An optimal control formulation for organ registration in augmented surgery

Guillaume Mestdagh

Université de Strasbourg

UNIVERSITÉ DE STRASBOURG

msii ÉCOLE DOCTORALE

École doctorale 269

Institut de recherche mathématique avancée

THÈSE

présentée par

Guillaume Mestdagh

soutenue le **13 décembre 2022**

pour obtenir le grade de

Docteur de l'université de Strasbourg

Discipline : **Mathématiques**

An optimal control formulation for organ registration in augmented surgery

Membres du jury

M. Emmanuel Maitre ..... *Rapporteur*
Professeur, Grenoble INP – Université Grenoble Alpes

M. Erwan Kerrien ..... *Rapporteur*
Chargé de recherche, Inria

Mme Maya de Buhan ..... *Examinatrice*
Chargée de recherche, CNRS

M. Yohan Payan ..... *Examinateur*
Directeur de recherche, CNRS

M. Yannick Privat ..... *Directeur*
Professeur, Université de Strasbourg

M. Stéphane Cotin ..... *Co-directeur*
Directeur de recherche, Inria

Université de Strasbourg

Guillaume Mestdagh

msii ÉCOLE DOCTORALE

An optimal control formulation for organ registration in augmented surgery

Résumé

La réalité augmentée est utilisée en chirurgie minimalement invasive pour permettre au personnel médical de suivre en temps réel les mouvements du foie du patient. Pour mettre à jour la déformation d'un organe virtuel, une méthode de recalage élastique aligne un modèle biomécanique pré-opératoire du foie avec une surface partielle observée pendant l'opération. Tandis qu'une grande partie des méthodes de recalage élastique consistent à introduire des forces fictives dans le modèle direct, notre approche vise à reconstruire la vraie densité de forces surfaciques qui a créé la déformation observée. Nous exprimons le problème de recalage dans le formalisme du contrôle optimal, en utilisant comme variable d'optimisation la distribution de forces qui s'applique à la surface de l'organe. En permettant de définir à l'avance un ensemble de forces admissibles, cette approche favorise les champs de déplacement ayant un sens physique. Nous commençons par étudier l'existence de solutions pour le problème continu et nous calculons des conditions d'optimalité de premier ordre. Puis nous présentons la méthode d'adjoint que nous avons implémentée afin de traiter le problème numériquement. Finalement, nous validons notre méthode au moyen de cas-test liés à l'application en chirurgie augmentée. Lors de ces essais, nous mesurons l'erreur de recalage, et nous cherchons également, dans un cas particulier, à donner un sens à la distribution de forces obtenue.

Mots-clefs Contrôle optimal – Simulation biomécanique – Chirurgie augmentée

Abstract

In minimally-invasive liver surgery interventions, augmented reality systems aim to help medical staff by tracking the motion of the patient's liver in real time. To compute the updated organ deformation, an elastic registration procedure aligns a pre-operative biomechanical model of the liver with intra-operative partial surface data. While many elastic registration procedures introduce artificial forces into the direct model to drive the registration, we propose an approach to reconstruct the surface loading that actually generated the observed deformation. The registration problem is formulated as an optimal control problem where the unknown is the surface force distribution that applies on the organ. Advantages of this approach include a greater control over the set of admissible forces distributions, which promotes physically-consistent displacement fields. We first study the existence of solutions and compute first-order optimality conditions for the continuous optimal control problem. Then we describe our implementation of an adjoint method to solve the problem numerically. Finally, we validate our method using test cases related to the application in augmented surgery. In these tests, we not only evaluate registration accuracy, but also give a meaning to the reconstructed distribution in a particular case.

Keywords Optimal control – Biomechanical simulation – Augmented surgery

Remerciements

Les premiers récipiendaires de cette liste de remerciements sont, sans surprise, Yannick Privat et Stéphane Cotin, qui ont pris le risque de me recruter en thèse, et qui ont monté ce projet, afin de tisser des liens entre leurs équipes respectives. Merci pour tout ce que vous m'avez appris, qu'il s'agisse de science ou de politique du milieu de la recherche. Merci aussi pour la confiance et l'autonomie que vous m'avez accordées tout au long du projet.

Je remercie Emmanuel Maitre et Erwan Kerrien d'avoir pris le temps de rapporter ce manuscrit, ainsi que Maya de Buhan et Yohan Payan qui ont accepté de participer à mon jury, alors qu'ils sont tous les quatre certainement très occupés.

Merci à mes collègues doctorants de l'Ufr de math/info pour leur compagnie, en particulier pendant le confinement, et merci à tous les locataires d'une aile abandonnée du pavillon Clovis Vincent pour ces parties de coinche aux annonces exceptionnelles.

Enfin, je remercie mes parents, car ils me soutiennent depuis de longues années déjà.

Notations

Γ	Observed surface.
Ω_0	Initial domain.
$\partial\Omega_\mathrm{D}, \partial\Omega_\mathrm{N}$	Dirichlet boundary, Neumann boundary.
d	Geometric space dimension.
g, u, p	Surface force distribution, displacement field and adjoint state in the continuous problem.
$\varepsilon(u)$	Linearized strain tensor.
Ω_u	Domain deformed by u.
d_{Ω_u}	Signed distance with respect to Ω_u.
S_0, S_u	Initial matching surface, current matching surface.
$J(u), R(g)$	Data attachment functional, regularization term.
$\mathrm{d}J(u)(v), \langle \mathrm{d}J(u), v\rangle$	Derivative of J in the general case and in the Gateaux-differentiable case.
$\mathcal{G}_M$	Admissible set $\{g \in L^\infty(\partial\Omega_\mathrm{N}) \mid \|g\|_{L^\infty} \leqslant M\}$.
$W^{1,p}_\mathrm{D}(\Omega_0)$	Functions of $W^{1,p}(\Omega_0)$ that vanish on $\partial\Omega_\mathrm{D}$.
x	Point in Ω_0 or Ω_u.
y	Point in Γ.
$d(y, S_u)$	Distance between y and S_u.
$\Pi_{S_u}(y)$	Set of projections of y onto S_u.
$p_{S_u}(y)$	Unique projection of y onto S_u.
$P_y(u)$	$\{x \in S_0 \mid \|x + u(x) - y\| = d(y, S_u)\}$.
$\mathbf{b}, \mathbf{u}, \mathbf{p}$	Nodal force distribution, displacement field and adjoint state in the discrete problem.
n	Number of mesh nodes.
$W(\mathbf{u})$	Elastic deformation energy.
$\mathbf{F}(\mathbf{u}), \mathbf{F}'(\mathbf{u}) = \mathbf{K}(\mathbf{u})$	Hyperelasticity residual and Jacobian.
$\mathbf{A}$	Stiffness matrix in linear elasticity.
$\mathbf{M}^\mathrm{T}, \mathbf{M}^{-\mathrm{T}}$	Transposed and inverse transposed matrix.

Contents

Introduction en français

La naissance de ce projet résulte des discussions entre Yannick Privat, professeur à l'Institut de recherche mathématique avancée de Strasbourg, et Stéphane Cotin, chef de l'équipe Inria Mimesis. L'équipe Mimesis est spécialisée, entre autres, dans les systèmes de chirurgie assistée par ordinateur, et, plus généralement, dans la simulation biomécanique en temps réel.

Comme la plupart des organes abdominaux, le foie est un organe mou, et par conséquent il subit souvent des déformations importantes. Lors d'une opération chirurgicale, ces déformations peuvent venir du changement de position du patient, de sa respiration ou encore de la manipulation de l'organe par le médecin. Par ailleurs, au cours d'une opération faiblement invasive, le médecin manipule les organes indirectement. Pour cela, il se sert d'instruments insérés à travers la peau du patient (voir figure 1), tandis qu'un retour vidéo affiché sur un écran lui permet de voir ce qu'il fait. Le mouvement du milieu, permanent, et le champ de vision, limité, sont autant de facteurs qui rendent les opérations faiblement invasives difficiles.

Pour aider les soignants à visualiser la déformation du foie au cours du temps, des systèmes de réalité augmentée ont été mis au point. Ceux-ci produisent une vue tridimensionnelle de l'organe dans sa configuration actuelle, qui fait apparaître les structures internes, tumeurs et autres vaisseaux sanguins. Dans ce manuscrit, nous nous intéressons principalement à la méthode de recalage élastique qui met à jour la déformation

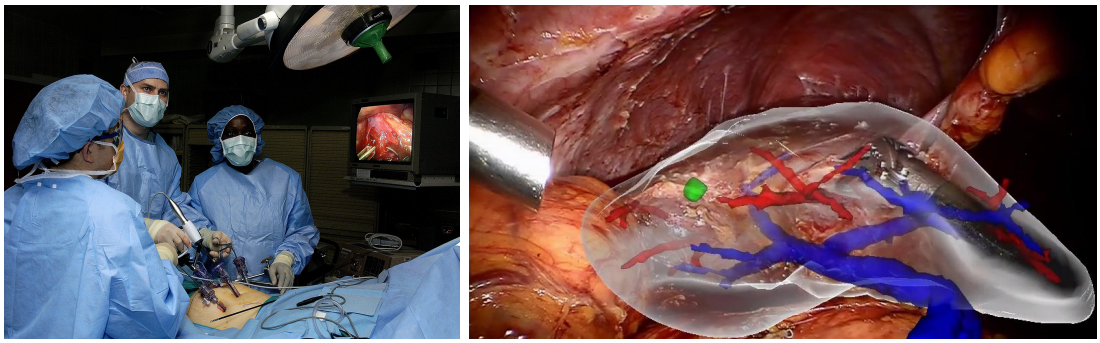


FIGURE 1 : Gauche : intervention en chirurgie laparoscopique. (Samuel Bendet, US Air Force, https://commons.wikimedia.org/wiki/File:Laparoscopic_stomach_surgery.jpg). Droite : vue du foie en réalité augmentée (stephanecotin.com).

du foie en combinant des données acquises avant et pendant l'opération. D'une part, un modèle biomécanique du foie est extrait d'images tomographiques réalisées quelques jours avant l'opération. D'autre part, une caméra laparoscopique fournit en temps réel un nuage de points montrant une partie de la surface du foie dans sa configuration actuelle. Le recalage vise à déterminer pour le modèle pré-opératoire une déformation qui correspond aux données observées.

Plusieurs difficultés sont liées au recalage élastique du foie. Tout d'abord, la déformation du foie doit être mise à jour en temps réel pendant le déroulement de l'opération. Notons cependant que le calcul en temps réel ne fait pas partie de nos prétentions dans ce manuscrit. Par ailleurs, les données per-opératoires, ne montrent qu'une partie de la surface du foie, et par conséquent le problème d'appariement de surfaces admet plusieurs solutions, qui n'ont pour la plupart rien à voir avec la physique du problème. Afin de calculer un champ de déplacement aussi proche que possible de la réalité, la méthode de recalage doit donc réduire au maximum l'ensemble des déplacements pouvant être associés à une observation donnée.

Chapitre 2. Revue de littérature et modélisation du problème

Les méthodes de recalage élastique reposent sur un modèle direct décrivant la mécanique du foie du patient. Ce modèle est souvent discrétisé par la méthode des éléments finis. En général, beaucoup d'attention est accordée au choix du modèle élastique décrivant les déformations du foie et ses interactions avec son environnement, si bien qu'une grande variété de modèles est utilisée dans la littérature. Habituellement, ces modèles sont de type hyperélastique. Cependant, lorsque le calcul en temps réel est requis, on optera plutôt pour un modèle dit co-rotationnel, qui permet d'engendrer des déformations non-linéaires en ne résolvant que des systèmes linéaires. Les conditions aux frontières utilisées sont également d'une grande variété et vont de la simple condition de Dirichlet au modèle complexe, décrivant les ligaments qui maintiennent le foie en place comme des solides hyperélastiques.

Concernant le processus de recalage lui-même, il consiste à appliquer des forces sur le maillage de foie pour créer une déformation. Une grande classe de méthodes est directement inspirée des méthodes de recalage d'image. On y retrouve des forces artificielles qui attirent le maillage vers le nuage de points observé, et le modèle élastique n'y joue qu'un rôle de régularisation. Dans ce type de méthode, les forces appliquées ne reflètent pas l'origine réelle de la déformation. Elles n'ont même rien à voir, puisqu'elles sont créées par un nuage de points qui n'existe pas réellement. D'autres approches se soucient davantage de savoir si le champ de déplacement engendré est plausible au vu de la physique. Dans cette deuxième classe de méthodes, un ensemble de forces ou de déplacements admissibles est défini à l'avance en se basant sur des hypothèses physiques. Les déformations calculées sont ainsi plus réalistes.

Notre approche s'identifie plutôt à la seconde catégorie. Nous exprimons le problème de recalage comme un problème de contrôle optimal. La formulation qui en découle est

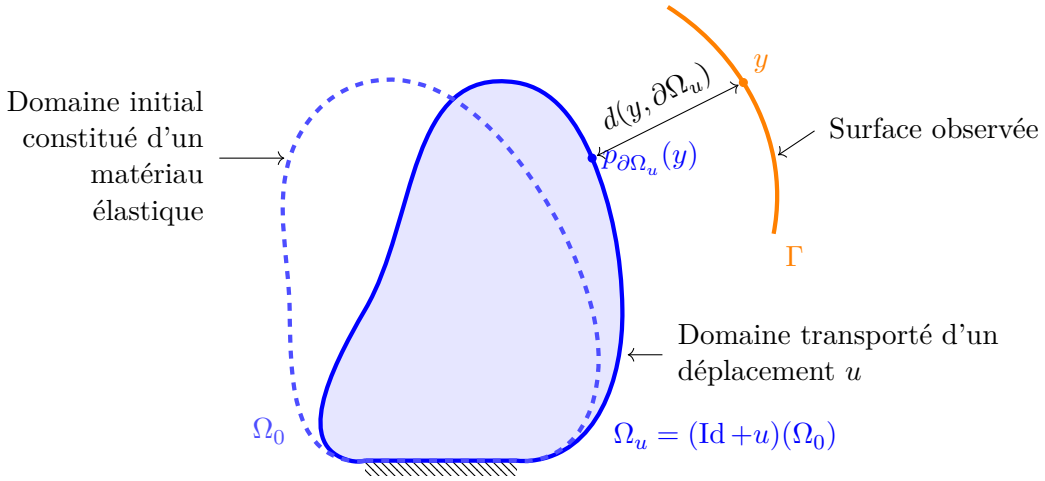


FIGURE 2 : Géométrie du problème. La distance entre un point $y \in \Gamma$ et sa projection orthogonale sur $\partial\Omega_u$ s'écrit $d(y, \partial\Omega_u)$.

très flexible, et permet de bénéficier de nombreux outils génériques d'optimisation pour étudier et résoudre numériquement le problème. Dans ce manuscrit, nous considérons le problème

$$\inf_{g \in \mathcal{G}_M} J(u) + \tfrac{\alpha}{2}\|g\|^2_{L^2(\partial\Omega_{\mathrm{N}})} \quad \text{sous la contrainte} \quad \begin{cases} \mathrm{div}(A\varepsilon(u)) &= 0 \quad \text{dans } \Omega_0 \\ u &= 0 \quad \text{sur } \partial\Omega_{\mathrm{Dirichlet}} \\ A\varepsilon(u)\cdot n &= g \quad \text{sur } \partial\Omega_{\mathrm{Neumann}}. \end{cases}$$

Ici, le champ de déplacement u est obtenu en appliquant la distribution de forces g sur la surface du maillage. L'ensemble des forces admissibles est défini par

$$\mathcal{G}_M = \left\{ g \in L^\infty(\partial\Omega_{\mathrm{N}}) \;\middle|\; \|g\|_{L^\infty(\partial\Omega_{\mathrm{N}})} \leqslant M \right\}.$$

Le problème de contrôle optimal fait intervenir la fonctionnelle d'attache aux données

$$J(u) = \tfrac{1}{2}\int_\Gamma d^2(y, S_u)\,\mathrm{d}y,$$

où S_u est la partie de surface de foie déformée qui doit atteindre la surface observée Γ, et $d(y, S_u)$ est la distance de y à S_u. Les notations sont illustrées en figure 2 dans le cas particuler $S_u = \partial\Omega_u$. Les coefficients $M > 0$ et $\alpha > 0$ peuvent être ajustés pour régulariser le champ de déplacement malgré la présence de bruit dans les données.

Chapitre 3. Formes et conditions aux limites mixtes

Pour étudier le problème de contrôle optimal d'un point de vue mathématique, il peut être utile de connaitre certains outils et résultats. Le recalage étant une opération de nature géométrique, les outils que nous présentons dans ce chapitre tournent autour des

formes, des distances et projections orthogonales, mais aussi des solutions continues du système de l'élasticité linéaire.

On peut définir la régularité d'une forme. Celle-ci correspond à la régularité des transformations qui permettent d'aplatir sa frontière. Nous présentons également la dérivée de forme d'Hadamard, qui donne la possibilité de calculer les dérivées d'une fonction dépendant d'une forme.

Les distances et projections orthogonales sont des outils utiles pour repérer un point par rapport à une forme. La régularité de ces applications dépend en réalité de la régularité de la forme concernée.

Les résultats de continuité pour le système d'élasticité linéaire avec conditions aux bords mixtes s'appuie sur le formalisme des ensembles réguliers au sens de Gröger. Nous présentons cette notion ainsi que certains théorèmes dont nous nous servons par la suite pour obtenir des résultats théoriques.

Chapitre 4. Existence de solutions et conditions d'optimalité

Avant de passer au problème de contrôle optimal proprement dit, nous étudions la fonctionnelle J. Bien qu'elle ne soit pas toujours différentiable au sens de Gateaux, la fonction J admet toujours des dérivées directionnelles, et est continue pour la norme de la convergence uniforme. Sa dérivée dans la direction v s'écrit

$$\mathrm{d}J(u)(v) = \int_\Gamma \min_{x \in P_y(u)} [v(x) \cdot (x + u(x) - y)] \, \mathrm{d}y,$$

où les points x de $P_y(u)$ vérifient $\|x + u(x) - y\| = d(y, S_u)$. Dans le cas où presque tous les points de Γ possèdent une unique projection sur S_u, $\mathrm{d}J(u)$ est une forme linéaire et J est différentiable au sens de Gateaux.

Nous étudions l'existence de solutions pour le problème de contrôle optimal en utilisant le formalisme des domaines Gröger-réguliers. La présence de conditions aux bords mixtes représente une difficulté pour montrer l'existence de solutions, car la régularité des déplacements obtenus s'en retrouve affaiblie. Dans ce travail, nous exploitons la régularité $W^{1,p}$ des solutions de l'élasticité linéaire. Nous proposons un résultat partiel limité à la dimension 2, le cas de la dimension 3 s'avérant nettement plus complexe. Toutefois, quand le système d'élasticité est remplacé par un système elliptique, nous montrons que des solutions existent quelle que soit la dimension.

Nous calculons les conditions d'optimalité de premier ordre pour plusieurs versions du problème de contrôle optimal. Elles font intervenir un état adjoint, qui représente la différentielle de J dans l'espace des contrôles, ainsi que des multiplicateurs de Lagrange pour gérer la contrainte L^∞ sur g. Du fait de cette contrainte, la précaution est de mise dans le calcul des conditions d'optimalité, puisqu'il n'est pas sûr que J soit Gateaux-différentiable à l'optimum. Le théorème ci-dessous est une version simplifiée du résultat principal de cette section. Il énonce les conditions d'optimalité pour un point critique où J est Gateaux-différentiable. Le résultat présenté dans le manuscrit est plus général

et ne fait pas l'hypothèse que J est Gateaux-différentiable. Par ailleurs, nous présentons des exemples où J est différentiable ou pas afin de fixer les idées.

Théorème. *Soit $g \in \mathcal{G}_M$ un minimum local du problème de contrôle optimal et u_g l'état associé. Supposons en outre que J est différentiable au sens de Gateaux en u_g et notons p l'état adjoint correspondant. Alors il existe un multiplicateur de Lagrange $\lambda \in L^2(\partial\Omega_{\mathrm{N}}, \mathbb{R})$, avec*

$$p.p.\ x \in \partial\Omega_{\mathrm{N}} \qquad \begin{cases} \lambda(x) & = 0 \quad si\ \|g(x)\| < M \\ \lambda(x) & \geqslant 0 \quad si\ \|g(x)\| = M, \end{cases}$$

tel que g vérifie la condition d'optimalité de premier ordre

$$p.p.\ x \in \partial\Omega_{\mathrm{N}} \qquad p(x) + (\alpha + \lambda(x))g(x) = 0.$$

Chapitre 5. Méthode d'adjoint pour résoudre le problème de recalage

Nous discrétisons le problème par la méthode des éléments finis. Le problème de contrôle optimal discrétisé s'écrit

$$\min_{\mathbf{b}\in\mathcal{B}} \Phi(\mathbf{b}) \qquad \text{avec} \qquad \Phi(\mathbf{b}) = J(\mathbf{u_b}) + R(\mathbf{b}),$$

où $\mathbf{b}$ est le vecteur des forces nodales s'appliquant sur les sommets du maillage, et $\mathbf{u_b}$ est le champ de déplacement solution du système élastique discrétisé $\mathbf{F}(\mathbf{u_b}) = \mathbf{b}$. Dans la formulation discrète, la seule variable sur laquelle le solveur d'optimisation agit est le contrôle $\mathbf{b}$. Le calcul des dérivées de $J(\mathbf{u_b})$ par rapport à $\mathbf{b}$ nécessite l'utilisation d'un état adjoint. Plus précisément, le gradient de la fonction objectif s'écrit

$$\nabla\Phi(\mathbf{b}) = \mathbf{p} + \nabla R(\mathbf{b}),$$

où l'état adjoint $\mathbf{p}$ est la solution du problème adjoint $\mathbf{F}'(\mathbf{u_b})^{\mathrm{T}}\mathbf{p} = \nabla J(\mathbf{u_b})$.

Notre implémentation de la méthode d'adjoint est constituée de trois procédures, chacune ayant un rôle particulier : évaluation de la fonctionnelle J, gestion du modèle d'éléments finis, résolution du problème d'optimisation (voir figure 3). Nous discutons successivement des choix techniques effectués pour chacune des procédures.

En premier lieu, la fonction d'attache aux données J dépend désormais de la position des nœuds du maillage et non plus d'un champ de déplacement continu. L'évaluation de J nécessite d'exécuter une procédure coûteuse pour trouver l'élément de surface du foie le plus proche de chaque point $y \in \Gamma$. Plusieurs procédures, dont une faisant appel à la distance signée par rapport à l'organe déformé, ont été testées. Notre choix s'est finalement porté sur une méthode dans laquelle les triangles candidats à l'appariement sont rangés dans une structure d'indexation spatiale, permettant ainsi d'effectuer des recherches de plus proche voisin très efficacement. Quant aux dérivées de J, elles peuvent

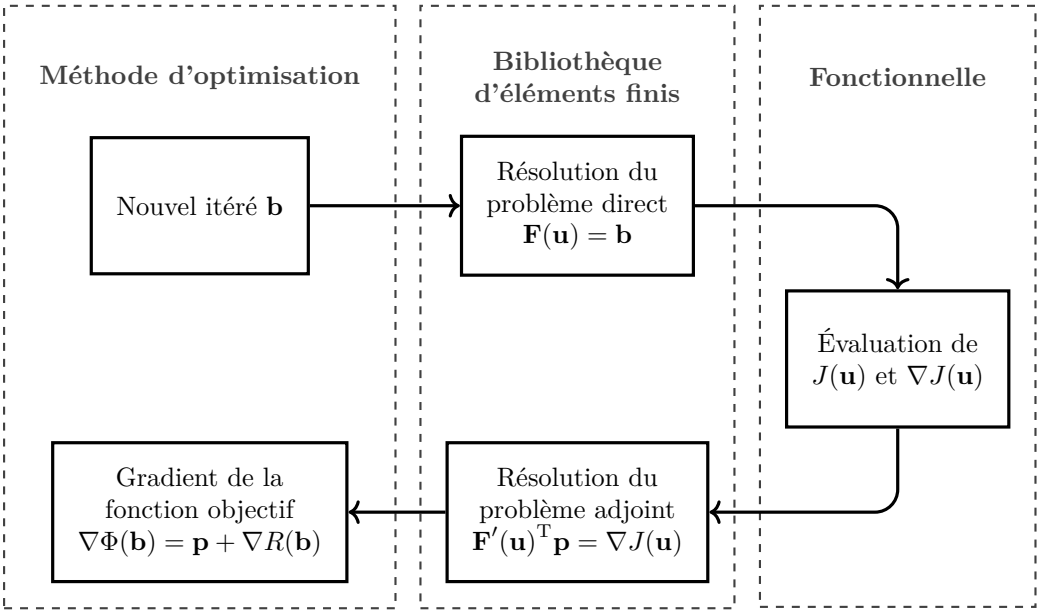


FIGURE 3 : Structure de la méthode d'adjoint. La chaîne directe (en haut) vise à évaluer la fonctionnelle étant donné un contrôle $\mathbf{b}$, tandis que la chaîne adjointe (en bas) transforme le gradient $\nabla J(\mathbf{u})$ en un gradient dans l'espace des contrôles.

être évaluées en appliquant directement la formule obtenue dans le chapitre sur l'analyse du problème.

La méthode qu'on utilise pour résoudre le problème d'élasticité direct n'est pas la même selon le modèle élastique choisi. Dans le cas linéaire, la matrice de raideur est assemblée puis factorisée en début d'exécution, puis la forme factorisée est utilisée pour résoudre aussi bien le problème direct que le problème adjoint. Dans le cas non-linéaire, c'est une autre paire de manches. La méthode de Newton est un outil classique pour traiter ce type problème variationnel. Cependant, bien qu'elle affiche un taux de convergence quadratique dans ses dernières itérations, la méthode de Newton sous sa forme traditionnelle n'offre aucun garantie de convergence si le point de départ est situé loin d'une solution. Comme la convergence du solveur direct est une condition *sine qua non* pour utiliser la méthode d'adjoint, nous décrivons une méthode de Newton à région de confiance, issue du domaine de l'optimisation numérique, pour laquelle stabilité et convergence sont garanties.

Afin de résoudre le problème d'optimisation, nous avons initialement implémenté quelques algorithmes de premier ordre à la main, avant de nous tourner vers des méthodes toutes faites disponibles dans la bibliothèque Scipy, et plus particulièrement une méthode de quasi-Newton à mémoire limitée. Dans le cas où on applique la contrainte L^∞, cette dernière est gérée de manière primale-duale. Nous illustrons la convergence des différents algorithmes à l'aide de problèmes simples.

Chapitre 6. Résultats numériques

La section des résultats numériques vise à illustrer les performances de la méthode d'adjoint du point de vue du problème de recalage. Nous commençons par quelques exemples avec des maillages circulaires ou sphériques. Ces examples sont l'occasion d'illustrer l'effet régularisant de la méthode d'adjoint. Nous montrons également qu'aucune garantie en terme d'erreur de recalage ne peut être attendue de la part de notre méthode. Enfin, nous nous demandons s'il est efficace d'utiliser une pénalisation L^2 ou une contrainte L^∞ pour filtrer le bruit dans les données.

Afin de savoir si notre approche est adaptée pour une utilisation en chirurgie augmentée, nous produisons des résultats de recalage basés sur les données du *Sparse Data Challenge*. Les données, acquises à l'aide d'un fantôme en silicone, comprennent un maillage de foie ainsi que 112 nuages de points représentant autant de configurations déformées différentes. Une fois le recalage effectué et les déformations reconstruites envoyées aux organisateurs, ces derniers indiquent l'erreur de recalage commise en moyenne. Nous obtenons la deuxième place du concours, avec un erreur moyenne de 3,3 mm, bien au-dessous des 5 mm requis par nos collègues cliniciens.

Dans le dernier cas-test, nous tentons d'estimer une force ponctuelle appliquée par un bras robot sur la surface du foie. Alors que les robots chirurgicaux embarquent rarement des capteurs d'efforts, mesurer les forces appliquées est capital pour éviter de causer des dégats sur le foie du patient. Nous créons des données synthétiques comprenant des déformations et les nuages de points associés, afin de simuler la manipulation du foie par un instrument au bout d'un bras robot. Puis nous reconstruisons une distribution de force sur un support restreint, limité à une petite zone sur la surface du foie. La force résultante est obtenue en additionnant toutes les forces nodales. Chacun de nos cinq cas-test met en scène une force dépendant du temps (50 pas de temps). L'erreur d'estimation de la force résultante est de 10 % en moyenne, et le délai de mise à jour d'environ 1 s, ce qui est encourageant.

Contributions de la thèse

Cette thèse comporte quatre contributions principales :

- Nous identifions le besoin exprimé dans la littérature pour une modélisation correcte des causes réelles de la déformation, et nous proposons une formulation basée sur le contrôle optimal pour répondre à ce besoin.

- Nous effectuons une analyse mathématique du problème de contrôle optimal, y compris l'étude de l'existence de solutions et le calcul des conditions d'optimalité de premier ordre.

- Nous décrivons une implémentation modulaire de la méthode d'adjoint pour le problème de recalage, et nous la validons en utilisant les données du *Sparse Data Challenge*.

- En guise d'application, nous proposons d'utiliser la méthode d'adjoint pour estimer une force ponctuelle appliquée par un instrument.

Les deux derniers points font l'objet de l'article de conférence suivant :

- Guillaume Mestdagh et Stéphane Cotin (2022). « An Optimal Control Problem for Elastic Registration and Force Estimation in Augmented Surgery ». In : *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022*. Sous la dir. de Linwei Wang et al. Cham : Springer Nature Switzerland, p. 74-83. isbn : 978-3-031-16449-1. doi : 10.1007/978-3-031-16449-1_8

Un autre article, davantage orienté vers l'analyse mathématique du problème, sera très prochainement soumis à une revue internationale à comité de lecture.

Chapter 1

Overview of the manuscript

This project results from discussions between Yannick Privat, professor at the *Institut de recherche mathématique avancée* in Strasbourg, and Stéphane Cotin, leader of the Inria Mimesis team. The domains of expertise of the Mimesis team include computer-assisted surgery systems, and, more broadly, real-time biomechanical simulation.

Compared to open surgery, minimally-invasive surgery is known to improve the outcome of surgical interventions, reducing pain, bleeding and risk of infection. During a minimally-invasive intervention, the surgeon manipulates organs using instrument inserted through small incisions in the patient's abdomen, and receives visual feedback on a screen, thanks to a laparoscopic camera introduced through one of the incisions (see Figure 1.1). In this context of indirect interaction and feedback, operations such as hepatic tumor resections remain challenging, as the surgeon navigates in an opaque organ, avoiding blood vessels and following the motion due to breathing and heartbeats.

When an augmented reality system is used, an up-to-date view of the liver in its current configuration is displayed to help the medical staff keep track of the position of internal structures, such as tumors and blood vessels. In this manuscript, we are

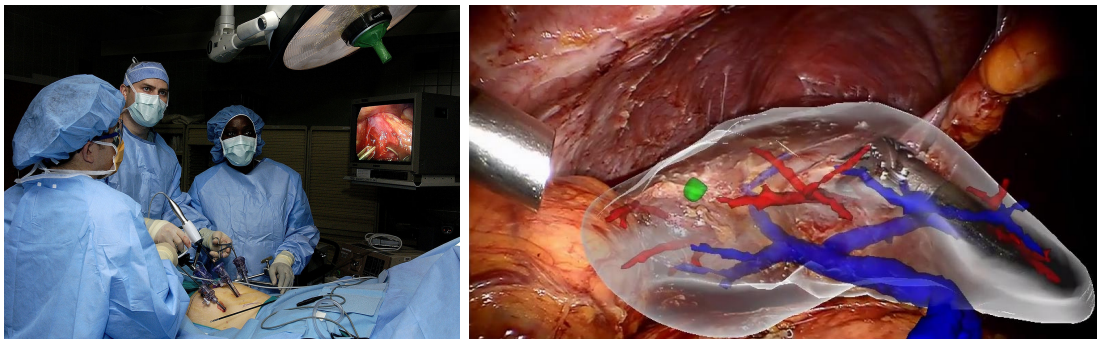


Figure 1.1: Left: a laparoscopic surgical intervention. (Samuel Bendet, US Air Force, https://commons.wikimedia.org/wiki/File:Laparoscopic_stomach_surgery.jpg). Right: Augmented view of the liver (stephanecotin.com).

interested in the elastic registration procedure that computes an updated deformation field in the liver volume by combining pre- and intra-operative data. Namely, this procedure aligns a mesh representing the liver as it appears in pre-operative images with a part of the liver surface as it appears on images from the laparoscopic camera during the intervention.

The elastic registration procedure raises several challenges. First, to update the deformed view as the intervention goes on, real-time computation is required. Note that we are not expecting to meet the real-time requirement in this manuscript. In addition, intra-operative data only show a part of the current liver surface, and as a consequence, many solutions to the matching problem exist, and most of them are not consistent with the problem physics. A successful registration approach should reduce as much as possible the range of admissible displacements for a given observed surface, to return a displacement field as close as possible to the true liver displacement.

Chapter 2. State of the art and problem modelling

An elastic registration procedure relies on a mechanical model of the patient's liver, and usually involves the finite element method. In general, much attention is given to the elastic model that describes the liver deformations and interactions with its surroundings (Section 2.2). Existing methods in the literature involve hyperelastic models, and switch to the co-rotational model when real-time computing is required. A wide range of boundary conditions is also used, going from Dirichlet boundary conditions on certain zones of the liver surface to complex elastic model for the ligaments holding the liver.

The registration process itself is often driven by forces applied on the liver mesh to create a deformation (Section 2.3). In a whole class of methods, inspired from the image registration domain, artificial forces simply attract the liver mesh toward the observed point cloud, while the elastic model mostly plays the role of a regularizer. In these methods, we observe that forces driving the registration do not reflect the real causes of displacement. In particular, these forces are created by intra-operative data points, which do not really exist. Other methods are more concerned with reconstructing physically-relevant displacement fields. In this second class of methods, forces and displacements are chosen among an admissible set defined from physical hypotheses, resulting in more realistic displacement fields.

Our approach is closer to the second category. We formulate the registration problem using the optimal control formalism, resulting in a flexible formulation which can be studied and solved numerically using a wide variety of standard optimization tools (Section 2.4). Namely, the optimal control problem studied in this manuscript has the form

$$\inf_{g\in\mathcal{G}_M} J(u) + \tfrac{\alpha}{2}\|g\|^2_{L^2(\partial\Omega_{\mathrm{N}})} \quad \text{subject to constraint} \quad \begin{cases} \operatorname{div}(A\varepsilon(u)) &= 0 & \text{in } \Omega_0 \\ u &= 0 & \text{on } \partial\Omega_{\mathrm{Dirichlet}} \\ A\varepsilon(u)\cdot n &= g & \text{on } \partial\Omega_{\mathrm{Neumann}}. \end{cases}$$

In this formulation, u is the displacement field generated by the force distribution g.

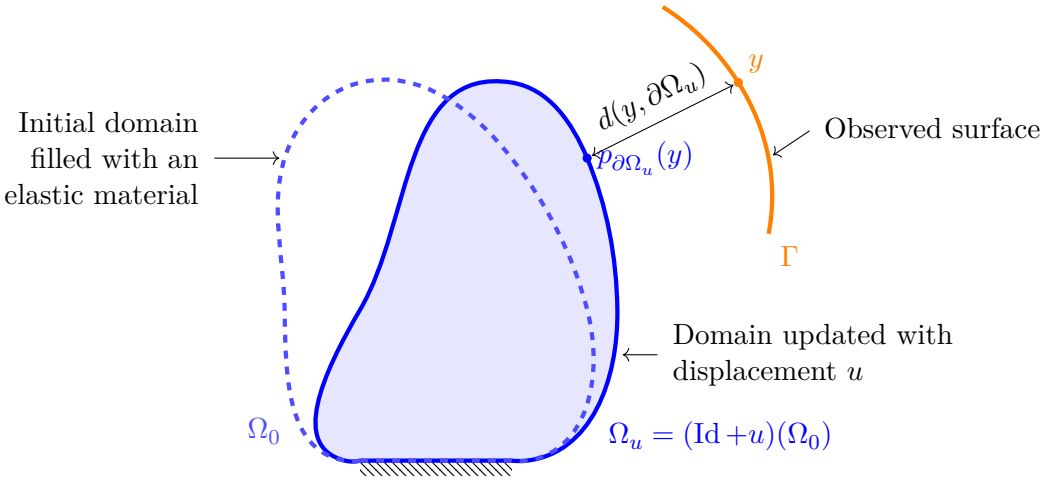


Figure 1.2: Problem geometry. The distance between a point $y \in \Gamma$ and its orthogonal projection onto Ω_u is denoted by $d(y, \partial\Omega_u)$.

The set of admissible forces is defined by

$$\mathcal{G}_M = \left\{ g \in L^\infty(\partial\Omega_{\mathrm{N}}) \;\middle|\; \|g\|_{L^\infty(\partial\Omega_{\mathrm{N}})} \leqslant M \right\}.$$

The optimal control formulation features the data attachment functional

$$J(u) = \tfrac{1}{2} \int_\Gamma d^2(y, S_u)\, \mathrm{d}y,$$

where S_u is the part of the deformed liver boundary that should match the observed surface Γ and $d(y, S_u)$ is the distance from y to S_u. Figure 1.2 illustrates the notation in the case $S_u = \partial\Omega_u$. The L^∞ bound $M > 0$ and the penalty coefficient $\alpha > 0$ can be tuned to enforce displacement regularity despite the presence of noise.

Chapter 3. A few tools around shapes and mixed boundary conditions

To obtain mathematical insight about the optimal control problem, some tools and existing results might be of help. As the registration problem is rather geometric, the tools presented in this chapter revolve around shapes, distances, projections, and also making sure that elastic displacements are continuous.

It is possible to define the regularity of a shape, based on transformations that should be applied to its boundary to obtain something flat (Section 3.1). We also describe the Hadamard boundary variation framework, which is useful to differentiate a function whose variable is a shape.

Distances and orthogonal projections are useful to locate a point with respect to a shape (Section 3.2). The regularity of those operations depend on the regularity of the concerned shape.

Continuity results for solutions to the linear elasticity system with mixed boundary conditions rely on the framework of Gröger-regular sets (Section 3.3). We describe this notion and cite a few theorems that are of use in the theoretical chapter.

Chapter 4. Existence of solutions and optimality conditions

Before we state some results about the optimal control problem, we study the differentiability of the functional J (Section 4.1). Though it is not always Gateaux differentiable, J always has directional derivatives, and is continuous for the uniform convergence norm. Its derivative in the direction v reads

$$\mathrm{d}J(u)(v) = \int_\Gamma \min_{x \in P_y(u)} [v(x) \cdot (x + u(x) - y)] \, \mathrm{d}y,$$

where points $x \in P_y(u)$ satisfy $\|x + u(x) - y\| = d(y, S_u)$. If almost every point in Γ has a single projection onto S_u, then $\mathrm{d}J(u)$ is a linear form and J is Gateaux differentiable at u.

We study the existence of solutions to the optimal control problem using the framework of Gröger-regular sets for partial differential equations with mixed boundary conditions (Section 4.2). The presence of mixed boundary conditions represents a difficulty when it comes to proving the existence of solutions, as they degrade the regularity of resulting displacement fields. Our results rely on the uniform $W^{1,p}$ regularity of solutions to the linear elasticity system. We present a partial result limited to dimension 2, as the three-dimensional case is more complex. However, when the elastic system is replaced by an elliptic system, we obtain existence of solutions in any dimension.

First-order optimality conditions are derived for several versions of the optimal control problem (Section 4.3). They involve an adjoint state, which represents the derivatives of J in the space of controls, as well as Lagrange multipliers to manage the L^∞ constraint on g. Due to the L^∞ constraint, computing optimality conditions requires special attention, as J is not guaranteed to be differentiable at an optimum. The theorem below is a simplified version of the main result of the section. It expresses the optimality conditions for a minimizer where J is Gateaux differentiable. Note that the result presented in the manuscript (Theorem 4.4) is more generic and does not assume Gateaux differentiability. In addition, some examples where J is differentiable or not are given in Section 4.1.

Theorem. *Let $g \in \mathcal{G}_M$ a local minimizer of the optimal control problem and u_g the associated state. Assume that J is Gateaux differentiable at u_j and denote by p the adjoint state. Then there exists a Lagrange multiplier $\lambda \in L^2(\partial\Omega_\mathrm{N}, \mathbb{R})$ with*

$$\text{for a.e. } x \in \partial\Omega_\mathrm{N} \qquad \begin{cases} \lambda(x) & = 0 \quad \text{if } \|g(x)\| < M \\ \lambda(x) & \geqslant 0 \quad \text{if } \|g(x)\| = M \end{cases}$$

such that g satisfies the first-order optimality condition

$$\text{for a.e. } x \in \partial\Omega_\mathrm{N} \qquad p(x) + (\alpha + \lambda(x))g(x) = 0.$$

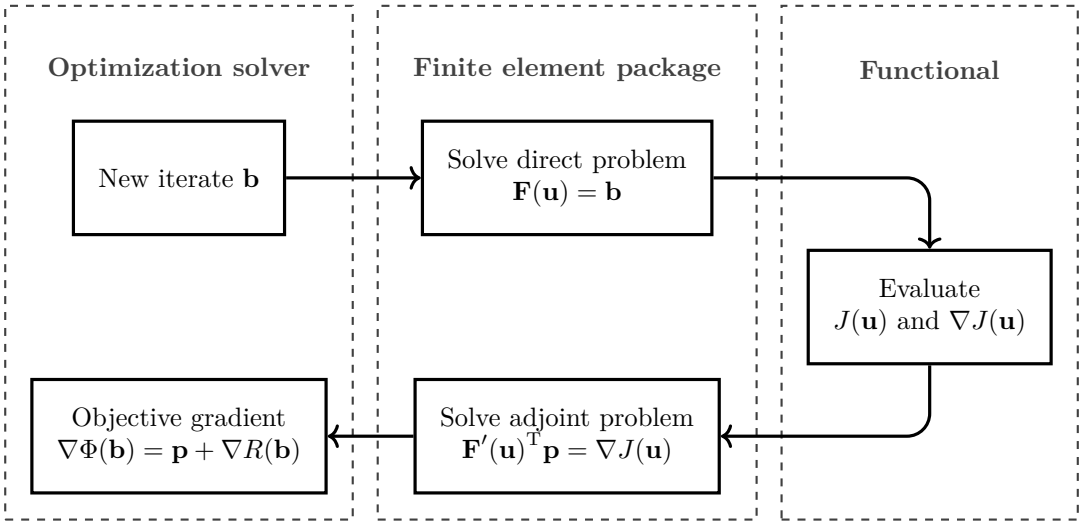


Figure 1.3: Structure of the adjoint procedure. The forward chain (top) aims to evaluate the functional for a given control $\mathbf{b}$, while the backward chain (bottom) transforms the gradient $\nabla J(\mathbf{u})$ into a gradient in the space of controls.

Chapter 5. An adjoint method to solve the registration problem

We discretize the mechanical problem using the finite element method. In the numerical formulation, the optimal control problem reads

$$\min_{\mathbf{b}\in\mathcal{B}} \Phi(\mathbf{b}) \qquad \text{with} \qquad \Phi(\mathbf{b}) = J(\mathbf{u}_\mathbf{b}) + R(\mathbf{b}),$$

where $\mathbf{b}$ is the nodal force distribution at the mesh vertices, and $\mathbf{u}_\mathbf{b}$ is the displacement field defined by the discretized elastic system $\mathbf{F}(\mathbf{u}_\mathbf{b}) = \mathbf{b}$. In the discretized formulation, the only variable seen by the optimization solver is the control $\mathbf{b}$. To compute derivatives of $J(\mathbf{u}_\mathbf{b})$ with respect to $\mathbf{b}$, an adjoint method is used (Section 5.2). Namely, the objective gradient reads

$$\nabla\Phi(\mathbf{b}) = \mathbf{p} + \nabla R(\mathbf{b}),$$

where the adjoint state $\mathbf{p}$ is solution to the adjoint problem $\mathbf{F}'(\mathbf{u}_\mathbf{b})\mathbf{p} = \nabla J(\mathbf{u}_\mathbf{b})$.

Our implementation of the adjoint method is divided into three parts. The three parts are responsible for evaluating the functional J, managing the finite-element discretization, and solving the optimization problem, respectively (see Figure 1.3). We successively discuss technical options regarding each one of these parts.

First, the discrepancy functional J is now a function of the mesh vertices positions and not of the continuous displacement field (Section 5.3). Evaluating J involves a costly procedure to find the closest triangle in the liver boundary to each point $y \in \Gamma$. Several procedures were implemented, including one based on the signed distance to the deformed organ. We finally chose a procedure where triangles that are candidates to

match Γ are stored in a spatial indexing structure, resulting in efficient nearest-neighbor searches. Concerning the derivatives of J, they can be evaluated using the derivative formula obtained in the theoretical chapter.

The algorithm used to solve the direct and adjoint elasticity problem depends on the chosen elastic model. When a linear model is used, the stiffness matrix is assembled and factorized as the procedure starts and the factorization is used for solving both direct and adjoint problems. However, solving the direct problem in the nonlinear case requires much more precaution. The Newton method is a standard tool for such nonlinear variational problems. Though it enjoys a fast convergence rate in the last iterations, the Newton method in its standard form is not guaranteed to converge when the starting point is too far from a solution. As enforcing convergence of the direct solver is critical for the adjoint method to succeed, we describe a trust-region Newton method, developed by the optimization community, where convergence and stability are guaranteed (Section 5.4).

To solve the optimization problem, we first implemented some artisanal methods, but we finally turned to off-the-shelf algorithms available in the Scipy package (Section 5.5). Namely, we use a limited-memory quasi-Newton method. In the L^∞-constrained case, the constraint is managed by a primal-dual strategy. We show results on simple toy problems to illustrate the convergence of our procedure.

Chapter 6. Numerical results

The results section aims to illustrate the performance of the adjoint method in terms of registration error. First, we show a few examples with spherical and circular meshes (Section 6.1). In these examples, we illustrate the regularizing effect of the adjoint method, and we show that no guarantee in terms or registration accuracy can be expected from the procedure. We also evaluate the efficiency of regularization approaches involving the L^2 penalty or the L^∞ constraint.

To evaluate the relevance of our approach in augmented surgery, we produce registration results using the Sparse Data Challenge dataset (Section 6.2). The dataset, generated from phantom data, consists of a liver mesh and 112 point clouds representing as many deformed configurations. Registration error statistics are provided by the organizers after we submit the reconstructed deformation computed for each point cloud. We obtained the second-best submission in the challenge, with an average 3.3 mm displacement error, far below the 5 mm required by our clinical collaborators.

In the last result, we try to estimate the net force applied by a robotic instrument on the liver boundary (Section 6.3). As several medical robots are shipped without force feedback, measuring the applied force is useful to avoid causing damages to the patient's liver. We create synthetic deformations and associated point clouds to simulate the liver being manipulated by a robotic instrument. Then we reconstruct a force distribution restricted to a small zone on the liver boundary (see Figure 1.4), and add up nodal forces to obtain a net force estimate. Each one of our five test cases features a time-dependent force (50 frames). We obtain an average force estimation error of 10 %, and an average

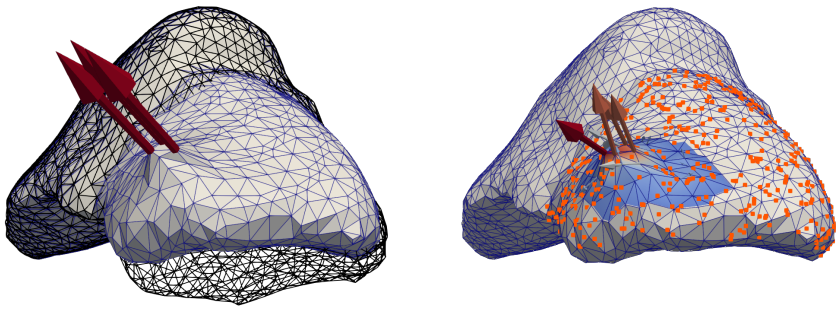


Figure 1.4: Left: synthetic deformation generated by a local force. Right: reconstructed deformation using the point cloud.

update time of 1 s, which is encouraging.

Contributions of the thesis

This thesis revolves around four main contributions.

- We identified the need for accurate modeling of the causes of displacement expressed in the literature and proposed an optimal control formulation to respond to this need.
- We proposed a mathematical analysis of the optimal control problem, including existence of solutions and first-order optimality conditions.
- We implemented a modular adjoint method[1] to solve registration problems, and validated our implementation using the Sparse Data Challenge dataset.
- We proposed an approach to evaluate a single net force applied by an instrument using our optimal control formulation.

The two last items are the object of the following conference article:

- Guillaume Mestdagh and Stéphane Cotin (2022). "An Optimal Control Problem for Elastic Registration and Force Estimation in Augmented Surgery". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022*. Ed. by Linwei Wang et al. Cham: Springer Nature Switzerland, pp. 74–83. ISBN: 978-3-031-16449-1. DOI: 10.1007/978-3-031-16449-1_8

Another article revolving around the mathematical analysis will soon be submitted to a peer-reviewed journal.

[1] Code available at https://github.com/gmestdagh/adjoint-elastic-registration.

Chapter 2

State of the art and problem modelling

2.1 Problem overview

2.1.1 A few facts about the liver

Let us begin with a few figures about the liver itself. Located just below the diaphragm, the liver is the heaviest internal organ in the human body. It plays a role in a large number of vital functions, including detoxificating blood, secreting hormones and regulating glycogen storage. Several ligaments hold the liver in place, including the falciform ligament and the left and right triangular ligaments (see Figure 2.1). As a consequence, the liver is not rigidly connected to a bone or a rigid support.

The liver weighs in average 1.5 kg (Gray and Lewis, 1918), and its dimensions are about 28 cm $\times$ 16 cm $\times$ 8 cm (Marchesseau et al., 2017). The matter that composes the liver, called the parenchyma, enters the category of soft tissues. It is nearly incompressible and its Young modulus amounts to approximately 10 kPa (Nikolaev and Cotin, 2020). Another consequence of liver tissues containing water is a density of $\rho = 1,070$ kg/m^3 (Niehues et al., 2010). The liver stiffness is increased by the presence of large blood vessels across the parenchyma.

2.1.2 The liver registration problem

A soft organ, the liver is constantly subject to large deformations. During a surgical operation, sources of such deformations include changes in the patient's position, breathing or organ manipulation by the surgeon. During minimally invasive surgery, the surgeon manipulates the liver using tools inserted through small incisions in the patient's abdomen and gets visual feedback on a screen using a laparoscopic camera. In this context, the constant motion and the limited field of view make it difficult for the surgeon to keep track of the position of the liver internal structures. Augmented reality systems have been designed to help the surgeon navigate through the surgical theater.

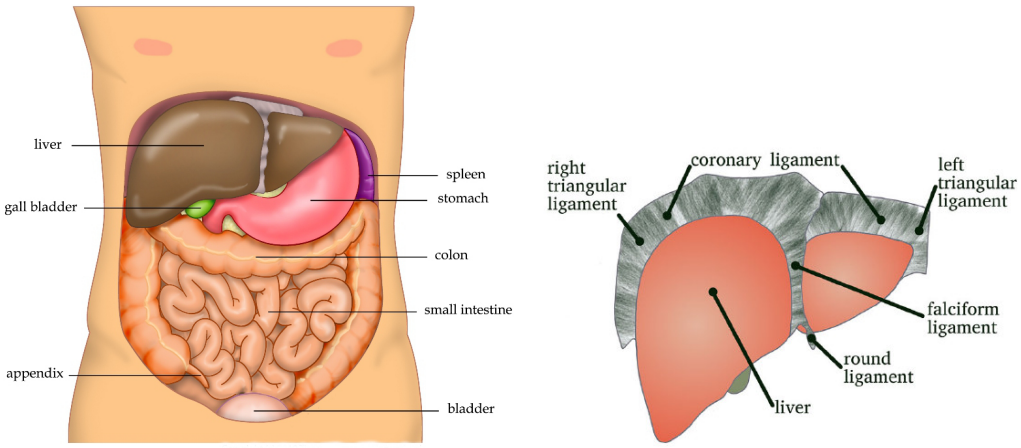


Figure 2.1: Left: the liver in the human body. (Ties van Brussels, https://commons.wikimedia.org/wiki/File:Anatomy_Abdomen_Tiesworks.jpg). Right: ligaments holding the liver (extracted from Nikolaev and Cotin, 2020). The falciform splits the parenchyma between the right lobe (on the left) and the left lobe (on the right).

The objective of augmented reality systems is to track the deformations of the patient's liver during the surgical operation, and provide the medical staff with an updated view of the organ, along with its internal structures such as tumors or blood vessels. In this project, we follow the pipeline proposed by Haouchine et al. (2013). In their pipeline, a pre-operative liver geometrical model and intra-operative images from the laparoscopic camera are combined through an elastic registration procedure to produce the updated liver tri-dimensional image.

