2013•2014
FACULTEIT WETENSCHAPPEN
master in de informatica

Masterproef

Computational science: Medical imaging using CUDA

Promotor :
Prof. dr. Philippe BEKAERT

Thomas Kovac

Proefschrift ingediend tot het behalen van de graad van master in de informatica

2013•2014
FACULTEIT WETENSCHAPPEN
master in de informatica

Masterproef

Computational science: Medical imaging using CUDA

Promotor :
Prof. dr. Philippe BEKAERT

Thomas Kovac
Proefschrift ingediend tot het behalen van de graad van master in de informatica

universiteit hasselt | Maastricht University

Abstract

As multiple sclerosis is known to cause atrophy and deformation in the brain, it also influences the shape and size of the corpus callosum. Longitudinal studies try to quantify these changes using medical image analysis techniques for measuring and analyzing the shape and size of a corpus callosum cross-sechtion embedded in a specially selected measurement plane. In this thesis, a framework has been implemented that automatically identifies and extracts the plane that contains the minimal cross-sectional area of the corpus callosum from a given MRI volume. The framework relies on deformable image registration for the segmentation and area calculation of the cross-section area of the corpus callosum embedded in a plane. Therefore, we used the free-form deformation transformation model, using B-splines, to chararacterize deformations based on a grid of control points. Computations that take place on a per-pixel basis have been transfered to the coarse grid of control points, leading to potential computational gains. To further improve the results of the registration process, a hierarchical multiresolution method is used that will increasingly refine the grid of control points. The whole registration process has also been accelerated by making use of the parallelization capabilities of a GPU. The registration process as well as the framework that uses deformable registration to identify and extract the plane containing the corpus callosum of minimal cross-sectional area, have been properly evaluated using synthetic and medical data.

Acknowledgements

I would like to express my special appreciation and thanks to my advisor Dr. Ir. Sammy Rogmans. Without his expertise, guidance, patience and enthusiasm, this thesis would not have been possible. Besides my advisor, I would also like to thank Dr. Ir. Dirk Smeets of icoMetrix for introducing me to this topic and providing me with the necessary medical data.

Special thanks goes to my friends, for reminding me that there is more to life than work. They enabled me to enjoy my free time to the fullest, but they also helped me to focus on my studies.

I would like to thank my love and best friend Heidi for her love, friendship and unyielding support.

Last but not least, I would like to thank my parents for their unconditional support, both financially and emotionally, throughout my entire life. Without their love, care, and patience, I would not be the person I am today.

Nederlandse Samenvatting

1 Multiple Sclerose

Multiple sclerose (MS) is een ontstekingsaandoening aan het centrale zenuwstelsel, waar er schade kan worden berokkend aan het myeline en aan axonen, maar het kan ook algemene atrofie en vervorming van de hersenen veroorzaken. Initieel is de ontsteking tijdelijk en zal de myelineschade worden hersteld, maar bij een patiënt die lijdt aan MS zal dit herstel na verloop van tijd afnemen. Het afnemen van myeline kan na verloop van tijd ervoor zorgen dat een patiënt zijn motorische functies verliest, maar ook zijn visuele en sensorische vaardigheden kunnen worden aangetast. Dit zijn maar enkele symptomen, maar ze zijn de meest opvallende.

Een Amerikaanse organisatie, genaamd "United States National Multiple Sclerosis Society", heeft vier klinische categorieën beschreven waarin een patiënt met MS kan worden geclassificeerd:

1. Relapsing-remitting MS (RRMS), waarin een patiënt lijdt aan onvoorspelbare aanvallen van MS, maar ook tijden van remissie kent;

2. Secondary progressive MS (SPMS), waar de patiënt eerst begint met RRMS, maar na een tijdje er geen sprake meer is van remissie;

3. Primary progressive MS (PPMS), wat wordt gekarakteriseerd door een geleidelijke afname van vaardigheden vanaf het begin, maar zonder, of met minimale, periodes van remissie;

4. Progressive relapsing MS (PRMS), wat een geleidelijke degeneratie veroorzaakt, gelijkend op PPMS, maar met af en toe een bijkomende MS aanval.

Hoewel de meest fundamentele aspecten van MS goed zijn beschreven in de medische literatuur, geeft het gebruik van medische beelden ons nog steeds nieuwe inzichten in de ontwikkeling van MS en hoe MS reageert op eventuele behandelingen. De medische beelden die meestal hiervoor worden gebruikt, zijn MRI's, ook wel magnetische resonantie beelden genoemd.

1.1 Corpus Callosum

Aangezien multiple sclerose atrofie in de hersenen kan veroorzaken, kan het dus ook atrofie veroorzaken in het corpus callosum. Het corpus callosum, wat "tough body" betekent in het Engels als men dit zou vertalen vanuit het Latijn, is de grootste bundeling van zenuwvezels in het hele zenuwstelsel. Het verbindt de twee hersenhelften en zorgt voor de uitwisseling van informatie tussen deze twee helften. Het corpus callosum is een uiterst vezelrijke structuur in de hersenen en er wordt verondersteld dat het corpus callosum minder gevoelig is voor vochtverlies. Door deze eigenschap, kan het corpus callosum beschouwd worden als een meer geschikte maatstaaf voor het analyseren van neurologische degeneratie en atrofie dan bijvoorbeeld het louter beschouwen van het hersenvolume, dat immers wel gevoelig is voor vochtverlies. Longitudinale studies proberen de verandering in volume en vorm van het corpus callosum te kwantificeren door gebruik te maken van medische analysetechnieken. Deze technieken analyseren en waarnemen veranderingen in vorm en volume door het oppervlakte van de dwarsdoorsnede van het corpus callosum te berekenen, meestal gebruikmakend van het mid-sagittaal vlak (MSP).

1.2 Minimale CC Oppervlakte Vlak Extractie

De meeste technieken die gehanteerd worden voor het vinden van het MSP, zijn ofwel op *symmetrie* ofwel op *kenmerken* gebaseerd. Technieken gebaseerd op symmetrie, stellen dat de hersenen bilaterale symmetrie vertonen. Het MSP wordt dan ook geselecteerd door deze symmetrie te maximaliseren. Kenmerkengebaseerde technieken, daarentegen, definiëren het MSP als het vlak dat het meeste overeenkomt met de interhemisferische fissuur. De zonet beschreven methodes hangen af van het juist identificeren van dit mid-sagitaal vlak. Als dit vlak niet optimaal werd gekozen, zal de berekening van het oppervlakte van de dwarsdoorsnede van het corpus callosum immers foutief zijn. Ook negeert het gebruik van het MSP vlak, op welke manier deze ook werd bekomen, de karakteristieken van het corpus callosum.

Ishaq probeerde de tekortkomingen van deze methodes te verhelpen door een nieuwe, geautomatiseerde techniek te hanteren [21]. Deze geautomatiseerde techniek zoekt naar een vlak met minimale dwarsdoorsnede oppervlakte van het corpus callosum en deze verzekert dat de oppervlakte van de dwarsdoorsnede correct wordt berekend.

2 Medische Beeldregistratie

Het zoeken naar dit vlak met minimale dwarsdoorsnede oppervlakte van het corpus callosum wordt verwezenlijkt door gebruik te maken van medische beeldregistratie. Medische beelden worden gebruikt voor het stellen van een diagnose, het plannen van een behandeling, het vaststellen van de progressie van een ziekte en het kan een chirurg helpen bij het uitoefenen van een chirurgische ingreep. Computertomografie (CT) en magnetische

resonantie beelden (MRI) van een hoofd worden vaak gebruikt voor het stellen van een diagnose alsook als hulpmiddel voor het plannen van een chirurgische ingreep, omdat deze twee modaliteiten complementaire informatie bevatten over botstructuur en over zachte weefsels. Traditioneel zullen beide beelden langs elkaar worden gelegd op een lichtbak. Het bestuderen van deze beelden vereist van de chirurg dat hij in zijn hoofd een mapping maakt van welke punten van het ene beeld overeenkomen met punten van het andere beeld. Dit is geen triviale taak en dit kan enkel correct gebeuren als de chirurg in kwestie veel ervaring heeft met dit probleem. De onzekerheid die gepaard gaat met de mentale mapping van punten, kan onzekerheid scheppen bij het stellen van een diagnose of bij het maken van een chirurgische planning.

Medische beeldregistratie zal deze beelden aligneren met behulp van een computer. Een belangrijk voordeel is dat een computer meer accuraat kan werken dan een mens, alsook kan het voor eventuele visualisatie zorgen van de twee gealigneerde beelden. Beeldregistratie is dus een proces dat de juiste geometrische vervorming probeert te achterhalen zodat twee beelden precies met elkaar kunnen worden gealigneerd. In deze thesis zal de volgende terminologie worden gehanteerd: het beeld dat niet wordt vervormd tijdens het proces, heet het *referentiebeeld* of het *statische beeld*. Het tweede beeld, dat stapsgewijs wordt vervormd zodat het kan worden gealigneerd met het eerste beeld, wordt vaak het *zwevende beeld* of het *template beeld* genoemd.

Elke beeldregistratie techniek die er maar bestaat, bevat de volgende drie componenten:

1. een **transformatiemodel**, ook wel *vervormingsmodel* genoemd, die bepaalt hoe een beeld vervormd kan worden naar een ander beeld;

2. een **maat van overeenkomst**, die de overeenkomst tussen het referentie beeld en het zwevende beeld meet;

3. een **optimalisatietechniek**, die de meest optimale transformatie parameters probeert te achterhalen in functie van de maat van overeenkomst.

Elk van de zonet opgesomde componenten kan op een verschillende manier worden geïmplementeerd. Overheen de jaren zijn er verschillende algoritmen ontworpen. Voor een overzicht kan de lezer terecht bij Sectie 4. De ontworpen beeldregistratie algoritmen kunnen worden geclassificeerd volgens verschillende criteria, die nu zullen worden besproken. De meeste criteria bespreken hoe twee beelden met elkaar worden vergeleken of hoe het zwevende beeld wordt vervormd om meer te gelijken op het referentiebeeld.

2.1 Mono-modale en Multi-modale Applicaties

Medische beelden kunnen worden bekomen in verschillende modaliteiten. Enkele van deze modaliteiten zijn de volgende: computertomografie (CT), röntgen (ook bekend als *X-ray*),

magnetische resonantie (MRI), positronemissietomografie (PET), echografie, tomoscintigrafie (ook bekend als *Single Photon Emission Tomography*, oftewel SPECT), enzovoorts. Elk type van modaliteit heeft een specifiek domein waarin het het beste gebruikt kan worden. MRI's en CT scans, bijvoorbeeld, zijn anatomische beelden van hoge resolutie, maar ze bevatten weinig fysiologische informatie. PET en SPECT beelden, daarentegen, bevatten veel fysiologische informatie, maar kunnen weinig anatomische informatie bieden. Het combineren van beelden van verschillende modaliteiten kan ervoor zorgen dat het resulterende beeld van hoge resolutie is en dat zowel fysiologische als anatomische informatie bevat.

Applicaties voor beeldregistratie die gebruik maken van medische beelden kunnen in twee categorieën worden opgedeeld: *mono-modale applicaties*, waar de applicatie beelden registreert die tot eenzelfde modaliteit behoren, en *multi-modale applicaties*, waar de afbeeldingen die worden geregistreerd vanuit verschillende modaliteiten afkomstig zijn. Het framework dat werd ontworpen voor deze thesis registreert beelden afkomstig van eenzelfde MRI volume, dus kan het worden geclassificeerd onder mono-modale applicaties.

2.2 Kenmerkgebaseerde en Voxelgebaseerde Maten van Overeenkomst

Een maat van overeenkomst wordt gebruikt door beeldregistratie applicaties om een bepaalde graad van overeenkomst tussen twee beelden vast te stellen. Hier kan men twee soorten van maatstaven onderscheiden, namelijk *kenmerkgebaseerde* en *voxelgebaseerde* maatstaven. Kenmerkgebaseerde registratie applicaties maken gebruik van punten, lijnen of oppervlaktes als kenmerken van een beeld en proberen de afstand tussen de overeenkomstige kenmerken van twee beelden te minimaliseren. Zulke registratie applicaties kunnen voor zowel mono-modale als multi-modale problemen worden gehanteerd, maar ze vereisen wel een kenmerkenextractiefase, in de vorm van segmentatie of *landmark-detectie*, die niet triviaal is. Als er fouten worden geïntroduceerd tijdens de kenmerken extractie fase, zal dit de correctheid van de registratie averechts beïnvloeden.

Zulke fouten kunnen worden vermeden door berekeningen te maken met de beeldintensiteiten in plaats van gebruik te maken van kenmerken van een beeld. Deze maatstaven worden ook *voxelgebaseerde* maten van overeenkomst genoemd en ze berekenen de graad van overeenkomstige informatie van beeldintensiteiten. Deze methodes zijn relatief simpel voor mono-modale applicaties, maar ze zijn echter meer complex voor multi-modale applicaties. In het laatste decennium zijn voxelgebaseerde maten van overeenkomst de vaste keuze geworden voor het meten van overeenkomst, meerendeel omdat ze accuraat en robuust zijn. Er zijn verschillende soorten maten van overeenkomst. Enkele van hen worden besproken in Sectie 2.3.2, zoals *kleinste kwadraten* (KK), *mutuele informatie* (MI) en *kruiscorrelatie* (KC). KK en KC worden merendeel gebruikt in mono-modale applicaties, terwijl MI daarentegen voor zowel mono-modale als multi-modale applicaties gebruikt kan

worden. Aangezien het ontworpen framework een mono-modaal probleem probeert op te lossen, wordt "kleinste kwadraten" als maat voor overeenkomst gebruikt.

2.3 Rigide en Niet-Rigide Transformatiemodellen

Het transformatiemodel bepaalt hoe een beeld vervormd kan worden naar een ander beeld. Het meest simpele model gaat ervan uit dat de transformatie zich rigide of affien gedraagt. Rigide transformatie compenseert enkel de algemene verschillen in rotatie en translatie. Affiene transformatie beschouwt ook nog eens schaalveranderingen en afschuivingen als extra mogelijkheden van vervorming. Dergelijke aannames zorgen ervoor dat de beeldregistratie beelden aligneert waarvan de inhoud als rigide kan worden beschouwd. De schedel, alsook andere botstructuren, vervormen niet in een gezond lichaam en kunnen dus als rigide worden beschouwd. Maar het menselijke lichaam bestaat ook uit zachte weefselstructuren, die niet als rigide objecten kunnen worden beschouwd. Dit zorgt ervoor dat de meeste beeldregistratie applicaties rekening moeten houden met niet-rigide transformaties om tot een correcte registratie te komen.

Bij niet-rigide beeldregistratie zal er naar een veld van translatievectoren gezocht worden die elke voxel van het zwevende beeld mapt met de overeenkomstige voxel in het referentiebeeld. Dit veld wordt ook het *vervormingsveld* genoemd. Iedere individuele voxel ondergaat een translatie zodat de lokale vervorming tussen twee beelden kan worden gecompenseerd. Terwijl het affiene transformatiemodel enkel 12 vrijheidsgraden gebruikt, zijn er veel meer vrijheidsgraden nodig bij een niet-rigide transformatie om zo het vervormingsveld voldoende te voorzien van flexibiliteit om lokale vervormingen te kunnen verwezenlijken.

Het is belangrijk om op te merken dat niet alle vervormingen fysiek mogelijk of acceptabel zijn en dat de beeldintensiteiteninformatie onvoldoende gedetailleerd kan zijn om een éénduidig vervormingsveld op te stellen. Daarom kan er gebruik gemaakt worden van een regularisatiemethode. Deze garandeert dat het vervormingsveld lokaal consistent is en dat de transformatie met gelijke maten verloopt. Zulke eigenschappen van geleidelijke vervorming kunnen op verschillende manieren aan een registratiealgoritme worden opgelegd. Een voorbeeld is het *vrije-vorm vervormingsmodel*, dat inherent voor geleidelijke vervorming zorgt. Deze kan nog gekoppeld worden aan een regularisatiemethode die niet-acceptabele vervormingen zal afstraffen en ervoor zorgt dat meer gunstige vervormingen worden gekozen.

Gegeven dat het corpus callosum een weefselstructuur is die in een patiënt met MS vervormd kan worden door atrofie, moet een beeldregistratiemethode rekening houden met niet-rigide transformaties. Deze thesis opteert dan ook voor het gebruik van het vrije-vorm vervormingsmodel, wat een niet-rigide transformatiemodel is.

2.3.1 Vrije-Vorm Vervormingsmodel

De oorsprong van vrije-vorm vervorming is terug te voeren naar het domein van computergesteund ontwerpen (in het Engels heet dit *computer-aided design*), maar het kan ook worden gebruikt bij medische beeldanalyse. Deze thesis maakt gebruik van een vrije-vorm vervormingsmodel dat is gebaseerd op B-splines. Het basisidee achter dit model is dat een object vervormd kan worden door de manipulatie van een mesh van controle punten. De resulterende transformatie van controle punten bepaalt de vervorming van het 3-D, of 2-D, object dat men wilde behandelen en produceert een geleidelijke, en een C^2 continue, vervorming. In vergelijking met thin-plate splines en elastic-body splines, zijn B-splines lokaal gecontroleerd, waardoor ze computationeel gezien efficiënt berekend kunnen worden, ook al zijn er veel controle punten in gebruik. Meer specifiek: de kubische B-splines zorgen ervoor dat een controlepunt enkel de transformatie van zijn lokale omgeving beïnvloedt.

2.4 Dimensionaliteit

Beeldregistratie kan worden verwezenlijkt met beelden van verschillende dimensionaliteiten. Deze dimensies kunnen puur spatiaal zijn, maar ook de tijd kan een dimensie vormen. In ieder geval, een probleem kan ook worden geclassificeerd afhankelijk van de hoeveelheid dimensies dat het probleem vereist. De meeste medische beeldregistratie applicaties registreren twee 3-D beelden met elkaar, waar de tijd buiten beschouwing wordt gelaten. Deze vorm van registratie behandelt normaliter twee tomografische datasets. Bij 2-D-2-D beeldregistratie worden 2-D vlakken vanuit tomografische data sets geëxtraheerd en deze worden vervolgens gebruikt als input voor het registratie proces. 3-D-3-D beeldregistratie applicaties zijn meer complex dan hun 2-D-2-D tegenhanger, aangezien het aantal parameters dat wordt gebruikt, alsook het datavolume, bij een 3-D-3-D beeldregistratie groter is dan bij een 2-D-2-D beeldregistratie. Men kan ook een 2-D-3-D registratie uitvoeren, waarbij men een vlak registreert op een spatiaal volume.

In deze thesis probeert men een template te registreren op een vlak dat uit een gegeven MRI-volume werd geëxtraheerd. Dit betekent dat het geïmplementeerde beeldregistratieframework aan 2-D-2-D beeldregistratie doet.

2.5 Optimalisatie

Optimalisatietechnieken worden door beeldregistratietechnieken gebruikt om tot optimale transformatieparameters te komen die ervoor zorgen dat twee beelden goed worden gealigneerd. Of met andere woorden: het bepaalt hoe de transformatieparameters worden aangepast zodat de twee te registreren beelden meer op elkaar gelijken. Er zijn verscheidene methodes voorgesteld om functies te optimaliseren. Dit betekent dat er voor een functie naar ofwel een mimimum ofwel een maximum wordt gezocht. Elk van deze metho-

des variëert in complexiteit. Goede optimalisatietechnieken bepalen snel en accuraat de nodige transformatieparameters. In niet-rigide applicaties wordt het kiezen van een juiste optimalisatietechniek bemoeilijkt door het feit dat hoe meer "niet-rigide", of flexibel, het transformatiemodel moet zijn, des te meer parameters er moeten zijn om de vervorming te kunnen beschrijven. Door het "lokale minima"-probleem, is de kans groot dat een optimalisatietechniek een stel parameters kiest dat een algemeen goed resultaat levert, maar niet het meest optimale resultaat.

Deze transformatieparameters worden ofwel onmiddellijk berekend, ofwel worden ze geleidelijk aan bepaald door middel van een "zoektocht". Deze laatste methode vindt de meest optimale transformatieparameters door een optimum, een minimum, te zoeken voor een bepaalde kostfunctie. In het geval van beeldregistratie, is de kostfunctie de maat van overeenkomst.

Veel optimalisatietechnieken maken gebruik van de eerste afgeleide van de kostfunctie om de robuustheid van het optimaliseren te verhogen maar ook om het aantal iteraties van de optimalisatie te beperken. Deze berekening brengt meestal een aanzienlijke computationele kost met zich mee. In ons geval zal een deel van de berekeningen dat nodig is voor het berekenen van de afgeleide van de kostfunctie, overeenkomen met de berekeningen die nodig zijn voor het berekenen van de kostfunctie. Hierdoor wordt de extra computationele kost verminderd.

De meest simpele optimalisatietechnieken, die gebruik maken van de eerste afgeleide, hanteren een techniek genaamd *steepest descent*. Startende van een gegeven positie, zullen deze methodes zoeken naar een optimum in de richting van de eerste afgeleide. Meer geavanceerde methodes, denk maar aan de quasi-Newton methodes, gebruiken ook de tweede afgeleide in de zoektocht naar een optimum.

De robuustheid en snelheid van een gekozen optimalisatietechniek kan worden verbeterd door een hiërarchische multi-resolutie methode te hanteren. Gebruikmakend van deze methode, zal het probleem opgesplitst worden in verschillende optimalisatiefases, waar in elke fase de complexiteit van het probleem geleidelijk aan wordt verhoogd. Startende van beelden waarvan de resoluties werden verlaagd en een transformatiemodel dat werd geïnitialiseerd met een ruw grid van controle punten, wordt geleidelijk aan de resolutie van de beelden verhoogt en wordt het grid van controlepunten steeds meer verfijnd totdat de gewenste accuraatheid wordt bereikt. De volgende sectie geeft hier meer uitleg over.

3 Overzicht

Voor deze thesis werd een framework ontworpen dat een vlak zal identificeren en extraheren dat een corpus callosum bevat met minimale dwarsdoorsnede oppervlakte, gebruikmakende van CUDA om het registratiegedeelte van het framework te versnellen. Normaliter zou een dergelijke techniek een MRI-volume vereisen waarin het corpus callosum pre-gesegmenteerd werd, maar deze segmentatie is meestal niet beschikbaar. Het

geïmplementeerde framework zal het corpus callosum segmenteren op het geëxtraheerde vlak. De segmentatie gebeurt door een template beeld, dat een gesegmenteerd corpus callosum bevat, te registreren op het geëxtraheerde vlak. Het oppervlakte van het corpus callosum wordt vervolgens berekend door de integraal te berekenen van de determinant van de Jacobiaan van het vervormingsveld. Het corpus callosumoppervlakte kan worden beschouwd als een functie van de extractieparameters van het te extraheren vlak, dus we gebruiken een optimalisatietechniek om een vlak te extraheren die een corpus callosum bevat met minimale dwarsdoorsnede oppervlakte.

In deze thesis zal er gebruik gemaakt worden van een niet-rigide transformatiemodel en een voxelgebaseerde maat van overeenkomst. Het geïmplementeerde framework zal *kleinse kwadraten* gebruiken als voxelgebaseerde maat van overeenkomst; het is dus een 2-D-2-D mono-modale beeldregistratieapplicatie. Een *vrije-vorm vervormingsmodel*, gebruikmakend van B-splines, zal worden gebruikt om niet-rigide transformaties te modelleren, alsook zal *steepest descent* gehanteerd worden als optimalisatietechniek.

Om de robuustheid en snelheid van het registratieproces te verbeteren, wordt een hiërarchische multi-resolutie methode gehanteerd. Van zowel het referentiebeeld als het zwevende beeld zal een Gaussiaanse piramide worden opgemaakt, die op elke laag van de piramide een geschaleerde versie van de afbeelding in kwestie bevat. Op het laagste niveau bevindt zich het beeld op volle resolutie. Deze resolutie van het beeld neemt af naar mate men stijgt in de piramide. Startende met de beelden op laagste resolutie, zal men beginnen met het registratieproces, gebruikmakend van een ruw grid van controlepunten. Het resultaat van de registratie van een lager resolutieniveau wordt gebruikt bij het resolutieniveau daarboven en het registratieproces wordt herhaald. Het registratie proces stopt zodra het niveau met de originele resolutie van de beelden werd behandeld. Door deze methode te hanteren kunnen globale vervormingen vroeg worden gedetecteerd op een niveau met lage beeldresolutie en worden lokale vervormingen gedetecteerd op niveau's met hogere beeldresolutie.

Het framework voor het vinden van het vlak met minimale corpus callosum dwarsdoorsnede oppervlakte alsook het framework dat de beeldregistratie afhandeld, worden gedetailleerd besproken in deze thesis. Beide raamwerken worden ook geëvalueerd en worden de resultaten uitvoerig besproken.

