditions and inverting the expression for the tangent stiffness of the entire structure. The expression of compliance is therefore not easy to obtain on soft robots in general. It is often non-linear and depends not only on the overall configuration of the robot, referred to here as $\mathbf{q}$, but also on the law of material behavior and boundary conditions.

Soft robots are often made of elastic materials (silicone, plastic, rubber, etc.) to which actuating forces are applied, as well as external forces (gravity, contact, etc.). The aim of modeling is to describe the movements created by these forces. In soft robotics, actuation forces are generally linked to an effort parameter (pressure in a cavity, tension in a tendon, torque exerted by a motor, etc.). This effort has a motion dual (cavity volume, tendon length, motor rotation, etc.). Mechanical compliance provides a very compact description of the coupling between these actuating force parameters and the movements they create on the structure at certain specific points.

We'll be talking about projected compliance, since we'll be looking at the value of this compliance in the actuator/effector/sensor subspaces, but also using subspaces linked to contact when the robot comes into contact with its environment.

The robot diagram below schematically describes the notion of projected compliance in the actuator/effector spaces and the link with the kinematics on a deformable robot.

In the spirit of virtual work, projected compliance allows us to describe a relationship between notions of variation in displacement $\boldsymbol{\Delta\delta}$ created by increments in force on actuators $\boldsymbol{\Delta\lambda}_{\mathrm{a}}$ (see figure 2). This relationship is linearized around the current configuration for small force

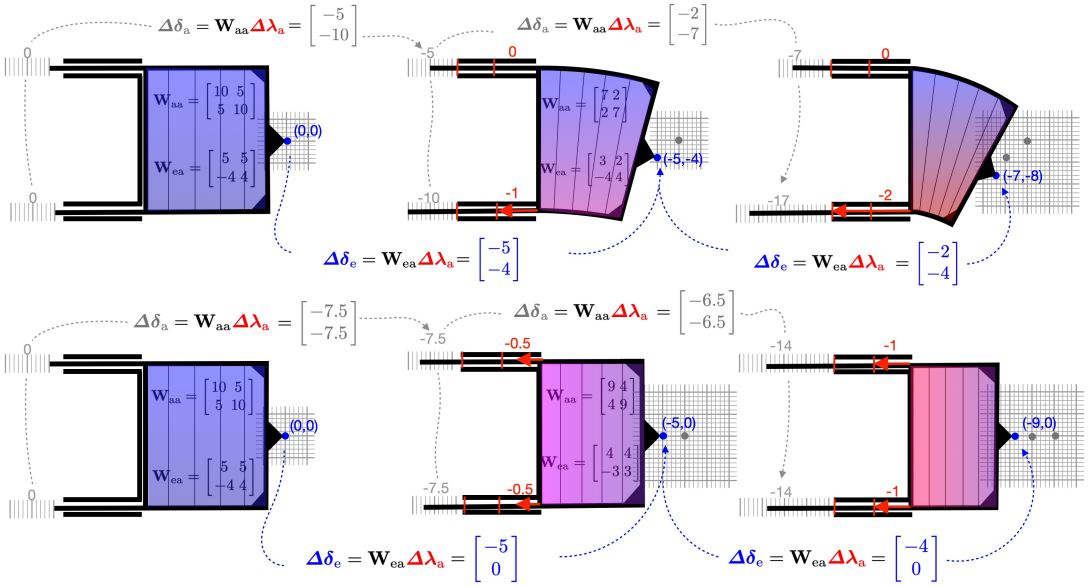


Figure 2: **Kinematics through the mechanical compliance of soft robots**: To describe the kinematics of a soft robot, the mechanical compliance of the manipulator structure is projected into actuator space with $\mathbf{W}_{aa}$ and into effector-actuator space with $\mathbf{W}_{ea}$. Thanks to these two operators, we can obtain the displacements of the actuators ($\Delta\delta_a$) and of the effector ($\Delta\boldsymbol{\delta}_e$) when increments of forces are applied to the actuators ($\Delta\boldsymbol{\lambda}_a$). To illustrate, the figure shows a dummy soft robot, actuated by two linear pistons. The robot's behavior is decomposed in three steps with two different sets of incremental actuator forces on the top and bottom lines. We show that the value of compliances is not constant due to non-linear deformations. In this example, the kinematics of the soft robot can be calculated using the equation 1.

increments. When observing motion from effector points, we use $\Delta\boldsymbol{\delta}_e = \mathbf{W}_{ea}(\mathbf{q})\Delta\boldsymbol{\lambda}_a$. Thanks to compliance projected into the effector/actuator space, we have here a compact expression for the variation in motion on the effectors created by an increment of forces on the actuators. Similarly, we can use projected compliance to describe the motion, in actuator space, created by this increment of actuator forces (forces are often coupled by the mechanics): $\Delta\boldsymbol{\delta}_a = \mathbf{W}_{aa}(\mathbf{q})\Delta\boldsymbol{\lambda}_a$. Here, compliance is projected on both sides (force/movement) into actuator space.

So, if we have the expression of $\mathbf{W}_{aa}(\mathbf{q})$ and $\mathbf{W}_{ea}(\mathbf{q})$ for the current robot configuration $\mathbf{q}$, we can describe the **kinematic model of soft manipulators**, by the form:

$$\Delta\boldsymbol{\delta}_e = \mathbf{W}_{ea}(\mathbf{q})\mathbf{W}_{aa}^{-1}(\mathbf{q})\Delta\boldsymbol{\delta}_a = \mathbf{J}_{SR}(\mathbf{q})\Delta\boldsymbol{\delta}_a \tag{1}$$

This expression gives the variation in movement created on the end-effector, by a variation in movement on the actuator. Thanks to projected compliance, we have a general expression for the Jacobian for soft robots $\mathbf{J}_{SR}(\mathbf{q}) = \mathbf{W}_{ea}(\mathbf{q})\mathbf{W}_{aa}^{-1}(\mathbf{q})$. This Jacobian has similar characteristics of the Jacobians used on rigid manipulators, in particular the loss of rank when there is a singularity.

In the case of the dummy robot presented in figure 2, the value of the jacobian at the starting position is:

$$\mathbf{J}_{SR}(\mathbf{q}) = \mathbf{W}_{ea}(\mathbf{q})\mathbf{W}_{aa}^{-1}(\mathbf{q}) = \begin{bmatrix} 5 & 5 \\ -4 & 4 \end{bmatrix} \begin{bmatrix} 10 & 5 \\ 5 & 10 \end{bmatrix}^{-1} = \begin{bmatrix} \frac{5}{15} & \frac{5}{15} \\ \frac{-12}{15} & \frac{12}{15} \end{bmatrix}$$

If we apply the first actuator displacement increments (see the passage from column 1 to column 2 in the fig 2) , we obtain the corresponding displacement of the effector:

$$\underbrace{\begin{bmatrix} \frac{5}{15} & \frac{5}{15} \\ \frac{-12}{15} & \frac{12}{15} \end{bmatrix}}_{\mathbf{J}_{SR}} \underbrace{\begin{bmatrix} -5 \\ -10 \end{bmatrix}}_{\Delta\boldsymbol{\delta}_a} = \underbrace{\begin{bmatrix} -5 \\ -4 \end{bmatrix}}_{\Delta\boldsymbol{\delta}_e} \quad \text{and} \quad \underbrace{\begin{bmatrix} \frac{5}{15} & \frac{5}{15} \\ \frac{-12}{15} & \frac{12}{15} \end{bmatrix}}_{\mathbf{J}_{SR}} \underbrace{\begin{bmatrix} -7.5 \\ -7.5 \end{bmatrix}}_{\Delta\boldsymbol{\delta}_a} = \underbrace{\begin{bmatrix} -5 \\ 0 \end{bmatrix}}_{\Delta\boldsymbol{\delta}_e}$$

Then, when the robot deforms, the Jacobian matrices (and the matrices of projected compliance) are changing due to deformation non-linearities (see matrices in column 2 in the Fig 2).

Note that the formula requires the matrix $\mathbf{W}_{aa}$ to be invertible. This matrix represents the mechanical coupling between actuators due to the deformable structure. In practice, we often try to decouple the actuators as much as possible in the design so this matrix is usually invertible. Given the high number of degrees of freedom on the deformable structure, it would be surprising, from a design point of view, to couple by actuation the same degrees of freedom in the same directions.

The compliance allow to describe the kinematics of soft deformable robots. We'll see in section 3 how this compliance is obtained for different types of actuators, and all the uses we can make of it, from robot kinematics, inverse kinematics, force estimation, motor/sensor coupling... But before going into detail, let's return to the original problem: how can we model and compute the projected compliance $\mathbf{W}_{ij}(\mathbf{q})$?

The remainder of the chapter will explain how this projected compliance can be obtained using FEM models, in particular to be able to use material behavior laws, and take account of the strong non-linearities in these models.

3 Notions of continuum mechanics

The aim of this section on continuum mechanics and numerical methods is to give a few basic notions. For more detailed information, we recommend the use of reference books such as [31]. First of all, as its name suggests, continuum mechanics considers solids to be filled by a continuum of material points. It's therefore a relatively macroscopic vision of mechanics, at a scale where the medium can be considered continuous (we're not at the scale of atoms, which make up matter in a discrete way).

3.1 Motion tracking

First of all, as in rigid robotics, the starting point is the parametrization of the motion. In continuum mechanics, motion is traditionally described using two different views, the so-called Lagrangian view and the so-called Eulerian view. Considering that most soft robots use elastic materials, the notion of a reference domain makes sense. This reference domain corresponds to the position assumed by the elastic structure when no force is applied to it.

In the Lagrangian view, all particles defined on the domain are tracked from their position in the reference configuration to the current, deformed configuration. In this approach, many calculations are performed in the reference configuration (this allows a certain number of pre-calculations to be performed if this configuration remains unchanged).

In practice, we can define an application $\mathbb{A}$ which tracks the motion. For any point $\mathbf{p} = [x, y, z]$ defined on the domain $\mathbb{A}$ gives its corresponding unique point $\mathbf{p}' = [x', y', z'] = \mathbb{A}(\mathbf{p})$ in the deformed domain.

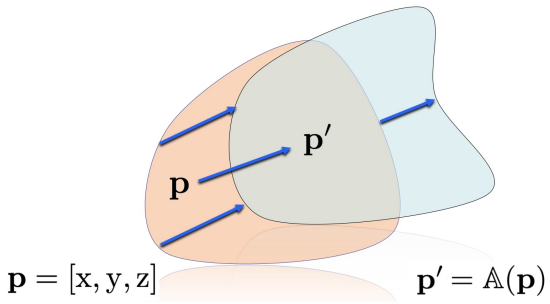


Figure 3: Lagrangian view: for any point $\mathbf{p}$ in the reference domain, the application $\mathbb{A}(\mathbf{p})$ provides $\mathbf{p}'$ the unique corresponding point in the deformed domain.

3.2 Strain tensors

Once this "motion tracking" has been set up for any point in the domain, the deformation is measured using the spatial derivation of this $\mathbb{A}$ function.

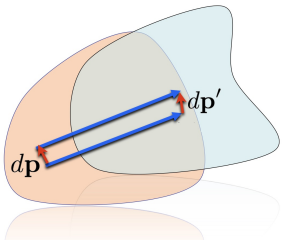


Figure 4: Illustration of the deformation gradient

In this way, we can look at the variation in motion of infinitely close points (see figure 4) in the deformed shape, compared with the initial shape.

$$d\mathbf{p}' = \frac{\partial \mathbb{A}}{\partial \mathbf{p}} d\mathbf{p} = \mathbf{F} d\mathbf{p} \tag{2}$$

In this equation $\mathbf{F} = \frac{\partial \mathbb{A}}{\partial \mathbf{p}}$ is the deformation gradient, which is arguably the simplest tensor to geometrically describe a differential of particle motion, and therefore potentially a deformation. Thus, if $\mathbf{F} = \mathbf{Id}$ ($\mathbf{Id}$ being the identity), at any point in the domain, then the body undergoes no deformation, but is potentially animated by a uniform rigid translation.

This tensor is said to be "translation invariant". To have a zero value when in pure translation, we can write this tensor with the gradient of displacements. We then define $\mathbb{A} = \mathbf{Id} + \mathbb{U}$, $\mathbb{U}$ being the displacement field of any point $\mathbf{p}$ of the domain to its corresponding deformed $\mathbf{p}'$. In this case, the deformation gradient can be re-written as

$$\mathbf{F} = \mathbf{Id} + \mathbf{grad}(\mathbb{U}) \tag{3}$$

However, the value of $\mathbf{F}$ can be influenced by movements other than deformation. For example, a rigid rotation will generate a value of $\mathbf{F} \neq \mathbf{Id}$. So we'll be looking to construct tensors that are independent of this rigid rotation. To do this, we can perform a polar decomposition of the tensor $\mathbf{F} = \mathbf{RU} = \mathbf{VR}$ where $\mathbf{U}$ and $\mathbf{V}$ are symmetric tensors and $\mathbf{R}$ is an anti-symmetric and

proper orthogonal tensor assimilated to a rotation matrix. $\mathbf{R}\mathbf{R}^T = \mathbf{R}^T\mathbf{R} = \mathbf{I}$. This is illustrated in 2D on figure 5

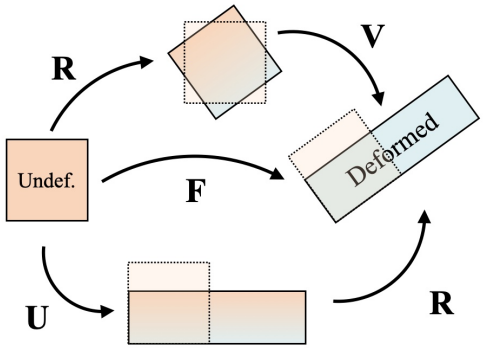


Figure 5: Right and Left polar decomposition of the deformation gradient (**TODO: in 3D ?**)

Comment: *Note that to compute a 3D rotation from a polar decomposition of* $\mathbf{F}$*, the tensor must be defined along the 3 directions. For slender domains like curve or surfaces, used for rods or shells, the rotation cannot be computed that way as there are some directions of non deformation (the section in rods, the thikness in shells). This is why, in the litterature, the* ***Cosserat theory*** *was developed with the use of a definition of domains full of infinitely small rigid bodies tracked both in translation and rotation.*

We can define two strain tensors: the right Cauchy-Green tensor:

$$\mathbf{C} = \mathbf{F}^T\mathbf{F} = (\mathbf{R}\mathbf{U})^T(\mathbf{R}\mathbf{U}) = \mathbf{U}^T\mathbf{R}^T\mathbf{R}\mathbf{U} = \mathbf{U}^2 \tag{4}$$

and the left Cauchy-Green tensor:

$$\mathbf{B} = \mathbf{F}\mathbf{F}^T = (\mathbf{V}\mathbf{R})(\mathbf{V}\mathbf{R})^T = \mathbf{V}\mathbf{R}\mathbf{R}^T\mathbf{V}^T = \mathbf{V}^2 \tag{5}$$

In practice, in the context of elasticity or hyperelasticity, the right Cauchy-Green tensor is more commonly used, as strain is analyzed in the undeformed domain.