On the one hand, pre-operative data consist of a liver model extracted from tomographic images. Such images accurately depict the internal structures of the liver. A tetrahedral mesh representing the liver is segmented from images and is embedded with an elastic model. On the other hand, intra-operative data are extracted from laparoscopic images provided in real time by the camera. A point cloud representing the part of the liver surface that is in the camera field of view is created. Figure 2.2 shows a pre-operative liver model, including tumors (green) and blood vessels (blue and red), along with a point cloud superimposed onto a laparoscopic intra-operative image. The augmented reality procedure follows three steps:

1. Intra-operative images are processed to extract a three-dimensional point cloud representing the current liver surface.

2. An initial rigid registration of the pre-operative model onto the point cloud is performed.

3. An elastic registration of the pre-operative biomechanical model is performed to

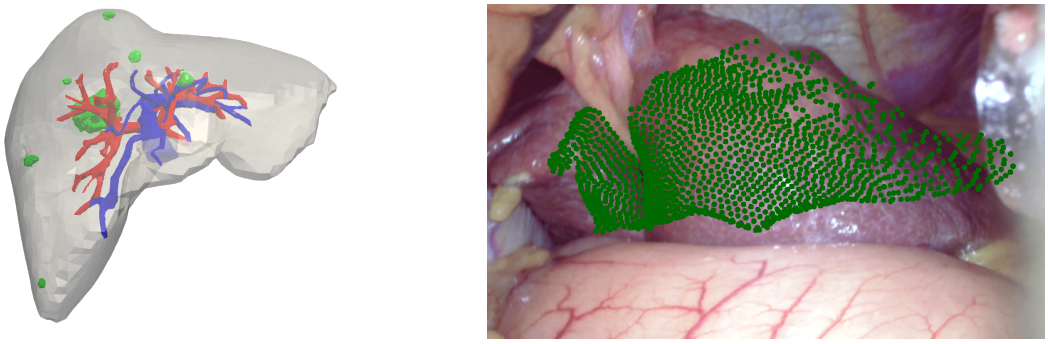


Figure 2.2: Left: liver geometry extracted from pre-operative tomographic images. Right: point cloud representing a part of the deformed liver surface.

match the intra-operative point cloud.

In this manuscript, we are interested in the third part, which is the elastic registration procedure. It consists in computing a displacement field defined on the liver mesh, so that the deformed liver surface matches the intra-operative point cloud. Our purpose is to reconstruct the displacement field we need to apply to the organ to obtain the current configuration (or deformed configuration) from a reference configuration (or initial configuration). Those two configurations are known through pre-operative images and intra-operative measures, respectively.

The elastic registration procedure raises several challenges. First, the augmented reality pipeline is supposed to run in a few tenth of milliseconds in order to provide real-time updates. Though we are not expecting to meet the real-time requirement in this work, we should keep in mind that involved procedures should be as fast as possible. In addition, an intra-operative point cloud is not sufficient to identify a unique displacement field. Therefore, the registration problem has a lot of solutions, and most of them are far from being physically relevant. As a consequence, some other information, such as physical hypotheses, should be added before we start thinking about registration accuracy. The chosen model and methods should result from a trade-off between the physical acceptability of the solution and the efficiency required by real-time execution.

Note that, in this study, we do not consider or compare to registration approaches based on machine learning (Brunet et al., 2019; Mendizabal et al., 2020). Though those modern methods are very efficient, we focus on getting some insight about the problem physics.

2.2 Elastic modelling of the liver

A key ingredient to include in a physics-based registration recipe is a good mechanical model. This model describes how the organ is deformed when forces are applied onto it. By providing an accurate description of the liver and its surroundings, one expects their model to behave like the real system in a direct simulation.

In this section, we describe a few models used in the literature to describe the liver. Beforehand, a short reminder about static elasticity problems seems in order. We first introduce the formulation of the elasticity system as a minimization problem, and then we describe a few common elastic models.

2.2.1 Deformation energy and forces

In this section, we give some context about static elasticity problems. We are only interested in static elasticity, which means that inertial effects are not taken into account. The phenomena we want to capture by performing liver registration are the deformations generated by external forces that change along time. Those external forces, whether they are created by the physician's gesture, contact with other organs of breathing movements, evolve very slowly compared to inertial forces. On the other hand, by considering a Young modulus $E = 10$ kPa, a density $\rho = 1070$ kg/m^3 and a Poisson ratio $\nu = 0.49$, we obtain that the velocity of mechanical waves in the liver reads

$$c = \sqrt{\frac{E}{2\rho(1+\nu)}} = 1.8 \text{ m/s}.$$

As the average liver length is approximately 28 cm, the typical duration of dynamical effects is 0.16 s. Therefore, inertial effects, such as vibrations or mechanical waves propagating across the liver parenchyma, are too fast to be captured by the augmented reality system.

An elastic material is a material that returns to its initial shape when left at rest. It is defined by an elastic energy that only depends on the current local deformation of the material (Ciarlet, 1988, Chapter 4). In particular, the elastic energy does not depend on the material velocity, the amplitude of its displacement or any past configuration. We consider a domain $\Omega \subset \mathbb{R}^3$ filled with an elastic material. The global deformation energy is obtained by accumulating the local energy associated to local deformation over the whole material. Namely, when a displacement field $u \in H^1(\Omega, \mathbb{R}^3)$ is applied to the material, the point $x \in \Omega$ is transported to $x + u(x)$ and the global elastic energy reads

$$W(u) = \int_\Omega w(\nabla u)\, \mathrm{d}x,$$

where $w \in C^2(\mathbb{R}^{3\times 3}, \mathbb{R})$ represents the local deformation energy. Here w is a function of ∇u, which means that adding a translation displacement to u does not change the deformation energy. Actually, the deformation energy is insensitive to any rigid displacement, including rotations. We discuss this aspect below when describing usual elastic models. The deformation energy is minimal when $u = 0$, i.e. the material is in its rest configuration. The energy cost to bring the material from its rest shape to a given displacement u is $W(u) - W(0)$.

External actions that apply onto the solid are represented by forces. Forces are a convenient way to provide a summary of various types of interaction by describing the first-order variation of the total energy around a given displacement u. Examples of

forces include gravity forces, that derive from a gravitational potential, or contact forces, that enforce a non-interpenetration condition between two materials. In the context of static elasticity, a force is represented by a linear potential energy. For instance, a surface force distribution $g \in L^2(\partial\Omega, \mathbb{R}^3)$ and a volume force distribution $f \in L^2(\Omega, \mathbb{R}^3)$ are represented by the energy terms

$$-\langle g, u\rangle = -\int_{\partial\Omega} g \cdot u \,\mathrm{d}s \qquad \text{and} \qquad -\langle f, u\rangle = -\int_{\Omega} f \cdot u \,\mathrm{d}x,$$

respectively.

Stable equilibrium configurations correspond to displacement fields where the system energy is at a minimum. In other words, the displacement generated by f and g is a solution to the optimization problem

$$\min_{u\in\mathcal{U}} \; W(u) - \int_{\Omega} f \cdot u \,\mathrm{d}x - \int_{\partial\Omega} g \cdot u \,\mathrm{d}s. \tag{2.1}$$

In (2.1), $\mathcal{U}$ is the set of admissible displacements, or feasible set. Typical feasible sets involve a Dirichlet condition on a subset $\partial\Omega_{\mathrm{D}} \subset \partial\Omega$ of the boundary to ensure the existence of a solution. From now on, we assume that a Dirichlet condition is applied, and we use the notation

$$\mathcal{U} = H^1_{\mathrm{D}}(\Omega) = \left\{ u \in H^1(\Omega, \mathbb{R}^3) \mid u = 0 \text{ on } \partial\Omega_{\mathrm{D}} \right\}.$$

The strong formulation of the elasticity system can be derived (at least formally), from the first-order optimality conditions of (2.1). Note that, for $v \in H^1_{\mathrm{D}}(\Omega)$ a small displacement,

$$\langle \mathrm{d}W(u), v\rangle = \int_{\Omega} w'(\nabla u) : \nabla v \,\mathrm{d}x = \int_{\partial\Omega} (w'(\nabla u)n) \cdot v \,\mathrm{d}s - \int_{\Omega} \mathrm{div}(w'(\nabla u)) \cdot v \,\mathrm{d}x,$$

where the last expression is a consequence of Green's formula. Thus, an equilibrium displacement u satisfies the variational equality

$$\forall v \in H^1_{\mathrm{D}}(\Omega) \quad \int_{\Omega} \left[-\mathrm{div}(w'(\nabla u)) - f\right] \cdot v \,\mathrm{d}x + \int_{\partial\Omega} (w'(\nabla u)n - g) \cdot v \,\mathrm{d}s = 0,$$

which yields the strong formulation

$$\begin{cases} -\mathrm{div}(w'(\nabla u)) &= f \quad \text{in } \Omega \\ w'(\nabla u)n &= g \quad \text{on } \partial\Omega_{\mathrm{N}} \\ u &= 0 \quad \text{on } \partial\Omega_{\mathrm{D}}, \end{cases} \tag{2.2}$$

where $\partial\Omega_{\mathrm{N}} = \partial\Omega \setminus \partial\Omega_{\mathrm{D}}$. In the context of elasticity, $w'(\nabla u)$ is called the *first Piola-Kirchhoff stress tensor*.

2.2.2 Common elastic models

A wide variety of elastic models exists in the literature, with very different properties (see Wex et al., 2015, for examples in biomechanics). Though the elastic problems we consider can be formulated using (2.1) or (2.2), the numerical method used to compute a displacement depends on the model complexity. In this section, we introduce three of the most common elastic models. In those models, the material resistance to deformation and volume changes are characterized by the *Lamé parameters*, which are deduced from the Young modulus E and the Poisson ratio ν by

$$\mu = \frac{E}{2(1+\nu)} \qquad \text{and} \qquad \lambda = \frac{E\nu}{(1+\nu)(1-2\nu)}. \tag{2.3}$$

Saint Venant-Kirchhoff model

The simplest nonlinear elastic model is the Saint Venant-Kirchhoff model (St Venant, 1844; Kirchhoff, 1852) which applies a quadratic penalty on the nonrigid part of the displacement.

Let us introduce a few notations. When a displacement u is applied to the elastic solid, a particle initially located at $x \in \Omega$ is transported to $x + u(x)$. The gradient of $\mathrm{Id} + u$, denoted by $F = (I + \nabla u)$, describes the local transformation around x, including nonrigid deformation and rotation. The tensor that is penalized in the Saint Venant-Kirchhoff energy is the *Green-Saint Venant strain tensor*

$$e(u) = \frac{F^{\mathrm{T}}F - I}{2} = \tfrac{1}{2}\left(\nabla u + \nabla u^{\mathrm{T}} + \nabla u^{\mathrm{T}}\nabla u\right).$$

Note that the rotation component of F is completely evacuated when $F^{\mathrm{T}}F$ is considered, and thus only nonrigid deformations such as shear and volume changes are seen by the Green-Saint Venant tensor.

The local deformation energy is characterized by Hooke's tensor A, defined by $A\varepsilon = 2\mu\varepsilon + \lambda \operatorname{tr}(\varepsilon) I$, with μ and λ from (2.3). Its expression reads

$$w(\nabla u) = \tfrac{1}{2} Ae(u) : e(u) = \mu |e(u)|^2 + \frac{\lambda}{2} (\operatorname{tr} e(u))^2.$$

Though it is a nonlinear model, the Saint Venant-Kirchhoff model involves a simple energy function and is not relevant for large deformations. In particular, it allows the deformation gradient F to be an indirect transformation, which translates as inverted triangles in a finite element simulation. On the other hand, solving a static problem requires to use a Newton method, which comes at a high computational cost. For these reasons, the Saint Venant-Kirchhoff model is not relevant for applications where fast computations are required or large deformations are involved.

Linear elasticity

The linear elastic model is obtained by performing a quadratic approximation of the Saint Venant-Kirchhoff energy around $u = 0$. Namely, the Green-Saint Venant tensor

$e(u)$ is replaced by the *linearized strain tensor* $\varepsilon(u) = \frac{1}{2}\left(\nabla u + \nabla u^{\mathrm{T}}\right)$ and the linear elastic energy reads

$$w(\nabla u) = \tfrac{1}{2} A\varepsilon(u) : \varepsilon(u) = \mu|\varepsilon(u)|^2 + \frac{\lambda}{2}\mathrm{div}(u)^2.$$

When a Dirichlet condition is involved, the existence and uniqueness of the solution for smooth domains is a consequence of the Lax-Milgram theorem and Korn's inequality (see a detailed explanation in Allaire, 2007, Section 2.3). The solution of (2.1) solves the linear partial differential equation

$$\begin{cases} -\mathrm{div}(A\varepsilon(u)) &= f \quad \text{in } \Omega_0 \\ A\varepsilon(u)\cdot n &= g \quad \text{on } \partial\Omega_{\mathrm{N}} \\ u &= 0 \quad \text{on } \partial\Omega_{\mathrm{D}}. \end{cases}$$

From a numerical point of view, the linear elastic model allows very fast computations, since a displacement is computed by solving a positive definite sparse linear system. Methods to solve a positive definite linear system include the Conjugate Gradient method (Hestenes and Stiefel, 1952) or direct methods involving the Cholesky factorization (see Nocedal and Wright, 2006, Appendix A). Direct methods are especially relevant when the elasticity system is solved several times, as the Cholesky factorization can be computed once and stored as long as needed.

Due to its definition, the linear elastic model only provides a first-order approximation of the displacement with respect to the force distribution. As a consequence it is only relevant for small displacements (which is a very subjective notion), typically when the material is very stiff compared to the forces applied onto it.

Neo-Hookean model

The Neo-Hookean model (Rivlin and Rideal, 1948) is more complex than the Saint Venant-Kirchhoff model, but it is far more relevant to describe soft materials undergoing large deformations. The local deformation energy penalizes the Green-Saint Venant deformation tensor $e(u)$ and the Jacobian determinant $J = \det(F)$. Though several formulations exist (Smith et al., 2018), we stick to a version previously used in the Mimesis team and software (Brunet, 2020),

$$w(\nabla u) = \mu\,\mathrm{tr}(e(u)) - \mu\ln(J) + \frac{\lambda}{2}(\ln J)^2.$$

A key feature of Neo-Hookean models is the presence of terms involving $\ln(J)$ in the expression. These terms are defined only when the local deformation F is direct (i.e. $\det(F) > 0$) and grow toward infinity as F becomes more and more singular. As a consequence, resulting displacement fields only involve direct local deformations, and deformed meshes do not exhibit inverted triangles.

In Figure 2.3, we apply a uniform volume force distribution $f(x) \propto (-4, 1)$ onto a two-dimensional rectangular beam, and we compute the resulting deformation using the

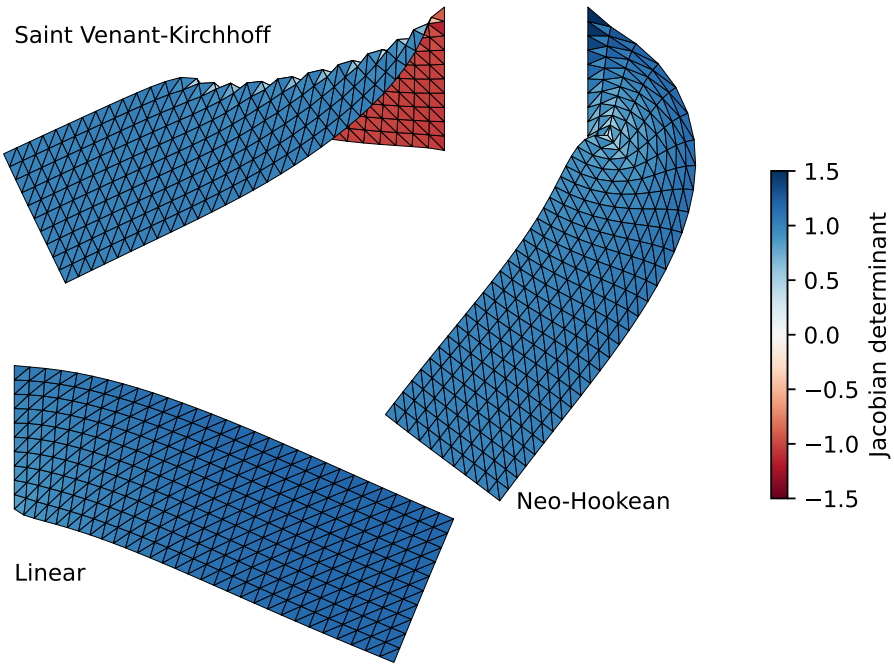


Figure 2.3: Resulting deformations for the uniform force distribution, using the linear model (bottom left), the Saint Venant-Kirchhoff model (top left) and the Neo-Hookean model (right).

three models described in this section. The color map represents the local Jacobian determinant of the transformation. Namely, inverted triangles are represented in red while non-inverted triangles are represented in blue. First, the linear deformation, which is just an amplification of an infinitesimal deformation, features a significant dilatation, which is common with linear deformations. In addition, rotations are penalized by the linear model, as the Green-Saint Venant tensor $e(u)$ is replaced with its linearized version. As a consequence, the resulting displacement does not involve a sufficient rotation. Rotation is better represented in the Saint Venant-Kirchhoff displacement, especially at the end of the beam. However, the displacement also features inverted triangles close to the clamped boundary. Finally, the Neo-Hookean displacement looks more realistic as it features a large rotation, and contains no inverted triangles. Even though some overlapping appears between triangles, every triangle is transformed in a direct way. Thus, the Neo-Hookean model is relevant for large deformations without self-contact.

Solving numerically a neo-Hookean problem using a Newton method might be challenging, as the problem residual is not defined for non-feasible points ($J < 0$). As a consequence, the solver should only generate iterates that lie in the feasible region.

Remark 2.1. Let us add a word about the existence of solutions for nonlinear models. The neo-Hookean model, along with some more generic models, falls into the category of polyconvex stored energy functions (Ball, 1976). For such functions, the existence

of a minimizer is guaranteed. In the case of Saint Venant-Kirchhoff materials, Raoult (1986) proves that the deformation energy function is not polyconvex, and proving the existence of minimizers is more difficult (see Ciarlet, 1988, Section 7.7 Remark (5) and Exercise 7.16). Though, it is possible to prove that the strong formulation (2.2) has solutions, using the implicit functions theorem (Ciarlet, 1988, Chapter 6).

2.2.3 Mechanical models in liver simulation

We now review a few models that were proposed in the computer-assisted intervention literature to model the liver and its surroundings. Such models should not only involve a constitutive law to describe the liver parenchyma, but also account for the interactions between the liver and its surroundings.

We first focus on constitutive laws used to model the liver parenchyma. Liver tissues, as well as most biological tissues, enter the category of soft materials and their simulation requires a hyperelastic model. In their extensive review Marchesseau et al. (2017) indicate that, in general, the liver parenchyma exhibits an isotropic behavior, and thus can be represented using an isotropic model. Used models in the literature include the classic Saint Venant-Kirchhoff (Delingette and Ayache, 2004) and Neo-Hookean (Miller et al., 2007) models, but also the more generic Ogden model (Nikolaev and Cotin, 2020), defined by

$$w(\nabla u) = \sum_{p=1}^{N} \frac{\mu_p}{\alpha_p} \left(\lambda_1^{\alpha_p} + \lambda_2^{\alpha_p} + \lambda_3^{\alpha_p} - 3 \right),$$

where $\lambda_1, \lambda_2, \lambda_3$ are the principal stretches and N, α_p, μ_p are material parameters. However, hyperelastic models often require heavy calculations, too heavy for real-time registration. A popular compromise between model accuracy and computational efficiency, the linear co-rotational model (Nesme et al., 2005) offers the possibility to simulate nonlinear deformations while solving linear systems. Adopters of the co-rotational model for liver registration include Suwelack et al. (2014), Plantefève et al. (2016) and Peterlík et al. (2018), among others.

The liver parenchyma is traversed by large blood vessels that are stiffer than liver tissues, resulting in an additional resistance to deformations. Peterlík et al. (2012) propose a liver biomechanical model including the mechanical properties of blood vessels. To meet the real-time requirement, the authors use the co-rotational model for the parenchyma, coupled with a beam model for blood vessels.

Another critical part of the mechanical model is the interaction between the liver and its surroundings. Though it is feasible to perform ex-vivo measures on the parenchyma to estimate mechanical parameters, estimating the parameters of ligaments and tissues that hold the liver is more difficult. As a consequence, boundary conditions applied to the liver mesh are often simple. To produce their synthetic validation data, Peterlík et al. (2018) apply homogeneous Dirichlet conditions on the parts of the boundary attached to ligaments and at the entry of the portal and the hepatic veins. Ozkan and Goksel (2018) consider that the liver is embedded in a larger elastic continuum, and set the boundary conditions as an approximation of the behavior of surrounding tissues. Peterlík

et al. (2017) represent ligaments surrounding the liver as linear springs holding the organ. They use a data assimilation procedure involving a Kalman filter to progressively estimate the associated stiffnesses. Their model was recently improved by Nikolaev and Cotin (2020), who use nonlinear springs and a better initialization procedure.

2.3 A review of some registration methods

We now turn to elastic registration methods. Based on a mechanical model of the liver, these methods apply forces onto the liver virtual model to generate a deformation. The choice of a registration method is as critical as the choice of a mechanical model, as the forces applied by the method determine whether the resulting deformation is physically relevant or not.

Before we focus on registration methods for augmented surgery, we give a bit of context about image registration methods, as those methods have been around for several decades and have probably inspired the augmented surgery community.

2.3.1 Image registration methods based on a mechanical model

Image registration is a vast and well established domain of image processing. In the medical domain, image registration is used to account for anatomy variability between patients (Christensen et al., 1997), to combine images from different modalities (D'Agostino et al., 2003), or to match data with an atlas (Wang and Staib, 2000). In this section, we describe some registration methods that involve a mechanical model, namely a solid or a fluid model. The review by Sotiras et al. (2013) adopts a wider scope and includes methods based on interpolation and geometry.

Let us consider an image registration problem involving two liver images. Each image is represented by a density function (1 for bone, 0 for void) with compact support defined on $\mathbb{R}^d$, where $d = 2$ or 3 is the image dimension. The template image, represented by the distribution t, is for instance an atlas where each part of the liver is identified. The source image, represented by the distribution s, may be a patient-specific image. During the registration process, a displacement field u is computed, so that the point x in the template image is associated to the point $x + u(x)$ in the source image. In other words, the template image is transported by a displacement field u to look like the source image. Usually, the displacement field u is the solution to a minimization problem, where the cost function F measures the discrepancy between t and $s \circ (\mathrm{Id} + u)$. A common choice for F (see for instance Rabbitt et al., 1995; Dupuis et al., 1998) is the least-squares error

$$F(u) = \tfrac{1}{2} \int_{\mathbb{R}^d} \left[t(x) - s(x + u(x)) \right]^2 \mathrm{d}x.$$

While the choice of F depends on the available data and image modalities, the chosen physical model is related to the expected qualities of u, especially its smoothness and the possibility to obtain large deformations.

Models based on elasticity

In an elastic registration process, we consider that the domain is filled with an elastic material, so that applying a displacement field has a cost in terms of deformation energy. The functional F defined in (2.3.1) plays the role of an external potential energy, which applies a loading onto the material to generate a deformation. The resulting displacement u is solution to the problem

$$\min_u \; F(u) + W(u), \tag{2.4}$$

where W is the elastic deformation energy, which plays the role of a regularization term.

First mentions of elastic registration appear in Broit's doctoral thesis (1981), followed by a series of articles involving his advisor where elastic registration is applied to match brain images with an atlas (Bajcsy and Kovačič, 1989; Gee and Bajcsy, 1999). While these early works involve a linear elastic model, hyperelastic models are later adopted (Rabbitt et al., 1995) to account for a wider range of deformations. Subsequent developments around hyperelastic models include struggling against inconsistent deformations, computing the reverse transformation along with the direct one, or writing more efficient numerical schemes.

Models based on viscosity

As they penalize the global deformation applied to the template image, elastic models may be too restrictive when large deformations are required. Christensen et al. (1996) prefer to use a model based on fluid mechanics. In a viscous model, the deformation energy is applied to the velocity field instead of the displacement field. As a consequence, the regularity of the final displacement field is preserved, but nothing prevents large deformations to occur.

The resulting registration is the consequence of an time-dependent displacement field $u : [0, t_\mathrm{f}] \times \Omega \to \mathbb{R}^d$. A particle initially at x is transported at $x + u(t, x)$ at time t. The evolution of u is driven in a Eulerian fashion by a velocity field $v : [0, t_\mathrm{f}] \times \Omega \to \mathbb{R}^d$ such that

$$\forall (t, x) \in [0, t_\mathrm{f}] \times \Omega \qquad \partial_t u(t, x) = v(t, x + u(x, t)).$$

To enforce the smoothness of the final displacement $u(t_\mathrm{f}, \cdot)$, v is the solution to the elasticity problem

$$\min_v \; \mathrm{d}F(u) \cdot v + W(u).$$

Several modern shape-matching methods rely on velocity-based models. In the approach by de Buhan et al. (2016), the authors use the shape optimization framework to register an unstructured mesh onto a fixed shape. Following the extension-regularization framework, they define the velocity field at each iteration as the solution to an elasticity problem. In the large deformation diffeomorphic metric mapping (LDDMM) framework (Dupuis et al., 1998; Beg et al., 2005), still used nowadays, the velocity field is solution to a variational problem involving the whole time interval.

2.3.2 Organ registration in augmented surgery

Augmented surgery represents a particular application of registration, as pre- and intra-operative data both represent the same organ. In this context, the mechanical model has a real meaning, as the computed deformation should represent the true organ deformation. In contrast, mechanical models only play a role of regularizer in the image registration methods mentioned above.

Forces resulting from a fictitious agent

A large number of organ registration procedures are inspired from the famous Iterative Closest Point algorithm by Besl and McKay (1992). Created for generic rigid registration, the Iterative Closest Point algorithm relies on orthogonal projections between the intra-operative point cloud $\Gamma = \{y_1, y_2, \cdots\}$ and the current liver mesh. At each iteration, the orthogonal projections $a_1, a_2, \cdots$ of points in Γ onto the liver mesh boundary are computed. Then, a rigid displacement u_{r}, solution to the least-square problem

$$\min_{u_{\mathrm{r}}} \ \tfrac{1}{2} \sum_j \|a_j + u_{\mathrm{r}}(a_j) - y_j\|^2,$$

is determined. This rigid displacement is applied to the liver mesh, projections onto the new mesh are computed and so on. The method stops when the least-square error does not decrease sufficiently between iterations. Clements et al. (2008) propose an improved version of the Iterative Closest Point algorithm, where landmark correspondence between the pre- and intra-operative data is taken into account.

Adaptations of the Iterative Closest Point algorithm to elastic registration often consist in introducting fictitious forces between the observed point cloud and the liver model. For Haouchine et al. (2013), these forces are generated by linear springs (see Figure 2.4). Using the same notation as above, the static elasticity problem solved at each iteration reads

$$\min_{u} \ \tfrac{1}{2} \sum_j k_j \|a_j + u(a_j) - y_j\|^2 + W(u),$$

where k_j is a stiffness parameter, which increases progressively along iterations to enforce the matching. Note how close to (2.4) this formulation is. Plantefève et al. (2016) replace the linear springs with a nonlinear attractive force. They model the registration process using dynamic elasticity to ensure the existence of a solution without Dirichlet boundary conditions. In a similar fashion, Suwelack et al. (2014) use an electrostatic attractive force between positively charged observed points and a negatively charged liver boundary.

Artificial forces sometimes appear in a more implicit way, taking the form of an optimization constraint in the mechanical model. To perform the registration, Peterlík et al. (2018) solve the optimization problem

$$\min_{u} \ W(u) \quad \text{subject to constraint} \quad C(u) = 0, \tag{2.5}$$

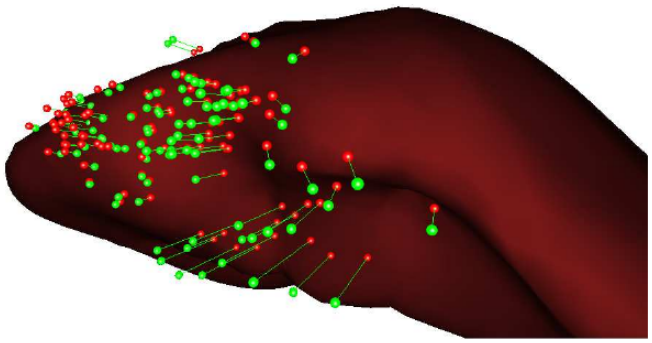


Figure 2.4: Observed data points (green) and their orthogonal projection onto the liver mesh (Haouchine et al., 2013).

where the constraint $C(u) = 0$ enforces correspondence between the current displacement field u and observed data. In an earlier work (Peterlík et al., 2014), the authors use the resulting Lagrange multipliers to estimate the boundary condition necessary to perform a registration between two configurations of the same organ. Approaches using similar sliding constraints are used in the context of pelvic system simulation (Courtecuisse et al., 2020) and brain-shift compensation (Morin et al., 2017).

Forces resulting from an inverse problem

Whether they derive from an imaginary potential energy or play the role of constraint-enforcing Lagrange multipliers, such fictitious forces do not correspond to a real physical phenomenon. They are created by data points, which do not really exist. As a consequence, in the methods above, forces that drive the registration are necessarily applied where data points are, without consideration for the true causes of displacement. If the registration problem involved landmarks located inside the organ, then registration forces would not even apply on the liver boundary. This often results in a poorly physically consistent displacement field, regardless of the direct model accuracy.

Rucker et al. (2014) address this issue in the context of open liver surgery. They control the imposed displacement on the posterior face of the liver (in contact with the support), while the anterior surface is observed by the camera. Heiselman et al. (2017) extend this work to laparoscopic surgery, by applying an imposed displacement on zones where ligaments hold the liver. Here, the authors use linear elasticity. In order to make computations faster, they create a basis of displacement fields associated to a range of non-homogeneous Dirichlet conditions. The authors use a similar displacement basis in a more recent work (Heiselman et al., 2020) to register the liver model onto slices obtained from ultrasound intra-operative measures. Özgür et al. (2018) propose a method for initial pose reconstruction when depth information is not available in laparoscopic images. They compute a rest shape by compensating the effect of gravity in pre-operative images, and they simulate pneumoperitoneum pressure and gravity load in intra-operative conditions to determine the final liver shape. Xie et al. (2022) use

surface-matching as a constraint in an optimization problem to update a force estimation between two successive frames. From a current force distribution g_n, they estimate a new force distribution g_{n+1} by solving the problem

$$\min_g \ \tfrac{1}{2}\|g - g_n\|^2 \quad \text{subject to constraint} \quad C(u_g) = 0,$$

where C is defined as in (2.5) and u_g is the displacement field generated by g. Note that, here, the constraint is not imposed in the direct problem, but in the inverse problem, so that no artificial protagonist is introduced into the direct problem.

2.4 Our contribution: an optimal control formulation

The approach studied in this manuscript falls in the category of registration methods based on an inverse problem. Methods described in the paragraph above lead to very accurate registration (see for instance the Vanderbilt result in Figure 6.8). However, these methods are often very integrated, specific to a given problem, and formulated in terms of matrix-vector operations. In addition, certain features, such as the linear displacement basis, are tailored for linear elasticity.

In this manuscript, we take a step backward and formulate the registration problem in the generic optimal control framework. Using an optimal control problem results in a very flexible formulation that can be easily modified to include new physical information or intra-operative data. The optimal control framework offers a wide range of generic tools to study and solve the problem.

Our contribution includes a mathematical study of the continuous registration problem and an implementation of an adjoint method to solve the problem numerically, even with a nonlinear elastic model. We also mention a possible application of our registration procedure to estimate a local force applied onto the liver.

In the remaining of this section, we introduce the notation we use all along the manuscript, and we introduce the optimal control formulation.

2.4.1 Mechanical model and data representation

The organ and its deformed configurations

We first define the notation for the liver constitutive law. Figure 2.5 illustrates the problem geometry. In its reference configuration, the liver is represented by an open domain $\Omega_0 \subset \mathbb{R}^d$ (with $d = 3$) filled with an elastic material. Due to its interactions with the environment, this continuum is subject to surface loadings, such as pressure or contact with surrounding tissues. As we neglect gravity, we do not consider body forces, though they may be easily accounted for as a parameter of the problem. The organ boundary $\partial\Omega_0$ splits into two parts, $\partial\Omega_{\mathrm{D}}$ and $\partial\Omega_{\mathrm{N}}$. On $\partial\Omega_{\mathrm{D}}$, a homogeneous Dirichet boundary condition applies, whereas $\partial\Omega_{\mathrm{N}}$ is subject to a Neumann condition involving a surface force distribution $g \in L^2(\partial\Omega_{\mathrm{N}}, \mathbb{R}^d)$. In our numerical experiments, Dirichlet

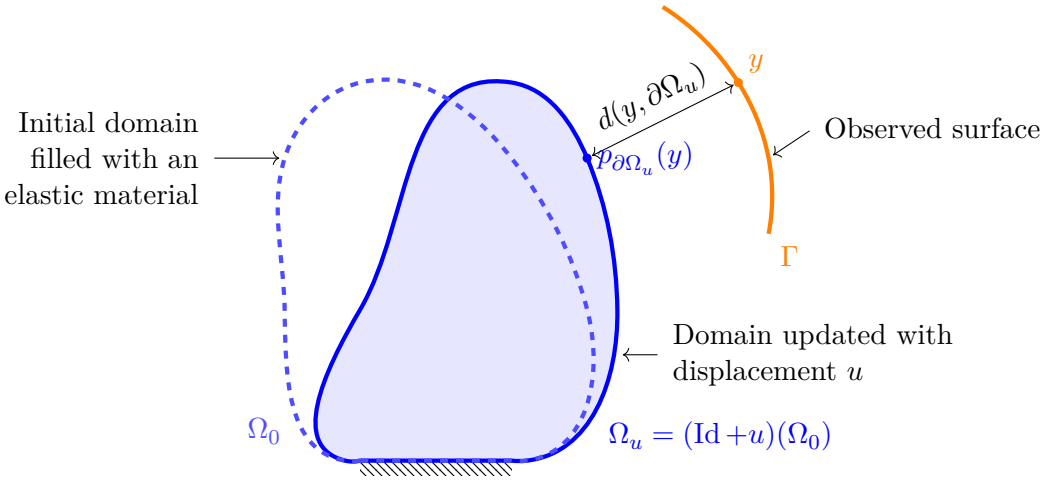


Figure 2.5: Problem geometry. The distance between a point $y \in \Gamma$ and its orthogonal projection onto $\partial\Omega_u$ is denoted by $d(y, \partial\Omega_u)$.

boundary conditions are set at the hepatic vein entry and in the region where the liver is in contact with the falciform ligament. The space of displacement fields is defined by

$$H^1_{\mathrm{D}}(\Omega_0) = \left\{ v \in H^1(\Omega_0) \,|\, v = 0 \text{ on } \partial\Omega_{\mathrm{D}} \right\},$$

and we denote by $u_g \in H^1_{\mathrm{D}}(\Omega_0, \mathbb{R}^d)$ the displacement field generated by a given force distribution $g \in L^2(\partial\Omega_{\mathrm{N}})$. In the case of linear elasticity, u_g solves the partial differential equation

$$\begin{cases} \operatorname{div}(A\varepsilon(u)) &= 0 \quad \text{in } \Omega_0 \\ u &= 0 \quad \text{on } \partial\Omega_{\mathrm{D}} \\ A\varepsilon(u) \cdot n &= g \quad \text{on } \partial\Omega_{\mathrm{N}}. \end{cases} \tag{2.6}$$

Unless specified otherwise, we use a linear elastic model to describe the liver parenchyma. In addition, we do not model blood vessels in this study.

We denote by $\Omega_u = \{x + u(x), x \in \Omega_0\}$ the volume occupied by the deformed organ. To be consistent with the numerical implementation, we do not expect $(\mathrm{Id} + u)$ to be a bijection. In the numerical framework, the liver is represented by a mesh and its boundary is represented by a set of triangles, but we do not take measures to detect or prevent self-intersection. When a boundary triangle is inside the deformed mesh, it is still considered to be a boundary element. For this reason, in the whole manuscript we always work with the surface $(\mathrm{Id} + u)(\partial\Omega_0)$, which is the boundary $\partial\Omega_0$ transported by a displacement u, even when it does not match with the boundary of Ω_u. However, the range of deformations expected in the application case usually does not include self-contact. In that case, both definitions coincide.

Data representation and objective function

While the reference configuration Ω_0 is known through pre-operative images, the current position shall be estimated with the help of data provided during the intervention by a measuring device. In the continuous problem, we represent observed data as a $(d-1)$-dimensional manifold Γ (a surface, actually). This data represents the current location of a part of $\partial\Omega_0$.

Due to the liver geometry, it is very likely that the surgeon has a vague idea of which part of the liver surface appears on the camera, and therefore which part of the pre-operative liver boundary should match Γ. The part of the liver boundary that is candidate to match Γ is denoted by $S_0 \subset \partial\Omega_0$. From now on, we assume that S_0 is compact. When S_0 is transported by a displacement u, we use the notation

$$S_u = (\mathrm{Id} + u)(S_0) \subset (\mathrm{Id} + u)(\partial\Omega_0).$$

In this context, the estimated displacement u should satisfy

$$\Gamma \subset S_u. \tag{2.7}$$

To enforce this condition, we compute u as the minimizer of a functional $J : C(\overline{\Omega}_0) \to \mathbb{R}$, which vanishes if and only if (2.7) holds. Moreover, it should not promote one part of S_0 in particular to match Γ.

The design of our objective function is inspired by the work by de Buhan et al. (2016). In their procedure, a meshed set Ω is transported along a velocity field to match a target shape Ω_T, known only through its signed distance function d_{Ω_T}, defined by

$$\forall y \in \mathbb{R}^d, \quad d_{\Omega_\mathrm{T}}(y) = \begin{cases} d(y, \partial\Omega_\mathrm{T}) & \text{if } x \notin \overline{\Omega}_\mathrm{T} \\ 0 & \text{if } x \in \partial\Omega_\mathrm{T} \\ -d(y, \partial\Omega_\mathrm{T}) & \text{if } x \in \Omega_\mathrm{T}. \end{cases}$$

The proposed objective function reads

$$J(\Omega) = \int_\Omega d_{\Omega_\mathrm{T}}(x)\,\mathrm{d}x. \tag{2.8}$$

It is minimized by using a gradient descent. The authors also considered the functional

$$P(\Omega) = \int_{\partial\Omega} d^2_{\Omega_\mathrm{T}}(x)\,\mathrm{d}s$$

to measure the discrepancy between the boundaries $\partial\Omega$ and $\partial\Omega_\mathrm{T}$ (Nardoni, 2017). However, this functional led to convergence difficulties, and for this reason they preferred to use (2.8).

Unlike de Buhan et al., we are not matching a full shape, but only an open surface, and we cannot use a signed distance function. Moreover we cannot penalize the distance to Γ on the whole boundary $\partial\Omega_0$, as only an unknown subpart of it should match Γ.

Finally, we use a least-squares functional similar to that in previous registration methods. Inspired from the Iterative Closest Point algorithm, the functional J reads

$$J(u) = \tfrac{1}{2} \int_\Gamma d^2(y, S_u)\, \mathrm{d}y. \tag{2.9}$$

Note that, as $d^2(y, S_u) = \min_{x \in S_u} \|x - y\|^2$, $J(u)$ vanishes if and only if $\Gamma \subset S_u$ up to a zero Lebesgue measure set. The expression (2.9) does not seem to promote a particular part of S_0 as a better candidate to match Γ, at least among feasible solutions u such that $J(u) = 0$. Though, evaluating the distance function involves orthogonal projections, which promotes the part of S_u that is closest to Γ.

In an improved version of the functional, pairings between points from Γ and points from S_u may be performed using something else than orthogonal projections. Other surface-matching techniques are studied in the registration community, including functional maps (Ovsjanikov et al., 2012) or varifolds (Antonsanti et al., 2021). For a review of surface-matching methods, see Sahillioğlu (2020).

Remark 2.2. A possible minor improvement in the expression for J consists in giving a weight to points of Γ, depending on their measure uncertainty. Taking in account different noise variances for the points in Γ could lead to an objective function of the form

$$J_\sigma(\Omega) = \tfrac{1}{2} \int_\Gamma \frac{d^2(y, S_u)}{\sigma^2(y)}\, \mathrm{d}y,$$

with $1/\sigma^2 \in L^\infty(\Gamma)$.

2.4.2 An optimal control formulation

While the role of J is to associate points from Γ and S_u to compute a discrepancy, the role of the optimal control formulation is to make sure that the chosen minimizer of J satisfies the physical model. A simple version of the optimal control problem reads

$$\inf_{g \in \mathcal{G}} J(u_g), \tag{2.10}$$

where $\mathcal{G} = L^2(\partial\Omega_\mathrm{N})$. The formulation (2.10) contains the minimal information necessary to compute a displacement field, and additional information should be added to reduce the set of solutions. For instance, new data, such as anatomical landmarks may be taken into account in the objective function, while information about the nature of forces that create the displacement should be included in the definition of $\mathcal{G}$. In the following paragraph, we consider two options to promote regular displacement fields despite the presence of noise in observed data. At the end of the section, we mention another formulation with a constraint on the displacement field.

Penalty on the control norm

If we solve (2.10), the algorithm finds a control such that almost every point of Γ is included in S_{u_g}. In the medical application, Γ is extracted from laparoscopic images using depth estimation. As a consequence, the intra-operative surface observation is polluted by noise. An exact matching is not what we look for, as it would result in a very irregular displacement field. A better result would be to make the liver boundary pass through the point cloud without matching every single point. To enforce control smoothness, we consider two options, namely a regularization term and a constraint. In both approaches, the goal is to prevent the control g from taking the large values it needs to reach every point in Γ.

In the regularized approach, a penalty term is included in the objective function, to prevent the control from oscillating too much. The regularized problem reads

$$\inf_{g\in\mathcal{G}} J(u_g) + R(g), \tag{2.11}$$

where $R(g)$ is a penalization term, for instance

$$R(g) = \tfrac{\alpha}{2}\int_{\partial\Omega_{\mathrm{N}}} \|g\|^2 \,\mathrm{d}s \quad \text{or} \quad R(g) = \tfrac{\alpha}{2}\int_{\partial\Omega_{\mathrm{N}}} \|\nabla_{\partial\Omega_{\mathrm{N}}} g\|^2 \,\mathrm{d}s,$$

and $\mathcal{G}$ is $L^2(\partial\Omega_{\mathrm{N}})$ or $H^1(\partial\Omega_{\mathrm{N}})$ depending on the chosen penalty. Though solving the regularized problem should result in a smoother control, it has a poor physical meaning, as the penalty is non-local. The algorithm could output a control with oscillations around Γ to save some cost on J, while creating an artificially smooth control on other parts of the boundary to save some cost on the penalty. Moreover, it is difficult to set a coefficient α based on physical considerations.

In the constrained approach, the feasible set is defined by

$$\mathcal{G}_M = \left\{ g \in L^\infty(\partial\Omega_{\mathrm{N}}) \mid \|g\|_{L^\infty(\partial\Omega_{\mathrm{N}})} \leqslant M \right\}. \tag{2.12}$$

Admissible force distributions should not take pointwise intensities greater than a upper bound $M > 0$. Though managing a pointwise constraint is more difficult than evaluating a penalty term, it seems easier to find a parameter M with a physical meaning. Namely, M may be chosen so that pressures greater than M cannot be expected in the human body.

For future reference, we write here the problem that is considered in Chapter 4. The problem, which involves both a L^2 penalty term an a L^∞ constraint, reads

$$\inf_{g\in\mathcal{G}_M} J(u_g) + \tfrac{\alpha}{2}\|g\|^2_{L^2(\partial\Omega_{\mathrm{N}})}. \tag{2.13}$$

Movement restriction on a subdomain

Information added into the optimal control formulation may also result from statistical observations. Assume that, in general, it is observed that the displacement field in a

subdomain of the liver does not exceed a certain magnitude. Namely, a subdomain $\omega_0 \subset \Omega_0$ is expected to remain close to its initial configuration during the surgical intervention. This information may be included in the problem by adding a constraint on the displacement norm on ω_0. The resulting problem reads

$$\inf_{g \in \mathcal{G}_M} J(u_g) \quad \text{subject to constraint} \quad \tfrac{1}{2} \int_{\omega_0} \|u_g\|^2 \, \mathrm{d}x \leqslant U, \tag{2.14}$$

where $U > 0$ is a parameter. Here, U can be estimated by measuring the average displacement amplitude in ω_0 during past experiments. Note that optimality conditions for problem (2.14) are derived in Chapter 4, but this approach is not implemented numerically, as we obtained satisfying results with the other approaches. However, it remains an interesting alternative to introduce physical information into the optimization problem.

Chapter 3

A few tools around shapes and mixed boundary conditions

One of our major contributions in this manuscript is to tackle the liver registration problem using a mathematical formalism. In this chapter, we introduce a few mathematical tools that should be of use to analyse the optimal control problem.

First, as the initial momentum of the project revolved around shape optimization, we give some definitions around shapes, shape functionals and shape optimization. The mathematical definition of a shape is not really different from what anyone would call a shape. A shape functional is a functional whose parameter is a shape, and shape optimization consists in minimizing a shape functional. However, shape optimization requires some dedicated tools, as inner products and other features from Hilbert spaces are not available in a set of shapes.

Then, to obtain insight about the discrepancy functional J and approaches involving closest points, we state some results around the signed distance function and orthogonal projections. Regularity properties of J are very similar to those of the signed distance function to a shape. They both depend on the uniqueness of an orthogonal projection of a point onto a shape. In addition, those two functions share the properties of applications defined using the min operator.

Finally, we introduce the notion of Gröger-regular set, which is useful to study partial differential equations with mixed boundary conditions. The Gröger-regular property requires sufficient regularity for the domain boundary, but also for the boundary between regions with Dirichlet boundary conditions and regions with Neumann boundary conditions. Using this framework, we state a few regularity results found in the literature, for the elasticity system, but also for elliptic systems. The results mentioned in this section are used to study the existence of a solution to our optimal control problem.

3.1 Shapes and shape functionals

Surface registration is a very geometric operation, and in this work we need to manipulate functionals that depend on a deformed shape. In addition, the regularity of solutions to

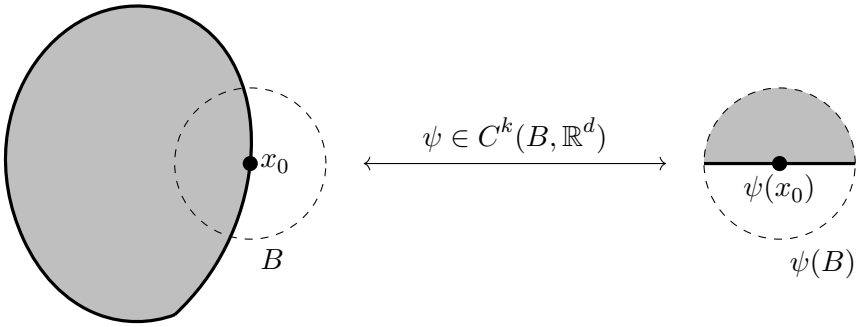


Figure 3.1: Local C^k regularity for a shape. The C^k-diffeomorphism ψ transforms the boundary part $\partial\Omega \cap B$ into a part of the plane of equation $x_d = 0$.

partial differential equations often depends on the domain regularity. For this, reason, we provide in this section some facts about shapes.

In this section, Ω is a domain, which means a bounded open subset of $\mathbb{R}^d$, and $\partial\Omega$ is its boundary. We might also use the term *shape* to emphasize the status of Ω as the variable of a shape functional, but it actually means the same as a domain. We first give a few definitions about shape regularity. Then, we introduce functionals that depend on a shape variable, and a tool to study their variations, Hadamard's boundary variation framework.

3.1.1 Regularity of a domain

In this section, we begin with some tools to describe a domain and obtain local information about its boundary. Then we briefly evoke the transposition of these notions in a numerical context.

Domains of class C^k

The local regularity of a shape Ω is assessed by considering the part of its boundary $\partial\Omega$ that lies in the vicinity of any point $x \in \partial\Omega$ and answering the following question: how regular is a diffeomorphism which transforms this part of the boundary into something flat, namely a part of an affine subspace of dimension $d-1$? When, for instance, a diffeomorphism of class C^k can be found for each point x of $\partial\Omega$, then Ω is said to have a C^k boundary (see Figure 3.1). A more formal definition (Gilbarg and Trudinger, 1977, Section 6.2) is given below.

Definition 3.1 (Domain of class C^k)**.** Let $0 \leqslant k \leqslant +\infty$. A bounded domain $\Omega \subset \mathbb{R}^d$ and its boundary are of class C^k when, at each point $x \in \partial\Omega$, there is a ball $B = B(x_0)$ and a C^k-diffeomorphism ψ from B to $D \subset \mathbb{R}^d$ such that

$$\psi(B \cap \Omega) \subset \mathbb{R}^d_+ \quad \text{and} \quad \psi(B \cap \partial\Omega) \subset \partial\mathbb{R}^d_+,$$

where $\mathbb{R}^d_+ = \left\{x \in \mathbb{R}^d \mid x_d > 0\right\}$. For $0 < \ell \leqslant 1$, domains of class $C^{k,\ell}$ are defined the same way.

The regularity of a domain Ω is useful to ensure regularity of solutions to partial differential equations defined on Ω, but also to study the variations of a shape functional when Ω is deformed. In the second situation, tools such as boundary normal or curvature sometimes need to be available to compute shape derivatives. These tools are defined right now.

We first define the boundary normal, which is characterized using the diffeomorphism ψ and its jacobian matrix (Delfour and Zolésio, 2011, Chapter 2, Remark 3.2).

Definition 3.2 (Outer normal)**.** Let Ω a domain with C^1 boundary, $x_0 \in \partial\Omega$ and ψ the diffeomorphism from Definition 3.1. Without loss of generality we assume $\psi(x_0) = 0_{\mathbb{R}^d}$, and we denote by $(e_1, \cdots, e_d)$ the canonical base of $\mathbb{R}^d$. The outer unit normal to $\partial\Omega$ at x_0 is defined by

$$n(x_0) = \frac{m(x_0)}{\|m(x_0)\|} \quad \text{where} \quad m(x_0) = -\left[\nabla\psi(x_0)^{\mathrm{T}}\right] e_d,$$

and $\nabla\psi$ is the Jacobian matrix of ψ.

Remark 3.1. To be convinced by the expression of $m(x_0)$, note that the tangent space to $\partial\Omega$ at x_0 is defined by $T_{x_0} = \operatorname{span}(\nabla\phi(0)e_1, \ldots, \nabla\phi(0)e_{d-1})$, where $\phi = \psi^{-1}$. Therefore we require $m(x_0)$ to satisfy

$$\begin{cases} \forall 1 \leqslant i \leqslant d-1 & m(x_0) \cdot \nabla\phi(0)e_i &= 0 \\ & m(x_0) \cdot \nabla\phi(0)e_d &= -1. \end{cases}$$

In other words,

$$\left[\nabla\phi(0)^{\mathrm{T}}\right] m(x_0) = -e_d \quad \text{and} \quad m(x_0) = -\left[\nabla\phi(0)^{-\mathrm{T}}\right] e_d = -\left[\nabla\psi(x_0)^{\mathrm{T}}\right] e_d.$$

To define curvature, it is easier to use the representation of C^k domains as local epigraphs, introduced below (Delfour and Zolésio, 2011, Chapter 2, Definition 5.2 and Theorem 5.2).

Definition 3.3 (Local epigraph)**.** A domain $\Omega \subset \mathbb{R}^d$ with nonempty boundary is said to be locally a C^k-epigraph, $k \geqslant 0$, when for each $x_0 \in \partial\Omega$, there is a ball B centered at 0, a hyperplane $S \subset \mathbb{R}^d$ with normal a and a C^k function $\varphi : S \cap B \to \mathbb{R}$ such that

$$\forall v \in B \quad x_0 + v \in \Omega \Leftrightarrow v \cdot a > \varphi(v_S) \quad \text{and} \quad x_0 + v \in \partial\Omega \Leftrightarrow v \cdot a = \varphi(v_S),$$

where v_S is the component of v along S.

Proposition 3.1. *For $k \geqslant 1$, a domain Ω has a C^k boundary if and only if it is locally a C^k-epigraph.*

For $k \geqslant 1$ and $0 < \ell \leqslant 1$, domains of class $C^{k,\ell}$ are characterized by their local epigraph representation. Note that the equivalence is not true for less regular domains, though. We can now state a definition for principal curvatures and mean curvature for C^2 domains.

Definition 3.4 (Curvature)**.** Let Ω a domain with C^2 boundary and $x_0 \in \partial\Omega$. Without loss of generality, we assume that the local epigraph representation of Ω around x_0 involves the tangent space to $\partial\Omega$ at x_0, i.e. $S = T_{x_0}$ and the inner normal vector $a = -n(x_0)$. Using the notation of Definition 3.3, the application $\varphi \in C^2(B \cap T_{x_0}, \mathbb{R})$ satisfies $\varphi(0) = 0$ and $\nabla\varphi(0) = 0$.

