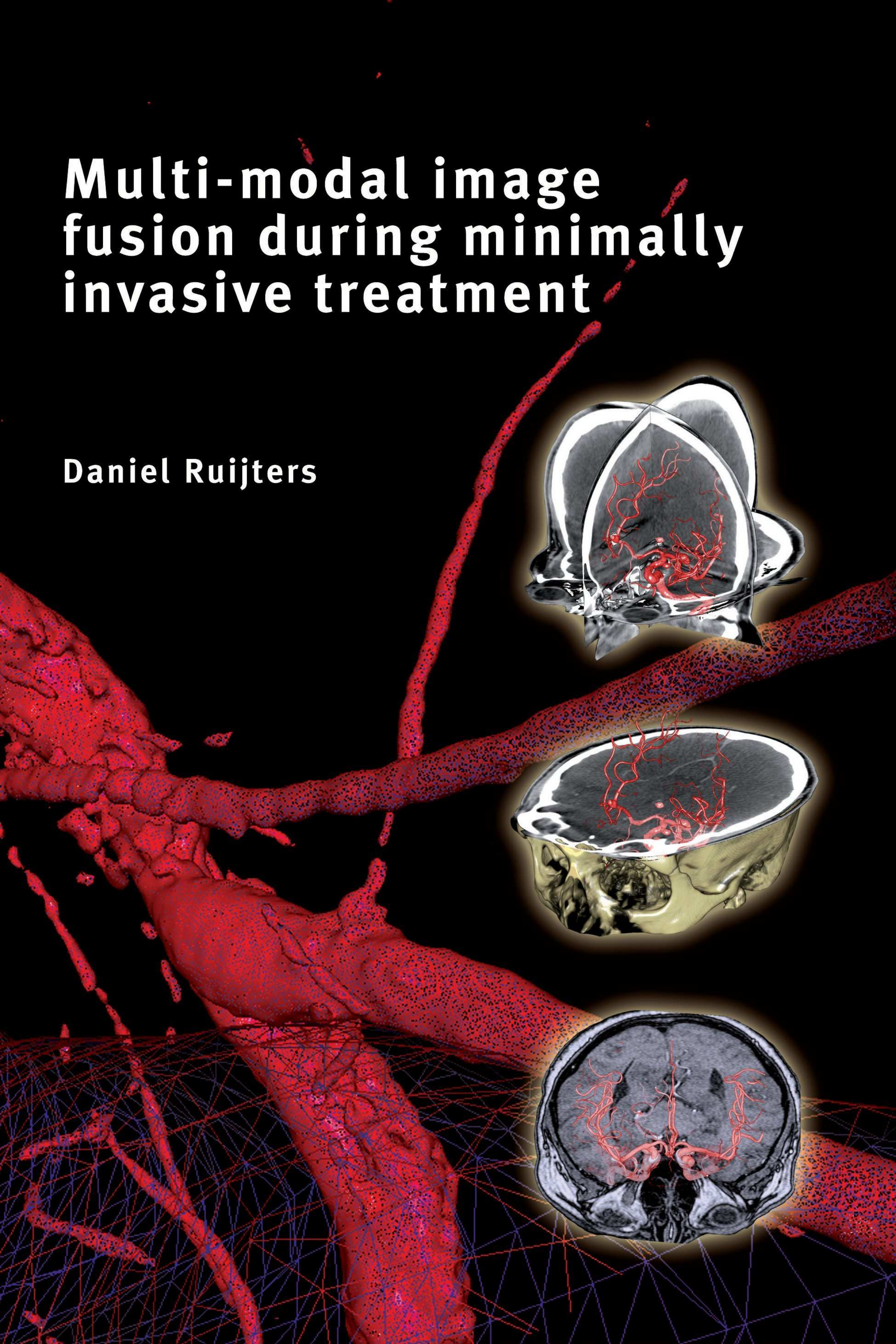
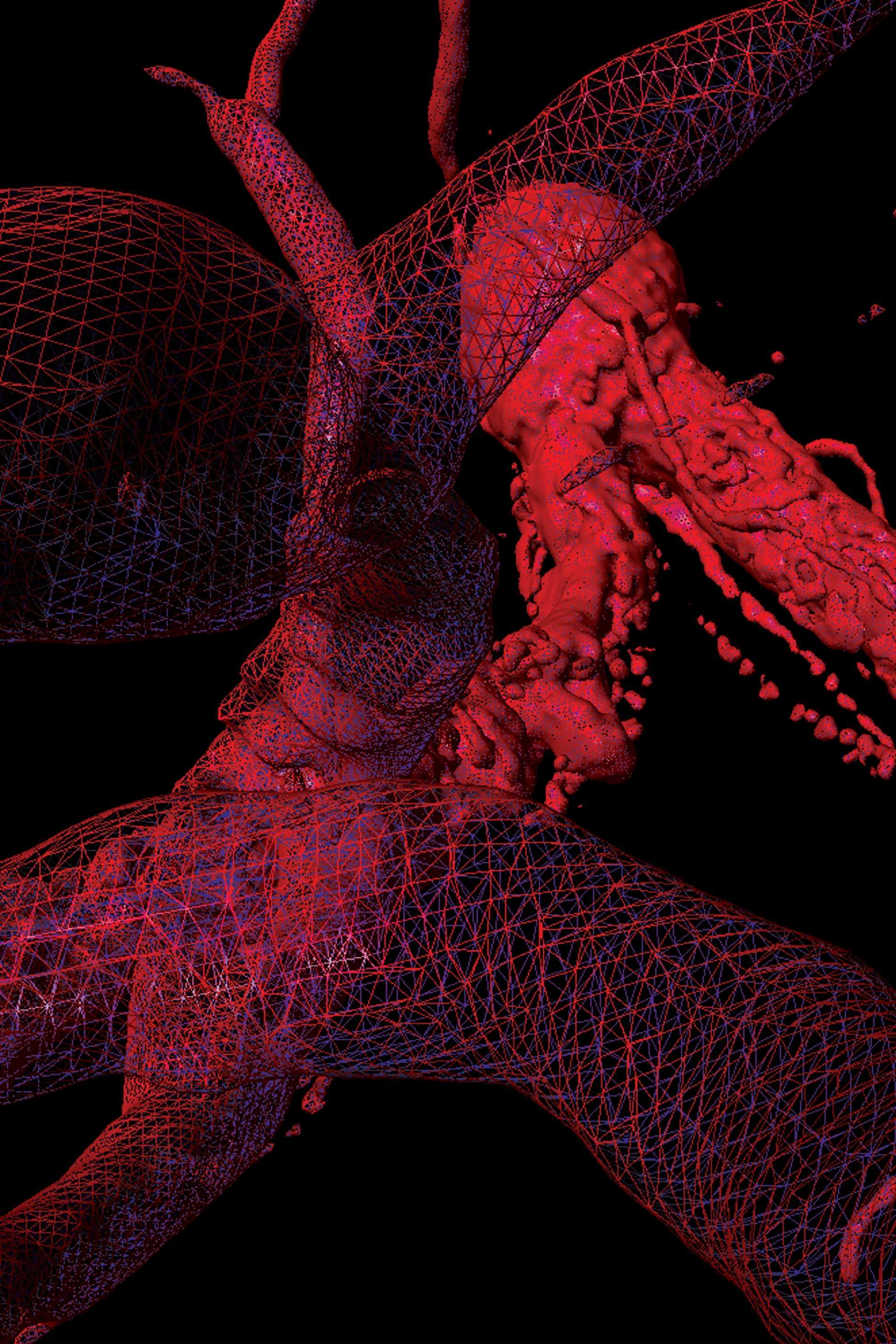
Multi-modal image fusion during minimally invasive treatment

Daniel Ruijters





KATHOLIEKE UNIVERSITEIT LEUVEN
FACULTEIT INGENIEURSWETENSCHAPPEN
DEPARTEMENT ELEKTROTECHNIEK (ESAT),
AFDELING PSI
Kasteelpark Arenberg 10, 3001 Heverlee (Belgium)
FACULTEIT GENEESKUNDE
DEPARTEMENT MORFOLOGIE EN MEDISCHE
BEELDVORMING, AFDELING RADIOLOGIE
Herestraat 49, 3000 Leuven (Belgium)

TU/e Technische Universiteit
Eindhoven
University of Technology
Faculteit Biomedische Technologie
BioMedical Image Analysis
PO Box 513, 5600 MB Eindhoven (the Netherlands)

Multi-modal image fusion during minimally invasive treatment

Jury:
Prof. Dr. P. A. J. Hilbers
(voorzitter publieke verdediging)
Prof. Dr. ir. Y. Willems
(voorzitter preliminaire verdediging)
Prof. Dr. ir. P. Suetens (promotor)
Prof. Dr. ir. B. M. ter Haar Romeny (promotor)
Prof. Dr. ir. D. Vandermeulen
Prof. Dr. ir. F. Maes
Prof. Dr. G. Marchal
Prof. Dr. ir. F. N. van de Vosse
Prof. Dr. N. H. J. Pijls
Prof. Dr. ir. J. J. van Wijk
Prof. Dr. I. Bloch

Proefschrift voorgedragen tot
het behalen van het doctoraat
in de ingenieurswetenschappen

door

Daniël RUIJTERS

ISBN: 978-94-6018-174-0
Wett. dep. D/2010/7515/13

15 February 2010

Copyright © 2010 by Daniel Ruijters

Alle rechten voorbehouden. Niets uit deze uitgave mag worden vermenigvuldigd en/of openbaar gemaakt worden door middel van druk, fotokopie, microfilm, elektronisch of op welke andere wijze ook, zonder voorafgaande schriftelijke toestemming van de uitgever.

All rights reserved. No part of this publication may be reproduced in any form by print, photoprint, microfilm or any other means without written permission from the publisher.

ISBN: 978-94-6018-174-0
Wett. dep. D/2010/7515/13

Multi-modal image fusion during minimally invasive treatment

Proefschrift

ter verkrijging van de graad van doctor aan de
Technische Universiteit Eindhoven, op gezag van de
rector magnificus, prof.dr.ir. C.J. van Duijn, voor een
commissie aangewezen door het College voor
Promoties in het openbaar te verdedigen
op maandag 15 februari 2010 om 17.00 uur

door

Daniël Simon Anna Ruijters

geboren te Heerlen

Dit proefschrift is goedgekeurd door de promotoren:

prof.dr.ir. B.M. ter Haar Romeny
en
prof.dr.ir. P. Suetens

to my beautiful daughter Eva

Committee:

prof.dr. P.A.J. Hilbers	Technische Universiteit Eindhoven, the Netherlands.
prof.dr.ir. Y. Willems	Katholieke Universiteit Leuven, Belgium.
prof.dr.ir. B.M. ter Haar Romeny	Technische Universiteit Eindhoven, the Netherlands.
prof.dr.ir. P. Suetens	Katholieke Universiteit Leuven, Belgium.
prof.dr.ir. F.N. van de Vosse	Technische Universiteit Eindhoven, the Netherlands.
prof.dr. N.H.J. Pijls	Technische Universiteit Eindhoven, the Netherlands.
prof.dr.ir. J.J. van Wijk	Technische Universiteit Eindhoven, the Netherlands.
prof.dr.ir. D. Vandermeulen	Katholieke Universiteit Leuven, Belgium.
prof.dr.ir. F. Maes	Katholieke Universiteit Leuven, Belgium.
prof.dr. G. Marchal	Katholieke Universiteit Leuven, Belgium.
prof.dr. I. Bloch	Ecole Nationale Supérieure des Télécommunications, Paris, France.