This tensor is used to write another widely-used strain tensor, the Green-Lagrange tensor $\mathbf{E} = \mathbf{C} - \mathbf{Id}$, which can also be written as the square norm of the difference between $d\mathbf{p}$ and $d\mathbf{p}'$:

$$\frac{1}{2}(\partial\mathbf{p}'^2 - \partial\mathbf{p}^2) = \frac{1}{2}(\partial\mathbf{p}^T\mathbf{F}^T\mathbf{F}\partial\mathbf{p} - \partial\mathbf{p}^2) = \partial\mathbf{p}^T \underbrace{\frac{1}{2}(\mathbf{F}^T\mathbf{F} - \mathbf{Id})}_{\mathbf{E}}\partial\mathbf{p} \tag{6}$$

Following this definition $\mathbf{E} = \frac{1}{2}(\mathbf{U} - \mathbf{Id})$. When we write the deformation gradient $\mathbf{F} = \mathbf{Id} + \mathbf{grad}(\mathbb{U})$, like in equation (3), we obtain an usual formulation of the Green-Lagrange tensor:

$$\mathbf{E} = \frac{1}{2}(\mathbf{grad}(\mathbb{U}) + \mathbf{grad}(\mathbb{U})^T + \mathbf{grad}(\mathbb{U})^T\mathbf{grad}(\mathbb{U})) \tag{7}$$

When the displacement field $\mathbb{U}$ is small, one can use the linear part of this deformation tensor:

$$\boldsymbol{\epsilon} = \frac{1}{2}(\mathbf{grad}(\mathbb{U}) + \mathbf{grad}(\mathbb{U})^T) \tag{8}$$

Comment: *In a corotational formulation, one can use the rotation* $\mathbf{R}$ *computed for the polar decomposition of* $\mathbf{F}$ *to build a rigid transformation around each point (translation and rotation). Then, the displacement field* $\mathbb{U}$ *of the neighbourhood of this point can be computed in a local frame. In general, this allows to use the small displacement tensor* $\boldsymbol{\epsilon}$ *for quite large rotations but locally small strains. But note that the corotational formulation can also be used for large strains using* $\mathbf{E}$

3.3 Stress tensors

So far, all manipulations have been purely geometric. Strain tensors provide an idea of the geometric deformation of the solid. But depending on the material's behavior, this deformation will create stresses in the domain, i.e. internal forces, which we will also describe with tensors.

The Cauchy tensor $\boldsymbol{\sigma}(\mathbf{p}')$ describes the state of stress at any point on the object and in all directions within the deformed shape. Its components are homogeneous with pressure (in Pa), since they correspond to a force exerted on a unit area around a point ($N.m^{-2}$). This tensor is symmetrical with $\boldsymbol{\sigma} = \boldsymbol{\sigma}^T$. The diagonal terms of the stress tensor correspond to tension/compression and the non-diagonal terms to shear.

At the boundary of the domain, these internal stresses balance out with the pressure forces $\overrightarrow{\mathbf{f}}$ exerted at the surface as illustrated in figure 6

$$\overrightarrow{\mathbf{f}} = \boldsymbol{\sigma}.\overrightarrow{n} \tag{9}$$

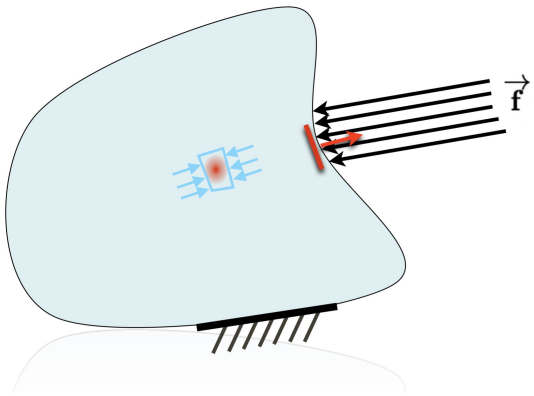


Figure 6: Stress corresponds to the internal force exerted on unit areas around the points of the domain. It has be be balanced with the forces at the boundary.

Comment: *The Cauchy tensor* $\boldsymbol{\sigma}$ *is therefore defined in the deformed form, on which the forces that creates the deformation are exerted. But as stated before we have defined the strain in the reference configuration. So, two additional stress tensors can be used. The First Piola Kirchhoff boldsymbol*σ_{PK1} *for which the strain is defined in the reference domain and the stress in the deformed configuration. And the Second Piola Kirchhoff stress tensor* $\boldsymbol{\sigma}_{PK2}$*, for which the stress is defined in the reference configuration.*

$$\boldsymbol{\sigma}_{PK2} = J\mathbf{F}^{-1}\boldsymbol{\sigma}\mathbf{F}^{-T} = \mathbf{F}^{-1}\boldsymbol{\sigma}_{PK1} \tag{10}$$

3.4 Constitutive laws

The stress being defined, the constitutive law will be used to characterize the mechanical behavior of the material. This law provides a relationship between strain and stress: given its properties the material will generate internal stress when deforming. In the context of this chapter, we will not detail all the possible constitutive laws but just provide a quick overview.

- **Hooke's law** Adapted to linear elastic deformations, it is probably the most simple constitutive law: it is considered that the stress is linearly proportional to the strain. As an elastic law, when there is no effort applied to the deformable body, there is no deformation. Note that this linear behavior is quite frequently observed at relatively low strain levels, and sometimes even with large displacements, i.e. with a non-linear strain tensor to take into account the rigid rotations. Many models (such as the Cosserat beam models widely used in the continuous robot literature) use an exact calculation of the geometric part of the strain, but employ a linear behavior law. In a fully 3D formulation (without the assumption of rods or shells), a linear relationship between the stress and strain tensors can be written using a 4th-order tensor. In practice, with Voigt's notations (stress and strain tensors in vectors of size 6, taking advantage of symmetries) we can write this relationship in matrix form, with a 6x6 matrix $\mathbf{C}$.

$$\boldsymbol{\sigma} = \mathbf{C}\mathbf{E} \tag{11}$$

  If the material is homogeneous and isotropic, two parameters are used by the behavior law: Young's modulus gives the material's level of stiffness and the Poisson ratio gives its level of compressibility. It should be noted that a totally incompressible material cannot be modeled with such a law: this would mean setting the Poisson ratio at 0.5, but the law diverges at such a level, and numerically this leads to a "locking" of deformations. To model complete incompressibility, numerical techniques are used (addition of a pressure unknown in the FEM formulation, use of Lagrange multipliers, etc.), which creates additional difficulties [11].

- **Anisotropy**: Some work demonstrated the interest of using anisotropic material to influence the kinematics of soft robots [27]. We can model anisotropic constitutive law by enriching Hooke's law with directions in which the stiffnesses are not the same. In this case, the behavior law is enriched with the use of several Young's Modules and a definition of the direction of the anisotropies [28].

- **Hyperelasticity**. Some materials, such as rubbers, silicones and other elastomers [18] [10], widely used in soft robotics, allow very high levels of deformation, while maintaining an elastic behavior (no residual deformations when the loading cancels). But this behavior becomes non-linear and is called hyperelastic. In practice, to calculate stress, an energy density function W is used. This function will depend on the strain, and the behavior law will be derived from this relationship:

$$\boldsymbol{\sigma} = \frac{\partial W(\mathbf{E})}{\partial \mathbf{E}} \tag{12}$$

  Of course, we need to define the function W. Here, the behavior is no longer necessarily linear. Several models exist: St Venant Kirchhoff (linear model), Arruda-Boyce, Mooney-Rivling, Ogden etc. The formulations of these models are often based on the invariants of the right-hand Cauchy Green tensor. These models sometimes have many parameters that

are difficult to find experimentally. However, there is a database for materials often used in soft robotics, which provides parameter fittings for the most common behavior laws [17].

- **Other behaviors** Other behaviors can be modeled, such as visco-elasticity [22]. Plasticity and fatigue can also be used but usually they require to keep the loading history.

3.5 Variational formulation

The behavior of the internal forces has been modeled as depending on the deformation of the medium and the stress generated by the behavior law. Continuum mechanics proposes to describe the dynamics of each point of the medium using the equation:

$$\rho\ddot{\mathbf{p}} = div(\boldsymbol{\sigma}) + f_{ext} \tag{13}$$

where ρ represents the mass density, $\ddot{\mathbf{p}}$ the acceleration of the particles, $div(\boldsymbol{\sigma})$ provides the internal force acting on each particle which are balanced with the external forces f_{ext} (gravity or boundary forces from the environment). If some parts of the soft robot are fixed, $d\mathbf{p} = 0$ can be imposed as a boundary condition of the deformable domain.

With this equation, the problem is written in the **strong form** for each point (and there is an infinite number) of the domain. In most of the 3D deformation cases, there is no analytical solution to that equation and that would provide the behavior of each point of the domain. For simplified domains, like beams or shells more analytical solution exist but there are still often very specific. Numerical methods are therefore used to solve the problem. In the case of deformable solid mechanics, methods based a weak form of the problem (which include finite elements) are particularly effective.

Here again, numerous works provide details on the mathematical way of obtaining the weak formulation and the foundations of the associated numerical methods. We can quickly draw a parallel of this principle with tools used in rigid robotics: Assuming that the strain-stress relationship is non-linear, we can write the variation of the density of the strain-energy over the entire volume v of the structure using equation 12 to obtain the weak form:

$$\partial\mathcal{W} = \int_v \partial W \, dV = \int_v \boldsymbol{\sigma}^T \partial\mathbf{E} \, dV \tag{14}$$

1. We find the variation of energy by summing the work of each point in the domain. We draw a first parallel between this equation and the **virtual work** used in rigid robot models: we could suppose that there is a joint at each point of the deformable robot. In the space of each articulation, the motion ($\approx$ the displacement) would be represented by the strain and the effort ($\approx$ the force) would be represented by the stress. Here we use Voigt notation to write the strain and stress (in particular the dot product $\boldsymbol{\sigma}^T\partial\mathbf{E}$). *Note that many approaches using Cosserat's theory explicitly use strain-stress space as the joint space of their model. Once the motion has been solved in this space, it can then be traced back to the motions using a method of integration along the curve, which would be far more complex to do in a surface or volume domain.*

2. By combining this integral form with equation 13, we can also obtain volume terms (notably kinetic energy) and equilibrium with surface terms (virtual work of surface forces, see in equation 9). So the second parallel we can make with classical modeling in robotics, is that this energetic approach brings to **Lagrangian mechanics**, (also widely used for articulated rigid structures) and that will be used in section 6 to integrate actuators, sensors, effectors and also contacts as constraints.

4 Computing soft robot configuration and compliance from Finite Element Modeling

As in the previous section, the aim is not to describe the finite element method of deformable structures in detail. The reader is encourage to read some books to improve the knowledge on such methods [21] [31].

The aim is to understand how this method can be used to model a soft robot, in particular to find its configuration as a function of force fields (actuators, internal forces, external forces, etc.) and to calculate the projected compliance of soft robots, which, as we have introduced in the first section, is a particularly useful tool for describing the motion behavior of soft robots.

To understand the usefulness of finite element methods for elastic solids, we need to go back to continuum mechanics and realize that the equations presented in the previous section have no analytical solution, except for simplified cases (simplified geometries, constitutive laws and loading).

4.1 Discussion about dimensionalities

In terms of continuum mechanics, we can distinguish between the space in which the points move (usually 2D or 3D, but in robotics we're using 3D) and the size of the objects: a very slender object (or robot) with a length that is very large compared to its thickness in the other two dimensions can use simplified rod theory.

String, beams, rods: The simplest model will be the stretched soft string, which only deforms along its direction of length. Then, there are several beam theories, which makes the simplifying assumption that the cross-section of the long object is not deformable. Euler-Bernoulli, Kirchhoff and Cosserat theories fall into this category. Depending on the model, strain calculation is not the same, but for all, it is usually supposed that the material is homogeneous, isotropic and obeying Hooke's law. From a robotics point of view, the model used for "continuum robots" generally fall into this category.