- The principal curvatures $\kappa_1, \ldots, \kappa_{d-1}$ of $\partial\Omega$ at x_0 are the eigenvalues of the Hessian matrix $\nabla^2\varphi(0)$.
- The mean curvature of $\partial\Omega$ at x_0 is defined by

$$H(x_0) = \sum_{i=1}^{d-1} \kappa_i = \Delta\varphi(0).$$

Remark 3.2. The curvatures are associated to the variations of the outer normal vector $n(x)$ as x moves along the boundary $\partial\Omega$. Namely, n and H satisfy

$$H = \mathrm{div}_{\partial\Omega}(n), \tag{3.1}$$

where $\mathrm{div}_{\partial\Omega}$ is the tangent divergence operator (Henrot and Pierre, 2005, Definition 5.4.7).

Remark 3.3. The definition of boundary curvature is still valid for domains of class $C^{1,1}$. In such domains, the normal vector n is a Lipschitz continuous function defined on $\partial\Omega$. Due to Rademacher's theorem, the normal vector function has derivatives at almost every point in $\partial\Omega$, and curvature can be defined at these points using (3.1).

Domains with Lipschitz boundary

Lipschitz regularity for sets is a weaker condition than C^k regularity for $k \geqslant 1$. It is based on local epigraphs involving Lipschitz continuous functions instead of C^k functions. We state a definition of Lipschitz sets in the case of domains with compact boundary (for instance bounded domains). When $\partial\Omega$ is not compact, several variants of the Lipschitz regularity exist, but they become all equivalent when $\partial\Omega$ is compact (Delfour and Zolésio, 2011, Chapter 2, Definition 5.1 and Theorem 5.1), which will always be the case in this work.

Definition 3.5 (Lipschitz domain)**.** A domain Ω with compact boundary is said to have a Lipschitz boundary when there exists a constant $c > 0$ such that Ω is locally the epigraph of a Lipschitz continuous function with Lipschitz constant c.

In other words, a Lipschitz domain is locally a $C^{0,1}$-epigraph involving functions φ that share a uniform Lipschitz constant. This condition is stronger than being a domain of class $C^{0,1}$: A Lipschitz domain is of class $C^{0,1}$ but some domains of class $C^{0,1}$ are not Lipschitz. Grisvard (2011, Chapter 1, Section 1.2) gives more details about the differences between those definitions.

Lipschitz domains are omnipresent when it comes to solving real-world problems. For instance, mechanical or thermal simulations involving industrial pieces with sharp edges or corners require to solve partial differential equations on Lipschitz domains. In particular, the normal n is defined almost everywhere on a Lipschitz boundary, as a consequence of Rademacher's theorem, and Green's formula is valid on Lipschitz domains (Grisvard, 2011, Chapter 1, Theorem 1.5.3.1), which is a good start for solving partial differential equations.

3.1.2 Shape functional and shape derivative

In this section, we introduce the notion of shape functional. A shape functional is a real-valued function which depends on a variable shape. Though continuity for such functionals can be defined using a topology on shapes sets, the absence of Hilbert space structure on shape sets makes it more difficult to define differentiability with respect to a shape. We first provide a definition for shape functionals, and then we introduce Hadamard's framework for shape differentiation.

In this document, a shape functional is defined as follows (Delfour and Zolésio, 2011, Chapter 7, Definition 2.1).

Definition 3.6 (Shape functional)**.** Let D a nonempty subset of $\mathbb{R}^d$ and $\mathcal{P}(D) = \{\Omega \mid \Omega \subset D\}$. A shape functional is an application $J : \mathcal{A} \to \mathbb{R}$, where $\mathcal{A} \subset \mathcal{P}(\mathcal{D})$, such that, for $\Omega \in \mathcal{A}$, and T a homeomorphism of $\overline{D}$ such that $T(\Omega) = \Omega$, the equality $J(\Omega) = J(T(\Omega))$ holds.

Common choices for D include $D = \mathbb{R}^d$ or D a compact subset of $\mathbb{R}^d$. The above definition emphasizes the independence of J with respect to continuous transformation that preserve Ω. In the case of open Lipschitz sets, it is common to apply a small deformation field onto the shape boundary to change the value of J.

Boundary variation framework

To study the variations of a shape functional J, it is convenient to use Hadamard's boundary variation method (see for instance Murat and Simon, 1976; Sokolowski and Zolésio, 1992; Henrot and Pierre, 2005, and references therein). To represent small variations of a bounded Lipschitz subset Ω of $\mathbb{R}^d$, we use the notation Ω_V to denote the set Ω transported by a displacement field $V \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$, namely

$$\Omega_V = (\mathrm{Id} + V)(\Omega) = \{x + V(x), x \in \Omega\}\,. \tag{3.2}$$

The definition for shape differentiability we give is stated in Allaire (2007, Chapter 6, Definition 6.15).

Definition 3.7 (Shape differentiability)**.** Let Ω a bounded subset of $\mathbb{R}^d$ with Lipschitz boundary.

The shape functional J is said to be shape differentiable in the direction $V \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$ whenever the application $t \mapsto J(\Omega_{tV})$ is differentiable at $t = 0$. If it exists, the directional shape derivative of J is defined by

$$\mathrm{d}J(\Omega)(V) = \lim_{t \searrow 0} \frac{J(\Omega_{tV}) - J(\Omega)}{t}. \tag{3.3}$$

In addition, J is said to be Fréchet differentiable at Ω whenever the application $V \mapsto J(\Omega_V)$ is Fréchet differentiable at 0 on $W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d)$.

This definition of shape differentiability, which involves Lipschitz continuous displacement fields, is appropriate for numerical computations, but several versions exist in the literature. Delfour and Zolésio (2011, Chapter 8, Definition 3.3) use a more general definition. They only consider C^∞ displacement fields with compact support and define the shape gradient as a distribution on $\mathbb{R}^d$. Sokolowski and Zolésio (1992, Chapter 2, Definition 2.19) state the definition with a continuous displacement field. Though, all these definitions consist in applying a displacement field to Ω.

We now state standard examples of shape differentiability results for shape functionals defined as integrals on a variable domain (Henrot and Pierre, 2005, Theorems 5.2.2 and 5.4.17). In the second proposition, we use the notation $C^{1,\infty} = W^{1,\infty} \cap C^1$.

Proposition 3.2. *Let $\Omega \subset \mathbb{R}^d$ a shape with Lipschitz boundary and $f \in W^{1,1}(\mathbb{R}^d, \mathbb{R}^d)$. The functional defined by $J : W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d) \to \mathbb{R}, V \mapsto \int_{\Omega_V} f \,\mathrm{d}x$ is differentiable at 0 and its differential reads*

$$\forall V \in W^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d) \quad \langle \mathrm{d}J(0), V \rangle = \int_{\partial\Omega} (V \cdot n) f \,\mathrm{d}s. \tag{3.4}$$

Proposition 3.3. *Let $\Omega \subset \mathbb{R}^d$ a shape with C^2 boundary and $g \in W^{2,1}(\mathbb{R}^d, \mathbb{R}^d)$. The functional defined by $J : C^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d) \to \mathbb{R}, V \mapsto \int_{\partial\Omega_V} g \,\mathrm{d}s$ is differentiable at 0 and its differential reads*

$$\forall V \in C^{1,\infty}(\mathbb{R}^d, \mathbb{R}^d) \quad \langle \mathrm{d}J(0), V \rangle = \int_{\partial\Omega_0} (V \cdot n) \left(\frac{\partial g}{\partial n} + Hg \right) \mathrm{d}s, \tag{3.5}$$

where H is the mean curvature of $\partial\Omega$.

A wide range of similar results exist in the literature. Standard shape functionals include functions based on the solution of a partial differential equation defined on Ω, or eigenvalues of differential operators such as the Laplacian. The interested reader is refered to Henrot and Pierre (2005, Chapter 5) and Delfour and Zolésio (2011, Chapter 9) for more information.

Structure theorem

Note that, in Proposition 3.2 and Proposition 3.3, the differential $\mathrm{d}J(0)$ has its support included in $\partial\Omega$. Moreover the shape derivatives depend only on the normal component of V on $\partial\Omega$. Indeed, the variation of J only depends on the movements of the shape boundary $\partial\Omega$. In addition, if Ω has a C^1 boundary and V only has a tangential component on $\partial\Omega$, then Ω is not changed by V at first order. This property of shape derivatives is stated in the structure theorem below (Henrot and Pierre, 2005, Theorem 5.9.2).

Theorem 3.1 (Structure). *Let $k \geqslant 1$, $\Omega \subset \mathbb{R}^d$ an open set with C^{k+1} boundary and J a shape functional. Assume that the application $\tilde{J} : C^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d) \to \mathbb{R}, V \mapsto J(\Omega_V)$ is Fréchet differentiable at 0. Then there exists a linear form ℓ, continuous on $C^k(\partial\Omega)$, such that*

$$\forall V \in C^{k,\infty}(\mathbb{R}^d, \mathbb{R}^d) \quad \left\langle \mathrm{d}\tilde{J}(0), V \right\rangle = \ell(V|_{\partial\Omega} \cdot n),$$

where n is the normal to $\partial\Omega$.

3.2 Orthogonal projection and signed distance function

Given a domain Ω with Lipschitz boundary and a point $y \in \mathbb{R}^d$, one can wonder whether y is inside or outside Ω, or how far from Ω y is. The signed distance function and the orthogonal projection are two tools to answer such questions. For a detailed and exhaustive presentation, the reader may refer to Delfour and Zolésio (2011, Chapter 7). Unless specified otherwise, $\|\cdot\|$ and $\cdot$ denote the Euclidean norm and scalar product on $\mathbb{R}^d$, respectively.

As we are working with the signed distance to $\partial\Omega$, we need a tool to differentiate functions based on a minimum. Such a theorem is provided by Danskin (1967, Chapter 3, Theorems I and II) in the case of functions defined on spaces of finite dimension.

Lemma 3.1 (Danskin's theorem). *Let $f : E \times K \to \mathbb{R}$ a continuous function, with E a vector space of finite dimension and K a compact subset of a Euclidean space. Assume that $(y, z) \mapsto \frac{\partial f}{\partial y}(y, z)$ exists and is continuous, and define $\varphi : E \to \mathbb{R}, y \mapsto \min_{z \in K} f(y, z)$.*

If y is a point of E and $w \neq 0$ is a direction, then φ has a directional derivative in the direction w at the point y, which reads

$$\lim_{t \searrow 0} \frac{\varphi(y + tw) - \varphi(y)}{t} = \min_{z \in K(y)} \left\langle \frac{\partial f}{\partial y}(y, z), w \right\rangle,$$

where $K(y) = \{z \in k \mid f(y, z) = \varphi(y)\}$ is the set of points in K which yield the minimum of $f(y, \cdot)$.

In addition, assume that $K(y)$ is a singleton. Then φ is differentiable in the sense of Fréchet at y. Last but not least, if $K(y)$ is a singleton for y in a neighborhood, then φ is C^1 on this neighborhood.

3.2.1 Generalities

Here we give a few definitions and properties about the signed distance function and the orthogonal projection.

Definition 3.8 (Signed distance function)**.** Let $\Omega \subset \mathbb{R}^d$ be an open set with Lipschitz boundary. The signed distance function with respect to Ω is defined by

$$\forall y \in \mathbb{R}^d, \quad d_\Omega(y) = \begin{cases} d(y, \partial\Omega) & \text{if } x \notin \overline{\Omega} \\ 0 & \text{if } x \in \partial\Omega \\ -d(y, \partial\Omega) & \text{if } x \in \Omega, \end{cases}$$

where $d(y, \partial\Omega) = \min_{x \in \partial\Omega} \|y - x\|$ is the Euclidean distance between y and $\partial\Omega$.

Definition 3.9 (Orthogonal projection)**.** Let Ω be an open domain. The set of orthogonal projections of y onto $\partial\Omega$ is defined by

$$\Pi_{\partial\Omega}(y) = \{x \in \partial\Omega \mid \|x - y\| = d(y, \partial\Omega)\}.$$

When $\Pi_{\partial\Omega}(y)$ is a singleton, we denote by $p_{\partial\Omega}(y)$ its element. The set of points which have several orthogonal projections onto $\partial\Omega$ is called the skeleton of $\partial\Omega$ and is denoted by Σ.

We state the following properties (Delfour and Zolésio, 2011, Chapter 7, Theorems 2.1 and 3.1) to better understand the relationship between the orthogonal projection and the signed distance function.

Proposition 3.4. *Let Ω be an open bounded Lipschitz set.*

1. *The signed distance function is Lipschitz continuous with Lipschitz constant 1. As a consequence, it is differentiable almost everywhere in $\mathbb{R}^d$ with $\|\nabla d_\Omega(y)\| \leqslant 1$.*
2. *For $y \in \mathbb{R}^d$, d_Ω^2 has directional derivatives at y and*

$$\forall w \in \mathbb{R}^d \quad \lim_{t \searrow 0} \frac{d_\Omega^2(y + tw) - d_\Omega^2(y)}{t} = \min_{z \in \Pi_{\partial\Omega}(y)} 2(y - z) \cdot w. \tag{3.6}$$

3. *A point $y \in \mathbb{R}^d$ has a unique projection $p_{\partial\Omega}(y)$ if and only if d_Ω^2 is Fréchet differentiable at y. In this case, $p_{\partial\Omega}(y) = y - \frac{1}{2}\nabla d_\Omega^2(y)$.*
4. *Σ is included in the set of points where d_Ω is not Fréchet differentiable, and Σ has zero Lebesgue measure in $\mathbb{R}^d$.*

Proof. 1. For $x, y \in \mathbb{R}^d$, a classic triangle inequality yields $|d(x, \partial\Omega) - d(y, \partial\Omega)| \leqslant \|y - x\|$. To prove the result with the signed distance function, we consider two cases, depending on whether x and y are on the same side of $\partial\Omega$. In the first case $x, y \in \overline{\Omega}$ or $x, y \notin \Omega$, and

$$|d_\Omega(x) - d_\Omega(y)| = |d(x, \partial\Omega) - d(y, \partial\Omega)| \leqslant \|x - y\|.$$

In the other case $x \in \overline{\Omega}$ and $y \notin \Omega$, then $[x, y] \cap \partial\Omega$ is nonempty. Considering $z \in [x, y] \cap \partial\Omega$, we obtain

$$\|y - x\| = \|y - z\| + \|z - x\| \geqslant d(y, \partial\Omega) + d(x, \partial\Omega) = |d_\Omega(y) - d_\Omega(x)|$$

and d_Ω is Lipschitz continuous on $\mathbb{R}^d$ with Lipschitz constant at most 1. The differentiability is a consequence of Rademacher's theorem (Evans and Gariepy, 2015, Theorem 3.2).

2. Note that $d^2_\Omega(y) = \min_{z\in\partial\Omega} \|z - y\|^2$, with $(z, y) \mapsto \|z - y\|^2$ a C^1 function, and $\partial\Omega$ a compact set. We obtain (3.6) by applying Lemma 3.1.

3. Assume that y has a unique projection point $p_{\partial\Omega}(y)$. Then Fréchet differentiability is also a consequence of Lemma 3.1, and we obtain $\nabla d^2_\Omega(y) = 2(y - p_{\partial\Omega}(y))$.

   Now, choose a point $y \in \mathbb{R}^d$ which has several projection points, and let $z_1 \neq z_2 \in \Pi_{\partial\Omega}(y)$. Note that the elements of $\Pi_{\partial\Omega}(y)$ are equidistant from y. Equation (3.6) and the Cauchy-Schwarz inequality yield

   $$\mathrm{d}d^2_\Omega(y)(z_1 - y) = \min_{z\in\Pi_{\partial\Omega}(y)} 2(y - z) \cdot (z_1 - y) = -2\|y - z_1\|^2.$$

   We compute the derivative in the opposite direction. As $z_2 \neq z_1$ and $\|z_1 - y\| = \|z_2 - y\|$, the Cauchy-Schwarz inequality yields $(y - z_2) \cdot (y - z_1) < \|z_1 - y\|^2$, and the derivative reads

   $$\mathrm{d}d^2_\Omega(y)(y - z_1) = \min_{z\in\Pi_{\partial\Omega}(y)} 2(y - z) \cdot (y - z_1) \leqslant 2(y - z_2) \cdot (y - z_1) < 2\|y - z_1\|^2.$$

   We conclude that $\mathrm{d}d^2_\Omega(y)$ is not a linear application and d^2_Ω is therefore not Fréchet differentiable at y.

4. This result is a consequence of points 1 and 3.

□

If $\partial\Omega$ is C^1, each point $y \in \mathbb{R}^d \setminus \Sigma$ can be associated to a unique pair of parameters consisting of its orthogonal projection onto $\partial\Omega$ and its signed distance to Ω. In this case, the normal vector is defined at every point of $\partial\Omega$, and thus y is given by the formula

$$y = p_{\partial\Omega}(y) + d_\Omega(y)\, n\left(p_{\partial\Omega}(y)\right),$$

where $n(x)$ is the outward normal vector at $x \in \partial\Omega$. If $\partial\Omega$ is only Lipschitz, some pairs of the form $(x, \ell) \in \partial\Omega \times \mathbb{R}$ might be associated with several points, especially if $\partial\Omega$ has an angular point at x. On Figure 3.2, y_2 and y_2' are associated to the same signed distance value and projection.

The points of Σ should be handled with care in this parametrization, as they can be associated with several sets of parameters. However, when computing derivatives, it is possible to choose one orthogonal projection depending on the direction with respect to which the derivative is computed.

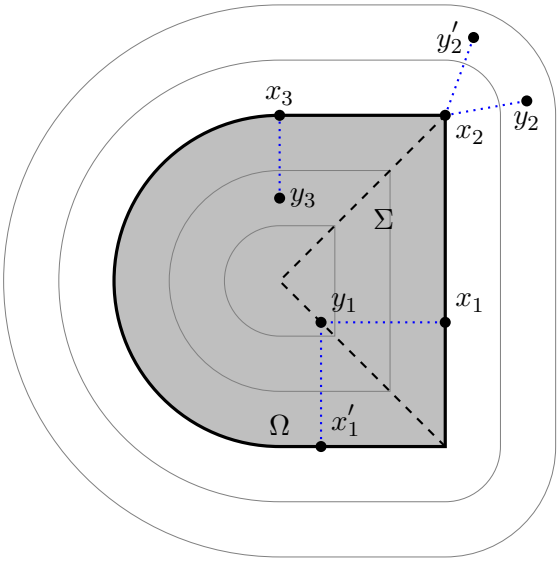


Figure 3.2: Set (in gray) with its skeleton (dashed line) and the signed distance level curves (gray lines). At the bottom, $y_1 \in \Sigma$, has several projections. On the right, note that $p_{\partial\Omega}(y_2) = p_{\partial\Omega}(y_2')$ and $d_\Omega(y_2) = d_\Omega(y_2')$. The gradient of d_Ω is defined at y_2 and y_2' but not at x_2. On the left, $\nabla^2 d_\Omega$ is not defined at y_3, as the curvature of $\partial\Omega$ is not defined at x_3.

To handle shape functionals based on the signed distance function, we need to know how the orthogonal projection and the signed distance evolve when y moves or when Ω is subject to a deformation. The following propositions gather some information found in Henrot and Pierre (2005), Delfour and Zolésio (2011) and Dapogny (2013).

3.2.2 Pointwise derivatives

The regularity of d_Ω away from Σ depends on the regularity of $\partial\Omega$, and its derivatives are defined using the normal vector to $\partial\Omega$ and its curvature. The following proposition (Dapogny, 2013, section 4.2.1) states sufficient conditions for first and second derivatives of d_Ω to exist.

Proposition 3.5. *Let Ω be a bounded open set with $C^{1,1}$ boundary, $y \in \mathbb{R}^d \setminus (\Sigma \cup \partial\Omega)$, and $x = p_{\partial\Omega}(y)$. The application d_Ω is differentiable in the sense of Fréchet at y and*

$$\nabla d_\Omega(y) = \frac{y - x}{d_\Omega(y)} = n(x),$$

where $n(x)$ is the outward normal to Ω at x.

In addition, if Ω has C^2 boundary and $y \in \mathbb{R}^d \setminus \overline{\Sigma}$, we denote by $(e_1, \cdots, e_{d-1})$ the directions of principal curvatures at x and $\kappa_1, \cdots, \kappa_{d-1}$ the associated principal curvatures.

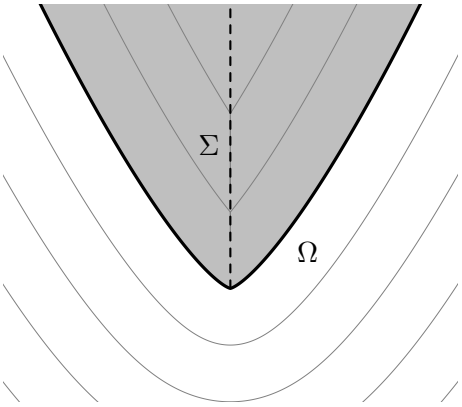


Figure 3.3: Set of equation $y < x^{4/3}$. The signed distance function is not C^1 at $(0, 0)$.

The signed distance is twice differentiable at y and

$$\nabla^2 d_\Omega(y) = \sum_{k=1}^{d-1} \frac{\kappa_k}{1 + \kappa_k d_\Omega(y)} e_k \otimes e_k. \tag{3.7}$$

Some examples of these conditions are illustrated on Figure 3.2. The signed distance function is not differentiable at y_1, as $y_1 \in \Sigma$. Moreover, the signed distance Hessian at y_3 is not defined, because the curvature of $\partial\Omega$ is not defined at x_3.

Remark 3.4. If d_Ω is differentiable at y but the normal is not defined at x, it is still possible to write $\nabla d_\Omega(y) = \frac{y-x}{d_\Omega(y)}$ (see y_2 and y_2' in Figure 3.2).

The following proposition (Delfour and Zolésio, 2011, Ch. 7, Th. 4.3) provides more specific information about the regularity of d_Ω in a vicinity of $\partial\Omega$.

Proposition 3.6. *Let Ω be a bounded domain.*

1. *If Ω is of class $C^{1,1}$, then d_Ω has $C^{1,1}$ regularity in a neighborhood W of $\partial\Omega$. Moreover,*
$$\forall y \in W \qquad \nabla d_\Omega(y) = n(p_{\partial\Omega}(y))$$
2. *If Ω is of class C^k, with $k \geqslant 2$, then d_Ω is C^2 in a neighborhood W of $\partial\Omega$ and d_Ω^2 is $C^{1,1}$ in W. An expression for the Hessian of d_Ω is given by (3.7).*

Working with a $C^{1,1}$ set, i.e. a set with bounded curvature, provides additional comfort as Σ is kept away from $\partial\Omega$. For instance the set of equation $y > x^{4/3}$ has C^1 boundary but not bounded curvature (See Figure 3.3). Thus, d_Ω is not differentiable on a neighborhood of $\partial\Omega$.

We finally state a proposition (Dapogny, 2013, Lemma 4.2) about the pointwise derivative of the orthogonal projection operator at points far from the skeleton of Ω.

Proposition 3.7. *Assume Ω has C^2 boundary, $y \in \mathbb{R}^d \setminus \overline{\Sigma}$ and $x = p_{\partial\Omega}(y)$. Denote by $(e_1, \cdots, e_{d-1})$ the directions of principal curvatures at x and $\kappa_1, \cdots, \kappa_{d-1}$ the associated principal curvatures. The orthogonal projection is Fréchet differentiable at y and its Jacobian matrix reads*

$$\nabla p_{\partial\Omega}(y) = \sum_{k=1}^{d-1} 1 - \frac{\kappa_k d_\Omega(y)}{1 + \kappa_k d_\Omega(y)} e_k \otimes e_k. \tag{3.8}$$

Proof. According to Proposition 3.5, the orthogonal projection of y is unique and given by the formula

$$p_{\partial\Omega}(y) = y - d_\Omega(y)\nabla d_\Omega(y).$$

As the hypotheses ensure the existence of the signed distance Hessian at y, the expression above is differentiable, with

$$\nabla p_{\partial\Omega}(y) = I - \nabla d_\Omega(y) \otimes \nabla d_\Omega(y) - d_\Omega(y)\ \nabla^2 d_\Omega(y).$$

To conclude, we use (3.7) and we notice that

$$I - \nabla d_\Omega(y) \otimes \nabla d_\Omega(y) = I - n(x) \otimes n(x) = \sum_{k=1}^{d-1} e_k \otimes e_k.$$

□

3.2.3 Shape derivatives

In this section, we discuss shape differentiability for the signed distance function to a given set. To be more specific, we only consider the squared signed distance d_Ω^2, which is actually the squared distance from $\partial\Omega$. We will see that there are some similarities between expressions for derivatives with respect to the shape and derivatives with respect to the point. Indeed, sometimes, deciding whether it is the point or the set that moves is only a question of reference frame.

In the following propositions (Dapogny, 2013, Section 4.2), we state Lipschitz continuity and directional differentiability with respect to the shape for d_Ω^2.

Proposition 3.8. *Let $y \in \mathbb{R}^d$, Ω a bounded Lipschitz domain, and*

$$B_{W^{1,\infty}}(0,1) = \left\{ w \in W^{1,\infty}(\Omega) \;\middle|\; \|w\|_{W^{1,\infty}} < 1 \right\},$$

the application $B_{W^{1,\infty}}(0,1) \to \mathbb{R}_+, w \mapsto \frac{1}{2} d^2(y, \partial\Omega_w)$ is Lipschitz continuous.

Proof. Let $w_1, w_2 \in B_{W^{1,\infty}}(0,1)$. Because $\|w_1\|_{W^{1,\infty}}, \|w_2\|_{W^{1,\infty}} < 1$, $(\mathrm{Id} + w_1)$ and $(\mathrm{Id} + w_2)$ are $W^{1,\infty}$ diffeomorphisms (Allaire, 2007, Chapter 6, Lemma 6.13).

Assume $d(y, \partial\Omega_{w_2}) \geqslant d(y, \partial\Omega_{w_1})$ and choose $x_1 \in \Pi_{\partial\Omega_{w_1}}(y)$. Then for $x_2 \in \partial\Omega_{w_2}$

$$d(y, \partial\Omega_{w_2}) - d(y, \partial\Omega_{w_1}) \leqslant \|y - x_2\| - \|y - x_1\| \leqslant \|x_2 - x_1\|.$$

The previous inequality holds for any $x_2 \in \partial\Omega_{w_2}$. In particular, we may consider $x = (\mathrm{Id}+w_1)^{-1}(x_1)$ and set $x_2 = x + w_2(x)$. With these notations,

$$\|x_2 - x_1\| = \|x + w_2(x) - x - w_1(x)\| = \|w_2(x) - w_1(x)\| \leqslant \|w_2 - w_1\|_{W^{1,\infty}}.$$

Note also that $w \mapsto d(y, \partial\Omega_w)$ is bounded on $B_{w^{1,\infty}}(0,1)$, with

$$d(y,\partial\Omega_w) = \min_{x\in\partial\Omega} \|x + w(x) - y\| \leqslant \min_{x\in\partial\Omega} \|x - y\| + \|w(x)\| \leqslant d(y,\partial\Omega) + 1.$$

We conclude the proof by noting that

$$\begin{aligned}\tfrac{1}{2}d^2(y,\partial\Omega_{w_2}) - \tfrac{1}{2}d^2(y,\partial\Omega_{w_1}) &= \tfrac{1}{2}\big(d(y,\partial\Omega_{w_2}) + d(y,\partial\Omega_{w_1})\big)\big(d(y,\partial\Omega_{w_2}) - d(y,\partial\Omega_{w_1})\big)\\ &\leqslant \big(d(y,\partial\Omega)+1\big)\|w_2 - w_1\|_{W^{1,\infty}}.\end{aligned}$$

□

Proposition 3.9. *Let $y \in \mathbb{R}^d$, and Ω a bounded Lipschitz domain. The application $w \mapsto \frac{1}{2}d^2(y,\Omega_w)$ defined on $W^{1,\infty}(\Omega)$ has directional derivatives at 0 with*

$$\lim_{t\searrow 0} \frac{\frac{1}{2}d^2(y,\Omega_{tw}) - \frac{1}{2}d^2(y,\Omega)}{t} = \min_{z\in\Pi_{\partial\Omega}(y)} w(z)\cdot(z-y). \tag{3.9}$$

Proof. We use the notations

$$g_y(t) = \tfrac{1}{2}d^2(y,\partial\Omega_{tw}) = \min_{x+tw(x)\in\partial\Omega_{tw}} \tfrac{1}{2}\|x + tw(x) - y\|^2,$$

and, for t small enough for tw to be a diffeomorphism,

$$K(t) = \Big\{x \in \partial\Omega \mid \tfrac{1}{2}d^2(y,\partial\Omega_{tw}) = \tfrac{1}{2}\|x + tw(x) - y\|^2\Big\} = (I+tw)^{-1}(\Pi_{\partial\Omega_{tw}}(y)). \tag{3.10}$$

For t close to 0, $K(t)$ is a nonempty subset of $\partial\Omega$. Also, the application $(t,x) \mapsto \frac{1}{2}\|x + tw(x) - y\|^2$ has a continuous partial derivative with respect to t. The result is a consequence of Lemma 3.1. □

3.3 Continuity and mixed boundary conditions

As mentioned earlier, in this document we are working with functionals based on geometric operations such as the orthogonal projection, and we consider shapes which are deformed using a displacement field. For this reason, it is important to give a meaning to the value of the displacement field at a single point. In other words, we need to manipulate displacement fields that are continuous and obtain convergences for the uniform convergence norm.

In this section, we enumerate a few continuity results for elliptic equations and for elasticity equations with mixed boundary conditions.

3.3.1 Some notation

From now on, we consider that Ω is an open domain with Lipschitz boundary $\partial\Omega$. We denote by $\partial\Omega_{\mathrm{N}}$ the open part of the boundary where a Neumann condition applies, and $\partial\Omega_{\mathrm{D}} = \partial\Omega \setminus \partial\Omega_{\mathrm{N}}$. The space $C(\overline{\Omega})$ contains functions that are continuous on $\overline{\Omega}$, endowed with the uniform convergence norm (the L^∞ norm). For $\kappa > 0$, we also define the space $C^{0,\kappa}(\Omega)$ of κ-Hölder continuous functions, endowed with the norm

$$\|u\|_{C^{0,\kappa}(\Omega)} = \|u\|_{L^\infty(\Omega)} + \sup_{x \neq y} \frac{|u(x) - u(y)|}{|x - y|^\kappa},$$

and for $1 \leqslant p \leqslant \infty$, $W^{1,p}(\Omega)$ is the Sobolev space of L^p functions whose first-order derivatives are also in $L^p(\Omega)$, where things are measured with the norm

$$\|u\|_{W^{1,p}(\Omega)} = \left(\int_\Omega |u|^p + |\nabla u|^p \, \mathrm{d}x \right)^{1/p}.$$

Those two spaces are interesting from the point of view of continuous solutions. To be more specific, for $p > d$, $W^{1,p}(\Omega)$ embeds continuously into $C^{0,\kappa}(\Omega)$ where $\kappa = 1 - d/p$ (Brezis, 1983, Theorem IX.12), and for $\kappa > 0$, the injection $C^{0,\kappa}(\Omega) \hookrightarrow C(\overline{\Omega})$ is compact (Brezis, 1983, Theorem IV.24), which means that a bounded sequence of $C^{0,\kappa}(\Omega)$ or $W^{1,p}(\Omega)$ (with $p > d$) converges uniformly, up to a subsequence. For this reason, we would like to see our solutions in such a space.

However, two main obstacles appear when it comes to obtaining sufficient regularity for solutions of elasticity systems with mixed boundary. First, the linear elastic system is not an elliptic equation, as its energy functional does not penalize rigid-body displacements. In addition, the presence of mixed boundary conditions degrades the regularity of solutions, even in C^1 domains. Gröger (1989) addressed the second difficulty by proposing a class of domains adapted to the study of partial differential equations with mixed boundary conditions. We now give a word about his framework.

Gröger regular sets for mixed-boundary conditions

To work with mixed boundary conditions, it is interesting to consider the subset $G = \Omega \cup \partial\Omega_{\mathrm{N}}$ of $\mathbb{R}^d$. The subset G contains all necessary information about the regularity of Ω and the distribution of Dirichlet and Neumann conditions on its boundary. Like in the definition of C^k domains, the definition of Gröger regular sets involves diffeomorphisms from a part of the surface to the unit ball of $\mathbb{R}^d$. Before we give a formal definition, we define the following subsets of $\mathbb{R}^d$:

$$\begin{aligned} B &= \{x \in \mathbb{R}^d \mid \|x\| < 1\} \\ B_+ &= \{x \in B \mid x_d > 0\} \\ D &= \{x \in B \mid x_d = 0\} \\ D_0 &= \{x \in D \mid x_1 < 0\} \end{aligned}$$

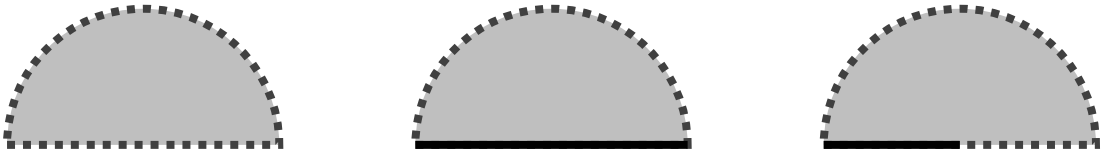


Figure 3.4: The elementary sets B_+, $B_+ \cup D$ and $B_+ \cup D_0$. A full line indicates that the boundary is included in the set. A set is Gröger-regular when its boundary locally looks like one of these sets, up to a Lipschitz diffeomorphism.

Definition 3.10 (Gröger-regular set)**.** Let Ω a bounded domain and $\partial\Omega_N$ a relatively open part of its boundary. We say that $G = \Omega \cup \partial\Omega_N$ is Gröger-regular when, for each $x \in \partial\Omega$, there is a neighborhood U of x and a Lipschitz diffeomorphism $\Phi : U \to B$ such that $\Phi(U \cap G)$ either coincides with B_+, or with $B_+ \cup D$ or with $B_+ \cup D_0$.

Figure 3.4 shows the sets B_+, $B_+ \cup D$ and $B_+ \cup D_0$. These three cases correspond to the case $x \in \partial\Omega \setminus \partial\Omega_N$, $x \in \partial\Omega_N$ and $x \in \partial(\partial\Omega_N)$, respectively. A simpler way to express this definition is that a bounded subset G is Gröger-regular when $\partial\Omega_N$ has a $W^{1,\infty}$ boundary in $\partial\Omega$ (Gröger, 1989). A very similar definition for Gröger-regular sets has been given independently by Droniou (2000, Condition (2.11)), where the author describes $\partial\Omega_N$ and $\partial\Omega_D$ as "well separated" when the regularity assumption is respected.

Haller-Dintelmann et al. (2009, Section 5) give a few more details about Gröger-regular sets in dimension 2 and 3. In particular, they note that in dimension 2, a domain G is Gröger-regular when $\partial\Omega$ is $W^{1,\infty}$, $\partial\Omega_N$ is a finite union of open arcs in $\partial\Omega$ and $\partial\Omega_D$ is a finite union of closed arcs, none of which is reduced to a single point. They also state the following characterization in dimension 3.

Proposition 3.10. *Assume Ω is a bounded domain of class $C^{0,1}$ and $\partial\Omega_N$ is open in $\partial\Omega$. Then $G = \Omega \cup \partial\Omega_N$ is Gröger-regular if and only if the two conditions below hold:*

1. *$\partial\Omega_D$ is the closure of its interior.*
2. *For $x \in \partial(\partial\Omega_N)$, there is a neighborhood U of x and a Lipschitz diffeomorphism Ψ such that $\Psi(U \cap \partial(\partial\Omega_N)) = (-1, 1)$.*

This characterization, which is not true in dimension larger than 3, implies that $W^{1,\infty}$ polyhedrons of $\mathbb{R}^3$ where $\partial(\partial\Omega_N)$ is a finite union of line segments are Gröger regular sets.

3.3.2 Linear elliptic equations

Though the linear elasticity system does not fall into this category, we first present some regularity results in the case of linear elliptic equations, for which a wider literature exists. Here we only state results for scalar equations, keeping in mind that uncoupled systems of dimension d reduce to d scalar equations.

A convenient space to study problems with mixed boundary conditions is the Sobolev space $W^{1,p}_{\mathrm{D}}(\Omega)$, defined as the closure of the set

$$\left\{ u \in C^\infty_{\mathrm{c}}(\mathbb{R}^d) \;\middle|\; \mathrm{supp}(u) \cap \partial\Omega_{\mathrm{D}} = \varnothing \right\} \tag{3.11}$$

for the usual norm in $W^{1,p}(\Omega)$. Functions contained in $W^{1,p}_{\mathrm{D}}(\Omega)$ have a zero trace on $\partial\Omega_{\mathrm{D}}$ but can take some nonzero values on $\partial\Omega_{\mathrm{N}}$.

The topological dual of $W^{1,p'}_{\mathrm{D}}(\Omega)$ is denoted by $W^{-1,p}_{\mathrm{D}}(\Omega)$, where p' is the conjugate exponent to p, defined by $1/p+1/p' = 1$. This dual contains, among others, linear forms of type $v \mapsto \int_\Omega f \cdot v \,\mathrm{d}x$ for a measurable function f, but also distributions supported by a submanifold of dimension $d-1$ included in G. In particular, it contains applications of the form $v \mapsto \int_{\partial\Omega_{\mathrm{N}}} g \cdot v \,\mathrm{d}x$, which correspond to Neumann conditions.

From now on, we consider that $\partial\Omega_{\mathrm{D}}$ has a nonzero relative Lebesgue measure in $\partial\Omega$.

We consider (scalar) linear partial differential equations of type

$$\begin{cases} -\mathrm{div}(A\nabla u) + bu &= f \quad \text{in } \Omega \\ A\nabla u \cdot n &= g \quad \text{on } \partial\Omega_{\mathrm{N}}, \\ u &= 0 \quad \text{on } \partial\Omega_{\mathrm{D}} \end{cases} \tag{3.12}$$

where $A \in L^\infty(\Omega, M_d(\mathbb{R}))$ satisfies the ellipticity condition

$$\forall x \ \in \Omega, \forall z \in \mathbb{R}^d \qquad m\|z\|^2 \leqslant z \cdot A(x) z \leqslant M\|z\|^2 \tag{3.13}$$

for some $M > m > 0$ and $b \in L^\infty(\Omega, \mathbb{R})$ is a nonnegative function. It is known, due to Lax-Milgram's theorem, that for $f \in H^{-1}(\Omega)$ and $g \in H^{-1/2}(\partial\Omega_{\mathrm{N}})$, (3.12) has a unique solution $u \in H^1_{\mathrm{D}}(\Omega)$.

In the context of Gröger-regular sets, Gröger (1989) studies (3.12) in terms of the operator $\mathcal{A} : W^{1,p}_{\mathrm{D}}(\Omega) \to W^{-1,p}_{\mathrm{D}}(\Omega)$ defined by

$$\langle \mathcal{A}u, v \rangle = \int_\Omega (A\nabla u \cdot \nabla v + buv) \,\mathrm{d}x. \tag{3.14}$$

His framework also includes the study of non-linear elliptic equations and the evolution of a regularity result when the domain is transformed using a Lipschitz diffeomorphism. We state here a simplified corollary of his regularity result (Gröger, 1989, Theorem 1).

Proposition 3.11. *Assume that $\Omega \cup \partial\Omega_{\mathrm{N}}$ is Gröger-regular. If $b > m$ almost everywhere in Ω, then there is a $q > 2$ such that for all $p \in [2, q]$, the mapping $\mathcal{A}$ is a continuous isomorphism from $W^{1,p}_{\mathrm{D}}(\Omega)$ to $W^{-1,p}_{\mathrm{D}}(\Omega)$.*

In other words, if for such a p the application $v \mapsto \int_\Omega fv \,\mathrm{d}x + \int_{\partial\Omega_{\mathrm{N}}} gv \,\mathrm{d}s$ is in $W^{-1,p}_{\mathrm{D}}(\Omega)$, then the solution of (3.12) is in $W^{1,p}_{\mathrm{D}}(\Omega)$. The author also provides for $\mathcal{A}^{-1}$ an estimate which depends on the ellipticity constant $(1 - m/M)^{1/2}$ and the domain regularity (see also Gröger and Rehberg, 1989). Though this article introduces a framework for the study of mixed boundary conditions, the result in Proposition 3.11 is not fully satisfying. Indeed, in our application case, Ω represents an organ, and we cannot easily make sure that it is regular enough so that $q > d$.

Remark 3.5. Even in the case of Poisson's equation ($A = I$), the presence of mixed boundary conditions degrades the regularity of solutions. While solutions to the Dirichlet problem are in $W^{1,p}$ for every $p > 2$, no regularity better than that of Proposition 3.11 can be expected.

A Hölder regularity result is proposed by Droniou (2000, Theorem 2.1). Though the author does not cite Gröger's work, the hypotheses he sets on the domain are similar to the definition of Gröger-regular sets. Once again, the author handles a general class of linear elliptic problems, involving for instance nonhomogeneous Dirichlet boundary conditions, a transport term and several configurations of boundary conditions. The regularity result adapted to our case is stated below.

Proposition 3.12. *Assume $\Omega \cup \partial\Omega_{\mathrm{N}}$ is Gröger-regular and choose $p > d$. If $f \in \left(W^{1,p'}(\Omega)\right)'$ and $g \in \left(W^{1-1/p',p'}(\partial\Omega_{\mathrm{N}})\right)'$, then there exists $\kappa \in (0, 1 - d/p]$ depending only on Ω, p, m, M such that the solution u of (3.12) is κ-Hölder continuous in Ω.*

In addition, if there is a constant $\Lambda > 0$ such that

$$\|b\|_{L^{dp/(d+p)}(\Omega)} + \|f\|_{\left(W^{1,p'}(\Omega)\right)'} + \|g\|_{\left(W^{1-1/p',p'}(\partial\Omega_{\mathrm{N}})\right)'} \leqslant \Lambda,$$

then there is a constant $C = C(\Omega, \partial\Omega_{\mathrm{N}}, m, M, p, \Lambda)$ such that

$$\|u\|_{C^{0,\kappa}(\Omega)} \leqslant C.$$

In Proposition 3.12, the Neumann data g is supposed to be a linear form which applies on $W^{1-1/p',p'}(\partial\Omega_{\mathrm{N}})$. This space contains the traces of functions in $W^{1,p'}_{\mathrm{D}}(\Omega)$ on $\partial\Omega_{\mathrm{N}}$. Therefore, f and g define elements of $W^{-1,p}_{\mathrm{D}}(\Omega)$ and another way to formulate the result is that $\mathcal{A}^{-1}$ maps continuously $W^{-1,p}_{\mathrm{D}}(\Omega)$ into $C^{0,\kappa}(\Omega)$.

Years later, using Gröger's formalism, Haller-Dintelmann et al. (2009) obtain a result similar to Proposition 3.12. In addition to the characterization of Gröger-regular sets mentioned above, they also give precisions about the interest of Hölder continuity results in the context of optimal control.

3.3.3 Elasticity equation

We now turn to the linear elasticity system. Though we already introduced the linear elasticity system in Section 2.2, we give some additional details here concerning the regularity of solutions. Unless stated otherwise, functions in this paragraph take their value in $\mathbb{R}^d$.

As opposed to elliptic equations, the elastic energy only penalizes the symmetric part of the displacement gradient (also known as the linearized strain tensor), denoted $\varepsilon(u) = \frac{1}{2}(\nabla u^{\mathrm{T}} + \nabla u)$. In this section A denotes the elasticity tensor, a linear operator which satisfies an ellipticity condition with respect to the strain tensor of the form

$$\forall x \in \Omega \quad \forall \varepsilon \in M_d(\mathbb{R}) \text{ such that } \varepsilon^{\mathrm{T}} = \varepsilon \qquad m\|\varepsilon\|^2 \leqslant A\varepsilon : \varepsilon \leqslant M\|\varepsilon\|^2$$

for some $M > m > 0$. In the context of the standard elasticity system, the linear elasticity tensor is defined by Hooke's law

$$A\varepsilon(u) = 2\mu\varepsilon(u) + \lambda \operatorname{tr}(\varepsilon(u)).$$

Remember that the linear elasticity equation reads

$$\begin{cases} \operatorname{div}(A\varepsilon(u)) &= f \quad \text{in } \Omega_0 \\ A\varepsilon(u)\cdot n &= g \quad \text{on } \partial\Omega_{\mathrm{N}}, \\ u &= 0 \quad \text{on } \partial\Omega_{\mathrm{D}} \end{cases} \tag{3.15}$$

with $g \in H^{-1/2}(\partial\Omega_{\mathrm{N}})$ a surface force field and $f \in H^{-1}(\Omega)$ stands for volume forces.

The weak formulation of (3.15) reads as follows: find a displacement field $u \in H^1(\Omega_0)$ such that

$$\forall v \in H^1_{\mathrm{D}}(\Omega_0) \quad \int_{\Omega_0} A\varepsilon(u) : \varepsilon(v)\,\mathrm{d}x = \int_{\Omega} f\cdot v\,\mathrm{d}x + \int_{\partial\Omega_{\mathrm{N}}} g\cdot v\,\mathrm{d}s.$$

A key result to establish the well-posedness of (3.15) is the equivalence between norms based on the strain tensor with the H^1 norm. We state it here as a lemma (Ciarlet, 1988, Chapter 6, Theorems 6.3-3 and 6.3-4).

Lemma 3.2. *Denote the space*

$$K_{\mathrm{D}}(\Omega) = \left\{ u \in L^2(\Omega) \mid \varepsilon(u) \in L^2(\Omega) \text{ and } u = 0 \text{ on } \partial\Omega_{\mathrm{D}} \right\}.$$

The spaces $K_{\mathrm{D}}(\Omega)$ and $H^1_{\mathrm{D}}(\Omega)$ are actually the same space. In addition, if $\partial\Omega_{\mathrm{D}}$ has positive measure in $\partial\Omega_0$, then the norm $u \mapsto \|\varepsilon(u)\|_{L^2(\Omega)}$ is equivalent to the H^1 norm.

Due to the presence of the symmetric gradient $\varepsilon(u)$ instead of ∇u, the linear elastic system is not included in the framework of elliptic partial differential equations, and as a consequence, fewer works exist about it in the literature. We present a few regularity works where the authors adapt techniques used in the context of elliptic systems to the elasticity system.

An early work is proposed by Shi and Wright (1994), where the authors adapt techniques used for elliptic equations to the linear elasticity system in a domain with C^1 boundary. In the case of data in $W^{-1,p}_{\mathrm{D}}(\Omega)$, they obtain $W^{1,p}$-estimates for p in an interval $(2, q)$ where q depends only on the domain and the elliptic constants of the system m and M.

An interesting result is proposed by Herzog et al. (2011). The authors adapt Gröger's framework to linear and non-linear equations that satisfy an ellipticity condition with respect to the strain tensor $\varepsilon(u)$. We state their theorem in the case of linear elasticity.

Proposition 3.13. *Assume $\Omega \cup \partial\Omega_{\mathrm{N}}$ is Gröger-regular. There is a $q > 2$ such that for all $p \in [2, q]$, the system (3.15) has a unique solution $u \in W^{1,p}_{\mathrm{D}}(\Omega)$ which depends Lipschitz continuously on the data, provided that f and g define elements of $W^{-1,p}_{\mathrm{D}}(\Omega)$. Namely, there is a constant C such that*

$$\|u\|_{W^{1,p}(\Omega)} \leqslant C\left(\|f\|_{W^{-1,p}(\Omega)} + \|g\|_{W^{-1,p}(\Omega)} \right).$$

This result is the best we found in the literature. Though it ensures Hölder continuity of solutions in dimension 2, it is not sufficient in dimension 3.

We end this section by quoting a regularity result in dimension 3 from Ciarlet (1988, Chapter 6, Theorem 6.3-6) which requires more regular data. The theorem is stated for Dirichlet boundary conditions, but the author explains that the result still holds in the case of Neumann conditions, i.e. $\partial\Omega_{\mathrm{N}} = \partial\Omega$. Though, the result is not true in the case of mixed boundary conditions, as the solution might be irregular in a vicinity of the interface between $\partial\Omega_{\mathrm{N}}$ and $\partial\Omega_{\mathrm{D}}$.

Solutions to a problem with Neumann boundary conditions are defined up to an infinitesimal rigid displacement. We denote by $T(\Omega) = \{w \mid \varepsilon(w) = 0\}$ the set of rigid translation in Ω. In particular, such displacements satisfy $\int_{\Omega_0} A\varepsilon(w) : \varepsilon(v)\,\mathrm{d}x = 0$ for any test function v, which means that they do not cost any elastic energy. Solutions to the elastic problem are measured with the seminorm

$$\|u\|_{W^{k,p}(\Omega)/T(\Omega)} = \inf_{w \in T(\Omega)} \|u + w\|_{W^{k,p}(\Omega)}. \tag{3.16}$$

As rigid translations are not penalized by the elastic deformation energy, they should not be encouraged by the external forces. In order for solutions to exist, the forces f and g should satisfy the following compatibility conditions

$$\forall w \in T(\Omega) \qquad \int_\Omega f \cdot w\,\mathrm{d}x + \int_{\partial\Omega} g \cdot w\,\mathrm{d}s = 0. \tag{3.17}$$

Keeping this condition in mind, we state the result for Neumann boundary conditions.

Proposition 3.14. *Assume $\partial\Omega \subset \mathbb{R}^3$ is of class C^2 with $\partial\Omega_{\mathrm{N}} = \partial\Omega$. Let $p \in [6/5, \infty)$, $f \in L^p(\Omega)$ and $g \in W^{1-1/p,p}(\partial\Omega)$ such that (3.17) holds, then the solutions of (3.15) are in $W^{2,p}(\Omega)$, and there is a constant C such that*

$$\|u\|_{W^{2,p}(\Omega)/R(\Omega)} \leqslant C\left(\|f\|_{L^p(\Omega)} + \|g\|_{W^{1-1/p,p}(\Omega)}\right).$$

Chapter 4

Existence of solutions and optimality conditions

4.1 Properties of the functional

To better understand the properties of the optimization problem, we first give a closer look at the functional (2.9). Remember that its expression reads

$$J(u) = \int_\Gamma j_y(u)\,\mathrm{d}y, \qquad \text{where} \qquad j_y(u) = \tfrac{1}{2}d^2(y, S_u). \tag{4.1}$$

Here, j_y denotes an elementary functional associated to a single point $y \in \Gamma$ and S_u denotes the part of the deformed boundary $(\mathrm{Id}+u)(\partial\Omega_0)$ that should match Γ. If Γ is a discrete set, for instance a point cloud, the integral in (4.1) is replaced by a sum. Remember also that, a priori, $J(u)$ is defined for a continuous displacement field $u \in C(\overline{\Omega}_0)$.

As J measures the discrepancy between the deformed surface S_u and the observed data Γ, its properties depend on the geometric relationship between those two surfaces. In particular, the orthogonal projection plays a key role as a transformation between Γ and S_u. On the one hand, the Gateaux differentiability of J depends on whether points from Γ have a unique projection point onto S_u. On the other hand, the regularity of descent directions is degraded when several points from Γ project onto the same point in S_u. In this section, we first discuss the functional differentiability, and then we mention the possibility to generate regular descent directions.

4.1.1 Directional differentiability and more

In this section, we study the differentiability of the functional J. In the context of optimization problems, computing derivatives is useful to find descent directions, i.e. directions that make the objective function decrease, and also to assess optimality. In our case, J is not always Gateaux differentiable, as for certain displacement fields u, J cannot be approximated around u using an affine function. However, we prove below that

J has directional derivatives. In addition, the expression (4.1) involves a min operation (hidden in the distance term). We show below that the min operator also appears in the derivative expression, which suggests that it is easier to make J decrease than increase. In other words, non-differentiability results in an abundance rather than in a lack of descent directions.

We now derive an expression for the derivatives of J. In a first technical lemma, we focus on the elementary functional j_y defined in (4.1). From there, we use the dominated convergence theorem to obtain a result for J.

As j_y is expressed in terms of distance to S_u, we compute derivatives using Danskin's theorem (Danskin, 1967), stated as Lemma 3.1 in this manuscript, to investigate directional differentiability. In his work, the author proposes conditions to obtain Fréchet differentiability and even C^1 regularity for the considered function, but those results are stated for finite-dimensional spaces. For this reason, we only use Lemma 3.1 for directional differentiability, by considering j_y along a fixed direction. This result is very close to and inspired from Proposition 3.8 and Proposition 3.9, but here the derivative is expressed with respect to the displacement field instead of the shape.

Differentiability is related with the number of orthogonal projections of a given point $y \in \Gamma$ onto the deformed surface S_u. In this chapter, we use the notation

$$P_y(u) = \{x \in S_0 \mid \|x + u(x) - y\| = d(y, S_u)\} = (\mathrm{Id} + u)^{-1} \Pi_{S_u}(y). \tag{4.2}$$

Points in $P_y(u)$ belong to the original surface S_0, but their images by $\mathrm{Id} + u$ are located at a distance $d(y, S_u)$ from y. When $P_y(u)$ is a singleton, it means that y has a single projection point onto S_u, and this projection point is the image of a single point from S_0 by $\mathrm{Id} + u$.