CIP-DATA LIBRARY TECHNISCHE UNIVERSITEIT EINDHOVEN

Daniel Ruijters

Multi-modal image fusion during minimally invasive treatment/ by Daniel Ruijters. - Eindhoven : Technische Universiteit Eindhoven, 2010.
A catalogue record is available from the Eindhoven University of Technology Library
ISBN: 978-94-6018-174-0
NUR 959
Trefw.: beeld gestuurde interventies / minimaal invasieve behandeling / multi-modaal / beeldregistratie en fusie / 3D beeldverwerking.
Subject headings: image guided interventions and therapy / minimal invasive treatment / multimodality / image registration and fusion / 3D image processing.

Copyright © 2010 by Daniel Ruijters

All rights reserved. No part of this material may be reproduced or transmitted in any form or by any means, electronic, mechanical, including photocopying, recording or by any information storage and retrieval system, without the prior permission of the copyright owner.

The publication of this work has been kindly sponsored by:

PHILIPS
sense and simplicity

Philips Healthcare, interventional X-Ray (iXR)

TU/e Technische Universiteit
Eindhoven
University of Technology

Technische Universiteit Eindhoven, Faculteit Biomedische Technologie

KATHOLIEKE UNIVERSITEIT
LEUVEN

Katholieke Universiteit Leuven, Faculteit Ingenieurswetenschappen

Acknowledgements

“No man is an island”. This line by the English poet John Donne (1572-1631) is certainly applicable when conducting a PhD thesis. Therefore I would like to thank all those who have provided me with the building blocks that allowed me to assemble this thesis.

There are many people who contributed in one way or another to this work. First of all, I would like to thank my promoters, Prof. Paul Suetens and Prof. Bart ter Haar Romeny. When I approached you with the idea to perform a PhD thesis in an unorthodox construction, involving two universities and a company, located in two countries, both of you immediately reacted with enthusiasm. Throughout the thesis you provided me with valuable feedback and reflections, while you offered me the freedom to determine the course of my investigations. At the end of the thesis I have experienced that an unusual construction can only be supported with the help of some flexibility and efforts from the side of the administration, board and secretaries of both universities. Thanks to everybody involved! Also I would like to express my gratitude to Philips, and in particular Jan Timmer and Hein Haas. You allowed me to carry out this thesis while being employed by Philips, and provided me with subjects and tasks that both served Philips and were of scientific relevance to my thesis. Further I want to thank Nijs van der Vaart and Eric von Reth for reviewing the many publications that I wrote, and allowing me to publish them.

I would like to show gratitude to the people who directly contributed to the content of this thesis and who often acted as co-authors of my scientific papers. Robert Homan and Peter Mielekamp, you were the principal engineers working on the wonderful 3D roadmapping and needle guidance applications, which provided me with a great platform to extent with multi-modality capabilities. Drazenko Babic, you provided me with the clinical views and needs, and you always stimulated me to come up with new ideas. I would like to express my appreciation to Niels Bakker and Onno Wink for your input regarding the cardiac applications. I have received valuable feedback and data from several colleagues at Philips and both universities, and in this context I would like to particularly mention Gert Schoonenberg, Yannick Morvan, Niels Noordhoek, Peter Eshuis, Anna Vilanova, Marijke Vermeer and Dirk Loeckx. Further I would like to acknowledge the jury members of this thesis. You provided very useful hints for improving this thesis manuscript, which I have gladly embraced. Also I want to thank the hospital staff that was involved in clinically evaluating the applications that were developed in the context of this thesis. The Philips service engineers and local clinical scientists have been essential in the successful installation of these applications. I am especially grateful to Rens Schoones, who helped to solve many critical issues that arose while I was trying to install my prototypes in hospitals far away, often located in a different time zone.

Thanks to the many nice colleagues at Philips, the TU/e and the KU Leuven. You have always provided a friendly and fruitful environment. Bram, Peter, Peter, Peter, Thijs, Robin, Robert-Jan, Rob, Roel, Niels, Niels, Sander, André, Wiet, Erik, Mark, Benno, Javier, Jaap, Edwin, Wietse, Jeremy, Ettore, John, Kasper, Casper, Jan, Fehim, Arno, Srood, Leendert, Arjen, Richard, Marcel, Wilfred, Everine, Alessandro, Angelique, Bart, Geert, Maarten, Ruud, Ruud, Rob, Chrysi, Liesbet, Dominique, Jeroen, An, Pieter, Pieter, Ellen, Alessandro, Hans, and all others, thank you! Of course I also would like to thank my parents, family, friends, my beloved Sveta, and very recently my beautiful Eva.

As Isaac Newton has said: “If I have seen a little further it is by standing on the shoulders of Giants”. None of this work would have been possible without the accomplishments of many who have remained unnamed here. The work performed by the scientific community, at Philips Research, and Philips development has formed the solid base on which this thesis could be built. I am grateful to all of you!

Danny Ruijters,
Eindhoven, February 2010

Abstract

Multi-modal image fusion during minimally invasive treatment

The volume of image guided interventions and therapy is rapidly increasing, because of associated better clinical outcome and reduced patient strain. During such minimally invasive medical treatment the clinician relies on radiological images, often produced in real-time, to guide the intervention. Prior to treatment, diagnosis and intervention planning is frequently performed using tomographical images, which provide a detailed representation of the patient's anatomy and pathology. In this thesis the fusion of those different types of images is presented, in order to provide the clinician with more relevant data to guide the procedure. Since the fusion is performed during the clinical intervention, it is essential that the technical steps can be executed within limited time. Furthermore, it is vital that the resulting fused representations are easy to interpret. The technical approaches that are described here to achieve this goal comprise fast and intuitive visualization of the fused data and rapid co-registration of multiple image datasets.

In order to achieve an optimal performance the parallel computation power of the graphics processing unit (GPU) has been exploited in the visualization and registration algorithms. Regarding visualization, a dedicated direct volume rendering approach was developed, taking the particularities of the GPU into account. This volume rendering technique has been applied in the efficient fused visualization of multiple datasets, and in the interactive rendering for autostereoscopic displays. An elastic B-spline driven registration method has been mapped on the GPU to accomplish minimal computation times. Furthermore, registration algorithms especially designed for peri-interventional usage were examined. A registration method only using the geometry information of the X-ray C-arm system has been described, and a dedicated registration algorithm targeted at real-time vascular images has been developed.

The proposed techniques have been validated individually, and have been evaluated together in three concrete clinical applications: multi-modal roadmapping for neuro-vascular treatment, multi-modal needle puncture planning and tracking, and CT fusion with X-ray angiography for stent placement within coronary artery disease treatment.

Multi-modale beeldfusie tijdens minimaal-invasieve behandelingen

Het aantal beeldgestuurde medische interventies neemt snel toe, vanwege de geassocieerde verbeterde klinische resultaten. Gedurende minimaal invasieve medische behandelingen vertrouwt de arts enkel op radiologische beelden om het verloop van de interventie te sturen. De diagnose en het behandelplan is vaak voorafgaand aan de behandeling uitgevoerd op basis van tomografische beelden. Deze bieden een gedetailleerde afspiegeling van de anatomie en pathologie van de patiënt. In dit proefschrift wordt de fusie van deze verschillende typen van beelden voorgesteld, om de arts meer relevante data om de procedure te leiden aan te bieden. Aangezien deze fusie tijdens de klinische interventie wordt verricht, is het essentieel dat de noodzakelijke technische stappen binnen een beperkte tijdsduur kunnen worden uitgevoerd. Verder is het van vitaal belang dat de resulterende beelden eenvoudig zijn te interpreteren tijdens de behandeling. De technische stappen, die in dit proefschrift worden beschreven, omvatten snelle en intuïtieve visualisatie van de gefuseerde data en snelle co-registratie van meerdere beelddatasets.

De parallelle rekenkracht van de grafische processor eenheid (GPU) wordt benut om optimale prestaties te behalen in de visualisatie- en registratiealgoritmes. Met betrekking tot visualisatie is er een directe volume rendering techniek ontwikkeld die rekening houdt met de specifieke eigenschappen van de GPU. Deze volume rendering techniek is vervolgens toegepast in de efficiënte gezamenlijke visualisatie van meerdere datasets, en in de interactieve rendering voor autostereoscopische schermen. Een elastische registratiemethode, gebaseerd op B-splines, is op de GPU geïmplementeerd om minimale rekentijden te bereiken. Verder zijn registratiealgoritmes onderzocht die bedoeld zijn voor peri-interventioneel gebruik. Een registratiemethode die enkel de geometrie informatie van het Röntgen C-arm systeem gebruikt is beschreven, en er is een registratiealgoritme ontwikkeld dat speciaal toegesneden is op real-time vasculaire beelddata.

De voorgestelde technische oplossingen zijn individueel gevalideerd, en zijn samengesteld geëvalueerd in drie concrete klinische toepassingen: multimodale roadmapping voor neurovasculaire behandeling, multimodale naald punctie planning en navigatie, en CT fusie met angiografische Röntgen voor het plaatsen van stents in vernauwingen in de kransslagader.

Nederlandse samenvatting

Multi-modale beeldfusie tijdens minimaal-invasieve behandelingen

1 Achtergrond

Bij een minimaal invasieve operatie wordt de patiënt behandeld via katheters, naalden of andersoortige instrumenten. Dit in tegenstelling tot traditionele chirurgie, waarbij de patiënt opengesneden wordt om de pathologie te behandelen. De navigatie van de minimaal invasieve instrumenten in het lichaam van de patiënt gebeurt met de hulp van medische beeldvormende apparatuur, zoals Röntgen en ultrasound. Aangezien dit type behandelingen minder trauma veroorzaken, worden ze over het algemeen geassocieerd met kortere verkoevertijden en betere klinische resultaten. Minimaal invasieve technieken worden dan ook ingezet voor een steeds groter palet van aandoeningen, en het volume per aandoening neemt eveneens toe. Voorafgaand aan de invasieve behandeling wordt er vaak een drie dimensionale tomografische afbeelding in een CT of MRI scanner gemaakt. Deze scan bevat een gedetailleerde afspiegeling van de anatomie en pathologie van de patiënt en wordt daarom gebruikt tijdens de diagnose en ook voor het opstellen van een behandelplan.