Contents

Abstract i

Acknowledgements iii

Nederlandse Samenvatting v
Multiple Sclerose . . . v
Corpus Callosum . . . vi
Minimale CC Oppervlakte Vlak Extractie . . . vi
Medische Beeldregistratie . . . vi
Mono-modale en Multi-modale Applicaties . . . vii
Kenmerkgebaseerde en Voxelgebaseerde Maten van Overeenkomst . . . viii
Rigide en Niet-Rigide Transformatiemodellen . . . ix
Vrije-Vorm Vervormingsmodel . . . x
Dimensionaliteit . . . x
Optimalisatie . . . x
Overzicht . . . xi

I Introduction and Problem Setting 1

1 Introduction 3
1.1 Thesis Outline . . . 4

2 Problem Setting 5
2.1 Multiple Sclerosis . . . 5
2.1.1 Categories . . . 5
2.1.2 Diagonosis and MRI . . . 6
2.1.3 Corpus Callosum . . . 6
2.2 Minimum CC Area Plane Extraction . . . 7
2.2.1 Measurement of Corpus Callosum Size . . . 8
2.3 Image Registration . . . 9

2.3.1 Mono-modal and Multi-modal Applications . . . 10
2.3.2 Feature-Based and Voxel-Based Similarity Measures . . . 11
2.3.3 Rigid and Non-rigid Transformation . . . 11
2.3.3.1 Free-Form Deformation . . . 12
2.3.4 Dimensionality . . . 12
2.3.5 Optimization . . . 13
2.4 Method Overview . . . 13

II Background and Previous Work 15

3 Background 17
3.1 Compute Unified Device Architecture . . . 17
3.1.1 Hardware Execution . . . 18
3.2 Medical Image Orientations . . . 19
3.2.1 Spacial Coordinates . . . 19
3.2.2 Voxel Ordering . . . 20
3.3 NIfTI . . . 20
3.3.1 NIfTI-1 Data Format . . . 22
3.4 Similarity Measures . . . 22
3.5 Splines and B-splines . . . 24
3.6 Image Warping . . . 26
3.7 Interpolation Methods . . . 27

4 Previous Work and Related Research 31

III Framework Algorithms Description and Implementation Details 33

5 Algorithms 35
5.1 MCAP Extraction and CC Segmentation . . . 35
5.1.1 Method Overview . . . 35
5.1.2 Template Preperation . . . 36
5.1.3 Slice Extraction . . . 37
5.1.4 Slice Registration . . . 37
5.1.5 Area Calculation . . . 38
5.1.6 Area Optimization . . . 39
5.2 Free-Form Deformable Registration Using B-splines . . . 39
5.2.1 Configuration . . . 40
5.2.1.1 Control Points . . . 40

5.2.1.2 Calculating the Displacement Field . . . 40
5.2.2 Deformation Regularization . . . 41
5.2.2.1 Diffusion . . . 42
5.2.2.2 Curvature . . . 42
5.2.3 Goal . . . 43
5.2.4 Optimization . . . 43
5.2.5 Multiresolution Approach . . . 45
5.2.5.1 Gaussian Pyramid . . . 46
5.2.5.2 Grid Subdivision . . . 46

6 Implementation Details 49
6.1 Image Registration Using CUDA . . . 49
6.1.1 Displacement Field by B-spline Interpolation . . . 50
6.1.2 Similarity Cost Function Gradient Calculation . . . 53
6.1.3 Framework Organization . . . 54
6.1.4 GPU Implementation . . . 56
6.1.4.1 Naïve Implementation . . . 56
6.1.4.2 Optimized Implementation . . . 57
6.2 Numerical Differentiation . . . 59
6.3 Spatial Filtering . . . 63
6.3.1 Discrete Convolution . . . 63
6.3.2 Gaussian Smoothing Filter . . . 63
6.3.3 Sobel Filter . . . 64

IV Evaluation and Conclusion 65

7 Evaluation 67
7.1 Deformable Registration Evaluation . . . 67
7.1.1 Synthetic Data . . . 67
7.1.1.1 Similarity after Registration . . . 68
7.1.1.2 Root Mean Square Error Metric . . . 68
7.1.1.3 Magnitude of Difference Metric . . . 68
7.1.1.4 Processing Time . . . 73
7.1.2 Medical Data . . . 74
7.2 Evaluation of MCAP Extraction . . . 76

8 Conclusion 83
8.1 Extraction of MCAP . . . 83
8.2 Deformable Image Registration . . . 84
8.3 Future Work . . . 84

Bibliography **87**

List of Figures

2.1 Corpus Callosum 7

3.1 Medical Image Orientation 19
3.2 Different plane names 21
3.3 The two most popular orientation schemes 21
3.4 Centered B-splines of degree 0 to 3 25
3.5 Calculating value for pixel $I_{x,y}$ by means of bilinear interpolation 28
3.6 Calculating value for point $P_{x,y,z}$ by means of trilinear interpolation 29

5.1 Implausible deformations 42
5.2 Catmull and Clark subdivision masks. The top left configuration is used to calculate a new control point with coordinates $(2i+1, 2j+1)$. With the top right configuration, a control point is calculated with the coordinates $(2i, 2j+1)$. Bottom left configuration is used to calculate a control point with the coordinates $(2i, 2j)$. The bottom right configuration is used to calculate a control point with coordinates $(2i+1, 2j)$. 47

6.1 A grid of control points superimposed on the pixels of the template image. Both marked pixels are located at the same relative offset within their respective tiles, so both will use the same $\beta_l(u)\beta_m(v)$ value. 51
6.2 Framework organization 55

7.1 Little checkerboard 69
7.2 Large checkerboard with different intensities 70
7.3 Similarity after registration 71
7.4 RMSE of the generated displacement field and the ground truth 72
7.5 Magnitude of Difference of generated displacement field and ground truth . 73
7.6 Processing time of registration using CPU, GPU and the optimized GPU versions 74
7.7 Image difference of the fixed image and warped moving image. Left is before the registration process and right is after the registration process. . 77

7.8 Sum of squared differences between the fixed image and the warped moving image before and after the registration process. 78
7.9 Correlation coefficient between the fixed image and the warped moving image before and after the registration process. 79
7.10 Graph of normalized CC areas of 8 MRI volumes 80
7.11 Comparison of normalized CC areas obtained using different initializations. Each bar represents the mean normalized area of the 8 MRI volumes used. For the top graph, an index $x = 1$ means that R_x was set to 1, while R_y and T_z were set to zero. 81

Part I

Introduction and Problem Setting

Chapter 1

Introduction

Multiple Sclerosis is an inflammatory disorder of the brain and spinal cord and it has been known to cause atrophy and deformation in the corpus callosum. Longitudinal studies try to quantify these changes by using medical image analysis techniques for measuring and analyzing the size and shape of the corpus callosum. These medical techniques mostly analyze and track changes in the corpus callosum by measuring the corpus callosum cross-sectional area by selecting a 2-D measuring plane, typically a mid-sagittal plane. This method depends on the accurate identification of the mid-sagittal plane. If this is done incorrectly, consequently the measurement of the corpus callosum area is also faulty. Therefore, an automation of finding a plane with minimal corpus callosum area is implemented to ensure that the measurement of the cross-sectional area of the corpus callosum is done correctly.

The employed method of finding a plane with minimal corpus callosum area depends heavily on deformable image registration. Therefore, this thesis focuses on the theoretical background and implementation of a registration algorithm. As the image registration process must be employed several times in search for the plane with minimal corpus callosum area, it is important that the registration is performed as quickly, and correctly, as possible. The use of a GPU can greatly improve computation time, so this thesis also focuses on algorithms and data structures that exploit the parallel computation capabilities of a GPU. The implemented framework is inspired by the work of various research groups and tries to combine the advantageous approaches into one method. As the registration framework is an integral part of finding a plane with minimal corpus callosum area, it has been evaluated using synthetic data as well as real medical data. The framework for finding a plane with minimal corpus callosum area has also been evaluated using medical data.

1.1 Thesis Outline

This thesis has been split into four parts. Each part will approach the topic from a different angle, but all parts build upon each other. A quick overview of these four parts is the following:

I Introduction and Problem Setting. After a short introduction, a general overview of the problem setting is given. It focuses on explaining what multiple sclerosis entails, how one extracts a plane with minimal cross-sectional corpus callosum area, how an image registration problem is described, and what was implemented for this thesis.

II Background and Previous Work. This part describes the vital concepts of the deformable registration process. All the necessary notions that are required for understanding the subsequent parts are described in full detail. Also, a chapter has been written that sheds light on related research conducted by other research groups.

III Framework Algorithms Description and Implementation Details. First, all the necessary algorithms that are needed for this thesis are outlined in a formal view. Afterwards, the implementation details are described in detail.

IV Evaluation and Conclusion. The evaluation contains the results of the experiments conducted on the image registration framework with synthetic and medical data. The framework of finding the plane of minimal corpus callosum area has also been evaluated and its results are thoroughly discussed. The conclusion summarises what was researched in this thesis and sheds some light on possible future work.

Chapter 2

Problem Setting

2.1 Multiple Sclerosis

Multiple sclerosis (MS) is primarily an inflammatory disorder of the brain and spinal cord in which lymphocytic infiltration leads to damage of myelin[1] and axons, atrophy and lesions. Initially, the inflammation is temporary and the remyelination occurs after the inflammation. However, with MS the remyelination is not durable. This demyelination generally causes the gradual loss of motor, sensory and visual skills, but many more symptoms and signs can occur [7].

2.1.1 Categories

The United States National Multiple Sclerosis Society has described four clinical categories that can describe a patient's situation concerning MS:

1. Relapsing-remitting MS (RRMS), in which the patient suffers from unpredictable attacks of MS but also knows periods of remission;

2. Secondary progressive MS (SPMS), where the patient starts with RRMS but over time the periods of remission fade away;

3. Primary progressive MS (PPMS), which is characterized by steady progression of disability from onset, but with no, or minimal, periods of remission or improvement;

4. Progressive relapsing MS (PRMS), which causes a steady degeneration similar to PPMS, but with additional attacks of MS [28].

[1] Electrically insulating material that forms a layer around the axon of a neuron

2.1.2 Diagonosis and MRI

Although the fundamental aspects of multiple sclerosis pathology have been well described in the scientific literature for more than 130 years, the ability to image the dynamics of the disease process in living subjects through neuroimaging studies continues to advance our understanding of the MS disease process and its response to therapy [38]. Because MS causes lesions and atrophy in the brain, magnetic resonance imaging is ideal for detecting these symptoms. For this reason, dual-echo, fluid-attenuated inversion recovery and postcontrast T1-weighted MRI sequences are regularly used to monitor the course of the disease in patients with confirmed MS and have been included in the diagnostic workup of patients in whom MS is suspected.

Other quantitative magnetic resonance-based techniques with a higher pathological specificity (like magnetization transfer-MRI, diffusion tensor-MRI and proton MR spectroscopy) have been extensively applied to measure disease burden within visible lesions and in the normal-appearing white matter and gray matter of MS patients at different stages of the disease.

These methods, combined with functional imaging techniques, are progressively improving our understanding of the factors associated with MS evolution [12].

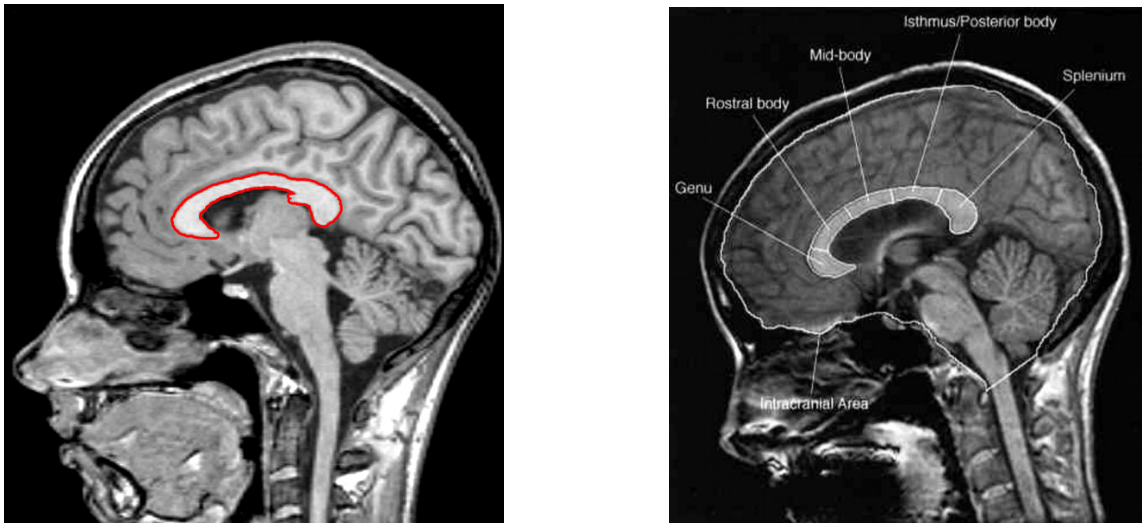
2.1.3 Corpus Callosum

The corpus callosum, Latin for "tough body", is by far the largest bundle of nerve fibers in the entire nervous system. It joins the two cerebral hemispheres, along with a relatively tiny fascicle of fibers called the "anterior commissure". The word "commissure" signifies a set of fibers connecting two homologous neural structures on opposite sides of the brain or spinal cord; thus the corpus callosum is sometimes called the great cerebral commissure. On Figure 2.1a one can see the corpus callosum delineated on a sagittal slice of a brain. Figure 2.1b shows the morphology of the corpus callosum.

Until about 1950 the function of the corpus callosum was a complete mystery. On rare occasions, the corpus callosum in humans is absent at birth, in a condition called *agenesis* of the corpus callosum. Occasionally it may be completely or partially cut by the neurosurgeon, either to treat epilepsy (thus preventing epileptic discharges that begin in one hemisphere from spreading to the other) or to make it possible to reach a very deep tumor.

In 1955 Ronald Myers, a graduate student studying at the University of Chicago proved that the corpus callosum provided the exchange of information between the two cerebral hemispheres, by performing experiments with animals whose chiasm and corpus callosum had both been surgically divided [19].

As previously stated, multiple sclerosis can cause lesions in the corpus callosum, but it can also cause generalized tissue loss (atrophy). Both afflictions affect the cross-sectional area of the corpus callosum. Changes in form and shape of the corpus callosum can have



(a) CC delineated on a sagittal slice of a brain (b) Corpus callosum morphology

Figure 2.1: Corpus Callosum

an effect on motor, sensory and visual skills [50]. The corpus callosum is a dense fibrous rich structure and it is hypothesized to be less sensitive to hydration effects. Therefore its area is potentially a more reliable measure of neuro-degeneration and atrophy than other measurements such as brian volume which may be susceptible to dehydration and rehydration effects [11].

2.2 Minimum CC Area Plane Extraction

As previously described in Section 2.1.3, the corpus callosum joins the two cerebral hemispheres. It acts as a bridge existing out of nerve fibers and it provides the exchange of information across the two hemispheres. Neurological diseases have been known to affect the shape and size of the anatomical structures in the brain. Measurement of this change and its correlation with disease progression has been one of the goals of clinical research [7, 19, 50].

MS often affects the brain ventricles width, overall brain width and specially the corpus callosum whose area loss has been documented in longitudinal studies [50, 51]. These effects on the corpus callosum size have generally been quantified by measuring the cross-sectional area of the corpus callosum.

Another popular brain morphometry measure is the brain volume. However, work by Duning et al. has challenged the use of the whole brain volume as a measure of brain

atrophy due to its susceptability to dehydration and rehydration effects [11], whereas the corpus callosum, being a dense fibrous structure, is hypothesized to be less sensitive to hydration effects and its area is potentially a more reliable measure of neuro-degeneration and atrophy [21].

2.2.1 Measurement of Corpus Callosum Size

In the studies previously mentioned, the changes in the corpus callosum size have been quantified by measuring the corpus callosum cross-sectional area imbedded in a measurement plane. Therefore, it is paramount that the accurate measurement of this change in corpus callosum area is dependent on the repeatable identification of the same corpus callosum cross-section in different scans. Typically, the mid sagittal plane (MSP) serves as this measurement plane.

Previously, many have focused on the identification and extraction of the MSP. Most of these MSP identification and extraction techniques can be classified ino either symmetry or feature based approaches, which differ from each other in the way the MSP is identified [21]. The symmetry based methods are based on the fact that the brain hemispheres display approximate bilateral symmetry. The MSP is therefore selected as the plane maximizing this symmetry. However, these symmetry based methods differ from each other in terms of how the symmetry criteria is defined and measured. In the feature based methods, the MSP is defined as the plane best matching the cerebral interhemispheric fissure in the human brain.

Ishaq emphasizes on two major disadvantages of using MSP as the plane for measurement of the corpus callosum area [21]. First, accurate and repeatable identification of the same corpus callosum cross-section is difficult due to potential changes in brain anatomy over time, which can potentially affect the interhemispheric symmetry and the shape of the interhemispheric fissure. Even small errors in the selection of the MSP have been found to mystify the interpretation of the actual changes in the corpus callosum area due to pathology. Second, these extraction methods only incorporate the information regarding the brain hemispheric symmetry and the interhemispheric fissure, but completely ignore the characteristics of the corpus callosum itself. However, the rate of corpus callosum atrophy and deformation can be independent of the rate of the hemispheric degeneration, therefore the repeatable extration of the same corpus callosum cross-section becomes difficult even for those cases where the brain hemispheres undergo minimal or no change between scans. These issues cast doubt on the reliability of employing the MSP as the measurement plane for measuring the corpus callosum area.

To this end, Ishaq proposed a novel an clinically meaningful criterion for defining an ideal measurement plane for the corpus callosum area measurement. It differs from the symmetry and feature based methods because it is based on finding the plane which optimizes certain physical properties of the corpus callosum itself, which is clinically meaningful and specifically tailored for the task at hand, that is, the measurement of

corpus callosum area changes and its correlation with disease progression. It has been known that the cross-sectional area of the corpus callosum is proportional to the number of nerve fibers passing through it. This implies that the maximum neural transmission between the hemispheres is bounded from above by the minimum cross-section of the corpus callosum bridge. Therefore, Ishaq stipulated that the minimum corpus callosum cross-sectional area is potentially a more appropriate measure of corpus callosum degeneration that the area of any other cross-section of the corpus callosum and the plane containing this cross-section is a more appropriate measurement plane than the MSP. He also states that the criterion proposed by him is not a new criterion for MSP extraction, rather, it is a novel basis for identification of a plane for measuring corpus callosum area change. For convenience, this minimum corpus callosum area plane will be shortened to "MCAP".

It is important to note that for a single MRI volume the MCAP is not guaranteed to be unique, that is, multiple planes in the brain may have the same minimum corpus callosum area. Since all of these planes restrict the neural transmission equally, identification of one of these planes is sufficient for our purposes.

2.3 Image Registration

As will be discussed in Section 5.1, the identification of the MCAP relies on image registration. Image registration is an important preprocessing step in medical image analysis. Medical images are used for diagnosis, treatment planning, disease monitoring and image guided surgery and are acquired using a variety of imaging modalities (see Section 2.3.1). The widespread use of both CT and MR-imaging of the head for diagnosis and surgical planning indicates that the physicians and surgeons gain important complementary information on bony and soft tissue anatomy from these two tomographic modalities. To monitor disease progress and growth of abnormal structures, images are acquired from subjects at different times or with different imaging modalities. In current practice, image volumes of each modality may be transfered to film to be examined side-by-side in the traditional light box, or they may be examined on a computer screen where window and level adjustments can be made interactively. Unfortunately, because of differences in the positioning and orientation of the head, field of view, and resolution, the correspondence between three-dimensional (3-D) points in different images can be difficult to determine. During the examination of the images, the information of the images is combined in the physician's mind to produce a mapping of points from one image space to another. This 3-D mapping relies on experience that allows the physician to recognize homologous anatomical features in CT and MRI. This combined image information is then used in making a diagnosis or in planning for surgical intervention. Uncertainty in the mental mapping from one image to another may lead to uncertainty in the diagnosis or planning [13, 31].

There are, therefore, potential benefits in improving the way these images are compared and combined. Computerized approaches offer potential benefits, particularly by accurately aligning the information in the different images and providing tools for visualizing the combined images [18]. Image registration is a task to reliably estimate the geometric transformation such that two images can be precisely aligned. In this thesis, the following terminology will be used: the image that is not changed during the registration process is often called the *reference* or *fixed image*. The second image that is transformed in such a manner that it increasingly resembles the fixed image, is often called the *template* or *moving image*.

Any registration technique consists out of these three components:

1. a **transformation model**, which relates the fixed and moving images;
2. a **similarity measure**, which measures the similarity between fixed and moving image;
3. an **optimization technique**, which determines the optimal transformation parameters as a function of the similarity function.

Each of these components can be implemented in different ways. Over the years, numerous algorithms have been proposed. For more information about some of these algorithms, look to Section 4. Next, important classification criteria of an image registration algorithm are discussed. The criteria mostly specify the manner in which images are compared and transformed.

2.3.1 Mono-modal and Multi-modal Applications

Medical images are acquired by use of different modalities. Some of these modalities are the following: Computer Tomography (CT), X-ray, Magnetic Resonance Imaging (MRI), Positron Emission Tomography (PET), Ultrasound, Single Photon Emission Computed Tomography (SPECT), etc. Each type of modality has its own strengths, for example: CT and MRI are anatomical images with high spatial resolution. However, their physiological information is limited. On the other hand, although PET and SPECT images can provide physiological information, spatial resolutions of both are too poor to provide clear anatomical information. Thus it would be advantageous to combine images from different modalities, so that the resulting image can provide both physiological and anatomical information with high spatial resolution for use in clinical diagnosis and therapy.

Image registration applications that use medical images can thus be divided into two categories: *mono-modal applications*, where the images to be registered belong to the same modality, as opposed to *multi-modal applications*, where the images to be registered stem from two different modalities [31]. Since our application registers MRI slices coming from the same volume, this implemented framework can be considered a mono-modal application.

2.3.2 Feature-Based and Voxel-Based Similarity Measures

A similarity measure is used by registration applications to determine the degree of alignment between two images. The two main approaches are *feature-based* and *voxel-based* similarity measures. Feature-based registration approaches usually utilise points, lines or surfaces as features and aim to minimize the distance between the corresponding features in the images. An advantage of feature-based registration is that it can be used for both mono- and multi-modality registration but the need for a feature extraction step, in form of landmark detection or segmentation, can be troublesome. Moreover, any error during the feature extraction stage, wether manual or automated, will adversely affect the registration and cannot be recovered at a later stage.

It is possible to avoid such errors by using the image intensities directly without the need for feature extraction. This relies on voxel-based similarity measures which aim to measure the degree of shared information in the image intensities. This is relatively simple in the case of mono-modality registration, but more complex for multi-modalality registration. Over the last decade, voxel-based similarity measures have become the method of choice for measuring image alignment, largely due to their robustness and accuracy [40]. There are different voxel-based similarity measures, as we will discuss in Section 3.4, like the *sum of squared differences* (SSD), *mutual information* (MI), and *correlation coefficient* (CC). SSD and CC are used in a mono-modal setting, while MI can also be used in both mono-modal and multi-modal applications. The implemented framework makes use of the sum of squared differences similarity measure, as it is a mono-modal application.

2.3.3 Rigid and Non-rigid Transformation

The transformation model specifies how one image can be transformed to another. The most simple model assumes the transformation behaves *rigid* or *affine*. Rigid registration only compensates for overal differences in pose by global transformation and rotation. Affine registration also includes scaling and skew. It can register images whose contents can be assumed to be a rigid object. A typical example are the human skull and bones, as they usually do not change shape in a healthy person. However, most of the human body cannot be considered as a rigid (or affine) transforming object and therefore many registration applications require a *non-rigid* transformation model for correct registration.

Non-rigid registration involves finding a field of displacement vectors that map each voxel from the moving image into the corresponding point in the reference image. It allows displacement of individual voxels such that local transformation between both images can be corrected for. While the affine transformation model requires no more than 12 degrees of freedom (DOFs), non-rigid registration involves a much larger number of DOFs to make the displacement field sufficiently flexible to recover local shape variability.

However, not all transformations are physically feasible or realistic and the image in-

tensity information may be insufficient to unambiguously define such a voxel-scale transformation field. Therefore, regularisation of the transformation field is used to impose local consistency or smoothness on the transformation and to propagate the results from areas with salient registration evidence into areas where registration features are largely absent [26, 40].

Such smoothness properties can be enforced in deformable registration algorithms by different means. For instance, a transformation model such as that of free-form deformations (FFD) can be chosen that inherently generates smooth deformations. In addition, a regularization strategy can be used that penalizes unlikely deformations and favor smooth candidates [44].

Given that the corpus callosum can change in shape and size if the patient has MS, a free-form deformation transformation model will be used in this thesis.

2.3.3.1 Free-Form Deformation

The origins of free-form deformation can be traced back to the area of computer aided design [2,45], but it can also be used in medical image analysis [41]. We have chosen for an FFD model based on B-splines. The basic idea of FFDs is to deform an object by manipulating an underlying mesh of control points. The resulting deformation controls the shape of a 3-D (or 2-D) object and produces a smooth and C^2 continuous transformation [41]. In contrast to thin-plate splines or elastic-body splines, B-splines are locally controlled, which makes them computationally efficient even for a large number of control points. In particular, the basis functions of cubic B-splines have limited support, meaning that changing a certain control point only affects the transformation in the local neighborhood of that control point.

2.3.4 Dimensionality

Image registration can be done using different image dimensions. These dimensions can be purely spatial, but time can also a added as a dimension. In either case, the problem can be further categorized depending on the number of spatial dimensions involved. Most current papers focus on 3-D-3-D registration of two images, where time is not involved. This type of registration normally applies the registration of two tomographic data sets. 2-D-2-D registration may apply to seperate slices from tomographic data. Compared with 3-D-3-D registration, 2-D-2-D registration is far less complex in terms of number of parameters and the volume of data which makes it faster than its 3-D-3-D registration counterpart. We reserve the 2-D-3-D registration for the direct alignment of a single tomographic slice to spatial data. Since most 2-D-3-D applications concern intra-operative procedures, they are heavily time-constrained and consequently have strong focus on speed issues connected with the computation time and the optimization step [30].

This thesis extracts a plane from a given MRI volume and registers a template image onto it. This means that the implemented registration framework deals in 2-D-2-D image registration.