Lemma 4.1. *Assume $\Omega_0 \subset \mathbb{R}^d$ is a bounded set with Lipschitz boundary, and consider u a continuous displacement field and y a fixed point in Γ.*

1. *The application j_y is locally Lipschitz continuous for the uniform convergence norm. The Lipschitz constant can be chosen independent from $y \in \Gamma$ but still depends on Γ.*

2. *In addition, j_y has directional derivatives at u. For $v \in C(\overline{\Omega}_0)$, the directional derivative in the direction v reads*

$$\mathrm{d}j_y(u)(v) = \min_{x \in P_y(u)} v(x) \cdot (x + u(x) - y).$$

where $P_y(u)$ is defined in (4.2).

Proof. Recalling that $S_u = (\mathrm{Id} + u)(S_0)$ and that S_0 is compact, we write

$$j_y(u) = \min_{x \in S_0} \tfrac{1}{2} \|x + u(x) - y\|^2. \tag{4.3}$$

1. Let $u_1, u_2 \in C(\overline{\Omega}_0)$. Because S_0 is compact and u_1 is continuous, there exists $x_1 \in S_0$ such that $j_y(u_1) = \frac{1}{2}\|x_1 + u_1(x_1) - y\|^2$. Now, using (4.3),

$$\begin{aligned} j_y(u_2) - j_y(u_1) &\leqslant \tfrac{1}{2}\|x_1 + u_2(x_1) - y\|^2 - \tfrac{1}{2}\|x_1 + u_1(x_1) - y\|^2 \\ &= \tfrac{1}{2}(2x_1 + u_2(x_1) + u_1(x_1) - 2y) \cdot (u_2(x_1) - u_1(x_1)) \\ &\leqslant \left(\max_{x\in S_0} \|x - y\| + \tfrac{1}{2}\|u_2 + u_1\|_{L^\infty(\Omega_0)}\right) \|u_2 - u_1\|_{L^\infty(\Omega_0)}. \end{aligned}$$

   As Γ is compact, the previous Lispchitz constant is controlled by

$$L = \max_{y\in\Gamma} \left(\max_{x\in S_0} \|x - y\|\right) + \tfrac{1}{2}\|u_2 + u_1\|_{L^\infty(\Omega_0)}.$$

2. Let $v \in C(\overline{\Omega}_0)$ a direction. To compute the derivative of j_y in the direction v, we apply Lemma 3.1 to the application

$$\begin{array}{rccl} f: & \mathbb{R}_+ \times S_0 & \to & \mathbb{R} \\ & (t, x) & \mapsto & \frac{1}{2}\|x + u(x) + tv(x) - y\|^2, \end{array}$$

   and we use the notation

$$K(t) = \{x \in S_0 \mid d(y, S_{u+tv}) = \|x + u(x) + tv(x) - y\|\}.$$

   Note that for $t > 0$, $K(t)$ is nonempty as S_0 is compact and $u + tv$ is continuous. Moreover, the derivative $\partial_t f$ exists and is continuous. As a consequence of Lemma 3.1 applied to the identity $j_y(u + tv) = \min_{x\in S_0} f(t, x)$, j_y is differentiable at u in the direction v, and its directional derivative reads

$$\mathrm{d}j_y(u)(v) = \lim_{t\searrow 0} \frac{j_y(u + tv) - j_y(u)}{t} = \min_{x\in K(0)} \partial_t f(0, x),$$

   hence the result with $P_y(u) = K(0)$.

□

Of course, if Γ contains a finite number of points, the derivative of J can be written using a simple sum. In the case where Γ is continuous and J is defined using an integral, we conclude using the dominated convergence theorem.

Proposition 4.1. *Let $\Omega_0 \subset \mathbb{R}^d$ a Lipschitz bounded set, and $u \in C(\overline{\Omega}_0)$. The functional (2.9) is directionally differentiable at u. Its derivative in the direction $v \in C(\overline{\Omega}_0)$ is given by*

$$\mathrm{d}J(u)(v) = \int_\Gamma \min_{x\in P_y(u)} [v(x) \cdot (x + u(x) - y)] \,\mathrm{d}y, \tag{4.4}$$

where $P_y(u)$ is defined in (4.2). In addition, denote by $\tilde{\Gamma} \subset \Gamma$ the set where $P_y(u)$ is not a singleton. For $y \in \Gamma \setminus \tilde{\Gamma}$, denote by x_y the single element of $P_y(u)$. If $\tilde{\Gamma}$ has zero relative Lebesgue measure in Γ, $\mathrm{d}J(u)$ is a linear continuous application on $C(\overline{\Omega}_0)$ defined by

$$\langle \mathrm{d}J(u), v\rangle = \int_{\Gamma\setminus\tilde{\Gamma}} v(x_y) \cdot (x_y + u(x_y) - y) \,\mathrm{d}y. \tag{4.5}$$

Proof. Let $B = B_{C(\overline{\Omega}_0)}(0, r)$ a small ball of radius $r > 0$. Due to Lemma 4.1, for $y \in \Gamma$, j_y is Lipschitz continuous on $u + B$. Therefore, for $v \in B$

$$\forall t \in (0,1) \quad \left| \frac{j_y(u+tv) - j_y(u)}{t} \right| \leqslant L\|v\|_{L^\infty(\Omega_0)} \leqslant Lr,$$

where $L > 0$ is independent from $y \in \Gamma$. Using the dominated convergence theorem, we pass to the limit in the expression

$$\frac{J(u+tv) - J(u)}{t} = \int_\Gamma \frac{j_y(u+tv) - j_y(u)}{t} \,\mathrm{d}y \xrightarrow[t \searrow 0]{} \int_\Gamma \mathrm{d}j_y(u)(v) \,\mathrm{d}y$$

to obtain the derivative.

Now, if we assume that $\tilde{\Gamma}$ has zero Lebesgue measure, formula (4.5) is easily derived from (4.4), and $\mathrm{d}J(u)$ is obviously linear. Remembering that the application $d(\cdot, S_u)$ is continuous, we obtain

$$|\langle \mathrm{d}J(u), v\rangle| \leqslant |\Gamma| \left(\max_{y \in \Gamma} d(y, S_u) \right) \|v\|_{L^\infty(\Omega_0)},$$

which shows that $\mathrm{d}J$ is continuous. □

As the expression of J is based on the distance with respect to S_u, $\mathrm{d}J$ is not always a linear form. Non-differentiability happens when $P_y(u)$ contains several points for a part of Γ with a positive relative Lebesgue measure. We see in the expression $P_y(u) = (\mathrm{Id} + u)^{-1}[\Pi_{S_u}(y)]$ that, for a given $y \in \Gamma$, this non-uniqueness is either caused by the projection operator Π_{S_u} or by the deformation $(\mathrm{Id} + u)$. In the first situation, y belongs to the skeleton of S_u and has several projections onto S_u. In the second situation, y may have a unique projection $p_{S_u}(y)$, but this projection is the image of several points of S_0 by the application $(\mathrm{Id} + u)$.

Figure 4.1 shows a configuration where Γ is included in the skeleton of S_0 (here $u = 0$). In this case, every point in Γ has two projections onto S_0, while $\mathrm{Id} + u = \mathrm{Id}$ is a bijection. If we consider the rigid translation $v \equiv V$ with $V \in \mathbb{R}^2$, we note that $\mathrm{d}J(0)(-v) = \mathrm{d}J(0)(v) < 0$. Thus, $\mathrm{d}J$ is not a linear application. In Figure 4.2, each point $y \in \Gamma$ has a single projection onto S_u, but this projection point is the image of both $x_1, x_2 \in S_0$ by $(\mathrm{Id} + u)$. Once again, each leg of the deformed set can move closer to Γ with the same effect on J. However, if only one leg moves away, from Γ, the value of J does not change as the other leg is not moving, which is again a nonlinear behavior.

Let us look more closely at the example from Figure 4.1. If we only consider rigid deformations $\mathrm{d}J(0)(v)$ and $\mathrm{d}J(0)(-v)$ can be both negative but cannot be both positive, as y can move closer to x_1 or x_2 but cannot move away from x_1 and x_2 at the same time. In Figure 4.2, one leg moving closer to Γ is sufficient to make J decrease, while *both* legs need to move away from Γ to make J increase. Thus, it seems easier to make J decrease than increase. Due to the presence of a min operator in the derivative of J, we can generalize this observation in a tiny lemma.

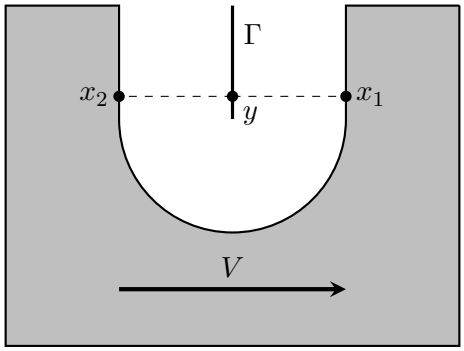


Figure 4.1: Non-differentiable case with $u = 0$ and $S_0 = \partial\Omega_0$. The point $y \in \Gamma$ is equidistant from x_1 and x_2, so that $P_y(u) = \{x_1, x_2\}$. The derivative of J has the same (negative) value in the directions $V\mathbf{1}_{\Omega_0}$ and $-V\mathbf{1}_{\Omega_0}$.

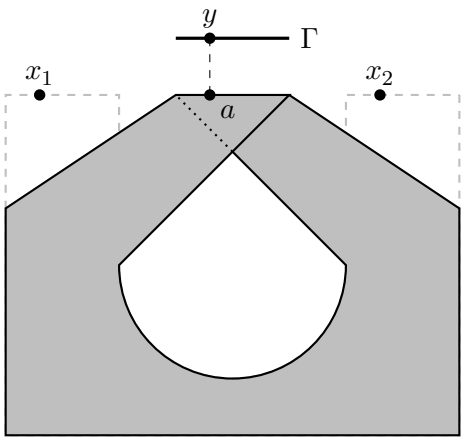


Figure 4.2: Non-differentiable case with self-intersection and, again, $S_0 = \partial\Omega_0$. The dashed set is Ω_0, while the gray set is Ω_u. The displacement field u transports both x_1 and x_2 to $a = p_{S_u}(y)$.

Lemma 4.2. *Let $u \in C(\overline{\Omega}_0)$ and v a direction. Then* $\mathrm{d}J(u)(-v) \leqslant -\mathrm{d}J(u)(v)$.

Proof. Let $(x_y)_{y\in\Gamma}$ a family such that $x_y \in P_y(u)$ and

$$\mathrm{d}J(u)(v) = \int_\Gamma v(x_y)\cdot(x_y + u(x_y) - y)\,\mathrm{d}y.$$

The directional derivative in the direction $-v$ reads

$$\int_\Gamma \min_{x\in P_y(u)} \left[-v(x)\cdot(x+u(x)-y)\right]\,\mathrm{d}y \leqslant \int_\Gamma -v(x_y)\cdot(x_y+u(x_y)-y)\,\mathrm{d}y,$$

□

A simple consequence of Lemma 4.2 is that a non-differentiable point cannot be a local minimizer of J. Non-differentiable minimizers are a pet peeve of continuous optimization solvers, as it is difficult to measure optimality around such points. Fortunately, the following lemma evacuates this possibility.

Lemma 4.3. *Let $u \in C(\overline{\Omega}_0)$ a local minimizer of J. Then J is Gateaux differentiable at u with* $\mathrm{d}J(u) = 0$.

Proof. As u is a local minimizer of J, every direction v satisfies $\mathrm{d}J(u)(v) \geqslant 0$. If v is a direction such that $\mathrm{d}J(u)(v) > 0$, then, due to Lemma 4.2, we obtain $\mathrm{d}J(u)(-v) < 0$, and $-v$ is a descent direction. As it is in contradiction with u being a local minimizer, the only possibility is $\mathrm{d}J(u)(v) = 0$. □

Last but not least, we try to obtain some insight about the first-order behavior of j_y and J in the non-differentiable case. In Figure 4.1, y has two projections onto S_u. If we consider all directions in the space of continuous displacement fields, we understand that, for a part of those directions, x_1 moves closer to y faster (or gets away from y slower) than x_2 , and the expression for $\mathrm{d}j_y(0)$ involves x_1. For the remaining bunch of directions, x_2 gets closer to y faster than x_1 and the expression for $\mathrm{d}j_y(0)$ involves x_2. We show below that those directions are organized in two closed cones, associated with x_1 and x_2, respectively. On each cone, the function $\mathrm{d}j_y(0)$ behaves like a linear application. In this very example, the cones are two half-spaces, separated by the hyperplane

$$H = \left\{ v \in C(\overline{\Omega}_0) \mid v(x_1)\cdot(x_1+u(x_1)-y) = v(x_2)\cdot(x_2+u(x_2)-y) \right\}.$$

The directions $v \in H$ belong to both cones and leave y in the skeleton of S_{u+tv} at first-order.

This behavior is similar for J. This time, each cone is associated with a family $(x_y)_{y\in\Gamma}$, with $x_y \in P_y(u)$, such that for each y, x_y statisfies the min in (4.4). Such a cone may contain only one direction, or represent the whole space in the differentiable case. We state a more formal result in the following proposition.

Proposition 4.2. *Let u a continuous displacement field, and $v \in C(\overline{\Omega}_0)$ a direction. We denote by $(x_y)_{y\in\Gamma}$ a family such that $x_y \in P_y(u)$ and*

$$\mathrm{d}J(u)(v) = \int_\Gamma v(x_y)\cdot(x_y + u(x_y) - y)\,\mathrm{d}y.$$

If $w \in C(\overline{\Omega}_0)$ is another direction associated with the same family $(x_y)_{y\in\Gamma}$, then for α and β two nonnegative coefficients,

$$\mathrm{d}J(u)(\alpha v + \beta w) = \int_\Gamma (\alpha v + \beta w)(x_y)\cdot(x_y + u(x_y) - y)\,\mathrm{d}y.$$

In other words, each direction in the cone generated by v and w is associated with the same family $(x_y)_{y\in\Gamma}$.

Proof. First, note that for $y \in \Gamma$, $x_y \in P_y(u)$, and as a consequence

$$\int_\Gamma \min_{x\in P_y(u)} \left[(\alpha v + \beta w)(x)\cdot(x + u(x) - y)\right]\,\mathrm{d}y \leqslant \int_\Gamma (\alpha v + \beta w)(x_y)\cdot(x_y + u(x_y) - y)\,\mathrm{d}y.$$

Now, if (x'_y) is another family with $x'_y \in P_y(u)$, then

$$\begin{aligned}\int_\Gamma v(x'_y)\cdot(x'_y + u(x'_y) - y)\,\mathrm{d}y &\geqslant \int_\Gamma v(x_y)\cdot(x_y + u(x_y) - y)\,\mathrm{d}y;\\ \int_\Gamma w(x'_y)\cdot(x'_y + u(x'_y) - y)\,\mathrm{d}y &\geqslant \int_\Gamma w(x_y)\cdot(x_y + u(x_y) - y)\,\mathrm{d}y,\end{aligned}$$

and, as $\alpha, \beta \geqslant 0$,

$$\int_\Gamma (\alpha v + \beta w)(x'_y)\cdot(x'_y + u(x'_y) - y)\,\mathrm{d}y \geqslant \int_\Gamma (\alpha v + \beta w)(x_y)\cdot(x_y + u(x_y) - y)\,\mathrm{d}y.$$

Finally, we obtain

$$\int_\Gamma \min_{x\in P_y(u)} \left[(\alpha v + \beta w)(x)\cdot(x + u(x) - y)\right]\,\mathrm{d}y = \int_\Gamma (\alpha v + \beta w)(x_y)\cdot(x_y + u(x_y) - y)\,\mathrm{d}y,$$

and $\alpha v + \beta w$ is also associated with the family (x_y). □

Instead of one single linear form $\mathrm{d}J(u)$ in the differentiable case, non-differentiable points yield as many linear forms as there are families of type $(x_y)_{y\in\Gamma}$. Each one of these linear forms ℓ satisfies

$$\langle \ell, v\rangle = \int_\Gamma v(x_y)\cdot(x_y + u(x_y) - y)\,\mathrm{d}y \geqslant \mathrm{d}J(u)(v).$$

As a consequence, a descent direction for one linear form is also a descent direction for J. In the numerical implementation, the projection algorithm only return one family (x_y), that is used to generate a descent direction.

4.1.2 Generate descent directions using extension-regularization

In this section, we only consider the Gateaux differentiable case. We discuss the possibility to compute descent directions to make the objective function decrease in the context of a displacement-based optimization procedure. In particular, we need descent directions to be continuous, so that the successive displacement fields in the optimization process remain in $C(\overline{\Omega}_0)$. Steepest descent directions are usually computed by finding a representant of $\mathrm{d}J(u)$ in a certain inner product. In infinite-dimensional spaces, such directions are not guaranteed to be as regular as the optimization variable u.

A classical technique to compute regular descent directions in shape optimization is the velocity extension-regularization method (see Doğan et al., 2007). When the objective function is Gateaux differentiable, this method consists in choosing the shape gradient in the H^1 inner product (or another elliptic inner product) as a descent direction. Due to elliptic regularity, resulting directions are more regular. We follow a similar approach and we describe several cases depending on the regularity of $\mathrm{d}J(u)$.

We fix a displacement field $u \in C(\overline{\Omega}_0)$. Following the extension-regularization framework, we look for descent directions $w \in C(\overline{\Omega}_0) \cap H^1_{\mathrm{D}}(\Omega_0)$ such that

$$\forall v \in C(\overline{\Omega}_0) \cap H^1_{\mathrm{D}}(\Omega_0) \qquad \int_{\Omega_0} A\nabla w : \nabla v \,\mathrm{d}x = -\langle \mathrm{d}J(u), v\rangle, \tag{4.6}$$

where A satisfies the ellipticity condition (3.13). In practice, the inner product associated to linear elasticity, defined by

$$\langle u, v\rangle_{\mathrm{el}} = \int_{\Omega_0} A\varepsilon(u) : \varepsilon(v) \,\mathrm{d}x,$$

is also used (see de Buhan et al., 2016; Dapogny et al., 2018, and associated codes). Indeed, an elastic model is more likely to produce a deformation that does not require remeshing. Though, in this section, we stick to the elliptic case, in order to exploit theoretical results presented in Section 3.3.2.

As we are interested in continuous descent directions, we now check the hypotheses of Proposition 3.12. Hölder continuity of solutions is guaranteed when $\mathrm{d}J(u)$ is an element of $W^{-1,p}_{\mathrm{D}}(\Omega)$, for some $p > d$. In other words, $\mathrm{d}J(u)$ should apply on functions in $W^{1,p'}_{\mathrm{D}}(\Omega)$, which is not always possible. Remember that p' is defined so that $1/p + 1/p' = 1$, and thus satisfies $p' < d/(d-1)$.

In order to better understand the differential $\mathrm{d}J$, we describe below a favorable case and a pathological case. The existence of different cases comes from the definition of $\mathrm{d}J$ as an integral on Γ. Here, the key parameter is not the number of projections of a given point $y \in \Gamma$ onto S_u, but rather the number of points from Γ that project onto a given point $a \in S_u$.

A favorable case

Though this case is based on a lot of hypotheses, it is quite common in practice. We assume that $(\mathrm{Id} + u)$ is a Lipschitz diffeomorphism from S_0 to S_u. For instance, when

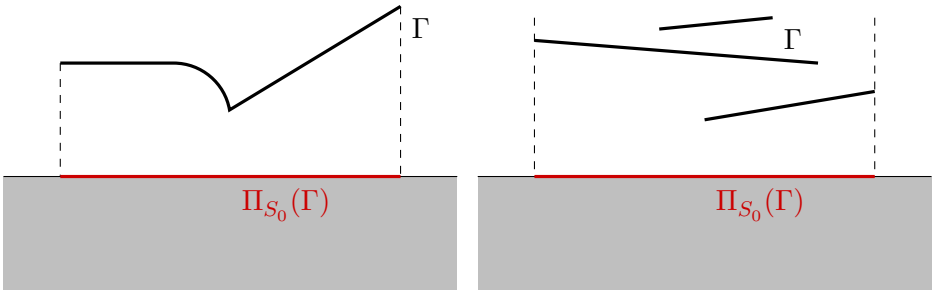


Figure 4.3: Examples of favorable cases for $u = 0$. On the left, Γ is in bijection with $\Pi_{S_0}(\Gamma)$. On the right, each connected part of Γ is in bijection with a subset of $\Pi_{S_0}(\Gamma)$.

$\|u\|_{W^{1,\infty}} < 1$, $(\mathrm{Id} + u)$ is guaranteed to be a Lipschitz diffeomorphism (Henrot and Pierre, 2005). We also assume that the orthogonal projection operator p_{S_u} defines a Lipschitz diffeomorphism between Γ and S_u. Finally, we define the Lipschitz diffeomorphism $\Psi = (\mathrm{Id} + u)^{-1} \circ p_{S_u}$. In other words, there is a Lipschitz change of variable between Γ and $\Psi(\Gamma) \subset S_0$. By doing the substitution in the expression of $\mathrm{d}J(u)$, we obtain

$$\langle \mathrm{d}J(u), v \rangle = \int_{\Psi(\Gamma)} v(x) \cdot (x + u(x) - \Psi^{-1}(x)) |\det \nabla \Psi^{-1}(x)| \, \mathrm{d}x.$$

As Ψ is a Lipschitz diffeomorphism, $\det \nabla \Psi^{-1}(x)$ is bounded, and we also note that

$$\|x + u(x) - \Psi^{-1}(x)\| \leqslant \max_{x \in S_u, y \in \Gamma} \|x - y\|.$$

Therefore, if $v \in W^{1,1}_{\mathrm{D}}(\Omega_0)$ and $M = \|\mathrm{Id} + u - \Psi^{-1}\|_{L^\infty} \|\det \nabla \Psi^{-1}\|_{L^\infty}$, there is a constant $C > 0$ such that

$$|\langle \mathrm{d}J(u), v \rangle| \leqslant M \|v\|_{L^1(\partial\Omega_0)} \leqslant MC \|v\|_{W^{1,1}(\Omega_0)}.$$

Here we used the trace theorem (Evans, 2010, Section 5.5, Theorem 1) for the second inequality. Therefore, $\mathrm{d}J(u)$ is represented by an element of $W^{1,1}_{\mathrm{D}}(\Omega_0)'$. As a consequence, it is possible to apply the regularity result and obtain Hölder continuity for descent directions.

Note that this calculation is still valid when Ψ is a $W^{1,p}$ diffeomorphism with $p > d$. In this case, $\mathrm{d}J(u)$ defines an element of $L^p(\partial\Omega_0)$. It is also possible to split Γ into a finite partition $\Gamma_1, \Gamma_2, \ldots$, each element of which is in bijection with its image by Ψ, and to express $\mathrm{d}J(u)$ as a sum of elements of $L^p(\partial\Omega_0)$. Figure 4.3 shows two examples of favorable cases with $u = 0$. In both cases, each connected part of Γ is diffeomorphic to its projection onto S_u.

A pathological case

The regularity of $\mathrm{d}J(u)$ is degraded when a subset of positive measure in Γ is associated by Ψ with a subset of dimension $d - 2$ or less in $\partial\Omega_0$. Thus, the linear form $\mathrm{d}J(u)$ can

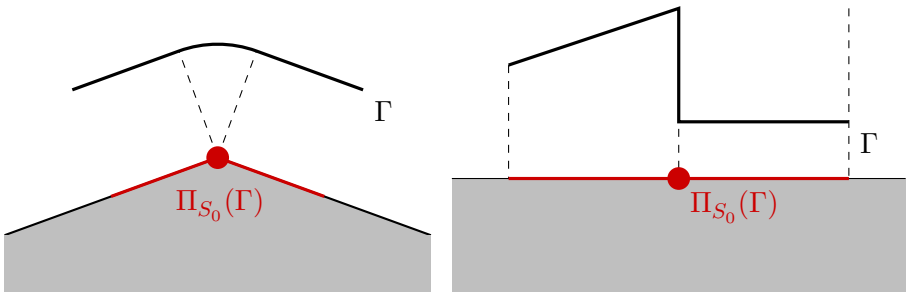


Figure 4.4: Examples of pathological cases for $u = 0$. In both cases, there is an accumulation point in Π_{S_0}, wich results in a loss of regularity for $\mathrm{d}J(0)$.

only apply to displacement fields for which the trace on a $(d-2)$-dimensional set has a meaning, for instance continuous displacement fields. An example is given in Figure 4.4. Note that this situation is likely to happen when the domain is a polyhedron, as corners tend to concentrate orthogonal projections.

An example of pathological case arises when Γ is a point cloud $\{y_1, y_2, \dots\}$ and $(\mathrm{Id} + u)$ is a Lipschitz diffeomorphism. If we denote $x_j = \Psi(y_j)$, the expression of $\mathrm{d}J(u)$ reads

$$\langle \mathrm{d}J(u), v \rangle = \sum_j v(x_j) \cdot (x_j + u(x_j) - y_j).$$

The linear form $\mathrm{d}J$ applies to displacement fields v that can be evaluated at every point $x_j \in \partial\Omega_0$. In other words, $\mathrm{d}J$ only applies on continuous displacement fields. As displacement fields in $W_{\mathrm{D}}^{1,p'}(\Omega_0)$ are not guaranteed to be continuous, we conclude that $\mathrm{d}J$ is not in $W_{\mathrm{D}}^{-1,p}(\Omega_0)$. Therefore, the Hölder continuity result cannot be applied to a descent direction computed with the elliptic inner product.

In such a pathological case, descent directions that are generated using the extension-regularization framework are not in $C(\overline{\Omega}_0)$, and thus cannot be used in the optimization process, at least from a continuous point of view. When solving the problem numerically in fixed dimension, regularity of descent directions is not a concern. However, if the mesh is refined, the discretized problem may approximate better the continuous problem and some irregularities may appear in the successive displacement fields generated by the optimization solver.

Remark 4.1. Haller-Dintelmann et al. (2009, Remark 4.2) note that Proposition 3.11 can be stated with p in an interval of type $[2-\delta, 2+\delta]$. As a consequence, if Proposition 3.11 yields a $\delta > 0$ such that $2 - \delta < d/(d-1)$ (which is true in dimension 2), then we have $\mathrm{d}J(u) \in C(\overline{\Omega}_0)' \subset W_{\mathrm{D}}^{-1,2-\delta}(\Omega_0)$, and resulting descent directions are in $W_{\mathrm{D}}^{1,2-\delta}(\Omega_0)$.

4.2 Existence of solutions

We now turn to the existence of solutions. Proving that solutions exist may be useful, even in a numerical context, to ensure that displacement fields returned by the registration algorithm are consistent with the continuous problem.

We use very standard tools to prove the existence of a solution, starting with a minimizing sequence in the space of controls, and then proving that the associated state sequence converges to a state that minimizes the functional J. In this section, the regularity of solutions to the elasticity system is critical to ensure the required convergences. Unfortunately, we did not find the relevant results in the literature, and we were not able to prove the existence of solutions in a general case.

We begin the section by considering a toy problem, obtained by replacing the elasticity system with an elliptic system, and we prove existence of solutions in a general case. Then, we turn to the problem involving the linear elasticity system, with a partial result. Finally, we consider a problem with stronger hypotheses, in order to obtain an existence result with the elasticity system. Remember that the feasible set $\mathcal{G}_M$ is defined in (2.12) by

$$\mathcal{G}_M = \left\{ g \in L^\infty(\partial\Omega_{\mathrm{N}}) \mid \|g\|_{L^\infty(\partial\Omega_{\mathrm{N}})} \leqslant M \right\}.$$

In this section, we also refer to Gröger-regular sets. This notion is defined and discussed in Section 3.3.

4.2.1 A full result with an elliptic equation

Before we consider the existence of solutions for the elastic model, we first consider a simpler toy problem. The main difficulty from this toy problem to the real problem (2.13) is to handle the regularity of solutions for the linear elasticity system. The toy problem is based on Poisson's equation and reads

$$\min_{g\in\mathcal{G}_M} J(u_g) \qquad \text{s.t.} \qquad \left\{ \begin{array}{rll} -\Delta u_g + u_g &= 0 & \text{in } \Omega \\ u_g &= 0 & \text{on } \partial\Omega_{\mathrm{D}}, \\ \partial_n u_g &= g & \text{on } \partial\Omega_{\mathrm{N}}, \end{array} \right. \tag{4.7}$$

The key result we use to prove the existence of solution is Proposition 3.12 which provides Hölder estimates for solutions of elliptic partial differential equations.

Theorem 4.1. *Assume that Ω_0 is Gröger-regular. Then the toy problem (4.7) has at least one solution.*

Proof. Consider a minimizing sequence (g_j) of elements of $\mathcal{G}_M$. As (g_j) is bounded in $L^\infty(\partial\Omega_{\mathrm{N}})$, it converges $\star$-weakly in $L^\infty(\partial\Omega_{\mathrm{N}})$, up to a subsequence. We denote its limit by g. Note that $\mathcal{G}_M$ is closed for the $\star$-weak topology in $L^\infty(\partial\Omega_{\mathrm{N}})$, and for this reason $g \in \mathcal{G}_M$. The proof falls into two parts. First, we prove that the sequence of associated states converges in some sense to the state associated to g . Then, we prove that the states actually converge uniformly to finish the proof.

Denote respectively by $u_j = u_{g_j}$ the state associated to g_j and by $u = u_g$ the state associated to g. We first show that the sequence of states (u_j) converges toward u in some sense. The state equations for u and u_j read

$$\forall v \in H^1_{\mathrm{D}}(\Omega) \quad \int_{\Omega_0} (\nabla u_j : \nabla v + u_j \cdot v) \,\mathrm{d}x = \int_{\partial\Omega_{\mathrm{N}}} g_j \cdot v \,\mathrm{d}s$$
$$\forall v \in H^1_{\mathrm{D}}(\Omega) \quad \int_{\Omega_0} (\nabla u : \nabla v + u \cdot v) \,\mathrm{d}x = \int_{\partial\Omega_{\mathrm{N}}} g \cdot v \,\mathrm{d}s.$$

Because (g_j) converges weakly in $L^2(\partial\Omega_{\mathrm{N}})$, we obtain

$$\forall v \in H^1_{\mathrm{D}}(\Omega) \quad \int_{\partial\Omega_{\mathrm{N}}} g_j \cdot v \,\mathrm{d}s \xrightarrow[j\to+\infty]{} \int_{\partial\Omega_{\mathrm{N}}} g \cdot v \,\mathrm{d}s.$$

Now, using the state equations satisfied by u and u_j results in

$$\forall v \in H^1_{\mathrm{D}}(\Omega) \quad \int_{\Omega_0} (\nabla u_j : \nabla v + u_j \cdot v) \,\mathrm{d}x \xrightarrow[j\to+\infty]{} \int_{\Omega_0} (\nabla u : \nabla v + u \cdot v) \,\mathrm{d}x,$$

i.e. $u_j \rightharpoonup u$ in $H^1(\Omega_0)$. In particular, $u_j \rightharpoonup u$ in $L^2(\Omega_0)$.

Now, we show that the convergence of (u_j) is actually uniform. Uniform convergence implies weak convergence in $L^2(\Omega_0)$, which ensures that the uniform limit of (u_j) is u.

Droniou (2000, Remark 2.3) notes that $L^{(d-1)p/d}(\partial\Omega_{\mathrm{N}}) \subset W^{1-1/p',p'}(\partial\Omega_{\mathrm{N}})'$ with continuous imbedding. As a consequence, the imbedding from $L^\infty(\partial\Omega_{\mathrm{N}})$ to $W^{1-1/p',p'}(\partial\Omega_{\mathrm{N}})'$ is also continuous and (g_j) is bounded in $W^{1-1/p',p'}(\partial\Omega_{\mathrm{N}})'$. Due to Proposition 3.12, there exists $\kappa > 0$ such that u_j is κ-Hölder continuous and there is a constant $C > 0$ such that

$$\forall j \qquad \|u_j\|_{C^{0,\kappa}(\Omega_0)} < C.$$

In other words, the u_j are bounded and equicontinuous. Due to Ascoli's theorem (Evans, 2010, Appendix C.7), up to a subsequence, (u_j) converges uniformly in $C(\overline{\Omega}_0)$ toward u. As J is continuous for the $L^\infty(\Omega)$ norm, $J(u)$ is the limit of $J(u_j)$ and u is a solution of (4.7). □

Remark 4.2. When $d = 2$, it is also possible to prove the existence of solutions for the (unconstrained) penalized problem $\min_{g\in L^2(\partial\Omega_{\mathrm{N}})} J(u_g) + \frac{\alpha}{2}\|g\|^2_{L^2(\partial\Omega_{\mathrm{N}})}$, with $\alpha > 0$. Indeed, taking $p = 3$, we obtain $(d-1)p/d = 3/2 < 2$, and thus $L^2(\partial\Omega_{\mathrm{N}}) \subset W^{1-1/p',p'}(\partial\Omega_{\mathrm{N}})'$. As the coerciveness of the penalization term makes sure that the minimizing sequence (g_j) is bounded in L^2, Proposition 3.12 yields a uniform bound on the Hölder norm of the (u_j) and the existence of a solution follows.

4.2.2 Extension to the linear elastic system

Given the theorems introduced in Section 3.3.3, we cannot prove the existence of solutions for problem (2.13) with linear elasticity as state equation without any further assumption. In order to obtain Hölder estimates for solutions to the linear elastic system, we need to obtain $q > d$ in Proposition 3.13. This is guaranteed in dimension 2, but not in dimension 3. In particular, q depends on Lamé coefficients and on the domain regularity. Since it seems restrictive to assume that the boundary of a liver is very regular, it is difficult to be sure that the domain is regular enough.

However, we can still have a look at the necessary modifications to adapt the existence proof to linear elasticity in dimension 2. We also highlight a specificity of dimension 2: the L^∞ constraint is not necessary in the proof provided that the penalty coefficient is positive (see Remark 4.2).

Theorem 4.2. *Assume that $d = 2$ and Ω_0 is Gröger-regular. If $\alpha > 0$ or $M < \infty$, then (2.13) has at least one solution.*

Proof. As previously, consider a minimizing sequence (g_j) of points in $L^2(\partial\Omega_{\mathrm{N}})$ and denote by (u_j) the sequence of associated states. Either due to the L^∞ constraint ($M < \infty$) or to the penalty term coercivity ($\alpha > 0$), (g_j) is bounded in $L^2(\partial\Omega_{\mathrm{N}})$ and converges weakly in L^2 toward $g \in L^2(\partial\Omega_{\mathrm{N}})$, up to a subsequence. As a consequence, we obtain

$$\forall v \in H^1_{\mathrm{D}}(\Omega_0) \quad \int_{\Omega_0} A\varepsilon(u_j) : \varepsilon(v)\,\mathrm{d}x \to \int_{\Omega_0} A\varepsilon(u) : \varepsilon(v)\,\mathrm{d}x,$$

where $u = u_g$ is the state associated to g. As the linear elasticity bilinear form is a scalar product on $H^1_{\mathrm{D}}(\Omega_0)$ (see Lemma 3.2), the previous equation means $u_j \rightharpoonup u$ in $H^1(\Omega_0)$.

We now use a compact embedding to prove that (u_j) converges uniformly toward u. Using Proposition 3.13, there is a $q > 2$ such that $g_j \in W^{-1,p}_{\mathrm{D}}(\Omega_0) \Rightarrow u_j \in W^{1,p}_{\mathrm{D}}(\Omega_0)$ for any $p \in [2, q]$. We choose $p = \min(4, q)$, so that $(d-1)p/d = p/2 \leqslant 2$. As mentioned in Remark 4.2, the embedding $L^2(\partial\Omega_{\mathrm{N}}) \hookrightarrow W^{1-1/p',p'}(\partial\Omega_{\mathrm{N}})'$ is continuous in dimension 2. Therefore, (g_j) is uniformly bounded as a sequence of $W^{-1,p}_{\mathrm{D}}(\Omega_0)$ and, using the estimate in Proposition 3.13, (u_j) is uniformly bounded as a sequence of $W^{1,p}_{\mathrm{D}}(\Omega_0)$. The Rellich-Kondrachov theorem (Adams and Fournier, 2003, Chapter 6, Theorem 6.3) states that the imbedding $W^{1,p}(\Omega_0) \hookrightarrow C(\overline{\Omega}_0)$ is compact, and, up to a subsequence, (u_j) converges uniformly toward u.

As J is continuous for the uniform convergence norm, $J(u) = \lim J(u_j)$. In addition, (g_j) converges weakly in $L^2(\partial\Omega_{\mathrm{N}})$ and $\|g\|_{L^2(\partial\Omega_{\mathrm{N}})} \leqslant \liminf \|g_j\|_{L^2(\partial\Omega_{\mathrm{N}})}$. Finally, g is a solution of problem (2.13). □

In dimension 3, the existence of solutions requires $M < \infty$ and is subject to the condition that the q in Proposition 3.13 satisfy $q > 3$. We did not find a sufficient condition in the literature to enforce this inequality.

4.2.3 A problem with stronger hypotheses

In this section, we transform problem (2.13) to apply Proposition 3.14. To this end, strong hypotheses are necessary. First, Ciarlet (1988) explains that the result is not valid if the interface $\overline{\partial\Omega_{\mathrm{D}}} \cap \overline{\partial\Omega_{\mathrm{N}}}$ is not empty. For this reason, we consider the problem with Neumann conditions. For the sake of simplicity, we consider the case $p = 2$.

The surface loading field g must be the trace on $\partial\Omega_0$ of a function in $H^1(\Omega_0)$. It is measured using the norm

$$\|g\|_{H^{1/2}(\partial\Omega_0)} = \inf\left\{\|w\|_{H^1(\Omega_0)}, w \in H^1(\Omega_0), w|_{\partial\Omega_0} = g\right\}.$$

The problem we consider is

$$\min_{g\in\mathcal{W}_M} J(u_g) \quad \text{subject to constraint} \quad \begin{cases} \operatorname{div}(A\varepsilon(u)) &= 0 \quad \text{in } \Omega_0 \\ A\varepsilon(u)\cdot n &= g \quad \text{on } \partial\Omega_0, \\ \int_{\Omega_0} u\,\mathrm{d}x &= 0 \end{cases} \tag{4.8}$$

where

$$\mathcal{W}_M = \left\{ g \in H^{1/2}(\partial\Omega_0) \;\middle|\; \|g\|_{H^{1/2}(\partial\Omega_0)} \leqslant M \text{ and } \forall w \in R(\Omega_0) \quad \int_{\partial\Omega_0} g\cdot w\,\mathrm{d}s = 0 \right\}.$$

Theorem 4.3. *Assume that $d = 3$, Ω_0 has a C^2 boundary and $p = 2$. Then problem (4.8) has at least one solution.*

Proof. We denote by (g_j) a minimizing sequence. As (g_j) is bounded in $H^{1/2}(\partial\Omega_0)$, it is bounded in $L^2(\partial\Omega_0)$ and converges weakly in $L^2(\partial\Omega_0)$, up to a subsequence, toward a limit $g \in L^2(\partial\Omega_{\mathrm{N}})$.

Solutions to the elastic problem with Neumann boundary conditions are equal up to a rigid translation. The set of equivalence classes is denoted $H^2(\Omega_0)/T(\Omega_0)$. By adding the condition $\int_{\Omega_0} u\,\mathrm{d}x = 0$, we select the class representative with the least H^2-norm. In other words, the state $u_j \in H^2(\Omega_0)$ associated to g_j satisfies

$$\|u_j\|_{H^2(\Omega_0)} = \min_{w\in R(\Omega_0)} \|u_j + w\|_{H^2(\Omega_0)} = \|u_j\|_{H^2(\Omega_0)/R(\Omega_0)}, \tag{4.9}$$

where notation is defined in (3.16).

Due to Proposition 3.14, the sequence (u_j) is uniformly bounded in $H^2(\Omega_0)$. Due to the continuous imbedding $H^2(\Omega_0) \hookrightarrow W^{1,6}(\Omega_0)$, (u_j) is also bounded in $W^{1,6}(\Omega_0)$, and it converges uniformly, up to a subsequence, toward a limit u. Therefore $J(u_j) \xrightarrow[j\to\infty]{} J(u)$.

We check that u is a solution associated to g. As the sequence (u_j) converges uniformly, it also converges weakly in $L^2(\Omega_0)$. Now, as (u_j) is bounded in $H^1(\Omega)$, it converges H^1-weakly, up to a subsequence, toward u. This last remark, together with the

elasticity equation and the weak convergence of (g_j) in $L^2(\partial\Omega_0)$, yields

$$\forall v \in H^1(\Omega_0) \int_{\Omega_0} A\varepsilon(v) : \varepsilon(u)\,\mathrm{d}x = \lim_{j\to\infty} \int_{\Omega_0} A\varepsilon(v) : \varepsilon(u_j)\,\mathrm{d}x$$
$$= \lim_{j\to\infty} \int_{\partial\Omega_0} v \cdot g_j\,\mathrm{d}s = \int_{\partial\Omega_0} v \cdot g\,\mathrm{d}s.$$

Finally, u is a state associated to g and g is a solution of problem (4.8). □

4.3 Optimality conditions

In this paragraph we derive optimality conditions for the optimal control problem (2.13). When J is Gateaux differentiable, it is standard to use an adjoint state p to transform descent directions in the space of states into descent directions in the space of controls. The adjoint state associated with a state u where J is differentiable is solution to the adjoint problem

$$\forall q \in H^1_{\mathrm{D}}(\Omega_0) \qquad \int_{\Omega_0} A\varepsilon(p) : \varepsilon(q)\,\mathrm{d}x = \langle \mathrm{d}J(u), q\rangle\,. \tag{4.10}$$

If $\mathrm{d}J(u)$ is in $H^{-1}_{\mathrm{D}}(\Omega_0)$, then, due to the Lax-Milgram theorem, p is unique and belongs to $H^1_D(\Omega_0)$. In the following optimality conditions, we make the hypothesis $\mathrm{d}J(u) \in H^{-1}_{\mathrm{D}}(\Omega_0)$, keeping in mind that optimality conditions are useful for the numerical application, where everything is much simpler (and finite-dimensional).

In a similar way to Lemma 4.3, we consider the case of minimizers for the unconstrained problem (2.11), where the objective function is necessarily differentiable. The corresponding optimality conditions are stated in the following proposition.

Proposition 4.3. *Let $g \in L^\infty(\partial\Omega_{\mathrm{N}})$ a local (unconstrained) minimizer of the application $F : g \mapsto J(u_g) + R(g)$. Then F is differentiable at g, and g satisfies the first-order optimality condition*

$$\forall h \in L^\infty(\partial\Omega_{\mathrm{N}}) \qquad \int_{\partial\Omega_{\mathrm{N}}} p \cdot h\,\mathrm{d}s + \langle \mathrm{d}R(g), h\rangle = 0,$$

where p is the adjoint state defined by (4.10).

Proof. For $g \in L^\infty(\partial\Omega_{\mathrm{N}})$ a control, the directional derivative of F in a direction $h \in L^\infty(\partial\Omega_{\mathrm{N}})$ reads

$$\mathrm{d}F(g)(h) = \mathrm{d}J(u_g)(w_h) + \langle \mathrm{d}R(g), h\rangle,$$

where $w_h \in H^1_{\mathrm{D}}(\Omega_0)$ is the state associated to h, defined by

$$\forall v \in H^1_{\mathrm{D}}(\Omega_0) \qquad \int_{\Omega_0} A\varepsilon(w_h) : \varepsilon(v)\,\mathrm{d}x = \int_{\partial\Omega_{\mathrm{N}}} h \cdot v\,\mathrm{d}s. \tag{4.11}$$

Now, if g is a minimizer of F, then for every direction h, g satisfies $\mathrm{d}F(g)(h) \geqslant 0$. Assume that there is a direction h such that $\mathrm{d}F(g)(h) > 0$. Then, using Lemma 4.2, we obtain

$$\mathrm{d}F(g)(-h) = \mathrm{d}J(u_g)(-w_h) - \langle \mathrm{d}R(g), h\rangle \leqslant -\mathrm{d}J(u_g)(w_h) - \langle \mathrm{d}R(g), h\rangle < 0.$$

In other words, $-h$ is a descent direction, which contradicts g being a local minimizer. As a consequence, a local minimizer g satisfies $\mathrm{d}F(g)(h) = 0$ for every direction $h \in L^\infty(\partial\Omega_{\mathrm{N}})$. Therefore, F is differentiable at g with $\mathrm{d}F(g) \equiv 0$.

In particular, J is differentiable at u_g. We combine (4.11) and (4.10) to obtain an expression of $\mathrm{d}J(u)$ as a function of h,

$$\langle \mathrm{d}J(u), w_h\rangle = \int_{\Omega_0} A\varepsilon(p) : \varepsilon(w_h)\,\mathrm{d}x = \int_{\partial\Omega_{\mathrm{N}}} p \cdot h\,\mathrm{d}s.$$

Finally, the differential of F reads

$$\langle \mathrm{d}F(g), h\rangle = \langle \mathrm{d}J(u), w_h\rangle + \langle \mathrm{d}R(g), h\rangle = \int_{\partial\Omega_{\mathrm{N}}} p \cdot h\,\mathrm{d}s + \langle \mathrm{d}R(g), h\rangle = 0,$$

hence the optimality condition. □

The proof of Proposition 4.3 relies on the fact that every direction $h \in L^\infty(\partial\Omega_{\mathrm{N}})$ is an admissible direction. In particular if h is an admissible direction, then $-h$ is also an admissible direction. When the optimization problem involves a L^∞ constraint, the cone of admissible directions is not the whole space anymore, in particular when the constraint is active. As a consequence, we cannot expect J to be Gateaux differentiable at local minimizers for the constrained problem.

When J is not differentiable, J can be approximated at first order using several linear forms ℓ (defined below), associated to families $(x_y)_{y\in\Gamma}$ (see Section 4.1 for more details). Each linear form approximates J on a cone of directions. Those multiple linear forms are represented in the space of controls by as many adjoint states p_ℓ, which satisfy

$$\forall q \in H^1_{\mathrm{D}}(\Omega_0) \qquad \int_{\Omega_0} A\varepsilon(p_\ell) : \varepsilon(q)\,\mathrm{d}x = \langle \ell, q\rangle = \int_\Gamma q(x_y) \cdot (x_y + u(x_y) - y)\,\mathrm{d}y. \quad (4.12)$$

For a given direction h in the space of controls, there is a linear form ℓ_0 associated to a family $(x_{y,0})$, such that

$$\mathrm{d}J(u_g)(w_h) = \langle \ell_0, w_h\rangle = \int_{\partial\Omega_{\mathrm{N}}} p_{\ell_0} \cdot h\,\mathrm{d}s, \quad (4.13)$$

where w_h is the corresponding direction in the space of displacements.

In the following proposition, we consider the constrained problem (2.13), where $R(g) = \frac{\alpha}{2}\|g\|^2_{L^2(\partial\Omega_{\mathrm{N}})}$, and we do not assume that F is differentiable. As a consequence, we prove an optimality condition that should hold for every adjoint state p_ℓ at this point. When F turns out to be differentiable at g, there is only one adjoint state p that should satisfy the first-order optimality condition below.

Theorem 4.4. *Let $g \in \mathcal{G}_M$ a local minimizer of problem (2.13) and u_g the associated state defined by (2.6). For $(x_y)_{y\in\Gamma}$ a family such that $\forall y \in \Gamma, x_y \in P_y(u_g)$, we consider the linear form ℓ and the adjoint state p_ℓ defined in (4.12). Then there exists a Lagrange multiplier $\lambda_\ell \in L^2(\partial\Omega_{\mathrm{N}}, \mathbb{R})$ with*

$$\text{for a.e. } x \in \partial\Omega_{\mathrm{N}} \qquad \begin{cases} \lambda_\ell(x) & = 0 \quad \text{if } \|g(x)\| < M \\ \lambda_\ell(x) & \geqslant 0 \quad \text{if } \|g(x)\| = M \end{cases}$$

such that g satisfies the first-order optimality condition

$$\text{for a.e. } x \in \partial\Omega_{\mathrm{N}} \qquad p_\ell(x) + (\alpha + \lambda_\ell(x))g(x) = 0. \tag{4.14}$$

Proof. • We first give a justification for the definition of the adjoint state. We use the notation $F(g) = J(u_g) + \frac{\alpha}{2}\|g\|^2_{L^2(\partial\Omega_{\mathrm{N}})}$. For a direction $h \in L^\infty(\partial\Omega_{\mathrm{N}})$, we consider the associated displacement field w_h , solution of (4.11). By combining (4.11) with (4.12), we obtain

$$\int_\Gamma w_h(x_y) \cdot (x_y + u(x_y) - y)\,\mathrm{d}y = \int_{\Omega_0} A\varepsilon(p_\ell) : \varepsilon(w_h)\,\mathrm{d}x = \int_{\partial\Omega_{\mathrm{N}}} p_\ell \cdot h\,\mathrm{d}s.$$

The derivative of F in the direction h satisfies

$$\begin{aligned} \mathrm{d}F(g)(h) &= \int_\Gamma \min_{x\in P_y(u)} w_h(x) \cdot (x + u(x) - y)\,\mathrm{d}y + \alpha \int_{\partial\Omega_{\mathrm{N}}} g \cdot h\,\mathrm{d}s \\ &\leqslant \int_\Gamma w_h(x_y) \cdot (x_y + u(x_y) - y)\,\mathrm{d}y + \alpha \int_{\partial\Omega_{\mathrm{N}}} g \cdot h\,\mathrm{d}s \\ &= \int_{\partial\Omega_{\mathrm{N}}} (p_\ell + \alpha g) \cdot h\,\mathrm{d}s. \end{aligned}$$

- We now characterize the cone of admissible directions at g. Admissible directions $h \in L^\infty(\partial\Omega_{\mathrm{N}})$ are only constrained at points where the L^∞ constraint is active. We split $\partial\Omega_{\mathrm{N}}$ into two disjoint subsets, the active set $\mathcal{A}$ and the inactive set $\mathcal{I}$, so that $\mathcal{A} \cup \mathcal{I} = \partial\Omega_{\mathrm{N}}$. The two subsets are defined by

$$\mathcal{A} = \{x \in \partial\Omega_{\mathrm{N}} \mid \|g(x)\| = M\} \quad \text{and} \quad \mathcal{I} = \{x \in \partial\Omega_{\mathrm{N}} \mid \|g(x)\| < M\}. \tag{4.15}$$

Admissible directions h satisfy the condition

$$\text{for a.e. } x \in \mathcal{A} \quad h(x) \cdot g(x) \leqslant 0.$$

To be more specific, the cone of admissible directions is the intersection $L^\infty(\partial\Omega_{\mathrm{N}}) \cap K$, where K is the closed cone

$$K = \left\{ h \in L^2(\partial\Omega_{\mathrm{N}}) \;\middle|\; \forall \lambda \in L^2(\partial\Omega_{\mathrm{N}}, \mathbb{R}_+) \quad \int_{\mathcal{A}} \lambda(h \cdot g)\,\mathrm{d}s \leqslant 0 \right\}.$$

Another expression for $K \subset L^2(\partial\Omega_{\mathrm{N}})$ reads

$$K = \left\{ h \in L^2(\partial\Omega_{\mathrm{N}}) \mid \forall q \in Q \quad \langle h, q \rangle_{L^2(\partial\Omega_{\mathrm{N}})} \leqslant 0 \right\},$$

where

$$Q = \left\{ \mathbf{1}_{\mathcal{A}} \lambda g, \lambda \in L^2(\partial\Omega_{\mathrm{N}}, \mathbb{R}_+) \right\}.$$

Here, Q is the polar cone of K (Rockafellar, 1970, Section 14). As Q is convex, the orthogonal projection s_Q of $-(p_\ell + \alpha g)$ onto Q satisfies

$$\forall q \in Q \quad \langle (p_\ell + \alpha g) + s_Q, q \rangle = -\langle -(p_\ell + \alpha g) - s_Q, q - s_Q \rangle \geqslant 0.$$

Therefore, $-(p_\ell + \alpha g) - s_Q$ belongs to K, and we obtain the orthogonal decomposition $(p_\ell + \alpha g) = -s_Q - s_K$, with $s_Q \in Q$ and $s_K \in K$.

- To derive the first-order optimality condition, we now consider the admissible direction

$$h = \frac{s_K}{\max(1, \|s_K\|)} \in L^\infty(\partial\Omega_{\mathrm{N}}) \cap K.$$

As g is a local minimizer, it results from Euler's inequation that

$$0 \leqslant \mathrm{d}F(g)(h) \leqslant \int_{\partial\Omega_{\mathrm{N}}} (p_\ell + \alpha g) \cdot h \, \mathrm{d}s = -\int_{\partial\Omega_{\mathrm{N}}} \min\left(\|s_K\|, \|s_K\|^2 \right) \mathrm{d}s.$$

Finally, we obtain $s_K = 0$, which yields the first-order optimality condition $p_\ell + \alpha g + s_Q = 0$.