In dit proefschrift worden technieken geïntroduceerd en uitgediept om de verschillende beschikbare beeldinformatiebronnen te mengen gedurende de behandeling. Deze technieken betreffen visualisatiemethoden en methoden voor de beeldregistratie van de verschillende beelden. De nadruk wordt in dit proefschrift gelegd op snelle algoritmes en intuïtieve visualisatie. De parallelle rekenkracht van de grafische processor eenheid (GPU) wordt benut om optimale prestaties te behalen in de visualisatie- en registratiealgoritmes. Snelheid is zeer belangrijk aangezien het rekenwerk wordt uitgevoerd terwijl de klinische interventie aan de gang is. Intuïtieve interactie is eveneens essentieel, aangezien de arts zijn aandacht moet verdelen over het verloop van de behandeling en de stimuli die van de patiënt en de vele apparaten in de operatieruimte afkomstig zijn. Bovendien leidt een intuïtieve interactie tot een kleinere kans op het maken van fouten.

2 Visualisatie

De intra-operatieve setting en dynamiek verschillen behoorlijk op belangrijke punten van die van een diagnostische omgeving. Dit resulteert in andere functionele eisen die aan de intra-operatieve visualisatie worden gesteld. De waarnemer van intra-operatieve beelden zit typisch niet achter een desktop werkstation, maar staat achter de behandeltafel waarop de patiënt zich bevindt. Dit beperkt de manier van interactie met het werkstation. Verder is de primaire focus van de behandelende arts bij voorkeur gericht op de patiënt en het verloop van de behandeling, en niet op de interactie met de computer. Aangezien de navigatie van de minimaal invasieve instrumenten geschiedt aan de hand van de live beelden die middels de beeldvormende apparatuur worden gemaakt, zijn deze beelden van eminent belang. Toch vormen zij slechts een van de bronnen van stimuli, die de arts moet verwerken tijdens de behandeling. Verder is het ook van belang om te realiseren dat er slechts beperkte tijd beschikbaar is om de beelden te interpreteren, zeker als er stress situaties optreden tijdens de behandeling.

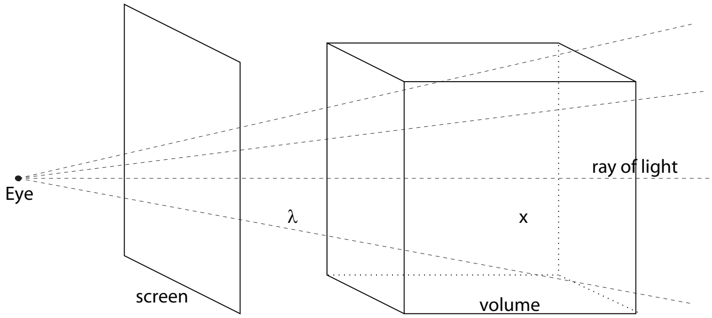
Al deze factoren leiden tot de eis dat de interactie met de beelden eenvoudig is en dat de intra-operatieve visualisaties makkelijk te interpreteren zijn, zonder compromissen te sluiten in de visualisatie van de klinisch relevante details. Dit maakt populaire diagnostische visualisatie methoden voor gefuseerde datasets zoals zij-aan-zij of schaakbord presentaties van 2D dwarsdoorsneden ongeschikt, aangezien zij teveel interactie vereisen en het teveel tijd kost om ze te interpreteren. In plaats daarvan is een eenvoudige 3D weergave vereist, die alle relevante details toont. Dit is een uitdagende opgave, aangezien gefuseerde data een zeer grote hoeveelheid informatie binnen een beperkt volume pakt, en de resulterende visualisatie vaak moeilijk in een oogopslag te behappen is. Verder is er vaak veel tijd nodig om een afbeelding uit de enorme hoeveelheid data te genereren, hetgeen interactieve manipulatie bemoeilijkt. Dit proefschrift beoogt om technische oplossingen te vinden voor de geschetste problemen.

2.1 Snelle volume rendering

Volume Rendering is een methode om drie dimensionale volumetrische voxel data direct (dus zonder voorbewerking) af te beelden op een twee dimensionaal vlak. Tijdens het uitvoeren van deze methode worden optische eigenschappen zoals kleur en transparantie aan elk punt in de continue ruimte toegekend. Dit gebeurd door de scalaire waarden op de discrete voxelposities te interpoleren in de continue ruimte. De scalaire waarden worden vervolgens afgebeeld op kleur en transparantie waarden middels een transferfunctie. De twee dimensionale afbeelding wordt verkregen door de volgende formule toe te passen op lichtstralen die door de continue ruimte gevolgd worden:

$$i = \int\limits_0^\infty c(x) \cdot e^{-\int\limits_0^x \tau(x')\,dx'} dx \tag{0.1}$$

Hier representeert i de resulterende kleur op de 2D afbeelding, $c(x)$ de kleur op positie x en $\tau(x)$ de lichtabsorptie (het tegenovergestelde van transparantie) op die positie,



Figuur 1: *Het volume render proces; de lichtstralen door de pixels van het scherm worden door het voxel volume geprojecteerd, en de volume render formule 0.1 wordt op deze trajecten toegepast.*

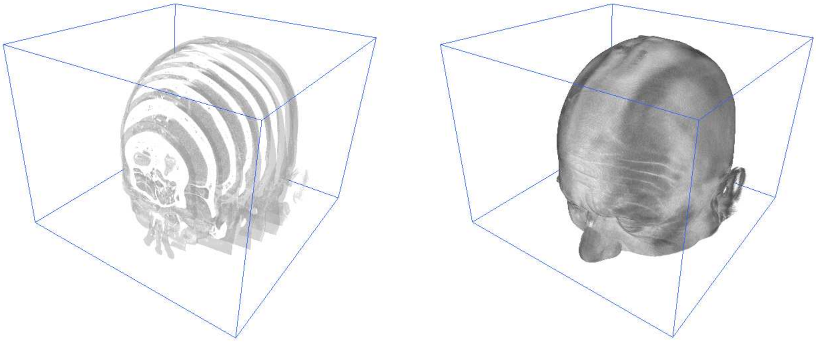
zie figuur 1. Deze volume render vergelijking kan worden benadert door een aantal discrete samples op de te volgen straal te nemen en in de volgende sommatie in te vullen:

$$i = \sum_{n=0}^{N} (\alpha_n \, c_n \cdot \prod_{n'=0}^{n} (1 - \alpha_{n'})) \tag{0.2}$$

Waarbij α_n voor de opaciteit en c_n voor de kleur van sample n staat. Deze formule kan zeer efficiënt worden uitgevoerd op de grafische hardware door op regelmatige afstanden dwarsdoorsneden van het volume te nemen, en deze middels zogenaamde alfa-blending met elkaar te mengen [1–8], zie figuur 2.

Bij volume rendering draagt slechts een heel klein gedeelte van alle voxels bij aan het eindresultaat. Dit komt doordat de meeste voxels volledig transparant zijn of verdekt worden door andere delen van het volume. Daarbij kunnen veel datasets als relatief ’leeg’ worden aangemerkt; meestal bevat slechts 5% tot 40% van alle voxels zichtbare data, en zelfs zeer gevulde CT en MR datasets overschrijden de 55% zeer zelden. Met name vasculaire datasets zijn vaak ’leeg’, aangezien de bloedvaten vanwege hun vorm een klein gedeelte van het volume vullen (typisch 1% tot 8%).

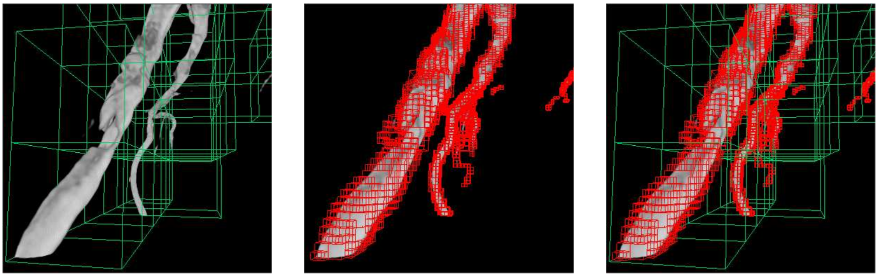
De leegte van het voxel volume kan worden benut om het volume proces te versnellen. Deze zogenaamde ’space-skipping’ strategie is reeds langere tijd bekend in de literatuur [9–12]. In het kader van dit proefschrift is er een ’space-skipping’ methode ontwikkeld die in het bijzonder de eigenschappen van de grafische hardware in aanmerking neemt door gebruik te maken van een dubbele data hiërarchie; Eerst wordt het voxel volume in grote blokken opgehakt. De omvang van deze blokken wordt zo afgestemd dat zij optimaal corresponderen met de capaciteiten van het textuurgeheugen op de grafische kaart. Lege blokken hoeven zelfs niet naar de grafische hardware te worden verstuurd. Dan wordt er vervolgens per blok een octree structuur opgebouwd, die de zichtbare data per blok representeert. Met behulp van de octree kan dan onzichtbare data worden overgeslagen tijdens het volume rendering proces, zie figuur 3. Dat deze strategie tot een behoorlijke extra snelheidswinst kan leiden blijkt uit tabel 1.



Figuur 2: *Links: Een middels volume rendering afgebeelde dataset met grote afstanden tussen de dwarsdoorsneden. Rechts: Dezelfde dataset, maar nu met kleine afstanden tussen de dwarsdoorsneden.*

2.2 Intuïtieve visualisatie

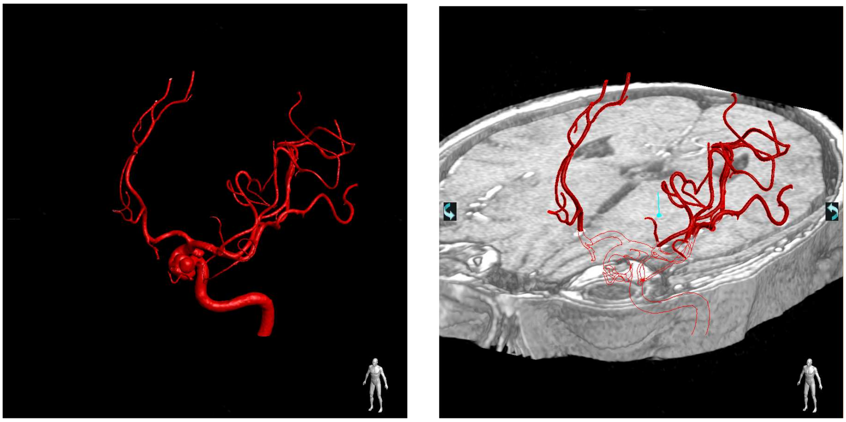
De intra-operatief geregistreerde data dient samen in een gefuseerd beeld te worden getoond. Deze gefuseerde visualisatie moet eenvoudig geïnterpreteerd kunnen worden, alle klinisch relevante details bevatten, en moet bovendien realtime gegenereerd worden. Om dit voor elkaar te krijgen is er een methode ontwikkeld die 3D vasculaire data mixt met zowel 3D data die de zachte weefsels toont, alsook 2D live fluoroscopische data. Daartoe wordt eerst de vasculaire dataset en zachte weefsel dataset geregistreerd en een mesh model van de vatenboom in de vasculaire dataset geëxtraheerd. Om de vatenboom en de zachte weefsels te mengen wordt de mesh eerst getekend en de data met de zachte weefsels daar door middel van volume rendering in gemengd. Het z-buffer zorgt ervoor dat de kleuren van de mesh op het juiste moment in de volume rendering vergelijking geweven worden. Zoals in figuur 4 te zien is, kunnen de delen van de vatenboom die verdekt worden door de zachte weefsels als silhouet getoond worden. Op die manier blijft de vorm van de hele vatenboom en de zachte weefsels zichtbaar, terwijl het ook duidelijk is waar beiden elkaar