Membranes, plates, shells: An object (or robot) with a large surface area relative to its thickness can also be analysed under simplified theory. Membranes, plates and shells are often analyzed by assumption that the thickness of the structure remains constant under loading. In the membrane case strains are small and are limited to in-plane components. The thickness of the structure is much smaller compared to its other dimensions. Plate models include bending moments but transverse shear effects are neglected. In shell theory the transverse shear is included. The thickness of the shell is small compared to its other dimensions, but not necessarily as small as in membranes or plates. Again for all these models, the material is often assumed to be homogeneous, isotropic, and obeying Hooke's law.

Volume deformations: Finally, if the object (or robot) is not significantly smaller in size than the others, then we use the general theory outlined above. As there is no assumption about the strain (the kinematics of the deformation), any constitutive law can be used in this context.

4.2 Elements: Geometry, topology, interpolation and nodes

These models, derived from continuum theory, can be computed numerically using finite element modeling, but the types of elements used will differ. Elements are defined by their topology in the sense that they will connect some nodes. As this nodes are defined in space, it also defines a geometry. Finally, we will assume that any point of the elements has its kinematics linked to the kinematics of the nodes through the interpolation.

Figure 7 provides some examples of elements and geometrical role and support interpolation. To integrate beam theory (Bernoulli, Kirchhoff, Cosserat), segments or curves are used. Interpolation functions are based on the curvilinear abscissa, and often have a polynomial form. Nodes may have more degree of freedom than just their translation. Some rotations can also be included in the formulation. For instance we can have frame nodes with 3 translations and 3 rotations on beam elements. For surface elements, this is the same approach: In plate or shell elements, some degrees of freedom in rotations are included in the formulation of the interpolation over the element. For volume element, usually, the degrees of freedom of nodes are only in translation but depending on the polynomial interpolation, the number of nodes can vary.

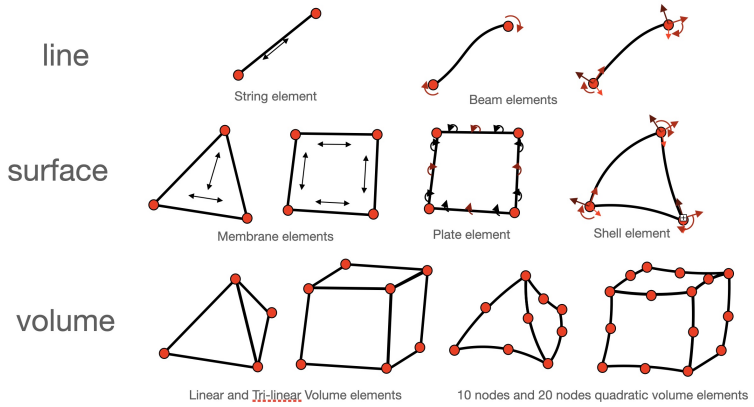


Figure 7: Examples of elements used in FEM

In all cases, these degrees of freedom of the elements are noted in a vector $\mathbf{x}$ in the following.

Interpolation functions are used to describe the behavior of the position (or displacement) and velocity field of all the points inside an element as a function of the positions (or displacements) and velocities of the nodes. These functions are often polynomials but many other approaches exist. By meshing the domain, i.e. the shape of the deformable object (in this case, the robot), we obtain a position and velocity field that is continuous over the entire domain, but whose motion is parameterized by the motion of the nodes. The mesh topology used to perform the FEM calculation will give the connections between the nodes, which are linked by elements. Some methods (called meshless methods) allow to compute these connections without a mesh, by neighborhood. In such case, the support of the interpolation is not defined by an element shape but by the function itself (this is not detailed here, see [20] for more information)

4.3 Internal forces computation and assembling

At the level of each element, we can therefore integrate the laws of mechanics using a variational formulation, integrating the interpolation defined on the element. This formulation enables the computation of the variation of the deformation energy as a function of the movement of the nodes.

The finite element interpolation is defined on the mesh which discretizes the 3D domain $\mathcal{D}$ on which we compute the deformations of the robot (in general the domain $\mathcal{D}$ corresponds to its shape). Let us consider that the mesh connects a set of $\mathcal{N}$ nodes. For any particule $\mathbf{p}$ in the

domain $\mathcal{D}$, the motion is interpolated using the node motion.

$$\mathbf{p}\ (x,y,z) = \sum_{i \in \mathcal{N}} N^i(x,y,z)\ \mathbf{x}^i$$

where $N^i(x,y,z)$ are piecewise polynomials with compact support. These polynomials are associated to nodes and their support is usually defined on one element. The spacial derivatives can now be computed using the interpolation, for instance:

$$\frac{\partial \mathbf{p}}{\partial x} = \sum \frac{\partial N^i(x,y,z)}{\partial x}\ \mathbf{x}^i \quad ; \qquad \frac{\partial \mathbf{p}}{\partial y} = \sum \frac{\partial N^i(x,y,z)}{\partial y}\ \mathbf{x}^i \quad ; \qquad \frac{\partial \mathbf{p}}{\partial z} = \sum \frac{\partial N^i(x,y,z)}{\partial z}\ \mathbf{x}^i$$

With a full Lagrangian approach, interpolations and derivatives are performed in the form at rest, enabling precalculations to be made. Thanks to the fact that the interpolation functions have a compact support, we can calculate the value of the strain and stress tensors element by element. Internal strain energy from equation (15) can be approximated by summing over each elements e:

$$\partial \mathcal{W}(\mathbf{x}) = \int_D \boldsymbol{\sigma}^T \partial \mathbf{E}\, dV \approx \sum \int_e \boldsymbol{\sigma}^T(\mathbf{x}) \partial \mathbf{E}(\mathbf{x})\, dV \tag{15}$$

The internal forces $\mathbf{F}(\mathbf{x}) = \frac{\partial \mathcal{W}(\mathbf{x})}{\partial \mathbf{x}}$ are obtained by the derivative of this deformation potential energy. Again, we can make a parallel with the principle of virtual work, as the work created by the nodes' movements need to be equal to the variation of the strain energy: $\partial \mathcal{W}(\mathbf{x}) = \mathbf{F}(\mathbf{x})^T \partial \mathbf{x}$.

By deriving the expression a second time, the tangent stiffness can be found at the level of each element. But the stiffnesses and internal forces calculated for each element add up at the nodes. We therefore assemble these internal forces and matrices according to the topology and numbering of the nodes. The result is the internal forces and the internal stiffness of the structure. This structural stiffness serves as the basis for calculating compliance (inverse of stiffness).

5 Equations of motion

Based on the FEM (or equivalent) formulation described above, the material behavior law is now integrated across all mesh elements. In this way, at the scale of a deformable robot, the internal forces created by the deformation of the robot structure can be recovered. The "structural" aspect is important, as the mechanical behavior of a soft robot depends on both the material used and the design of its hole structure. For a numerical model of an elastic or hyperelastic robot, the internal forces depend mainly on the position of the nodes, and sometimes on the velocity (viscoelastic case).

Solving the equations of motion on a FEM model of the soft robot involves calculating the positions and velocities of the mesh nodes. Depending on the soft robot and the application, quasi-static or dynamic approaches can be used. Dynamic effects on soft robots, such as vibrations, are often unwanted and avoided. In practice, in particular for medical applications, actuations of soft robots are usually done at low velocities. In such cases, the inertia forces vanish. However if the robot has no fixed point or no rigid base, the dynamic approach is mandatory (for locomotion for instance).

5.1 Quasi-static motion

The conditions to use a quasi-static approach are twofold: (1) some nodes of the FEM model of the robot are fixed or attached to a rigid part (2) the velocities or the mass/stiffness ratio are

sufficiently low so that inertial terms vanish. For instance, at micro scale the mass decreases much faster than stiffness. For micro soft manipulators, even at high speeds, mass can be neglected [16].

The configuration of the robot at a given time is then obtained by solving the static equilibrium between the internal forces computed by the FEM model $\mathbf{F}(\mathbf{x})$ (where $\mathbf{x}$ is the position vector of the FE nodes) and the external loads $\mathbf{F}_{ext}$, including the efforts exerted by the actuators. In some case, we can add the forces created by the gravity $\mathbf{Mg}$. Note that in FEM, using *mass lumping* on volume elements can lead to constant mass matrices that do not depends on the position of the nodes. It is not the case with beam FEM elements for instance.

$$\mathbf{F}(\mathbf{x}) + \mathbf{Mg} + \mathbf{F}_{ext} = \mathbf{0} \tag{16}$$

In the general case, the internal forces are a non-linear function. At each step i of the simulation, we compute a linearization of $\mathbf{F}(\mathbf{x})$ by using a Taylor series expansion, leading to this first order approximation:

$$\mathbf{F}(\mathbf{x}_i) \approx \mathbf{F}(\mathbf{x}_{i-1}) + \frac{\partial \mathbf{F}(\mathbf{x}_{i-1})}{\partial \mathbf{x}} d\mathbf{x} = \mathbf{F}(\mathbf{x}_{i-1}) + \mathbf{K}(\mathbf{x}_{i-1}) d\mathbf{x} \tag{17}$$

Where $d\mathbf{x} = \mathbf{x}_i - \mathbf{x}_{i-1}$ and $\mathbf{K}$ is the tangent stiffness matrix mentioned in section 4.3 that depends on the current position of the FE nodes. This matrix is obtained by assembling the contributions of each element of the FEM model.

Combined with equation 16, it provides a linear matrix system $\mathbf{A}x = \mathbf{b}$ to solve at each time stem:

$$\underbrace{-\mathbf{K}(\mathbf{x}_{i-1})}_{\mathbf{A}} \underbrace{d\mathbf{x}}_{x} = \underbrace{\mathbf{F}(\mathbf{x}_{i-1}) + \mathbf{Mg} + \mathbf{F}_{ext}}_{\mathbf{b}} \tag{18}$$

At each step, the matrix $\mathbf{K}(\mathbf{x}_{i-1})$ is updated with the position of the nodes of the previous step. By solving this linear equation, a new value of $d\mathbf{x}$ can be obtained and the solution converges when $\mathbf{x}_i = \mathbf{x}_{i-1}$. This can be considered as a Newton-Raphson iterative method, in particular if $\mathbf{Mg}$ and $\mathbf{F}_{ext}$ are constant and non dependant on $\mathbf{x}$. In the next section (section 6), we'll look at how to integrate actuator forces, which are not constant given forces but depend on the structure's internal forces.

Note that the matrix $\mathbf{A}$ *is usually highly sparse and positive. However it is rank deficient: the rank of the matrix corresponds* rank $=$ s $-$ 6 *with* s *being the size of the matrix. The 6 redundant equations corresponds to the rigid degrees of freedom. To inverse the matrix, we need to remove (at least) these 6 degrees of freedom. In practice we often fix all the nodes that are attached to a static support. One possible numerical technique is to remove the lines and column corresponding to these nodes from the equilibrium equation, considering that* $d\mathbf{x} = 0$ *for these nodes. An other technique (in particular when the rigid support is moving) is to use Lagrange multipliers, introduced in the following section.*

5.2 Dynamic motion

To model the dynamics of a deformable robot using FEM, we rely on the second law of Newton:

$$\mathbf{M}(\mathbf{x})\mathbf{a} = \mathbf{F}(\mathbf{x}, \mathbf{v}) + \mathbf{M}(\mathbf{x})\mathbf{g} + \mathbf{F}_{ext} \tag{19}$$

The mechanical states is given by $\mathbf{x}$ the position and $\mathbf{v}$ the velocities of the FEM mesh nodes. With the general term $\mathbf{F}(\mathbf{x}, \mathbf{v})$, the computation of the internal forces can also include viscosity effects. The vector $\mathbf{a}$ is the acceleration of these nodes. The mass matrix $\mathbf{M}(\mathbf{x})$ is always positive

definite. The mass may be a function of the node position, but when using mass lumping, the matrix is constant and diagonal. In the following we will use simply $\mathbf{M}$.

The deformable dynamics evolves in time and need to be integrated. For that we use a time numerical scheme. The choice of this scheme has a big influence on the results:

- **With explicit schemes**, at each time step, a new value of the acceleration $\mathbf{a}^+$ can be obtained by inverting the mass matrix. The current position $\mathbf{x}^-$ and velocity $\mathbf{v}^-$ of the nodes are computed using values of the previous steps. After the computation of the new acceleration $\mathbf{a}^+$, the new velocities $\mathbf{v}^+$ and positions $\mathbf{x}^+$ can be obtained by applying the explicit time integration scheme (using a time step h).

$$\left\{\begin{array}{l}\mathbf{a}^+ = \mathbf{M}^{-1}(\mathbf{F}(\mathbf{x}^-, \mathbf{v}^-) + \mathbf{M}\mathbf{g} + \mathbf{F}_{ext}) \\ \mathbf{v}^+ = f(\mathbf{a}^+, \mathbf{v}^-, h) \\ \mathbf{x}^+ = f(\mathbf{a}^+, \mathbf{v}^-, \mathbf{x}^-, h)\end{array}\right. \tag{20}$$

  With explicit scheme, the computation of the non linear internal forces are quite easy to obtain. And combined with mass lumping (with a diagonal matrix $\mathbf{M}$) the computation of a new acceleration is straightforward and can be easily computed in parallel. However, this method is **conditionally stable**: h should be sufficiently small, in particular when the the size of the elements is small (detail mesh) and mass/stiffness ratio is low.