□

We now turn to the problem with subdomain restriction. For the sake of simplicity, we assume that J is Gateaux differentiable at the considered local minimizer. However, we keep in mind that, in the case of a non-differentiable minimizer, the derived optimality condition applies to every adjoint state p_ℓ. Problem (2.14) involves a differentiable constraint on the state. We manage this state constraint using another adjoint state $z \in H^1_{\mathrm{D}}(\Omega_0)$, defined by

$$\forall q \in H^1_{\mathrm{D}}(\Omega_0) \quad \int_{\Omega_0} A\varepsilon(z) : \varepsilon(q) \, \mathrm{d}x = \int_{\omega_0} u \cdot q \, \mathrm{d}x. \tag{4.16}$$

Though we derive optimality conditions for this problem, the state-constrained approach is not considered in the following chapters.

Theorem 4.5. *Let $g \in L^\infty(\partial\Omega_{\mathrm{N}})$ a local minimizer for problem (2.14) and let u_g the associated state defined by (2.6). Assume also that J is differentiable at u_g with $\mathrm{d}J(u_g) \in H^{-1}_{\mathrm{D}}(\Omega_0)$. Then there exists a Lagrange multiplier $\lambda \in L^2(\partial\Omega_{\mathrm{N}}, \mathbb{R})$ with*

$$\text{for a.e. } x \in \partial\Omega_{\mathrm{N}} \qquad \begin{cases} \lambda(x) & = 0 \quad \text{if } \|g(x)\| < M \\ \lambda(x) & \geqslant 0 \quad \text{if } \|g(x)\| = M \end{cases}$$

and a real multiplier $\eta \in \mathbb{R}_+$ such that g satisfies the first-order optimality condition

$$\text{for a.e. } x \in \partial\Omega_{\mathrm{N}} \qquad p(x) + \lambda(x)g(x) + \eta z(x) = 0. \tag{4.17}$$

Moreover, one has either $\eta = 0$ or $\int_{\omega_0} \|u\|^2 \,\mathrm{d}x = U$.

Proof. We use the notation $F(g) = J(u_g)$, $C(g) = \int_{\omega_0} \|u_g\|^2 \,\mathrm{d}x$, and for a direction $h \in L^\infty(\partial\Omega_{\mathrm{N}})$, w_h is the associated state defined by (4.11). Due to their definitions, the adjoint states p and z satisfy

$$\forall h \in L^\infty(\partial\Omega_{\mathrm{N}}) \quad \langle \mathrm{d}F(g), h\rangle = \int_{\partial\Omega_{\mathrm{N}}} p \cdot h \,\mathrm{d}s \quad \text{and} \quad \langle \mathrm{d}C(g), h\rangle = \int_{\partial\Omega_{\mathrm{N}}} z \cdot h \,\mathrm{d}s$$

We also define $\mathcal{A}$ and $\mathcal{I}$ as previously.

If $C(g) < U$, i.e. the constraint on u_g is inactive, the same procedure as above results in (4.14), which is (4.17) with $\eta = 0$.

Now we consider the case $C(g) = U$. Admissible directions h satisfy $h \cdot g \leqslant 0$ in $\mathcal{A}$ and $\int_{\partial\Omega_{\mathrm{N}}} z \cdot h \,\mathrm{d}s \leqslant 0$. It is easy to check that such directions h belong to $L^\infty(\partial\Omega_{\mathrm{N}}) \cap K$, where the closed cone $K \subset L^2(\partial\Omega_{\mathrm{N}})$ is defined by

$$K = \left\{ h \in L^2(\partial\Omega_{\mathrm{N}}) \,\middle|\, \forall \lambda \in L^2(\partial\Omega_{\mathrm{N}}, \mathbb{R}_+) \quad \int_{\mathcal{A}} \lambda(h \cdot g) \,\mathrm{d}s \leqslant 0 \text{ and } \int_{\partial\Omega_{\mathrm{N}}} z \cdot h \,\mathrm{d}s \leqslant 0 \right\}.$$

Let us check that the polar cone to K reads

$$Q = \left\{ \mathbf{1}_{\mathcal{A}} \lambda g + \eta z, \lambda \in L^2(\partial\Omega_{\mathrm{N}}, \mathbb{R}_+), \eta \in \mathbb{R}_+ \right\}.$$

Consider $q = \mathbf{1}_{\mathcal{A}} \lambda g + \eta z \in Q$, with $\lambda \in L^2(\partial\Omega_{\mathrm{N}}, \mathbb{R}_+)$ and $\eta \in \mathbb{R}_+$. Then for every $h \in K$,

$$\langle h, q\rangle_{L^2(\partial\Omega_{\mathrm{N}})} = \int_{\mathcal{A}} \lambda(h \cdot g) \,\mathrm{d}s + \eta \int_{\partial\Omega_{\mathrm{N}}} z \cdot h \,\mathrm{d}s \leqslant 0.$$

Now consider a direction $h \in L^2(\partial\Omega_{\mathrm{N}})$ such that $\langle q, h\rangle_{L^2(\partial\Omega_{\mathrm{N}})} \leqslant 0$ for all $q \in Q$. By considering $q_1 = \mathbf{1}_{\mathcal{A}} \lambda g \in Q$ and $q_2 = z \in Q$, we obtain $h \in K$. Therefore, for $h \in L^2(\partial\Omega_{\mathrm{N}})$, the equivalence

$$h \in K \Leftrightarrow \forall q \in Q \quad \langle h, q\rangle_{L^2(\partial\Omega_{\mathrm{N}})} \leqslant 0$$

holds and Q is the polar cone to K.

As Q is convex, the orthogonal projection p_Q of $-p$ onto Q satisfies

$$\forall q \in Q \quad \langle p + p_Q, q\rangle = -\langle -p - p_Q, q - p_Q\rangle \geqslant 0,$$

which results in the unique orthogonal decomposition $p = -p_Q - p_K$ with $p_Q \in Q$ and $p_K \in K$.

If we consider the admissible direction $h = \frac{p_K}{\max(1, \|p_K\|)}$, Euler's inequation results in

$$0 \leqslant \int_{\partial\Omega_{\mathrm{N}}} h \cdot p \,\mathrm{d}s = -\int_{\partial\Omega_{\mathrm{N}}} \min\left(\|p_K\|, \|p_K\|^2\right) \mathrm{d}s.$$

As a consequence, $p_K = 0$ and $p + p_Q = 0$, which is the optimality condition we were looking for. □

Chapter 5

An adjoint method to solve the registration problem

To solve the registration problem numerically, we discretize the optimal control formulation using the finite element method. The finite element method is relevant to handle domains with all kinds of shapes. For this reason it is widely used in the Mimesis team, especially through the home-brewed SOFA framework (Allard et al., 2007). For all numerical investigations, we adopt a discretize-then-optimize approach, which means that the entire formulation is transformed into a finite-dimensional problem, and then solved using numerical tools for finite-dimensional optimization. The reasons for this choice are multiple. First, as we do not change or refine meshes during the online procedure, the problem dimension is fixed and going back to continuous considerations is not necessary. In addition, using expressions from the continuous framework may represent a lack of robustness, as the solver may fail because the discretized problem does not approximate the continuous problem well enough. We really need our numerical methods to be as simple as possible, and forgetting that the liver mesh actually represents a real liver seems to avoid unnecessary considerations, at the expense of continuous consistency.

In this chapter, we give some details about our implementation of an adjoint method. This choice results from our initial intention to use shape optimization. Adjoint methods are also relevant in optimal control when the direct problem is handled by a black-box software such as SOFA, and for this reason we kept using an adjoint method. Modularity is actually a key advantage of the adjoint method, as it can be broken down into several procedures, each of which can be handled by a separate protagonist. In our case, these procedures consist in evaluating the registration functional, managing the elastic model, and solving the optimization problem, respectively.

After defining notations to express the registration problem in terms of matrix and vector operations, we present an overview of the adjoint method and its structure. In the end of the chapter we consider separately each procedure involved in the adjoint pipeline. We first describe the chosen option to perform orthogonal projections onto the mesh boundary when evaluating the cost function. Then, we mention Newton methods that can be used to solve nonlinear elastic problems. Finally, we discuss the choice of an

optimization solver and show some examples to illustrate their convergence properties.

5.1 Finite element discretization of the problem

In order to perform numerical computations, we derive a discretized version of the problem using the finite element method. The domain Ω_0 is represented by a tetrahedral mesh $\mathcal{T}_0$, and functions of interest are represented by P1 finite element functions defined on $\mathcal{T}_0$. When a deformation is applied to the mesh $\mathcal{T}_0$, the deformed mesh is known through the position of its vertices $x_1, \cdots, x_n \in \mathbb{R}^3$. We denote by $\mathbf{x} = (x_1, \cdots, x_n)$ the vector containing the coordinates of all vertices of the deformed mesh, and $\mathbf{x}^0$ the positions vector in the reference configuration. We also use $\mathbf{u} = \mathbf{x} - \mathbf{x}^0 = (u_1, \cdots, u_n)$ to denote the displacement of vertices with respect to the initial configuration. With a slight abuse of notation, we use the same bold letter $\mathbf{u}$ to denote the P1 finite element function associated to the displacement field in $\mathcal{T}_0$. As a P1 finite element function is defined by its value at the mesh vertices, a finite element function and a vector represented by the same letter actually contain exactly the same information.

In the discrete framework, we write the linear elasticity equation (3.15) using the matrix formulation

$$\mathbf{A}\mathbf{u} = \mathbf{S}\mathbf{g}, \tag{5.1}$$

where the force distribution $\mathbf{g}$ is a P1 finite element function defined only on the boundary elements of $\partial\Omega_{\mathrm{N}}$. If we denote by $I_{\mathbf{g}}$ the set of indices associated to vertices of $\partial\Omega_{\mathrm{N}}$, one can write $\mathbf{g} = (g_i)_{i \in I_{\mathbf{g}}}$, resulting in a vector $\mathbf{g}$ smaller than $\mathbf{u}$. Here, the stiffness matrix $\mathbf{A}$ stands for the linear elasticity inner product and satisfies

$$\mathbf{v}^{\mathrm{T}}\mathbf{A}\mathbf{u} = \int_{\Omega_0} A\varepsilon(\mathbf{u}) : \varepsilon(\mathbf{v}) \, \mathrm{d}x,$$

while the matrix $\mathbf{S}$ represents the L^2 inner product on $\partial\Omega_{\mathrm{N}}$, which satisfies the equality

$$\mathbf{u}^{\mathrm{T}}\mathbf{S}\mathbf{g} = \int_{\partial\Omega_{\mathrm{N}}} \mathbf{u} \cdot \mathbf{g} \, \mathrm{d}s.$$

Here, $\mathbf{A}$ and $\mathbf{S}$ were obtained by expressing the involved bilinear forms in a basis of the finite element space. In the context of nonlinear elasticity, a similar development involving the same basis leads to a nonlinear system of the form

$$\mathbf{F}(\mathbf{u}) = \mathbf{S}\mathbf{g}.$$

We give more details about this system at the end of the chapter.

Though we produced an early implementation where we control the surface force distribution $\mathbf{g}$, we proceed in a different way in our main implementation, for two reasons. First, choosing $\mathbf{g}$ as an optimization variable would require to perform several products between $\mathbf{S}$ and a vector. Namely, at least two products involving $\mathbf{S}$ or its adjoint per evaluation of the objective function would be necessary, whereas in a real-time context, it

is preferable to make computations as light as possible. In addition, the SOFA framework does not provide an implementation of $\mathbf{S}$. As a consequence, we control the nodal forces instead, which is consistent with the habits of the Mimesis team. The nodal forces are represented by the vector $\mathbf{b} = \mathbf{S}\mathbf{g} = (b_1, \cdots, b_n)$. An element $b_k \in \mathbb{R}^3$ can be interpreted as the force (in Newtons) that applies on the k-th node, so that the work of the force distribution (also known as compliance), reads

$$\int_{\partial\Omega_N} \mathbf{u} \cdot \mathbf{g} \, \mathrm{d}s = \mathbf{u}^T \mathbf{b} = \sum_{i=1}^{n} u_i \cdot b_i.$$

In this context, the Neumann boundary $\partial\Omega_N$ is defined in terms of nodes where we allow b_k to be nonzero. In particular, b_k is zero for every node that is not on the mesh boundary. The state equation in the linear case and in the nonlinear case reads

$$\mathbf{A}\mathbf{u} = \mathbf{b} \qquad \text{and} \qquad \mathbf{F}(\mathbf{u}) = \mathbf{b},$$

respectively. From now on, in the manuscript, we use the $\mathbf{b}$-formalism, even to describe optimization methods from the early implementation that are implemented using the $\mathbf{g}$-formalism.

In the context of the adjoint method, the optimization procedure is performed in the space of nodal forces vectors $\mathbf{b}$. The inner product used by generic optimization solvers to compute descent directions and measure optimality conditions is the standard inner product in $\mathbb{R}^{3n}$ defined by $\langle \mathbf{u}, \mathbf{v} \rangle = \mathbf{u}^T \mathbf{v} = \sum u_i \cdot v_i$. As a consequence, in this chapter gradients and derivatives are always expressed using this inner product, with the notation

$$\langle \mathrm{d}f(\mathbf{u}), \mathbf{v} \rangle = \nabla f(\mathbf{u})^T \mathbf{v} = \sum_{i=1}^{n} \partial_i f(\mathbf{u}) \cdot v_i, \tag{5.2}$$

where f is a scalar function.

A consequence of this choice is a less significant consistency with the continuous optimal control problem, as the space of nodal forces is not a finite element space, and the vector of nodal forces $\mathbf{b}$ has no counterpart in the continuous problem. In addition, as opposed to inner products in function spaces, the standard inner product in $\mathbb{R}^{3n}$ is completely blind to geometric information such as the size of triangles. In other words, we are not even trying to perform iterations that approximate iterations in the continuous case. However, every standard optimization solver uses the standard inner product in the discrete vector space. To be consistent with the continuous case, we would need to implement optimization solvers that work with the L^2 inner product, at the cost of several additional matrix-vector products and system inversions. Finally, working with nodal forces at the expense of consistency with the continuous framework is relevant in view of real-time requirements and available software.

5.2 Adjoint method

The adjoint method is a standard approach for solving optimal control problems, based on the optimality conditions derived in Section 4.3. In particular, an adjoint state $\mathbf{p}$ is

used to compute the derivatives of the objective function with respect to the control. A general presentation of the adjoint method is given in Section 5.2.1.

In an adjoint method, the direct problem and the optimization problem are handled in separate procedures, which makes the resulting program very modular. In Section 5.2.2, we give a few details about our implementation of the adjoint method.

5.2.1 Overview of the method

The adjoint method is very specific to optimal control problems, where the entire system state is defined by the choice of a control. We consider the discrete version of the optimal control problem (2.11), featuring the nodal force vector $\mathbf{b}$ in the role of the control, and the displacement field $\mathbf{u}$ as the state. A possible formulation reads

$$\min_{\mathbf{u}\in\mathbb{R}^{3n},\mathbf{b}\in\mathcal{B}} \quad J(\mathbf{u}) + R(\mathbf{b}) \quad \text{subject to constraint} \quad \mathbf{F}(\mathbf{u}) = \mathbf{b}, \tag{5.3}$$

where $\mathcal{B}$ is the set of admissible controls. Though the elastic system $\mathbf{F}(\mathbf{u}) = \mathbf{b}$ is denoted as an optimization constraint, it is a particular constraint, as a given control $\mathbf{b}$ yields a unique state $\mathbf{u_b}$ so that the pair $(\mathbf{u_b}, \mathbf{b})$ is feasible. In an adjoint method, everything is expressed as a function of $\mathbf{b}$, making $\mathbf{b}$ the only variable controlled by the optimization solver. The resulting optimization problem reads

$$\min_{\mathbf{b}\in\mathcal{B}} \Phi(\mathbf{b}) \qquad \text{where} \qquad \Phi(\mathbf{b}) = J(\mathbf{u_b}) + R(\mathbf{b}). \tag{5.4}$$

Here, the elastic system is hidden in the objective function Φ. In particular, each evaluation of the objective value $\Phi(\mathbf{b})$ requires to solve the elastic system to compute $\mathbf{u_b}$.

Now, to solve (5.4) using a first-order optimization method, computing the objective gradient $\nabla\Phi(\mathbf{b})$ is also required. The derivation is similar to that in Section 4.3. If $\mathbf{h}$ is an admissible direction in the space of controls, the corresponding direction in the space of states $\mathbf{w}$ is solution of the tangent system $\mathbf{F}'(\mathbf{u_b})\mathbf{w} = \mathbf{h}$, where $\mathbf{F}'$ denotes the Jacobian of the elastic residual. With this notation, the first-order variation of Φ reads

$$\nabla\Phi(\mathbf{b})^{\mathrm{T}}\mathbf{h} = \nabla J(\mathbf{u})^{\mathrm{T}}\mathbf{w} + \nabla R(\mathbf{b})^{\mathrm{T}}\mathbf{h}.$$

The adjoint state $\mathbf{p}$ is used to transform the functional gradient $\nabla J(\mathbf{u_b})$ in the space of states into a gradient in the space of controls. It is defined as the solution to the adjoint system $\mathbf{F}'(\mathbf{u_b})^{\mathrm{T}}\mathbf{p} = \nabla J(\mathbf{u_b})$. Note that here the constraint Jacobian $\mathbf{F}'(\mathbf{u_b})$ is a symmetric matrix, as it represents the Hessian of the discretized elastic energy. We obtain

$$\nabla J(\mathbf{u_b})^{\mathrm{T}}\mathbf{w} = \mathbf{p}^{\mathrm{T}}\mathbf{F}'(\mathbf{u_b})\mathbf{w} = \mathbf{p}^{\mathrm{T}}\mathbf{h},$$

and, finally,

$$\nabla\Phi(\mathbf{b}) = \mathbf{p} + \nabla R(\mathbf{b}).$$

This gradient can then be used by an optimization solver to minimize Φ. Algorithm 1 gives a summary of the adjoint procedure to compute $\nabla\Phi(\mathbf{b})$. Problem (5.4) can be fed to

generic first-order primal solvers, such as gradient descents or quasi-Newton solvers. By using solvers for constrained optimization problems, it is possible to handle constraints on $\mathbf{b}$. If the optimization problem involves constraints on $\mathbf{u}$, however, calculations similar to those above are necessary to transform those constraints into constraints on $\mathbf{b}$.

Algorithm 1: Computation of the objective gradient using an adjoint method.

Data: Current iterate $\mathbf{b}$
Compute the displacement $\mathbf{u}$ by solving $\mathbf{F}(\mathbf{u}) = \mathbf{b}$
Evaluate $J(\mathbf{u})$ and $\nabla J(\mathbf{u})$
Compute the adjoint state $\mathbf{p}$ by solving $\mathbf{F}'(\mathbf{u})^{\mathrm{T}}\mathbf{p} = \nabla J(\mathbf{u})$
Result: $\nabla\Phi(\mathbf{b}) = \mathbf{p} + \nabla R(\mathbf{b})$

Remark 5.1. In the space of displacements, the first-order descent direction for J returned by the adjoint method reads

$$\mathbf{w} = \mathbf{F}'(\mathbf{u})^{-1}\mathbf{F}'(\mathbf{u})^{-\mathrm{T}}\nabla J(\mathbf{u}),$$

Though it ensures the regularity and feasibility of displacement fields, this expression seems less natural than $\nabla J(\mathbf{u})$ as a descent direction for J, due to the change of metrics created by the elastic model. If the Jacobian $\mathbf{F}'(\mathbf{u})$ is poorly conditioned, using an adjoint method can degrade the numerical properties of the problem, making convergence more difficult.

As mentioned earlier, implementing an adjoint method is very convenient, as the direct solver and the optimization solver are separated from each other. In the next section, we briefly describe the software produced during the thesis.

5.2.2 A modular implementation of the adjoint method

We first produced an early implementation of the adjoint method using the FreeFem++ software (Hecht, 2012). This implementation was used to test the adjoint method on two-dimensional examples, with homemade procedures and optimization algorithms. Then we switched to Python, in order to benefit from the features of a modern language, such as object oriented programming. It was also the occasion to use state-of-the-art scientific libraries to solve larger-scale problems in a reasonable time. As mentioned above, this early implementation works with the surface force finite element function $\mathbf{g}$ as the control variable, while the Python implementation uses $\mathbf{b}$ as the control variable. For the sake of consistency, we always consider that $\mathbf{b}$ is the control in the description of the implemented procedures.

Our Python implementation involves three separate procedures that communicate together through simple interfaces, using the NumPy linear algebra library. Figure 5.1 shows an overview of the adjoint procedure to evaluate the shape functional and its gradient.

On the one hand, the shape functional procedure is responsible for evaluating the value and gradient of the functional J for a given displacement field $\mathbf{u}$. It is expected to return an accurate gradient, as otherwise the optimization solver would fail to solve the problem with a tight tolerance. This part of the program is implemented using the Trimesh package and involves an exact projection onto the deformed triangular surface of the discretized organ. We give details about this procedure in Section 5.3.

On the other hand, the optimization solver is responsible for making the objective function decrease. In our implementation, only first-order methods are supported, as our shape functional procedure does not provide second-order information. We use the standard solvers available in the Scipy package. In particular, quasi-Newton solvers are the reference choice when only first-order information is available. The choice of an optimization solver is discussed in Section 5.5.

Remark 5.2. Note that using second order solvers such as the Newton method would require careful attention, as the objective Hessian $\nabla^2\Phi$ is full due to the presence of the elastic problem. In addition, even if the Hessian is not assembled, performing Hessian-vector products might require to solve several linear systems, resulting in a heavy computational cost.

Finally, the central element in the implementation is the finite element package, that manages the elastic problem. The finite element package is responsible for connecting the optimization solver, that works in the space of controls, with the shape functional procedure, that works in the space of states. This part of the program is critical for the success of the registration process. First, it should give accurate results for the same reasons as the shape functional procedure. But most of all, the direct solver should not fail, in particular when it involves an iterative algorithm. When a linear elastic model is used, the stiffness matrix is precomputed using Fenics or SOFA, then converted into a Scipy matrix and factorized before the registration starts. Thus, the direct and adjoint problem are solved in a fast and robust way. However, when the elastic model is nonlinear, the direct problem is solved iteratively using a Newton method. In this last case, enforcing convergence requires special attention, as discussed in Section 5.4.

5.3 Discretized shape functional

In this section, we describe a procedure to evaluate the shape functional J numerically. In the discrete framework, the deformed surface $\partial\Omega_{\mathbf{u}}$ is a triangular mesh, and defining S_0 consists in selecting a set of triangles that are used to evaluate J. The observation Γ is provided as the point cloud

$$\Gamma = \{y_1, \cdots, y_p\} \subset \mathbb{R}^d.$$

Due to this lack of structure, the objective function cannot be defined as an integral on Γ. Thus, we use the expression

$$J(\mathbf{u}) = \tfrac{1}{2p} \sum_{i=1}^{p} d^2(y_i, S_{\mathbf{u}}). \tag{5.5}$$

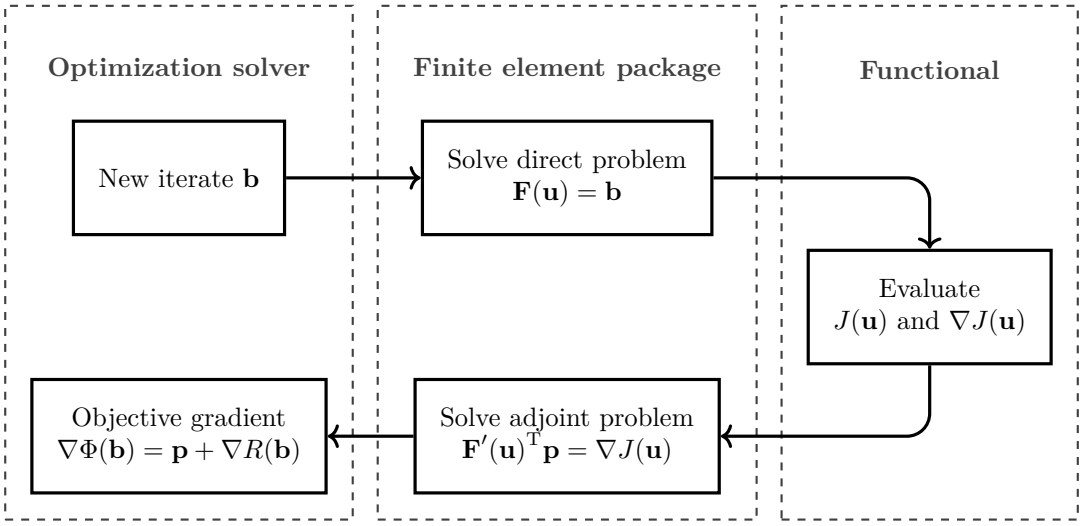


Figure 5.1: Structure of the objective evaluation procedure. The forward chain (top) aims to evaluate the functional for a given control $\mathbf{b}$, while the backward chain (bottom) transforms the gradient $\nabla J(\mathbf{u})$ into a gradient in the space of controls.

With this notation, $\sqrt{2J(\mathbf{u})}$ is the quadratic mean of distances from a point in Γ to the deformed mesh surface.

Beside the fact that the integral is transformed into a sum, the discrete version of J is very close to the continuous one, as the points from Γ are projected *exactly* onto $S_{\mathbf{u}}$. The projection procedure exploits the triangular mesh structure of $S_{\mathbf{u}}$. It involves a nearest neighbor search that determines the closest triangle from y. As one nearest neighbor search is performed per point in Γ, its implementation should be efficient. We discuss such procedures at the end of this section.

First, we briefly explain how the derivative formula computed in Lemma 4.1 adapts to the finite element configuration.

5.3.1 Evaluation of the functional gradient

We now choose a point $y \in \Gamma$ and we derive a practical formula to evaluate the gradient of the elementary application

$$j : \mathbf{u} \mapsto \tfrac{1}{2} d^2(y, S_{\mathbf{u}}). \tag{5.6}$$

The expression $d(y, S_{\mathbf{u}})$ denotes the distance between y and its projection a onto $S_{\mathbf{u}}$. First, note that here the projection point a is supposed to be unique. Actually, even when y has several projection points onto $S_{\mathbf{u}}$, the nearest neighbor search algorithm is supposed to return only one of them. This detail may be a problem when it comes to obtaining reproducible results, as the projection point depends on the search procedure. However, cases with multiple projections are rare, as for almost every displacement field $\mathbf{u}$, each point in Γ has a unique projection. In addition, a case with multiple projections means that several descent directions are available, and by choosing one of

these projections, the algorithm selects one of those directions. Thus, the optimization procedure is not likely to fail because of that.

Since we are dealing with P1 finite element functions, differentiating j with respect to the displacement field $\mathbf{u}$ means differentiating j with respect to the displacements $(u_1, \cdots, u_n)$ at the mesh vertices. Thus, the gradient $\nabla j(\mathbf{u})$ is defined by its vertex-wise components $(\partial_1 j(\mathbf{u}), \cdots, \partial_n j(\mathbf{u}))$. We assume that the projection point a falls into the triangle $x_{k_1} x_{k_2} x_{k_3}$. Provided that a is the only projection point, the nodes x_{k_1}, x_{k_2} and x_{k_3} are the only ones whose displacement has an impact on the first-order variation of j. Indeed, if another mesh node moves, the triangle containing a is not affected and the value of j remains constant at first order. Therefore,

$$\forall k \notin \{k_1, k_2, k_3\} \qquad \partial_k j(\mathbf{u}) = \frac{\partial j}{\partial u_k}(\mathbf{u}) = 0.$$

We now apply Lemma 4.1 to determine the nonzero components of $\nabla j(\mathbf{u})$. We denote by $\theta_1, \theta_2, \theta_3$ the barycentric coordinates of a in the triangle $x_{k_1} x_{k_2} x_{k_3}$, so that

$$a = \theta_1 x_{k_1} + \theta_2 x_{k_2} + \theta_3 x_{k_3},$$

where $\theta_1, \theta_2, \theta_3 \geqslant 0$ and $\theta_1 + \theta_2 + \theta_3 = 1$. If $\mathbf{v}$ is a perturbation of the displacement field $\mathbf{u}$, Lemma 4.1 yields

$$\langle \mathrm{d}j(\mathbf{u}), \mathbf{v} \rangle = \mathbf{v}(a) \cdot (a - y) = \left(\theta_1 v_{k_1} + \theta_2 v_{k_2} + \theta_3 v_{k_3} \right) \cdot (a - y) \,.$$

This expression confirms that $\nabla j(\mathbf{u})$ has at most three nonzero components. Finally, the components of $\nabla j(\mathbf{u})$ read

$$\forall i \in \{1, 2, 3\} \quad \partial_{k_i} j(\mathbf{u}) = \theta_i (a - y) \qquad \forall k \notin \{k_1, k_2, k_3\} \quad \partial_k j(\mathbf{u}) = 0. \tag{5.7}$$

Formula (5.7) contains the cases where a falls onto an edge or a vertex. These cases appear when one or two of the barycentric coefficients are set to zero, respectively. Figure 5.2 illustrates the gradient formula in a two-dimensional case. On the left, y lies in the polar cone to $S_{\mathbf{u}}$ at x_k and its projection falls onto x_k, resulting in a gradient fully supported by the node k. On the right, a belongs to a boundary edge and $\nabla j(\mathbf{u})$ has nonzero components at x_k and x_m, which are proportional to the corresponding barycentric coefficients.

We just computed the gradient of the elementary function j_i associated to a single point $y_i \in \Gamma$. By summing all the elementary gradients, we obtain the global gradient of the functional

$$\nabla J(\mathbf{u}) = \tfrac{1}{p} \sum_{i=1}^{p} \nabla j_i(\mathbf{u}). \tag{5.8}$$

Remark 5.3. Computing the first-order derivatives of j using automatic differentiation is very convenient, as the first-order derivatives of j are the same as the derivatives of the application

$$\mathbf{u} \mapsto \tfrac{1}{2} \| \theta_1 x_{k_1} + \theta_2 x_{k_2} + \theta_3 x_{k_3} - y \|^2,$$

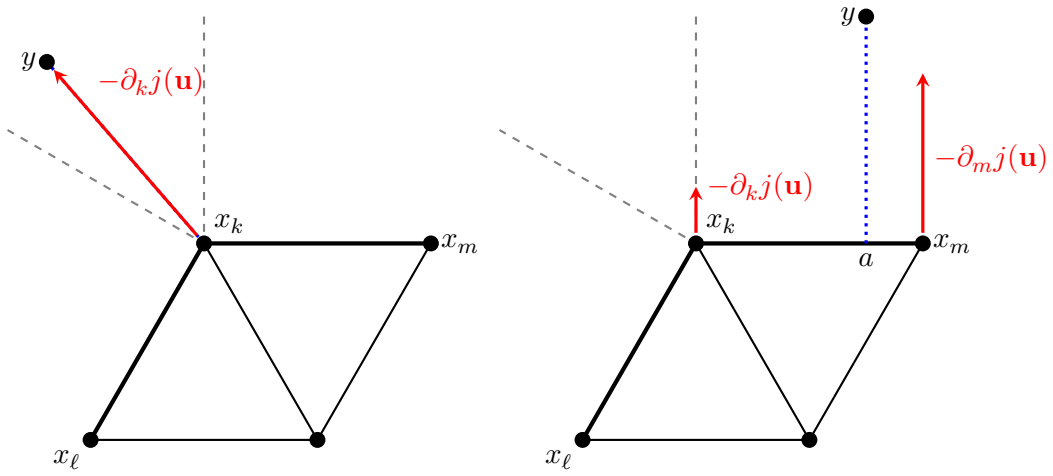


Figure 5.2: Illustration of $\nabla j(\mathbf{u})$ in two configurations. Points x_ℓ, x_k and x_m are located on the mesh boundary, while the last vertex is inside the mesh. The red arrows represent the components of the descent direction $-\nabla j(\mathbf{u})$. On the right, the equality $y - a = -\partial_k j(\mathbf{u}) - \partial_m j(\mathbf{u})$ holds.

with the convention $x_k = x_k^0 + u_k$. In other words, the barycentric coefficients can be treated as constants in the gradient computation. For instance, assume that the nearest neighbor search procedure returns the indices `k1`, `k2`, `k3` and the corresponding barycentric coordinates `t1`, `t2`, `t3`. Then, the elementary gradient $\nabla j(\mathbf{u})$ can be computed using, for instance the following lines in PyTorch:

```
# x is the vector of node positions, y is the point from Gamma.
k1, k2, k3, t1, t2, t3 = nearest_neighbor_procedure(x, y)
x.requires_grad = True # Save operations performed on x from now on.
j = 0.5 * norm(t1*x[k1] + t2*x[k2] + t3*x[k3] - y)**2
                                                    # Functional value
j.backward() # Backpropagation
nabla_j = x.grad # Functional gradient
```

However, this remark is not true when it comes to evaluating the second-order derivatives of j. In this case, it is also necessary to save the operations performed on $\mathbf{x}$ to obtain $\theta_1, \theta_2, \theta_3$, as they depend on the displacement field.

5.3.2 Computation of orthogonal projections

We now discuss the nearest neighbor procedure that is used to find the nearest triangle from a point y. Computing $J(\mathbf{u})$ requires to compute the orthogonal projection of each point of Γ onto $S_{\mathbf{u}}$, which means that the search procedure is run once per point in Γ. As a consequence, efficiency in this part of the program is critical for the global performance of the algorithm.

Though we did not investigate quantitatively the performance of orthogonal projections, we give a few details about the procedures involved in our code. A naive approach

would consist in evaluating the distance between each point in Γ and each triangle in $S_{\mathbf{u}}$, which leads to a complexity of $O(p \times n_{\text{tri}})$, where $p = |\Gamma|$ and n_{tri} is the number of triangles.

Except for the naive approach, we did not implement the procedures below ourselves. We used existing functions available in Python packages.

Signed distance field: a continuous approach We first considered an approach involving a signed distance field with respect to the deformed organ. Here, we only consider the case $S_0 = \partial\Omega_0$. Remember that, whenever the projection of a point y is unique, it can be expressed as a function of the signed distance field $d_{\Omega_{\mathbf{u}}}$ by the expression

$$p_{\partial\Omega_{\mathbf{u}}}(y) = y - d_{\Omega_{\mathbf{u}}}(y)\nabla d_{\Omega_{\mathbf{u}}}(y). \tag{5.9}$$

We implemented this method using the MshDist software (Dapogny and Frey, 2012), resulting in faster projections than the naive method, especially in three-dimensional cases. The procedure begins by evaluating the signed distance function $d_{\Omega_{\mathbf{u}}}$ at the vertices of a background tetrahedral mesh, so that the signed distance and its gradient can be interpolated a each point $y \in \Gamma$. Then the coordinates of the orthogonal projection are computed using (5.9). Finally, some interpolation work is required to find the triangle containing the projection point. Figure 5.3 shows a signed distance field computed on a background mesh in a bi-dimensional case.

An advantage of this approach is that the most expensive operation has to be executed once for all points in Γ. Though, the signed distance field has to be recomputed every time $\mathbf{u}$ changes. The signed distance approach comes with several other drawbacks. First, it is necessary to use a background mesh which should always contain the organ model while it is subject to deformations. If the deformed organ does not stay inside the background mesh, the procedure may fail. In addition, the signed distance field is approximated using a P1 finite element function. For the signed distance estimation to be accurate, the background mesh needs to be dense, which makes the computation costly. However, even with a dense background mesh, the signed distance gradient may be inaccurate, especially close to the skeleton of $\partial\Omega_{\mathbf{u}}$. In the same vein, the signed distance value may be inaccurate close to $\partial\Omega_{\mathbf{u}}$, making it impossible to solve the optimization problem with a tight tolerance. For those reasons, despite its apparent interest from a continuous point of view, an approach based on the signed distance is not adequate for our needs.

Spatial indexing structures While the signed distance approach is based on the discretization of a continuous process, the solution we chose is based on the polyhedral structure of the discretized organ and explicitly consists in finding the closest triangle from y. The procedure is based on spatial indexing structures. Such structures are used for instance in computer graphics for collision detection (James and Pai, 2004).

We use a procedure available in the Trimesh Python package[1], based on a R-tree (Guttman, 1984). The functioning of such a structure is well beyond the scope of this

[1] https://trimsh.org/

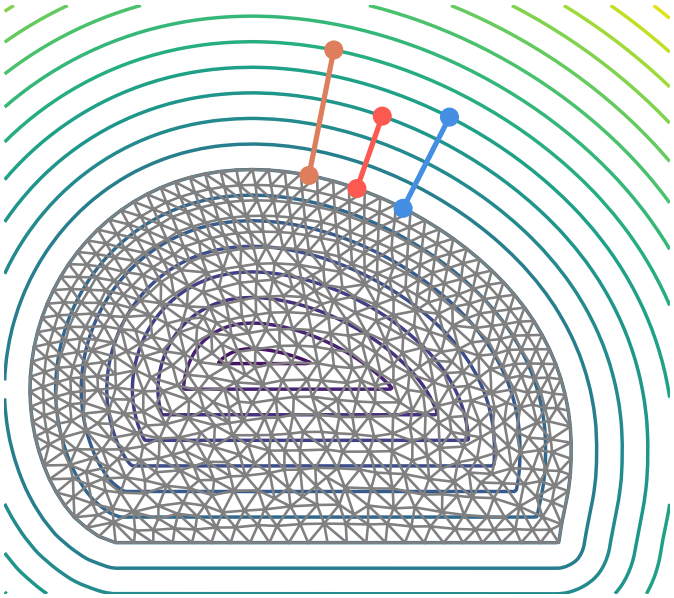


Figure 5.3: Signed distance field with respect to a two-dimensional mesh. Projection directions are orthogonal to the level curves of the signed distance field.

manuscript, but here is the main idea. In an R-tree, triangles are gathered in small groups represented by their bounding boxes. Those small boxes are gathered in bigger boxes, which are gathered in even bigger boxes, and so on. This results in a hierarchy of boxes, represented by a tree structure. When searching for the closest triangle to a point $y \in \Gamma$, the algorithm does not need to explore smaller boxes if they are contained in a bigger box that is already too far from y. This results in an average query complexity of $O(\log n_{\text{tri}})$. Figure 5.4 shows an example of such a box hierarchy on a surface mesh. The performance of an R-tree is determined by the insertion algorithm used to build the tree. More complex insertion algorithms are more costly but result in more efficient queries. A popular variant is the R*-tree (Beckmann et al., 1990), where the area and overlapping of boxes is minimized. Later, packing algorithms (Leutenegger et al., 1997; García R et al., 1998) have proven efficient to add several elements into the tree in one procedure.

The R-tree approach is particularly relevant in our application case. First, the procedure returns the triangle on which the projected point falls, along with barycentric coordinates. In addition, it is possible to select the triangles that are candidates to match the observed point cloud, when a signed distance field can only be computed with respect to the full shape.

Let us add a qualitative remark about the performance of the R-tree approach. In this manuscript, we use the algorithm from Trimesh as a black box but we are far from an efficient implementation. At each evaluation, a new R-tree is built and then one query per point in Γ is performed. As the mesh topology does not change along the deformation process, it is not necessary to build a new tree from zero before each evaluation of the

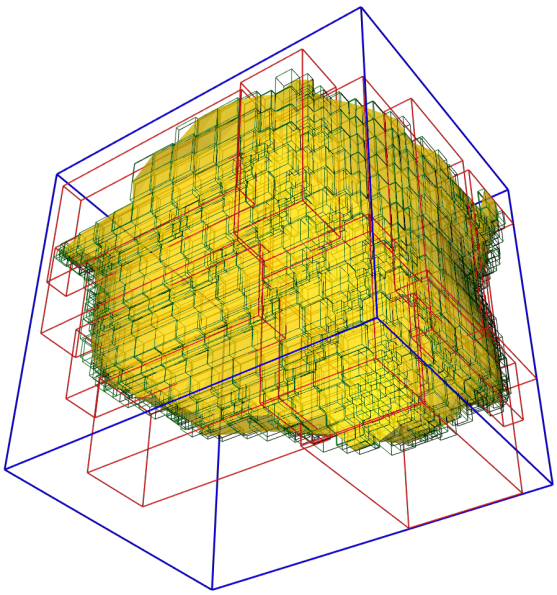


Figure 5.4: Bounding box hierarchy around a triangular surface mesh. Small boxes are the bounding boxes of single triangles, while the biggest box is the bounding box of the whole surface. Red boxes represent an intermediate level of boxes.

functional, only the coordinates of boxes and triangles need to be updated. Maintaining the same indexing structure along iterations and performing hierarchy changes only when necessary might improve performance.

Brute-force returns on the GPU Due to the real-time requirements, we are not expecting to deal with very dense meshes or point clouds. While spatial indexing structures are able to store millions of points, we only expect to compare at most a few thousands points with a few thousands triangles. In this case, evaluating the distances between every point and every triangle is still possible in a reasonable time, due to the highly parallel structure of graphic cards (or GPU for Graphics Processing Unit). In particular, this brute force approach is implemented in a function from the PyTorch3D library and might be considered for a GPU implementation of the registration procedure. Also, note that implementations of spatial indexing structures on the GPU exist in the literature (Prasad et al., 2015).

5.4 Newton methods for static elasticity problem

We now turn to the procedure that manages the elastic model using a finite element discretization. Namely, this procedure is responsible for solving the direct problem to transform a force distribution $\mathbf{b}$ into a displacement field $\mathbf{u}$, and to solve the adjoint problem to transform a gradient $\nabla J(\mathbf{u})$ in the space of displacements into an adjoint

state $\mathbf{p}$. When a linear model is used, the role of the finite element package boils down to solving linear systems involving the stiffness matrix. Even in the nonlinear case, the adjoint problem remains linear, and an adjoint states is computed by assembling and solving a linear system. For this reason, we focus here on the most challenging part, which is to solve the direct problem in the nonlinear case. This task is usually taken care of using a Newton method. Though, some precautions must be taken to ensure the convergence of the algorithm.

Invented in the 17th century, the Newton method (also known as the Newton-Raphson method) remains nowadays the go-to method to solve nonlinear variational problems such as static elasticity (Kelley, 1995, Chapter 5 and references therein). In a recent study, Morch et al. (2022) compare the performance of Newton-type methods of order 2 and higher on a few selected problems, and evaluate the effect of damping on the method robustness. The most computationally efficient method in their benchmark turns out to be the standard Newton method, even though other variants exhibit a higher convergence rate in theory. Other approaches, such as quasi-Newton methods (Gelin and Picart, 1988) show interesting performance in mechanical simulation (Liu et al., 2017; Yusa et al., 2021). However, we focus here on the Newton method and some of the remarks we formulate also apply to quasi-Newton methods.

In Section 2.2, we formulated the nonlinear elastic problem as the minimization problem (2.1). Here, we consider a discretized version of (2.1). After defining some notations, we describe the standard Newton algorithm and a variant adapted to optimization problems.

Remember that the discretized displacement field is represented by a vector $\mathbf{u} \in \mathbb{R}^{3n}$, where n is the number of vertices in the mesh, and forces are represented by $\mathbf{b} \in \mathbb{R}^{3n}$. The discrete optimization problem reads

$$\min_{\mathbf{u} \in \mathcal{U}} \quad W(\mathbf{u}) - \mathbf{b}^{\mathrm{T}}\mathbf{u}, \tag{5.10}$$

where we keep using $\mathcal{U}$ to denote the feasible set in the discrete case, and we denote the first-order optimality conditions by

$$\mathbf{F}(\mathbf{u}) - \mathbf{b} = 0 \quad \text{where} \quad \mathbf{F} = \nabla W. \tag{5.11}$$

5.4.1 Newton method for variational problems

The most common approach to solving nonlinear elastic problems is to solve the optimality condition (5.11) as a variational problem. In this context, the Newton method takes the form of a fixed-point algorithm, where a linear approximation of (5.11) is solved at each iteration. The first-order development of the residual reads

$$\mathbf{F}(\mathbf{u} + \mathbf{w}) = \mathbf{F}(\mathbf{u}) + \mathbf{K}(\mathbf{u})\mathbf{w} + \mathrm{o}(\|\mathbf{u}\|),$$

where $\mathbf{K}(\mathbf{u}) = \mathbf{F}'(\mathbf{u})$ denotes the residual Jacobian at $\mathbf{u}$ (also called the stiffness matrix). The Newton step $\mathbf{w}$ is the solution of the tangent system

$$\mathbf{K}(\mathbf{u})\mathbf{w} = \mathbf{b} - \mathbf{F}(\mathbf{u}),$$

Algorithm 2: Standard Newton method for nonlinear variational systems.

Data: Initial iterate $\mathbf{u}_0$, tolerance ϵ
for *k = 0, 1, 2, ...* **do**
  Evaluate the residual $\mathbf{r}_k = \mathbf{b} - \mathbf{F}(\mathbf{u}_k)$
  if $\|\mathbf{r}_k\| \leqslant \epsilon$ **then**
    return $\mathbf{u}_k$
  end
  Compute a step $\mathbf{w}_k$ by solving $\mathbf{K}(\mathbf{u}_k)\mathbf{w}_k = \mathbf{r}_k$
  $\mathbf{u}_{k+1} \leftarrow \mathbf{u}_k + \mathbf{w}_k$
end

and the next iterate is defined as $\mathbf{u}_+ = \mathbf{u} + \mathbf{w}$.

The main source of cost in the method is the inversion of a linear system involving $\mathbf{K}(\mathbf{u})$ at each iteration. This operation can either be taken care of by a direct solver, for instance involving a LU factorization (Nocedal and Wright, 2006, Appendix A), or by an iterative solver such as GMRES (Saad and Schultz, 1986). Though the principles of such solvers are out of the scope of this manuscript, we briefly describe the advantages of both families of solvers in Newton methods. Iterative solvers are more relevant for large-scale systems, as they do not require to assemble the system matrix. In addition, they can solve systems inexactly to gain time. For instance, the linear solver may be used with a large tolerance in the first Newton iterations and then the tolerance becomes tighter along iterations to enforce superlinear convergence of the Newton method (Nocedal and Wright, 2006, Chapter 11, Theorem 11.3). On the other hand, iterative solvers are sensitive to the system matrix conditioning and often need preconditioning to perform efficiently. Direct solvers, which rely on a factorization of the system matrix, are less sensitive to bad conditioning and can achieve higher accuracy in system solutions. Though matrix assembly and factorization represent a significant overhead, modern parallel implementations of direct solvers are very efficient. For these reasons, direct solvers are often preferred in our application domain, where linear systems keep reasonable sizes in order to keep up with the real-time requirement. The interested reader may refer to the book by Saad (2003) and the review by Davis et al. (2016) for more information about iterative and direct solvers, respectively.

Algorithm 2 gives a summary of a standard Newton method. Newton methods are popular due to their fast convergence rate, provided that $\mathbf{F}$ is regular enough and $\mathbf{K}$ has good properties in a vicinity of the solution. The following theorem (Nocedal and Wright, 2006, Chapter 11, Theorem 11.2) states the theoretical convergence rate of the method.

Theorem 5.1. *Suppose that* $\mathbf{F}$ *is continuously differentiable, let* $\mathbf{u}^\star$ *a solution of problem (5.11) such that* $\mathbf{K}(\mathbf{u}^\star)$ *is nonsingular, and let* $(\mathbf{u}_k)$ *the sequence of iterates generated by Algorithm 2. Then when* $\mathbf{u}_k$ *is close enough to* $\mathbf{u}^\star$*, the Newton method converges at a superlinear rate, i.e.*

$$\mathbf{u}_{k+1} - \mathbf{u}^\star = \mathrm{o}\left(\|\mathbf{u}_k - \mathbf{u}^\star\|\right).$$

In addition, if $\mathbf{F}$ *is Lipschitz continuously differentiable, then convergence is quadratic, i.e.*

$$\mathbf{u}_{k+1} - \mathbf{u}^\star = \mathrm{O}\left(\|\mathbf{u}_k - \mathbf{u}^\star\|^2\right).$$

Theorem 5.1 only states a local convergence result. In other words, the initial iterate should be close enough to the solution for the method to converge toward this solution. As a consequence, the Newton method suffers from a lack of robustness when iterations start far away from a solution.

In addition, monitoring convergence along iterations is sometimes difficult. The stopping criterion in Algorithm 2 is the residual norm $\|\mathbf{r}_k\|$. Though the residual norm decreases dramatically when getting close to the solution, it is not supposed to decrease in general and can even take very large values before convergence. As a consequence it is difficult to evaluate the progress of the method during execution.

In optimization algorithms based on the Newton method, some features were added to better enforce and monitor convergence, at the expense of efficiency.

5.4.2 Newton method for optimization problems

We now consider the elasticity problem under the form (5.10). Treating a nonlinear elastic problem as a minimization problem is a way to obtain useful insight, especially by looking at the properties of the objective function. In particular, the evolution of the objective value along iterations is convenient to monitor convergence in the first iterations. It is complementary to the residual norm, which is more meaningful in the last iterations. By requiring the objective function to decrease along iterations, optimization algorithms enforce global convergence toward a local minimizer of the problem. In this paragraph, we describe a trust-region Newton method that solves optimization problems. Trust-region methods have already been considered in the context of nonlinear mechanics (Gross and Krause, 2009; Youett et al., 2019). While we mainly refer to Nocedal and Wright (2006), the interested reader might find more specific information about trust-region methods in the book by Conn et al. (2000).

Though it is based on the standard Newton method, the trust-region Newton method uses a slightly different formalism. During an iteration, the objective function $\mathbf{u} \mapsto W(\mathbf{u}) - \mathbf{b}^{\mathrm{T}}\mathbf{u}$ is approximated around a given iterate $\mathbf{u}_k$ by its second-order Taylor expansion. The local quadratic model reads

$$m_k(\mathbf{w}) = \underbrace{\left(W(\mathbf{u}_k) - \mathbf{b}^{\mathrm{T}}\mathbf{u}_k\right)}_{\text{objective value}} + \underbrace{\left(\mathbf{F}(\mathbf{u}_k) - \mathbf{b}\right)}_{\text{objective gradient}}{}^{\mathrm{T}}\mathbf{w} + \tfrac{1}{2}\mathbf{w}^{\mathrm{T}}\underbrace{\mathbf{K}(\mathbf{u}_k)}_{\text{objective Hessian}}\mathbf{w}. \tag{5.12}$$

Now, the Jacobian $\mathbf{K}$ plays the role of the Hessian of W, and its properties depend on the convexity of W. In particular, it is positive definite whenever W is strictly convex. Here we assume that $\mathbf{K}(\mathbf{u})$ is nonsingular everywhere.

A standard Newton step, starting from $\mathbf{u}_k$, lands on the critical point of m_k. However, this critical point is a minimizer of m_k only when $\mathbf{K}(\mathbf{u}_k)$ is positive definite. Otherwise, it may be a saddle point or a maximum of m_k, while a minimum does not necessarily exist.

For this reason, the Newton step is reformulated into an optimization subproblem which aims to minimize the local model on a ball centered at $\mathbf{u}_k$. The ball, called the *trust-region*, defines a zone where the quadratic model m_k provides a suitable approximation of the objective function, and enforces the existence of a solution to the subproblem. The *trust-region subproblem* reads

$$\min_{\mathbf{w}} \quad m_k(\mathbf{w}) \quad \text{subject to} \quad \|\mathbf{w}\| < \Delta_k \tag{5.13}$$

where $\Delta_k > 0$ is the trust-region radius. Provided that m_k is strictly convex and Δ_k is large enough, solving (5.13) boils down to performing a standard Newton step. This situation arises in particular when iterates approach a solution of the optimization problem, and thus the trust-region Newton method benefits the convergence rates stated in Theorem 5.1.

In practice, the trust-region problem is approximately solved using a truncated Conjugate Gradient method (Nocedal and Wright, 2006, Algorithm 7.2), which stops upon trust-region bound violation or generation of a direction of nonpositive curvature. In a recent paper, Dahito and Orban (2019) compare the performance of the Conjugate Gradient method and the Conjugate Residual method for trust-region problems.

Algorithm 3: Trust-region mechanism (Nocedal and Wright, 2006, Alg. 4.1).

Data: Initial radius Δ_0, reject tolerance $\eta \in [0, 1/4)$
for *k = 0, 1, 2, ...* **do**
  Compute a step $\mathbf{w}$ by solving (5.13)
  Evaluate ρ_k
  `// Adjust trust-region radius`
  if $\rho_k < 1/4$ **then**
    Reduce trust-region: $\Delta_{k+1} < \Delta_k$
  else if $\rho_k < 1/4$ *and* $\|\mathbf{w}\| = \Delta_k$ **then**
    Expand trust-region: $\Delta_{k+1} > \Delta_k$
  else
    $\Delta_{k+1} = \Delta_k$
  end
  `// Accept or reject step`
  if $\rho_k > \eta$ **then**
    Accept step: $\mathbf{u}_{k+1} = \mathbf{u}_k + \mathbf{w}$
  else
    Reject step: $\mathbf{u}_{k+1} = \mathbf{u}_k$
  end
end

The trust-region mechanism is summarized in Algorithm 3. To accept or reject a step $\mathbf{w}$ as well as to adjust the trust-region radius, the mechanism relies on the coefficient

$$\rho_k = \frac{f(\mathbf{u}_k) - f(\mathbf{u}_k + \mathbf{w})}{m_k(0) - m_k(\mathbf{w})},$$

where $f(\mathbf{u}) = W(\mathbf{u}) - \mathbf{b}^{\mathrm{T}}\mathbf{u}$ denotes the objective value. The coefficient ρ_k compares the model reduction and the actual reduction allowed by the step $\mathbf{w}$. In particular, when a step $\mathbf{w}$ does not decrease the objective function as much as predicted by the model, it is rejected and the trust-region radius is decreased. Though rejecting steps represent an additional computational cost, it ensures the convergence of the trust-region Newton method, as stated in the following theorem (Nocedal and Wright, 2006, Chapter 4, Theorem 4.1).