Figuur 3: *Een fragment van een volume gerenderde afbeelding met (links) de brick blokken zichtbaar, (midden) de octrees zichtbaar, en (rechts) beide structuren zichtbaar.*

Grafische kaart	a	b	Versnelling a/b
nVidia QuadroFX 3000 AGP	25.5 fps	2.2 fps	11.6
nVidia QuadroFX 3400 PCIx	73.5 fps	9.6 fps	7.66
ATi FireGL X1, xy aligned	83.3 fps	0.23 fps	362
ATi FireGL X1, non xy aligned	27.4 fps	0.23 fps	119
ATi Radeon 9000 mobility	9.35 fps	0.26 fps	36.0
3DLabs Wildcat 7110	21.3 fps	0.38 fps	56.1

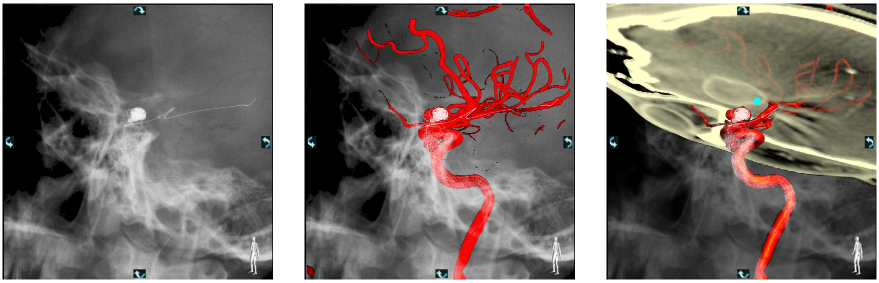
Tabel 1: *Gemiddelde beeldverversingssnelheden met (a) de optimale combinatie van brick blokken en octrees, en (b) GPU volume rendering zonder brick blokken en octrees.*



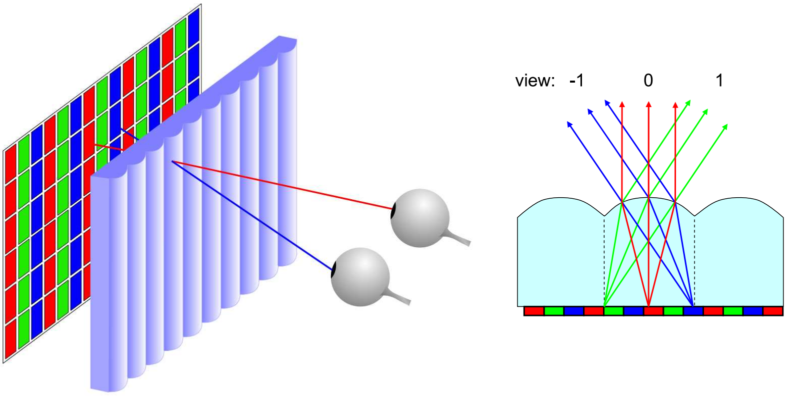
Figuur 4: *De silhouet visualisatie maakt het mogelijk om de verdekte delen van de vatenboom in relatie met de contextuele data te tonen, terwijl het beeld toch eenvoudig te bevatten blijft. Links: de cerebrale bloedvaten, gesegmenteerd in een 3DRA dataset. Rechts: de cerebrale bloedvaten gecombineerd met een volume gerenderd gedeelte van een MR dataset. Het aneurysma, dat door de MR data bedekt is, blijft dankzij het silhouet toch zichtbaar.*

raken. Het live fluoroscopisch beeld kan daar vervolgens overheen gelegd worden, en afhankelijk van het onderliggend onderwerp anders bewerkt worden, zie figuur 5.

Om de 3D vorm tijdens de klinische interventie in een enkele oogopslag behapbaar te maken is het mogelijk om de data op een autostereoscopisch scherm weer te geven. Stereoscopische perceptie staat een complete drie dimensionale indruk toe, zonder dat de klinische dataset daarbij bewogen hoeft te worden. Dit leidt tot minder interactie met de computer en daardoor tot tijdswinst tijdens de interventie. Om 3D data op een autostereoscopisch scherm te tonen moet deze vanuit meerdere kijkrichtingen (negen voor het scherm dat wij gebruiken) gevisualiseerd worden, zie figuur 6. Aangezien dit een behoorlijke impact op de visualisatiesnelheid heeft hebben we een aanpak onderzocht waarbij we de resolutie van de negen beelden dynamisch aanpassen, afhankelijk van de gevraagde snelheid en de beschikbare rekencapaciteit. Als er veel veranderingen van de beelden nodig zijn wordt de resolutie



Figuur 5: *Links: een fluoroscopie beeld. Midden: het fluoroscopie beeld gemixt met de 3DRA vatenboom. Rechts: het fluoroscopie beeld gemixt met de 3DRA vatenboom en een dwarsdoorsnede van een CT dataset.*

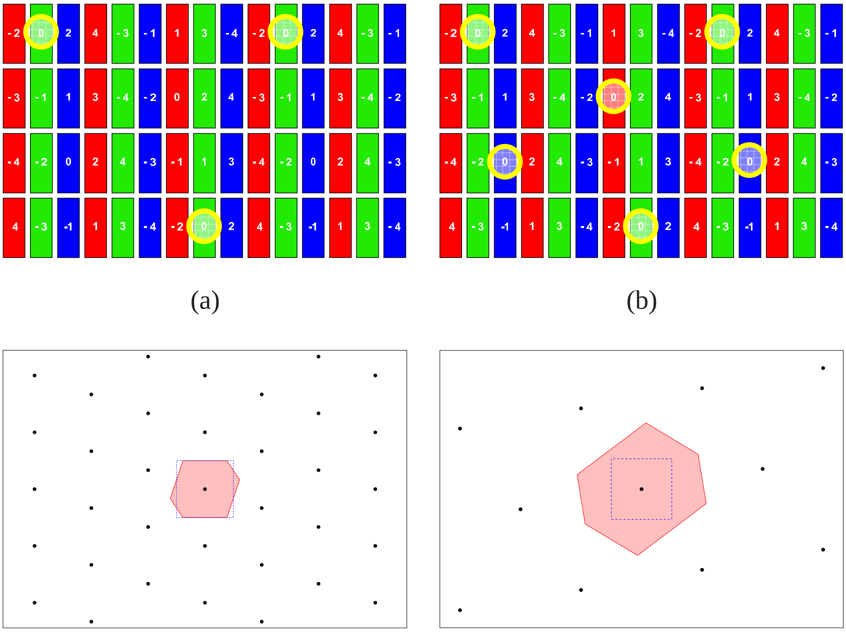


Figuur 6: *Het licht van de LCD sub-pixels wordt in verschillende richtingen afgebogen door de lenticulaire lenzen.*

omlaag geschroefd, om zo de benodigde snelheid te halen, terwijl bij trage of geen veranderingen de scene in een hoge resolutie getekend kan worden. Daarbij is de optimale resolutie afhankelijk van het pixelgrid, dat bij autostereoscopische schermen typisch geen orthogonaal grid is. De optimale resolutie van de verschillende kijkrichtingen is te bepalen door dit non-orthogonale grid in het frequentiebereik te onderzoeken, zie figuur 7.

3 Registratie

Registratie is een proces waarbij het doel is om twee beelddatasets van hetzelfde onderwerp spatieel dusdanig op elkaar af te beelden, dat dezelfde anatomie in beide beelden over elkaar heen ligt. Meestal wordt daarbij een van beide datasets spatieel gemanipuleerd (de floating dataset), terwijl de ander stil ligt (de referentie dataset). De toepassing van beeldregistratie tijdens een klinische interventie stelt grenzen aan



-2	0	2	4	-3	-1	1	3	-4	-2	0	2	4	-3	-1
-3	-1	1	3	-4	-2	0	2	4	-3	-1	1	3	-4	-2
-4	-2	0	2	4	-3	-1	1	3	-4	-2	0	2	4	-3
4	-3	-1	1	3	-4	-2	0	2	4	-3	-1	1	3	-4

(a)

-2	0	2	4	-3	-1	1	3	-4	-2	0	2	4	-3	-1
-3	-1	1	3	-4	-2	0	2	4	-3	-1	1	3	-4	-2
-4	-2	0	2	4	-3	-1	1	3	-4	-2	0	2	4	-3
4	-3	-1	1	3	-4	-2	0	2	4	-3	-1	1	3	-4

(b)

(c)

(d)

Figuur 7: *(a) Het LCD pixel grid met daarin het nummer van de kijkrichting waarin de sub-pixels worden afgebogen. De groene sub-pixels die in kijkrichting 0 (rechtdoor) worden getoond zijn omcirkeld. (b) Alle sub-pixels, onafhankelijk van hun kleur, die in kijkrichting 0 (rechtdoor) worden getoond zijn omcirkeld. (c) Het rooster van de groene sub-pixels voor kijkrichting 0 in het frequentiebereik. De Voronoi cel van het rooster is roze gemarkeerd. In blauw is de Nyquist frequentie van het orthogonale grid van de gegenereerde beelden gemarkeerd. Aangezien de Voronoi cel niet het hele Nyquist bereik omvat kan er lichte aliasing optreden in de hogere frequenties. (d) Het rooster van de sub-pixels voor kijkrichting 0, onafhankelijk van hun kleur. Aangezien het Nyquist frequentiebereik (blauw) binnen de Voronoi cel (roze) valt, zal er geen aliasing in het intensiteitsbeeld optreden.*

het algoritme en de mogelijkheid om te interacteren met de gebruiker. De voornaamste beperking is de beschikbare rekentijd. Ondanks de stormachtige ontwikkelingen van de rekenkracht van moderne computers hebben registratiealgoritmes de neiging om enkele minuten tot zelfs meerdere uren in beslag te nemen. Voor het gebruik tijdens interventionele behandelingen is dat echter niet acceptabel.

Veel registratiemethodieken hebben profijt van interactie met de gebruiker. Ze hebben input van de gebruiker nodig om de registratietaak uit te voeren, of presteren aanzienlijk beter na een grove initialisatie door de gebruiker. De mogelijkheden tijdens de klinische interventie zijn echter beperkt. De arts, die aan de behandeltafel staat, heeft vaak minder ergonomische en exacte inputapparatuur tot zijn beschikking (o.a. vanwege steriliteit). Verder is de tijd en aandacht die de arts aan de invoer voor het registratiealgoritme kan besteden beperkt. De methoden die in de context van dit proefschrift zijn ontwikkelt nemen deze beperkingen in aanmerking.