- **With implicit schemes**, the model is **unconditionally stable**, however the system is more difficult to solve because the current position $\mathbf{x}^+$ and velocity $\mathbf{v}^+$ of the nodes used in the dynamic system depends on the current acceleration $\mathbf{a}^+$. A non linear system has to be solved which takes much more time, but the time step h can be chosen freely.

$$\left\{\begin{array}{l}\mathbf{M}\mathbf{a}^+ = \mathbf{F}(\mathbf{x}^+, \mathbf{v}^+) + \mathbf{M}\mathbf{g} + \mathbf{F}_{ext} \\ \mathbf{v}^+ = f(\mathbf{a}^+, \mathbf{v}^-, h) \\ \mathbf{x}^+ = f(\mathbf{a}^+, \mathbf{v}^+, \mathbf{x}^-, h)\end{array}\right. \tag{21}$$

Both strategies have their advantages and drawbacks. In the following, we have developed the method for implicit scheme, in particular to introduce more easily the non-smooth mechanics of contacts, as proposed in [2]. Indeed, when a contact occurs on a FEM node, the velocity of this node instantaneously changes and the acceleration is not defined. We will then use a low order time stepping scheme (backward Euler) which replaces the acceleration by a change of velocity $d\mathbf{v}$, and forces by impulses (forces multiplied by the time step):

$$\left\{\begin{array}{l}\mathbf{M}d\mathbf{v} = h\mathbf{F}(\mathbf{x}_i, \mathbf{v}_i) + h\mathbf{M}\mathbf{g} + h\mathbf{F}_{ext} \\ d\mathbf{v} = \mathbf{v}_i - \mathbf{v}_{i-1} \\ \mathbf{x}_i = \mathbf{x}_{i-1} + h\mathbf{v}_i\end{array}\right. \tag{22}$$

We can used the Taylor series expansion, leading to the following first order approximation:

$$\mathbf{F}(\mathbf{x}_i, \mathbf{v}_i) \approx \mathbf{F}(\mathbf{x}_{i-1}, \mathbf{v}_{i-1}) + \underbrace{\frac{\partial \mathbf{F}}{\partial \mathbf{x}}}_{\mathbf{K}} d\mathbf{x} + \underbrace{\frac{\partial \mathbf{F}}{\partial \mathbf{v}}}_{\mathbf{D}} d\mathbf{v} \tag{23}$$

where $\mathbf{D}$ is the damping matrix. By combining with the time stepping (equation 22), we obtain the following linear system $\mathbf{A}x = \mathbf{b}$ to solve at each time step:

$$\underbrace{\left(\mathbf{M} - h\mathbf{D} - h^2\mathbf{K}\right)}_{\mathbf{A}} \underbrace{d\mathbf{v}}_{x} = \underbrace{h\mathbf{F}(\mathbf{x}_{i-1}, \mathbf{v}_{i-1}) + h^2\mathbf{K}\mathbf{v}_{i-1} + h\mathbf{M}\mathbf{g} + h\mathbf{F}_{ext}}_{\mathbf{b}} \tag{24}$$

Note that here, the matrix $\mathbf{A}$ is always positive definite, thanks to the presence of the mass matrix. The system can be solved even if the robot is not attached to any rigid or static parts.

In both cases (quasi-statics and dynamics), we have introduced the equations that leads to the solving of a sparse linear matrix system $\mathbf{A}x = \mathbf{b}$ with matrix $\mathbf{A}$ and vector $\mathbf{b}$ changing at each step. In the static case, the matrix $\mathbf{A}$ represents the tangent stiffness of the deformable object and in the dynamic case, the matrix $\mathbf{A}$ can be considered as the tangent of the impedance integrated by the numerical scheme.

6 Lagrangian mechanics formulation

As seen in paragraph 3.5, in a weak form (finite element method or other numerical method), the calculation of internal forces due to deformations is derived from an energy potential. This is one of the foundations of Lagrangian mechanics. The advantage of an energetic approach such as Lagrangian mechanics is that different robot motion parameterization spaces can be used, while guaranteeing energetic equivalence. Moreover, in the previous section, to build the linear system solved at each time step, we have considered that the external forces do not depend on the position of the nodes which is not always true, in particular for computing actuator forces or contact forces. We will therefore introduce **Lagrange multipliers** to model these forces. The equation of motion to be solved at each time step will become:

$$\begin{cases} \mathbf{A}x = \mathbf{b} + \mathbf{H}^T\boldsymbol{\lambda} & \text{in the static case} \\ \mathbf{A}x = \mathbf{b} + h\mathbf{H}^T\boldsymbol{\lambda} & \text{in the dynamic case} \end{cases} \tag{25}$$

In the specific case of modeling a deformable robot, we will then distinguish 3 parameterization spaces:

1. **FEM DOFs, node motion or equivalent**: To be able to solve continuum mechanics using a numerical method, the number of degrees of freedom must be sufficiently high for convergence. In FEM, the objective is that by changing the mesh size, the solution found by the numerical method is essentially the same. To achieve this goal, and in particular when we want to achieve convergence on the strain field (which is a derivative of the node displacement field), we will often reduce the size of the elements and considerably increase the size of the model, which can take us far away from the computation times required for use in robotics.

2. **Reduced DOFs, Reduced order model**: The number of degrees of freedom generated by numerical methods is much higher than the usual number of degrees of freedom used in robotics. But the objective is not the same as above. In robotics, the aim is to parameterize all the configurations the robot can take during its use and interaction with its environment, whereas with numerical methods, the aim is convergence. To use the model in robotics, we can therefore project the FEM model into a very reduced subspace of degrees of freedom, by reparameterizing the motion with a reduced number of DOFs. However, care must be taken not to alter the accuracy of the FEM model.

3. **Sub-spaces: actuators, effectors, sensors and external constraints**: The actuation space is very important in robotics: it determines the motions that can be controlled on the robot. Robots are often largely under-actuated: the actuation space is a subspace of all parameterized deforming movements. In the general case of a deformable robot, the actuation space is coupled: an effort on an actuator leads to a movement or a change

in effort on the other actuators. In addition, the robot's movement will be controlled at certain points called "end-effectors", often located at the end of the robot, but sometimes at other points to control the configuration. Here too, we're talking about a subspace of possible movements. Finally, sensors can be placed on the robot to define a subspace of movements that can be measured and external interactions, like contacts, will be defined as constraints on a subpart of the DOFs.

In both cases, the dimensions of the parameterization spaces (2) and (3) are much smaller than (1). But it's important to note that (2) will give a parameter space describing all possible robot movements. We can choose to reduce these possible movements, but the space (2) must always remain larger than the subspaces defined in (3), otherwise hyperstatisms could occur. Moreover, the mapping between the different spaces is not made in the same way. The mapping between the degrees of freedom of the FEM (1) and the reduced DOFs (2) is made by stiffness projection, whereas the mapping between the degrees of freedom of the FEM (1) (or the reduced DOFs(2)) to the subspaces (3) is made by compliance projection. We will describe the two projections in the following subsections

6.1 DOF reduction and stiffness projection

Lagrangian mechanics modeling stipulates that whatever coordinate system is chosen, energy must be conserved. Deformation energy is potential energy. The principle of virtual work can be applied to modify the coordinate system, without affecting the energy. So if we introduce new dofs $\mathbf{q}$ with a mapping function $\mathbb{M}$ such as

$$\mathbf{x} = \mathbb{M}(\mathbf{q}) \text{ and } d\mathbf{x} = \underbrace{\frac{\partial \mathbb{M}}{\partial \mathbf{q}}}_{\mathbf{J}} d\mathbf{q} \tag{26}$$

The work of the forces $\mathbf{F}$ must remain the same with the new coordinate space defined by $\mathbf{q}$. Let $\boldsymbol{\tau}$ be the forces in this space. If the work w is the same in the two coordinates system for the same displacement increment, we have:

$$w = d\mathbf{q}^T \boldsymbol{\tau} = d\mathbf{x}^T \mathbf{F} \Leftrightarrow d\mathbf{q}^T \boldsymbol{\tau} = d\mathbf{q}^T \mathbf{J}^T \mathbf{F} \Leftrightarrow \boldsymbol{\tau} = \mathbf{J}^T \mathbf{F} \tag{27}$$

As in rigid robotics, we use the Jacobian $\mathbf{J}$ and the transposed Jacobian $\mathbf{J}^T$ to project the mechanics into a reduced motion parameter space. Two use cases are now explained to show the interest of such a projection.

- **stiffening**: In the case of hybrid "soft-rigid" modeling, such a projection can be used. When the stiffness of the material is sufficiently high in certain regions, or when the boundary conditions impose rigid movements on certain parts of the mesh, we will consider a subpart of the nodes do not undergo any deformation. This part of the mesh is then considered to have a rigid motion $(\mathbf{x}_r, \mathbf{v}_r)$ and can therefore be parameterized with a transformation (translation and rotation) $(\mathbf{q}_1)$ with only 6 degrees of freedom. If many nodes are *stiffened*, the number of degrees of freedom can be significantly reduced. Of course several independant rigid regions can be modeled. In that case the dimension of $(\mathbf{q}_1)$ is 6 multiplied by the number of independant rigid regions. The motion of the other part of the nodes $(\mathbf{x}_{\overline{r}}, \mathbf{v}_{\overline{r}})$ is still considered unknown. We will therefore define an identity mapping (often including a renumbering) with the space of degrees of freedom $\mathbb{I}(\mathbf{q}_2)$. For this part of the mapping, the number of degrees of freedom does not change.

$$\begin{cases} \mathbf{x}_r = \mathbb{T}(\mathbf{q}_1) & \mathbf{v}_r = \mathbf{J}_\mathbb{T}\dot{\mathbf{q}}_1 \\ \mathbf{x}_{\overline{r}} = \mathbb{I}(\mathbf{q}_2) & \mathbf{v}_{\overline{r}} = \mathbf{I}_\mathbb{I}\dot{\mathbf{q}}_2 \end{cases} \tag{28}$$

When applied on equation 25 in the static case, we obtain the following system to be solved.

$$\begin{bmatrix} \mathbf{J}_\mathbb{T}^T\mathbf{A}_{rr}\mathbf{J}_\mathbb{T} & \mathbf{J}_\mathbb{T}^T\mathbf{A}_{r\overline{r}}\mathbf{I}_\mathbb{I} \\ \mathbf{I}_\mathbb{I}^T\mathbf{A}_{\overline{r}r}\mathbf{J}_\mathbb{T} & \mathbf{I}_\mathbb{I}^T\mathbf{A}_{\overline{r}\overline{r}}\mathbf{I}_\mathbb{I} \end{bmatrix} \begin{bmatrix} d\mathbf{q}_1 \\ d\mathbf{q}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{J}_\mathbb{T}^T \\ \mathbf{I}_\mathbb{I}^T \end{bmatrix} \left(\mathbf{b} + \mathbf{H}^T\boldsymbol{\lambda}\right) \tag{29}$$

The system to be solved is therefore smaller in size, but the matrix projection calculations can add computational cost, so computation time is not necessarily faster than for the full system. On the other hand, this modeling approach is very useful for roboticists, as we find the rigid transformations $\mathbb{T}(\mathbf{q}_1)$ and associated Jacobians $\mathbf{J}_\mathbb{T}$ which are widely used in rigid robotics. On the other hand, FEM models with strong heterogeneities (with very steep and very soft regions) often lead to problems of poor conditioning. The approach outlined here helps to overcome them.

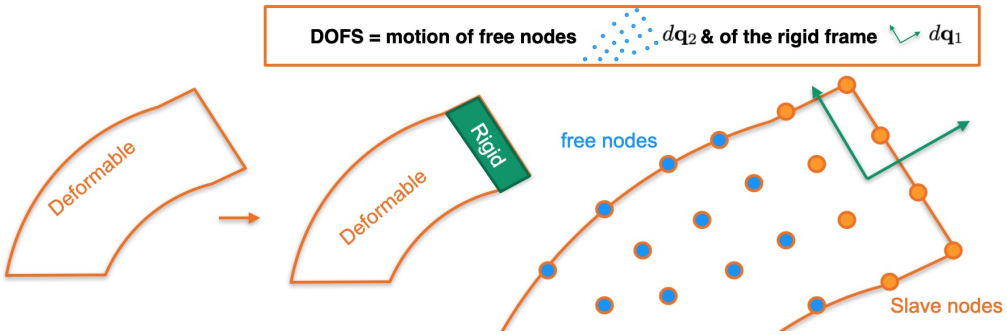


Figure 8: Modeling the stiffening of a part of a deformable model through reduction of DOFs

In dynamics, note that for high-speed movements, the $\mathbf{J}_\mathbb{T}(\mathbf{q}_1)$ matrix is non-linear and some derivative terms of the form $\dot{\mathbf{J}}_\mathbb{T}\dot{\mathbf{q}}_1$ will appear when calculating acceleration. Same, for very strong external forces: it could be useful sometimes to add the apparent stiffness (created by the non-linearity of the mapping) $\frac{\partial \mathbf{J}_\mathbb{T}(\mathbf{q}_1)\mathbf{F}_{ext}}{\partial \mathbf{q}_1}$ in the matrix $\mathbf{A}$. For the moment, these terms have not yet been explored to any deep extent. For the sake of brevity, we won't elaborate on these points here.

- **Model order reduction**: To overcome the difficulty of quickly calculating the full FEM model, one possible approach is to reduce the size of the model by projecting the equations from the FEM into a reduced space. This technique also allows to obtain a dynamic reduced model. In engineering tools, a well known projection technique is the modal analysis. After calculating the structure in FEM, the main deformation modes are calculated and the model is projected in this space. We can thus define the size of the model according to the frequency range of vibration we are interested in (generally, the FEM model generates many numerical vibrations of very high frequency, which can be assimilated to numerical noise). The modal analysis is often limited to linear deformations but can be extended to non-linear [3].