Theorem 5.2. *Assume $\eta > 0$ in Algorithm 3. Assume also that $\|\mathbf{K}\|$ is uniformly bounded, that the objective function $W(\mathbf{u}) - \mathbf{b}^{\mathrm{T}}\mathbf{u}$ is bounded below and that $\mathbf{F}$ is Lipschitz continuous. Let $(\mathbf{u}_k)$ be the iterates generated by the trust-region Newton method. Then*

$$\lim_{k\to\infty} \mathbf{F}(\mathbf{u}_k) - \mathbf{b} = 0.$$

In Figure 5.5, we applied random nodal forces to the 2D beam from Figure 2.3, resulting in a very irregular displacement field. We plotted the convergence statistics of the standard and trust-region Newton method, in the case of a Saint Venant-Kirchhoff model and in the case of a Neo-Hookean model. Due to the good properties of the Saint Venant-Kirchhoff energy functional, the standard Newton method behaves well and converges much faster than the trust-region method. On the other hand, the trust-region method is the only one to converge in the more difficult case of the Neo-Hookean model. While the standard Newton methods fails as soon as an iterates lands onto a non-feasible point, the trust-region mechanism allows the trust-region method to take shorter steps when necessary.

The very aggressive standard Newton method and the more careful trust-region Newton method are only two examples among a full spectrum of variants with various levels of convergence enforcement. Strategies we did not mention include line search algorithms (Nocedal and Wright, 2006, Chapter 3), which are relevant for convex problems and quasi-Newton methods. The energy-minimization nature of static elasticity problems gives access to a large choice of strategies to enforce convergence. Compromises between all those strategies should be found to combine efficiency requirements with the robustness necessary in sensitive applications such as augmented surgery.

5.5 Optimization procedure

In this section, we discuss the choice of an appropriate optimization algorithm to solve the optimal control problem in the unconstrained case and in the pointwise-constrained case. Note that in this manuscript we do not solve the state-constrained problem (2.14) numerically. The chosen algorithms should satisfy two main requirements. First, second-order information is not available for the objective function Φ, thus Newton methods are excluded from possible choices. For this reason, we rather turn toward first-order methods and quasi-Newton algorithms. Second, we are dealing with optimization problems that can be large, and thus storing full matrices is often impossible. For this reason,

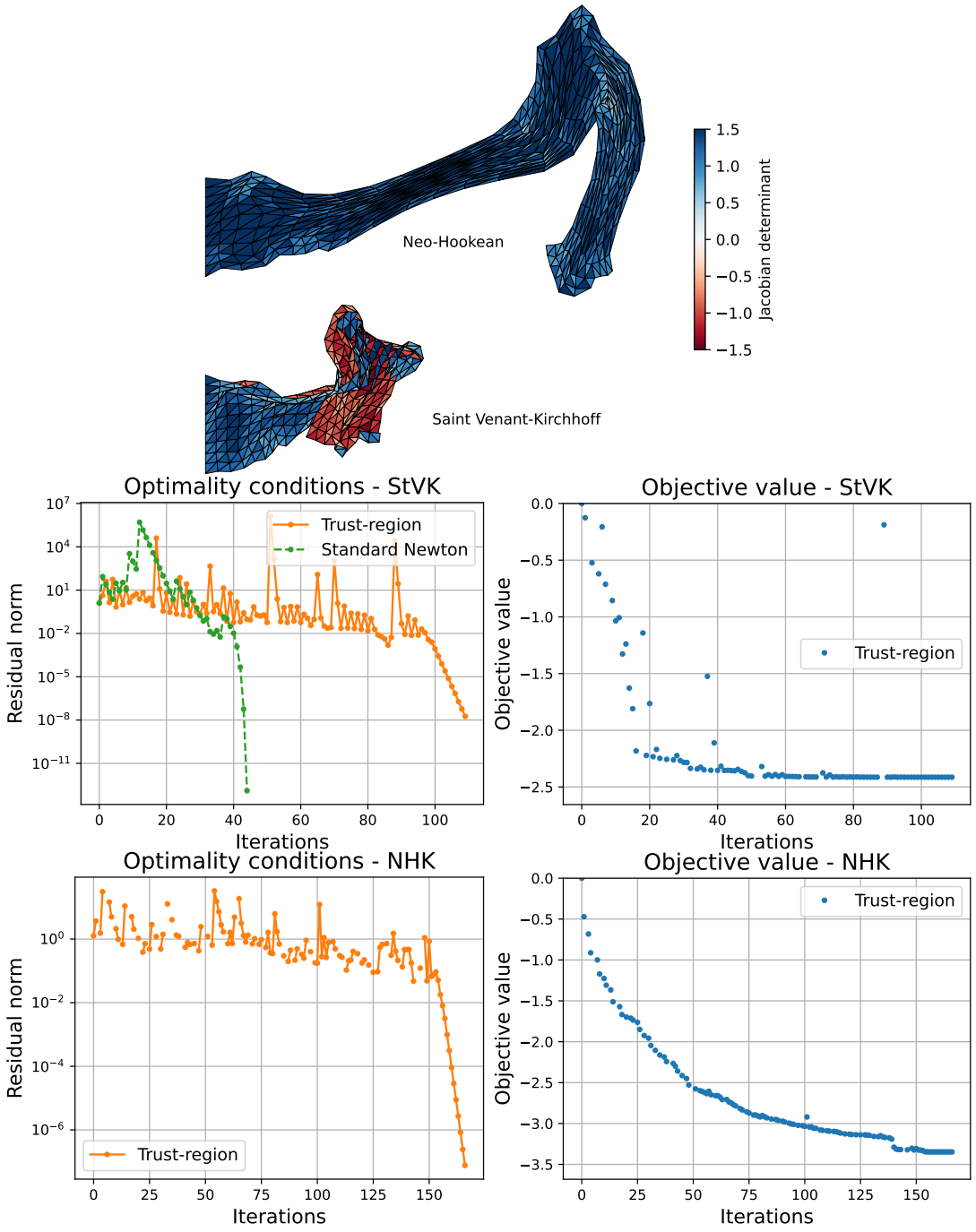


Figure 5.5: Resulting deformation and convergence statistics of the standard Newton method and the trust-region method (including rejected steps for the trust-region method). Missing points in the Neo-Hookean case correspond to infeasible points where the objective function evaluates to $+\infty$.

chosen optimization software should support sparse matrices and limited-memory quasi-Newton Hessian approximations.

In this section, we use the notation $\mathbf{g} = \nabla\Phi(\mathbf{b})$ *to denote the objective gradient. It has nothing to do with the surface force distribution.*

5.5.1 Unconstrained problem

We first illustrate the adjoint method on a registration problem whithout L^∞ constraint. Dropping the L^∞ constraint makes sense in a finite dimensional context, as this constraint is not indispensable for solutions to exist. In addition, displacement regularity can be enforced by cheaper means such as penalty terms, which saves the cost of constraint management. As far as the optimization solver is concerned, the first-order optimality conditions for problem (5.4) in the unconstrained case read

$$\nabla\Phi(\mathbf{b}) = 0.$$

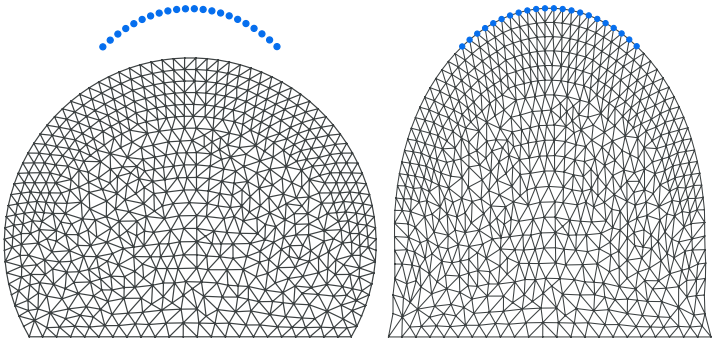
First-order methods

In the early FreeFem++ implementation, we implemented two simple first-order methods to validate the approach. We first used a standard gradient descent with backtracking linesearch (Armijo, 1966). Then we implemented an accelerated gradient descent (Nesterov, 1983), where some inertia is given to the sequence of iterates to avoid zigzagging phenomena (see also Su et al., 2016).

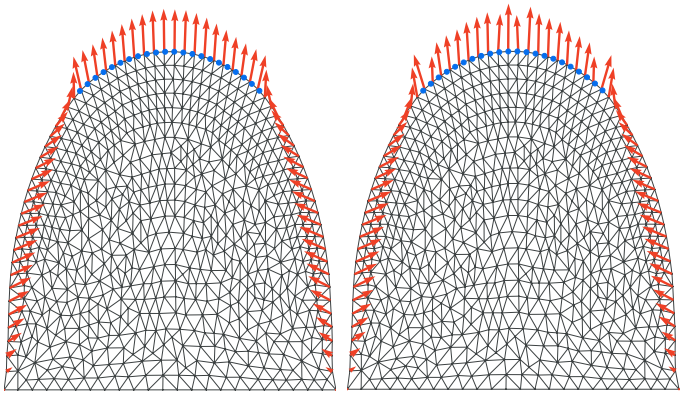
In Figure 5.6, we show an example of registration involving a two-dimensional mesh. A point cloud is generated from a deformed configuration and used to perform the registration. As expected, the Nesterov accelerated gradient descent decreases the objective function faster than the standard gradient descent, especially in the first iterations (Figure 5.6c). However, note (Figure 5.6b) that the surface force distribution reconstructed by the Nesterov method is more irregular than the one reconstructed by the gradient descent. Indeed, accelerated and higher-order methods converge faster because it is easier for them to reach irregular surface forces distributions. On the other hand, the gradient descent method is very sensitive to the regularizing effet of the adjoint method, but it is also the reason why its convergence is slow.

An off-the-shelf quasi-Newton solver

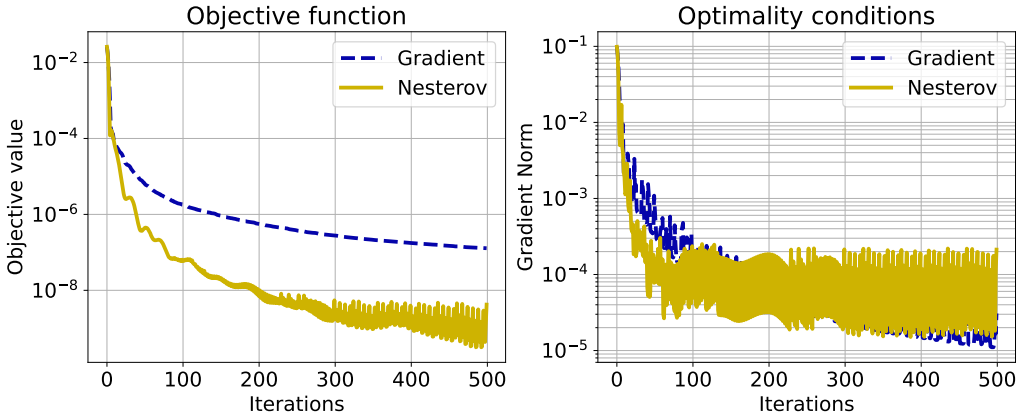
In the Python implementation, we use a standard quasi-Newton solver available in the Scipy package, namely the L-BFGS-B algorithm (Byrd et al., 1995). This limited-memory BFGS method is normally tailored for bound-constrained problems, but remains relevant for our case. We create a toy problem by applying a surface loading onto a truncated sphere mesh of radius 1 (10,385 vertices) with a Dirichlet boundary condition applied on its base. Then, we sample the deformed surface to create a point cloud of 200 points. We perform the registration on a coarser mesh (907 vertices), resulting in a direct problem of size 2,721. More details about the toy problem are given in Section 6.1.



(a) Initial (left) and deformed (right) configurations with generated point cloud.



(b) Resulting deformations for the standard (left) and accelerated (right) gradient descents.



(c) Convergence statistics.

Figure 5.6: Results for the 2D unconstrained problem after 500 iterations of the standard and accelerated gradient methods.

For both data generation and reconstruction, a linear elastic model is used, with $E = 1$ and $\nu = 0.49$. We use the adjoint method to estimate nodal forces on boundary nodes that are not subject to a Dirichlet boundary condition, resulting in an optimization problem with $418 \times 3 = 1254$ unknowns. We set the penalty term to zero, and we use a limited-memory BFGS method available in the Scipy package, with at most 200 stored updates. Figure 5.7a illustrates the data generation process and Figure 5.7b shows the reconstructed deformation along with the nodal forces on the coarser mesh. The reconstructed deformation is correctly fitting the pointcloud. Figure 5.7c shows the convergence statistics. After 184 iterations (4.5 s on our configuration), the objective gradient norm has decreased by a factor 10^5. Meanwhile, the objective value, which is supposed to converge toward zero, has also decreased by 10^5, which means that the average distance $d(y, S_{\mathbf{u}})$ has decreased by a factor $10^{2.5} \approx 316$.

5.5.2 Problem with pointwise constraint

In the discrete context, the L^∞ constraint takes the form of a pointwise (or should we say nodewise) constraint on the components b_i of the control. Though the physical meaning is not exactly the same as the continuous formulation, the spirit is similar, namely avoiding too strong nodal forces. If we denote by $I_{\mathbf{b}}$ the set of nodes where b_i is nonzero, the constraint reads

$$\forall i \in I_{\mathbf{b}} \qquad \tfrac{1}{2}\|b_i\|^2 \leqslant \tfrac{1}{2}M^2, \tag{5.14}$$

and the first-order optimality conditions read

$$\forall i \in I_{\mathbf{b}} \qquad \begin{cases} \partial_i \Phi(\mathbf{b}) + \mu_i b_i & = 0 \\ \mu_i & \geqslant 0 \\ \tfrac{1}{2}\|b_i\|^2 - \tfrac{1}{2}M^2 & \leqslant 0 \\ \tfrac{1}{2}\mu_i \left(\|b_i\|^2 - M^2\right) & = 0, \end{cases} \tag{5.15}$$

where $\mu_i \in \mathbb{R}_+$ is the Lagrange multiplier associated with the i-th pointwise constraint.

Projected gradient and fixed-point iteration

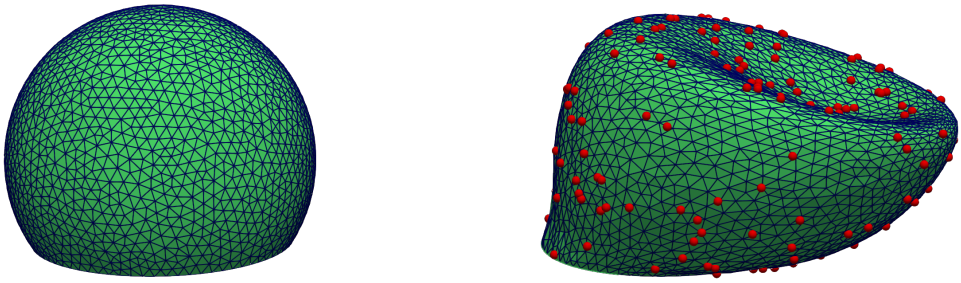
Like previously, we preferred to use simple first-order algorithms to solve the pointwise constrained problem in the FreeFem++ implementation. As the pointwise constraint applies separately on each component b_i, it is very easy to perform projections onto the feasible set

$$\mathcal{B} = \left\{ (b_i)_{i \in I_{\mathbf{b}}} \mid \forall i \in I_{\mathbf{b}} \quad \|b_i\| \leqslant M \right\}$$

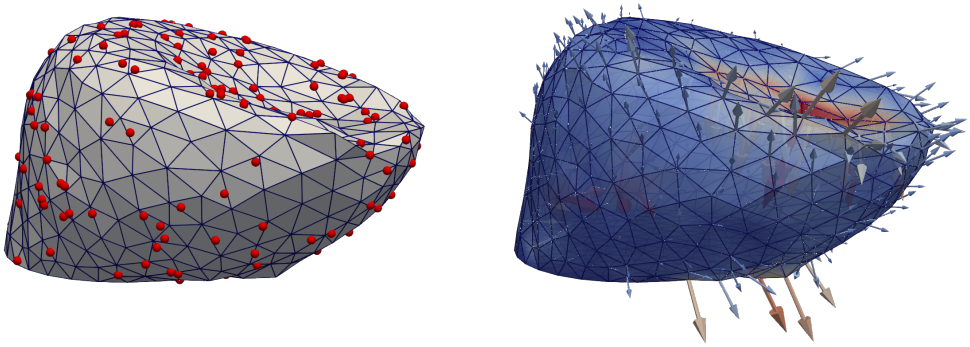
by projecting each b_i onto a ball of radius M. Thus, the projection operation reads

$$\forall i \in I_{\mathbf{b}} \quad \mathrm{Proj}(\mathbf{b})_i = \begin{cases} b_i & \text{if } \|b_i\| \leqslant M \\ M \frac{b_i}{\|b_i\|} & \text{if } \|b_i\| > M. \end{cases}$$

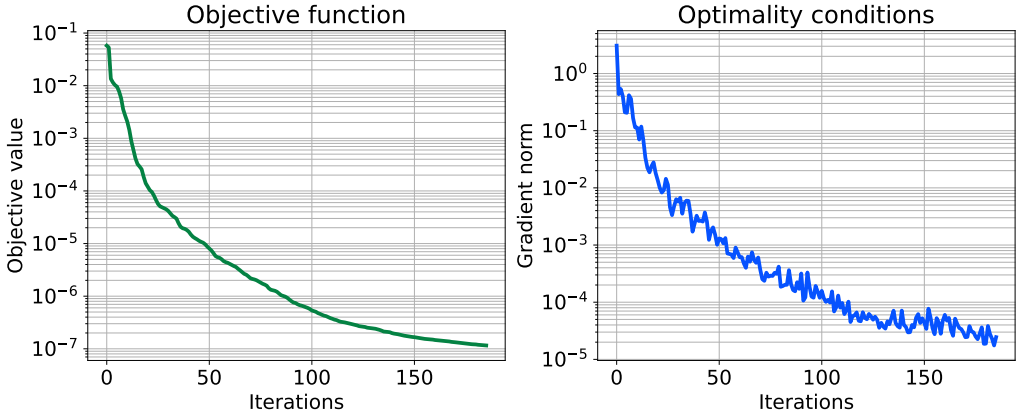
We exploited those cheap projections in a projected gradient method (Bertsekas, 1976). An overview of the projected gradient iteration is given in Algorithm 4. At



(a) Initial sphere mesh and deformed configuration with generated point cloud.



(b) Reconstructed deformation and nodal forces that generated the optimal displacement.



(c) Convergence statistics.

Figure 5.7: Reconstruction results for the 3D unconstrained problem.

each iteration, the step length is selected using a projected backtracking linesearch, and optimality is measured using the projected gradient norm $\|\operatorname{Proj}(\mathbf{b} - \mathbf{g}) - \mathbf{b}\|$, where $\mathbf{g} = \nabla\Phi(\mathbf{b})$.

Algorithm 4: A projected gradient descent

Data: Initial guess $\mathbf{b}_0$, tolerance ε
for $k = 0, 1, 2, \ldots$ **do**
  Evaluate objective gradient: $\mathbf{g}_k = \nabla\Phi(\mathbf{b}_k)$
  if $\|\operatorname{Proj}(\mathbf{b}_k - \mathbf{g}_k) - \mathbf{b}_k\| < \varepsilon$ **then**
    return $\mathbf{b}_k$
  end
  Select a step size β_k by doing a linesearch
  Compute the next iterate $\mathbf{b}_{k+1} = \operatorname{Proj}(\mathbf{b}_k - \beta_k \mathbf{g}_k)$
end

We also implemented a variant of the projected gradient descent, inspired from fixed-point methods. When they converge, fixed-point methods based on optimality conditions may yield fast and robust convergence. In our variant, we replace the orthogonal projection by a so-called *fixed-point step*, derived from the optimality conditions. When $R(\mathbf{b}) = 0$, the first line in (5.15) can be reformulated

$$\forall i \in I_{\mathbf{b}} \qquad p_i + \mu_i b_i = 0.$$

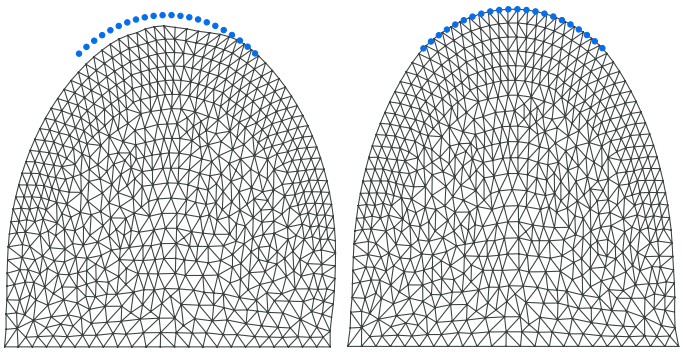
As a consequence, at optimality, b_i should be colinear to p_i if $\mu_i > 0$. To enforce this condition, the fixed-point step reads

$$\operatorname{FPStep}(\mathbf{b})_i = \begin{cases} b_i & \text{if } \|b_i\| \leqslant M \\ -M \frac{p_i}{\|p_i\|} & \text{if } \|b_i\| > M. \end{cases}$$

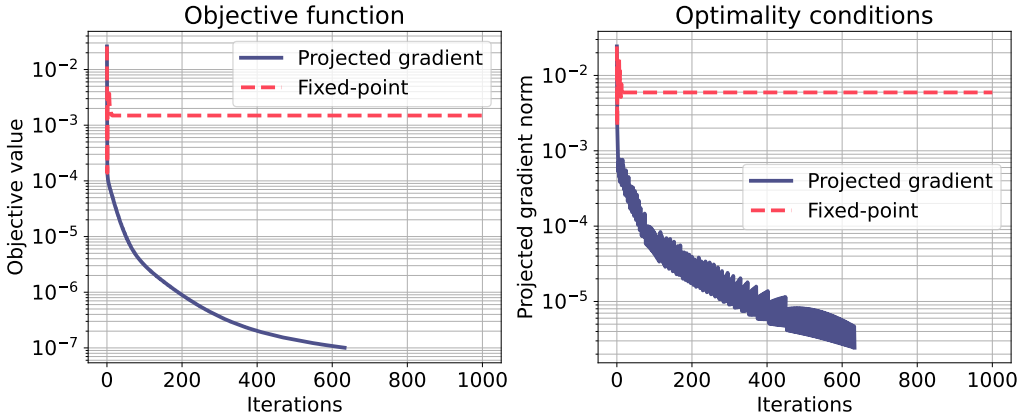
The FPStep operator is used in the last line of Algorithm 4 instead of the orthogonal projection. Note that the fixed-point step has a stronger effect than the projected gradient step. If we only consider components that are affected by the step, the fixed point step is equivalent to applying a projected gradient step with $\beta_k = +\infty$ to these components. This is the reason why the fixed-point method has good convergence properties in good conditions but might also show some instability.

Figure 5.8 shows the convergence results on a simple two-dimensional problem. In the case of the projected gradient, the projected gradient norm decreases by a factor 10^4 in about 600 iterations, and a visually correct matching is achieved. The fixed-point method, however, seems to converge in a few iterations, but not toward a solution of the optimization problem. In particular, the resulting deformed mesh does not match the point cloud. Other tests with the fixed-point method exhibit behaviors where the mesh oscillates between two deformations, none of which is a solution of the optimization problem.

After several tentatives, we decided to drop the fixed-point approach due to its instability, and to keep using standard optimization methods.



(a) Resulting deformations for the fixed-point (left) and projected gradient (right) algorithms.



(b) Convergence statistics.

Figure 5.8: Results for the 2D problem after 1000 iterations of the projected gradient method and the fixed-point method. Constraint violation is not plotted, as projected method only generate feasible iterates.

A primal-dual algorithm with BFGS approximation

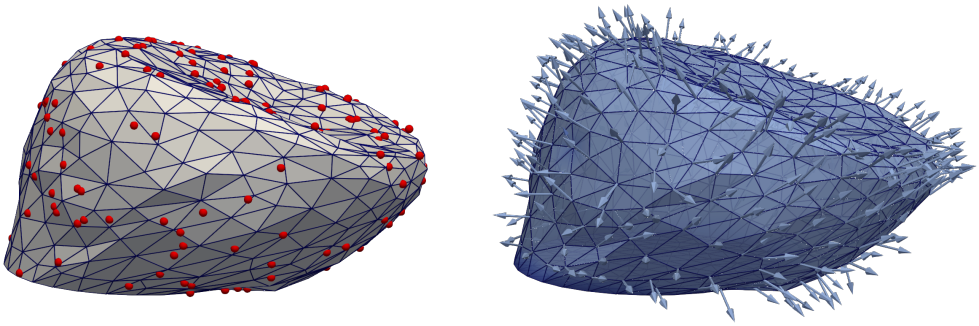
While using projected directions is very convenient in the context of gradient descents, it is way more difficult when using methods involving a change of metrics, such as Newton and quasi-Newton methods (Bertsekas, 1982). As a consequence, standard projected-directions algorithms are limited to problems with bound constraints (Lin and Moré, 1999; Byrd et al., 1995), while our L^∞ constraint is nonlinear. For this reason, we turned to primal-dual methods in the Python implementation. Though the functioning of such methods is out of the scope of this manuscript, let us mention that they manage constraints using Lagrange multipliers (Nocedal and Wright, 2006, Chapters 17–19). Primal-dual methods include interior-points methods such as Ipopt (Wächter and Biegler, 2006), sequential quadratic programming algorithms like SLSQP (Kraft, 1988), or augmented Lagrangian methods (Hestenes, 1969).

We encountered difficulties to find an adequate optimization method, as most primal-dual programs are Newton methods that require the objective Hessian, or only provide full BFGS approximations, which is not relevant for large problems. Though, we still present an example involving the same sphere problem as before, where we add constraint (5.14) with $M = 10^{-2}$ (The maximal nodal force in the solution to the unconstrained problem was $\|\mathbf{b}\|_\infty = 3 \cdot 10^{-2}$). We solve the optimization problem using the sequential quadratic programming algorithm SLSQP by Kraft (1988), available in the Scipy library.

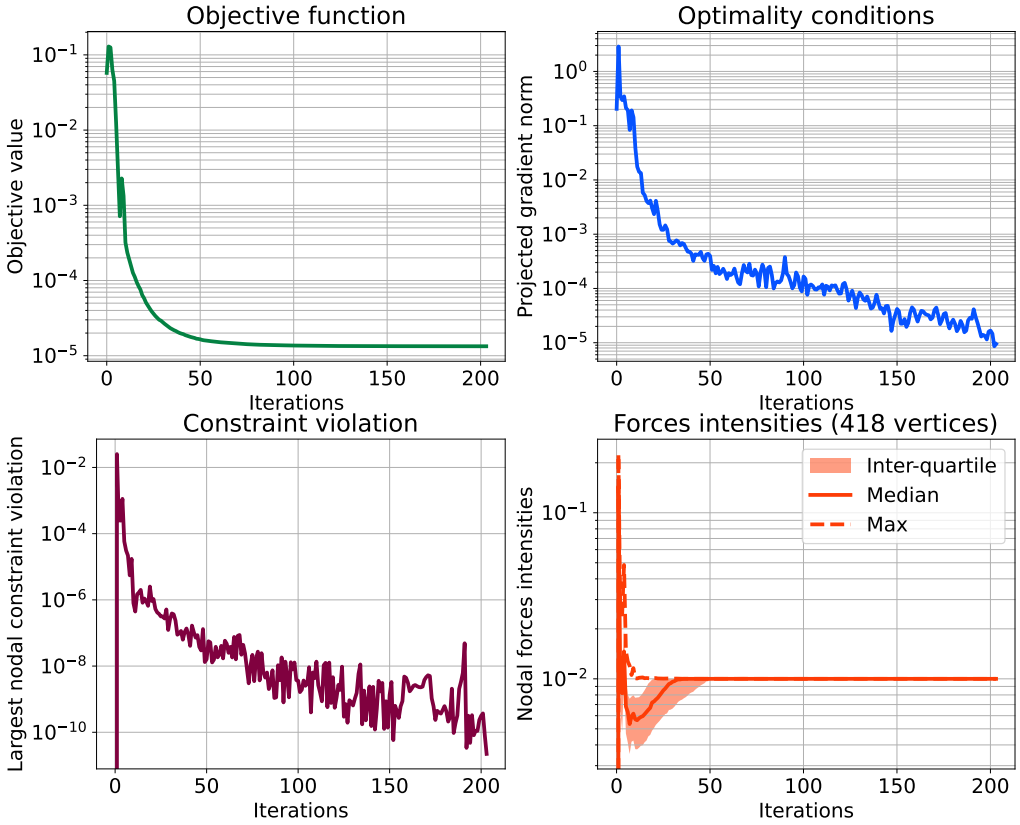
Figure 5.9a shows the deformed sphere after the constrained reconstruction, with the same color scale as in Figure 5.7b for nodal forces. Note that nodal forces do not exceed the limit M in intensity. The convergence statistics are plotted in Figure 5.9b. As the internal variables of the optimization solver are not available from the Python interface, we used the projected gradient norm to measure optimality conditions. Note that, due to the constraint, the objective function does not decrease toward zero as previously, but converges toward a positive value. In addition, the forces intensities graph, along with the plotted reconstruction, suggest that every component of the nodal force distribution tends to reach the maximal allowed intensity M. Note that, as soon as all constraints are active (iteration 50), the projected gradient norm seems to decrease with more difficulty.

These examples shows the feasibility of a L^∞ constraint on the control. However, the L^∞ constraint is a nonlinear constraint that requires a primal-dual method to solve the optimization problem. Such methods are more relevant when second-order information about the objective function is available and are difficult to use with quasi-Newton approximations. On the other hand, assembling a Hessian matrix for the adjoint method is out of question due to the composite nature of the objective function Φ. For this reason, we do not investigate algorithms to deal with the constrained problem any further.

Remark 5.4. The best choice to manage the L^∞ constraint is probably the interior-points strategy, as it only generates that satisfy the constraint. Solving the elasticity problem with large forces using the Newton method requires many iterations, and in addition the optimization solver might converge toward a minimizer devoid of physical meaning. The interior-points strategy circumvents this possibility by preventing the magnitude of nodal forces to exceed M.



(a) Reconstructed deformation and nodal forces. The color scale is the same as in Figure 5.7b.



(b) Convergence statistics for the quasi-Newton method.

Figure 5.9: Reconstruction results for the 3D problem with pointwise constraint.

Chapter 6

Numerical results

We now present a few applications of our approach in the context of augmented surgery. As a patient's life is at stake during a surgical operation, an augmented reality system for operation rooms is expected to be robust and accurate. In addition, such a method should be able to update the augmented view in real-time. With the following numerical tests, we try to evaluate the performances of our approach.

6.1 Preliminary investigations featuring the toy problems

Before we consider cases from the application domain, we present a couple of examples involving the toy problems used to illustrate the optimization methods in Section 5.5.

2D toy problem

In the case of the FreeFem++ implementation, the test case involves a two-dimensional truncated circular mesh of radius 1, where the distance from the disc center to the flat face is 1/2. The flat face of the mesh is subject to a homogeneous Dirichlet condition, while forces are applied on the circular boundary to generate a displacement. A point cloud with 22 points is created using a mesh with 690 nodes. An elastic deformation is applied to the mesh and a point cloud is extracted from the deformed boundary. For data generation as well as for reconstruction, a linear elastic model is used with $E = 1$ and $\nu = 0.49$.

3D toy problem

The toy problem for the Python implementation is a three-dimensional version of the previous one. We create a mesh with the shape of a truncated sphere of radius 1, where the distance from the sphere center to the truncating plane is 1/2. The flat surface of the mesh is subject to a homogeneous Dirichlet condition, while forces are applied on the round surface to generate a displacement. After an elastic deformation is applied to the mesh, the round surface is sampled to create point clouds with different sizes. The registration procedure involves a similar mesh. The flat surface is still subject to

a homogeneous Dirichlet condition, while the round surface is deformed to match the point cloud. In addition, nodal forces are allowed on every node which is not subject to the Dirichlet condition (i.e. every node from the round surface).

We generate data using a mesh with 10,385 nodes and 6,702 faces. Linear elasticity is used with $E = 1$ and $\nu = 0.49$. Unless stated otherwise, the same model and parameters are used in the examples below. Figure 5.7a shows the initial mesh and the generated deformation. The mesh used for registration is generally coarser.

6.1.1 No accuracy guarantee, even in an ideal case

Let us put it straight from the start: our registration method does not come with any guarantee in terms of displacement accuracy. Compared to other methods, an increased displacement accuracy might result from a more rigorous physical modeling, but no theoretical result confirms that.

To illustrate this disclaimer, we evaluate the displacement accuracy of the procedure in an ideal case. First, the point cloud is very dense (10,000 points), contains no noise and therefore provides a good representation of the deformed surface. In addition, the same mesh is used for data creation and for reconstruction, which evacuates possible discrepancies between two meshes representing the same shape.

The evolution of the objective function is plotted in Figure 6.1, along with the error statistics in terms of displacement field and force distribution. The objective function keeps decreasing during the execution, and evaluates to $2 \cdot 10^{-7}$, meaning that the quadratic mean of the distance between data points and the mesh approximately amounts to $6 \cdot 10^{-4}$. Though, the displacement error seems to remain stable after 50 iterations. The displacement error settles between 0.006 and 0.3 for half of the points, and the maximal error is 0.6. These figures are to be compared to the sphere radius, which is 1. The lower part of Figure 6.1 shows the evolution of the magnitude and error distribution among nodal forces. These results give us a few insights about the registration procedure. First, the reconstructed nodal forces do not exceed the maximal intensity of nodal forces used to create data (see also Figure 6.2a). As the objective function only measures the discrepancy between the point cloud and the reconstructed surface, the algorithm tends to choose among the simplest deformations that make them match, regardless of the original deformation. Here, a simple deformation is one that requires low forces intensities. The (absolute) error on nodal forces seems to increase at the same rate as forces intensities along iterations, and both distributions remain in the same order of magnitude. As a consequence, we understand that, in general, it might be difficult to give a quantitative meaning to the computed distribution, especially when it contains a large number of degrees of freedom.

In Figure 6.2, the true and reconstructed deformations are compared. Green shapes represent the true deformation, while yellow shapes represent the reconstructed deformation. The true and reconstructed nodal forces distributions are plotted in Figure 6.2a. Despite the large error between individual nodal forces, the same accumulation of forces on top of the mesh appears in both distributions. In Figure 6.2b, we superposed the original and reconstructed surfaces. Though the surfaces match well with each other,

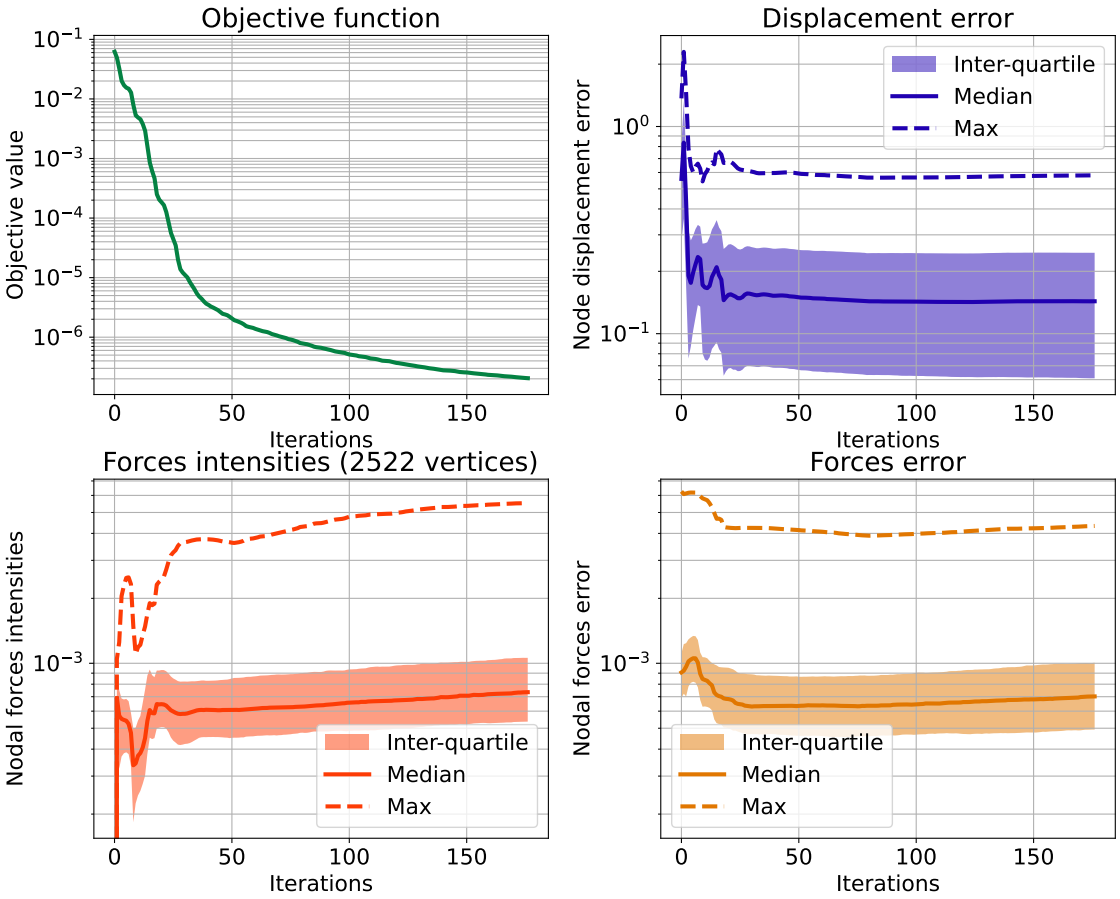


Figure 6.1: Top: objective value and displacement error at mesh nodes. Bottom: Intensity and error statistics for nodal forces.

the individual position of each node was not retrieved by the procedure. In Figure 6.2c, we applied the same pattern to the true and reconstructed surfaces to highlight the difference of position between both surfaces. Though both surfaces are consistent with the point cloud, points from the initial sphere were not transported at the same place by the true and reconstructed displacement fields. This behavior is expected, as nothing prevents the reconstructed surface from sliding along the surface represented by the point cloud. In Figure 6.2d, we plotted the displacement error in the reconstructed mesh. The displacement error increases when we move away from the fixed boundary, and it reaches 0.6 at the top of the object (the sphere radius is 1).

To increase the accuracy of the method, more information should be included in the model, such as landmark correspondence, curvature information or distinguishing free and loaded surfaces.

6.1.2 Regularizing effect of the adjoint method

An advantage of the optimal control approach is that the mesh boundary nodes where forces apply can be chosen in advance, according to the physical model. In contrast, standard registration methods, by adding artificial forces depending on the data, impose that the forces that create the displacement be located close to data points. Using the 2D toy problem, we compare the registration result from the adjoint method with the result from a displacement-based method, which is similar to standard registration methods.

In the displacement-based method we implement, the optimization problem reads

$$\min_{\mathbf{u}\in\mathbb{R}^{3n}} \quad J(\mathbf{u}).$$

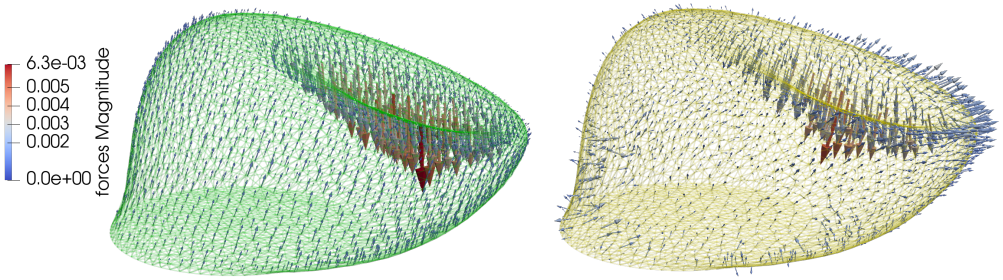
Here, the optimization variable is $\mathbf{u}$. To make the objective function decrease, $\mathbf{u}$ should move in the direction opposite to the gradient of J. Following the extension-regularization framework (discussed in Section 4.1.2), instead of moving in the standard gradient direction $-\nabla J(\mathbf{u})$, we compute a gradient using the linear elastic inner product. This results in the scaled iteration

$$\mathbf{u}_+ = \mathbf{u} - \beta \mathbf{A}^{-1} \nabla J(\mathbf{u}),$$

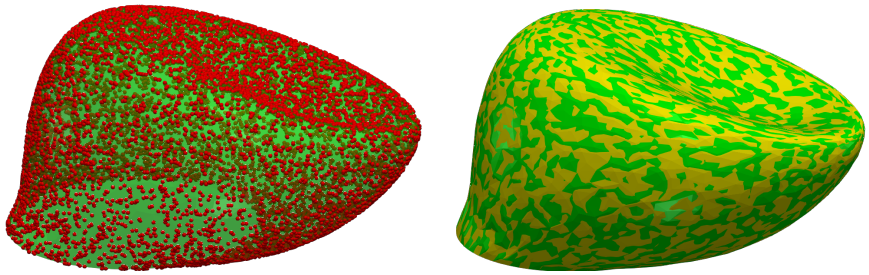
where $\mathbf{A}$ is the stiffness matrix. In other words, the opposite gradient direction $-\nabla J(\mathbf{u})$ is transformed into a force that generates a direction descent in the space of displacements, which is exactly what is done in standard registration methods. As $\nabla J(\mathbf{u})$ is entirely supported by boundary nodes, the successive displacement fields are ensured to be elastic displacements created by a surface force distribution.

Remark 6.1. If $\nabla J(\mathbf{u})$ had some components inside the mesh, for instance if there were some landmarks inside the mesh, the resulting force distribution would also have components inside the mesh. This is how little control on the force distribution support we have when using a displacement-based method.

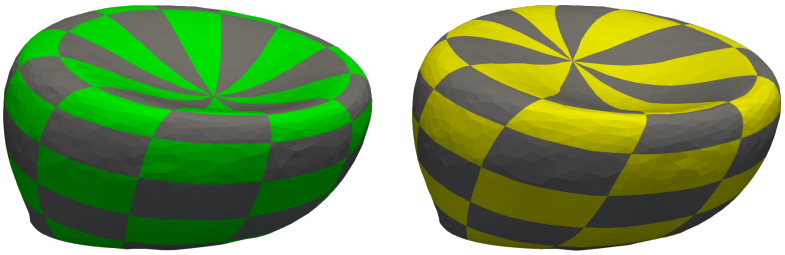
The displacement fields resulting from the adjoint and displacement-based methods are plotted in Figure 6.3a, along with the surface force distribution necessary to create



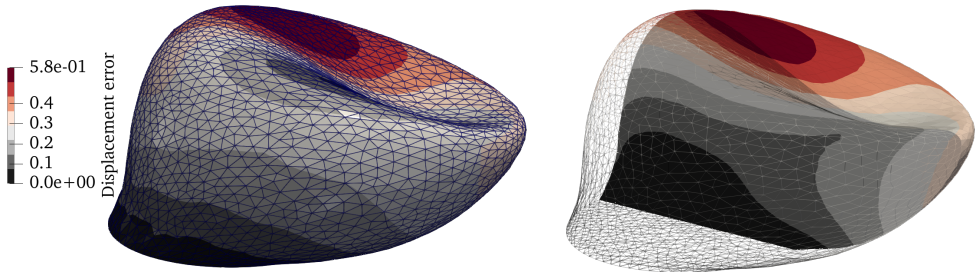
(a) Comparison between true (left) and reconstructed (right) forces.



(b) Point cloud and superposition of the true and reconstructed surface.



(c) Discrepancy between the true (left) and reconstructed (right) surface.



(d) Displacement error.

Figure 6.2: Registration results in the ideal case.

the displacement. For the displacement-based method, these forces are located only on vertices close to data points, which supported the gradient $\nabla J(\mathbf{u})$ along iterations. The deformation computed using the adjoint method is more regular and the force distribution is spread all over the mesh boundary. Let us be more specific about the impact of the adjoint method. Both deformations are solutions to the optimal control problem, as the feasible set *allows* (but does not require) force distributions to be spread all over the mesh boundary. But the forces in the right figure actually spreading all over the mesh boundary is a consequence of the regularizing effect of the adjoint method. Another optimal control method would not necessarily return such a regular distribution without an additional regularization mechanism.

Figure 6.3b shows the convergence and displacement error statistics. As the toy problem was generated using a regular force distribution, it could be expected that the volume registration error is lower in average for the adjoint method. The objective function graph illustrates the degradation of numerical properties mentioned in Remark 5.1. By adding a regularization layer when computing descent directions, the adjoint method makes also convergence more difficult. As the problem has a geometrical meaning, the space of displacements (in other words the geometrical space) is more relevant to solve it, and for this reason convergence is way faster for the displacement-based method.

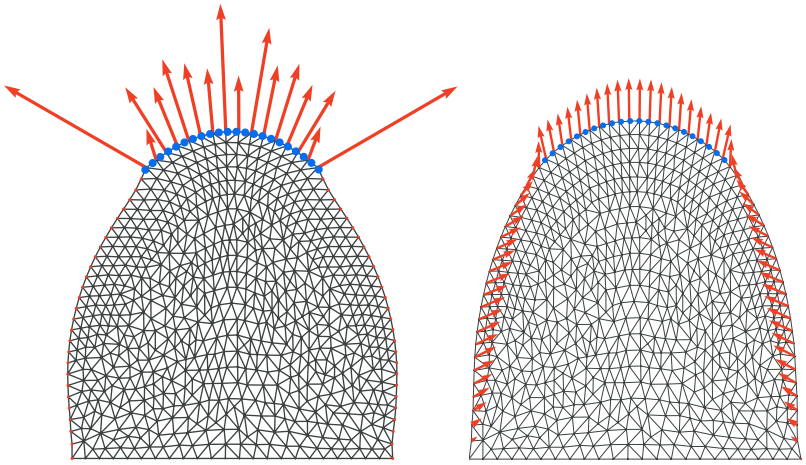
6.1.3 Irregularity due to noise

In the context of augmented surgery, the point cloud is the result of an image processing pipeline, and as a consequence it comes with a certain amount of noise. In this case, the deformed mesh from the registration process should not meet every point in the cloud, as an irregular point cloud would yield an artificially irregular surface on the augmented reality view. Such numerical artefacts might be misleading for the medical staff. It is preferable that the mesh boundary pass through the point cloud without matching each point individually.

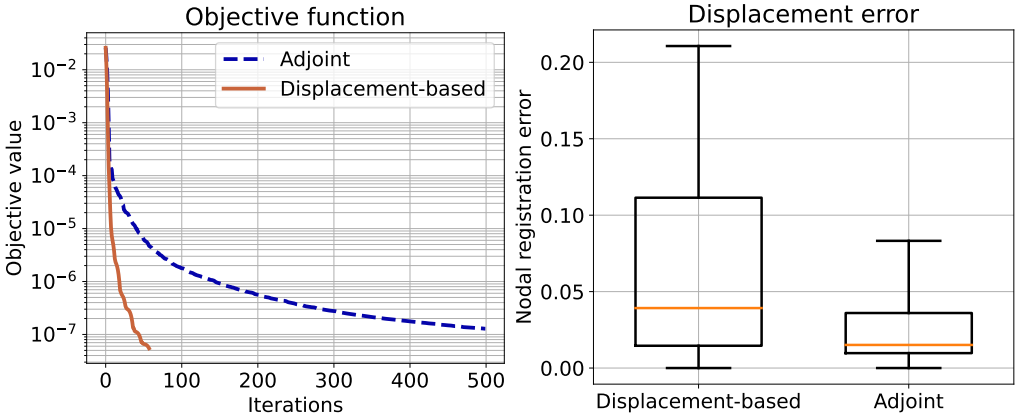
In this section we consider two simple approaches to regularize the computed displacement field in the mesh, in the context of the FreeFem++ implementation (i.e. where the control is $\mathbf{g}$). Those two approaches are those suggested by the optimal control formulation. In the penalized approach, we set $R(\mathbf{g}) = \frac{\alpha}{2}\|\mathbf{g}\|^2_{L^2}$, and the L^∞ constraint is dropped. In the constrained approach, the penalty function R is set to 0 while the L^∞ bound is set to a positive value M.

Comparison on a partial matching case

We experiment those two solutions on the two-dimensional toy problem. We add gaussian noise with standard deviation $\sigma = 0.02$ to the point cloud, and we solve the registration problem using Ipopt with a full BFGS Hessian approximation. The displacement error statistics for several parameters are displayed in Figure 6.4a. The median error seems constant across box plots, but the maximal displacement error depends on the chosen strategy. The penalty strategy did not manage to decrease the displacement error. Starting from the same distribution as the unregularized case for small coefficients, the



(a) Displacement-based (left) and adjoint-based (middle) registration results.



(b) Evolution of the objective function and node displacement errors.

Figure 6.3: Comparison between the displacement-based approach and the adjoint method.

displacement error does not reach a minimum before the penalty is too strong for the mesh to match the point cloud. On the other hand, a significant decrease is achieved with the constraint strategy, with a minimum for $M = 0.4$. Note that the final force distribution in the noise-free case has a maximal intensity of 0.5.

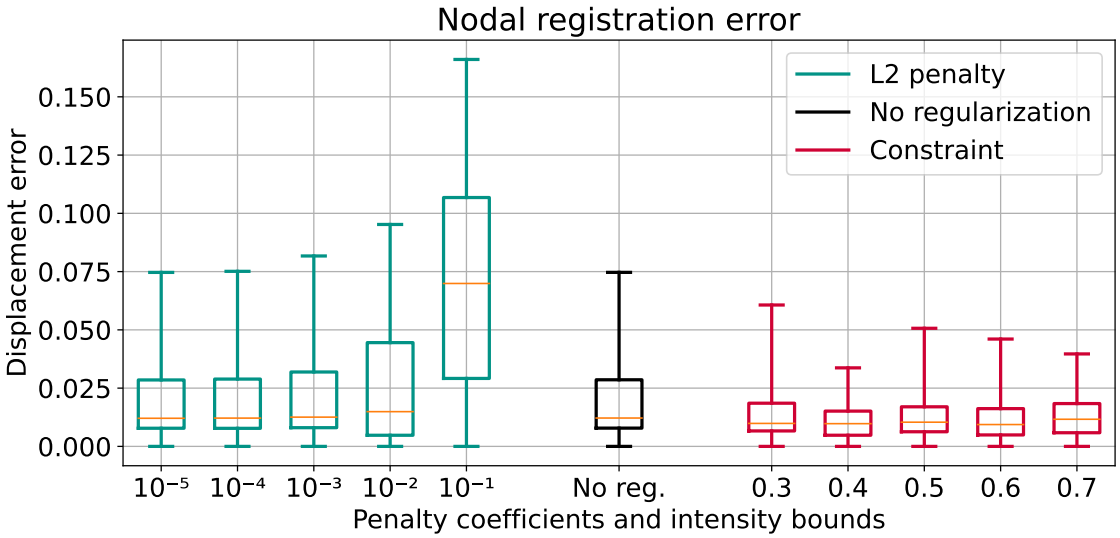
In Figure 6.4b, we plotted the resulting force distribution for the unregularized case, the constrained case that achieves the minimal displacement error ($M = 0.4$), and the first penalized case where the error distribution is impacted ($\alpha = 10^{-1}$). The constraint and penalty strategies both result in more regular surface forces, but the constrained result still features an irregular surface. However, the case $M = 0.4$ did a satisfying job. The corresponding displacement errors are plotted in Figure 6.4c. It is noteworthy that the main source of error is not the noise in Γ, but rather the point cloud not covering the whole mesh boundary, resulting in large displacement error far from the points. This phenomenon appears in the unregularized and the penalty case. Finally, the lower error in the constrained case is just an uncontrolled consequence of the mesh being pushed inwards to compensate for the intensity bound. On the other hand, the penalty approach just resulted in a more regular surface without impacting the remaining of the mesh, which is no more than what we expect from a regularization strategy. Far from breaking the tie between the two strategies, this example shows that using a regularization has an impact on the region close to the point cloud as well as on regions far from the point cloud.