2.3.5 Optimization

Optimization techniques are used to obtain the optimum transformation parameters required for aligning the images, or in other words: it defines how the transformation parameters are adjusted to improve the image similarity. A lot of methods have been proposed for function optimization (finding the minimum or maximum value of a function), with varying complexity. Good optimization algorithms determine the transformation parameters reliably and quickly. In non-rigid registration applications choosing or designing an optimize can be difficult because the more "non-rigid" (or flexible) the transformation model, the more parameters are generally required to describe it. Due to the local minima problem there is more chance of choosing a set of parameters which result in a good overall result, but not the optimal one [31].

The transformation parameters are computed either directly or searched for. This last method finds the optimal transformation parameters by finding an optimum of some cost function defined on the parameter space. The cost function is the similarity measure, discussed in Section 2.3.2.

Many optimization methods employ the first derivative of the cost function, to increase robustness and to reduce the number of iterations. However, the calculation might require a substantial computational cost [26]. In our case, part of the calculations required to obtain the first order derivatives overlaps with the calculations required for the functional value and, therefore, the extra cost is limited.

The simplest classes of multidimensional optimisation methods with calculation of the first derivative use steepest descent. Starting from a given position, these methods will look for the optimum along the direction of the derivative. More advanced methods, like quasi-Newton methods, will gradually build up the second derivative (or Hessian).

The robustness and speed of the chosen optimization method can be further increased by using a hierarchical multiresolution approach [26, 41]. Using this approach, several optimization stages are performed, while in each stage the complexity of the problem are gradually increased. Starting from a downscaled images and a coarse transformation model, the images are gradually upscaled and the transformation model is refined until the required accuracy is reached.

2.4 Method Overview

A framework was created that will identify and extract the plane that contains the corpus callosum cross-section with minimum area, using CUDA to speed up the registration part

of the framework. Normally, such a technique would require an MRI volume with a pre-segmented corpus callosum bridge, which is typically not readily available. To that end, the framework will segment the corpus callosum area embedded in the extracted plane. The corpus callosum is segmented by deformably registering a 2-D template containing a segmented corpus callosum to the extracted plane. The corpus callosum area is then calculated from the integral of the determinant of the Jacobian of the displacement field. The corpus callosum area is a function of the plane extraction parameters, so we use an optimization algorithm to obtain the plane which contains the corpus callosum with minimal cross-sectional area.

In this thesis we will be using a non-rigid transformation model and a voxel-based similarity measure. The proposed framework will use *sum of squared difference* as voxel-based similarity measure, restricting the framework to handle only mono-modality problems. To be more specific, 2-D-2-D mono-modal registration will be performed. A *free-form deformation model*, using B-splines, will be used to model the non-rigid deformations, and *steepest descent* is the method used for optimization.

In order to improve robustness and speed of the framework, a hierarchical multiresolution approach is adopted. A Gaussian pyramid of both the fixed and moving image will be built that will contain the resampled versions of the images at decreasing resolutions. Starting with the pair of images at the lowest resolution, registration is performed using a coarse grid of control points. The registration results from a previous resolution level are used at the higher resolution level and the registration is run again, stopping only when the full image resolution is reached. Adopting this approach, large deformations can be recovered early at low resolution and more detailed deformations are observed at the increasingly finer resolution levels.

The framework for finding the plane with with minimal corpus callosum cross-sectional area and the image registration framework are both discussed in detail in the following chapters. They are also both evaluated and their results are properly discussed.

Part II

Background and Previous Work

Chapter 3

Background

In this chapter, a thorough background will be provided of the used technologies and applied algorithms. The core technology used in this thesis is image registration. In this thesis, this consists out of the free-form deformation model, a similarity measure for calculating the resemblance between two images and the calculation of displacement fields. Since the framework relies on B-splines for the free-form deformation model, it is interesting to study splines a little closer.

3.1 Compute Unified Device Architecture

CUDA stands for "Compute Unified Device Architecture" and it is the hardware and software architecture that enables NVIDIA GPUs to execute programs written in C/C++, Fortran, OpenCL, DirectCompute, and other languages. The CUDA Architecture includes a unified shader pipeline, allowing each and every arithmetic logic unit (ALU) on the chip to be marshaled by a program intending to perform general-purpose computations. Because NVIDIA intended this new family of graphics processors to be used for general purpose-computing, these ALUs were built to comply with IEEE requirements for single-precision floating-point arithmetic and were designed to use an instruction set tailored for general computation rather than specifically for graphics. Furthermore, the execution units on the GPU were allowed arbitrary read and write access to memory as well as access to a software-managed cache known as *shared memory*. All of these features of the CUDA Architecture were added in order to create a GPU that would excel at computation in addition to performing well at traditional graphics tasks [42].

A CUDA program calls parallel *kernels*. A *kernel* executes in parallel across a set of parallel threads. The programmer, or compiler, organizes their threads in *thread blocks*, which are in turn organized in *grids* of thread blocks. Each thread within a thread block executes an instance of the kernel. Such a thread has a thread identifier within its thread block, a program counter, registers, per-thread private memory, inputs and output

results. A thread block is a set of concurrently executing threads that can cooperate among themselves through barrier synchronization and shared memory. A thread block has a block ID within its grid. A grid is an array of thread blocks that execute the same kernel, read inputs from global memory, write results to global memory, and synchronize between dependent kernel calls. In the CUDA parallel programming model, each thread has a per-thread memory space used for register spills and function calls. Each thread block has a per-block shared memory space used for inter-thread communication, data sharing, and result sharing in parallel algorithms. Grids of thread blocks share results in Global Memory space after kernel-wide global synchronization [35].

Next, a simple example will be given to show the advantages that CUDA can offer. Consider the code in Listing 3.1, run on CPU, that will make the sum of two vectors, A and B, and store the result in C. The same functionality can also be run on a GPU, as seen in Listing 3.2. There is no reason in particular why one needs to do this. The intention was to show how a particular operation, like the addition of two vectors, can be implemented on a graphics processor.

Listing 3.1: Example CPU code

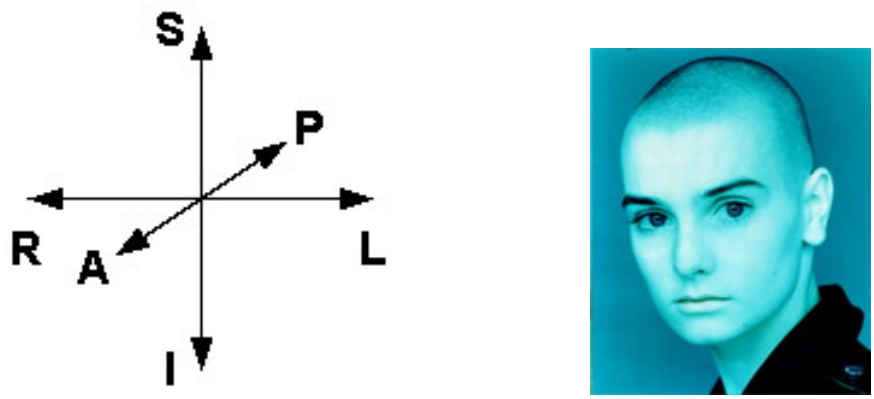
```
void add(int *a, int *b, int *c, int N)
{
    int tid = 0;
    while(tid < N)
    {
        c[tid] = a[tid] + b[tid];
        ++tid;
    }
}
```

Listing 3.2: Example GPU code

```
__global__ void add(int *a, int *b, int *c, int N)
{
    int tid = threadIdx.x;
    if(tid < N)
        c[tid] = a[tid] + b[tid];
}
```

3.1.1 Hardware Execution

CUDA's hierarchy of threads maps to a hierarchy of processors on the GPU; a GPU executes one or more kernel grids; a streaming multiprocessor (SM) executes one or more thread blocks; and CUDA cores and other execution units in the SM execute threads. The SMs, in the Fermi architecture (which was used for this thesis) executes threads in groups of 32 threads, which is called a warp. While programmers can generally ignore a



(a) Orientations of the human brain: **R**ight, **L**eft, **A**nterior, **P**osterior, **I**nferior and **S**uperior

(b) Head orientation reference

Figure 3.1: Medical Image Orientation

warp execution for functional correctness and think of programming one thread, they can greatly improve performance by having threads in a warp execute the same code path and access memory in nearby addresses [35].

3.2 Medical Image Orientations

3.2.1 Spacial Coordinates

In dealing with MRI data, it's necessary to be familiar with conventions and terminology used to describe orientation. "Up", "down", "front" and "back" are not used when talking about medical images, because they have confusing meanings when dealing with patients in different orientations (e.g.: lying down). To talk about locations in space in the neighborhood of the brain, one must be able to talk more precisely about sets of axes, including which direction is positive, and the order in which they are going to be listed when describing a point's coordinates. There are three axes (see Figure 3.1a), which could be used in any order, and where either direction could be positive, giving a total of 48 possible axis schemes.

In MRI practice, are two schemes who are most popular: *RAS* and *LAS*, respectively Figures 3.3a and 3.3b. To describe a certain scheme of orientation, it is common to sum up three axis names. These three axes have a positive direction, but the ordering of the names is also important, for example: *LAS* is not the same as *ALS*. RAS is a right-handed coordinate system: the thumb points in the **R**ight direction, the index finger in the direction of the **A**nterior and the middle finger points in the direction of the **S**uperior,

whereas LAS is a left-handed coordinate system. This is significant when performing matrix and vector math, where a right-handed coordinate system is customarily used.

Using an X, Y, Z coordinate system, the RAS direction scheme is expressed as follows $+X = R$, $+Y = A$ and $+Z = S$. This means that the positive X, Y and Z axis fall together with the R, A, S directions specified by the RAS direction scheme.

There are also three possible names for slice planes, but these do not specify the directions of their axes. They are suggestive, but insufficient to describe the order of voxels in a file:

- **Axial**, which is a R-L x A-P plane;
- **Coronal**, which is a R-L x S-I plane;
- and **Sagittal**, which is a A-P x S-I plane.

Figure 3.2 aids in visualizing the different names for slice planes.

3.2.2 Voxel Ordering

A number of MRI file formats (e.g.: Analyze, AFNI, NIfTI, ...) store voxel intensities as a stream of intensity numbers into a file in some agreed-upon manner. These formats require recording a number of characteristics of the image file, including voxel order, other attributes relating to conditions of image acquisitions, processing steps that have been performed, and so on. In general terms, voxels are stored in sequence along a row, one row after another, one slice after another. Like the case of the three orientation axes, there are again 48 different storing order possibilities. For RAS, this means voxels are ordered from left to right to form a row and from posterior to anterior to form a slice. Slices are stores from inferior to superior. The difference between RAS and LAS is that to form a row, the intensities are stored from right to left while the rest is the same as in RAS [62].

The voxel-ordering and spatial coordinates used in this thesis are discussed in Section 5.1.3.

3.3 NIfTI

The use of digital tools has proven useful at all stages of neuroimaging, as previously mentioned in Section 2.3. It allows scientists to control highly sophisticated imaging instruments and to make sense of the vast amounts data generated by them. While these tools are paramount to taking full advantage of the promise of neuroimaging, current tools have been developed piecemeal, by scientists who are interested in answering particular neuroscience questions rather than in producing software products that are optimized for meeting the many and varied needs of the broader research community. It is, therefore,

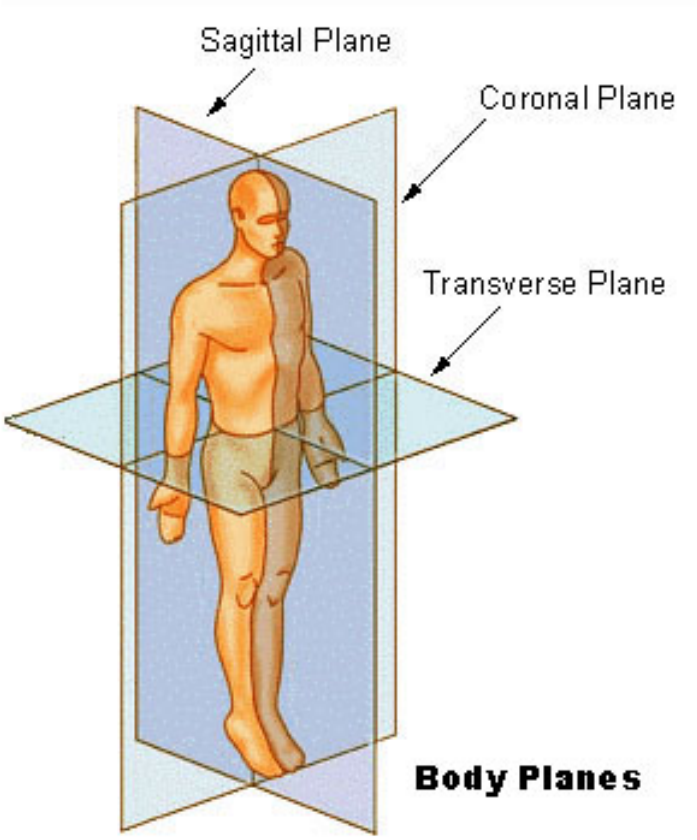
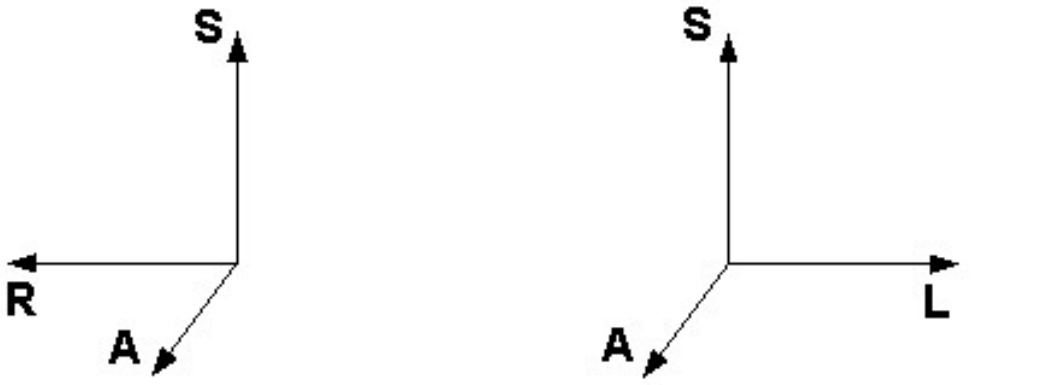


Figure 3.2: Different plane names



(a) **R**ight **A**nterior **S**uperior (b) **L**eft **A**nterior **S**uperior

Figure 3.3: The two most popular orientation schemes

not surprising that many of these tools are not as robust, generally useful, or easy to use. While a handful of neuroimaging tools are suitable for general use, many important tools are not widely available. Furthermore, even those that are in general use make varying assumptions, use different algorithms, or implement similar algorithms in different ways.

This Tower of Babel problem raises important concerns about the compatibility between different tools and it also limits the ability of scientists to rigorously compare their findings. The **N**euroimaging **I**nformatics **T**echnology **I**nitiative (NIfTI) is meant to work with the tool-user and tool-developer communities to address these needs. They provide coordinated and targeted service, training, and research to speed the development and enhance the utility of tools related to neuroimaging [20].

3.3.1 NIfTI-1 Data Format

NIfTI-1 is adapted from the widely used Analyze 7.5 file format. The hope is that older non-NIfTI-aware software that uses the ANALYZE 7.5 format will still be compatible with NIfTI-1. NIfTI-1 uses the "empty space" in the ANALYZE 7.5 header to add several new features. For more information about these new features, we refer to the website[1]. One of the more important features of the NIfTI-1 data format is the standardized way to store datasets, which resolves a lot of problems introduced in Section 3.3. The MRI volumes used in this thesis have all been stored in NIfTI-1 files. This guarantees us that the to be used data is not ambiguously defined.

3.4 Similarity Measures

As described in Section 2.3.2, there are two main approaches for measuring similarity: *feature-based* and *voxel-based* similarity measures. Feature-based registration approaches usually utilise points, lines or surfacesand aim to minimize the distance between the corresponding features in the images, while voxel-based approaches aim to measure the degree of shared information in the images their intensities. Over the last decade, voxel-based similarity measures have become the method of choice for measuring image alignment, largely due to their robustness and accuracy [40].

Before proceeding, first some terminology is required. From this point, we will call the fixed image "I_f" and the moving image "I_m". Each of these images will be two dimensional, because the fixed image is a two-dimensional plane extraced from a 3-D volume "V" and the moving image is the template which will be transformed to resemble the fixed image. We denote the domain of the image volume with dimensions (X, Y, Z) as $\Theta = \{(x, y, z)|0 \leq x \leq X, 0 \leq y \leq Y, 0 \leq z \leq Z\}$. The 2-D images, with dimensions (X, Y) will be denoted as follows: $\Omega = \{(x, y)|0 \leq x \leq X, 0 \leq y \leq Y\}$. When a certain certain pixel is discussed, it will be denoted by $\mathbf{p}$.

[1] `http://nifti.nimh.nih.gov/nifti-1`

One of the simplest similarity measures is the *sum of squared differences* (SSD) between images,

$$SSD(I_f, I_m) = \frac{1}{N} \sum_{\mathbf{p} \in \Omega} (I_f(\mathbf{p}) - I_m(\mathbf{T}(\mathbf{p})))^2, \tag{3.1}$$

which is minimized during registration. In Equation 3.1, N is the total number of pixels and $\mathbf{T}(\mathbf{p})$ is a transformation function that maps a pixel $\mathbf{p}$ to its new position [16, 18].

Another frequently used similarity measure, is the slightly modified SSD version called *sum of absolute differences* (SAD). When a small number of pixels are expected to have very large intensity differences between the images I_f and I_m, SSD falls short and one can apply SAD to gain better results. This problem might arise, for example, if contrast material is injected into the patient between the acquisition of images or if the acquisition of images are acquired during an intervention and instruments are in different positions relative to the subject in the two acquisitions. SAD takes the follow mathematical form:

$$SAD(I_f, I_m) = \frac{1}{N} \sum_{\mathbf{p} \in \Omega} \mid I_f(\mathbf{p}) - I_m(\mathbf{T}(\mathbf{p})) \mid . \tag{3.2}$$

These similarity measures, SSD and SAD, can be used for mono-modal registration. With multi-modal registration the situation is quite different. There is, in general, no simple relationship between intensities in images of different modalities. Here, the intensity mapping function is a complicated function and no simple arithmetic operation on the pixel values is going to produce a single derived image from which we can quantify misregistration.

It can be useful to think of image registration as trying to maximize the amount of shared information in two images. In a very qualitative sense, we might say that if two images of the head are correctly aligned then corresponding structures will overlap, so we will have two ears, two eyes, one nose and so forth. When the images are out of alignment, we will have duplicate versions of these structures from the two input images [18].

Two different groups, Collignon and Maes at KU Leuven, Belgium and Viola and Wells [60, 61] at the Massachusetts Institute of Technology, Cambridge, almost simultaneously but independently of each other, introduced Maximization of Mutual Information (MMI) of image intensities as a new registration criterion. Mutual Information (MI), or relative entropy, is a basic concept from information theory, which can be considered a non-linear generalization of cross-correlation. MI measures the statistical dependence between two random variables or the amount of information that one variable contains about the other [8]. The MMI registration criterion postulates that the MI of the image intensity values of corresponding pixel pairs is maximal if the images are geometrically aligned [29]. The mutual information of two images I_f and I_m is

$$MI(I_f, I_m) = H(I_f) + H(I_m) - H(I_f, I_m), \tag{3.3}$$

with $H(I_f)$ and $H(I_m)$ being the entropy of I_f and I_m respectively and $H(I_f, I_m)$ being their joint entropy. Entropy is a known to be a measure of the amount of uncertainty about a certain variable. The outcome of the MI calculation can be interpreted as follows:

- If the two images I_f and I_m are independent of each other, $MI(I_f, I_m)$ will equal to zero, meaning that $H(I_f) + H(I_m) = H(I_f, I_m)$;
- If the two images are one-to-one related, $MI(I_f, I_m) = H(I_f) = H(I_m)$;
- If the result is somewhere between the first two conditions, the two images I_f and I_m share some information.

Since this thesis handles only mono-modal registration problems, the complex MI and MMI similarity measures are not necessary to be implemented. SSD is being used for measuring similarity between the extracted plane and the given 2-D template image.

3.5 Splines and B-splines

In most image processing applications, the pictures to be manipulated are represented by a set of uniformly spaced sampled values. Finding a general mechanism for switching between the continuous and discrete signal domains is one of the fundamental issues in signal processing [58, 59]. It is a problem that arises during the acquisition process where an analog signal is to be converted into a sequence of non-integer numbers. Although most processing algorithms are derived within a purely discrete framework, there are a variety of problems best formulated by considering an image as a real-valued function $f(x, y)$. When one wants to digitalize a signal, Shannon's sampling theory is applied, describing an equivalence between a band-limited function and its equidistant samples taken at a frequency that is superior or equal to the Nyquist rate [48]. Although this theory has had an enormous impact on the signal processing community, it stil has a number of problems. One of those problems is that it relies on the use of ideal filters, which are devices not commonly found in nature. Also, the band-limited hypothesis is in contradiction with the idea of a finite signal. There are other problems, but that would take us too far from topic.

Unser provided arguments in favor of an alternative approach that uses *splines*, which is equally justifiable on a theoretical basis, and which offers many practical advantages. An interesting note: splines are slightly older than Shannon's sampling theory. They were first described in 1946 by Schoenberg, where he laid the mathematical foundations for the subject [43]. He showed how one could use splines to interpolate equally spaced samples of a function and he also introduced B-splines, which are used in this thesis.

Splines are piecewise polynomials with pieces that are smoothly connected together. These joining points are called *knots*. For a spline of degree n, each segment is a polynomial of degree n and therefore, $n+1$ coefficients are needed. There is an additional smoothness

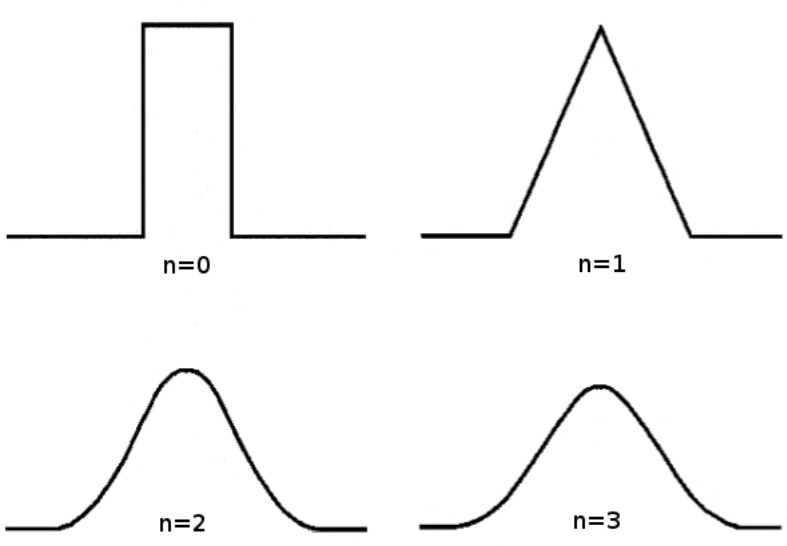


Figure 3.4: Centered B-splines of degree 0 to 3

constraint that imposes the continuity of the spline and its derivative up to order $n-1$ at the knots, so that, effectively, there is only one degree of freedom per segment. Schoenberg states that these splines are uniquely characterized in terms of a B-spline expansion

$$s(x) = \sum_{k \in Z} c(k)\beta^n(x-k), \tag{3.4}$$

which involves the integer shifts of the central B-spline of degree n denoted by $\beta^n(x)$. The parameters of this model are the B-spline coefficients c(k). B-splines, that will be defined below, are symmetrical, bell-shaped functions constructed from the $n+1$-fold convolution of a rectangular pulse β^0:

$$\beta^0(x) = \begin{cases} 1, & -\frac{1}{2} < x < \frac{1}{2} \\ \frac{1}{2}, & |x| = \frac{1}{2} \\ 0, & \text{otherwise} \end{cases} \tag{3.5}$$

$$\beta^n(x) = \underbrace{\beta^0 \times \beta^0 \times \ldots \times \beta^0}_{(n+1) \text{ times}}. \tag{3.6}$$

The B-splines of degrees 0 to 3 are shown in Figure 3.4. Of all the possible polynomial splines, cubic splines tend to be the most popular in applications [58]. In this thesis, cubic B-splines will be used within the free-form deformation model. Cubic B-splines can be

represented as follows:

$$\beta_l^3(t) = \begin{cases} \beta_0^3(t) & = (1-t)^3/6, \\ \beta_1^3(t) & = (3t^3 - 6t^2 + 4)/6, \\ \beta_2^3(t) & = (-3t^3 + 3t^2 + 3t + 1)/6, \\ \beta_3^3(t) & = t^3/6. \end{cases} \tag{3.7}$$

3.6 Image Warping

Similarity measures have been covered, showing how they are used for calculating how much two images resemble each other. In Section 5.2, the FFD transformation model will be discussed, giving an overview of how displacement field will be calculated with the B-splines introduced in Section 3.5, that is used for transforming the moving image I_m so it will resemble the fixed image I_f. The actual transformation is done by means of *image warping*.

Once a transformation has been computed, information is required on how to move each individual pixel in the image that is being transformed. A *displacement field* stores this information, relating the positions of pixels between the fixed and moving images. This displacement field is a function $u : \Omega \to \mathbb{R}^d$ on the image domain Ω, where d is the dimensionality. In Equation 3.1, we already introduced the transformation function $\mathbf{T}(\mathbf{p})$. The general form of a transformation function is the following:

$$\mathbf{T} : \Omega \to \Omega; \qquad \mathbf{T}(\mathbf{p}) = \mathbf{p} + u(\mathbf{p}). \tag{3.8}$$

This transformation function transforms pixel coordinates $\mathbf{p}$ in the fixed image I_f to coordinates in the moving image I_m by means of an identity mapping and the corresponding value of the displacement field. How the displacement field u is computed, depends on the properties of particular applications. In this thesis, a B-spline based transformation function is employed.