Een van de invalshoeken die zijn uitgewerkt betreft GPU acceleratie van niet-rigide registratie. Hierbij wordt het vervormingsveld van de floating dataset gevormd door kubische B-splines met controlepunten op regelmatige afstanden. De versnelling van het algoritme wordt enerzijds gehaald uit het parallellisme in de GPU, en anderzijds door de kubische B-splines op efficiënte wijze te berekenen. Het is namelijk mogelijk om een kubische B-spline uit een aantal lineaire interpolaties op te bouwen (voor 3D kunnen 64 directe samples vervangen worden door 8 lineaire interpolaties) [13]. Aangezien lineaire interpolatie op de GPU ongeveer even snel is als het direct samplen van de data levert dit een aanmerkelijk voordeel op. De similariteitsmaat E die door ons registratiealgoritme gebruikt wordt dient in de volgende vorm te kunnen worden uitgedrukt:

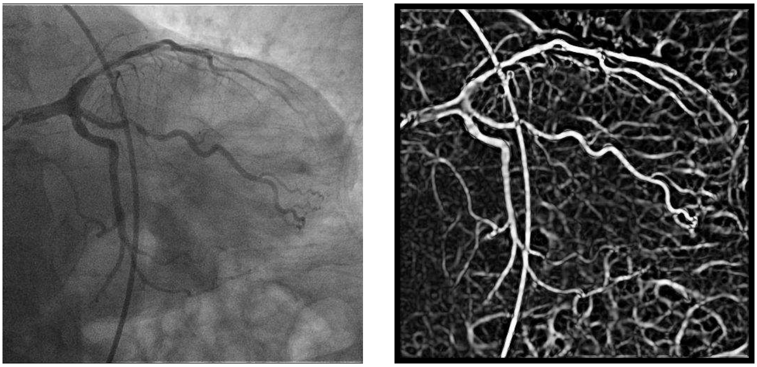
$$E = \frac{1}{\|I\|} \sum_{\vec{i} \in I} e\left(A(\vec{i}),\, B(\vec{\tau}(\vec{i}))\right) \tag{0.3}$$

Waarbij I de set van pixelposities is, e de bijdrage aan de similariteitsmaat per pixel, A het referentiebeeld, B het floating beeld, en τ het B-spline vervormingsveld. De afgeleide naar de B-spline controlepunten ziet er als volgt uit:

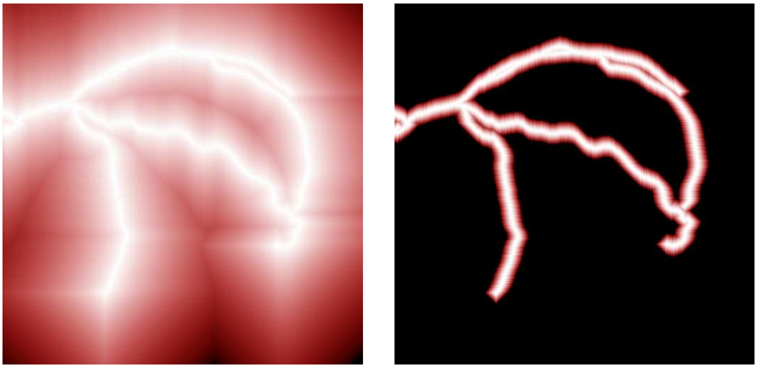
$$\frac{\delta E}{\delta c_{j,k}} = \frac{1}{\|I\|} \sum_{\vec{i} \in I} \frac{\delta e(\vec{i})}{\delta B^{\tau}(\vec{i})} \left.\frac{\delta B(\vec{x})}{\delta x_k}\right|_{\vec{x}=\vec{\tau}(\vec{i})} \frac{\delta \tau_k(\vec{i})}{\delta c_{j,k}} \tag{0.4}$$

Hierbij staat c_j voor een controlepunt en index k voor de k-de component in de vector (de as). Het kennen van de afgeleide heeft als voordeel dat een efficiëntere optimalisatiestrategie benut kan worden, zoals b.v. quasi-Newton, waardoor de registratie sneller uitgevoerd kan worden. De GPU implementatie bestaat uit twee stappen; In de eerste stap wordt het floating beeld vervormd, de contributie aan de similariteitsmaat per pixel bepaald, en het gradiënt per pixel. In de tweede stap wordt vervolgens de afgeleide per controlepunt berekend uit de informatie in de eerste stap. Deze aanpak leidt tot een snelheidswinst van ongeveer factor 50 ten opzichte van een rechttoe rechtaan CPU versie.

Een andere registratiemethode die uitgewerkt is in dit proefschrift betreft de 2D-3D registratie van vaten in fluoroscopie en CT beelden. De primaire toepassing van deze methode is de registratie van hartkransslagaders. Van deze vaten zijn namelijk geen subtractie angiografie beelden beschikbaar, en daardoor is de segmentatie van de live fluoroscopie beelden niet triviaal. In onze aanpak vermijden we dit probleem door geen expliciete segmentatie van de fluoroscopie beelden uit te voeren. In plaats daarvan passen we een vesselness filter op deze beelden toe [14], en gebruiken we het resultaat daarvan direct in de voorgestelde similariteitsmaat, zie figuur 8. Voorafgaand aan de registratie worden de bloedvaten gesegmenteerd in de CT dataset. De gesegmenteerde vaten worden voor elke nieuwe spatiële transformatie geprojecteerd op het vlak van de fluoroscopische beelden en daar wordt vervolgens een afstandstransformatie op toegepast, zie figuur 9. De similariteitsmaat is het in-product van het vesselness en het afstandstransformatie beeld. Onze proeven met gesimuleerde en klinische data laten zien dat deze aanpak beter werkt dan de iterative closest point (ICP) [15] methode.



Figuur 8: *Links: Röntgenbeeld van de hartkransslagaders. Rechts: Vesselness transformatie van hetzelfde Röntgenbeeld.*

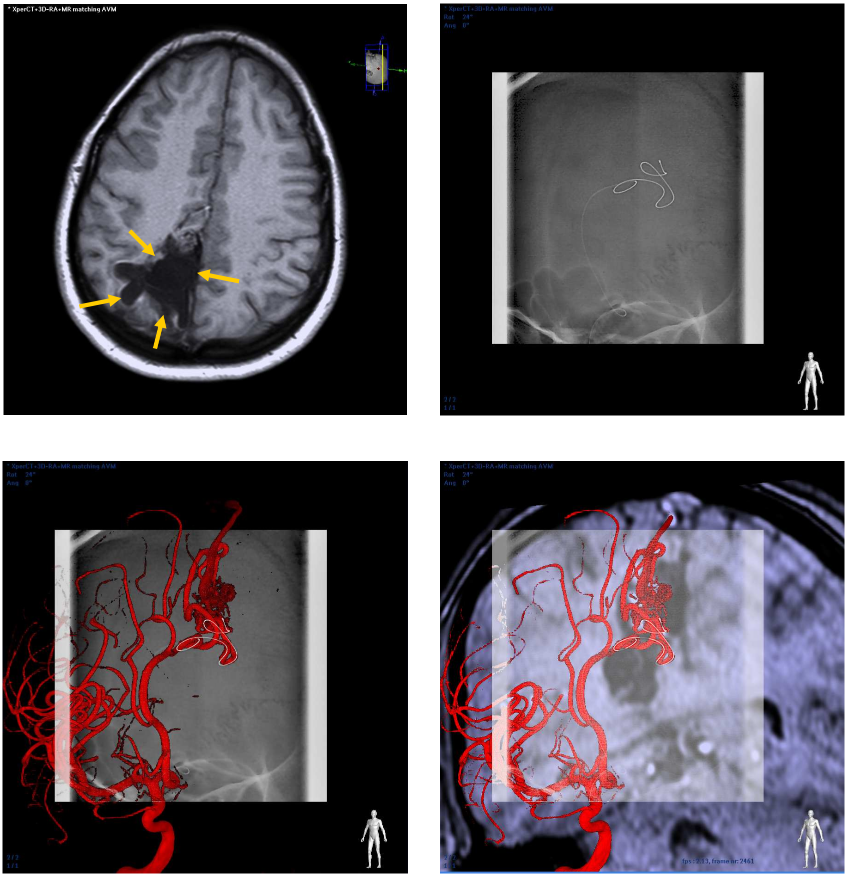


Figuur 9: *Links: Afstandstransformatie van de geprojecteerde hartkransslagaders uit de 3D CT dataset. Rechts: Hetzelfde beeld gekwadrateerd.*

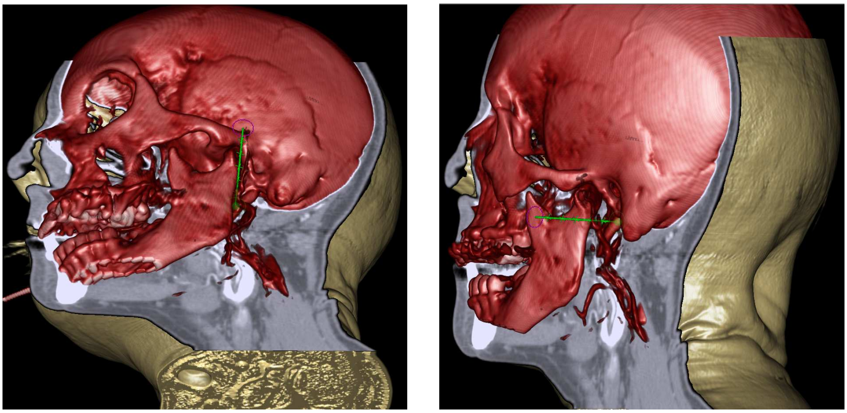
4 Conclusies

De hier beschreven technische oplossingen zijn gezamenlijk geëvalueerd in een drietal concrete klinische toepassingen: 1) Het navigeren van de katheter op basis van meerdere beeldinformatiebronnen tijdens de behandeling van arterio-veneuze malformatie (AVM) in de hersenen, zie figuur 10. 2) Het plannen en navigeren van een punctienaald op basis van diagnostische CT beelden en tegelijkertijd live fluoroscopische beelden, zie figuur 11. 3) Het optimaal plaatsen van een stent in een vernauwing in een kransslagader, eveneens op basis van navigeren en plannen met behulp van diagnostische CT beelden gemixt met live angiografische beelden, zie figuur 12.