  Other techniques, such as POD, allow to compute the principal modes of deformations according to a data set pre-computed from an SVD decomposition. This technique first

requires a calculation phase on the complete model. But then, the model motion can be projected through Φ in a reduced space α:

$$\Phi^T\mathbf{M}\Phi\ddot{\alpha} = \Phi^T\mathbf{F}(\Phi\alpha) + \Phi^T\mathbf{M}\mathbf{g} + \Phi^T\mathbf{F}_e xt + \Phi^T\mathbf{H}^T\boldsymbol{\lambda}$$

The approach is compatible with non-linear model and the approach allows to dramatically reduce the number of DOFs of the models (typically a space of 50 modes can be sufficient for a complex robot) [12] By further reducing the state space of the system (by tolerating larger modeling errors) state observers can be introduced and perform dynamic control [26].

After integration of the dynamics using equation 24 or using the quasi-static assumptions, the reduced system to be solved at each step is:

$$\begin{cases} \Phi^T\mathbf{A}\Phi d\alpha = \Phi^T\mathbf{b}(\Phi\alpha) + \mathbf{H}^T\boldsymbol{\lambda} & \text{in the static case} \\ \Phi^T\mathbf{A}\Phi d\alpha = \Phi^T\mathbf{b}(\Phi\alpha) + h\mathbf{H}^T\boldsymbol{\lambda} & \text{in the dynamic case} \end{cases} \tag{30}$$

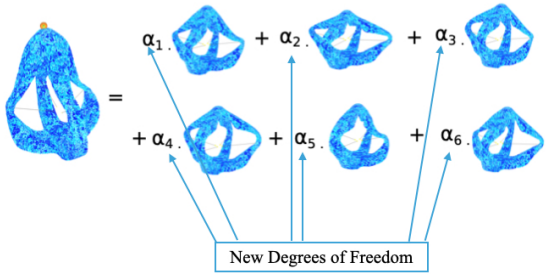


Figure 9: Reduction of the degrees of freedom: In this figure, we suppose that the motion of the soft robot can be captured by a linear combination of 6 modes. The intensity of these modes are the new degrees of freedom

6.2 Compliance projection

The compliance projection is based on condensation. This technique reduces the equations of equilibrium and motion on the loaded degrees of freedom. The unloaded degrees of freedom then become slaves. In general, on soft robots, actuation is not defined in a single node. Therefore, one can condense by adding Lagrangian constraints to the system. Let's suppose a robot made with one cavity and we load the structure of the robot by changing the volume $\boldsymbol{\delta}_v(\mathbf{x})$ of this cavity (for instance with pressured air or water). This volume depends on the position of the nodes of the mesh that are on the surface of the cavity. We can add the constraint $\boldsymbol{\delta}_v(\mathbf{x}) = v_{\text{input}}$ to load the structure and deform the robot.

By solving this non-linear constraint system, a pressure will be applied on these nodes, so only a sub-part of the nodes of the robot model will be loaded. In practice, to solve the system, the model is linearized around the current position $\mathbf{x}$ and a Lagrange multiplier $\boldsymbol{\lambda}$ is introduced to replace the external loads $\mathbf{R}$ and compute the pressure:

$$\begin{cases} \mathbf{F}(\mathbf{x}+\Delta\mathbf{x}) + \mathbf{M}\mathbf{g} + \mathbf{H}^T\boldsymbol{\lambda} & \approx & \mathbf{F}(\mathbf{x}) + \mathbf{K}(\mathbf{x})d\mathbf{x} + \mathbf{M}\mathbf{g} + \mathbf{H}^T\boldsymbol{\lambda} = \mathbf{0} \\ \boldsymbol{\delta}_v(\mathbf{x}+\Delta\mathbf{x}) & \approx & \boldsymbol{\delta}_v(\mathbf{x}) + \mathbf{H}d\mathbf{x} = v_{\text{input}} \end{cases}$$

with $\mathbf{K}(\mathbf{x}) = \frac{d\mathbf{F}}{d\mathbf{x}}$ and $\mathbf{H} = \frac{d\boldsymbol{\delta}_v}{d\mathbf{x}}$. At equilibrium, we can obtain the instantaneous relationship between the change of pressure $\boldsymbol{\lambda} \rightarrow \boldsymbol{\lambda} + \Delta\boldsymbol{\lambda}$ and the change of volume as a single equation by condensation of the mechanical system:

$$\Delta\boldsymbol{\delta}_v = \boldsymbol{\delta}_v(\mathbf{x} + d\mathbf{x}) - \boldsymbol{\delta}_v(\mathbf{x}) = \underbrace{\mathbf{H}\mathbf{K}(\mathbf{x})^{-1}\mathbf{H}^T}_{\mathbf{W}_{aa}} \Delta\boldsymbol{\lambda} \tag{31}$$

With $\mathbf{W}_{aa}$ a projected compliance in the actuator space. With several actuators, this matrix provide the mechanical coupling between actuators. By playing with the definition of functions δ (we can define δ_a for the actuators, δ_e for the end-effectors and δ_s for the sensors), we can obtain the coupling between actuators and end-effectors ($\mathbf{W}_{ea}$) or between sensors and end-effectors ($\mathbf{W}_{se}$). Kinematic models of the robot can be derived from this:

$$\Delta\delta_e = \underbrace{\mathbf{W}_{ea}\mathbf{W}_{aa}^{-1}}_{\mathbf{J}} \Delta\delta_a$$

where $\mathbf{J}$ is the jacobian of the soft robot.

For inverse kinematics, one can, by the optimisation $min(\boldsymbol{\delta}_e)_{\boldsymbol{\lambda}_a}$ compute the inverse model by minimizing the distance between the end-effector position and an objective position, by actuating the structure through $\boldsymbol{\lambda}_a$. The same strategy can be used to measure a force at the end-effector position $\boldsymbol{\lambda}_e$, by minimizing the distance between the sensor values in the model and in real-life $min(\boldsymbol{\delta}_s)_{\boldsymbol{\lambda}_e}$.

Note that the problem being non-linear, the condensation requires the computation and the factorization of $\mathbf{K}(\mathbf{x})$ which is a very large matrix, in order to obtain $\mathbf{W}$. More details will be provided in the next section.

But at this point, we'd like to remind you of the different spaces in which the equations of deformable motion are written, and more specifically here, the equations of deformable robots. This is illustated on figure 10 First, there's the behavior of materials that are subjected to deformation. To characterize motion and effort in materials, we have used Green-Lagrange strain $\mathbf{E}$ and stress $\boldsymbol{\sigma}$. Then we have interpolated and integrated the deformations over the shape of the robot thanks to Finite Element Method. The motion is discretized at the level of the nodes, thanks to their position $\mathbf{x}$ and the internal forces $\mathbf{F}(\mathbf{x})$. After that we have projected the mechanics in a subspace and we have defined reduced dofs $\mathbf{q}$ and efforts $\boldsymbol{\tau}$. Finally, we did the projection in the actuator and effector spaces with $\boldsymbol{\delta}$ and $\boldsymbol{\lambda}$. In all these changes of space, there's a guarantee of energy equivalence, with the principle of virtual work.

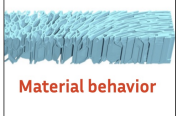
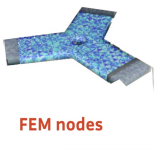
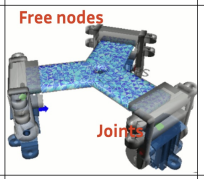
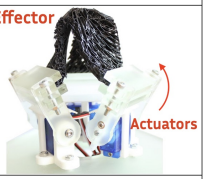
Space	*Constitutive law*	*Finite Element Model*	*DOF Reduction*	*Compliance Projection*
Illustration				
Motion / Effort	$\mathbf{E}$ / $\boldsymbol{\sigma}$	$\mathbf{x}$ / $\mathbf{F}$	$\mathbf{q}$ / $\boldsymbol{\tau}$	$\boldsymbol{\delta}$ / $\boldsymbol{\lambda}$

Figure 10: The different spaces in which the deformable mechanics of soft robots have been described.

7 Actuator models

Having described how actuator space is introduced in the form of a Lagrange multiplier, we'll now be able to describe in more detail the meaning of this multiplier for three different types of actuation.

7.1 Pressure based Actuation

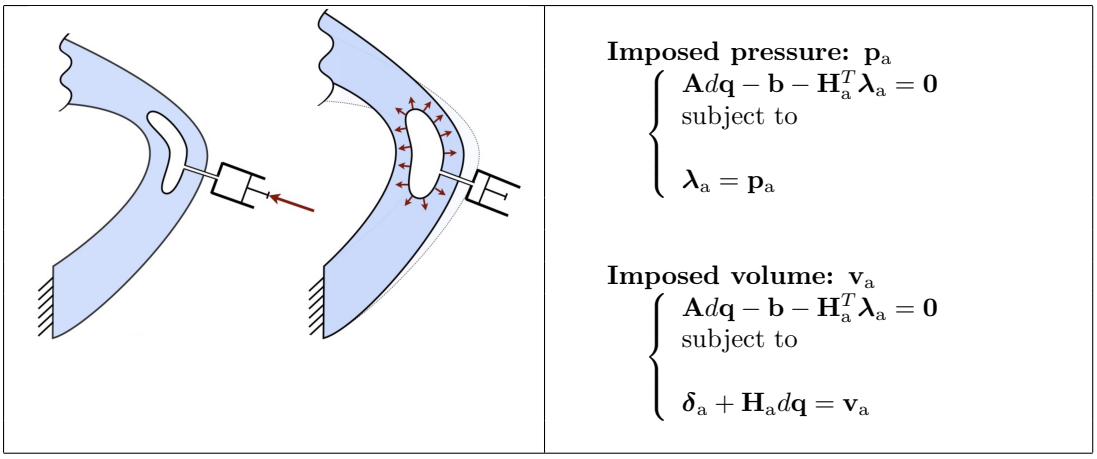


Figure 11: Pressure based actuator

The pressure inside the cavity $\boldsymbol{\lambda}_a$ is distributed as a force on the nodes placed at the cavity surface thanks to the matrix $\mathbf{H}_a^T$. When the pressure is imposed, the value of the Lagrange multiplier is known ($\boldsymbol{\lambda}_a = \mathbf{p}_a$). This is often the case with pneumatic actuators. But when using hydraulic actuation, the volume in the cavity is imposed and the pressure $\boldsymbol{\lambda}_a$ is unknown. The Lagrange multiplier is solved by adding a new equation which forces the local linearization of the volume $\boldsymbol{\delta}_a$ computed on the FEM mesh to follow the imposed volume $\mathbf{v}_a$. The figure 7.1 presents the systems of equations to be solved in both cases of imposed pressure or imposed volume. Note that in the case of liquid, if the cavity is big, the weight of the liquid could also influence the behavior [23] .

7.2 Tendon based Actuation

In the tendon case (see figure 12), we assume that the cable is infinitely stiff and has no internal bending and torsion forces. In such case, we can use a geometrical representation of the cable (described below). In [1], a deformable rod model of the tendon is coupled with a FEM model of the robot, but this approach is not detailed here.

In the case of a geometric model of the cable, the Lagrange multiplier $\boldsymbol{\lambda}_a$ is the tensile force inside the cable. This force is distributed over the structure according to the cable path. Each time the cable path is inflected (deviating from the straight line), tangential forces appear, proportional to the tensile force. We can therefore find a matrix $\mathbf{H}_a$ which distributes the force over the nodes of the mesh (nodes 2,3,4 on the figure). Note that if this cable path does not correspond to a series of segments and nodes in the FEM mesh, then the matrix $\mathbf{H}_a$ also contains interpolation values of the points defining the cable with respect to the mesh nodes.

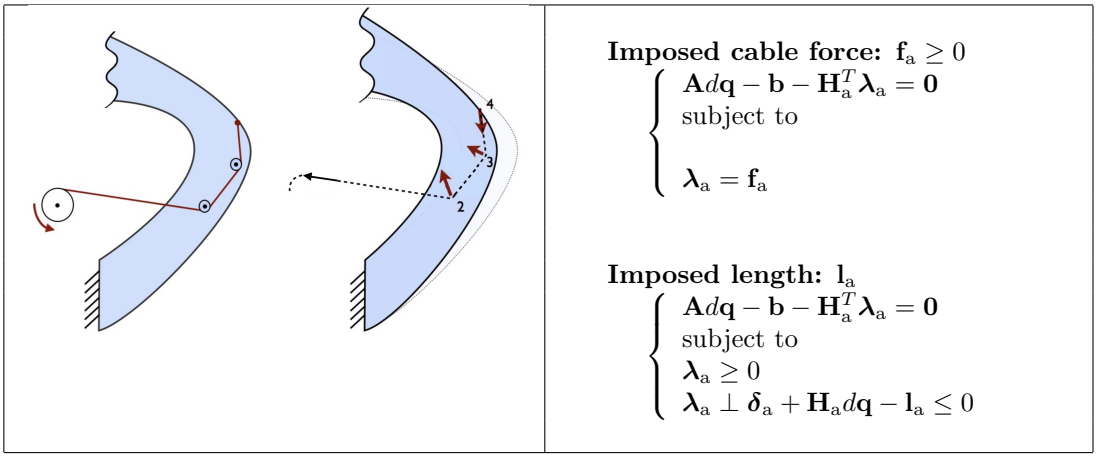


Figure 12: Tendon (Cable) actuator

When the force $\mathbf{f}_a$ is known and imposed, it must be positive (one can only pull on a cable, not push) and in that case the value of the Lagrange multiplier $\boldsymbol{\lambda}_a$ is known. When the cable is controlled in position, we can impose the length of the cable $\mathbf{l}_a$... only when the cable is taut. We thus have to solve the problem with two inequalities: the force is positive and the maximum length of the cable is the imposed length (the length of the cable is linearized given the current configuration of the robot). Between this two inequalities, there is a complementarity: the force is strictly positive ($\boldsymbol{\lambda}_a > 0$) only when the cable is taut ($\boldsymbol{\delta}_a + \mathbf{H}_a d\mathbf{q} - \mathbf{l}_a = 0$) and $\boldsymbol{\lambda}_a = 0$ when the cable is loose ($\boldsymbol{\delta}_a + \mathbf{H}_a d\mathbf{q} - \mathbf{l}_a < 0$).