There are two ways of how image warping can be achieved. One method is that for each position $\mathbf{p}$ of the template image, the corresponding intensity value is stored in the new image. This process is referred to as *forward* warping, since pixels are moved "forward" from the coordinate frame of the old image (I_m) to the new image (I'_m). A problem with this method is that it can be that not every pixel in the new image I'_m is assigned a value and some pixels can be assigned several times.

The second method, called *backward* warping, eliminates this problem. The main difference is that now for every pixel of the new image I'_m a coordinate in the original image is computed, where its intensity value originates from. Backward warping calculations can produce non-integer values. In that case, an interpolation method must be used to obtain intensity values at the coordinates between pixels (Section 3.7) [44]. In this thesis, bilinear and trilinear interpolators will be used for interpolation purposes, as discussed in Section 3.7.

3.7 Interpolation Methods

The problem of constructing a continuously defined function from given discrete data is unavoidable whenever one wishes to manipulate the data in a way that requires information not included explicitly in the data. In this age of ever-increasing digitization in the storage, processing, analysis, and communication of information, it is not difficult to find examples of applications where this problem occurs. The relatively easiest and in many applications often most desired approach to solve the problem is *interpolation*, where an approximating function is constructed in such a way as to agree perfectly with the usually unknown original function at the given measurement points. The word "interpolation" originates from the Latin verb *interpolare*, a contraction of "inter", meaning "between", and "polare", meaning "to polish". That is to say, to smooth in between given pieces of information. [32].

Interpolation is a basic tool used extensively in tasks such as zooming, shrinking, rotating, and geometric corrections. They can be considered as *resampling* methods. Fundamentally, interpolation is the process of using known data to estimate values at unknown locations [15].

As previously stated in Section 3.6, a displacement field is calculated and used to find where a certain pixel's intensity value originated from. These coordinates can be non-integer values and therefore an interpolation method must be used to obtain the intensity values at the coordinates between pixels. Since this thesis employs a 2-D-2-D registration method for solving a mono-modal problem, it is sufficient to use *bilinear interpolation* as an interpolation method.

Bilinear interpolation is an extension of linear interpolation for interpolating functions of two variables on a regular 2-D grid. The linear interpolar that is used in bilinear interpolation takes the following form:

$$LI(u_1, u_2, w) = u_1(1 - w) + u_2 w. \tag{3.9}$$

The resulting value of $LI(u_1, u_2, w)$ is calculated from two known data values u_1 and u_2, and the value w signifies a weight.

Consider a 2-D image I with dimensions (U, V) as $\Omega = \{(u, v)|0 \leq u \leq U, 0 \leq v \leq V\}$, and a point $I_{x,y}$ with non-integer coordinates. To calculate the intensity value for $I_{x,y}$, see Figure 3.5, bilinear interpolation uses the four neighboring pixels to approximate the point with non-integer coordinates using the following steps:

$$\begin{aligned} t &= LI(I_{u,v}, I_{u+1,v}, w_1) && w_1 = x - u \\ s &= LI(I_{u,v+1}, I_{u+1,v+1}, w_1) \\ I_{x,y} &= LI(t, s, w_2) && w_2 = y - v. \end{aligned} \tag{3.10}$$

The values w_1 and w_2 can be seen as the fractional parts of the coordinates x and y of point $I_{x,y}$.

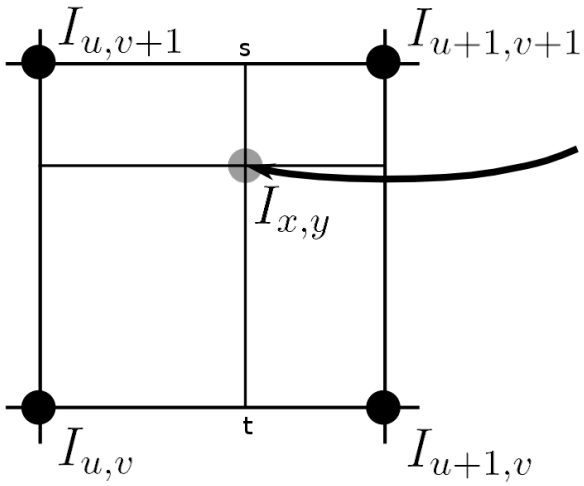


Figure 3.5: Calculating value for pixel $I_{x,y}$ by means of bilinear interpolation

Bilinear interpolation is not the only interpolator employed in this thesis. To extract a 2-D plane, with (some) non-integer coordinates, from a 3-D volume, a *trilinear interpolation* method is used. Like bilinear interpolation is an extension to linear interpolation, trilinear interpolation extends bilinear interpolation.

Consider a volume with dimensions (X, Y, Z) as $\Theta = \{(x, y, z)|0 \leq x \leq X, 0 \leq y \leq Y, 0 \leq z \leq Z\}$, and a point $P_{x,y,z}$ with non-integer coordinates. To calculate the intensity value for $P_{x,y,z}$, see Figure 3.6, trilinear interpolation uses the eight neighboring voxels[2] to approximate the point with non-integer coordinates using the following steps:

$$\begin{aligned}
r &= LI(I_{u,v,w}, I_{u+1,v,z}, f_1) && f_1 = x - u \\
s &= LI(I_{u,v,w+1}, I_{u+1,v,w+1}, f_1) \\
t &= LI(I_{u,v+1,w}, I_{u+1,v+1,w}, f_1) \\
u &= LI(I_{u,v+1,w+1}, I_{u+1,v+1,w+1}, f_1) \\
v &= LI(r, s, f_2) && f_2 = z - w \\
w &= LI(t, u, f_2) \\
p &= LI(v, w, f_3) && f_3 = y - v.
\end{aligned} \tag{3.11}$$

The values f_1, f_2, and f_3 can be seen as the fractional parts of the coordinates x, y, and z of point $p_{x,y,z}$.

[2]A voxel is a **vo**lumetric pi**xel**

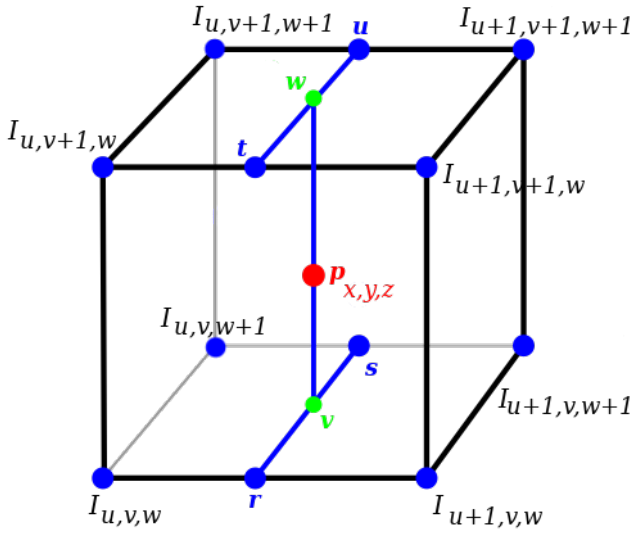


Figure 3.6: Calculating value for point $P_{x,y,z}$ by means of trilinear interpolation

Chapter 4

Previous Work and Related Research

In this section, we aim to give the reader an overview of related research that served as a theoretical and practical foundation for the work described in this thesis, but we also give some examples of applications and settings where deformable registration is used.

To date, there are multiple deformable registration algorithms proposed and validated. This includes thin-plate splines (1993) [4], viscous fluid registration (1996) [6], surface matching (1996) [57], finite-element models (1997) [33], spline-based registration (1997) [55], demons registration (1998) [56], and B-spline registration (1999) [41]. As one can see, there are a plethora of registration methods, but the choice of a certain image registration method for a particular application is largely unsettled [46]. The choice for a certain registration method can be determined by considering several factors, namely: the choice of similarity measure to be used, the transformation model, and the optimization process. Maintz and Viergever have made a survey of registration methods and their possible use in specific application domains [30].

Spline-based registration methods are currently very popular. This is because of their flexibility and robustness provide the ability to perform mono-modal and multi-modal registration. The appealing characteristics of both free-form deformation and spline-based methods are the most important reason why many studies have been conducted involving these techniques. As stated in Section 2.3.3.1, the origin of free-form deformation can be traced back to the area of computer aided design. In their paper, Sederberg and Parry described FFD as a method for sculpturing solid models [45]. They accomplished this deformation by manipulating a geometric model its surrounding space via a grid of control points. Szeliski and Coughlan noted that a dense deformation field can be interpolated from a coarse grid of moving control points [55]. A few years later, Rueckert et al. present a more specialized method using cubic B-splines curves to define a displacement field which maps voxels in a moving image to those in a reference image [41]. Each individual voxel movement between reference and moving image, is parameterized in terms of uniformly spaced control points that are aligned with the voxel grid. The displacement vectors are obtained via interpolation of the control point coefficients, using piecewise, continuous

B-spline basis functions.

Besides the popularity of spline-based registration methods and their potential to greatly improve the geometric precision for a variety of medical procedures, they are usually computationally intensive [47]. Shackleford points to reports of algorithms requiring hours to compute for demanding image resolutions [1, 39], depending on the specific algorithm implementation. To remedy these shortcomings, Shackleford has proposed a GPU-based image registration design to accelerate both the B-spline interpolation problem as well as the cost-function gradient computation that makes use of coalesced accesses to the GPU global memory and an efficient use of shared memory.

Applications benefiting from deformable registration include interventional procedures such as image-guided surgery where deformations of the brain can be tracked during surgery. This gives a surgeon better visualization and reduces the amount of the unresected tumor [17]. Another possible application may be in the domain of image-guided radiotherapy, where deformable registration can improve geometric and dosimetric accuracy of radiation treatments [63].

This thesis uses deformable registration for extracting a plane with minimal corpus callosum area from a given medical image, for the purpose of detecting changes in the size of the corpus callosum and correlating it with MS progression. Previously, clinical studies have mostly focused on tracking and analyzing the changes in the corpus callosum by measuring the cross-sectional area of the corpus callosum and correlating in with MS progression [50, 51]. This was done through selecting a 2-D measuring plane from a MRI volume, typically the mid-sagittal plane, and measuring the area of the corpus callosum cross-section in this plane. This method is highly dependent on the accurate selection of the measuring plane, as it influences the measurement of the corpus callosum area. As longitudinal studies require patients to undergo several scans over a long period of time, factors such as the error in positioning of the human head in the MRI scanner between two scans can potentialy be a source of error in the selection of the mid-sagittal plane and consequently the measurement of the corpus callosum area. Ishaq proposed a method for finding a plane with minimal corpus callosum area and tries to mitigate the previously mentioned errors [21].

Part III

Framework Algorithms Description and Implementation Details

Chapter 5

Algorithms

5.1 MCAP Extraction and CC Segmentation

The method proposed by Ishaq identifies accurately the MCAP and simultaneously segments the corpus callosum cross-section embedded on it. Normally, such a technique requires an MRI volume with a pre-segmented corpus callosum bridge, which is typically not readily available.

His solution addressed this problem by equipping the optimization based plane extraction framework with a mechanism for continuously segmenting the corpus callosum cross-section embedded in the extracted plane. The corpus callosum is segmented by *deformably registering* a 2-D template containing a segmented corpus callosum to the extracted plane. The corpus callosum area is then calculated from the *integral of the determinant of the Jacobian of the displacement field*. This corpus callosum area is a function of the plane extraction parameters. Optimization of this function obtains the plane with minimum corpus callosum area.

The following paragraphs will give a detailed description of how the framework finds the plane with minimum corpus callosum area. The steps proposed by Ishaq were implemented with some key differences.

5.1.1 Method Overview

The goal is to extract a plane from a MRI volume which embeds the corpus callosum cross-section with the minimum area. This cross-sectional area of the corpus callosum will be denoted as A_{cc}, and the plane which embeds this minimal area will be denoted as P_{ext}. In other words, this means that the area A_{cc} can be written as a function with P_{ext} as its parameter:

$$A_{cc}(P_{ext}). \tag{5.1}$$

This function must be minimized with respect to the parameter P_{ext}. In order to optimize Equation 5.1, the value of A_{cc} must be calculated, what entails taking the folling three steps:

1. Extract a 2-D slice specified by the parameter P_{ext} from an MRI volume. This entails resampling a plane in the volume. The orientation and position of this plane are parameterized over two rotations (R_x, R_y) round the X- and Y-axes respectively, and one translation (T_z) along the Z-axis. Together, these parameters form the parameter
$$P_{ext} = (R_x, R_y, T_z). \tag{5.2}$$
This step will be described in more detail in Section 5.1.3;

2. Segment the corpus callosum cross-section in the slice extracted in step 1. This segmentation is performed by registering a 2-D template with a segmented corpus callosum to the extracted slice. In this thesis, the registration is done with a free-form deformable transformation model and with the sum of squared differences as similarity measure, as discussed in Sections 3.4 and 3.5 respectively. The results of this step are the B-spline transformation coefficients (see Section 5.1.4);

3. Calculate the A_{cc} area, given the B-spline coefficients obtained in step 2. The area A_{cc} is calculated by integrating the determinant of the Jacobian of the displacement field over all the points on the template which lie *inside* the corpus callosum [9]. The details of this particular step are discussed in Section 5.1.5.

All these steps together calculate the area of A_{cc} with respect to the parameter P_{ext}. Given Equation 5.2, Equation 5.1 can be rewritten as follows:
$$A_{cc}(R_x, R_y, T_z). \tag{5.3}$$
The details of the area optimization are given in Section 5.1.6.

Ishaq mentions that one can also propose an alternative framework which segments the whole corpus callosum bridge in a given volume, by registering it in 3-D to a pre-segmented template volume and then finding MCAP by slicing the corpus callosum bridge and measuring the corpus callosum area. However, the corpus callosum is mostly a featureless organ. Therefore, such a 3-D registration can potentially cause anatomically different slices from the template and target corpus callosums to map to each other, while this is unlikely to happen in the current solution.

5.1.2 Template Preperation

Since the template is a vital part in finding the MCAP, it is manually extracted from the MRI volume. Using Slicer[1], the right slice is sought for and saved as a NIfTI-image.

[1] 3D Slicer is a free, open source software package for visualisation and image analysis. `http://www.slicer.org`

Together with this image, the corpus callosum is segmented and this result, being a binary map, is also saved as a Nifti-image.

The slice to be used as template, is the central sagittal slice, because it is hypothesized to lie not far in 3-D from the sought for objective slice with the minimal cross-sectional corpus callosum area, the MCAP. Since the corpus callosum is not too different from the corpus callosum segmented from the template, smooth and localized deformations are ensured during the registration.

5.1.3 Slice Extraction

The extraction of a plane is done by positioning a 2-D slicing plane in the MRI volume and sampling the intensities on the plane. Sampling is done by trilinear interpolation, as discussed in Section 3.7. This position of the slicing plane in the volume is dependent on its orientation and distance from the origin of the reference coordinate system. The coordinate system maps the anterior, superior and left directions to the positive X, Y, and Z axes respectively and is different from the RAS and LAS coordinate systems introduced in Section 3.2.1.

The slicing plane orientation can be specified by two angles between its normal vector and two of the reference axes. The center of the slice plane is taken as the center of the rotation. For an initial sagittal plane, the distance from the origin is specified by the distance T_z, expressed in millimeters and introduced in Equation 5.2, between the center of the slice and the origin of the reference coordinate system, parallel to the Z axis. Next, the orientation is specified by the rotations R_x and R_y, also introduced in Equation 5.2, around the X and Y axes respectively. Both rotations are expressed in degrees.

Before these rigid transformations are performed, the center of the coordinate system is also the center of the slicing plane. These three parameters are used to extract a 2-D slice, which can be oblique or orthogonal, out of a 3-D MRI volume. Image intensities are approximated using trilinear interpolation if the do not lie at sample grid points. The extracted slice can also be expressed as a function *ExtractSlice*:

$$I(x', y') = ExtractSlice(V_{\text{MRI}}, R_x, R_y, T_z), \tag{5.4}$$

where V_{MRI} denotes the MRI volume, x' and y' denote the local 2-D coordinates of the extracted slice and R_x, R_y and T_z are the rigid transformation parameters introduced in Equation 5.2.

5.1.4 Slice Registration

The next step is to segment the corpus callosum from the extracted slice. This is done by deformably registering the presegmented template to the extracted slice from Section 5.1.3. As previously stated, the registration is done with a hierarchical multiresolution

approach. This means that the registration of the extracted slice and the template is first performed on a coarse level, where both images their dimensions are reduced by a factor of two. Starting from a downscaled image and a coarse transformation model, the image is gradually upscaled and the transformation model is refined until the required accuracy is reached. Adopting this approach, large deformations can be recovered early at low resolution and more detailed deformations are observed at the increasingly finer resolution levels.

As mentioned in Section 3.5, the 2-D-2-D registration of the template image to the extracted slice is done by the free-form deformable transformation model, using B-splines. The control points are arranged in a regular and uniform grid. Bilinear interpolation is used for approximating the transformation of pixels lying between control points. See Section 3.7 for more information on (bilinear) interpolation. For a similarity measure, the sum of squared differences (Section 3.4) is used, because of the mono-modal nature of the registration problem. The optimization is handled by employing the *gradient descent* method, that will be discussed in length in Section 5.2.4. The result of this registration process is a set of B-spline coefficients, specifying the spatial transformation of the template.

5.1.5 Area Calculation

As denoted in Sections 3.5 and 3.6, the B-spline coefficients from the registration process are used to generate a displacement field, which can be called $\overrightarrow{V}$. This displacement vector field is used to find for every pixel (x^*, y^*) of the template image a coordinate (x', y') in the extracted slice , that is,

$$(x', y') = (x^*, y^*) + \overrightarrow{V}(x^*, y^*). \tag{5.5}$$

The determinant of the Jacobian J of the displacement vector field $\overrightarrow{V}$ is used for finding the area of the corpus callosum, returning a scalar field $S = |J(\overrightarrow{V})|$. When one calculates the integral of S over all the points in the template which lie inside of the corpus callosum, one finds the area A_{cc} of the corpus callosum of the extracted slice. The integral is written as follows:

$$A_{cc} = \int_{(x^*,y^*)\in CC} S(x^*, y^*)dx^*dy^*. \tag{5.6}$$

Ishaq notes that the displacement vector field $\overrightarrow{V}$ maps each point (x^*, y^*) to a point (x', y') in "real space", including non-integer locations. Therefore, the corpus callosum area calculated through the Jacobian of the displacement vector field is not affected by the discretization of the position of the corpus callosum boundary points and is an accurate way of measuring the corpus callosum area after the image registration.

5.1.6 Area Optimization

Using the parameters R_x, R_y, and T_z of the extracted slice, the corpus callosum area is minimized. In each area optimization iteration, the corpus callosum area A_{cc} is calculated with respect to these R_x, R_y, and T_z parameters that are used to extract a slice from the MRI volume. As previously described, the template is deformably registered to the extracted slice to segment this corpus callosum area. The R_x, R_y, and T_z parameters are optimized by means of the gradient descent optimization strategy.

Ishaq mentions that the upper and lower bounds for these three parameters are -2.0 and 2.0 (R_x and R_y are expressed in degrees and T_z in millimeters) respectively, because the corpus callosum bridge is well defined in this interval. Beyond this interval, the corpus callosum bridge starts to diffuse into the brain hemispheres. During the initialization step, the sagittal plane in the center of the volume is used, that is, the plane extracted with the following parameters: $R_x = 0$, $R_y = 0$, and $T_z = 0$.

5.2 Free-Form Deformable Registration Using B-splines

As discussed in Section 2.3, image registration exists out of three crucial components, namely: (1) a **transformation model** which relates the fixed and moving images, (2) a **similarity measure** which measures the similarity between fixed and moving image, and (3) an **optimization technique** which determines the optimal transformation parameters as a function of the similarity function. So, image registration is an optimization problem, where a target function must be minimized. In this case, the target function is the similarity function. A set of parameters are manipulated in an iterative manner, resulting in an optimal, minimal solution so that the template image resembles the fixed image. In the following sections, image registration will be discussed in more detail. Schwarz has described a high level view of the basic steps needed to be taken to perform image registration [44]:

- **Initialization**. Since the free-form deformation model depends on a grid of control points, these must be evenly distributed across the 2-D surface. All the control points are set to zero, and the fixed and template images are scaled so their intensity values are in the range $[0, 1]$.

- **Iteration**. Starting from the initialized grid of control points from the previous step, a displacement field is calculated and applied to the template image. Next, the similarity measure, that acts as the target function and must be optimized, is evaluated based on the newly calculated image in comparison to the fixed image. The control points displacement is also computed; this is handled by steepest descent optimization. After the control points have been updated, the next iteration begins.

- **Termination**. If the target functions has reached a predefined threshold value, or if it converges, no more iterations are performed. The final configuration of the control points is returned as the optimal solution to the registration problem. The displacement field and the warped template image can be stored for further use. In case of a multiresolution approach, the control points displacement field is used to initialize the control points of the next level.

5.2.1 Configuration

When one wants to represent a signal or an image, one usually has the choice of two options. The first is to use an exact representation of the signal, by which $f(x, y)$ precisely interpolates the sampled values. The second is to use an approximate representation of the signal, in which the function parameters are determined by minimizing some measure of the discrepancy between pixel values and $f(x, y)$ at the grid points. This last approach usually has fewer DOFs than the previous one, which may make it more robust in the presence of noise [59]. As described in [25, 41], B-splines approximation is used as a transformation model for image registration. In this thesis, we use the FFD model as the transformation model which makes use of B-splines. B-splines were introduced in Section 3.5.

5.2.1.1 Control Points

In Section 2.3.3.1, when describing the free-form deformation model, it was made apparent that FFDs depend on a grid of control points, evenly distributed across a 2-D surface. This grid of control points is defined as follows. Let Φ denote a $n_x \times n_y$ mesh of control points $\phi_{i,j}$ of uniform spacing δ. This spacing can be set arbitrarily, but in this thesis, the spacing is kept the same in both dimensions. This results in an equidistant spacing of the control points across the 2-D surface. The dimensions n_x and n_y of the control point grid take the following form:

$$n_x = \left\lceil \frac{m_x}{s_x} + 3 \right\rceil, \qquad n_y = \left\lceil \frac{m_y}{s_y} + 3 \right\rceil, \tag{5.7}$$

where m_x and m_y are respectively the X and Y dimensions of the template image, and s_x and s_y are the control point spacing distances. s_x and s_y will both equal to the uniform spacing δ mentioned earlier.

5.2.1.2 Calculating the Displacement Field

Section 3.6 introduced image warping, which makes use of a displacement field for calculating the warped image. In the case of image registration, the template will be warped in each iteration, until it resembles the fixed image. The FFD transformation model used in

this thesis uses the grid of control points to calculate the displacement field for the template image. The FFD can be written as the 2-D tensor product of 1-D cubic B-splines

$$\mathbf{T}(x, y, \phi) = \sum_{l=0}^{3} \sum_{m=0}^{3} \beta_l(u)\beta_m(v)\phi_{i+l,j+m}. \tag{5.8}$$

The parameters i, j, u and v are described as follows

$$i = \left\lfloor \frac{x}{n_x} \right\rfloor - 1, \qquad j = \left\lfloor \frac{y}{n_y} \right\rfloor - 1, \tag{5.9}$$

$$u = \frac{x}{n_x} - \left\lfloor \frac{x}{n_x} \right\rfloor, \qquad v = \frac{y}{n_y} - \left\lfloor \frac{y}{n_y} \right\rfloor. \tag{5.10}$$

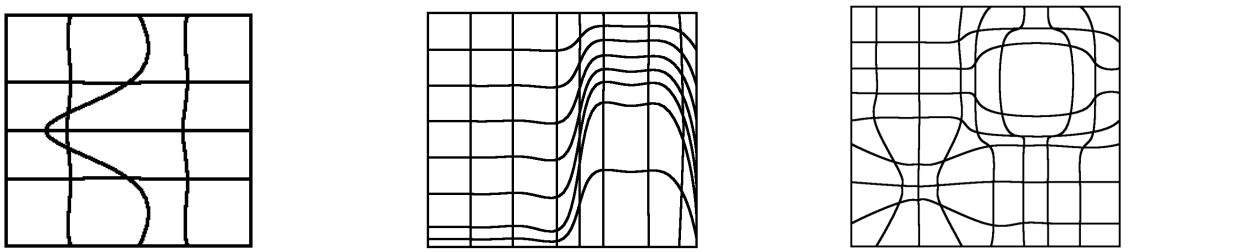
For more information about these four parameters, see Section 6.1.1. The summation in Equation 5.8 takes place over the control points in the neighborhood pixel $\mathbf{p}(x, y)$. Since cubic B-splines are defined by four pieces, see Equation 3.7, the neighborhood exists out of 16 control points. This explains the size of the grid of control points, since the rows and columns of control points outside the template image boundaries are required so that the control point neighborhood is defined for every pixel in the image. The β_l and β_m notations of Equation 5.8 represent the respectively l-th and m-th basis function of the B-spline, as defined in Equation 3.7. The control points Φ act as paramaters of the B-spline FFD and the degree of non-rigid deformation which can be modeled depends essentially on the resolution of the mesh of control points Φ. A large spacing of control points allows modeling of global non-rigid deformations, while a small spacing of control points allows modeling of highly local non-rigid deformations. At the same time, the resolution of the control point mesh defines the number of DOFs and consequently, the computational complexity.