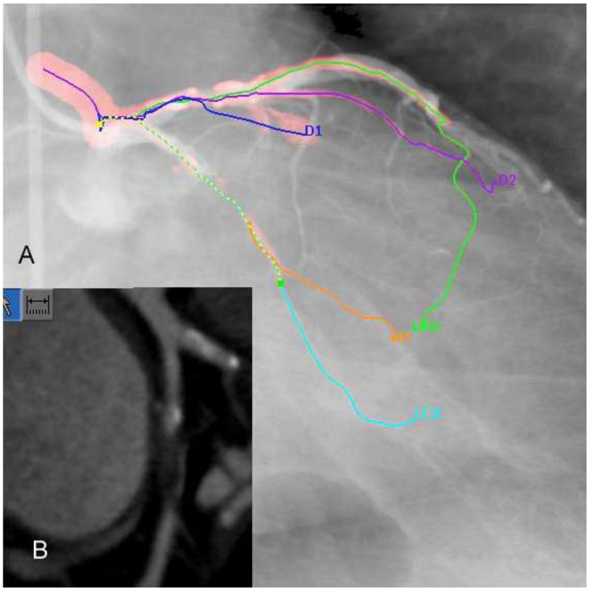
De algoritmes zijn hiertoe in klinische prototypes geïmplementeerd, die door het ziekenhuispersoneel zelfstandig bediend konden worden. De evaluatie heeft plaats



Figuur 10: *Linksboven: Een MR beeld toont een arterioveneuze malformatie (AVM) en het getroffen hersenweefsel (gele pijlen). Rechtsboven: Het live fluoroscopiebeeld zonder contrastmiddel laat de voerdraad zien, maar niet de relatie met de vatenboom en zachte weefsels. Linksbeneden: Het fluoroscopiebeeld gemixt met de 3DRA vatenboom voegt de vasculaire context toe aan de live data. Rechtsbeneden: Het fluoroscopiebeeld, de 3DRA vatenboom en een dwarsdoorsnede van de MR data. De MR dwarsdoorsnede staat altijd parallel haaks op de kijkrichting, en is gepositioneerd op het voerdraad uiteinde.*



Figuur 11: *De pre-operatieve CT data (geel) en de intra-operatieve C-arm cone-beam CT data (rood) worden samen met het geplande pad (groen) getoond. Links: schuin aanzicht van links. Rechts: Posterieur schuin aanzicht.*



Figuur 12: *(a) Gefuseerd beeld van cardiac CT data (rood) en live Röntgenbeelden (grijs) voor de navigatie in een chronisch totaal-geoccludeerd (CTO) vat. Het door middel van de katheter geïnjecteerd contrastmiddel (wit) penetreert de circumflex (LCX) niet, terwijl het traject van het vat wel zichtbaar blijft via de CT overlay. (b) De corresponderende gekromde MPR, laat de CTO en retrograde vulling van het vat zien.*

gehad in een achttal ziekenhuizen in Europa, Noord-Amerika en Azië. Tijdens vijf internationale medische conferenties zijn er live puncties en angiografische behandelingen uitgezonden vanuit de interventiekamer waarbij de genoemde prototypes werden gebruikt. De beschreven technologieën zijn vervolgens geïntegreerd in commercieel beschikbare producten (Philips Allura 3D-RA, met meer dan 500 exemplaren verkocht, en Philips Allura XperGuide), die in ziekenhuizen over de hele wereld geïnstalleerd zijn.

In alle klinische toepassingen werden meerdere beelddatabronnen tijdens de interventie gecombineerd in een coherent samenhangend gefuseerd beeld dat de klinisch relevante gegevens op een behapbare wijze presenteert. De voorgestelde technische oplossingen dragen hier in hoge mate aan bij, zowel betreffende de intuïtieve visualisatie alsook de beperkte rekentijd van de algoritmes. Beiden zijn zeer van belang om deze technieken routinematig in te zetten tijdens klinische behandelingen.

List of Acronyms

1D	one dimensional
2D	two dimensional
3D	three dimensional
3DRA	three dimensional rotational angiography
4D	four dimensional
AAA	abdominal aortic aneurysm
AVM	arteriovenous malformation
CABG	coronary artery bypass grafting
CAD	coronary artery disease
CC	cross-correlation
CPU	central processing unit
CT	computed tomography
CTA	computed tomography angiography
CTO	chronic total occlusion
CUDA	compute unified device architecture
DICOM	digital imaging and communications in medicine
DQE	detective quantum efficiency
DRR	digitally reconstructed radiograph
DSA	digital subtraction angiography
DT	distance transform
EVAR	endovascular aneurysm repair
GB	gigabyte
GPS	global positioning system
GPU	graphics processing unit
FD	flat detector
HU	Hounsfield units
ICP	iterative closest point
IGIT	image guided interventions and therapy
II	image intensifier
LAD	left anterior descending artery
LCD	liquid-crystal display

MB	megabyte
MI	mutual information
MPR	multi-planar reformat
MR	magnetic resonance
OR	operating room
PC	personal computer
PCI	percutaneous coronary intervention
RAM	random access memory
RCA	right coronary artery
RGB	red green blue
RMS	root mean square
SIMD	single instruction, multiple data
SSD	sum of squared differences
SSE	streaming SIMD extensions
SVG	saphenous vein graft
TV	television
VRML	virtual reality modeling language

Contents

Acknowledgements **i**

Abstract **iii**

Nederlandse samenvatting **v**

1 Achtergrond v
2 Visualisatie vi
2.1 Snelle volume rendering vi
2.2 Intuïtieve visualisatie viii
3 Registratie x
4 Conclusies xiii

List of Acronyms **xvii**

1 Introduction **1**

1.1 Background 1
1.2 Objectives of the thesis 1
1.3 Overview of the thesis 2
1.4 Major contributions 3

2 Interventional X-ray **5**

2.1 Angiographic X-ray 5
2.2 First in-vivo catheterizations 5
2.3 The X-ray C-arm 7
2.4 3D reconstruction 9
2.4.1 X-ray attenuation 9
2.4.2 Filtered back-projection 10
2.4.3 Cone-beam reconstruction 11

Part I: Fused Visualization **15**

3 Fast Volume Rendering **19**

3.1 Introduction 19

3.2 Related work . . . 19
3.3 Volume Rendering . . . 21
3.4 Interpolation . . . 23
3.5 Pre- versus post-lookup . . . 24
3.6 Bottlenecks . . . 26
3.7 Method . . . 28
3.8 Bricking . . . 29
3.9 Early ray termination . . . 30
3.10 Octree . . . 31
3.11 Results . . . 32
3.12 Conclusions . . . 35

4 Fusion of Vascular, Soft-tissue and X-ray data 37
4.1 Introduction . . . 37
4.2 Method . . . 38
    4.2.1 Volume and mesh blending . . . 38
    4.2.2 Silhouette overlaying . . . 39
    4.2.3 Blending with the 2D X-ray image . . . 41
4.3 Results and discussion . . . 42

5 Autostereoscopic visualization 45
5.1 Introduction . . . 45
5.2 State of the art . . . 46
5.3 The multiview lenticular display . . . 47
5.4 The different angular views . . . 48
5.5 Resolution considerations . . . 49
5.6 Dynamic resolution . . . 52
5.7 Results . . . 54
5.8 Clinical setup . . . 55
5.9 Conclusions . . . 57

Part II: Intra-interventional Registration 61

6 Registration algorithms 65
6.1 Introduction . . . 65
6.2 Spatial transformation . . . 66
    6.2.1 Affine transformations . . . 66
    6.2.2 Non-affine transformations . . . 66
6.3 Similarity measure . . . 67
6.4 Optimization . . . 69
6.5 Validation . . . 70
6.6 Machine-based 2D-3D Registration . . . 71
    6.6.1 Introduction . . . 71
    6.6.2 Calibration . . . 71
    6.6.3 Projection . . . 73

7 GPU-acceleration in Elastic Image Registration 75
- 7.1 Introduction . . . 75
- 7.2 Uniform B-spline interpolation . . . 76
  - 7.2.1 Cubic B-spline interpolation . . . 77
  - 7.2.2 GPU-accelerated cubic B-spline evaluation . . . 78
  - 7.2.3 Accuracy and performance . . . 81
- 7.3 GPU-accelerated Elastic Registration . . . 82
  - 7.3.1 Similarity measure . . . 82
  - 7.3.2 Deformation field . . . 83
  - 7.3.3 Derivatives . . . 84
- 7.4 GPU implementation . . . 85
  - 7.4.1 Similarity measure & derivatives . . . 85
  - 7.4.2 Results . . . 86
  - 7.4.3 Discussion . . . 88
- 7.5 Conclusions . . . 89

8 Vesselness-based 2D-3D Registration 91
- 8.1 Introduction . . . 91
- 8.2 Related work . . . 91
- 8.3 Method . . . 92
  - 8.3.1 Spatial mapping . . . 92
  - 8.3.2 Vesselness filter . . . 93
  - 8.3.3 Similarity measure . . . 94
  - 8.3.4 Optimization strategy . . . 96
- 8.4 Results . . . 96
- 8.5 Discussion . . . 99

Part III: Clinical Applications 101

9 Real-time 3D Multimodality Fusion in Neuroangiography 103
- 9.1 Introduction . . . 103
- 9.2 Method . . . 104
  - 9.2.1 Pre-processing . . . 104
- 9.3 Clinical use . . . 105
- 9.4 Results . . . 106
  - 9.4.1 Robustness . . . 106
  - 9.4.2 Computation time . . . 107
- 9.5 Discussion . . . 109
- 9.6 Conclusions . . . 111

10 Multimodal Registration in Needle Guidance 113
- 10.1 Introduction . . . 113
- 10.2 Methods and materials . . . 114
  - 10.2.1 Procedural technique . . . 114
  - 10.2.2 Patients and materials . . . 117

10.3 Results 118
10.4 Discussion 118
10.5 Conclusions 120

11 Multimodal Registration for Coronary Artery Disease Interventions 121
11.1 Introduction 121
11.2 Methods and materials 123
11.2.1 Pre-interventional acquisition and analysis 123
11.2.2 Peri-interventional use 125
11.3 Results 126
11.3.1 Case report 128
11.4 Discussion 129
11.5 Conclusions 132

12 Conclusions 133
12.1 Outlook and future work 136

Bibliography 137

List of publications 155

Curriculum Vitae 161

A A Coordinate Space Framework 163
A.1 Introduction 163
A.2 Related work 164
A.3 Design basics 165
A.4 Transforming space 166
A.5 Querying geometry objects 167
A.6 Traversing the scene graph 167
A.7 Operator overloading 168
A.8 Real life examples 169
A.8.1 Mouse click on a voxel volume 169
A.8.2 Defining points of interest 169
A.8.3 Gantry-tilt CT volumes 170
A.8.4 Follow camera orientation 171
A.9 Conclusions 171

Chapter 1

Introduction

1.1 Background

Advances in medical scanning technology provide a wide spectrum of valuable and complementary information about a patient's pathology, anatomy, and physiology. The signals that are produced by these scanners differ in dimensionality, scale, extent, and biological origin. The technological and clinical advances have brought a tremendous growth in the use of radiological images during the last decades. The optimal exploitation of this wealth of information in the clinical treatment is a difficult task. Especially the combination of the information produced by the different scanning techniques may be very useful, since the complementary information may lead to a better insight, but also poses significant technical hurdles.