7.3 Direct actuation with stiffness projection

In section 6.1, we've seen how to model the absolute stiffening (no more deformation) of a part of the robot. This stiffening is often required to actuate the deformation with motors directly connected to these rigid parts. The most common case is probably that illustrated in figure 13, where a motor drives the rotation of part of the deformable model's nodes. It's also common to have the same thing happen, but with a translating motor. We can also find this in more specific cases, such as (rigid) magnets placed in a deformable robot driven by a magnetic field. In all these cases, there is a stiffening combined with an actuation placed on these rigid zones.

As described in equation 29, we will reduce the degrees of freedom of the deformable model by separating the nodes placed on the rigid part r and the other nodes $\overline{r}$. In the case of figure 6.1, the degree of freedom of the rigid part is directly given by the change of angle and denoted $d\boldsymbol{\theta}_1$ (subscript 1 is used to keep the similarity with equation 29). The motion of all these nodes placed on the rigid boundary can be parameterized by this angle. The other nodes keep their degrees of freedom, which are set in $d\mathbf{q}_2$.

It can be seen that when torque is applied to the motors, $\boldsymbol{\lambda}_a = \boldsymbol{\tau}_a$, the torque is applied at the rotating rigid degree of freedom. Thus, the expression of the matrix $\mathbf{H}_a^T$ gives the identity on the first line (which concerns the expression of the forces of the rigid part) and 0 on the second (which concerns the forces on the nodes remaining free).

Like the previous actuators, the motion can be imposed. In the example, a rotation angle $\boldsymbol{\theta}_a$ can be imposed by the motor (if controlled in position). In such case, the value of the torque $\boldsymbol{\lambda}_a$ is unknown and depends on the overall equilibrium of the robot structure. A new equation is thus define to solve the Lagrange multiplier, which corresponds to impose this angle. In such

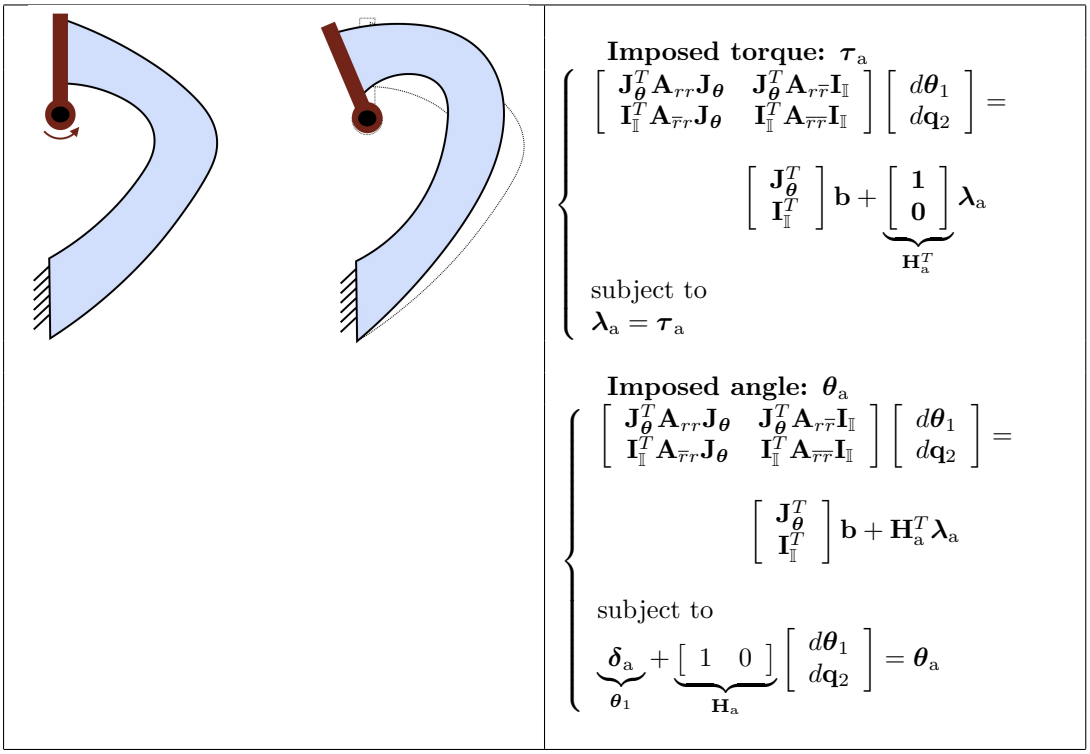


Figure 13: Direct actuation on rigid part

case, the constrain equation is very intuitive: $\boldsymbol{\delta}_\text{a}$ corresponds to the value of the angle $\boldsymbol{\theta}_1$ at the beginning of the step and the variation $d\boldsymbol{\theta}_1$ should be found so that $\boldsymbol{\theta}_1 + d\boldsymbol{\theta}_1 = \boldsymbol{\theta}_\text{a}$.

8 Compliance based models of soft robots

In the previous section, the models of 3 types of actuator used in soft robotics are presented. In each case, we have used of a Lagrange multiplier $\boldsymbol{\lambda}_\text{a}$ to obtain the actuator force, a matrix $\mathbf{H}_\text{a}$ which distributes the actuator forces over the deformable system, and its transpose, the matrix $\mathbf{H}_\text{a}$ which projects the motion in the dof space into the actuator space. In the previous chapter, we saw that in such a case, we could use the compliance projection of the deformable robot model (see equation 31). These two aspects will be combined here to describe the kinematics of the soft robot and, above all, to calculate its inverse model very efficiently, in real time using the projected compliance.

This section shows the end result of the entire modeling process in the form of projected compliance, introduced at the beginning of the chapter. As we said at the beginning of the chapter (and even in the title!), we believe this operator of projected compliance is key to the modeling of soft robots. The following paragraphs are therefore essential to understand how to derive a truly generic model of soft robot.

8.1 Effector model and projected compliance

We call *effectors*, some degrees of freedom of the soft robot, that we wish to control. As said previously, soft-robot are usually underactuated so, we can only control a subpart of the degrees of freedom. As for actuators, a matrix $\mathbf{H}_\text{e}$ will be defined for the effectors. This matrix allows to obtain the motion of the effector $\boldsymbol{\Delta\delta}_\text{e}$ given the motion of the degrees of freedom of the FEM model.

$$\boldsymbol{\Delta\delta}_\text{e} = \mathbf{H}_\text{e} d\mathbf{q} \tag{32}$$

In practice, we often define some nodes of the mesh as the effectors so that:

$$\mathbf{H}_\text{e} = \begin{bmatrix} 0 & \cdots & 0 & \mathbf{I} & 0 & \cdots & 0 \end{bmatrix}$$

where $\mathbf{I}$ is the identity matrix of the dimension of the selected node (3 for a position node, 6 for a position and orientation node, like for beam and shell elements see subsection 4.2). Sometimes it is interesting to define several effector points and/or to control particular directions (x, y but not z for exemple). The principle remains the same, it only changes the size of $\boldsymbol{\Delta\delta}_\text{e}$ and the identity matrix $\mathbf{I}$. A Lagrange multiplier $\boldsymbol{\lambda}_\text{e}$ and the matrix $\mathbf{H}_\text{e}^T$ can be also defined to apply a load on the effectors. If no load is applied on the effector point(s), then $\boldsymbol{\lambda}_\text{e} = 0$.

In some cases, such as locomotion for instance, the effector can be defined as the center of mass(or the barycenter of the nodes). In these cases, $\boldsymbol{\Delta\delta}_\text{e}$ provides the instantaneous displacement of the center of mass of the robot. The jacobian matrix is the same for all nodes i and is given by:

$$\textit{Center of mass}: \ \mathbf{H}_\text{e}^i = \frac{1}{w} \quad \textit{Barycenter}: \ \mathbf{H}_\text{e}^i = \frac{1}{\#nodes}$$

with $\#nodes$ the total number of nodes and w the mass of the object.

Thanks to the compliance projection presented in equation 31, we can obtain the mechanical coupling in the effector and actuator spaces. Starting from a given configuration $\mathbf{A}d\mathbf{q} - \mathbf{b} - \mathbf{H}_\text{a}^T\boldsymbol{\lambda}_\text{a} = \mathbf{0}$ and introducing $\Delta\boldsymbol{\lambda}_\text{a}$ a small change of actuation force, the motion created in the effector and actuator spaces can be computed by

$$\boldsymbol{\Delta\delta}_\text{e} = \underbrace{\mathbf{H}_\text{e}\mathbf{A}^{-1}\mathbf{H}_\text{a}^T}_{\mathbf{W}_\text{ea}}\boldsymbol{\Delta\lambda}_\text{a} \quad \textit{and} \quad \boldsymbol{\Delta\delta}_\text{a} = \underbrace{\mathbf{H}_\text{a}\mathbf{A}^{-1}\mathbf{H}_\text{a}^T}_{\mathbf{W}_\text{aa}}\boldsymbol{\Delta\lambda}_\text{a}$$

with $\mathbf{W}_\text{ea}$ and $\mathbf{W}_\text{aa}$ the smallest possible operators that provides respectively the coupling by the computed compliance of the robot between actuators and effectors and between actuators.

8.2 Kinematics and inverse model by optimization

As explained in figure 2, thanks to $\mathbf{W}_\text{ea}$ and $\mathbf{W}_\text{aa}$, we can compute the local kinematics of the soft robot, i.e, the jacobian between motion in actuator space and motion in effector space.

$$\mathbf{J}_\text{sr} = \mathbf{W}_\text{ea}\mathbf{W}_\text{aa}^{-1} \quad \textit{and} \quad \boldsymbol{\Delta\delta}_\text{e} = \mathbf{J}_\text{sr}\boldsymbol{\Delta\delta}_\text{a}$$

It has to be emphasized that the values of $\mathbf{W}_\text{ea}(\mathbf{q})$ and $\mathbf{W}_\text{aa}(\mathbf{q})$ are not constant in general and depends on the configuration $\mathbf{q}$ of the robot, as the matrices $\mathbf{A}(\mathbf{q})$ and $\mathbf{H}(\mathbf{q})$ depends on $\mathbf{q}$. The computation of matrices $\mathbf{W}$ can be challenging to compute in real-time, in particular if the size of the matrix $\mathbf{A}$ is big (i.e. a dense FEM mesh). However, the matrix $\mathbf{A}$ is highly sparse and always positive definite. Some very efficient solvers based on the factorization of matrix $\mathbf{A}$ are available in packages for linear algebra like Eigen [13], Csparse [6] and others. Moreover, the computation of this matrix $\mathbf{W}$ in real-time has been optimized in different cases, like for slender structure

(continuum robots) [8] or for corotational case [24] with an approximation. Recent techniques, based on GPUs, have been developed to compute in few ms a very good approximation of the matrix in dense contact cases (with a matrix $\mathbf{W}$ involving more than 200 constraints on a meshes with more than 5000 nodes !) [29].

Inverse model of the soft robot can be computed by pseudo-inverse of the $\mathbf{J}_{\text{sr}}$ but quadratic optimisation is a better option to introduce some bounds and deal with the redundancy of the robots. In practice, let's provide a desired position $\mathbf{p}_{\text{e}}$ to the effector. We suppose that the previous step of the simulation provides a position $\mathbf{q}_0$ of the deformable model and $\boldsymbol{\delta}_{\text{e}}(\mathbf{q}_0)$ provides the current position of the effector. We thus would like to modify the actuation of the robot (i.e. find $\boldsymbol{\Delta\lambda}_{\text{a}}$) so that the effector moves $\boldsymbol{\Delta\delta}_{\text{e}}$ to minimize the distance $(\boldsymbol{\delta}_{\text{e}}(\mathbf{q}_0) + \boldsymbol{\Delta\delta}_{\text{e}} - \mathbf{p}_{\text{e}})^2$. Introducing $\boldsymbol{\delta}_{\text{e0}} = \boldsymbol{\delta}_{\text{e}}(\mathbf{q}_0) - \mathbf{p}_{\text{e}}$ the violation of the desired motion at the beginning of the control step, we can write the following optimisation problem to find the actuation:

$$\begin{cases} \min\limits_{\boldsymbol{\Delta\lambda}_{\text{a}}} (\boldsymbol{\delta}_{\text{e0}} + \boldsymbol{\Delta\delta}_{\text{e}})^2 \text{ (with } \boldsymbol{\Delta\delta}_{\text{e}} = \mathbf{W}_{\text{ea}}\boldsymbol{\Delta\lambda}_{\text{a}}) \\ \text{subject to} \\ \boldsymbol{\delta}_{\min} \leq \boldsymbol{\delta}_{\text{a0}} + \boldsymbol{\Delta\delta}_{\text{a}} \leq \boldsymbol{\delta}_{\max} \text{ (with } \boldsymbol{\Delta\delta}_{\text{a}} = \mathbf{W}_{\text{aa}}\boldsymbol{\Delta\lambda}_{\text{a}}) \\ \text{(if cable) } \boldsymbol{\lambda}_{\text{a}} + \boldsymbol{\Delta\lambda}_{\text{a}} \geq 0 \end{cases}$$

where $\boldsymbol{\delta}_{\min}$ and $\boldsymbol{\delta}_{\max}$ are the actuating stops which define the limits of the actuators movements. An other constraint on the maximum speed of the actuator $-\boldsymbol{\Delta}_{\max} \leq \boldsymbol{\Delta\delta}_{\text{a}} \leq \boldsymbol{\Delta}_{\max}$ can also be defined. This optimization problem falls into the category of QP problems for which numerous solvers are also available.