This model has not only shown to be computationally more efficient than other alternatives due to the local support of B-splines, also B-splines are easily scalable and have good multiresolution properties [24, 26]. As previously stated in Section 2.3.5, to improve robustness and to alleviate some computational cost, a hierarchical multiresolution approach is handled in this thesis, in which the resolution of the control mesh is increased along with the image resolution, in a coarse to fine fashion.

5.2.2 Deformation Regularization

It is worth noting that not all types of deformations are physically plausible. Therefore, registration algorithms are often regulated using a technique that evaluates a given candidate deformation and penalizes it if it is "unregular". See Figures 5.1a, 5.1b and 5.1c for examples of irregular deformations. Usually in practice, these regularization techniques rely on rather simple mathematical properties of deformation. Most regularizers exploit

the smoothness of displacement fields, as shown by [41]. This displacement field is smooth if it has no "harsh jumps". In other words: the direction and magnitude of displacements in a certain neighborhood change gradually, not abruptly. Measuring a gradual change of displacement implies the use of derivatives of the displacement fields. The two most used methods for regularization are *diffusion* and *curvature* regularizers.



(a) Folding (b) Irregular (c) Expansion/compression

Figure 5.1: Implausible deformations

5.2.2.1 Diffusion

A diffusion regularizer stems from the physical world, namely the heat diffusion equation. This equation describes how heat is distributed in a given medium over time [34]. How heat from a static source is distributed over a cooler medium can be used as an analogy, as that local displacements are expected to spread over a certain region. The diffusion regularizer uses the first order derivatives of a displacement field and can be denotes as follows:

$$R_D(u) = \sum_{\mathbf{p}\in\Omega} \|\nabla u_x(\mathbf{p})\|^2 + \|\nabla u_y(\mathbf{p})\|^2, \tag{5.11}$$

where u_x is the x-component of the displacement field u, $\nabla = (\partial/\partial x, \partial/\partial y)^T$ is the gradient operator, $\mathbf{p}$ are the pixel coordinates, and $\|.\|$ is the Euclidean vector norm. This kind of regularizer is usually used in a global cost function which is to be minimized during optimization (see Section 5.2.3). This means that for a good displacement field u, $R_D(u)$ is minimal. This thesis will make use of a diffusion regularizer.

5.2.2.2 Curvature

An other regularizer is the curvature regularizer. This kind of regularization is based on second order derivatives [34]. The curvature regularization is achieved by adding the unmixed second partial derivatives $\Delta = \partial^2/\partial x^2 + \partial^2/\partial y^2$, where Δ is a Laplace operator. This function can also be written as follows:

$$R_C(u) = \sum_{\mathbf{p}\in\Omega} (\Delta u_x(\mathbf{p}))^2 + (\Delta u_y(\mathbf{p}))^2. \tag{5.12}$$

An important characteristic of a curvature regularizer is that it is invariant under affine transformations, which means that translations, rotations and scaling are not penalized.

5.2.3 Goal

The registration problem can be described as finding the optimal deformation so that the template image maps perfectly onto the fixed image. Since the deformation is calculated by means of the control points, one can deduce that the optimal control point configuration has to be found. This translates to optimizing a certain cost function

$$E(\phi) = S(\phi) + \alpha R(\phi), \tag{5.13}$$

which entails evaluating two terms. The first term, $S(\phi)$, is the similarity term which measures the similarity of two images. In our case, this is the *sum of squared differences* as described in Section 3.4. Equation 3.1 can be written as follows

$$S(\phi) = \frac{1}{N} \sum_{\mathbf{p} \in \Omega} (I_f(\mathbf{p}) - I_m(\mathbf{T}(\mathbf{p}, \phi)))^2, \tag{5.14}$$

where $T(\mathbf{p}, \phi)$ equals the one described by Equation 5.8 if you replace the term $\mathbf{p}$ by the pixel coordinates (x,y). The second term is called a *regularity term* and its function is to penalize control point displacements that potentially lead to implausible deformations, as discussed in Section 5.2.2. The weighting factor $\alpha \in \mathbb{R}$ governs the strength of the regulation. This thesis adopts the regularizer used by Schwarz [44], which essentially is a diffusion regularizer that is applied to the control points as opposed to the displacement field [41]. As the number of control points is typically orders of magnitude less than the number of elements in the displacement field, it makes computationally more sense to regularize the control points. The proposed regularizer can be described as follows:

$$R(\phi) = \frac{1}{N} \sum_{(i,j)} \|\nabla \phi_x(i,j)\|^2 + \|\nabla \phi_y(i,j)\|^2, \tag{5.15}$$

where N is the total amount of control points and ∇ denotes a discrete approximation of the gradient operator based on central differences. The latter will be further explained in Section 6.2.

5.2.4 Optimization

To describe optimization in general, we refer to Bertsekas [3]. Mathematical models of optimization can be generally represented by a *constraint set* X and a *cost function* f. The set of X consists of the available decisions x and the cost $f(x)$ is a scalar measure of

undesirability of choosing decision x. One wants to find an optimal solution decision so that

$$f(x^*) \leq f(x), \qquad \exists x^* \in X, \forall x \in X. \tag{5.16}$$

Optimization problems can either be *continuous* or *discrete*. Continuous problems are those where the constraint set X is infinite (or unconstrained) and has a "continuous" character. Examples of such problems are those where there are no constraints or where X is specified by some equations and inequalities. These problems are generally analyzed using the mathematics of calculus and convexity.

Discrete problems are those that are not continuous, usually because of the finiteness of the constraint set X. Typical examples of such problems are combinatorial problems, like scheduling, route planning and matching. Image registration is in need of continuous optimization methods, more specifically: a gradient descent optimization method is used, where one tends to mimimize a certain function $f(x)$.

A vector x^* is an *unconstrained minimum* of f if it is no worse than its neighbors; that is if there exists an $\epsilon > 0$ such that

$$f(x^*) \leq f(x), \qquad \forall x \text{ with } \|x - x^*\| < \epsilon. \tag{5.17}$$

Most of the interesting algorithms for this problem rely on an important idea, called *iterative descent* that works as follows: start at some point x^0 (an initial guess) and successively generate vectors x^1, x^2, ..., such that f decreases at each iteration, that is

$$f(x^{k+1}) < f(x^k), \qquad k = 0, 1, \ldots. \tag{5.18}$$

In doing so, we successively improve our current solution estimate and hope to decrease f all the way to its minimum. Most descent algorithms specify a stepsize α^k, so that

$$f(x^k + \alpha^k d^k) < f(x^k), \qquad k = 0, 1, \ldots. \tag{5.19}$$

The simplest and most famous of these methods is the method of *steepest descent*, also called *gradient descent*, first proposed by Cauchy in 1847 [53]. With gradient descent, d^k takes the following form:

$$d^k = -\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|}. \tag{5.20}$$

Choosing the stepsize wisely is important, for it has an effect on the correctness of the result but also on the computation time. In the simplest rule of this type, an intial stepsize s is chosen and if the corresponding vector $x^k + sd^k$ does not yield an improved value of f, that is, $f(x^k + sd^k) \geq f(x^k)$, the stepsize is reduced, perhaps repeatedly, by a certain factor until the value of f is improved. While this method often works in practice, it is theoretically unsound because the cost improvement obtained at each iteration may not be substantial enough to guarantee convergence to a minimum.

The Armijo rule is essentially the successive reduction rate just described, but suitably modified to eliminate the theoretical convergence difficulty. The following fixed scalars are chosen: s, β, and σ, with $0 < \beta < 1$, and $0 < \sigma < 1$. So we set $a^k = \beta^m s$, where m is the first nonnegative integer for which

$$f(x^k) - f(x^k + \beta^m s d^k) \geq -\sigma \beta^m s \nabla f(x^k) d^k. \tag{5.21}$$

In other words, the stepsizes $\beta^m s$, $m = 0, 1, \ldots$, are tried successively until the unequality in Equation 5.21 is satisfied. Usually σ is chosen close to zero, for example, $\sigma \in [10^{-5}, 10^{-1}]$. The reduction factor β is usually chosen from 1/2 to 1/10, depending on the quality of the initial step s. In this thesis the following values were used: $\sigma = 10^{-4}$, $\beta = 1/2$ and $s = 1$.

5.2.5 Multiresolution Approach

In Sections 2.3.5 and 5.1.4 we mentioned the use of a multiresolution approach. Rueckert proposed the use of multiresolution pyramids in order to achieve the best compromise between the degree of nonrigid deformation and computational cost. His method involved the increase of the resolution of the grid of control points, along with the image resolution, in a coarse to fine fashion. This method would recover global nonrigid deformations at a coarse level and local deformations at the finer levels [41]. The final deformation would be a combination of the control point configurations from all resolution levels. It is worth mentioning that the computational complexity increases with the control point grid resolution.

Implementing this method includes the creation of two image pyramids, one for the extracted slice and one for the template image, but also a way to adjust the resolution of the grid of control points. Therefore the multiresolution approach is defined by the following steps:

- **Resampling.** This entails the generation of two Gaussian pyramids, one for the extracted slice and one for the template image. The base level of a pyramid is the full-resolution of the input image, and each level that follows contains the image of the previous level, but at half the resolution.

- **Registration.** As previously stated, the registration starts with the coarsest level of both image pyramids, containing the images at the lowest resolution. The registration steps were explained in previous sections. An important note, the control point spacing is kept constant across all resolution levels.

- **Subdivision.** The control point configuration obtained as a registration result on one resolution level is used to generate a finer grid of control points to be used with the next higher resolution images. The registration and subdivision steps are repeated until the full-resolution images have been processed.

5.2.5.1 Gaussian Pyramid

A Gaussian pyramid consists of low-pass filtered, downsampled images of the preceding level of the pyramid, where the base level is defined as the original image [10]. More formally, let the 2-D original image be denoted by $I(x, y)$. The Gaussian pyramid is defined recursively as follows,

$$G_0(x, y) = I(x, y), \text{ for level l} = 0, \tag{5.22}$$

$$G_l(x, y) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} w(m, n) G_{l-1}(2x + m, 2y + n), \tag{5.23}$$

where $w(m, n)$ is a suitable Gaussian smoothing kernel, and the parameter l denotes the level. In other words, each level is smoothed by means of a convolution filter (see Sections 6.3.1 and 6.3.2) before its resolution is reduced.

5.2.5.2 Grid Subdivision

Once the registration process on a certain level of the Gaussian pyramid is finished, the control point configuration reflects the deformation of this level. Before starting the registration process of the next level begins, the control point grid has to be subdivided so that there are twice as many control points, since that the image resolution of the next level is also doubled. Initializing the control points on the new level to zero, as is done on the coarsest level, is not possible because then the registration result from the previous level would be lost.

The new grid has to be constructed from the old one, by keeping every other point and inserting a new control point between every pair on the coarse grid. A straightforward approach would be to insert new control points by averaging their neighbors on the coarse grid. However, another approach is handled in this thesis. Catmull and Clark [5] suggest subdivision masks that can be seen in Figure 5.2. Using this method, exactly the same displacement field can be obtained from the subdivided control point grid as on the coarse grid, just at twice the resolution. In one dimension, there are two possible configurations for new control points. A point can either be placed in the subdivided grid at a position that corresponds to a control point on the coarse grid, or between two control points. In Schwarz's thesis [44], he explained it as follows.

Let φ^t denote the control points of pyramid level t and let $t-1$ be the next, finer level. Let $\mathbf{p} = (1/8, 6/8, 1/8)$ and $\mathbf{q} = (0, 1/2, 1/2)$ denote vectors containing the weights. Then the weights in $\mathbf{q}$ are used for the directions in which a new control point is between two control points on the coarser grid. Control points in the other directions are weighted by $\mathbf{p}$. For instance, in the computation of control point $\varphi^{t-1}(2i, 2j + 1)$ the neighboring old control points in the y direction would be weighted by $\mathbf{q}$ and the weights in $\mathbf{p}$ would be

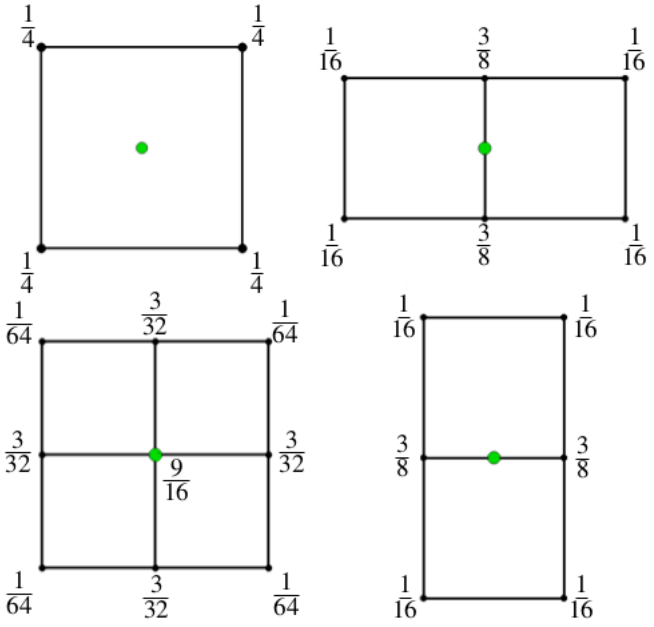


Figure 5.2: Catmull and Clark subdivision masks. The top left configuration is used to calculate a new control point with coordinates $(2i+1, 2j+1)$. With the top right configuration, a control point is calculated with the coordinates $(2i, 2j+1)$. Bottom left configuration is used to calculate a control point with the coordinates $(2i, 2j)$. The bottom right configuration is used to calculate a control point with coordinates $(2i+1, 2j)$.

applied to the neighbors in the x direction. Calculating the control point values can be done as follows [14]:

$$\begin{aligned}
\varphi^{t-1}(2i, 2j) &= \sum_{l=0}^{2}\sum_{m=0}^{2} \mathbf{p}_l \; \mathbf{p}_m \; \varphi^t(i+l-1, j+m-1) \\
\varphi^{t-1}(2i+1, 2j) &= \sum_{l=0}^{2}\sum_{m=0}^{2} \mathbf{q}_l \; \mathbf{p}_m \; \varphi^t(i+l-1, j+m-1) \\
\varphi^{t-1}(2i, 2j+1) &= \sum_{l=0}^{2}\sum_{m=0}^{2} \mathbf{p}_l \; \mathbf{q}_m \; \varphi^t(i+l-1, j+m-1) \\
\varphi^{t-1}(2i+1, 2j+1) &= \sum_{l=0}^{2}\sum_{m=0}^{2} \mathbf{q}_l \; \mathbf{q}_m \; \varphi^t(i+l-1, j+m-1)
\end{aligned} \tag{5.24}$$

Chapter 6

Implementation Details

In this chapter a detailed description will be provided of the implemented framework.

6.1 Image Registration Using CUDA

The GPU is an attractive platform to accelerate compute-intensive algorithms (such as image registration) due to its ability to perform many arithmetic operations in parallel, as made apparent in Section 3.1. Section 5.2 gave a detailed description of what image registration entails. The actual implementation details will be explained in this section. For this thesis, the implementation was modeled after the work of Shackleford [46,47,49]. Shackleford's efforts resulted in the creation of the Plastimatch[1] framework.

Before going into further detail, an overview will be given of what has been implemented to achieve the image registration functionality. This is done by Algorithm 1. Note that the cost function exists out of two terms: the similarity measure and the regulizer. The total cost function is calculated on GPU, but only the calculations concerning the similarity measure are explained in this thesis, since this is the part where the computation speed can be significantly increased. The explanation of how to calculate the regulizer term and its derivative, are explained in Sections 5.2.3 and 6.2 respectively.

With image registration, a displacement field is calculated that is used to warp the template in such a manner that it resembles the fixed image (in our case, the extracted slice). As was mentioned in Section 5.2, the displacement field is calculated using the grid of control points and the overal registration process can be viewed as an optimization problem as the similarity measure performed on the two images, has to be minimized. One must also take into consideration the regularization term described in Section 5.2.3. This requires that we evaluate:

1. C, the cost function corresponding to a given set of B-spline coefficients;

[1] `http://plastimatch.org/`

Algorithm 1 The image registration algorithm used in this thesis

calculate the Gaussian pyramids of both the fixed and template images
initialize the grid of control points Φ. This happens at the coarsest levels of the Gaussian pyramids.
repeat
  calculate the cost function gradient with respect to the B-spline coefficients in Φ:

$$\nabla C = \frac{\partial C}{\partial \Phi^l}$$

  while $\|\nabla C\| > \epsilon$ **do**
    recalculate the control points $\Phi = \Phi + \mu \frac{\nabla C}{\|\nabla C\|}$
    recalculate the cost function gradient ∇C
  end while
  increase the control point grid resolution by calculating new control points Φ^{l+1} from Φ^l.
  increase image resolution of both images by going to a higher level in their respective Gaussian pyramids.
until finest level of resolution is reached.

2. $\partial C/\partial \Phi$, the change in the cost function with respect to the B-spline coefficient values Φ at each individual control point.

For simplicity, $\partial C/\partial \Phi$ will be called *cost function gradient* throughout the thesis. The registration process then becomes one of iteratively defining B-spline coefficients Φ by generating a displacement field, by means of B-spline interpolation, and use it to warp the template image, evaluating the cost function C by means of a similarity measure and a regulizer, calculating the cost function gradient $\partial C/\partial \Phi$ for each control point, and performing the optimization (in our case, this is gradient descent, as shown in Section 5.2.4) to generate the next set of B-spline coefficients. As mentioned by Shackleford, the B-spline interpolation and the gradient calculation are the two most time-consuming stages within the overall registration process. Therefore, these two stages have been accelerated with CUDA.

6.1.1 Displacement Field by B-spline Interpolation

A grid of uniformly-spaced control points is superimposed on the grid of pixels of the template image, as shown in Figure 6.1. This results in that the image is partitioned into many equally sized 5×5 tiles. Every vector of the displacement field is influenced by the 16 control points in the tile's immediate neighborhood and the B-spline basis

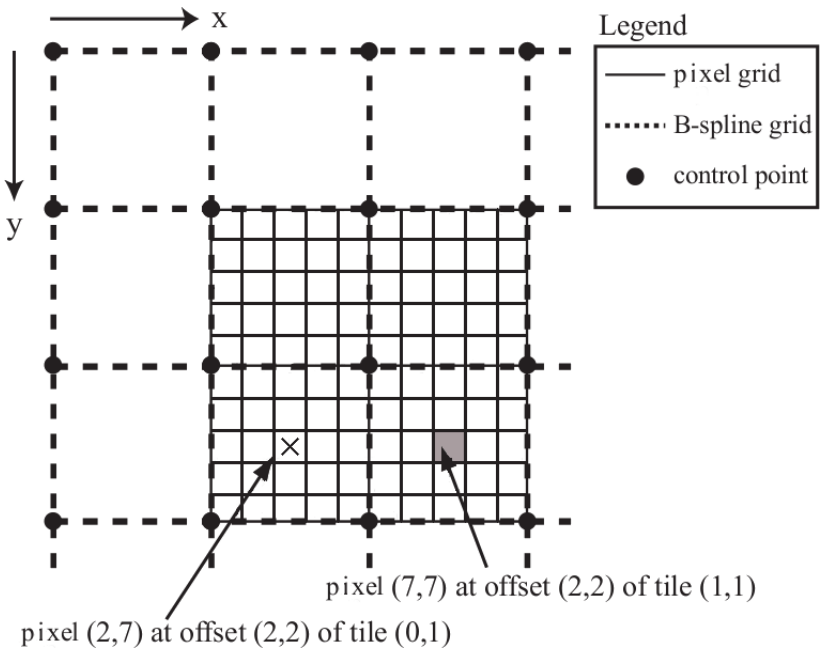


Figure 6.1: A grid of control points superimposed on the pixels of the template image. Both marked pixels are located at the same relative offset within their respective tiles, so both will use the same $\beta_l(u)\beta_m(v)$ value.

function product evaluated at the pixel. The latter is only dependant on the pixel's local coordinates within the tile. For example, in Figure 6.1 one can see that both marked pixels have the same local coordinates within their respective tiles, namely $(2, 2)$, which will result in the same B-spline basis function product value at these two pixels. This property allows the pre-computation of all the relevant B-spline basis function values once, instead of recalculating these values for each individual tile.

The displacement field calculation was explained in Section 5.2.1.2, but for reading convenience, some aspects will be repeated in the following paragraph. The B-spline interpolation used for the calculation of the x-component of the displacement vector for a certain pixel with coordinates (x, y) is

$$\mathbf{T}(x, y, \phi) = v_x(x, y) = \sum_{m=0}^{3}\sum_{l=0}^{3} \beta_l(u)\beta_m(v)\phi_{x,i+l,j+m}, \tag{6.1}$$

where ϕ_x is the spline coefficient defining the x-component of the displacement vector for one of the 16 control points that influence the pixel. The dimensions n_x and n_y of the control point grid take the following form:

$$n_x = \left\lceil \frac{m_x}{s_x} + 3 \right\rceil, \qquad n_y = \left\lceil \frac{m_y}{s_y} + 3 \right\rceil, \tag{6.2}$$

where m_x and m_y are respectively the X and Y dimensions of the template image, and s_x and s_y are the control point spacing distances. The parameters i and j are the indices of the tile, consisting out of control points, within which the pixel (x, y) falls, that is

$$i = \left\lfloor \frac{x}{n_x} \right\rfloor - 1, \qquad j = \left\lfloor \frac{y}{n_y} \right\rfloor - 1. \tag{6.3}$$

The local coordinates of the pixel within this tile, normalized between $[0, 1]$, are

$$u = \frac{x}{n_x} - \left\lfloor \frac{x}{n_x} \right\rfloor, \qquad v = \frac{y}{n_y} - \left\lfloor \frac{y}{n_y} \right\rfloor. \tag{6.4}$$

Finally, the cubic B-spline basis function β_l^3 along the x directions is given by

$$\beta_l^3(u) = \begin{cases} \beta_0^3(u) & = (1-u)^3/6, \\ \beta_1^3(u) & = (3u^3 - 6u^2 + 4)/6, \\ \beta_2^3(u) & = (-3u^3 + 3u^2 + 3u + 1)/6, \\ \beta_3^3(u) & = u^3/6. \end{cases} \tag{6.5}$$

and similary for β_m^3 along the y direction. As previously mentioned, a straightforward implementation would result in calculations that could easily be reduced. Implementing a data structure that exploits the symmetrical features that emerge as a result of the grid alignment, makes the implementation of Equation 6.1 much faster. Shackleford considers the following optimizations:

- All pixels within a single tile use the same set of 16 control points to compute their respective displacement vectors. This means that for each tile in the image, the corresponding set of control point indices can be pre-computed and stored in a lookup table (LUT), called the *index LUT*.
- Equation 6.4 shows that for a tile of dimensions $n_w = n_x \times n_y$, the number of $\beta(u)\beta(v)$ combinations is limited to n_w values. Furthermore, as shown in Figure 6.1, two pixels belonging to different tiles but with the same local coordinates, will be subject to identical $\beta(u)\beta(v)$ products. This means that a lookup table, called the *multiplier LUT*, can be calculated containing the pre-computed $\beta(u)\beta(v)$ product for all the normalized coordinate combinations.

For each pixel, the absolute coordinates (x, y) within the image dimensions are used to calculate the tile indices that the pixels falls within as well as the pixel's local coordinates within the tile, using Equations 6.3 and 6.4 respectively. These tile indices will be used to access the index LUT, which will provide coordinates of the 16 neighboring control points that influence the pixel's interpolation calculation. The pixel's local coordinates within the tile will be used to retrieve the appropriate, pre-calculated $\beta(u)\beta(v)$ product from

the multiplier LUT. Using these lookup tables, the displacement field calculation can be considerably optimized.

Once the displacement field is calculated, it is used to warp the template image. Bilinear interpolation, which was explained in Section 3.4, is used to determine intensities of pixels at non-integer coordinates. Once warped, the template image is compared to the fixed image (in our case, the extracted slice) by means of the similarity measure. This similarity measure is one of the cost function terms that has to be optimized. In our case, the similarity measure is the SSD, as explained in Section 3.4, that is

$$C_{sim} = \frac{1}{N} \sum_{y=0}^{Y} \sum_{x=0}^{X} (I_f(x,y) - I_m(x+v_x, y+v_y))^2, \tag{6.6}$$

where C_{sim} is the similarity measure part of the cost function, N is the amount of pixels in the template image, X and Y are the template image's dimensions, and (v_x, v_y) form the displacement field vector for the template image pixel with coordinates (x, y). The similarity part of the cost function will be called the *similarity cost function* throughout the rest of this thesis.

6.1.2 Similarity Cost Function Gradient Calculation

As was explained in Section 5.2.4, the gradient descent optimization requires the partial derivatives of the similarity cost function with respect to each control point (B-spline) coefficient value. The lookup tables introduced in the previous section not only accelerate the B-spline interpolation stage, it also accelerates the similarity cost function gradient calculation. The similarity cost function gradient can be considered as the change in the similarity cost function with respect to the coefficient values Φ at each individual control point. This similarity cost function gradient can be decomposed and can be computed independently, for a given control point at the grid coordinates (κ, λ), as

$$\frac{\partial C_{sim}}{\partial \Phi_{(\kappa,\lambda)}} = \frac{1}{N} \sum_{(x,y)}^{16 \text{ tiles}} \frac{\partial C}{\partial \overrightarrow{v}(x,y)} \frac{\partial \overrightarrow{v}(x,y)}{\partial \Phi}, \tag{6.7}$$

where the summation is performed over all the pixels (x, y) of the template image contained in the 16 tiles found in the control point's local support region. By means of this decomposition, the similarity cost function gradient's dependencies on the similarity cost function and spline coefficients can be independently evaluated. The first term, $\partial C_{sim}/\partial \overrightarrow{v}(x,y)$, depends only on the similarity cost function. The second term, $\partial \overrightarrow{v}(x,y)/\partial \Phi$, describes how the displacement field changes with respect to the control points. This last term is only dependent on the B-spline parameterization and the pixel's

location; it is computed as

$$\frac{\partial \overrightarrow{v}(x,y)}{\partial \Phi} = \sum_{l=0}^{3}\sum_{m=0}^{3} \beta_l^3(u)\beta_m^3(v). \tag{6.8}$$

This only needs to be computed once, since it remains constant over all the optimization iterations. The pre-calculated $\beta(u)\beta(v)$ product is available via the multiplier LUT.