Comparison on a full matching case

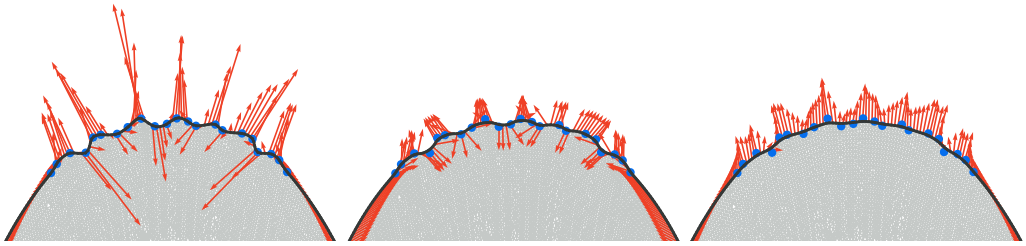
For the sake of curiosity, we also compare the two regularization strategies on a full surface-matching case. Now, the point cloud represents the whole deformed surface, so that the error is less likely to come from the limited point cloud coverage. The error statistics are plotted in Figure 6.5, along with the force distribution and displacement error field for a few selected cases. It seems that constrained approaches managed to reduce the displacement error where it is the largest (close to certain points). This time, the penalty strategy was effective to decrease the displacement error. In particular, when $\alpha = 10^{-3}$, the maximal displacement error is three times smaller than in the unregularized case.

The conclusion of this section is certainly not that one of those two strategies is better than the other in terms of noise regularization. The choice of a strategy should take account of other arguments in favor of one or the other option. Those arguments include ease of implementation for the penalty and the possibility to reject excessive forces for the constraint.

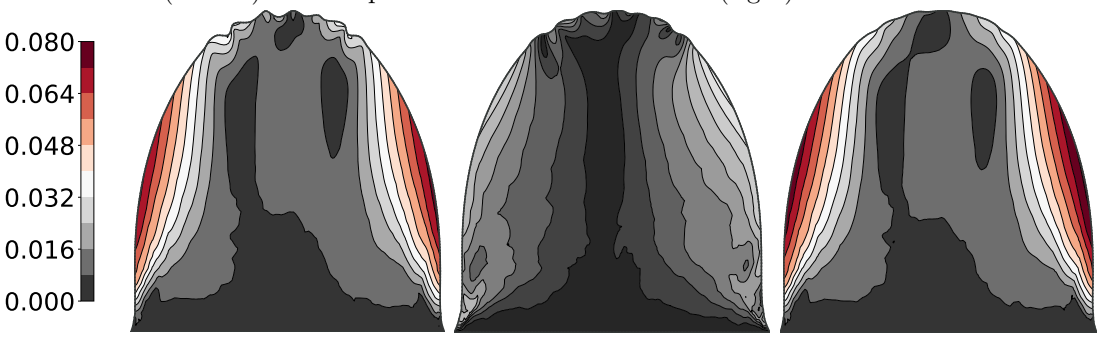
Remark 6.2. In the early implementation, we also experimented the H^1 penalty, defined by $R(\mathbf{g}) = \frac{\alpha}{2}\|\nabla \mathbf{g}\|^2_{L^2}$. However, when the penalty coefficient α was sufficiently large to regularize the mesh boundary, it also resulted in a quasi-uniform force distribution. In addition, this penalty was difficult to use in the three-dimensional cases, where the optimization variables are the nodal forces $\mathbf{b}$. For these reasons, we quickly dropped this option.



(a) Nodal registration error for several penalty coefficients and constraint bounds.

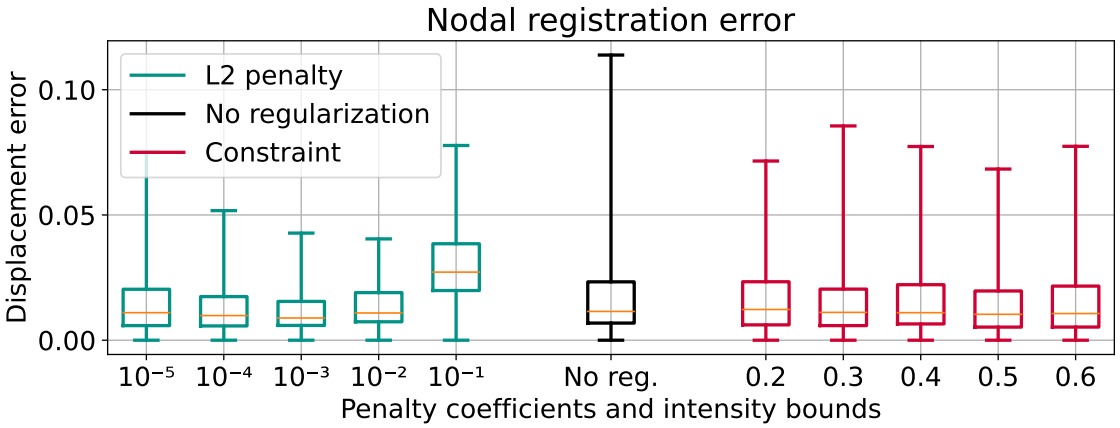


(b) Resulting force distributions for the case without regularization (left), the constrained case with $M = 0.4$ (middle) and the penalized case with $\alpha = 10^{-3}$ (right).

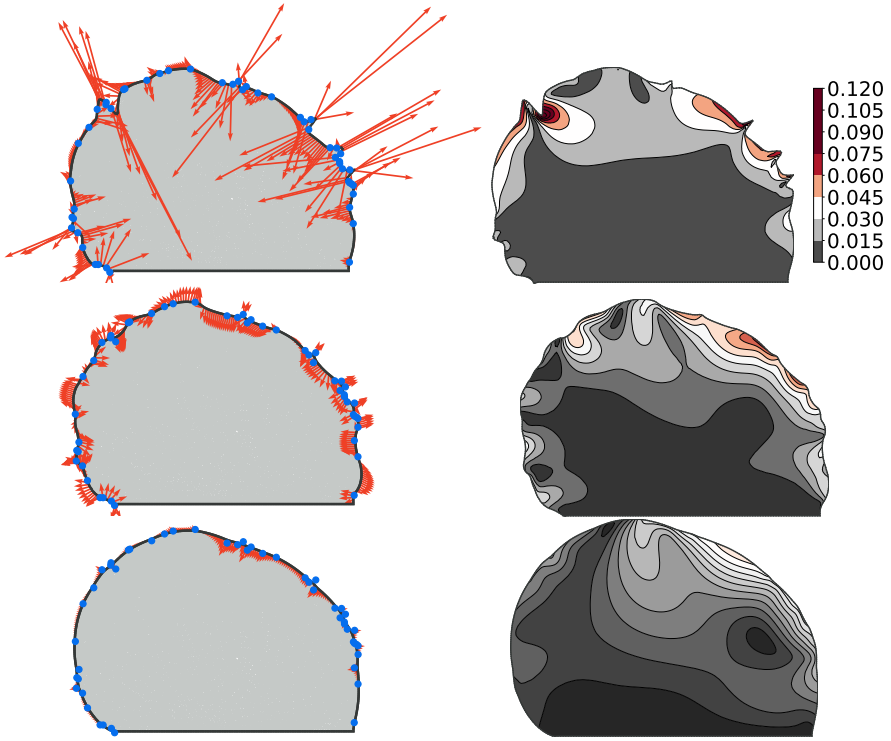


(c) Displacement error inside the mesh for the three cases.

Figure 6.4: Comparison of several kinds of regularization for the partial matching case.



(a) Nodal registration error for several penalty coefficients and constraint bounds.



(b) Resulting force distributions and displacement error for the case without regularization (top), the constrained case with $M = 0.5$ (middle) and the penalized case with $\alpha = 10^{-3}$ (bottom).

Figure 6.5: Comparison of several kinds of regularization for the full matching case.

6.2 Sparse Data Challenge dataset

The accuracy of a registration method is critical for the augmented surgery application. A good accuracy allows the surgeon to target small structures inside the liver while avoiding blood vessels. In practice, clinical collaborators of the Mimesis team usually require an accuracy of 5 millimeters. Though we did not state any theoretical result about accuracy, here we evaluate this criterion experimentally.

We use the adjoint method on a liver registration problem involving the *Sparse Data Challenge*[1] dataset. The dataset consists of one tetrahedral mesh representing a liver phantom in its initial configuration and 112 point clouds acquired from deformed configurations of the same phantom (Collins et al., 2017; Brewer et al., 2019). To generate data, the challenge organizers used a silicon liver phantom. They created deformations by laying the phantom on irregular supports on its posterior face, and acquired pictures of the anterior face with a camera to produce the point clouds (Rucker et al., 2014, Figure 2). The phantom also contains 159 targets whose position is measured in reference and deformed configurations to establish ground truth data. The position of targets remains unknown to us, and the registration error is computed by the challenge website after we upload the final mesh position associated with each point cloud. By keeping the ground truth hidden from participants, the challenge organizers ensure that there is no bias associated with knowledge of the ground truth.

Due to the acquisition process, the forces that create the deformation are only applied on the posterior surface of the liver, while the surface represented by the point cloud is the anterior surface. As a consequence, we label the posterior surface as the *loaded surface*, where the force distribution is allowed to take nonzero values, while the anterior surface is labeled as the *matching surface*, which is used to evaluate the objective function. Figure 6.6 illustrates the definition of the matching and loaded surfaces.

We first perform a rigid alignment between the matching surface and the point cloud using the standard Iterative Closest Point method (Besl and McKay, 1992). The provided point clouds are split into four parts which correspond to parts of the liver that are easily identifiable, namely, the right and left ridges, the falciform region and the general surface. This information is used to initialize the rigid registration process. The point cloud and the mesh after rigid registration are shown in Figure 6.6.

We now turn to the elastic registration process. To ensure uniqueness of the solution to the direct elastic problem, we set a fixed boundary condition on a small zone of the posterior face. As the liver mesh represents a phantom, no information about blood vessels is available and for this reason we just choose six adjacent triangles close to the center of the posterior face to apply the fixed displacement constraint. We solve problem (5.3) (with $R(\mathbf{b}) = 0$) using the adjoint method. As only small deformations are involved in this dataset, we use a linear elastic model for the liver phantom, with $E = 1$ and $\nu = 0.4$. The procedure is stopped after 200 iterations (80s on our configuration).

Let us add a word about the choice of elasticity parameters, which are not specified by the challenge organizers. Actually, a small part of the ground truth is available to

[1] Challenge website: sparsedatachallenge.org.

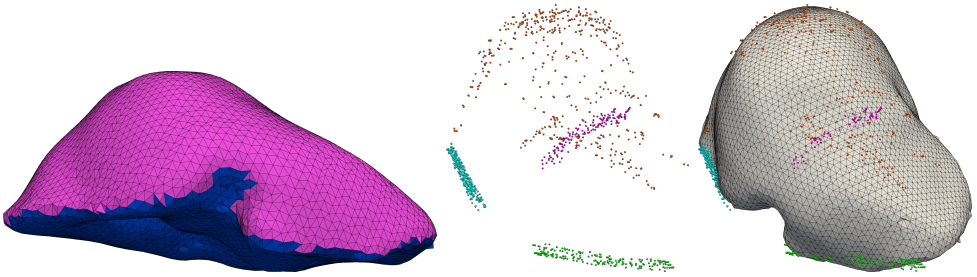


Figure 6.6: Left: Initial liver mesh with matching surface on top and loaded surface at the bottom. Right: Point cloud and liver mesh after rigid registration. Point colors correspond to the left ridge (blue), the right ridge (green), the falciform region (magenta), and remaining points (yellow), respectively.

participants in order to calibrate some parameters. This sample consists of the position of 35 targets (out of 159) after deformation for 4 cases (out of 112). The partial ground truth was used to determine the Poisson ratio ν. Concerning the Young modulus E, it has no impact in this study, and for this reason we set it to 1. Indeed, in this example we retrieve a displacement, and we are not interested in the force intensities. Even for nonlinear models, E is a multiplicative scaling parameter in the elastic energy expression, which mean that the force distribution necessary to generate a given displacement field is proportional to E. As a consequence, changing the Young modulus just means changing the intensity of estimated forces, while the computed displacement remains the same.

Figure 6.7 shows the target registration errors associated with the partial ground truth data provided by the challenge authors (position of 35 targets for 4 point clouds). The box plot shows that significant improvement is achieved by the elastic registration step with respect to the rigid registration result. In average, the elastic registration improves the target registration error by 46%. However, these error results are very partial, while the only error statistics for the full dataset are those displayed by the challenge website after submission of the results. In Table 6.1, we reported the target registration error statistics for all datasets. We compare these results with other submissions to the challenge in Figure 6.8. We obtained the second-best result displayed on the challenge website. In particular, our performance is very close to that of the Vanderbilt team (the challenge organizers), who also use approaches based on a mechanical model (Heiselman et al., 2020). This confirms that approaches that care about respecting the direct model are relevant in augmented surgery.

6.3 Local force estimation

Is it possible to give a meaning to the force distribution reconstructed by the optimal control method? If it applies onto a large part of the liver boundary, probably not. However, when the distribution is very local, estimating the resulting net force may be considered. In this section we consider an example associated to robotic surgery.

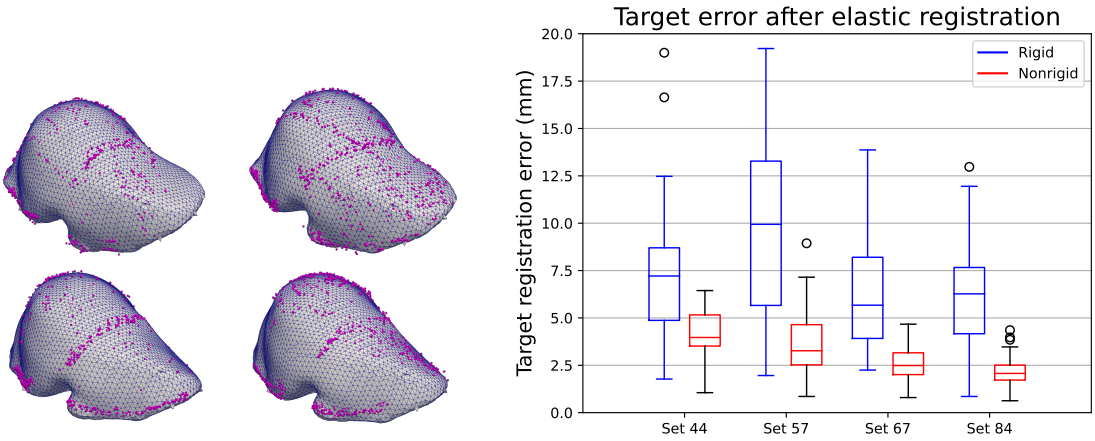


Figure 6.7: Target registration error for datasets where ground truth is available (from left to right, top to down : sets 44, 57, 67, 84). As these datasets were used to calibrate the method, results are better for these sets than in average.

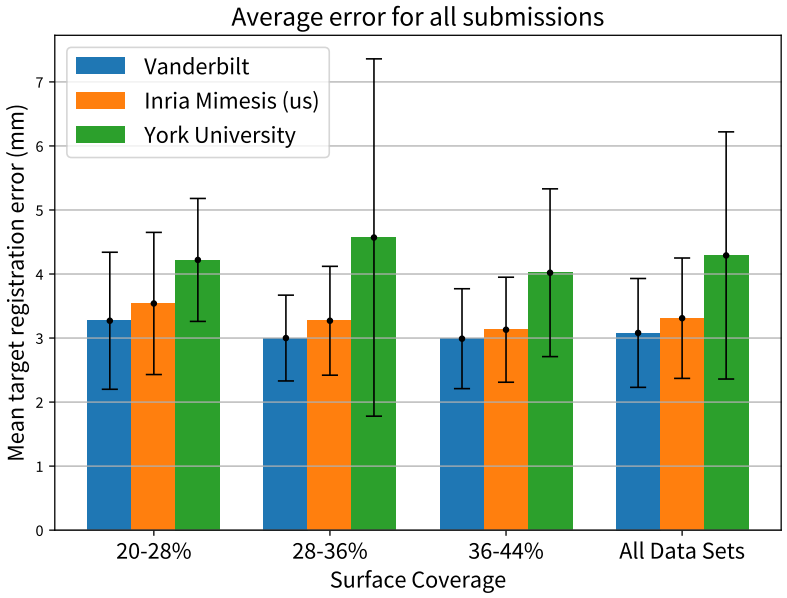


Figure 6.8: Comparison of our accuracy results with other submissions. Our column is the middle one.

Surface Coverage	Average	Standard deviation	Median
20-28 %	3.54	1.11	3.47
28-36 %	3.27	0.85	3.19
36-44 %	3.13	0.82	3.13
All data sets	3.31	0.94	3.19

Table 6.1: Target registration error statistics (in millimeters) for all datasets, as returned by the website after submission.

In remote laparoscopic surgery, the surgeon manipulates the patient's liver by controlling a robotic arm. In this context, estimating the force applied by the robotic instrument is necessary, as an excessive force applied onto the organ may cause serious damage to living tissues and threaten the intervention outcome. On the other hand, it is difficult to embed force sensors on instruments that evolve in an environment like the human body, mostly due to geometry, biocompatibility and asepsis constraints (Marbán et al., 2014). For this reason, certain standard surgical systems, such as the famous daVinci robot are not equipped with force sensors. Indirect methods based on image processing have been proposed to keep track of the applied force without a sensing system inside the patient's body (Nazari et al., 2021).

Here we estimate a force in a context similar to that in Haouchine et al. (2018), where the authors monitor the evolution of the force applied by an instrument using a sequence of point clouds extracted from laparoscopic frames. We generate synthetic test cases using a liver mesh of 3,046 vertices and a linear elastic model ($E = 20,000$ Pa, $\nu = 0.45$). Dirichlet conditions are applied in the region where the hepatic vein enters the liver and in the falciform region. To simulate an instrument manipulating the liver, we apply a very local force onto the mesh. We then sample the deformed mesh around the force application point to create a point cloud. Figure 6.9 shows the displacement generated by synthetic local forces, along with the reconstructed deformation using the sampled point cloud of 500 points.

6.3.1 Single estimation

We begin by estimating the single local force from the case illustrated in Figure 6.9. To do so, we allow the force distribution $\mathbf{b}$ to be nonzero only in a small zone (about 50 vertices) surrounding the triangles concerned by the traction force. We solve the optimization problem with a relative tolerance of $5 \cdot 10^{-4}$ on the objective gradient norm, and we compute the net force estimation as the sum $f_{\text{est}} = \sum_i b_i \in \mathbb{R}^3$ of all the nodal forces of the reconstructed distribution $\mathbf{b}$. The estimated force is compared to the ground truth f_{true} by computing the relative error $\|f_{\text{est}} - f_{\text{true}}\|/\|f_{\text{true}}\|$. Note that working with nodal forces makes it easy to work with several meshes, as we do not have to take the area of surface triangles into account.

In figure Figure 6.9, the same mesh is used to generate the data and to perform the registration. As a consequence, performing an accurate registration is easy, as the

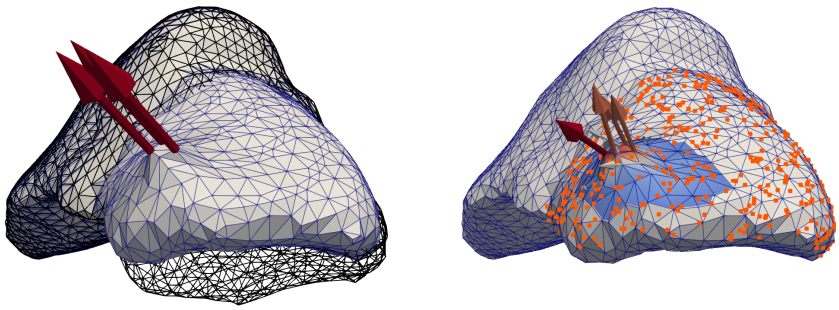


Figure 6.9: Left: synthetic deformation generated by a local force. The black wireframe represents the reference configuration. Right: reconstructed deformation using the point cloud. Nodal forces are summed to produce a net force estimation.

shape of tetrahedrons appears in the point cloud. The reconstructed force distribution and the estimation error are represented in Figure 6.10, on the left. At the end of the optimization process, the relative error on the net force is below 1%. Actually, we noticed that, with a denser point cloud (5,000 points), the reconstructed force seems to converge toward the ground truth force. This bias appears when the same discretization is used for data generation and reconstruction. On the right part of Figure 6.10, we show the result of registration performed with a different mesh (3,829 vertices), obtained by remeshing the original mesh. The force distribution is noisier and nodal forces take larger values, as tetrahedrons must be stretched to match the point cloud. Though, the noise is somehow filtered by summing all nodal forces, leaving a net force estimation with an error below 10%.

In Figure 6.11, we present the same effect with a refined mesh, (19,234 nodes) obtained by splitting the original mesh. The L^∞ norm of nodal forces, plotted in the top right graph, increases along iterations. The bottom images show the force distribution at iterations 57 (when the estimation error is minimal) and 293 (last iteration). Though the force distribution at the last iteration is very noisy and takes large values, the estimation error remains stable around 8%.

6.3.2 Estimation of a variable force

To mimic the action of a robotic tool manipulating the liver in the field of view of a laparoscopic camera, we create sequences of point clouds by applying a time-dependent force to a small spot on the liver surface. For each frame in the sequence, the resulting displacement is computed from the current force and a part of the deformed boundary is sampled to create a point cloud of 500 points. Each sequence consists of 50 forces, with a displacement of about 1 mm in a random direction between two successive forces. We create 5 sequences, where the force is applied on different points.

The registration is performed using the remeshed mesh of 3,829 vertices, already used in Figure 6.10 on the right. For each sequence, the observed point clouds are successively

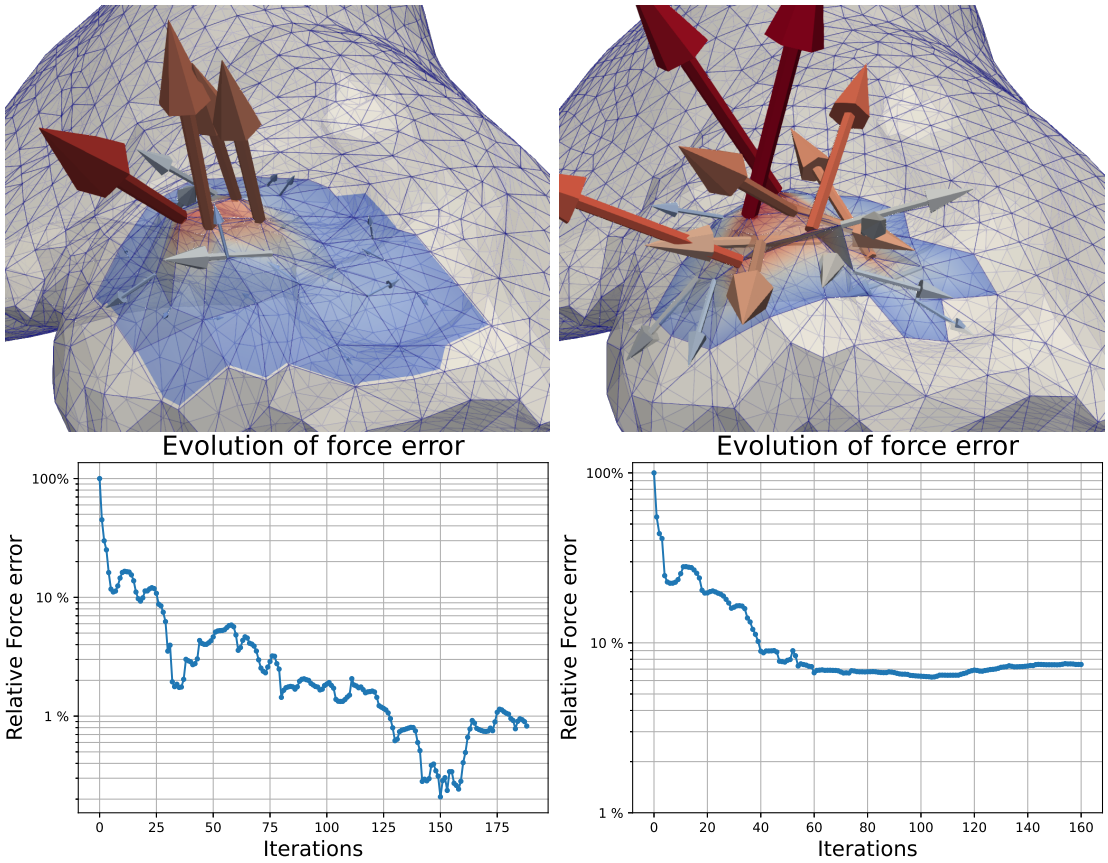


Figure 6.10: Reconstructed force distribution and evolution of the force estimation error for the original mesh (left) and the remeshed mesh (right). Nodal forces are summed to produce a net force estimation.

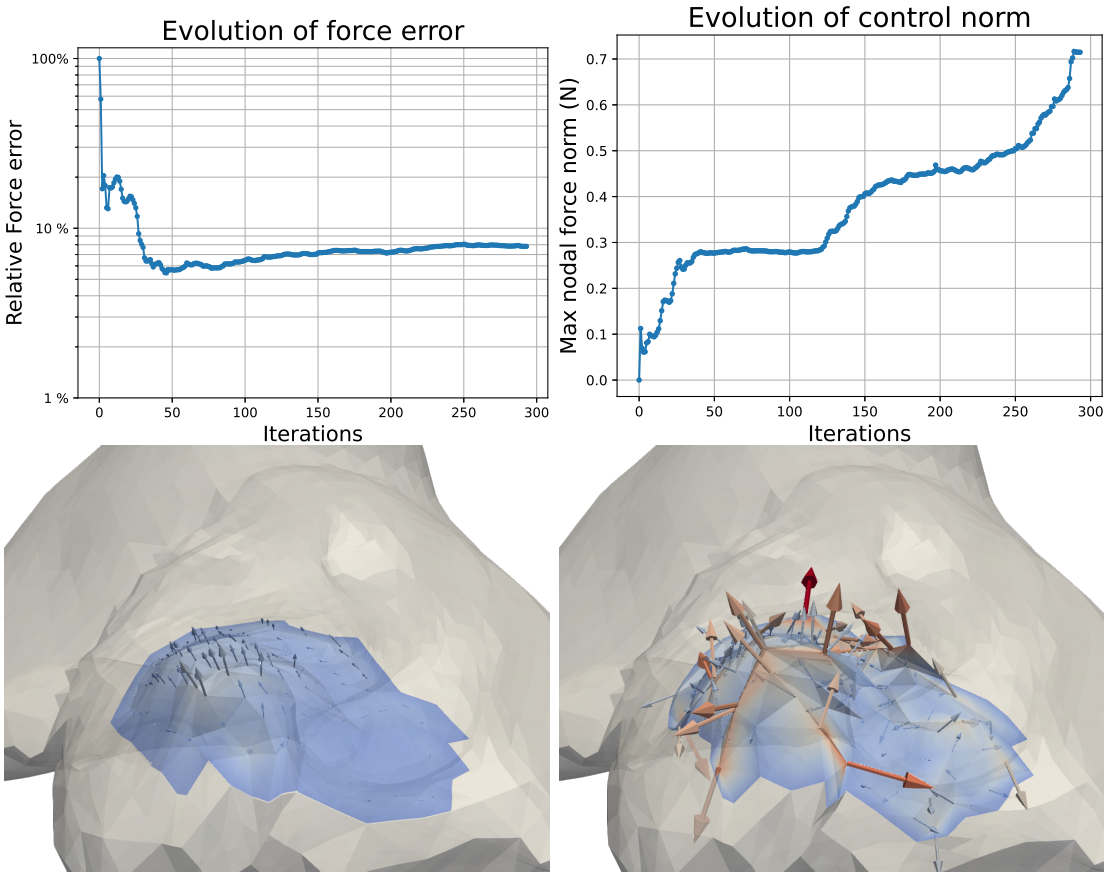


Figure 6.11: Top: Evolution of the force estimation error and the maximal nodal force for the refined mesh. Bottom: reconstructed force distribution at iterations 57 (left) and 293 (right).

Sequence	N. eval.	Update Time	Relative error
Case 1	9.2	1.42 s	8.9 %
Case 2	5.6	0.85 s	16.2 %
Case 3	5.5	0.84 s	5.7 %
Case 4	5.4	0.82 s	4.4 %
Case 5	6.2	0.96 s	15.0 %

Table 6.2: Average number of objective evaluations, execution time per update and relative error for each sequence.

fed to the procedure to update the force estimation by solving the optimization problem. The optimization algorithm is initialized with the last reconstructed distribution, so that only a few iterations are required for the update. Table 6.2 shows the average error obtained for each sequence, together with the average execution time of the updates and the number of evaluations of the objective function. The noise in the reconstructed distribution results in an intensity smaller than the reference, which is the main cause of an overall error of 10 % in average. In all cases, the update time is below 1.5 s in average, which is an encouraging result to show the feasibility of our approach in a real-time context.

Note that this test case is only a toy problem, which is far from representing clinical conditions. In our result, the main source of error is the use of two different meshes for data generation and registration. However, other sources of error arise in a clinical context, as the physical model only provides a rough approximation of the real system. For instance, the estimated net force is proportional to the Young modulus used in the liver model. Patient-specific parameters may be estimated using elastography, but nowadays this procedure is not part of a standard clinical routine. According to Oudry et al. (2014), an error of 20 % is typical for clinical elastographic measurements. The elastographic error, combined with other sources of error, might amount to an overall error that prohibits the estimation of the robot force. Though, our results are at least encouraging in terms of performance. In a clinical software, it is not expected to use meshes much finer than those used in this study, in particular because of memory requirements due to computing on graphic processors.

6.3.3 Toward force estimation with a hyperelastic model

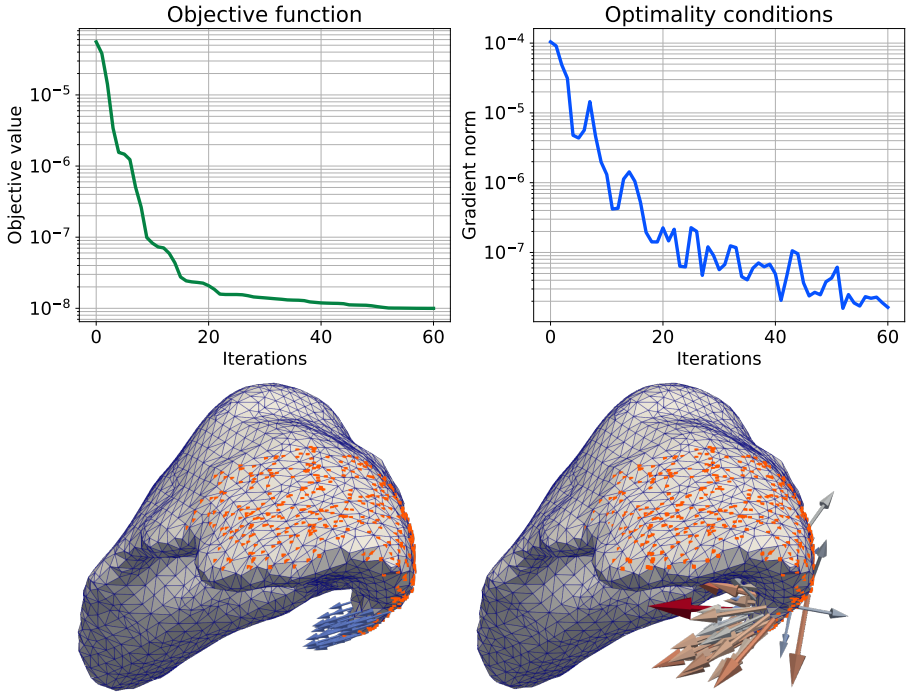
All along the project, we encountered difficulties to use the adjoint method with a nonlinear hyperelastic model, because of the instability of standard Newton methods available in most finite element packages. This point is discussed in Section 5.4. However, we are now in a position to present an early convergence result involving a Neo-Hookean model. Elastic parameters are the same as previously.

Like previously, we created synthetic deformations by applying a local force onto the liver mesh. As hyperelastic models are less rigid than the linear elastic model, applying nodal forces onto one or two triangles only results in a very irregular displacement field.

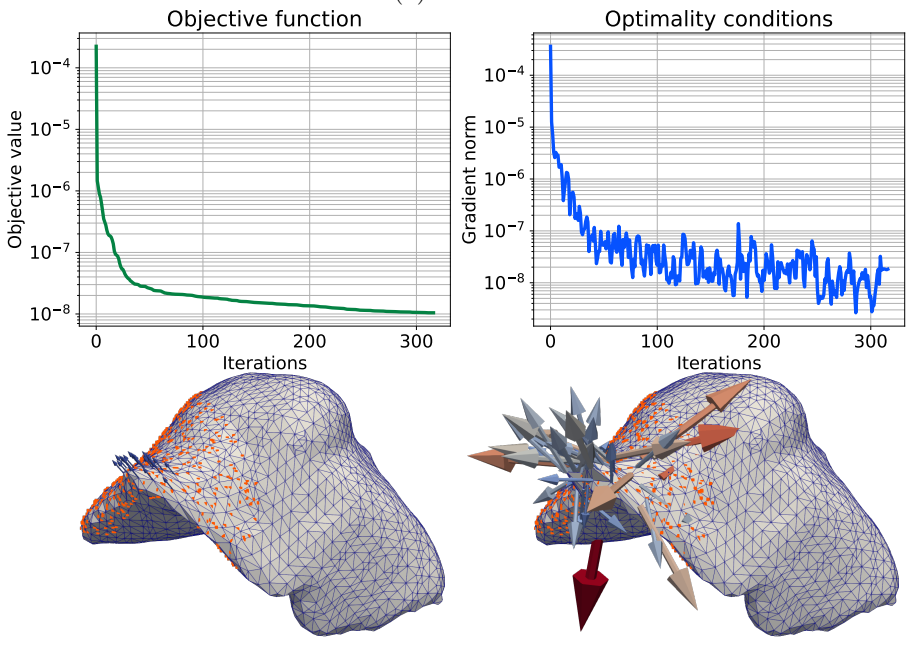
To produce more realistic deformations, we spread the synthetic force on a slightly larger zone. The reconstructed force distribution is also spread on a zone larger than before.

In this example, the finite element model is managed using the Dolfin package (Logg and Wells, 2010; Logg et al., 2012). To solve the direct problem, we combine both Newton methods described in Section 5.4. For the first iterations, we use a trust-region Newton method available in the Scipy package. This Newton method is guaranteed to converge, but it is also very slow. As a conjugate gradient method is used to solve linear subproblems, the trust-region Newton method can only solve the nonlinear problem up to a residual norm of 10^{-9}. To obtain a solution with a tighter tolerance, we perform a few iterations with the standard Newton method available in Dolfin to reach a residual norm of 10^{-14}.

Figure 6.12 shows convergence statistics and resulting deformations for two test cases. Here the same mesh is used for data generation and reconstruction. In both cases, we performed as many adjoint iterations as possible, letting the solver fail when the direct solver accuracy is not sufficient to compute descent directions. Note that this failure did not happen after the same number of iterations for both examples. For case 1, iterations stopped after 10 min 49 s, while for case 2 they stopped after 1 h 4 min 11 s (on a single thread). However, in both cases the adjoint method managed to decrease the gradient norm by approximately four orders of magnitude. This result shows the feasibility of the adjoint method with a nonlinear elastic model. From the point of view of the inverse problem, however, the very noisy force distributions illustrate the difficulty associated to an enlarged support for nodal forces.



(a) Case 1.



(b) Case 2.

Figure 6.12: Convergence statistics and resulting deformations for two test cases with the nonlinear model. The true deformation and force distribution are plotted on the left, while the reconstructed deformation and force distributions are plotted on the right.

Chapter 7

Future developments and conclusion

This project results from discussions between Yannick Privat and Stéphane Cotin, about the possibility to use a mathematical point of view to tackle the liver registration problem. The early work was quite prospective, and the optimal control approach we came up with is the result of numerous discussions, negociations and also false leads. While some choices result from tests and comparison between several methods, other choices were made by default in order to move forward in the project. The first category includes the choice of tree-based procedures to perform orthogonal projections, while the second category includes the choice to use an adjoint method or to set Dirichlet boundary conditions. In this chapter, we discuss a few possible enhancements for our work. We did not implement those options by lack of time, but they could improve computational performances or remove difficulties from the problem.

7.1 Avoid unnecessary difficulties by dropping Dirichlet boundary conditions

All along this manuscript, we only considered static elasticity problems involving a Dirichlet boundary condition. The main role of the Dirichlet boundary condition is to ensure the existence of a solution displacement u for a given force distribution g. Its role is also, to a lesser extent, to take into account anatomic factors that limit the liver motion. In liver surgery, the Dirichlet condition is often set at the hepatic vein entry, or sometimes at places where ligaments hold the liver (e.g. the falciform region). Note that the Dirichlet boundary is necessary because a static elastic model is used. When a time-dependent dynamic model is used, existence of an equilibrium configuration is not required.

However, using a Dirichlet condition is far from being the best option, and it was even a source of difficulty in certain parts of the project. From a theoretical point of view, the presence of mixed boundary conditions degrades the regularity of solutions to the

elasticity system. In particular, it is common to see stress accumulation at the interface between Dirichlet and Neumann boundary conditions. Due to this lack of regularity, we failed to prove the existence of a solution to the optimization problem in the general case. From a numerical point of view, setting a Dirichlet boundary condition before elastic registration sets the initial rigid registration result in stone. If the rigid registration went wrong, this cannot be corrected during elastic registration. Finally, from the modeling point of view, a hard Dirichlet condition does not even model accurately liver/ligaments. Ligaments, as well as main blood vessels, limit the organ motion but do not prevent it completely. For all these reasons, getting rid of the Dirichlet boundary condition should remove a difficulty from the problem. Though, an alternative mechanism should be used to ensure the uniqueness of solutions to the direct problem.

Replacing Dirichlet condition with stiffnesses

A first possibility is to relax Dirichlet boundary conditions into Robin boundary conditions. Instead of being fixed, the corresponding surface $\partial\Omega_{\mathrm{D}}$ is subject to a force that attracts it toward a target position u_{D}. This model seems particulary relevant when ligaments are modeled as linear or nonlinear springs. For instance, Nikolaev and Cotin (2020) model ligaments that hold the liver as a neo-Hookean material, whose stiffness is estimated using a Kalman filter. To keep things simple, assume that interactions with ligaments are represented using a cubic stiffness. The elasticity problem then reads

$$\min_{u} \quad W(u) + \frac{\kappa}{3} \int_{\partial\Omega_{\mathrm{D}}} \|u - u_{\mathrm{D}}\|^3 \,\mathrm{d}s - \int_{\partial\Omega_{\mathrm{N}}} g \cdot u \,\mathrm{d}s, \tag{7.1}$$

where $\kappa > 0$ is the spring stiffness. With the same notation as (2.2), the resulting strong formulation reads

$$\begin{cases} -\mathrm{div}(w'(\nabla u)) &= 0 & \text{in } \Omega_0 \\ w'(\nabla u)n &= g & \text{on } \partial\Omega_{\mathrm{N}} \\ w'(\nabla u)n &= \kappa\|u_{\mathrm{D}} - u\|(u_{\mathrm{D}} - u) & \text{on } \partial\Omega_{\mathrm{D}}. \end{cases} \tag{7.2}$$

Using stiffnesses instead of Dirichlet boundary conditions may result in a more accurate modeling of the liver mechanics as well as in more flexibility to adjust rigid transformation in the shape matching problem. In addition, provided that forces due to ligaments are not concentrated at one point, the resulting displacement is solution to a Neumann problem with $w'(\nabla u)n \in L^\infty(\partial\Omega_0)$, which might ensure better regularity than mixed boundary conditions.

Note that rigid displacements should be penalized enough by the stiffnesses to avoid unwanted large rotations of the liver during registration. In particular, the stiffness coefficients κ should be large enough, and the restrained surfaces $\partial\Omega_{\mathrm{D}}$ should be spread all around the surface so that the liver can resist against torques. Last but not least, this strategy requires to know the elastic parameters of ligaments holding the liver.

A pure traction elastic problem

Another option consists in using only Neumann boundary conditions and work in a subset of $H^1(\Omega_0, \mathbb{R}^3)$ to ensure uniqueness of solutions to the direct problem. In this situation, all forces that apply on the liver boundary are represented by the unknown Neumann condition. Constraints and penalties on the force distribution g may be specialized, depending on types of forces that are expected on each part of the liver boundary. In particular, larger intensities may be allowed on regions that were previously subject to Dirichlet conditions, as thoses forces are responsible for balancing the forces that drive the registration.

Let us have a look at the linear elasticity case. The former Dirichlet region is now included in the Neumann region, and we denote by $\partial\Omega_{\mathrm{F}}$ the region where a free boundary condition must be applied. It is known that the null space of the bilinear form

$$a(u, v) = \int_{\Omega_0} A\varepsilon(u) : \varepsilon(v) \, \mathrm{d}x$$

is the space of uniform translations $T = \{\mathbb{1}_{\Omega_0} x, x \in \mathbb{R}^3\}$. Thus, the space $H^1(\Omega_0, \mathbb{R}^3)$ can be decomposed into the direct sum $T \oplus T^\perp$ (either for the L^2 or H^1 inner product), where

$$T^\perp = \left\{ v \in H^1(\Omega_0, \mathbb{R}^3) \;\middle|\; \int_{\Omega_0} v \, \mathrm{d}x = 0 \right\}.$$

Provided that the surface force distribution belongs to the zero-mean subspace

$$S^\perp = \{\mathbb{1}_{\partial\Omega_{\mathrm{N}}} x, x \in \mathbb{R}^3\}^\perp = \left\{ g \in L^2(\partial\Omega_{\mathrm{D}}) \;\middle|\; \int_{\partial\Omega_{\mathrm{N}}} g \, \mathrm{d}x = 0 \right\},$$

then solutions in $H^1(\Omega_0, \mathbb{R}^3)$ to the system

$$\begin{cases} -\mathrm{div}(A\varepsilon(u)) &= 0 \quad \text{in } \Omega_0 \\ A\varepsilon(u)n &= g \quad \text{on } \partial\Omega_{\mathrm{N}} \\ A\varepsilon(u)n &= 0 \quad \text{on } \partial\Omega_{\mathrm{F}}. \end{cases} \tag{7.3}$$

are equal up to a uniform translation. In particular, there is a unique solution u_g that belongs to $T^\perp$. Note that in this case, u is the least-norm representant of solutions to (7.3). Finally, to be consistent with the numerical development, we say a word about the case $g \notin S^\perp$, where (7.3) is replaced by the least-squares problem

$$\min_{u \in T^\perp} \tfrac{1}{2} \int_{\partial\Omega_{\mathrm{N}}} \|A\varepsilon(u) - g\|^2 \, \mathrm{d}x \quad \text{s.c} \quad \begin{cases} -\mathrm{div}(A\varepsilon(u)) &= 0 \quad \text{in } \Omega_0 \\ A\varepsilon(u)n &= 0 \quad \text{on } \partial\Omega_{\mathrm{F}}. \end{cases} \tag{7.4}$$

Of course, as $S^\perp$ is the orthogonal of $S = \{\mathbb{1}_{\partial\Omega_{\mathrm{N}}} x, x \in \mathbb{R}^3\}$ in $L^2(\partial\Omega_{\mathrm{D}})$, solutions to (7.4) satisfy

$$A\varepsilon(u)n = g - \frac{\mathbb{1}_{\partial\Omega_{\mathrm{N}}}}{|\partial\Omega_{\mathrm{N}}|} \int_{\partial\Omega_{\mathrm{N}}} g \, \mathrm{d}x.$$

Though the study of the continuous problem needs to be consolidated, the full-Neumann configuration can be translated in our numerical framework without increasing too much the size of the direct system. We now derive a linear elasticity operator adapted to this new situation. When it does not include Dirichlet conditions, the stiffness matrix $\mathbf{A}$ does *not* have full rank. With the same notation as before, its kernel and range read

$$T = \ker \mathbf{A} = \{(x, x, \cdots, x), x \in \mathbb{R}^3\} \qquad \text{and} \qquad T^\perp = \operatorname{im} \mathbf{A} = \left\{ \mathbf{v} \in \mathbb{R}^{3n} \;\middle|\; \sum_{i=1}^{n} v_n = 0 \right\}.$$

To construct the direct problem matrix, we also introduce the matrices $\mathbf{S}$ and $\mathbf{S_b}$ defined by

$$\forall x \in \mathbb{R}^3 \qquad \mathbf{S}x = (x, \cdots, x), \qquad [\mathbf{S_b}x]_i = \begin{cases} x & \text{if } i \in I_\mathbf{b} \\ 0 & \text{otherwise;} \end{cases}$$

$$\forall \mathbf{v} \in \mathbb{R}^{3n} \qquad \mathbf{S}^{\mathrm{T}}\mathbf{v} = \sum_{i=1}^{n} v_i, \qquad \mathbf{S_b}^{\mathrm{T}}\mathbf{v} = \sum_{i \in I_\mathbf{b}} v_i.$$

The matrix $\mathbf{S_b}$ has the same effect as $\mathbf{S}$, but its action only affects nodes corresponding to $\partial\Omega_{\mathrm{N}}$ and has no effect on nodes corresponding to $\partial\Omega_{\mathrm{F}}$ or nodes inside the domain. The direct problem can be formulated in a compact way using the system

$$\left(\begin{array}{c|c} \mathbf{A} & \mathbf{S_b} \\ \hline \mathbf{S}^{\mathrm{T}} & 0 \end{array}\right) \begin{pmatrix} \mathbf{u} \\ \overline{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{b} \\ \overline{0} \end{pmatrix}, \tag{7.5}$$

where $\mathbf{b}$ still only has nonzero coefficients at nodes in $I_\mathbf{b}$. Let us check that this system does what is expected from it. The first line reads $\mathbf{Au} = \mathbf{b} - \mathbf{S_b}\lambda$. A necessary and sufficient condition for solutions to this line to exist is to set λ as the mean of $(b_i)_{i \in I_\mathbf{b}}$, so that $\mathbf{b} - \mathbf{S_b}\lambda$ lies in the range of $\mathbf{A}$. Here, by using $\mathbf{S_b}$ instead of $\mathbf{S}$, we correct $\mathbf{b}$ with a uniform vector field defined only on the nodes of $I_\mathbf{b}$ and avoid to artificially add forces to other nodes. It also ensures that, the adjoint state $\mathbf{p}$ resulting from the adjoint system solution satisfies $\sum_{i \in I_\mathbf{b}} p_i = 0$, which is consistent with the requirement $\mathbf{b} \in T^\perp$. Now, with the first line only, $\mathbf{u}$ is defined up to an element of $\ker \mathbf{A}$. By ensuring $\mathbf{u} \in T^\perp$, the second line $\mathbf{S}^{\mathrm{T}}\mathbf{u} = 0$ enforces the uniqueness of $\mathbf{u}$, which is, again, the least-norm solution of the first line. Thus, dropping the Dirichlet boundary only adds 3 lines to the direct problem.

Coupled with a good shape-matching functional, those two solutions should be good candidates to replace the Dirichlet condition. Compared to the stiffness-restricted problem, the restriction-free problem seems more relevant when little information from the direct model is available, e.g. ligament stiffness is unknown. However, it can result in too many degrees of freedom, especially if the translational component of the displacement is also chosen as an optimization variable instead of being set to zero.

7.2 Jump to lightspeed with neural networks

After looking at update time results in Section 6.3 it is not clear that achieving real-time registration using the adjoint method is feasible, even with a better implementation of the adjoint procedure and a good trade-off between accuracy and computation cost. The adjoint method shares processor time with other parts of the pipeline, including point cloud generation from laparoscopic images and displaying the augmented view. This constatation holds when a linear model is used and the stiffness matrix is factorized once when the procedure begins. When a nonlinear model is used, however, solving the direct problem requires an iterative method, which means several linear systems to assemble and solve. In that case, at least, it seems clear that real-time computation is not feasible as is.

A neural network to predict elastic displacements

Like biomechanical simulation, deep learning is a central research interest in the Mimesis team, and for this reason we considered replacing the iterative nonlinear solver with a neural network. The network, developed in the team by Odot et al. (2022), is a fully-connected multi-layer perceptron with four layers of size $3n$. In such a structure, each neuron in a given layer is connected to every neuron from the previous layer and to every neuron from the next layer. Figure 7.1 shows the network structure. The result $\mathbf{z}_i$ of the i-th layer is obtained from the previous layer using the formula

$$\forall\ 1 \leqslant i \leqslant 4 \qquad \mathbf{z}_i = f_i(\mathbf{W}_i \mathbf{z}_{i-1} + \mathbf{c}_i), \tag{7.6}$$

where parameters $\mathbf{W}_i$ and $\mathbf{c}_i$ result from the network training. The function f_i applies pointwise an activation function to each component of its input. Here the chosen activation function is the Parametric Rectified Linear Unit, defined by

$$\mathrm{PRelu}(x) = \begin{cases} x & \text{if} \quad x \geqslant 0 \\ ax & \text{if} \quad x \leqslant 0, \end{cases}$$

where $a \geqslant 0$ is a learned parameter specific to each neuron. The network input receives the nodal force distribution $\mathbf{z}_0 = \mathbf{b}$ and the displacement field $\mathbf{u} = \mathbf{z}_4$ is read at the other end of the network. The network is trained by showing it a wide range of force distributions, along with the corresponding displacement fields. As neural networks mainly involve matrix-vector products and pointwise operations, implementations take advantage of the architecture of graphic processors to perform very fast predictions. A prediction operation, denoted $\mathbf{u} = \mathbf{N}(\mathbf{b})$, is performed in approximately five milliseconds on a modern computer.

Optimization involving the network

The development of the adjoint method is carried out jointly with Alban Odot, while the network design and training is taken care of exclusively by Alban Odot. The structure

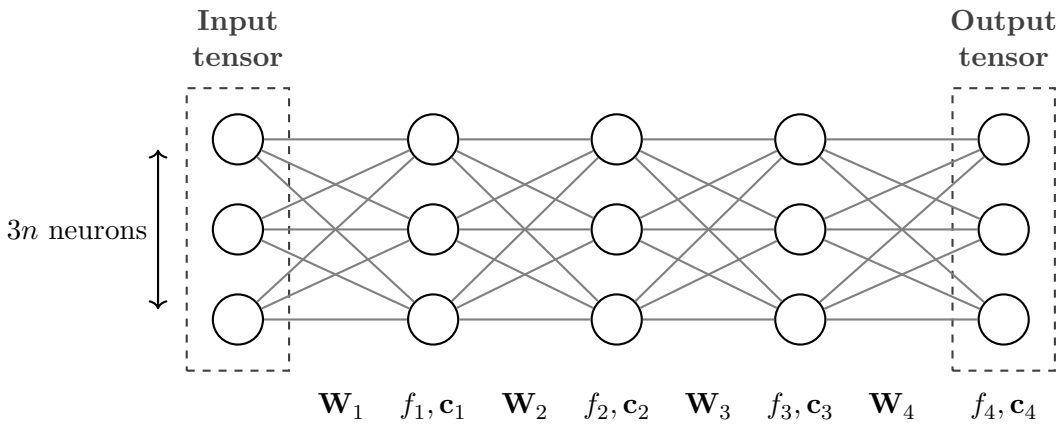


Figure 7.1: Structure of the multi-layer perceptron. The $\mathbf{W}_i$, $\mathbf{c}_i$ and f_i represent the parameters associated to each step of the network.

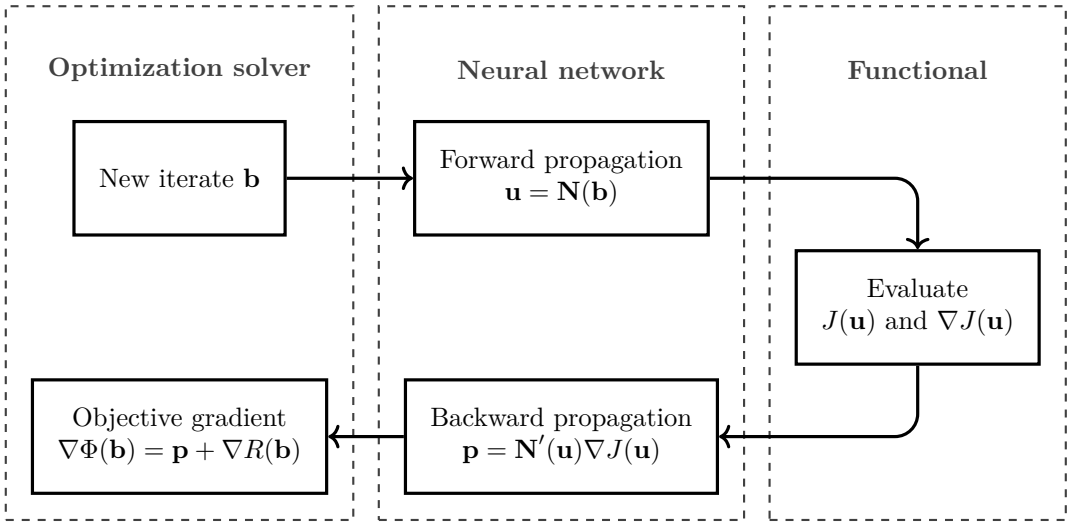


Figure 7.2: Structure of the adjoint method with the neural network. Compared to Figure 5.1, only the middle part has changed.

of the adjoint method involving the neural network is shown in Figure 7.2. In our current implementation, the same optimization solver as before is used. However, network operations are run on the graphic card, using the PyTorch framework. The functional evaluation is performed on the GPU as well, by using the brute-force projection function from PyTorch3D mentioned in Section 5.3. As the optimization solver does not run on the GPU, this implementation requires a lot of exchange between the central processor and the GPU, which is a known source of overhead. In a future version, quasi-Newton operations should run on the GPU to minimize inter-device exchanges.