Image guided interventions and therapy (IGIT) are becoming increasingly popular in todays healthcare system. The minimally invasive nature of these procedures is often preferred over open surgeries because less trauma to the patient's body is caused, which generally is associated with easier and faster recovery. Interventional radiologists and surgeons are also becoming more experienced and comfortable in performing these procedures. The minimally invasive interventional clinicians use instruments such as needles and catheters to perform the diagnostic and therapeutic procedures, which are guided by imaging equipment.

1.2 Objectives of the thesis

It is the topic of this thesis to combine multiple sources of image data into a coherent presentation for usage during minimally invasive treatment, assuring usable and cognitively adequate interaction by the interventionalist.

During minimally invasive procedures the clinician guides the treatment based on the real-time intervention image flow. Often there are already diagnostic images available, prior to the intervention, frequently also used for treatment planning. The integration of these pre-interventional data sources with the intra-interventional images can lead to an improved information basis during the clinical procedure.

The pre-interventional data can provide complementary pathological, anatomical, and/or physiological data. Furthermore, the data fusion allows to project the preoperative treatment planning on the real-time image data, which can provide a valuable roadmap to guide the procedure.

There are several constraints to the usage of several sources of image data. First of all, there are usually strict computation time restrictions to calculations that are being executed during the clinical procedure, since the patient is prepared for the interventional treatment, and an expensive clinical team and equipment are waiting. Since many algorithms can only start to work after data has been acquired during the course of the intervention, it is inevitable that those programs will occupy some procedure time. Furthermore, it is of greatest importance that the interaction with the computer is easy, and imposes the smallest possible cognitive strain on the physician. The clinician has to focus primarily on the treatment itself, and is often overloaded with many stimuli, originating from a multitude of devices, interventional staff, and pathological anatomy. Therefore, great care has to be given to the user interaction and visualization of the fused data.

It is the objective of this thesis to present practical technical solutions to the topic of peri-interventional image fusion. It should be possible to utilize these solutions in the clinical practice, without loss of general applicability. Furthermore, it is the goal to demonstrate the value of the technical solutions by employing them in a number of clinical applications.

1.3 Overview of the thesis

This thesis is presented in three main parts; The visualization techniques that were used to accomplish efficient and easy to interpret images during the clinical intervention are presented in part I. The registration methods that serve to obtain the integration of the multi-modal data during the intervention are described in part II. Finally, the clinical experience with the assembly of those techniques can be found in part III. However, before we dive into the first part, the background of those techniques and clinical applications is sketched. Therefore, the thesis starts with an overview of the X-ray imaging techniques that are encountered in a catheterization laboratory (cathlab) and their historical context in chapter 2.

Then part I begins with the topic of accelerating volume rendering by using the vast processing power of the Graphics Processing Unit (GPU) in chapter 3. The examination of the various bottlenecks that are encountered within the GPU, has led to an optimized rendering approach, using a double space skipping hierarchy. The following chapter 4 extends the volume rendering method to deal with multiple three-dimensional and two-dimensional datasets. This enables the fused representation of multi-modal data, which is used in part III. Special attention has been paid to maintain interactive frame rates, which is of utmost importance for visualization of image data that is being used for interventional guidance. This part concludes with chapter 5, which describes the application of the earlier introduced rendering techniques to autostereoscopic displays. Such displays allow a viewer to perceive depth without the aid of external glasses. The challenge posed by such displays is the fact that they need the same scene rendered from a number of viewing positions, and therefore impose a high load on the rendering system.

Part II first provides a general context to image-based registration algorithms in chapter 6. This chapter also briefly describes registration based on the mechanical X-ray geometry parameters, which is always real-time and does not depend on landmarks being present in the image. In order to achieve fast elastic registrations, a prerequisite in order to use them interventionally, chapter 7 explores how the GPU can be used to accelerate this task. This chapter especially focusses on the precision, the implementation and the performance aspects. Chapter 8 presents a new similarity measure that especially has been developed for the registration of two- and three-dimensional datasets containing the coronary arteries. It also provides the validation that has been conducted on this method.

The clinical experience that was gathered with these techniques is presented in part III. Chapter 9 describes the clinical aspects of using multi-modality registration and fused visualization in the roadmapping of intravascular devices for neuroangiography. The focus lies on the treatment of arteriovenous malformations. Chapter 10 presents the approach that was developed to plan and guide percutaneous needle insertions. This chapter especially describes the application of this technique within the embolization of paragangliomas (glomus tumors). The clinical application of image fusion in the treatment of coronary artery disease is the topic of chapter 11. Finally, chapter 12 concludes this thesis with a summary and discussion of the obtained results.

A design pattern, which was developed to manage many coordinates systems in large software packages, is described in appendix A. This design pattern was used in all the clinical applications that are presented in part III, and aided considerably in dealing with a vast number of coordinate spaces in a flexible and transparent manner.

Since the technical scope of this thesis is rather wide (containing both visualization and registration aspects), there is not a separate chapter dealing with the state of the art. The related work is described instead in the respective chapters throughout the thesis.

1.4 Major contributions

The major contributions of this thesis are:

- The introduction of a double space-skipping hierarchy to GPU-accelerated volume rendering, employing bricks and octrees. Since this solution is tailored to the bottlenecks found in the GPU, it helps to reach the maximum performance, especially when volume rendering large datasets that even might exceed the memory available to the GPU.

- An analysis of the optimal resolution for rendering to autostereoscopic displays. This analysis is then used in dynamically balancing the resolution to achieve a balance between maximum detail and reasonable performance.

- The in-depth exploration of the precision aspects of GPU-accelerated B-spline evaluation. Especially when the GPU is applied for general purpose image processing tasks in a clinical context, it is of highest importance to know its precision and numerical behavior.

- The application of GPU acceleration to elastic image registration. In order to apply image registration during a clinical intervention, it is essential that its results are available within a limited time frame. The GPU acceleration helps in achieving this goal.
- A novel similarity measure has been developed to register two- and three-dimensional vascular data. This method was deemed necessary since existing methods did not perform well enough for the task of registering the coronary arteries. The new approach was found to improve on this task, as has been demonstrated by the validation results.
- The evaluation of aforementioned techniques in the clinical practice. The methods described in this thesis have been applied in various clinical interventions. The results have been reported in the medical literature and as such have become part of the state of the art.
- The development of a coordinate space management framework. This framework allows to administrate many dynamically linked coordinate spaces. Furthermore, it removes the explicit conversion from the programming code, and therefore is less prone to programming errors.

The idea of the coordinate space framework was born during a discussion with Jeroen Terwisscha van Scheltinga. Obviously, many people have been involved in the clinical evaluation of the techniques. All other mentioned points are the sole work of the author of this thesis.

Chapter 2

Interventional X-ray

2.1 Angiographic X-ray

Throughout 1895 Wilhelm Conrad Röntgen (1845-1923) was examining the external effects of various vacuum tubes. During an experiment on November 8, 1895, whereby a cardboard was blocking all visible light, he noticed a fluorescent effect on a nearby cardboard, which was painted with barium platinocyanide. This motivated Röntgen to conduct a series of experiments, from which he speculated that the fluorescent effect was caused by a new type of radiation, which he temporarily named ’X-rays’ [16]. It would be unjustified to attribute the discovery of X-radiation merely to coincidence. Röntgen had planned to use the barium platinocyanide painted cardboard in a next series of experiments, and it would have been likely that he would have discovered X-rays in those trials. In 1901 Röntgen was awarded the first Nobel prize in physics for his discovery.

Already in December of 1895 Röntgen produced the first human radiograph, by imaging his wive’s hand, see figure 2.1a. Within two months of Röntgen’s discovery, Haschek and Lindenthal managed to demonstrate the blood vessels in a cadaver hand by injecting a suspension of chalk and cinnabar (mercury sulfide) in oil [17], see figure 2.1b. A comprehensive overview of the consequent steps leading to modern percutaneous coronary angiography is given by Meier [18].

2.2 First in-vivo catheterizations

In 1929 Werner Forssmann performed the first heart catheterization in a living human being; himself [19, 20]. With the help of a somewhat reluctant accomplice he anesthetized his own elbow and performed a cut-down on his left arm and inserted a well-oiled ureteral catheter via the left antecubital vein. His aim was to insert the tube for a pre-measured distance in order to reach the right ventricle. After walking down to the X-ray department in the basement with the tube dangling from his arm, he continued the procedure under fluoroscopy, with the aid of a mirror held by his accomplice. However, because of the length of the catheter he was only able to

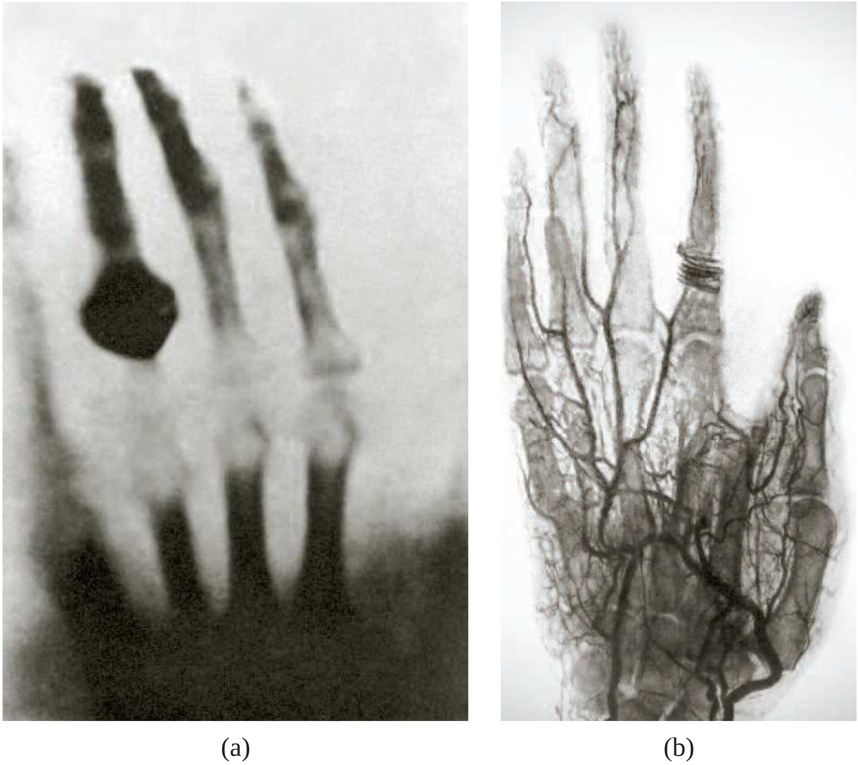

(a) (b)

Figure 2.1: *(a) Roentgen's first human radiograph, made in 1895. (b) First angiographic recording, taken from a cadaver by Haschek and Lindenthal in 1896.*

reach the right atrium. He then made some radiographs as documentary evidence, see figure 2.2. Forssmanns superior, the surgeon Professor Sauerbruch, was not amused by Forssmanns experiments. His response was “Mit solchen Kunststücken habilitiert man sich in einem Zirkus und nicht an einer anständigen Deutschen Klinik” (Tricks like that qualify you for a circus and not for a leading German clinic). In the ensuing row, Forssmann lost his job, but he shared the Nobel Prize for Medicine with Cournand and Richards in 1956 for his ground breaking work in heart catheterization [18].