Since the SSD similarity measure is utilised, the first term (see Equation 6.7) can be written in terms of the template image's spatial gradient $\nabla I_m(x,y)$ (see Section 6.3.3) as

$$\frac{\partial C_{sim}}{\partial \overrightarrow{v}(x,y)} = 2 \times (I_f(x,y) - I_m(x+v_x, y+v_y))\nabla I_m(x,y). \tag{6.9}$$

Equation 6.9 shows that it depends on the intensity values of the fixed image (the extracted slice) and the template image, I_f and I_m respectively, as well as the current value of the displacement field $\overrightarrow{v}$. Meaning, that during each iteration, the displacement field will change and that results in the modification in the correspondence between the fixed and template images. This also means that, unlike $\partial \overrightarrow{v}/\partial\Phi$, $\partial C_{sim}/\partial \overrightarrow{v}$ needs to be recalculated during each iteration of the optimization. With both terms calculated, they can be combined using the chain rule from Equation 6.7, wich can be written in terms of the control point coordinates (κ, λ) as

$$\frac{\partial C_{sim}}{\partial \Phi_{(\kappa,\lambda)}} = \frac{1}{N}\sum_{(\kappa,\lambda)}^{16\text{ tiles}}\sum_{b=0}^{s_y}\sum_{a=0}^{s_x} \frac{\partial C_{sim}}{\partial \overrightarrow{v}_{(x,y)}} \times \sum_{m=0}^{3}\sum_{l=0}^{3} \beta_l^3\left(\frac{a}{s_x}\right)\beta_m^3\left(\frac{b}{s_y}\right), \tag{6.10}$$

where a and b are the unnormalized local coordinates of a pixel inside its respective tile, and x and y represent the absolute coordinates of a pixel within the template image. Here, x and y can be defined in terms of the control point coordinates and summation indices as follows:

$$x = s_x(\kappa - l) + a, \qquad y = s_y(\lambda - m) + b. \tag{6.11}$$

6.1.3 Framework Organization

The implemented image registration framework uses the GPU for calculations as shown in Figure 6.2. The B-spline interpolation and the cost function gradient, as explained in Sections 5.2.1.2 and 6.1.2 respectively, are implemented on the GPU by means of CUDA, while the optimization stage is performed on CPU. During each iteration, the optimizer, working on the CPU, calculates new parameters to update the control points so that the cost function is minimized. When a minima has been reached, the registration process stops.

When analyzing Figure 6.2, one can see that both the evaluated cost function and its gradient must be transfered from GPU to the CPU for every iteration of the image

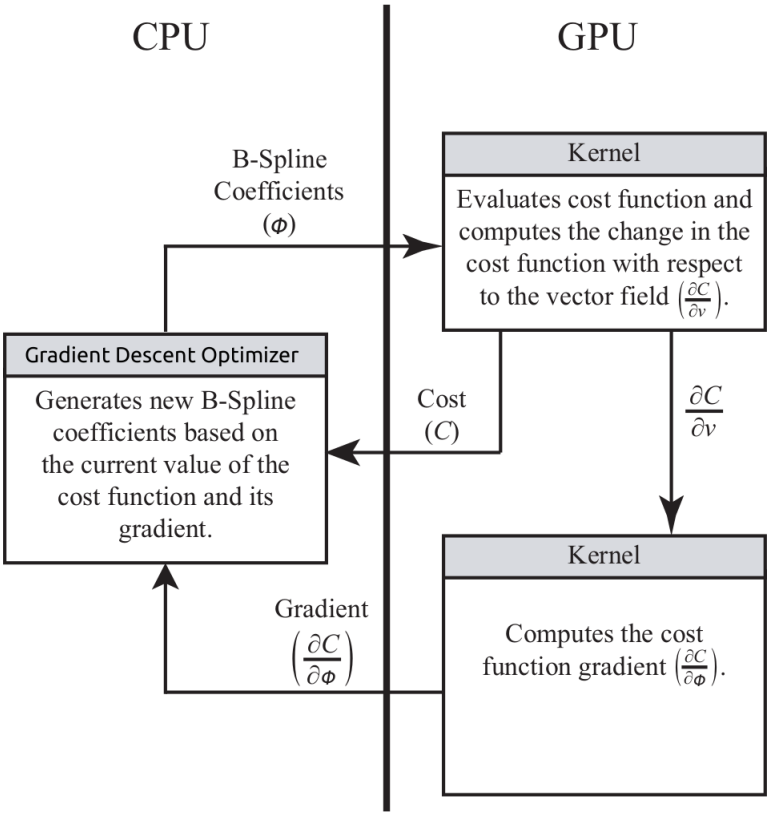


Figure 6.2: Framework organization

registration process. Transfers between the CPU and GPU memories are the most costly in terms of time, but Shackleford observed in his experiments that the CPU-GPU communication overhead demands roughly 0.14% if the total algorithm execution time. This fact allows the conclusion that the CPU-GPU transfers do not affect the overal algorithm performance.

6.1.4 GPU Implementation

As previously mentioned, the similarity cost function evaluation, the $\partial C_{sim}/\partial \overrightarrow{v}$ term, the $\partial \overrightarrow{v}/\partial \Phi$ term and the similarity cost function gradient $\partial C_{sim}/\partial \Phi$ are calculated on the GPU. This section will denote in pseudo-code what the kernels look like. Algorithm 2 shows how the similarity cost function is evaluated and how the $\partial C_{sim}/\partial \overrightarrow{v}$ term is calculated. Algorithm 3 shows how a kernel calculates the $\partial C_{sim}/\partial \Phi$ term. This is however a naïve implementation how the algorithm should be implemented on the GPU. A more optimized implementation will be discussed in Section 6.1.4.2, while the naïve implementation will be described by Section 6.1.4.1.

6.1.4.1 Naïve Implementation

The described kernels in Algorithms 2 and 3 show that the calculations for image registration can easily be calculated in parallel. The kernel described in Algorithm 2, for example, shows that one thread is launched per pixel in the fixed image. The coordinates for each pixel are deferred from the CUDA thread indices. Here, the template image is already warped with the displacement field calculated by using Equation 6.1. One can see that the difference between the fixed image and warped template image their intensities can be used for calculating the similarity cost function and the similarity cost function gradient, which is very convenient since no two seperate functions must be implemented. The similarity cost function and $\partial C_{sim}/\partial \overrightarrow{v}$ values are stored in the GPU global memory when the kernel finishes its calculations.

The kernel described by Algorithm 3 calculates the cost gradient function vector $\partial C_{sim}/\partial \Phi$ for a control point. As previously described, this gradient calculation makes use of the $\partial C_{sim}/\partial \overrightarrow{v}$ and $\partial \overrightarrow{v}/\partial \Phi$ terms, as described in Equation 6.7. The kernel is launched with as many threads as there are control points, where each thread calculates the $\partial C_{sim}/\partial \Phi$ value for each control point. As shown in the pseudo-code, the coordinates (κ, λ) are deferred from the CUDA thread indices. Each control point's gradient is influenced by 16 neighboring tiles, and the template pixel values that these tiles contain. Once the calculations are finished, the results are stored in the global memory of the GPU.

Algorithm 2 Kernel that calculates the similarity cost function value and the $\partial C_{sim}/\partial v$ value for a pixel

Require: I_f is the fixed image, I'_m is the **warped** template image, ∇I_f is the spatial gradient of I_f, ∇I_m is the spatial gradient of I_m, x and y are obtained by the thread indices of CUDA

/* Calculate the SSD difference for the similarity cost function score */
$D = I_f(x, y) - I'_m(x, y)$
$C(x, y) = D^2$

/* Compute $\partial C_{sim}/\partial \overrightarrow{v}$ and store it in global memory of GPU */
$(\partial C_{sim}/\partial v(x, y)).x = 2 \times D \times \nabla I'_{m,x}(x, y)$
$(\partial C_{sim}/\partial v(x, y)).y = 2 \times D \times \nabla I'_{m,y}(x, y)$

6.1.4.2 Optimized Implementation

Although the implementation described in Section 6.1.4.1 is a good way to exploit the parallelization capabilities of a GPU, it suffers from serious performance deficiency as the kernel described in Algorithm 3 does a lot of redundant load operations [46, 47, 49]. As each set of pixels within each tile is influenced by the neighboring 16 control points, so does each of these pixels in each tile influence the gradient calculation of each of these neighboring control points. As can be seen in Algorithm 3, the gradient value for each control point is influenced by its neighbors. This also implies that when two different threads calculate their respective contribution to a certain one control point, and they both need to calculate the contribution of one the same control point, they must each load the same $\partial C_{sim}/\partial \overrightarrow{v}$ values from the same tile. The only thing these two threads do different is that they must each use different basis-function products when computing the $\partial \overrightarrow{v}/\partial \Phi$ term to obtain their respective contributions to the $\partial C_{sim}/\partial \Phi$ term. To gain a visible view of the problem, consider the following equations. Thread 1 calculates the following contribution of a tile:

$$\sum_{s_x, s_y} \frac{\partial C_{sim}}{\partial \overrightarrow{v}(x, y)} \beta_0(u)\beta_0(v). \tag{6.12}$$

Thread 2 calculates his contribution of the same tile, but with different l and m values:

$$\sum_{s_x, s_y} \frac{\partial C_{sim}}{\partial \overrightarrow{v}(x, y)} \beta_1(u)\beta_2(v). \tag{6.13}$$

In both Equations 6.12 and 6.13, the u and v values are the normalized position of a pixel within the tile. One can clearly see that although these two threads are executed inde-

Algorithm 3 Kernel that calculates the $\partial C_{sim}/\partial \Phi$ value for a control point

Require: Use the $\partial C_{sim}/\partial \overrightarrow{v}$ calculated in Algorithm 2, and obtain the (κ, λ) coordinates of the control point

/* Iterate through the 16 tiles affecting this control point to calculate $\partial C_{sim}/\partial \Phi$. */
$A_x = A_y = 0$;
for m = 0 to 3 **do**
  for l = 0 to 3 **do**
    $t_x = \kappa - l$; // X component of tile index
    $t_y = \lambda - m$; // Y component of tile index
    for j = 0 to s_y **do**
      for i = 0 to s_x **do**
        /* Absolute x and y pixel coordinates of template image */
        $x = (s_x \times t_x) + i$;
        $y = (s_y \times t_y) + j$;
        if x and y fall within the bounds of the template image **then**
          $U = \beta_l(u)\beta_m(v)$;
          $A_x = A_x + U \times \partial C_{sim}/\partial \overrightarrow{v}_{t_x}(i)$;
          $A_y = A_y + U \times \partial C_{sim}/\partial \overrightarrow{v}_{t_y}(j)$;
        end if
      end for
    end for
  end for
end for

/* Store the $\partial C_{sim}/\partial \Phi$ solution for the control point (κ, λ) in GPU global memory*/
$(\partial C_{sim}/\partial \Phi(\kappa, \lambda)).x = A_x$;
$(\partial C_{sim}/\partial \Phi(\kappa, \lambda)).y = A_y$;

pendently of each other in parallel, each thread will end up loading the same $\partial C_{sim}/\partial \overrightarrow{v}$ values, as they are handling the same tile.

This redundant loading of $\partial C_{sim}/\partial \overrightarrow{v}$ values can be mitigated by implementing the following two stages. The first stage will read all the $\partial C_{sim}/\partial \overrightarrow{v}$ values of a certain tile from global memory. Any given pixel tile is influenced by (and influences) 16 neighboring pixel tiles, meaning that there are 16 different possible (l,m) combinations. For a certain tile with a certain (l,m) combination, the following must be calculated

$$\overrightarrow{Z}_{(\kappa,\lambda,l,m)} = \sum_{b=0}^{s_y} \sum_{a=0}^{s_x} \frac{\partial C_{sim}}{\partial \overrightarrow{v}(x,y,z)} \beta_l(u)\beta_m(v), \tag{6.14}$$

where the values for x and y can be calculated using Equation 6.11. The operation described in Equation 6.14 is performed for the 16 possible (l,m) combinations, resulting in 16 $\overrightarrow{Z}$ values per tile. This operation will be implemented as a GPU kernel. Each of these $\overrightarrow{Z}$ values is a partial solution to the gradient computation for a certain control point within the grid. Therefore, we allocate for each control point within the grid an array (or bins) that can hold up to 16 of these partial gradient computation $\overrightarrow{Z}$ values. Once the 16 $\overrightarrow{Z}$ values for a certain tile are computed, each of these values must be inserted into the correct control point's bin of partial gradient computation $\overrightarrow{Z}$ values. In other words, when a tile computes the 16 $\overrightarrow{Z}$ values, these values will not only be written to different control points, but to different bin slots within the control point. The second stage of the gradient computation will simply sum up these 16 $\overrightarrow{Z}$ for each control point.

The two proposed stages are implemented as GPU kernels and are described by Algorithms 4 and 5. The kernel described in Algorithm 4 will be launched with 16 threads operating on a single tile. Each one of these threads will read a portion of the $\partial C_{sim}/\partial \overrightarrow{v}$ values if this tile into shared memory, so that each of these 16 threads doesn't constantly have to read out of global memory. Once they have been read into shared memory, each thread will compute the $\partial C_{sim}/\partial \Phi$ value with the appropriate (l,m) values. When a thread finishes this calculation, it puts the computed value into the correct control points's correct bin.

Stage two is described by Algorithm 5. Each thread that executes this kernel represents a control point. It simply sums up all the 16 values of the control point in question.

6.2 Numerical Differentiation

The cost function exists out of two terms:

1. the **similarity measure**, which was described in detail in Sections 3.4 and 5.2.3;
2. the **regularizer** term, which was described in Section 5.2.3.

Algorithm 4 Optimized kernel design for calculating the $\partial C_{sim}/\partial \Phi$ value for a control point. Stage 1.

Require: Use the $\partial C_{sim}/\partial \overrightarrow{v}$ calculated in Algorithm 2, get the thread block IDs (cpX,cpY), get the thread IDs within the thread block (t_x,t_y)

/* The block IDs (cp_x,cp_y) represent a control point with coordinates (κ,λ). Each thread (t_x,t_y) calculates a contribution to a different control point using the $\partial C_{sim}/\partial \overrightarrow{v}$ values if this tile. Each thread loads a part of the $\partial C_{sim}/\partial \overrightarrow{v}$ values if this tile into shared memory *smem*. This eliminates the redundant loads from global memory. */

/* After reading into shared memory is done (left out for brevity) */
__synchthreads();

/* A_x and A_y will contain the contribution */
$A_x = A_y = 0$;
for j = 0 to s_y **do**
  for i = 0 to s_x **do**
    /* U = $\beta_l(u)\beta_m(v)$ */
    $A_x = A_x + U \times \partial C_{sim}/\partial \overrightarrow{v}_x(i)$;
    $A_y = A_y + U \times \partial C_{sim}/\partial \overrightarrow{v}_y(j)$;
  end for
end for

/* *dc_dp_buckets* is a $[s_x s_y \times 16 \times 2]$ matrix that will contain the seperate partial gradient contributions (for both x- and y-dimensions) of all the control point's 16 neighbors. N_x is the width of the control point grid. */
if $cp_x + t_x$ and $cp_y + t_y$ fall within the bounds of the control point grid **then**
  $dc\_dp\_buckets[cp_x + t_x + (cp_y + t_y)N_x][(3 - t_x) + 4(3 - t_y)][0] = A_x$;
  $dc\_dp\_buckets[cp_x + t_x + (cp_y + t_y)N_x][(3 - t_x) + 4(3 - t_y)][1] = A_y$;
end if

Algorithm 5 Optimized kernel design for calculating the $\partial C_{sim}/\partial\Phi$ value for a control point. Stage 2.

Require: *dc_dp_buckets* calculated in Algorithm 5, obtain the (κ, λ) coordinates of the control point

/* A_x and A_y will contain the total gradient value of the control point in question */
$A_x = A_y = 0$;

/* cp_t will be the control point index, N_x will be the width of the control point grid, and (cp_x,cp_y) represent a control point with coordinates (κ,λ) */
$cp_t = 16(cp_x + cp_y N_x)$

for i = 0 to 16 **do**
  $A_x = A_x + dc\_dp\_buckets[cp_t][i][0]$;
  $A_y = A_y + dc\_dp\_buckets[cp_t][i][1]$;
end for

/* Store the $\partial C_{sim}/\partial\Phi$ solution for the control point (κ, λ) in GPU global memory*/
$(\partial C_{sim}/\partial\Phi(\kappa, \lambda)).x = A_x$;
$(\partial C_{sim}/\partial\Phi(\kappa, \lambda)).y = A_y$;

Calculating the cost function gradient entails the calculation of the gradient of the similarity measure in function of the control points, but also the gradient of the regularizer term. The latter will be explained in this section.

Equation 5.15 shows that the regularizer term is calculated by means of the ∇ operator, which is a discrete approximation of the gradient operator based on central differences [37]. Using this central differences method, the first partial derivative in the x direction $\nabla\phi_x(i,j)$ can be denoted as follows:

$$\nabla\phi_x(i,j) = (D_x\phi_x(i,j), D_y\phi_x(i,j)) \tag{6.15}$$

$$\begin{array}{ll} D_x\phi_x(i,j) & = \frac{1}{2s_x}(\phi_x(i+1,j) - \phi_x(i-1,j)), \\ D_y\phi_x(i,j) & = \frac{1}{2s_y}(\phi_x(i,j+1) - \phi_x(i,j-1)), \end{array} \tag{6.16}$$

where s_x and s_y represent the control point spacing in x and y direction. The first partial derivative in the y direction $\nabla\phi_y(i,j)$ can be denoted as follows:

$$\nabla\phi_y(i,j) = (D_x\phi_y(i,j), D_y\phi_y(i,j)) \tag{6.17}$$

$$\begin{array}{ll} D_x\phi_y(i,j) & = \frac{1}{2s_x}(\phi_y(i+1,j) - \phi_y(i-1,j)), \\ D_y\phi_y(i,j) & = \frac{1}{2s_y}(\phi_y(i,j+1) - \phi_y(i,j-1)). \end{array} \tag{6.18}$$

As explained in Sections 5.2.3 and 5.2.4, in order to optimize the cost function, its gradient must be calculated. This not only entails the calculation of the derivative of the similarity measure, but also the calculation of the derivative of the regularizator. To find the regularizator gradient, the second partial derivatives can be constructed as central difference approximations of the first partial derivatives. The second partial derivative of the regularizator in the x direction can be stated as follows:

$$\frac{\partial R(\phi_x)}{\partial\phi_x(i,j)} = -\frac{2}{N}\Delta\phi_x(i,j) = -\frac{2}{N}(D_{xx}(i,j) + D_{yy}(i,j)), \tag{6.19}$$

where Δ is represented as the discrete version of the Laplace operator. D_{xx} denotes a central difference approximation of the second unmixed partial derivative in the x direction and can be written as follows:

$$\begin{array}{ll} D_{xx}\phi_x(i,j) & = \frac{1}{2s_x}(D_x\phi_x(i+1,j) - D_x\phi_x(i-1,j)), \\ & = \frac{1}{2s_x}\left(\frac{1}{2s_x}(\phi_x(i+2,j) - \phi_x(i,j)) - \frac{1}{2s_x}(\phi_x(i,j) - \phi_x(i-2,j))\right), \\ & = \frac{1}{4s_x^2}(\phi_x(i-2,j) - 2\phi_x(i,j) + \phi_x(i+2,j)), \end{array} \tag{6.20}$$

where $D_x\phi_x$ was explained by Equation 6.16. Equation 6.20 is the second partial derivative in the x direction, it can easily be represented for the y direction. This version, however, will not be described as the reader can derive it for himself.

6.3 Spatial Filtering

Spatial filtering is typically used for image processing, like smoothing an image or calculating its spatial derivative, in a broad spectrum of applications. The term *filter* comes from the frequency domain processing field, where it refers to accepting (or rejecting) certain frequency components. Spatial filtering and filtering in the frequency domain have a one-on-one correspondence, although one can only perform non-linear filtering in the spatial domain and not in the frequency domain. It is not the purpose of this thesis go into further detail of spatial filtering theory, therefore, we will only discuss the spatial filters used in the implemented framework.

6.3.1 Discrete Convolution

There are two ways to perform spatial filtering, namely by means of *correlation* or *convolution*. Both methods move a filter mask over the image, multiply each pixel in the range of the filter mask with the corresponding weighting factor of the mask, add up these products and store the result at the center pixel [15, 22]. The mechanics of convolution and correlation are the same, except that for convolution, the filter mask is rotated 180°. Convolution can be written as follows:

$$(w * I)(x, y) = \sum_{s=-c}^{c} \sum_{t=-c}^{c} w(s, t) I(x - s, y - t), \tag{6.21}$$

where w is the convolution kernel, r is the side length of w and $c = \lfloor r/2 \rfloor$, I is the image that is to be processed, and x and y are the image's pixel coordinates. The formula shows that the result of the summation is written to the central pixel of the current position of the convolution operation. When processing the border of the image, the filter mask ranges over the edge of the image. To mitigate this problem, a border is added to the image with half the width of the filter mask (this is the c parameter in Equation 6.21). Next, this border area is filled with zeros or this area mirrors the boundary of the image.

6.3.2 Gaussian Smoothing Filter

Gaussian filters are useful for smoothing images or detecting edges after smoothing. The multiresolution approach of the framework requires that the fixed and template images must be smoothed with a Gaussian filter, by means of convolution, upon reducing their resolution. This convolution filter is obtained by sampling the Gaussian function

$$g(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}. \tag{6.22}$$

The weights of the convolution kernel decrease smoothly to zero when the distance from the origin increases, meaning that the image values near the central location are more

important than the values that are more remote. The σ parameter determines how broad, or focused, the neighborhood will be. It is said that 95% of the total weight will be contained within 2σ of the center [54]. In this thesis, a 5×5 convolution kernel is utilized and the σ parameter is set to 1.

6.3.3 Sobel Filter

Section 6.1.2 describes how the cost function gradient is computed. One of these calculation steps is to calculate the $\partial C_{sim}/\partial \overrightarrow{v}$, using the template image's spatial gradient $\nabla I_m(x, y)$. A Sobel convolution filter can be used to calculate these gradients. The gradient of a certain function $f(x, y)$ is a two-dimensional vector $[\partial f/\partial x, \partial f/\partial y]^T$, which means that when one calculates the gradient of an image, one must calculate the gradient for the x direction as well as for the y direction. For a certain image I, the two components of the gradient take the following form:

$$\frac{\partial I}{\partial x} = \frac{1}{8}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} * I, \qquad \frac{\partial I}{\partial y} = \frac{1}{8}\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I, \tag{6.23}$$

where $*$ denotes the discrete convolution operator. These Sobel kernels use a weight value of 2 in the center coefficient to achieve some smoothing by giving more importance to the center point.

Part IV

Evaluation and Conclusion

Chapter 7

Evaluation

Given that the method for extracting the corpus callosum area relies heavily on deformable registration, the quality of the registration directly affects the accuracy of the corpus callosum registration and its area measurement. Several experiments have been conducted in order to evaluate the deformable registration that was implemented. Synthetic data has been used to demonstrate the effectiveness and performance of the registration framework, but the registration framework has also been evaluated with medical data. Section 7.1 contains the results of the performed experiments. In Section 7.2, the robustness of the method of finding the plane with minimal corpus callosum area will be evaluated.

7.1 Deformable Registration Evaluation

As stated above, the deformable registration has been evaluated by using both synthetic data and medical data. Section 7.1.1 will discuss the evaluation using synthetic data and Section 7.1.2 will contain the results of the evaluation using medical data.

7.1.1 Synthetic Data

To evaluate the effectiveness and performance of the registration framework, ground truth experiments have been conducted. The main idea of ground truth experiments is that some sort of data is used which is known to be correct in some sense. The precision and error introduced by the algorithm that has to be evaluated, can be assessed by comparing the results with the ground truth. In this case, a synthetic image is deformed using a known control point configuration. This deformed image will serve as the reference image, while the original, undeformed image will act as the template image. The quality of the registration can be measured by comparing the displacement field obtained by the registration framework with the ground truth displacement field computed from the known control point configuration.

For the experiments, two synthetic images where used as shown in Figures 7.1a and 7.2a. Figure 7.1a is a simple checkerboard image of size 256×256, while Figure 7.2a is a checkerboard pattern of size 256×256 with with varying intensities. For both images, a seperate control point configuration was created that were sinusiodal in both x- and y-direction and with a fixed control point spacing. Using these control point configurations, the displacement fields were calculated for each original image together with the resulting warped images as shown by Figures 7.1b and 7.2b. Next, the registration framework is utilized with various control point spacings ranging from 5 to 50 pixels with increments of 5.

7.1.1.1 Similarity after Registration

The first metric that has been observed, is the similarity of the images after the registration process. The results can be seen in Figure 7.3. One can see that the optimal solution is achieved when a control point spacing of 15 to 20 pixels is handled. Using a control point spacing of 25 pixels also generates a reasonable result, but anything higher than 25 pixels produces an undesirable outcome. In practice, one typically uses relatively small control point spacings in order to capture subtle image details, as was in our case.