The whole backward chain is taken care of automatically by the automatic differentiation module in PyTorch. Yet, let us add a word about backward propagation in the neural network. This operation replaces the adjoint problem and involves the adjoint operator of the neural network. In Figure 7.1, information now circulates from the right to the left. The rightmost tensor receives the gradient $\mathbf{s}_4 = \nabla J(\mathbf{u})$ in the space of displacements, while the adjoint state $\mathbf{p} = \mathbf{s}_0$ is read at the left extremity. The operation to progress a layer backward is the adjoint operation to (7.6), namely

$$\forall\ 1 \leqslant i \leqslant 4 \qquad \mathbf{s}_{i-1} = \mathbf{W}_i^{\mathrm{T}} \nabla f_i(\mathbf{W}_i \mathbf{z}_{i-1} + \mathbf{c}_i)\mathbf{s}_i,$$

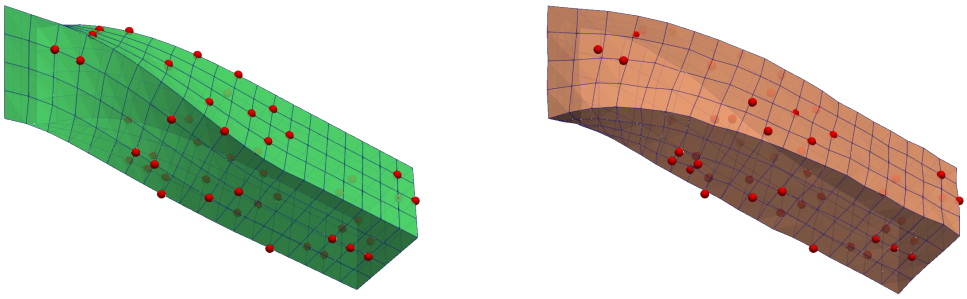
where $\nabla f_i(\mathbf{W}_i \mathbf{z}_{i-1} + \mathbf{c}_i)$ is a diagonal matrix. In other words, the information circulates in the same wires and using the same weights as forward propagation. Each neuron, instead of applying the activation function, multiplies its input by the activation function derivative that has been saved during forward propagation.

We obtained early results using a parallelepipedic beam mesh with a Dirichlet condition at one extremity. A synthetic deformation is created by applying a force distribution on the beam surface and its surface is sampled to create a point cloud. A reconstructed deformation is obtained in approximately 300 ms on our configuration. Figure 7.3 shows the synthetic and reconstructed beam deformation, along with convergence statistics. Beside the fact that the reconstructed beam is twisted the wrong way, the pointcloud matching seems visually acceptable. Despite the use of a L^2 penalty, high-frequency oscillations appear in the reconstructed deformation but keep a reasonable amplitude. Note that the Dirichlet condition is not enforced but learned by the network, and Dirichlet nodes may be subject to a small displacement in certain cases.

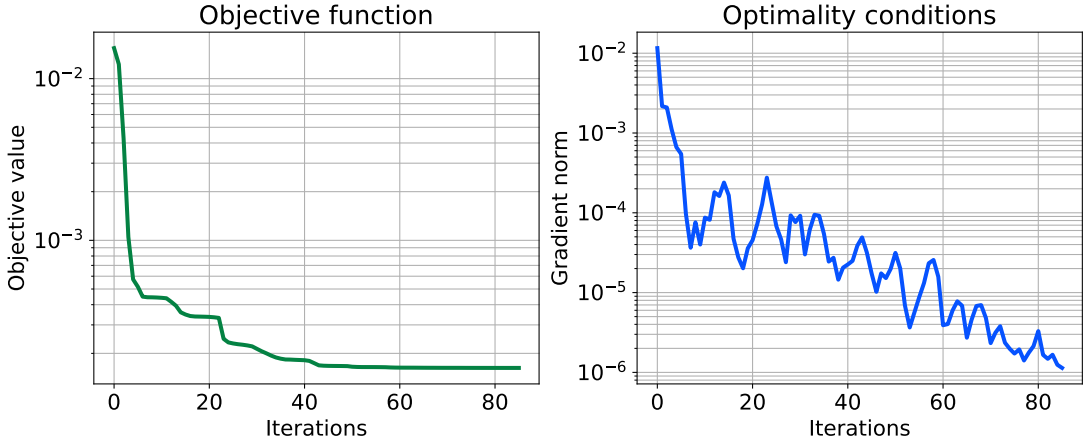
The beam result is encouraging, but our network-based optimization approach is far from being operational. In the next step of our investigations, we intend to perform nonlinear registration and net force estimation using a liver mesh.

7.3 Improve robustness with primal-dual PDE management

The whole numerical implementation presented in this manuscript revolves around an adjoint method, where the only variable seen by the optimization solver is the control $\mathbf{b}$. To converge up to a sufficiently tight tolerance, the optimization solver needs the adjoint method to return an objective value and gradient with high precision. In other



(a) Synthetic (left) and reconstructed deformation (right).



(b) Convergence statistics.

Figure 7.3: Results of the network-based optimization procedure.

words, the whole adjoint pipeline, including the direct solver, the functional evalutation procedure and the adjoint solver (see Figure 5.1) should return a result with an error as close as possible to machine precision. For this reason, the adjoint method is relevant when the elasticity problem is solved using a direct procedure, such as a direct linear solver for the linear case or the neural network in the nonlinear case. However, when the elastic problem is solved using an iterative method, expecting a result with a close-to-machine precision at each iteration is very costly and unrealistic at the same time. In particular, if a variation of the nodal forces is too small, the solver returns the previous displacement field as a good enough solution and the whole process fails because the objective function does not decrease. In this section, we discuss an alternative to the adjoint method where elasticity constraint is managed using Lagrange multipliers. We first mention two primal-dual algorithms, the Augmented Lagrangian method and the Sequential Quadratic Programming method, that are good candidates for our application. As both methods are often used in the context of second-order methods, we also discuss the feasibility of evaluating a numerical Hessian for the shape functional J.

Primal-dual solvers

In a primal-dual algorithm, $\mathbf{u}$ and $\mathbf{b}$ are both optimization variables in (5.3), while the adjoint state $\mathbf{p}$ is used as a Lagrange multiplier to manage the constraint $\mathbf{F}(\mathbf{u}) = \mathbf{b}$. The min-max problem associated to (5.3) reads

$$\min_{\mathbf{u},\mathbf{b}} \max_{\mathbf{p}} \; L(\mathbf{u},\mathbf{b},\mathbf{p}) \quad \text{where} \quad L(\mathbf{u},\mathbf{b},\mathbf{p}) = J(\mathbf{u}) + R(\mathbf{b}) - \mathbf{p}^{\mathrm{T}}(\mathbf{F}(\mathbf{u}) - \mathbf{b}).$$

During an iteration of the Augmented Lagrangian method (Hestenes, 1969), the primal and dual variables are updated alternatively. First, a minimization subproblem involving a variant of the Lagrangian is solved with a fixed Lagrange multiplier $\mathbf{p}$. The minimization subproblem reads

$$\min_{\mathbf{u},\mathbf{b}} J(\mathbf{u}) + R(\mathbf{b}) - \mathbf{p}^{\mathrm{T}}(\mathbf{F}(\mathbf{u}) - \mathbf{b}) + \tfrac{\mu}{2}\|\mathbf{F}(\mathbf{u}) - \mathbf{b}\|^2. \tag{7.7}$$

Compared to the standard Lagrangian, the augmented Lagrangian involves an additional penalty term to better enforce the constraint. In the second part of the iteration, the multiplier is updated using

$$\mathbf{p}_+ = \mathbf{p} - \mu(\mathbf{F}(\mathbf{u}) - \mathbf{b}),$$

so that, along iterations, the sequence of solutions to (7.7) converges toward the solution to (5.3). A simple summary of the Augmented Lagrangian iteration is given in Algorithm 5. The reader interested in more details about the tuning of parameters may consult Nocedal and Wright (2006, Chapter 17). The Augmented Lagrangian method is flexible, as the user can choose how to solve the minimization subproblem. In particular, constraints on $\mathbf{u}$ or $\mathbf{b}$ can either be handled by the inner subproblem solver, or managed by the augmented Lagrangian method by introducing additional Lagrange multipliers.

Algorithm 5: Augmented Lagrangian strategy. (Nocedal and Wright, 2006, Alg.17.3)

Data: Initial multiplier $\mathbf{p}_0$, initial penalty coefficient μ_0, constraint violation tolerance η
for *k = 0, 1, 2, ...* **do**
  Find a pair $(\mathbf{u}_k, \mathbf{b}_k)$ that minimizes $L(\mathbf{u}, \mathbf{b}, \mathbf{p}_0) + \frac{\mu_k}{2}\|\mathbf{F}(\mathbf{u}) - \mathbf{b}\|^2$
  if $\|\mathbf{F}(\mathbf{u}) - \mathbf{b}\| \leqslant \eta$ **then**
    return $(\mathbf{u}_k, \mathbf{b}_k)$
  end
  Adjust Lagrange multiplier: $\mathbf{p}_{k+1} = \mathbf{p}_k - \mu_k(\mathbf{F}(\mathbf{u}_k) - \mathbf{b}_k)$
  Choose a new penalty coefficient $\mu_{k+1} \geqslant \mu_k$
end

Another method for equality-constrained problems is the Sequential quadratic programming method (Nocedal and Wright, 2006, Chapter 18). The main idea of this algorithm is to apply the Newton method to solve the first-order optimality conditions

$$\begin{cases} \nabla J(\mathbf{u}) - \mathbf{F}'(\mathbf{u})^{\mathrm{T}}\mathbf{p} &= 0 \\ \nabla R(\mathbf{b}) + \mathbf{p} &= 0 \\ \mathbf{b} - \mathbf{F}(\mathbf{u}) &= 0. \end{cases}$$

Namely, the quadratic subproblem that is assembled at each iteration to compute a step $(\mathbf{w}, \mathbf{h})$ reads

$$\min_{\mathbf{w},\mathbf{h}} \quad \begin{pmatrix} \nabla J(\mathbf{u}) \\ \nabla R(\mathbf{b}) \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \mathbf{w} \\ \mathbf{h} \end{pmatrix} + \begin{pmatrix} \mathbf{w} \\ \mathbf{h} \end{pmatrix}^{\mathrm{T}} \begin{pmatrix} \nabla^2 J(\mathbf{u}) - \mathbf{F}''(\mathbf{u})^{\mathrm{T}}\mathbf{p} & \\ & \nabla^2 R(\mathbf{b}) \end{pmatrix} \begin{pmatrix} \mathbf{w} \\ \mathbf{h} \end{pmatrix}$$

$$\text{subject to constraint} \quad \mathbf{F}(\mathbf{u}) + \mathbf{F}'(\mathbf{u})\mathbf{w} = \mathbf{b} + \mathbf{h},$$

where $\nabla^2\mathbf{F}(\mathbf{u})^{\mathrm{T}}\mathbf{p} = \frac{\partial}{\partial \mathbf{u}}(\nabla\mathbf{F}(\mathbf{u})^{\mathrm{T}}\mathbf{p})$. Of course, sequential quadratic programming algorithms feature mechanisms to ensure convergence, such as a line search, trust-region or filter procedure. Though we do not give more details about sequential quadratic programming methods, let us add that sequential quadratic programming methods are usually expected to perform better on optimization problems with nonlinear constraints.

As they involve both $\mathbf{u}$ and $\mathbf{b}$ as optimization variables, primal-dual methods often lead to large problems and require to solve large linear systems. However, in our case, the objective function is splitted between a function of $\mathbf{u}$ and a function of $\mathbf{b}$, leading to sparse systems. This splitting may even be exploited by using methods such as the Alternating direction method of multipliers (see Ouyang et al., 2015, and references therein). Finally, primal dual methods might be interesting if computations are sped up using reduced bases. In that case, an iterative procedure would still be necessary to perform the optimization, but the optimization problem would keep a reasonable size.

Computing a Hessian for J

In primal-dual methods, the displacement field $\mathbf{u}$ and the functional J are manipulated directly by the optimization solver instead of being hidden in the depths of the adjoint procedure. As a consequence, it makes sense to consider second-order information for J. Sequential Quadratic Programming methods require to solve a quadratic subproblem involving the Hessian matrix $\nabla^2 J(\mathbf{u})$. Here, using a BFGS approximation is tricky, as it would deteriorate the sparsity structure of the system matrix. Concerning the Augmented Lagrangian method, solving the minimization subproblem with an increasing accuracy using a first-order or quasi-newton method would result in very long iterations. In both case, being able to evaluate a numerical Hessian for J seems profitable.

The strategy to compute numerically the Hessian of J is the same as the one to evaluate the gradient of J. For a given point $y \in \Gamma$, the projection point $p_{\partial\Omega_{\mathbf{u}}}(y)$ lies either on a vertex, inside an edge or inside a triangle of the deformed mesh. In addition, in the majority of cases, when a small perturbation $\mathbf{v}$ is applied to $\mathbf{u}$, the projection point $p_{\partial\Omega_{\mathbf{u}+\mathbf{v}}}(y)$ is expected to remain on the same element. As a consequence, the elementary function j_y is locally equal to the squared distance between y and a point, a line or a plane.

Let us take an example based on Figure 5.2. On the left, j_y is locally equal to $\mathbf{u} \mapsto \frac{1}{2}\|y - x_k\|^2$, and the only nonzero term in the Hessian matrix $\nabla^2 j_y(\mathbf{u})$ is the diagonal block $\partial^2_{kk} j_y(\mathbf{u}) = I$ corresponding to x_k. On the right, however, j_y is locally equal to $\frac{1}{2}d^2(y,(x_k x_m))$, and the Hessian matrix should contain 4 nonzero blocks, at the lines and columns corresponding to x_k and x_m.

To compute those Hessian coefficients, we use the change of variable

$$v = \frac{x_k + x_m}{2} - y \qquad \text{and} \qquad u = \frac{x_k - x_m}{2},$$

and we define the function

$$f(u,v) = \tfrac{1}{2}d^2(y,(x_k x_m)) = \tfrac{1}{2}\left(\|v\|^2 - \theta^2\|u\|^2\right), \quad \text{where} \quad \theta = \theta(u,v) = -\frac{\langle u, v\rangle}{\|u\|^2}.$$

The expression for θ is obtained by minimizing $\|v + \theta u\|$, while the expression for $f(u,v)$ is obtained by noticing that $v + \theta u$ and θu are orthogonal and applying the Pythagorean theorem (see notations in Figure 7.4). Here, θ depends on u and v and its derivatives read

$$\frac{\partial\theta}{\partial v}(u,v) = -\frac{u}{\|u\|^2} \qquad \frac{\partial\theta}{\partial u}(u,v) = -\frac{v + 2\theta u}{\|u\|^2}.$$

It can be verified that the first-order derivatives of f,

$$\frac{\partial f}{\partial v}(u,v) = v + \theta u \qquad \frac{\partial f}{\partial u}(u,v) = \theta(v + \theta u),$$

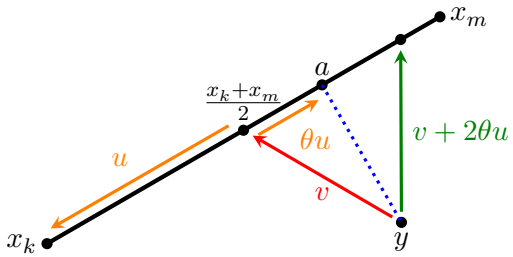


Figure 7.4: Notations for computing the Hessian of j_y.

are consistent with the expressions obtained in Section 5.3. After a new round of differentiation, the second-order derivatives of f read

$$\begin{pmatrix} \partial^2_{vv} f & \partial^2_{uv} f \\ \partial^2_{vu} f & \partial^2_{uu} f \end{pmatrix} (u, v) = \begin{pmatrix} I \\ \theta I \end{pmatrix} \begin{pmatrix} I \\ \theta I \end{pmatrix}^{\mathrm{T}} - \frac{1}{\|u\|^2} \begin{pmatrix} u \\ v + 2\theta u \end{pmatrix} \begin{pmatrix} u \\ v + 2\theta u \end{pmatrix}^{\mathrm{T}},$$

and by applying the change of variable, we obtain the second derivatives of j_y

$$\begin{aligned} \begin{pmatrix} \partial^2_{kk} j_y & \partial^2_{mk} j_y \\ \partial^2_{km} j_y & \partial^2_{mm} j_y \end{pmatrix} (\mathbf{u}) &= \begin{pmatrix} \theta_k I \\ \theta_m I \end{pmatrix} \begin{pmatrix} \theta_k I \\ \theta_m I \end{pmatrix}^{\mathrm{T}} \\ &\quad - \frac{1}{\|x_k - x_m\|^2} \begin{pmatrix} (a-y) + \theta_k (x_k - x_m) \\ -(a-y) + \theta_m (x_k - x_m) \end{pmatrix} \begin{pmatrix} (a-y) + \theta_k (x_k - x_m) \\ -(a-y) + \theta_m (x_k - x_m) \end{pmatrix}^{\mathrm{T}}, \end{aligned}$$

where $\theta_k = (1+\theta)/2$ and $\theta_m = (1-\theta)/2$ are the barycentric coefficients of the projection points in the element $[x_k, x_m]$.

From this expression, it is difficult to obtain visual insight about j_y, but it shows the feasibility of computing a Hessian matrix. In the case of a triangle element, computing second-order derivatives is much more involved, as our change of variable is not applicable anymore. However, a numerical implementation should take advantage of automatic differentiation to compute Hessian coefficients.

It might seem strange to compute the second-order derivatives of a function which is not Gateaux differentiable everywhere. However, as the optimization method converges, displacement steps get smaller and pairings between the mesh boundary elements and observed points are not expected to change too much. On the other hand, second-order information can be helpful in the first iterations of the optimization solver, to avoid taking too large steps.

7.4 Conclusion

In this manuscript, we presented a registration procedure based on an optimal control formulation of the registration problem. We not only proposed a structure for the implementation of a registration method, but also some mathematical insight that might be

of use to understand the behavior of the discretized procedure. Of course, our approach is far from being mature, and many other choices could have been made, such as a more clever functional J, a more accurate physical model or more efficient numerical methods. However, new mathematically sound approaches in augmented surgery might be derived by adapting tools from the generic optimal control framework. Proposing open and easily adaptable procedures should help democratize computer-assisted surgery, letting more people around the world benefit from safer care and go back home from the hospital in good health.

Bibliography

Adams, Robert A. and John J. F. Fournier (2003). *Sobolev spaces*. Second. Vol. 140. Pure and Applied Mathematics (Amsterdam). Elsevier/Academic Press, Amsterdam, pp. xiv+305. ISBN: 0-12-044143-8.

Allaire, Grégoire (2007). *Conception optimale de structures*. Vol. 58. Mathématiques & Applications (Berlin) [Mathematics & Applications]. Springer-Verlag, Berlin, pp. xii+278. ISBN: 978-3-540-36710-9; 3-540-36710-1.

Allard, Jérémie, Stéphane Cotin, François Faure, Pierre-Jean Bensoussan, François Poyer, Christian Duriez, Hervé Delingette, and Laurent Grisoni (2007). "SOFA - an Open Source Framework for Medical Simulation". In: *MMVR 15 - Medicine Meets Virtual Reality*. Vol. 125. Studies in Health Technology and Informatics. Palm Beach, United States: IOP Press, pp. 13–18.

Antonsanti, Pierre-Louis, Joan Glaunès, Thomas Benseghir, Vincent Jugnon, and Irène Kaltenmark (2021). "Partial Matching in the Space of Varifolds". In: *Information Processing in Medical Imaging*. Ed. by Aasa Feragen, Stefan Sommer, Julia Schnabel, and Mads Nielsen. Cham: Springer International Publishing, pp. 123–135. ISBN: 978-3-030-78191-0.

Armijo, Larry (1966). "Minimization of functions having Lipschitz continuous first partial derivatives". In: *Pacific J. Math.* 16, pp. 1–3. ISSN: 0030-8730.

Bajcsy, Ruzena and Stane Kovačič (1989). "Multiresolution elastic matching". In: *Computer Vision, Graphics, and Image Processing* 46.1, pp. 1 –21. ISSN: 0734-189X. DOI: 10.1016/S0734-189X(89)80014-3.

Ball, John M. (1976). "Convexity conditions and existence theorems in nonlinear elasticity". In: *Archive for Rational Mechanics and Analysis* 63.4, pp. 337–403. ISSN: 1432-0673. DOI: 10.1007/BF00279992.

Beckmann, Norbert, Hans-Peter Kriegel, Ralf Schneider, and Bernhard Seeger (1990). "The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles". In: *SIGMOD Rec.* 19.2, 322–331. ISSN: 0163-5808. DOI: 10.1145/93605.98741.

Beg, M. Faisal, Michael I. Miller, Alain Trouvé, and Laurent Younes (2005). "Computing Large Deformation Metric Mappings via Geodesic Flows of Diffeomorphisms". In: *International Journal of Computer Vision* 61.2, pp. 139–157. DOI: 10.1023/B:VISI.0000043755.93987.aa.

Bertsekas, Dimitri P. (1976). "On the Goldstein-Levitin-Polyak gradient projection method". In: *IEEE Trans. Automatic Control* AC-21.2, pp. 174–184. ISSN: 0018-9286. DOI: 10.1109/tac.1976.1101194.

— (1982). "Projected Newton methods for optimization problems with simple constraints". In: *SIAM J. Control Optim.* 20.2, pp. 221–246. ISSN: 0363-0129. DOI: 10.1137/0320018.

Besl, Paul J. and Neil D. McKay (1992). "Method for registration of 3-D shapes". In: *Sensor Fusion IV: Control Paradigms and Data Structures*. Ed. by Paul S. Schenker. Vol. 1611. International Society for Optics and Photonics. SPIE, pp. 586 –606. DOI: 10.1117/12.57955.

Brewer, E. Lee, Logan W. Clements, Jarrod A. Collins, Derek J. Doss, Jon S. Heiselman, Michael I. Miga, Chris D. Pavas, and Edward H. Wisdom III (2019). "The image-to-physical liver registration sparse data challenge". In: *Medical Imaging 2019: Image-Guided Procedures, Robotic Interventions, and Modeling*. Ed. by Baowei Fei and Cristian A. Linte. Vol. 10951. International Society for Optics and Photonics. SPIE, pp. 364 –370. DOI: 10.1117/12.2513952.

Brezis, Haïm (1983). *Analyse fonctionnelle*. Collection Mathématiques Appliquées pour la Maîtrise. [Collection of Applied Mathematics for the Master's Degree]. Théorie et applications. [Theory and applications]. Masson, Paris, pp. xiv+234. ISBN: 2-225-77198-7.

Broit, Chaim (1981). "Optimal registration of deformed images". PhD thesis. University of Pennsylvania.

Brunet, Jean-Nicolas (2020). "Exploring new numerical methods for the simulation of soft tissue deformations in surgery assistance". Theses. Université de Strasbourg.

Brunet, Jean-Nicolas, Vincent Magnoux, Benoît Ozell, and Stéphane Cotin (2019). "Corotated meshless implicit dynamics for deformable bodies". In: *WSCG 2019 - 27th International Conference on Computer Graphics, Visualization and Computer Vision*. Pilsen, Czech Republic: Západočeská univerzita. DOI: 10.24132/CSRN.2019.2901.1.11.

de Buhan, Maya, Charles Dapogny, Pascal Frey, and Chiara Nardoni (2016). "An optimization method for elastic shape matching". In: *C. R. Math. Acad. Sci. Paris* 354.8, pp. 783–787. ISSN: 1631-073X. DOI: 10.1016/j.crma.2016.05.007.

Byrd, Richard H., Peihuang Lu, Jorge Nocedal, and Ciyou Zhu (1995). "A Limited Memory Algorithm for Bound Constrained Optimization". In: *SIAM Journal on Scientific Computing* 16.5, pp. 1190–1208. DOI: 10.1137/0916069.

Christensen, G. E., S. C. Joshi, and M. I. Miller (1997). "Volumetric transformation of brain anatomy". In: *IEEE Transactions on Medical Imaging* 16.6, pp. 864–877. ISSN: 1558-254X. DOI: 10.1109/42.650882.

Christensen, G. E., R. D. Rabbitt, and M. I. Miller (1996). "Deformable templates using large deformation kinematics". In: *IEEE Transactions on Image Processing* 5.10, pp. 1435–1447. ISSN: 1941-0042. DOI: 10.1109/83.536892.

Ciarlet, Philippe G. (1988). *Mathematical elasticity. Vol. I*. Vol. 20. Studies in Mathematics and its Applications. Three-dimensional elasticity. North-Holland Publishing Co., Amsterdam, pp. xlii+451. ISBN: 0-444-70259-8.

Clements, Logan W., William C. Chapman, Benoit M. Dawant, Robert L. Galloway Jr., and Michael I. Miga (2008). "Robust surface registration using salient anatomical features for image-guided liver surgery: Algorithm and validation". In: *Medical Physics* 35.6Part1, pp. 2528–2540. DOI: 10.1118/1.2911920.

Collins, Jarrod A., Jared A. Weis, Jon S. Heiselman, Logan W. Clements, Amber L. Simpson, William R. Jarnagin, and Michael I. Miga (2017). "Improving Registration Robustness for Image-Guided Liver Surgery in a Novel Human-to-Phantom Data Framework". In: *IEEE Transactions on Medical Imaging* 36.7, pp. 1502–1510. DOI: 10.1109/TMI.2017.2668842.

Conn, Andrew R., Nicholas I. M. Gould, and Philippe L. Toint (2000). *Trust-region methods*. MPS-SIAM series on optimization. Society for Industrial and Applied Mathematics. ISBN: 978-0-89871-460-9. DOI: 10.1137/1.9780898719857.

Courtecuisse, H., Z. Jiang, O. Mayeur, J. F. Witz, P. Lecomte-Grosbras, M. Cosson, M. Brieu, and S. Cotin (2020). "Three-dimensional physics-based registration of pelvic system using 2D dynamic magnetic resonance imaging slices". In: *Strain* 56.3, e12339. DOI: 10.1111/str.12339.

D'Agostino, Emiliano, Frederik Maes, Dirk Vandermeulen, and Paul Suetens (2003). "A viscous fluid model for multimodal non-rigid image registration using mutual information". In: *Med-*

ical Image Analysis 7.4. Medical Image Computing and Computer Assisted Intervention, pp. 565 –575. ISSN: 1361-8415. DOI: 10.1016/S1361-8415(03)00039-2.

Dahito, Marie-Ange and Dominique Orban (2019). "The Conjugate Residual Method in Linesearch and Trust-Region Methods". In: *SIAM Journal on Optimization* 29.3, pp. 1988–2025. DOI: 10.1137/18M1204255.

Danskin, John M. (1967). *The theory of max-min and its application to weapons allocation problems*. Econometrics and Operations Research, Vol. V. Springer-Verlag New York, Inc., New York, pp. ix+126.

Dapogny, Charles (2013). "Shape optimization, level set methods on unstructured meshes and mesh evolution". 2013PA066498. PhD thesis. Paris 6, 1 vol. (402 p.)

Dapogny, Charles and Pascal Frey (2012). "Computation of the signed distance function to a discrete contour on adapted triangulation". In: *Calcolo* 49.3, pp. 193–219. ISSN: 0008-0624. DOI: 10.1007/s10092-011-0051-z.

Dapogny, Charles, Pascal Frey, Florian Omnès, and Yannick Privat (2018). "Geometrical shape optimization in fluid mechanics using FreeFem++". In: *Struct. Multidiscip. Optim.* 58.6, pp. 2761–2788. ISSN: 1615-147X. DOI: 10.1007/s00158-018-2023-2.

Davis, Timothy A., Sivasankaran Rajamanickam, and Wissam M. Sid-Lakhdar (2016). "A survey of direct methods for sparse linear systems". In: *Acta Numerica* 25, 383–566. DOI: 10.1017/S0962492916000076.

Delfour, M. C. and J.-P. Zolésio (2011). *Shapes and geometries*. Second. Vol. 22. Advances in Design and Control. Metrics, analysis, differential calculus, and optimization. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, pp. xxiv+622. ISBN: 978-0-898719-36-9. DOI: 10.1137/1.9780898719826.

Delingette, Hervé and Nicholas Ayache (2004). "Soft Tissue Modeling for Surgery Simulation". In: *Computational Models for the Human Body*. Vol. 12. Handbook of Numerical Analysis. Elsevier, pp. 453–550. DOI: 10.1016/S1570-8659(03)12005-4.

Doğan, G., P. Morin, R. H. Nochetto, and M. Verani (2007). "Discrete gradient flows for shape optimization and applications". In: *Comput. Methods Appl. Mech. Engrg.* 196.37-40, pp. 3898–3914. ISSN: 0045-7825. DOI: 10.1016/j.cma.2006.10.046.

Droniou, Jérôme (2000). "Solving convection-diffusion equations with mixed, Neumann and Fourier boundary conditions and measures as data, by a duality method". In: *Adv. Differential Equations* 5.10-12, pp. 1341–1396. ISSN: 1079-9389.

Dupuis, Paul, Ulf Grenander, and Michael I. Miller (1998). "Variational problems on flows of diffeomorphisms for image matching". In: *Quart. Appl. Math.* 56.3, pp. 587–600. ISSN: 0033-569X. DOI: 10.1090/qam/1632326.

Evans, Lawrence C. (2010). *Partial differential equations*. Second. Vol. 19. Graduate Studies in Mathematics. American Mathematical Society, Providence, RI, pp. xxii+749. ISBN: 978-0-8218-4974-3. DOI: 10.1090/gsm/019.

Evans, Lawrence C. and Ronald F. Gariepy (2015). *Measure theory and fine properties of functions*. Revised. Textbooks in Mathematics. CRC Press, Boca Raton, FL, pp. xiv+299. ISBN: 978-1-4822-4238-6.

García R, Yván J., Mario A. López, and Scott T. Leutenegger (1998). "A Greedy Algorithm for Bulk Loading R-Trees". In: *Proceedings of the 6th ACM International Symposium on Advances in Geographic Information Systems*. GIS '98. Washington, D.C., USA: Association for Computing Machinery, 163–164. ISBN: 1581131151. DOI: 10.1145/288692.288723.

Gee, James C and Ruzena K Bajcsy (1999). "Elastic matching: Continuum mechanical and probabilistic analysis". In: *Brain warping* 2, pp. 183–197.

Gelin, J. C. and P. Picart (1988). "Use of quasi-Newton methods for large strain elastic-plastic finite element computations". In: *Communications in Applied Numerical Methods* 4.4, pp. 457–469. DOI: 10.1002/cnm.1630040402.

Gilbarg, David and Neil S. Trudinger (1977). *Elliptic partial differential equations of second order*. Grundlehren der Mathematischen Wissenschaften, Vol. 224. Springer-Verlag, Berlin-New York, pp. x+401. ISBN: 3-540-08007-4.

Gray, Henry and Warren H Lewis (1918). *Anatomy of the human body*. Lea and Febiger, Philadelphia, p. 1396.

Grisvard, Pierre (2011). *Elliptic problems in nonsmooth domains*. Vol. 69. Classics in Applied Mathematics. Reprint of the 1985 original [MR0775683], With a foreword by Susanne C. Brenner. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, pp. xx+410. ISBN: 978-1-611972-02-3. DOI: 10.1137/1.9781611972030.ch1.

Gröger, Konrad (1989). "A $W^{1,p}$-estimate for solutions to mixed boundary value problems for second order elliptic differential equations". In: *Math. Ann.* 283.4, pp. 679–687. ISSN: 0025-5831. DOI: 10.1007/BF01442860.

Gröger, Konrad and Joachim Rehberg (1989). "Resolvent estimates in $W^{-1,p}$ for second order elliptic differential operators in case of mixed boundary conditions". In: *Math. Ann.* 285.1, pp. 105–113. ISSN: 0025-5831. DOI: 10.1007/BF01442675.

Gross, Christian and Rolf Krause (2009). "On the Convergence of Recursive Trust-Region Methods for Multiscale Nonlinear Optimization and Applications to Nonlinear Mechanics". In: *SIAM Journal on Numerical Analysis* 47.4, pp. 3044–3069. DOI: 10.1137/08071819X.

Guttman, Antonin (1984). "R-Trees: A Dynamic Index Structure for Spatial Searching". In: *SIGMOD Rec.* 14.2, 47–57. ISSN: 0163-5808. DOI: 10.1145/971697.602266.

Haller-Dintelmann, R., C. Meyer, J. Rehberg, and A. Schiela (2009). "Hölder continuity and optimal control for nonsmooth elliptic problems". In: *Appl. Math. Optim.* 60.3, pp. 397–428. ISSN: 0095-4616. DOI: 10.1007/s00245-009-9077-x.

Haouchine, N., J. Dequidt, I. Peterlík, E. Kerrien, M. Berger, and S. Cotin (2013). "Image-guided simulation of heterogeneous tissue deformation for augmented reality during hepatic surgery". In: *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 199–208. DOI: 10.1109/ISMAR.2013.6671780.

Haouchine, Nazim, Winnie Kuang, Stephane Cotin, and Michael Yip (2018). "Vision-Based Force Feedback Estimation for Robot-Assisted Surgery Using Instrument-Constrained Biomechanical Three-Dimensional Maps". In: *IEEE Robotics and Automation Letters* 3.3, pp. 2160–2165. DOI: 10.1109/LRA.2018.2810948.

Hecht, F. (2012). "New development in FreeFem++". In: *J. Numer. Math.* 20.3-4, pp. 251–265. ISSN: 1570-2820.

Heiselman, Jon S., Logan W. Clements, Jarrod A. Collins, Jared A. Weis, Amber L. Simpson, Sunil K. Geevarghese, T. Peter Kingham, William R. Jarnagin, and Michael I. Miga (2017). "Characterization and correction of intraoperative soft tissue deformation in image-guided laparoscopic liver surgery". In: *Journal of Medical Imaging* 5.2, pp. 1 –12. DOI: 10.1117/1.JMI.5.2.021203.

Heiselman, Jon S., William R. Jarnagin, and Michael I. Miga (2020). "Intraoperative Correction of Liver Deformation Using Sparse Surface and Vascular Features via Linearized Iterative Boundary Reconstruction". In: *IEEE Transactions on Medical Imaging* 39.6, pp. 2223–2234. DOI: 10.1109/TMI.2020.2967322.

Henrot, Antoine and Michel Pierre (2005). *Variation et optimisation de formes*. Vol. 48. Mathématiques & Applications (Berlin). Une analyse géométrique. Springer, Berlin, pp. xii+334. ISBN: 978-3-540-26211-4; 3-540-26211-3. DOI: 10.1007/3-540-37689-5.

Herzog, Roland, Christian Meyer, and Gerd Wachsmuth (2011). "Integrability of displacement and stresses in linear and nonlinear elasticity with mixed boundary conditions". In: *J. Math. Anal. Appl.* 382.2, pp. 802–813. ISSN: 0022-247X. DOI: 10.1016/j.jmaa.2011.04.074.

Hestenes, Magnus R. (1969). "Multiplier and gradient methods". In: *J. Optim. Theory Appl.* 4, pp. 303–320. ISSN: 0022-3239. DOI: 10.1007/BF00927673.

Hestenes, Magnus R and Eduard Stiefel (1952). "Methods of Conjugate Gradients for Solving Linear Systems". In: *Journal of Research of the National Bureau of Standards* 49.6.

James, Doug L. and Dinesh K. Pai (2004). "BD-Tree: Output-Sensitive Collision Detection for Reduced Deformable Models". In: *ACM SIGGRAPH 2004 Papers*. SIGGRAPH '04. Los Angeles, California: Association for Computing Machinery, 393–398. ISBN: 9781450378239. DOI: 10.1145/1186562.1015735.

Kelley, Carl T (1995). *Iterative methods for linear and nonlinear equations*. Frontiers in Applied Mathematics. SIAM, pp. xiii+156. ISBN: 978-0-89871-352-7.

Kirchhoff, Gustav Robert (1852). "Über die Gleichungen des Gleichgewichtes eines elastischen Körpers bei nicht unendlich kleinen Verschiebungen seiner Theile". In: *Sitzungsberichte der Mathematisch-Naturwissenschaftlichen Classe der Kaiserlichen Akademie der Wissenschaften in Wien IX* 9, pp. 762–773.

Kraft, Dieter (1988). *A software package for sequential quadratic programming*. Tech. rep. DFVLR-FB 88-28. Köln, Germany: DLR German Aerospace Center – Institute for Flight Mechanics.

Leutenegger, S.T., M.A. Lopez, and J. Edgington (1997). "STR: a simple and efficient algorithm for R-tree packing". In: *Proceedings 13th International Conference on Data Engineering*, pp. 497–506. DOI: 10.1109/ICDE.1997.582015.

Lin, Chih-Jen and Jorge J. Moré (1999). "Newton's method for large bound-constrained optimization problems". In: vol. 9. 4. Dedicated to John E. Dennis, Jr., on his 60th birthday, pp. 1100–1127. DOI: 10.1137/S1052623498345075.

Liu, Tiantian, Sofien Bouaziz, and Ladislav Kavan (2017). "Quasi-Newton Methods for Real-Time Simulation of Hyperelastic Materials". In: *ACM Trans. Graph.* 36.3. ISSN: 0730-0301. DOI: 10.1145/2990496.

Logg, Anders and Garth N. Wells (2010). "DOLFIN: Automated Finite Element Computing". In: *ACM Trans. Math. Softw.* 37.2. ISSN: 0098-3500. DOI: 10.1145/1731022.1731030.

Logg, Anders, Garth N. Wells, and Johan Hake (2012). "DOLFIN: a C++/Python finite element library". In: *Automated Solution of Differential Equations by the Finite Element Method: The FEniCS Book*. Ed. by Anders Logg, Kent-Andre Mardal, and Garth Wells. Springer Berlin Heidelberg, pp. 173–225. ISBN: 978-3-642-23099-8. DOI: 10.1007/978-3-642-23099-8_10.

Marbán, Arturo, Alicia Casals, Josep Fernández, and Josep Amat (2014). "Haptic Feedback in Surgical Robotics: Still a Challenge". In: *ROBOT2013: First Iberian Robotics Conference: Advances in Robotics, Vol. 1*. Ed. by Manuel A. Armada, Alberto Sanfeliu, and Manuel Ferre. Cham: Springer International Publishing, pp. 245–253. ISBN: 978-3-319-03413-3. DOI: 10.1007/978-3-319-03413-3_18.

Marchesseau, Stéphanie, Simon Chatelin, and Hervé Delingette (2017). "Nonlinear Biomechanical Model of the Liver". In: *Biomechanics of Living Organs*. Ed. by Yohan Payan and Jacques Ohayon. Vol. 1. Translational Epigenetics. Oxford: Academic Press, pp. 243–265. DOI: 10.1016/B978-0-12-804009-6.00011-0.

Mendizabal, Andrea, Eleonora Tagliabue, Tristan Hoellinger, Jean-Nicolas Brunet, Sergei Nikolaev, and Stéphane Cotin (2020). "Data-driven simulation for augmented surgery". In: *Developments and Novel Approaches in Biomechanics and Metamaterials*. Ed. by Bilen Emek Abali and Ivan Giorgio. Vol. 132, pp. 71–96. DOI: 10.1007/978-3-030-50464-9.

Mestdagh, Guillaume and Stéphane Cotin (2022). "An Optimal Control Problem for Elastic Registration and Force Estimation in Augmented Surgery". In: *Medical Image Computing and*

Computer Assisted Intervention – MICCAI 2022. Ed. by Linwei Wang, Qi Dou, P. Thomas Fletcher, Stefanie Speidel, and Shuo Li. Cham: Springer Nature Switzerland, pp. 74–83. ISBN: 978-3-031-16449-1. DOI: 10.1007/978-3-031-16449-1_8.

Miller, Karol, Grand Joldes, Dane Lance, and Adam Wittek (2007). "Total Lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation". In: *Communications in Numerical Methods in Engineering* 23.2, pp. 121–134. DOI: 10.1002/cnm.887.

Morch, Hélène, Sibo Yuan, Laurent Duchêne, Ridha Harzallah, and Anne Marie Habraken (2022). "A review of higher order Newton type methods and the effect of numerical damping for the solution of an advanced coupled Lemaitre damage model". In: *Finite Elements in Analysis and Design* 209, p. 103801. ISSN: 0168-874X. DOI: 10.1016/j.finel.2022.103801.

Morin, Fanny, Hadrien Courtecuisse, Ingerid Reinertsen, Florian Le Lann, Olivier Palombi, Yohan Payan, and Matthieu Chabanas (2017). "Brain-shift compensation using intraoperative ultrasound and constraint-based biomechanical simulation". In: *Medical image analysis* 40, pp. 133–153. DOI: 10.1016/j.media.2017.06.003.

Murat, François and Jacques Simon (1976). *Sur le contrôle par un domaine géométrique*. Tech. rep. RR-76015. Laboratoire d'Analyse Numérique.

Nardoni, Chiara (2017). "Mesh deformation strategies in shape optimization. Application to forensic facial reconstruction". Theses. Université Pierre et Marie Curie - Paris VI.

Nazari, Ali A., Farrokh Janabi-Sharifi, and Kourosh Zareinia (2021). "Image-Based Force Estimation in Medical Applications: A Review". In: *IEEE Sensors Journal* 21.7, pp. 8805–8830. DOI: 10.1109/JSEN.2021.3052755.

Nesme, Matthieu, Yohan Payan, and François Faure (2005). "Efficient, Physically Plausible Finite Elements". In: *Eurographics*. Short papers. Dublin, Ireland.

Nesterov, Yu. E. (1983). "A method for solving the convex programming problem with convergence rate $O(1/k^2)$". In: *Dokl. Akad. Nauk SSSR* 269.3, pp. 543–547. ISSN: 0002-3264.

Niehues, S. M., J. K. Unger, M. Malinowski, J. Neymeyer, B. Hamm, and M. Stockmann (2010). "Liver volume measurement: reason of the difference between in vivo CT-volumetry and intraoperative ex vivo determination and how to cope it". In: *European Journal of Medical Research* 15.8, p. 345. ISSN: 2047-783X. DOI: 10.1186/2047-783X-15-8-345.

Nikolaev, Sergei and Stéphane Cotin (2020). "Estimation of boundary conditions for patient-specific liver simulation during augmented surgery". In: *International Journal of Computer Assisted Radiology and Surgery* 15.7, pp. 1107–1115. DOI: 10.1007/s11548-020-02188-x.

Nocedal, Jorge and Stephen J. Wright (2006). *Numerical optimization*. Second. Springer Series in Operations Research and Financial Engineering. Springer, New York, pp. xxii+664. ISBN: 978-0387-30303-1; 0-387-30303-0.

Odot, Alban, Ryadh Haferssas, and Stephane Cotin (2022). "DeepPhysics: A physics aware deep learning framework for real-time simulation". In: *International Journal for Numerical Methods in Engineering* 123.10, pp. 2381–2398. DOI: https://doi.org/10.1002/nme.6943.

Oudry, Jennifer, Ted Lynch, Jonathan Vappou, Laurent Sandrin, and Véronique Miette (2014). "Comparison of four different techniques to evaluate the elastic properties of phantom in elastography: is there a gold standard?" In: *Physics in Medicine and Biology* 59.19, pp. 5775–5793. DOI: 10.1088/0031-9155/59/19/5775.

Ouyang, Yuyuan, Yunmei Chen, Guanghui Lan, and Eduardo Pasiliao Jr. (2015). "An accelerated linearized alternating direction method of multipliers". In: *SIAM J. Imaging Sci.* 8.1, pp. 644–681. DOI: 10.1137/14095697X.

Ovsjanikov, Maks, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas (2012). "Functional Maps: A Flexible Representation of Maps between Shapes". In: *ACM Trans. Graph.* 31.4. ISSN: 0730-0301. DOI: 10.1145/2185520.2185526.

Özgür, Erol, Bongjin Koo, Bertrand Le Roy, Emmanuel Buc, and Adrien Bartoli (2018). "Pre-operative liver registration for augmented monocular laparoscopy using backward–forward biomechanical simulation". In: *International Journal of Computer Assisted Radiology and Surgery* 13.10, pp. 1629–1640. ISSN: 1861-6429. DOI: 10.1007/s11548-018-1842-3.

Ozkan, Ece and Orcun Goksel (2018). "Compliance boundary conditions for patient-specific deformation simulation using the finite element method". In: *Biomedical Physics & Engineering Express* 4.2, p. 025003. DOI: 10.1088/2057-1976/aa918d.

Peterlík, Igor, Hadrien Courtecuisse, Christian Duriez, and Stéphane Cotin (2014). "Model-Based Identification of Anatomical Boundary Conditions in Living Tissues". In: *IPCAI 2014 - 5th International Conference on Information Processing in Computer Assisted Interventions*. Fukuoka, Japan. DOI: 10.1007/978-3-319-07521-1_21.

Peterlík, Igor, Hadrien Courtecuisse, Robert Rohling, Purang Abolmaesumi, Christopher Nguan, Stéphane Cotin, and Septimiu Salcudean (2018). "Fast elastic registration of soft tissues under large deformations". In: *Medical Image Analysis* 45, pp. 24–40. ISSN: 1361-8415. DOI: 10.1016/j.media.2017.12.006.

Peterlík, Igor, Christian Duriez, and Stéphane Cotin (2012). "Modeling and Real-Time Simulation of a Vascularized Liver Tissue". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2012*. Ed. by Nicholas Ayache, Hervé Delingette, Polina Golland, and Kensaku Mori. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 50–57. ISBN: 978-3-642-33415-3. DOI: 10.1007/978-3-642-33415-3_7.

Peterlík, Igor, Nazim Haouchine, Lukáš Ručka, and Stéphane Cotin (2017). "Image-Driven Stochastic Identification of Boundary Conditions for Predictive Simulation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2017*. Ed. by Maxime Descoteaux, Lena Maier-Hein, Alfred Franz, Pierre Jannin, D. Louis Collins, and Simon Duchesne. Cham: Springer International Publishing, pp. 548–556. ISBN: 978-3-319-66185-8. DOI: 10.1007/978-3-319-66185-8_62.

Plantefève, Rosalie, Igor Peterlík, Nazim Haouchine, and Stéphane Cotin (2016). "Patient-Specific Biomechanical Modeling for Guidance During Minimally-Invasive Hepatic Surgery". In: *Annals of Biomedical Engineering* 44.1, pp. 139–153. ISSN: 1573-9686. DOI: 10.1007/s10439-015-1419-z.

Prasad, Sushil K., Michael McDermott, Xi He, and Satish Puri (2015). "GPU-based Parallel R-tree Construction and Querying". In: *2015 IEEE International Parallel and Distributed Processing Symposium Workshop*, pp. 618–627. DOI: 10.1109/IPDPSW.2015.127.

Rabbitt, Richard D., Jeffrey A. Weiss, Gary E. Christensen, and Michael I. Miller (1995). "Mapping of hyperelastic deformable templates using the finite element method". In: *Vision Geometry IV*. Ed. by Robert A. Melter, Angela Y. Wu, Fred L. Bookstein, and William D. K. Green. Vol. 2573. International Society for Optics and Photonics. SPIE, pp. 252 –265. DOI: 10.1117/12.216419.

Raoult, Annie (1986). "Non-polyconvexity of the stored energy function of a Saint Venant-Kirchhoff material". eng. In: *Aplikace matematiky* 31.6, pp. 417–419.

Rivlin, R. S. and Eric Keightley Rideal (1948). "Large elastic deformations of isotropic materials IV. further developments of the general theory". In: *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences* 241.835, pp. 379–397. DOI: 10.1098/rsta.1948.0024.

Rockafellar, R. Tyrrell (1970). *Convex Analysis*. Princeton University Press. ISBN: 9780691015866.

Rucker, D. Caleb, Yifei Wu, Logan W. Clements, Janet E. Ondrake, Thomas S. Pheiffer, Amber L. Simpson, William R. Jarnagin, and Michael I. Miga (2014). "A Mechanics-Based Nonrigid Registration Method for Liver Surgery Using Sparse Intraoperative Data". In: *IEEE Transactions on Medical Imaging* 33.1, pp. 147–158. DOI: 10.1109/TMI.2013.2283016.

Saad, Youcef and Martin H. Schultz (1986). "GMRES: A Generalized Minimal Residual Algorithm for Solving Nonsymmetric Linear Systems". In: *SIAM Journal on Scientific and Statistical Computing* 7.3, pp. 856–869. DOI: 10.1137/0907058.

Saad, Yousef (2003). *Iterative methods for sparse linear systems*. SIAM, pp. xvii+520. ISBN: 978-0-89871-534-7.

Sahillioğlu, Yusuf (2020). "Recent advances in shape correspondence". In: *The Visual Computer* 36.8, pp. 1705–1721. ISSN: 1432-2315. DOI: 10.1007/s00371-019-01760-0.

Shi, Peter and Steve Wright (1994). "Higher integrability of the gradient in linear elasticity". In: *Math. Ann.* 299.3, pp. 435–448. ISSN: 0025-5831. DOI: 10.1007/BF01459793.

Smith, Breannan, Fernando De Goes, and Theodore Kim (2018). "Stable Neo-Hookean Flesh Simulation". In: *ACM Trans. Graph.* 37.2. ISSN: 0730-0301. DOI: 10.1145/3180491.

Sokolowski, Jan and Jean-Paul Zolésio (1992). *Introduction to shape optimization*. Vol. 16. Springer Series in Computational Mathematics. Shape sensitivity analysis. Springer-Verlag, Berlin, pp. ii+250. ISBN: 3-540-54177-2. DOI: 10.1007/978-3-642-58106-9.

Sotiras, Aristeidis, Christos Davatzikos, and Nikos Paragios (2013). "Deformable medical image registration: A survey". In: *IEEE transactions on medical imaging* 32.7, p. 1153. ISSN: 0278-0062. DOI: 10.1109/tmi.2013.2265603.

St Venant, Adhémar Barré de (1844). "Sur les pressions qui se développent à l'intérieur des corps solides lorsque les déplacements de leurs points, sans altérer l'élasticité, ne peuvent cependant pas être considérés comme très petits". In: *Bull. Soc. Philomath.* 5, pp. 26–28.

Su, Weijie, Stephen Boyd, and Emmanuel J. Candès (2016). "A differential equation for modeling Nesterov's accelerated gradient method: theory and insights". In: *J. Mach. Learn. Res.* 17, Paper No. 153, 43. ISSN: 1532-4435.

Suwelack, Stefan, Sebastian Röhl, Sebastian Bodenstedt, Daniel Reichard, Rüdiger Dillmann, Thiago dos Santos, Lena Maier-Hein, Martin Wagner, Josephine Wünscher, Hannes Kenngott, Beat P. Müller, and Stefanie Speidel (2014). "Physics-based shape matching for intraoperative image guidance". In: *Medical Physics* 41.11, p. 111901. DOI: 10.1118/1.4896021.

Wächter, Andreas and Lorenz T. Biegler (2006). "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". In: *Mathematical Programming* 106.1, pp. 25–57. ISSN: 1436-4646. DOI: 10.1007/s10107-004-0559-y.

Wang, Yongmei and Lawrence H. Staib (2000). "Physical model-based non-rigid registration incorporating statistical shape information". In: *Medical Image Analysis* 4.1, pp. 7 –20. ISSN: 1361-8415. DOI: 10.1016/S1361-8415(00)00004-9.

Wex, Cora, Susann Arndt, Anke Stoll, Christiane Bruns, and Yuliya Kupriyanova (2015). "Isotropic incompressible hyperelastic models for modelling the mechanical behaviour of biological tissues: a review". In: *Biomedical Engineering / Biomedizinische Technik* 60.6, pp. 577–592. DOI: doi:10.1515/bmt-2014-0146.

Xie, Hujin, Jialu Song, Yongmin Zhong, Chengfan Gu, and Kup-Sze Choi (2022). "Constrained finite element method for runtime modeling of soft tissue deformation". In: *Applied Mathematical Modelling* 109, pp. 599–612. ISSN: 0307-904X. DOI: 10.1016/j.apm.2022.05.020.

Youett, Jonathan, Oliver Sander, and Ralf Kornhuber (2019). "A Globally Convergent Filter-Trust-Region Method for Large Deformation Contact Problems". In: *SIAM Journal on Scientific Computing* 41.1, B114–B138. DOI: 10.1137/17M1142338.

Yusa, Yasunori, Shota Miyauchi, and Hiroshi Okada (2021). "Performance investigation of quasi-Newton-based parallel nonlinear FEM for large-deformation elastic-plastic analysis over 100 thousand degrees of freedom". In: *Mechanical Engineering Journal* 8.3, pp. 21–00053–21–00053. DOI: 10.1299/mej.21-00053.