The matrix of the QP is $\mathbf{Q} = \mathbf{W}_{\text{ea}}^T\mathbf{W}_{\text{ea}}$ which is symmetric positive by construction, however it is not always fully definite (convex problem). Indeed, when the number of actuators is greater than the size of the effector space, multiple configuration can lead to a minimum. In practice, there is redundancy in the actuation and several solution are possible. Some QP algorithms are able to find one solution among all possible, however, the solution may oscilate between possible solutions.

To have a stable behavior, we add to the objective function a minimization of the mechanical work of the actuator force $w_{\text{a}} = \boldsymbol{\Delta\delta}_{\text{a}}^T(\boldsymbol{\lambda}_{\text{a}} + \boldsymbol{\Delta\lambda}_{\text{a}}) = \boldsymbol{\Delta\delta}_{\text{a}}^T\boldsymbol{\lambda}_{\text{a}} + \boldsymbol{\Delta\lambda}_{\text{a}}^T\mathbf{W}_{\text{aa}}^T\boldsymbol{\Delta\lambda}_{\text{a}}$. Matrix $\mathbf{W}_{\text{aa}}$ being symmetric definite and positive, $\epsilon\mathbf{W}_{\text{aa}}$ can be added in the QP matrix $\mathbf{Q} = \mathbf{W}_{\text{ea}}^T\mathbf{W}_{\text{ea}} + \epsilon\mathbf{W}_{\text{aa}}$ with a sufficiently small coefficent ϵ to regularize the problem while keeping a precise solution.

8.3 Sensor model and external forces measurement

In the same way as we defined the actuators and effectors on soft robots, we can extend the formulation to the extrinsic or intrinsic sensors that can be placed on the robot. Let's imagine for instance, a camera system with marker points measuring the position of points on the robot (extrinsic) and a sensor measuring curvature on a robot (intrinsic). In the first case the measure of the position sensor is a selection of node position on the robot mesh $\boldsymbol{\delta}_{\text{s}} = \mathbf{H}_{\text{s}}\mathbf{q}$ with $\mathbf{H}_{\text{s}} = \begin{bmatrix} 0 & \cdots & 0 & \mathbf{I} & 0 & \cdots & 0 \end{bmatrix}$.

In the second case, the output of the sensor is a continuous non-linear function of the node position $\boldsymbol{\delta}_{\text{s}}(\mathbf{q}) = Curv(\mathbf{q})$ which provides the local curvature measured by the sensor. On a given position a first order approximation of this function can be computed using $\boldsymbol{\delta}_{\text{s}}(\mathbf{q} + d\mathbf{q}) \approx \boldsymbol{\delta}_{\text{s}}(\mathbf{q}) + \mathbf{H}_{\text{s}}d\mathbf{q}$. This formulation is then very generic. In [15] a flex sensor is used and $\boldsymbol{\delta}_{\text{s}}$ is found by fitting a function on data collected from the simulation and sensors during a defined motor trajectory phase.

To estimate the shape of the robot, we suppose that we have a vector of measured quantities $\boldsymbol{\delta}_s^m$. When some external forces are applied at on effectors, we can introduce a optimization

problem to estimate these external forces $\boldsymbol{\lambda}_\mathrm{e}$:

$$\begin{cases} \min_{\boldsymbol{\Delta\lambda}_\mathrm{e}} (\boldsymbol{\delta}_\mathrm{s}(\mathbf{q}) + \mathbf{H}_\mathrm{s} d\mathbf{q} - \boldsymbol{\delta}_s^m)^2 \\ \text{with} \\ \boldsymbol{\Delta\delta}_\mathrm{s} = \mathbf{H}_\mathrm{s} d\mathbf{q} = \mathbf{W}_\mathrm{se} \boldsymbol{\Delta\lambda}_\mathrm{e} \end{cases}$$

If more sensors are used, it can also estimate the errors on the actuator forces based on the model:

$$\begin{cases} \min_{\boldsymbol{\Delta\lambda}_\mathrm{e}, \boldsymbol{\Delta\lambda}_\mathrm{a}} (\boldsymbol{\delta}_\mathrm{s}(\mathbf{q}) + \mathbf{H}_\mathrm{s} d\mathbf{q} - \boldsymbol{\delta}_s^m)^2 \\ \text{with} \\ \boldsymbol{\Delta\delta}_\mathrm{s} = \mathbf{H}_\mathrm{s} d\mathbf{q} = \mathbf{W}_\mathrm{se} \boldsymbol{\Delta\lambda}_\mathrm{e} + \mathbf{W}_\mathrm{sa} \boldsymbol{\Delta\lambda}_\mathrm{e} \end{cases}$$

For this inverse problem to be solved under the good conditions (i.e. with a unique solution), the number of independent measurements on the sensors must be greater than the number of unknowns on the effectors and actuators. The rank (and the conditioning) of matrices $\mathbf{W}_\mathrm{se}$ and $\mathbf{W}_\mathrm{sa}$ are also key to obtain a full rank matrix in the Quadratic Programming optimisation that are used to solved this optimisation. For more details on the sensors see [30] or [19]. Such approach can be used in closed look control strategies [4].

9 Contact interaction with the environment

Most of today's rigid manipulators are designed to have the greatest possible rigidity between their base and end-effector. It is this rigidity (high stiffness) that gives them good positioning accuracy and repeatability in space. One of the consequences of this design is to limit interaction and contact with the environment. Numerous algorithms have been developed for anti-collision and obstacle avoidance, to prevent any interaction with the environment outside the end-effector. On current collaborative robots, if a contact or collision is detected on the robot's body, the control law provides for the robot to stop its movement, notably for safety reasons.

On humanoid robots, contact are considered for trajectory optimisation, but they are limited to robot's feet and hands, all other mechanical contacts with the environment are usually avoided. Compliant commands implemented on some collaborative robots can be used to define a target force applied to a contact on the robot's end-effector, sometimes in certain directions. But here again, these commands are limited to contacts on the end effector of the robot.

The specificity of soft robots is their high mechanical compliance, enabling them to come into contact with their direct environment over part of their body. This ability to come into contact also changes the way model are conceived. Contact modeling is an integral part of modeling this type of robot. In this section, we discuss the general principles of modeling these contacts, which remains an active research topic.

9.1 Contact model

The first step is to determine the points of contact between the robot model and the model of its environment. Collision detection may appear simple from a geometric point of view, but it is complex from a computational point of view. Indeed, the natural complexity is quadratic: it would consist in testing the set of geometric primitives (often triangles) of the robot model with the set of primitives of the environment model. This quadratic complexity can rapidly become the bottleneck for the computation of the contacting models. This is why many works have proposed algorithmic solutions to compute efficiently the collision detection between deformable objects [25].

Once contact has been detected, one needs a behavioral law to describe the physics of contact. In the context of continuum mechanics, we use Signorini's law to describe the normal part of the contact using a complementarity constraint. This law provides a macroscopic view of contact, consistent with continuum mechanics and multi-contact cases [14]. In contact zones, at each point, a signed distance $\delta_c \geq 0$ is defined between the two solids in contact, which is normal to the contact. This distance must remain positive or zero to prevent interpenetration of the solids (which would be not physical). Furthermore, Signorini's law defines the force (or stress) $\lambda_c \geq 0$ along the same normal direction to the contact surface. This force is also positive or zero to mean that contact can repel objects but cannot attract them to each other (excluding normal adhesion or bonding between objects). Finally, there is complementarity between two cases: (1) contact where the distance is zero at the point of contact and the repulsive force is non-zero $\delta_c = 0$ and $\lambda_c > 0$ (2) non-contact where the distance is strictly positive at the point of contact and there can be no repulsive force $\delta_c > 0$ and $\lambda_c = 0$. To deal with the multi-contact case, the distances and forces at contact are gathered in vector form with $\boldsymbol{\delta}_c$ and $\boldsymbol{\lambda}_c$ respectively. Complementarity means that the vector product between these two vectors is zero, so they are perpendicular. Signorini's law is written:

$$0 \leq \boldsymbol{\delta}_c \perp \boldsymbol{\lambda}_c \geq 0 \tag{33}$$

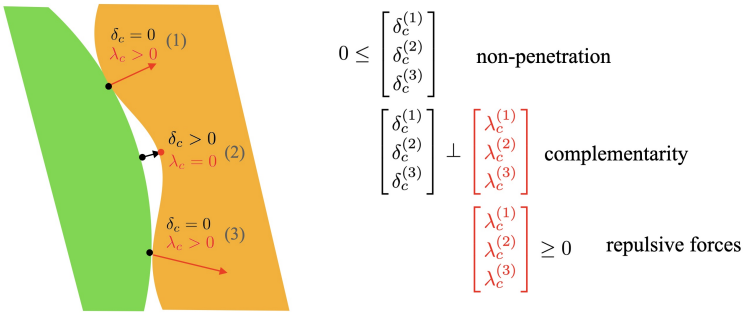


Figure 14: Signorini's unilateral law for multicontact case: the three principles of this law are (i) to prevent penetration at contact points, (ii) to have a complementarity between distance and contact force and (iii) to impose repulsive forces. The drawn forces are applied to the yellow object but opposite forces are applied to the green solid.

In 3D, at each point of contact, the two directions tangent to the surface form a plane in which the instantaneous sliding motion at contact is defined. It is therefore in this plane that we can define a law of friction, such as Coulomb friction. Friction is also defined by the complementary constraint between two states: (1) adhesion (or dry friction) which corresponds to the total suppression of tangent sliding and a force strictly included in the Coulomb cone and (2) sliding (or dynamic friction) which corresponds to the case where the relative motion is non-zero in the tangent plane and opposes (in direction) the force that is strictly on the surface of the Coulomb cone. However, so as not to unnecessarily complicate the remaining of the chapter, we'll consider the frictionless case. To add Coulomb friction, please refer to [9] and [7].

9.2 Direct Solving process using compliance projection

In section 7, to introduce actuation in the deformable mechanical models of robots, we have used Lagrange multipliers and a constraint based formulation. Contact will provide additional

constraints and complementarity conditions. Moreover, the model of the environment will provide additional degrees of freedom, leading to the following constraint problem:

$$\text{solve} \quad \begin{cases} \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_e \end{bmatrix} \begin{bmatrix} d\mathbf{q} \\ d\mathbf{q}_e \end{bmatrix} - \begin{bmatrix} \mathbf{b} \\ \mathbf{b}_e \end{bmatrix} - \begin{bmatrix} \mathbf{H}_{\text{a}}^T \boldsymbol{\lambda}_{\text{a}} + \mathbf{H}_{\text{c}}^T \boldsymbol{\lambda}_{\text{c}} \\ \overline{\mathbf{H}}_{\text{c}}^T \boldsymbol{\lambda}_{\text{c}} \end{bmatrix} = \mathbf{0} \\ \text{with} \\ \\ \boldsymbol{\delta}_{\text{a}} + \mathbf{H}_{\text{a}} d\mathbf{q} = \mathbf{u} \\ 0 \leq \boldsymbol{\delta}_{\text{c}} + \mathbf{H}_{\text{c}} d\mathbf{q} + \overline{\mathbf{H}}_{\text{c}} d\mathbf{q}_e \perp \boldsymbol{\lambda}_{\text{c}} \geq 0 \end{cases}$$

With $\mathbf{A}_e$, $d\mathbf{q}_e$ and $\mathbf{b}_e$ represents the matrix, the degrees of freedom and the forces vector of the deformable model representing the environment. By the action-reaction principle, contact forces are applied to the nodes of the robot deformable models through $\mathbf{H}_{\text{c}}^T$ matrix and to the nodes of the environment model through the $\overline{\mathbf{H}}_{\text{c}}^T$ matrix. The distance at each contact point is also influence by the motions of both the robot and its environment leading to a final distance at the end of the step being: $\boldsymbol{\delta}_{\text{c}} + \mathbf{H}_{\text{c}} d\mathbf{q} + \overline{\mathbf{H}}_{\text{c}} d\mathbf{q}_e$.

The robot and environment models are only coupled by a few contact equations. In general, the number of degrees of freedom of the model (size of $d\mathbf{q}$ and $d\mathbf{q}_e$) is much greater than the number of contacts. It is therefore interesting to solve the problem indirectly by the following steps and the use of the projection of compliance through condensation, as seen in equation 31.

$$\begin{cases} \text{Step 1 : solve the linear system :} \\ \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{A}_e \end{bmatrix} \begin{bmatrix} d\mathbf{q}^0 \\ d\mathbf{q}_e^0 \end{bmatrix} - \begin{bmatrix} \mathbf{b} \\ \mathbf{b}_e \end{bmatrix} = \mathbf{0} \\ \\ \text{Step 2 : solve the Mixed Complementarity Problem (find } \boldsymbol{\lambda}_{\text{a}} \text{ and } \boldsymbol{\lambda}_{\text{c}}\text{) :} \\ \begin{cases} \boldsymbol{\delta}_{\text{a}} + \underbrace{\mathbf{H}_{\text{a}} \mathbf{A}^{-1} \mathbf{H}_{\text{a}}^T}_{\mathbf{W}_{\text{aa}}} \boldsymbol{\lambda}_{\text{a}} + \underbrace{\mathbf{H}_{\text{a}} \mathbf{A}^{-1} \mathbf{H}_{\text{c}}^T}_{\mathbf{W}_{\text{ac}}} \boldsymbol{\lambda}_{\text{c}} = \mathbf{u} \\ 0 \leq \boldsymbol{\delta}_{\text{c}} + \underbrace{\mathbf{H}_{\text{c}} \mathbf{A}^{-1} \mathbf{H}_{\text{a}}^T}_{\mathbf{W}_{\text{ca}}} \boldsymbol{\lambda}_{\text{a}} + \underbrace{(\mathbf{H}_{\text{c}} \mathbf{A}^{-1} \mathbf{H}_{\text{c}}^T + \overline{\mathbf{H}}_{\text{c}} \mathbf{A}_e^{-1} \overline{\mathbf{H}}_{\text{c}}^T)}_{\mathbf{W}_{\text{cc}}} \boldsymbol{\lambda}_{\text{c}} \perp \boldsymbol{\lambda}_{\text{c}} \geq 0 \end{cases} \\ \\ \text{Step 3 : find the final motion :} \\ \begin{cases} d\mathbf{q} = d\mathbf{q}^0 + \mathbf{A}^{-1} (\mathbf{H}_{\text{a}}^T \boldsymbol{\lambda}_{\text{a}} + \mathbf{H}_{\text{c}}^T \boldsymbol{\lambda}_{\text{c}}) \\ d\mathbf{q}_e = d\mathbf{q}_e^0 + \mathbf{A}_e^{-1} \overline{\mathbf{H}}_{\text{c}}^T \boldsymbol{\lambda}_{\text{c}} \end{cases} \end{cases} \tag{34}$$

To solve the linear system (step 1) and find the final motion (step 3), a factorization of matrices $\mathbf{A}$ and $\mathbf{A}_e$ or iterative solvers can be employed. This factorization can be reused to obtain matrices $\mathbf{W}_{\text{ij}}$. For step 2, a convex optimisation solver is used to find the solution to the Mixed Complementarity Problem (MCP).