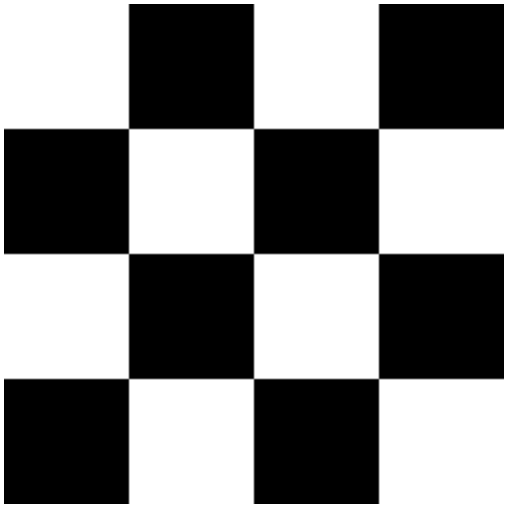
7.1.1.2 Root Mean Square Error Metric

The next experiment calculated the Root Mean Square Error (RMSE) between the ground truth displacement field and the displacement field obtained from the registration process. The results can be found in Figure 7.4. Given that the images that were used were checkerboard patterns with a lot of homogeneous regions, the final transformation can result in displacement vectors that differ from the ground truth [27]. Although this has no influence on the final transformed image, it does lead to an increased RMSE. The results show that the most preferable control point spacing configuration is around 20 pixels for the large checkerboard pattern and 25 pixels for the little checkerboard pattern.

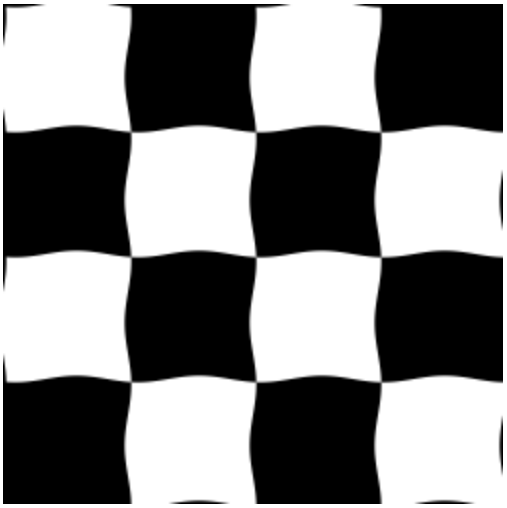
7.1.1.3 Magnitude of Difference Metric

The image registration problems are not the only technology that handle displacement fields. For instance, optical flow uses displacement fields for describing changes between moving images in sequence [36]. One of the measures handled, is the *magnitude of difference*. This is a straightforward approach that averages the magnitude of difference between all vectors of the displacement field of the ground truth and the displacement field generated by a certain algorithm. The simple magnitude of difference can be denotes as follows

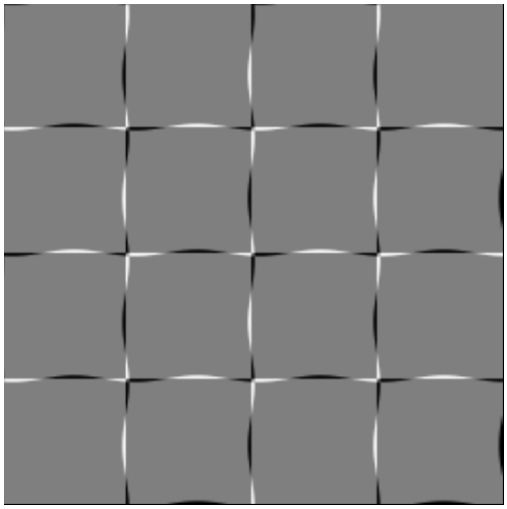
$$e_{\text{mod}}(c, e) = \|c - e\|, \tag{7.1}$$



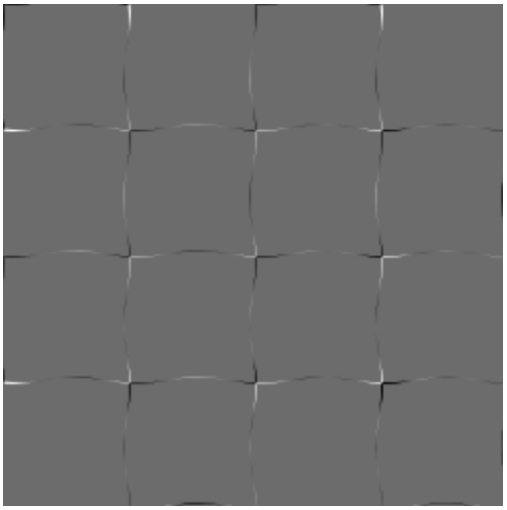
(a) Original image



(b) Warped image



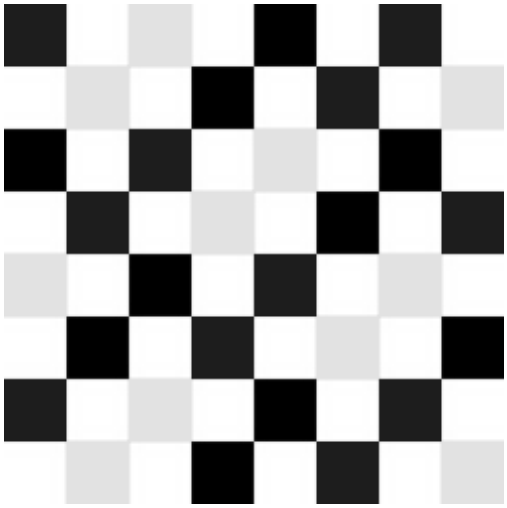
(c) Image difference: Before registration



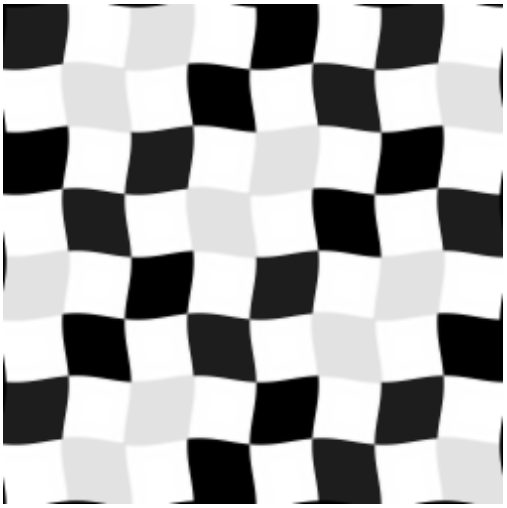
(d) Image difference: After registration

Figure 7.1: Little checkerboard

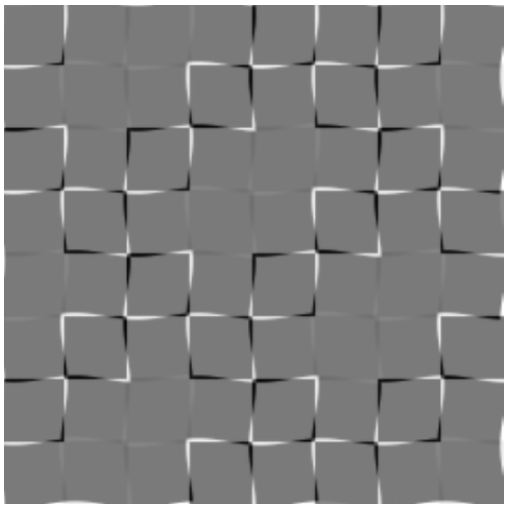
where c is a ground truth vector and e the estimated vector calculated by a certain algorithm. The measure can be used to compare two displacement fields f_c and f_e,



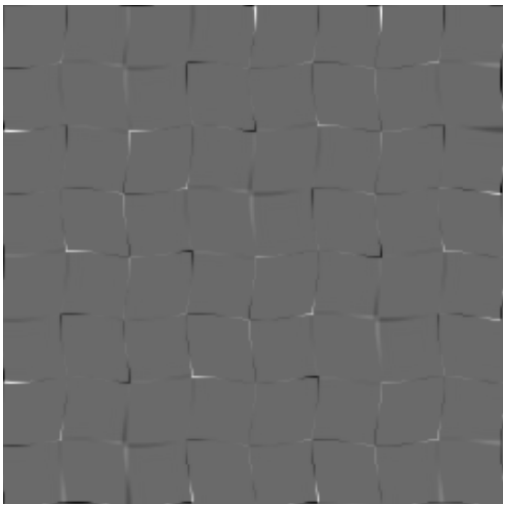
(a) Original image



(b) Warped image



(c) Image difference: Before registration



(d) Image difference: After registration

Figure 7.2: Large checkerboard with different intensities

consisting out of N vectors by the following equation

$$E_{\text{mod}}(f_c, f_e) = \frac{1}{M} \sum_{x \in \Omega} e_{\text{mod}}(f_c(x), f_e(x)). \tag{7.2}$$

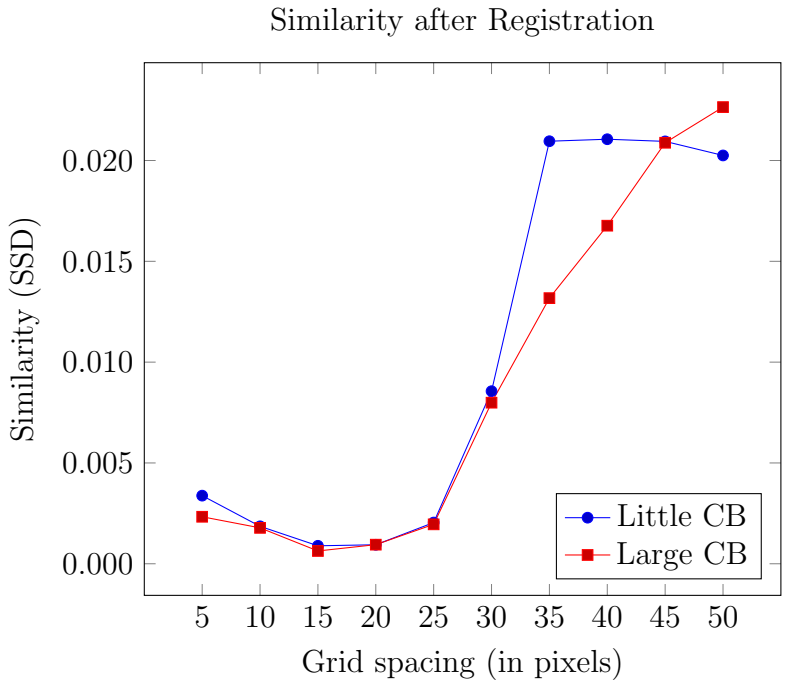


Figure 7.3: Similarity after registration

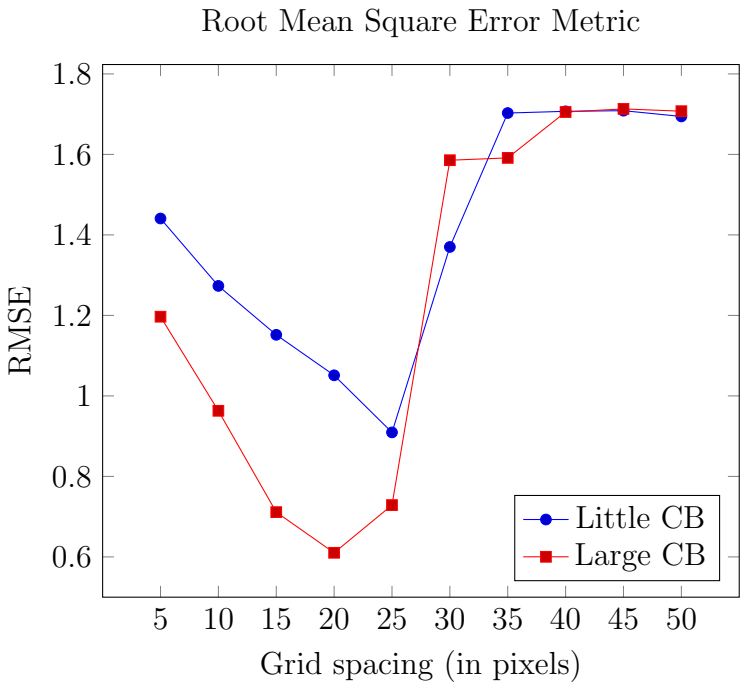


Figure 7.4: RMSE of the generated displacement field and the ground truth

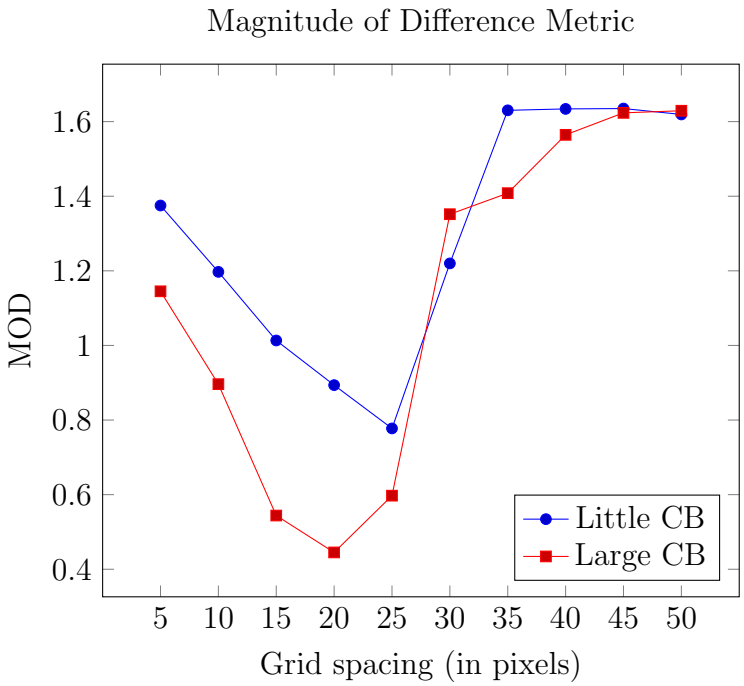


Figure 7.5: Magnitude of Difference of generated displacement field and ground truth

The results of this experiment can be viewed in Figure 7.5. Sharp averages indicate good registration results, meaning that a control point configuration around 20 pixels for the large checkerboard pattern and 25 pixels for the little checkerboard patttern gives the best result.

7.1.1.4 Processing Time

For this thesis, three versions of the image registration implementation were written. The naïve implementation described in Section 6.1.4.1 was implemented on GPU and on CPU. The optimized version described in Section 6.1.4.2 was only implemented on GPU. All three implementations will generate the same result, but will perform the registration at different speeds. Figure 7.6 shows the impact of different control point spacings on the three implemented registration versions. Both the GPU and CPU versions of the naïve implementation are susceptible to grid spacing, because a coarse grid spacing means bigger tiles to compute and a lot of redundant $\partial C_{sim}/\partial \overrightarrow{v}$ loads. The performed experiments showed that when comparing the GPU and CPU version of the naïve implementation, for a grid spacing of 5 pixels, the GPU performed about 40 times faster than the CPU. When reaching a grid spacing of 50 pixels, it only performed 3 times faster than the CPU.

The optimized GPU version, however, is rather agnostic to grid spacing. At a grid

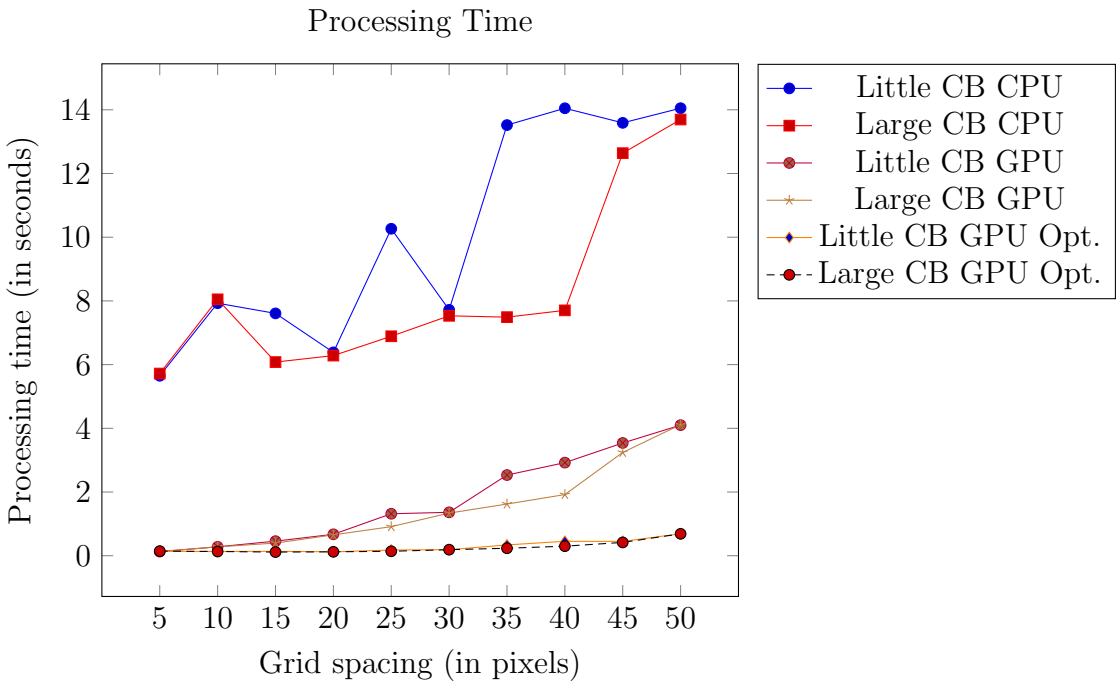


Figure 7.6: Processing time of registration using CPU, GPU and the optimized GPU versions

spacing of 10 pixels, it performed about 60 times faster than the naïve CPU version. When using a grid spacing of 50 pixels, the optimized version calculated its results about 20 times faster than its naïve CPU counterpart. Since the image registration framework is an essential part in finding the corpus callosum plane with minimal cross-sectional area, it is paramount that this part is properly optimized.

7.1.2 Medical Data

The deformable registration framework has also been evaluated using medical data. To assess the quality of the registration, two measurements were used, namely the *sum of squared differences* (SSD) measurement and the *correlation coefficient* (CC) measurement. SSD has already been described in this thesis in Section 3.4. The correlation coefficient measurement, however, is another similarity measure that is used frequently.

This measurement can be denoted as follows:

$$CC(I_f, I_m) = \frac{\sum_{\mathbf{p}\in\Omega}(I_f(\mathbf{p}) - \bar{I}_f)(I_m(\mathbf{T}(\mathbf{p})) - \bar{I}_m(\mathbf{T}(\mathbf{p})))}{\sqrt{\sum_{\mathbf{p}\in\Omega}(I_f(\mathbf{p}) - \bar{I}_f)^2 \sum_{\mathbf{p}\in\Omega}(I_m(\mathbf{T}(\mathbf{p})) - \bar{I}_m(\mathbf{T}(\mathbf{p})))^2}}, \tag{7.3}$$

where in this thesis I_f is the fixed image, I_m is the moving image, $I_m(\mathbf{T}(\mathbf{p}))$ is the transformed moving image, $\bar{I}_f$ is the average intensity of the fixed image, $\bar{I}_m(\mathbf{T}(\mathbf{p}))$ is the average intensity of the transformed moving image, and $\mathbf{p}$ denotes a certain pixel's coordinates. The correlation coefficient makes the implicit assumption that after registration, the images differ only by Gaussian noise. Another, however slight less strict, assumption is that when performing the registration, there is a linear relationship between the intensity values of both images [16]. The correlation coefficient calculation results in a value between -1 and 1, where 1 is total positive correlation, 0 is no correlation, and -1 means that there is total negative correlation.

To evaluate the performance of the registration framework on medical data, we have used three MRI data sets retrieved from the Open Access Series of Imaging Studies (OASIS) project[1] as well as a MRI volume obtained from icoMetrix[2]. The OASIS project is aimed at making MRI data sets of the brain available to the scientific community. The volume of icoMetrix has a size of $79 \times 95 \times 68$ voxels, while the three volumes from OASIS have a size of $128 \times 256 \times 256$ voxels. However, the three volumes from OASIS are anisotropic in nature. Each of the volumes have been resampled so that they are isotropic, which results in a volume of size $160 \times 256 \times 256$. For each of these volumes, we extracted a plane that roughly resembles the mid sagittal plane that would serve as the fixed image. Next, we also extracted from each volume a plane that has been rotated over several degrees from the central pixel of each volume's extracted mid sagittal plane. The reason behind the choice of these planes, is that this is exactly what will be done when searching for the plane with minimal cross-sectional corpus callosum area. Figure 7.7 contains the image differences from before (left) and after (right) the registration process. Ideally, the difference images would only show the underlying noise of the image acquisition. However, this not the case; the effect of misregistration due to motion is also visible in the difference images. One can see that the differences images on the right in Figure 7.7 show that the registration process compensates for motion, as the difference in the images has been significantly reduced.

Figures 7.8 and 7.9 show the results of the registration in terms of the sum of squared differences and correlation coefficient measurements respectively. The results clearly show that the registration process increases correlation between the used images and, by doing so, also increases the similarity.

[1] `http://oasis-brains.org`
[2] `http://www.icometrix.com`

7.2 Evaluation of MCAP Extraction

Figure 7.10 shows a graph of normalized corpus callosum areas for 8 MRI volumes agains optimizer iterations. 7 of these MRI volumes came from the OASIS project and the other one came from icoMetrix. The results for each volume have been normalized to the corpus callosum area at the start of the optimization process. This normalized area at the start of the iteration is that of the plane extracted from the volume using the parameters $R_x = R_y = T_z = 0$. The results show a distinct decrease and convergence of the corpus callosum area for almost all examined volumes.

For the examined volumes, we register a percentage drop ranged from a minimum of 1.4614%, a maximum of 7.3102% with a mean decrease in area of 4.36% and a median decrease of 4.67%. As stated by Ishaq [21] and longitudinal study performed by Juha [23], the minimization in corpus callosum area is potentially significant, given that approximately 33% of the reduction in corpus callosum area can be explained by atrophy.

We also tested the framework's robustness by using diffferent initializations. The results can be viewed in Figure 7.11. Each graph compares the mean normalized area after registration using different initialization parameters to the mean normalized area after registration using the standard initialization ($R_x = R_y = T_z = 0$). The figures, from top to bottom, were generated by varying one of the plane extraction parameters. The X axis of each plot denotes which parameter was variable. Each variable parameter was tested in a range from -2 to 2, with a step size of 1, while keeping the other two parameters set to zero. This range was chosen as Ishaq stipulated that one can expect the plane extraction parameters to have values within this range. In each plot, you can see the results of the standard initialization at $x = 0$. One can easily see that some parameter setting produce a result that is worse than that of the standard initialization, but some perform better. This can be explained by the fact that the patient's head was slightly rotated when capturing the MRI volume and that the template image used was not the most optimal central sagittal plane. By using an other than standard initalization, one can compensate for a head's rotation. If the correct central sagittal plane were to be used, the standard initialization would outperform any other initialization. These results underline the importance of the use of a decent template image and a correct input MRI volume.

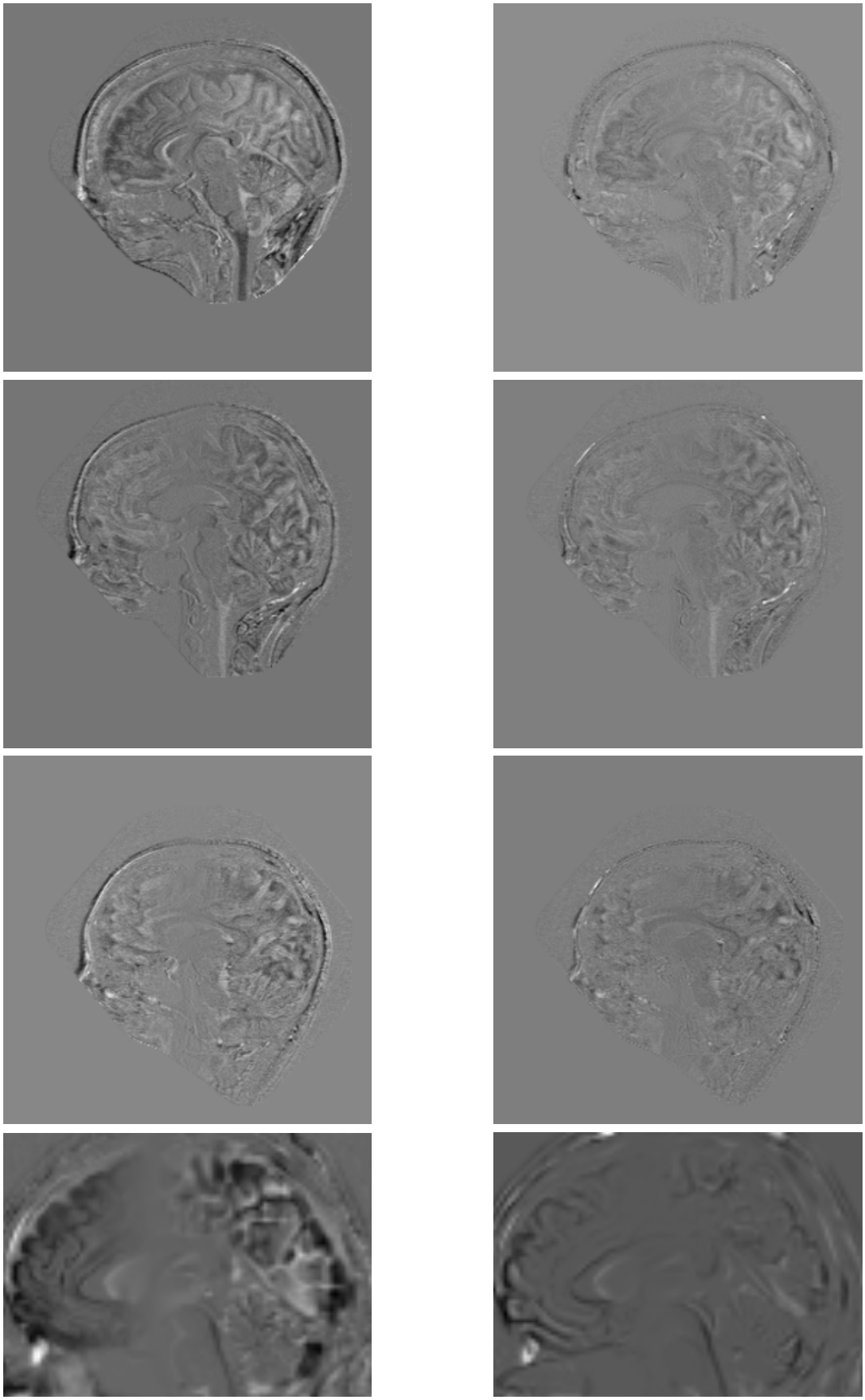


Figure 7.7: Image difference of the fixed image and warped moving image. Left is before the registration process and right is after the registration process.

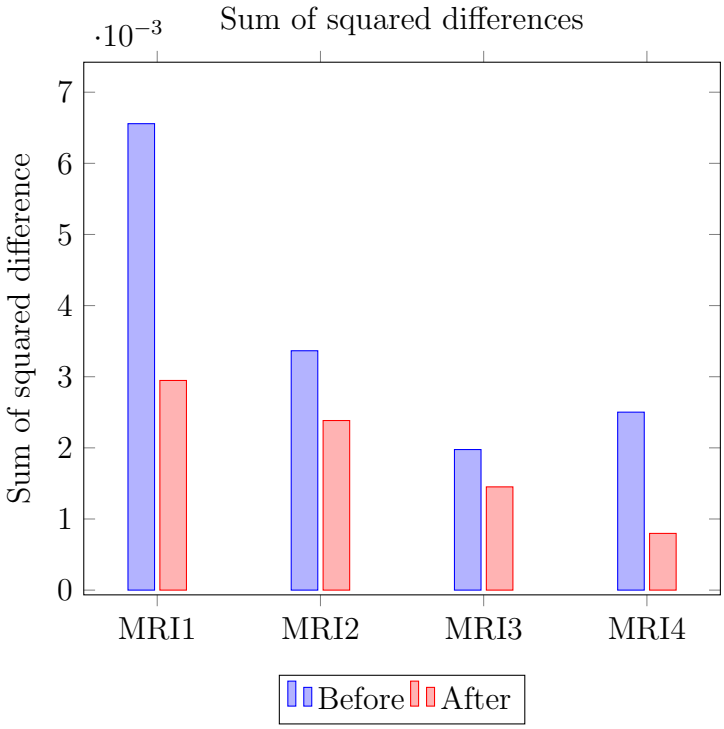


Figure 7.8: Sum of squared differences between the fixed image and the warped moving image before and after the registration process.

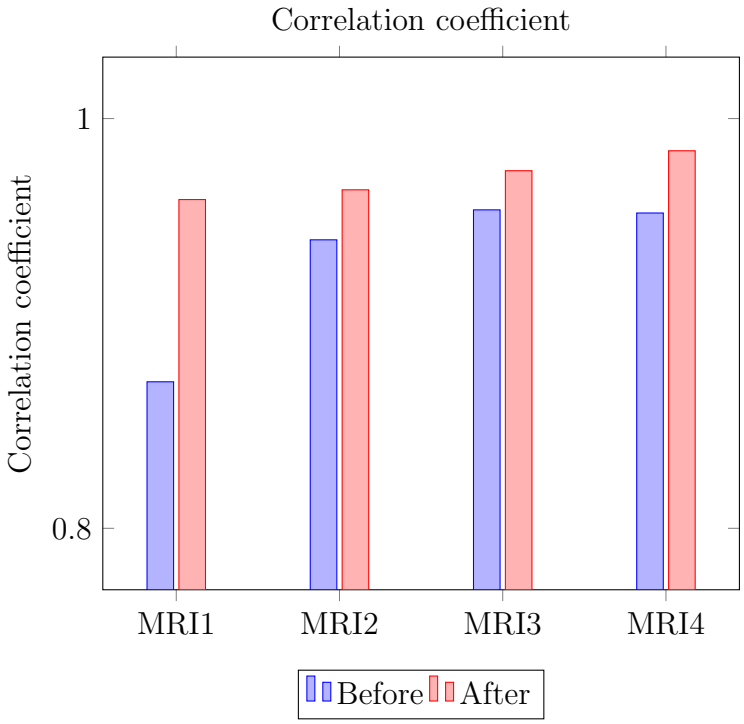


Figure 7.9: Correlation coefficient between the fixed image and the warped moving image before and after the registration process.

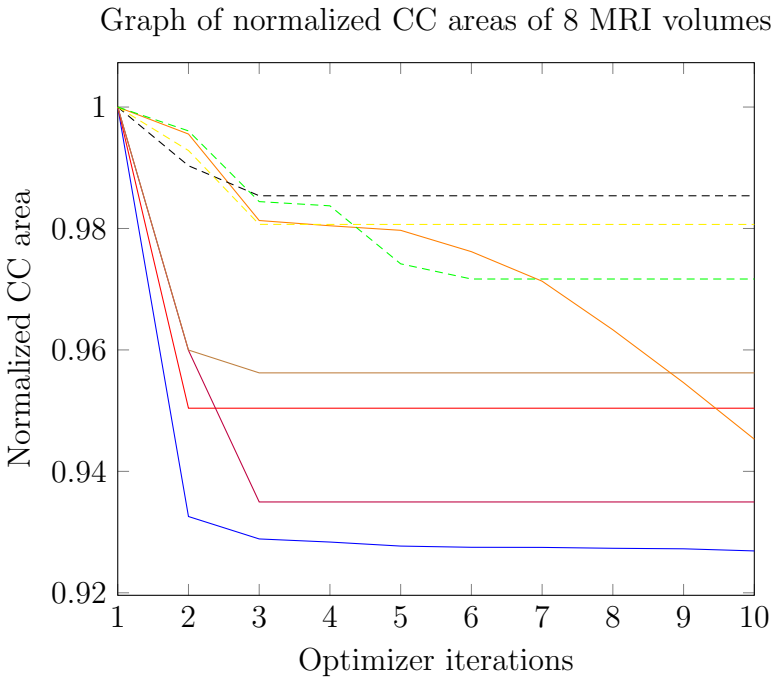


Figure 7.10: Graph of normalized CC areas of 8 MRI volumes

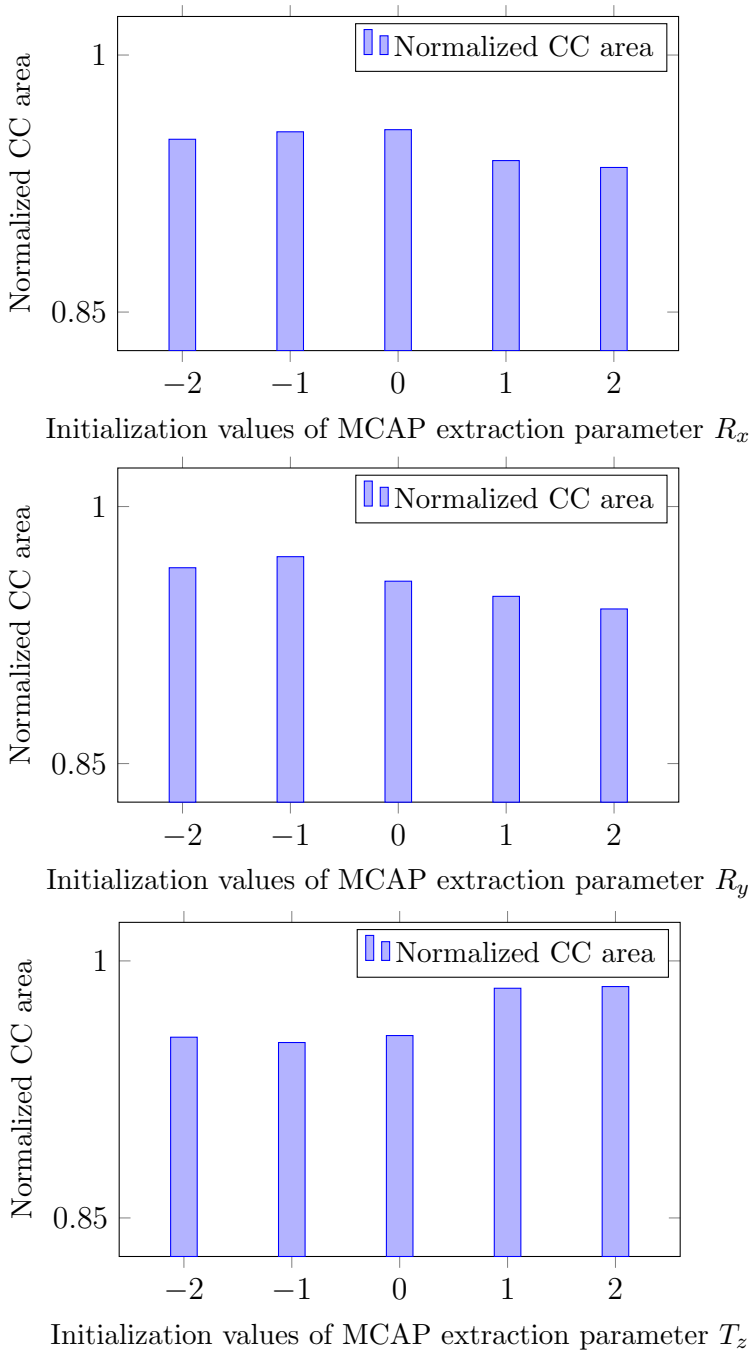


Figure 7.11: Comparison of normalized CC areas obtained using different initializations. Each bar represents the mean normalized area of the 8 MRI volumes used. For the top graph, an index $x = 1$ means that R_x was set to 1, while R_y and T_z were set to zero.

Chapter 8

Conclusion

This thesis contains the theoretical and practical aspects of deformable image registration, but also of finding the plane with minimal corpus callosum area has been described. Evaluation of both aspects has been conducted. This chapter will reiterate what was implemented and a few ideas for possible future work are also given.

8.1 Extraction of MCAP

The developed framework can automatically extract a plane with minimal corpus callosum area and simultaneously segment the corpus callosum. Keep in mind that this process is automatic after the generation of the template. This is a novel method, introduced by Ishaq, that can aid longitudinal studies. The method used, treats the corpus callosum area as a function of the plane extraction parameters and it uses deformable registration to generate a displacement field that can be used for the calculation of the corpus callosum area. The gathered results show a clear decrease and convergence to the plane with minimal corpus callosum area. However, one must make sure that the MRI volumes to be used are correctly orientated and that the used template image correctly contains the central sagittal plane. Were the optimal central sagittal plane used as the template, then the standard initialization would outperform any other initialization.

The obtained results cannot be compared to other, existing methods, because none of the existing methods try to achieve the same objective, which is the identification of minimal corpus callosum area and therefore, such a comparison would not be meaningful. As stated by Ishaq, this work can benefit future studies on longitudinal analysis of the change in corpus callosum area with the progression of different neurological diseases.

8.2 Deformable Image Registration

Deformable registration is a crucial part in finding the plane with minimal corpus callosum area. The algorithm for deformable registration has been described in detail. Free-form deformation has been used, which allows to model flexible deformations by means of a limited grid of control points, instead of manipulating each pixel individually as is done in deformable registration methods based on dense deformation fields. A regularization term is also used that will penalize deformations that are implausible. In other words, it remedies the ill-posedness of the deformable registration problem. The regularization term used in this thesis is applied to the grid of control points and not to the displacement field as is usually done.

Finding a plane with minimal corpus callosum area requires multiple image registrations, therefore, the registration framework has been optimized using CUDA. The algorithms, optimizations and data structures introduced in Section 6.1 reduce the complexity of the B-spline registration process. Highly parallel and scalable designs for computing both the sum of squared differences similarity measure and its derivative with respect to the B-spline parameterization were implemented. The speed and robustness of the image registration process were determined using both synthetic and medical data. The acceleration of the GPU process ranges from 20 to 60 times faster than the naïve CPU implementation, depending on the grid spacing used for the control points.

8.3 Future Work

Although a lot of work went into the implementation of the registration framework and for finding the plane of minimal corpus callosum, there are still some aspects that can be improved or require continued research.

- **Similarity measures.** This thesis used the sum of squared differences as a similarity measure for the deformable registration process. It serves its purpose well, since that the fixed images and template images used stem from the same MRI volume, it is a mono-modal problem and there is no need for a more complicated similarity measure. However, the framework may merit if other similarity measures were implemented. Mutual information was mentioned in Section 3.4 and it is used in multi-modal registration, since MI does not rely on image intensities directly.

- **Regularization.** Diffusion regularization was used in this thesis, but other regularizators exist. Curvature was mentioned in Section 5.2.2, and it can be worth it to study the results using this kind of regularizator. Another example, that was not mentioned in this thesis, is linear elasticity [52].

- **Optimization strategies.** The optimization strategy used in this thesis is gradient descent. There are other strategies that use gradients or approximate gradients.

Newtonian methods optimize a problem using second order deriviations, or the Hessian matrix. Calculating second order derviations, however, can be computationally expensive. Quasi-Newton methods, on the other hand, don't calculate the Hessian matrix directly, but they approximate it. The most popular quasi-Newton method is the *Broyden-Fletcher-Goldfarb-Shanno* algorithm, that can greatly increase the robustness of an optimization problem.

- **Additional evaluation.** Additional evaluation of the registration framework should be performed using medical data. Since different types of medical images have different characteristics, only further studies can reveal potential shortcomings of the registration framework.

- **Better CC segmentation methods.** Ishaq stated that with the potential development of better methods for corpus callosum segmentation, these methods can be incorporated in the MCAP extraction framework to replace the current registration based segmentation method.

- **Longitudinal studies.** Currently, no longitudinal MS data was in our possession. In the future, with the availability of longitudinal data for MS, the plane extraction framework should be employed for detecting change in the size of the corpus callosum with the progression of MS.

Bibliography

[1] S. Aylward, J. Jomier, S. Barre, B. Davis, and L. Ibanez. Optimizing ITK's Registration Methods for Multi-processor, Shared-memory Systems. *MICCAI Workshop on Open Source and Open Data*, 2007.

[2] A.H. Barr. Global and Local Deformations of Solid Primitives. *SIGGRAPH Computer Graphics*, 18(3):21–30, January 1984.

[3] D.P. Bersekas. *Nonlinear Programming*. Athena Scientific, 1999.

[4] F. Bookstein and D. Green. A feature space for derivatives of deformation. *Information Processing in Medical Imaging - Lecture Notes in Computer Science*, 687:1–16, 1993.

[5] E. Catmull and J. Clark. Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355, November 1978.

[6] G. Christensen, R. Rabbitt, and M. Miller. Deformable templates using large deformation kinematics. *IEEE Transactions on Image Processing*, 5(10):1435–1447, 1996.

[7] A. Compston and A. Coles. Multiple Sclerosis. *The Lancet*, 372(9648):1502–1517, 2008.

[8] T.M. Cover and J.A. Thomas. *Medical Image Registration*. John Wiley & Sons, Inc., 1991.

[9] C. Davatzikos, M. Vaillant, S.M. Resnick, J.L. Prince, S. Letovsky, and R.N. Bryan. A Computerized Approach for Morphological Analysis of the Corpus Callosum. *Journal of Computer Assisted Tomography*, 20(1):88–97, January-February 1996.

[10] K.G. Derpanis. The Gaussian Pyramid, 2005.

[11] T. Duning and S. Kloska. Dehydration confounds the assessment of brain atrophy. *Neurology*, 64(3):548–550, 2005.

[12] M. Filippi, M.A. Rocca, and N. De Stefano. Magnetic Resonance Techniques in Multiple Sclerosis: The present and the future. *Archives of Neurology*, 68(12):1514–1520, 2011.

[13] J.M. Fitzpatrick, D.L.G. Hill, Y. Shyrn, J. West, C. Studholme, and C.R. Maurer. Visual Assessment of the Accuracy of Retrospective Registration of MR and CT Images of the Brain. *IEEE Transactions on Medical Imaging*, 17(4):571–585, August 1998.

[14] D.R. Forsey and R.H. Bartels. Hierarchical B-spline Refinement. *SIGGRAPH Computer Graphics*, 22(4):205–212, June 1988.

[15] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Pearson International Edition, 2008.

[16] J.V. Hajnal, D.L.G. Hill, and D.J. Hawkes. *Medical Image Registration*. CRC Press, 2001.

[17] T. Hartkens, D.L. Hill, A.D. Castellano-Smith, D.J. Hawkes, C.R. Jr. Maurer, A.J. Martin, W.A. Hall, H. Liu, and C.L. Truwit. Measurement and analysis of brain deformation during neurosurgery. *IEEE Transactions on Medical Imaging*, 22(1):82–92, January 2003.

[18] D.L.G. Hill, P.G. Batchelor, M. Holden, and D.J. Hawkes. Medical Image Registration. *Physics in Medicine and Biology*, 46(3):R1–R45, March 2001.

[19] D.H. Hubel. *Eye, brain and vision*. WH Freeman, 1995.

[20] Neuroimaging Informatics Technology Initiative. Neuroimaging Informatics Technology Initiative: Background Information. `http://www.grahamwideman.com/gw/brain/orientation/orientterms.htm`, December 2013.

[21] O. Ishaq. Algorithms for Image Analysis of Corpus Callosum Degeneration for Multiple Sclerosis. Master's thesis, Simon Fraser University, 2008.

[22] B. Jähne. *Digital Image Processing*. Springer, 2002.

[23] M. Juha, S. Leszek, F. Sten, B. Jakob, F. Olof, and K.W. Maria. Non-age-related Callosal Brain Atrophy in Multiple Sclerosis: A 9-year Longitudinal MRI Study Representing Four Decades of Disease Development. *Journal of Neurology, Neurosurgery, and Psychiatry*, 78:375–380, 2007.

[24] J. Kybic and M. Unser. Fast parametric elastic image registration. *IEEE Transactions on Signal Processing*, 12(11):1427–1442, November 2003.

[25] S. Lee, G. Wolberg, and S.Y. Shin. Scattered Data Interpolation with Multilevel B-Splines. *IEEE Transactions on Visualization and Computer Graphics*, 3(3), September 1997.

[26] D. Loeckx. *Automated nonrigid intra-patient image registration using B-splines*. PhD thesis, Katholieke Universiteit Leuven, 2006.

[27] D. Loeckx, P. Slagmolen, F. Maes, D. Vandermeulen, and P. Suetens. Nonrigid Image Registration Using Conditional Mutual Information. *IEEE Transactions on Medical Imaging*, 29(1):19–29, January 2010.

[28] F. Lublin, S. Reingold, and National Multiple Sclerosis Society USA. Defining the clinical course of multiple sclerosis: Results of an international survey. *Neurology*, 46(4):907–911, April 1996.

[29] F. Maes, D. Vandermeulen, and P. Suetens. Medical Image Registration Using Mutual Information. *Proceedings of the IEEE*, 91(10):1699–1722, October 2003.

[30] J.B.A. Maintz and M.A. Viergever. A survey of medical image registration. *Medical Image Analysis*, 2(1):1–36, 1998.

[31] V.R.S. Mani and S. Arivazhagan. Survey of Medical Image Registration. *Journal of Biomedical Engineering and Technology*, 1(2):8–25, April 2013.

[32] E. Meijering. A Chronology of Interpolation: From Ancient Astronomy to Modern Signal and Image Processing. *Proceedings of the IEEE*, 90(3):319–342, March 2002.

[33] D.N. Metaxas. *Physics-Based Deformable Models: Applications to Computer Vision, Graphics and Medical Imaging*. Kluwer, 1997.

[34] J. Modersitzki. *Numerical Methods for Image Registration*. Oxford University Press Series: Numerical Mathematics and Scientific Computing, 2004.

[35] NVIDIA. NVIDIA's Next Generation CUDA Compute Architecture: Fermi. Technical report, NVIDIA Corporation, 2009.

[36] M. Otte and H.M. Nagel. Estimation of optical flow based on higher-order spatiotemporal derivatives in interlaced and non-interlaced image sequences. *Artificial Intelligence*, 78(1-2):5–43, October 1995.

[37] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C - The Art of Scientific Computing*. Cambridge University Press, 1992.

[38] W. Rashid and D. Miller. Recent Advances in Neuroimaging of Multiple Sclerosis. *Semin Neurol*, 28(1):46–55, July 2008.

[39] G.K. Rohde, A. Aldroubi, and B.M. Dawant. The adaptive bases algorithm for intensity-based nonrigid image registration. *IEEE Transactions on Medical Imaging*, 22(11):1470–1479, 2003.

[40] D. Rueckert and P. Aljabar. Non-Rigid Registration of Medical Images: Theory, Methods, and Applications. *Signal Processing Magazine, IEEE*, 27(4):113–119, July 2010.

[41] D. Rueckert, L.I. Sonoda, C. Hayes, D.L.G. Hill, M.O. Leach, and D.J. Hawkes. Nonrigid Registration Using Free-Form Deformations: Application to Breast MR Images. *IEEE Transactions on Medical Imaging*, 18(8), August 1999.

[42] J. Sanders and E. Kandrot. *CUDA by Example: An Introduction to General-Purpose GPU Computing*. Addison-Wesley Professional, 2010.

[43] I.J. Schoenberg. Contributions to the Problem of Approximation of Equidistant Data by Analytic Functions. *The Quarterly of Applied Mathematics*, 4:45–99, 1946.

[44] L.A. Schwarz. Non-rigid Registration Using Free-from Deformations. Master's thesis, Technische Universität München, 2007.

[45] T.W. Sederberg and S.R. Parry. Free-form Deformation of Solid Geometric Models. *SIGGRAPH Computer Graphics*, 20(4):151–160, August 1986.

[46] J.A. Shackleford. *High-Performance Image Registration Algorithms for Multi-Core Processors*. PhD thesis, Drexel University, 2011.

[47] J.A. Shackleford, N. Kandasamy, and G.C. Sharp. On developing B-spline registration algorithms for multi-core processors. *Physics in Medicine and Biology*, 55:6329–6351, October 2010.

[48] C.E. Shannon. Communication in the Presence of Noise. *Proceedings of the IEEE*, 86(2):447–457, February 1998.

[49] G.C. Sharp, M. Peroni, R. Li, J.A. Shackleford, and N. Kandasamy. Evaluation of Plastimatch B-Spline Registration on the EMPIRE10 Data Set. *Medical Image Analysis for the Clinic: A Grand Challenge*, pages 99–108, 2010.

[50] J.H. Simon. Brain Atrophy in Multiple Sclerosis: What We Know and Would Like to Know. *Multiple Sclerosis*, 12(6):679–687, December 2006.

[51] J.H Simon, L.D. Simon, M.K. Campion, R.A. Rudick, D.L. Cookfair, R.M. Herndon, J.R. Richert, A.M. Salazar, J.S. Fischer, D.E. Goodkin, N. Simonian, M. Lajaunie, D.E. Miller, K. Wende, A. Martens-Davidson, R.P. Kinkel, F.E. Munschauer, and C.M. Brownscheidle. A Longitudinal Study of Brain Atrophy in Relapsing Multiple Sclerosis. *Neurology*, 12(6):679–687, 2006.

[52] W.S. Slaughter. *The Linearized Theory of Elasticity*. Birkhuser, 2002.

[53] J.A. Snyman. *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms*. Springer Science+Business Media, Inc., 2005.

[54] G. Stockman and L.G. Shapiro. *Computer Vision*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[55] R. Szeliski and J. Coughlan. Spline-Based Image Registration. *International Journal of Computer Vision*, 22(3):199–218, 1997.

[56] J.P. Thirion. Image matching as a diffusion process: an analogy with Maxwell's demons. *Medical Imaging Analysis*, 2(3):243–260, 1998.

[57] P. Thompson and A. Toga. A surface-based technique for warping three-dimensional images of the brain. *IEEE Transactions on Medical Imaging*, 15(4):402–417, 1996.

[58] M. Unser. Splines: A Perfect Fit for Signal and Image Processing. *IEEE Signal Processing Magazine*, 16(6):22–38, November 1999.

[59] M. Unser, A. Aldroubi, and M. Eden. B-Spline Signal Processing: Part I - Theory. *IEEE Transactions on Signal Processing*, 41(2):821–832, February 1993.

[60] P. Viola and W.M. Wells. Alignment by Maximization of Mutual Information. *International Journal of Computer Vision*, 24(2):137–154, 1997.

[61] W.M. Wells, P. Viola, H. Atsumi, S. Nakajima, and R. Kikinis. Multi-modal volume registration by maximization of mutual information. *Medical Image Analysis*, 1(1):35–51, 1996.

[62] G. Wideman. Orientation and Voxel-Order Terminology: RAS, LAS, LPI, RPI and XYZ. `http://www.grahamwideman.com/gw/brain/orientation/orientterms.htm`, June 2003.

[63] T. Zhang, Y. Chi, E. Meldolesi, and D. Yan. Automatic delineation of on-line head-and-neck computed tomography images: toward on-line adaptive radiotherapy. *International Journal of Radiation Oncology, Biology, Physics*, 68(2):522–530, June 2007.

Auteursrechtelijke overeenkomst

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
Computational science: Medical imaging using CUDA

Richting: **master in de informatica-multimedia**
Jaar: **2014**

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Voor akkoord,

Kovac, Thomas

Datum: **23/06/2014**