9.3 Inverse modeling with contact

In section 8.2 we saw that inverse modeling could be found on soft robots thanks to a QP (Quadratic Programming) optimization method based on condensed compliance models. In the previous section, we showed that this condensed compliance can also be found in step 2, at the contact level, to write the MCP (Mixed Complementarity Problem) that allows us to know the contact (and actuation) forces in a direct problem. We show here that we can combine these two optimization problems to find the inverse model of soft robots in contact with their environment. To do this the 3 steps described in equation (34) are repeated, but the MCP in step 2 is replaced

a QPCC (Quadratic Problem with Complementarity Conditions) to follow a trajectory defined in a at the effector $\boldsymbol{\delta}_{\mathrm{e}}(t)$. At step 2, the goal is then to minimize $\boldsymbol{\delta}_{\mathrm{e}}(t)$ by using actuation and contacts.

$$\left\{\begin{array}{l}\text{Step 2 : solve the Quadratic Problem with Complementarity Constraints (QPCC)}\\ \min\limits_{\boldsymbol{\Delta\lambda}_{\mathrm{a}},\boldsymbol{\Delta\lambda}_{\mathrm{c}}} (\boldsymbol{\delta}_{\mathrm{e0}}+\boldsymbol{\Delta\delta}_{\mathrm{e}})^2 \text{ (with } \boldsymbol{\Delta\delta}_{\mathrm{e}}=\mathbf{W}_{\mathrm{ea}}\boldsymbol{\Delta\lambda}_{\mathrm{a}}+\mathbf{W}_{\mathrm{ec}}\boldsymbol{\Delta\lambda}_{\mathrm{c}})\\ \text{with the actuator constraints like}\\ \boldsymbol{\delta}_{\min}\leq\boldsymbol{\delta}_{\mathrm{a0}}+\boldsymbol{\Delta\delta}_{\mathrm{a}}\leq\boldsymbol{\delta}_{\max} \text{ (with } \boldsymbol{\Delta\delta}_{\mathrm{a}}=\mathbf{W}_{\mathrm{aa}}\boldsymbol{\Delta\lambda}_{\mathrm{a}}+\mathbf{W}_{\mathrm{ac}}\boldsymbol{\Delta\lambda}_{\mathrm{c}})\\ \text{and with to contact constraints}\\ 0\leq\boldsymbol{\delta}_{\mathrm{c0}}+\boldsymbol{\Delta\delta}_{\mathrm{c}}\perp\boldsymbol{\lambda}_{\mathrm{c}}+\boldsymbol{\Delta\lambda}_{\mathrm{c}}\geq 0 \text{ (with } \boldsymbol{\Delta\delta}_{\mathrm{c}}=\mathbf{W}_{\mathrm{ca}}\boldsymbol{\Delta\lambda}_{\mathrm{a}}+\mathbf{W}_{\mathrm{cc}}\boldsymbol{\Delta\lambda}_{\mathrm{c}})\end{array}\right.$$

One result of using this optimization can be seen in the figure 15 from [5]. It shows a soft robot, driven by 8 cables and a translational movement of its base. It is shown that when the obstacle is taken into account, the inverse model finds (in an open loop) the actuation that enables it to slide along the obstacle in order to reach the goal (see the movie). Once again, optimization is based essentially on projected compliance in the actuator, effector and contact spaces.

Figure 15: The goal position is illustrated by a green circle in both figures. On the left, the obstacle is not modeled in the inverse model. The obstacle prevents the real robot from reaching the goal. On the right, however, contact with the obstacle is taken into account in the inverse model, enabling the real robot to find a solution to the inverse problem of reaching the goal position while sliding over the obstacle. Figures from [5]

10 Conclusion

The aim of this chapter is to demonstrate the modeling of soft robots through the calculation of mechanical compliance. We have discussed the definition of mechanical compliance, its numerical calculation depending on the material properties and using the finite element method. We've seen how it can be used to model soft robots behavior. We hope this chapter will help democratize this approach, which is not fully focused solely on the FEM aspect, but rather on numerical methods for condensed models. In particular, we feel it's important to emphasize that mechanical compliance can be calculated using numerical methods other than FEM.

We believe that our approach to the kinematics of soft robots, by passing this compliance projected into the robot's useful spaces (actuator spaces, effector spaces, sensor spaces, contact spaces, etc.), is today the most generic. In particular, the properties of the (possibly heterogeneous) materials used to manufacture the robot are taken into account when calculating the robot's kinematics.

Only little discussion on the robots dynamics has been included in this chapter, as it was felt more important to focus on the mechanical compliance. However, the method is generally

compatible with dynamics, especially if we combine with the model reduction method to make calculations fast enough.

Work has been carried out or is in progress to complement this approach, notably with closed-loop control (static and dynamic) [30] [4] [26] [15]. There is undoubtedly still a long way to go, particularly in the multi-physics aspects of sensors and actuators, but also in the acceleration of computation time to be able to use these approaches in on-line predictive control.

Acknowledgment

To write this chapter, the author benefited from the support of the French Agency for Research (project COSSEROOTS under Grant ANR-20-CE33-0001) and the European Unionâs Horizon 2020 research and innovation program (project SimCardioTest under grant agreement No 101016496)

References

[1] Yinoussa Adagolodjo, Federico Renda, and Christian Duriez. Coupling numerical deformable models in global and reduced coordinates for the simulation of the direct and the inverse kinematics of soft robots. *IEEE Robotics and Automation Letters*, 6(2):3910–3917, 2021.

[2] D Baraff and A Witkin. Large steps in cloth simulation. In *Computer graphics proceedings, annual conference series*, pages 43–54. Association for Computing Machinery SIGGRAPH, 1998.

[3] Jernej Barbič and Doug L James. Real-time subspace integration for st. venant-kirchhoff deformable models. *ACM transactions on graphics (TOG)*, 24(3):982–990, 2005.

[4] Thor Morales Bieze, Frederick Largilliere, Alexandre Kruszewski, Zhongkai Zhang, Rochdi Merzouki, and Christian Duriez. Finite element method-based kinematics and closed-loop control of soft, continuum manipulators. *Soft robotics*, 5(3):348–364, 2018.

[5] Eulalie Coevoet, Adrien Escande, and Christian Duriez. Optimization-based inverse model of soft robots with contact handling. *IEEE Robotics and Automation Letters*, 2(3):1413–1419, 2017.

[6] Tim Davis. Csparse: a concise sparse matrix package, 2005.

[7] Christian Duriez. *Real-time haptic simulation of medical procedures involving deformations and device-tissue interactions*. Université des Sciences et Technologie de Lille-Lille I, 2013.

[8] Christian Duriez, Stéphane Cotin, Julien Lenoir, and Paul Neumann. New approaches to catheter navigation for interventional radiology simulation. *Computer aided surgery*, 11(6):300–308, 2006.

[9] Christian Duriez, Frederic Dubois, Abderrahmane Kheddar, and Claude Andriot. Realistic haptic rendering of interacting deformable objects in virtual environments. *IEEE transactions on visualization and computer graphics*, 12(1):36–47, 2005.

[10] Pasquale Ferrentino, Seyedreza Kashef Tabrizian, Joost Brancart, Guy Van Assche, Bram Vanderborght, and Seppe Terryn. Fea-based inverse kinematic control: Hyperelastic material characterization of self-healing soft robots. *IEEE Robotics & Automation Magazine*, 29(3):78–88, 2021.

[11] Mihai Frâncu, Arni Asgeirsson, Kenny Erleben, and Mads JL Rønnow. Locking-proof tetrahedra. *ACM Transactions on Graphics (TOG)*, 40(2):1–17, 2021.

[12] Olivier Goury and Christian Duriez. Fast, generic, and reliable control and simulation of soft robots using model order reduction. *IEEE Transactions on Robotics*, 34(6):1565–1576, 2018.

[13] Gaël Guennebaud, Benoît Jacob, Philip Avery, Abraham Bachrach, Sebastien Barthelemy, et al. Eigen v3, 2010.

[14] Noboru Kikuchi and John Tinsley Oden. *Contact problems in elasticity: a study of variational inequalities and finite element methods*. SIAM, 1988.

[15] Margaret Koehler, Thor Morales Bieze, Alexandre Kruszewski, Allison M Okamura, and Christian Duriez. Modeling and control of a 5-dof parallel continuum haptic device. *IEEE Transactions on Robotics*, 2023.

[16] Maxence Leveziel, Wissem Haouas, Guillaume J Laurent, Michaël Gauthier, and Redwan Dahmouche. Migribot: A miniature parallel robot with integrated gripping for high-throughput micromanipulation. *Science Robotics*, 7(69):eabn4292, 2022.

[17] Luc Marechal, Pascale Balland, Lukas Lindenroth, Fotis Petrou, Christos Kontovounisios, and Fernando Bello. Toward a common framework and database of materials for soft robotics. *Soft robotics*, 8(3):284–297, 2021.

[18] Seri Mastura Mustaza, Yahya Elsayed, Constantina Lekakou, Chakravarthini Saaj, and Jan Fras. Dynamic modeling of fiber-reinforced soft manipulator: A visco-hyperelastic material-based continuum mechanics approach. *Soft robotics*, 6(3):305–317, 2019.

[19] Stefan Escaida Navarro, Olivier Goury, Gang Zheng, Thor Morales Bieze, and Christian Duriez. Modeling novel soft mechanosensors based on air-flow measurements. *IEEE Robotics and Automation Letters*, 4(4):4338–4345, 2019.

[20] 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, 2008.

[21] Janusz S Przemieniecki. *Theory of matrix structural analysis*. Courier Corporation, 1985.

[22] Jian Qu, Kamran Khan, Songhe Meng, and Anastasia Muliana. Modeling nonlinear viscoelastic responses of flexible composites for soft robotics applications. *Mechanics of Advanced Materials and Structures*, 30(14):2793–2805, 2023.

[23] Alejandro Rodríguez, Eulalie Coevoet, and Christian Duriez. Real-time simulation of hydraulic components for interactive control of soft robots. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4953–4958. IEEE, 2017.

[24] Guillaume Saupin, Christian Duriez, and Stephane Cotin. Contact model for haptic medical simulations. In *International Symposium on Biomedical Simulation*, pages 157–165. Springer, 2008.

[25] Matthias Teschner, Stefan Kimmerle, Bruno Heidelberger, Gabriel Zachmann, Laks Raghupathi, Arnulph Fuhrmann, M-P Cani, François Faure, Nadia Magnenat-Thalmann, Wolfgang Strasser, et al. Collision detection for deformable objects. In *Computer graphics forum*, volume 24, pages 61–81. Wiley Online Library, 2005.

[26] Maxime Thieffry, Alexandre Kruszewski, Christian Duriez, and Thierry-Marie Guerra. Control design for soft robots based on reduced-order model. *IEEE Robotics and Automation Letters*, 4(1):25–32, 2018.

[27] Félix Vanneste, Olivier Goury, and Christian Duriez. Enabling the control of a new degree of freedom by using anisotropic material on a 6-dof parallel soft robot. In *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*, pages 636–642. IEEE, 2021.

[28] Félix Vanneste, Olivier Goury, Jonas Martinez, Sylvain Lefebvre, Herve Delingette, and Christian Duriez. Anisotropic soft robots based on 3d printed meso-structured materials: design, modeling by homogenization and simulation. *IEEE robotics and automation letters*, 5(2):2380–2386, 2020.

[29] Ziqiu Zeng and Hadrien Courtecuisse. Efficient parallelization strategy for real-time fe simulations. *arXiv preprint arXiv:2306.05893*, 2023.

[30] Zhongkai Zhang, Jeremie Dequidt, Alexandre Kruszewski, Frederick Largilliere, and Christian Duriez. Kinematic modeling and observer based control of soft robot using real-time finite element method. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5509–5514. IEEE, 2016.

[31] Olek C Zienkiewicz and Robert L Taylor. *The finite element method for solid and structural mechanics*. Elsevier, 2005.

RESEARCH CENTRE
LILLE – NORD EUROPE

Parc scientifique de la Haute-Borne
40 avenue Halley - Bât A - Park Plaza
59650 Villeneuve d'Ascq

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-0803