Charles Theodore Dotter can be credited for pioneering the field of interventional radiology. He was the first to describe flow-directed balloon catheterization, the double-lumen balloon catheter, the safety guidewire, and the “J” tipped guidewire. Percutaneous transluminal angioplasty was his landmark contribution [21]. Dotter's nonconservative ways (*e.g.*, performing balloon angioplasty on patients that were referred explicitly for diagnosis only) hindered the acceptance of percutaneous minimally invasive interventions, and the medical society remained reluctant to accept the ideas of Dotter, until Andreas Grüntzig, a Zurich cardiologist, published the first percutaneous transluminal coronary angioplasty in 1977 [22]. Grüntzig invented the miniaturized balloon-tipped catheter and developed the technique of coronary angio-

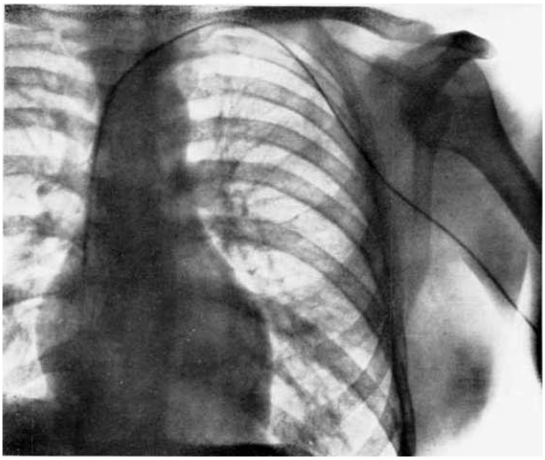


Figure 2.2: *The first in-vivo hearth catheterization, performed by Forssmann on himself in 1929.*

plasty, based on the work of Dotter. Restenosis and the need for repeat interventions, however, remained a severe limitation. This led to the development of a metallic intravascular scaffold, known as a "stent", which is permanently placed at the stenotic location [23] (see *e.g.*, figure 2.3). Charles Dotter already introduced the concepts of percutaneous arterial stenting and stent grafting by placing the first percutaneous "coilspring graft" in the femoral artery of a dog. The first human implantation of coronary stents was performed by Ulrich Sigwart and Jacques Puel in 1986 [24].

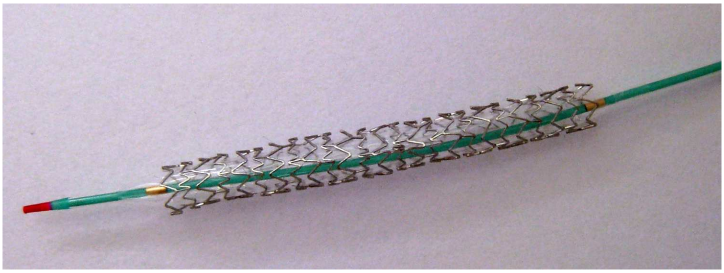


Figure 2.3: *Example of a modern cardiac stent (Boston Taxus Express* 2.75×28 *mm) and guide wire.*

2.3 The X-ray C-arm

Before the introduction of the C-arm system, the only possibility to perform live X-ray acquisitions during surgery was the fluoroscope. This was a hand-held device with a fluorescent screen, virtually unchanged since 1896, with a very poor light intensity. To compensate high X-ray doses were used, leaving the surgeon poorly protected in

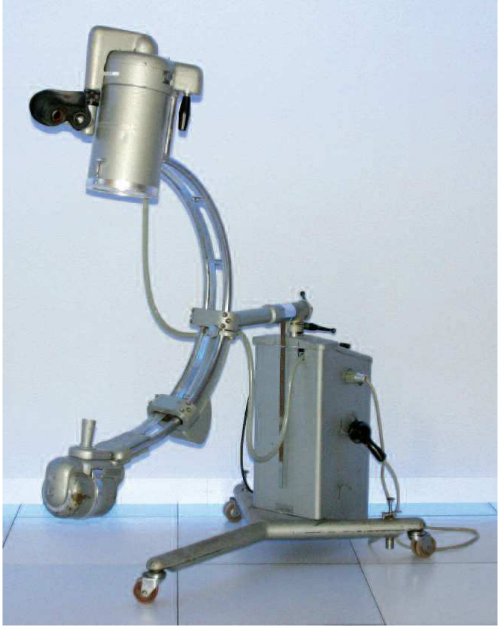


Figure 2.4: *The first X-ray C-arm system: the Philips BV20.*

the X-ray beam. More detailed studies required radiography on film to be developed during the surgical procedure.

The first C-arm was developed together with the image intensifier (II) by the German Philips Medical Systems organization in the early 1950s [25]. After a more robust system was developed, mainly by Jacques Hoogeveen of the Philips Medical Systems factory in Eindhoven, it was commercially released in 1955 under the name BV20 (“Bildverstärker” or “Beeldversterker” with a 20 mA X-ray tube), see figure 2.4. The bow could rotate in a propeller-fashion movement and slide through a sleeve, providing a large degree of freedom in selecting a projection angle. This mechanical approach to reaching rotational freedom remains unchanged until present date. The II of the BV20 projected the intensified X-ray image directly on a pair of goggles, that could be viewed by one person, often in an awkward position. In 1958 the BV20 was equipped with an industrial TV chain instead of the goggles, which made the X-ray image more accessible. In this way the whole surgical team could follow the X-ray image, leading to better informed and faster operations. This II-TV C-arm can be considered the first modality that enabled image guided interventional treatment on a routinely basis.

After these two revolutionizing innovations a period of evolution followed. Fixed mounted larger C-arms next to the mobile ones, brought more mechanical stability and reproducibility. The components of the imaging chain were gradually improved, and the arrival of digital image processing enabled digital image enhancement and archiving. However, the essential design of the C-arm remained unchanged until the arrival of the flat detector (FD), which replaced the II. Solid-state digital radiogra-

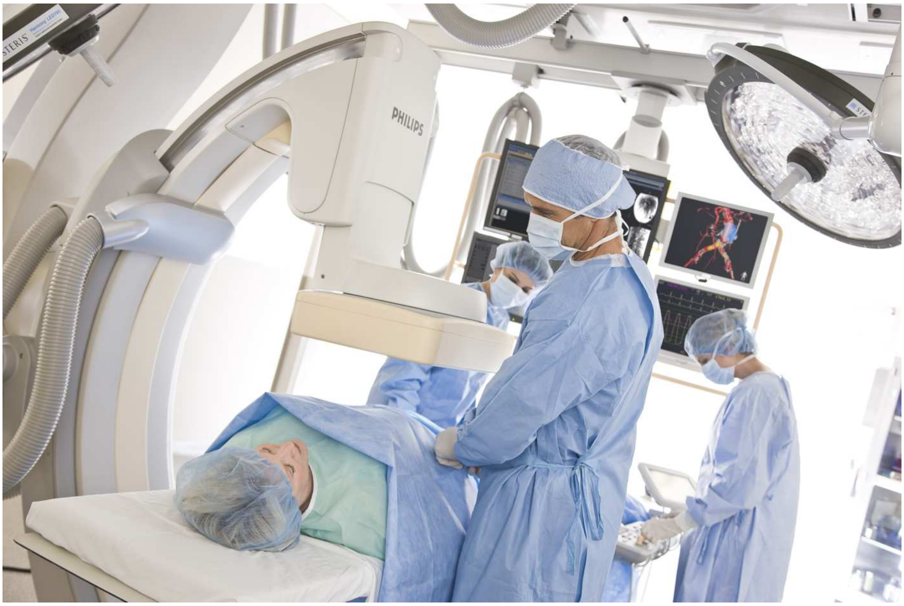


Figure 2.5: *An intervention being performed, using a modern flat detector C-arm system.*

phy detectors, commonly known as flat detectors, emerged just before the turn of the millennium (figure 2.5). This new generation of digital image detectors contains a thin layer X-ray absorptive material combined with an electronic active matrix array. Principally two types exist; the indirect conversion (X-ray scintillator-based) and direct conversion (X-ray photoconductor-based) types [26]. The production of the flat detector bears a lot of resemblances with the production of micro-chips and LCD displays, and was facilitated by the development of cost-effective production of large LCD displays. Studies have shown that the FD performs superior in terms of X-ray dose efficiency and image quality expressed as Detective Quantum Efficiency (DQE) [27–30]. Furthermore it lacks the pincushion image deformation and sensibilities to external magnetic fields that were characteristic for the II systems. During clinical interventional treatment also the smaller form factor of the detector is an advantage in the already crowded intervention room. The minimally invasive clinical applications that are presented in this work in chapters 9, 10 and 11 are based on the live image guidance using this type of fixed mounted flat detector C-arm equipment.

2.4 3D reconstruction

2.4.1 X-ray attenuation

Suppose we have a thin piece of uniform radiopaque material of thickness dx. From an X-ray beam of N photons, N_{abs} photons are absorbed by the material. The physical effects leading to the photon absorption are explained in detail in *e.g.*, [31]. The

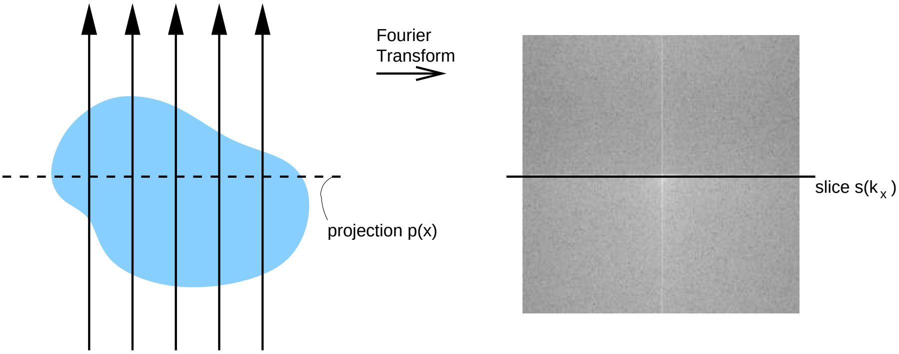


Figure 2.6: *The projection slice theorem.*

amount of absorbed photons is linearly proportional to the amount of entering photons, the thickness dx and the radiopacity of the material, which is expressed by its *linear attenuation coefficient* μ.

$$dN = -N_{abs} = -N \cdot \mu \cdot dx \tag{2.1}$$

When refraction and emission effects are disregarded, the situation above can be easily extended to an X-ray beam traveling through non-uniform radiopaque material by integrating the equation over x. The resulting formula is called the *Beer-Lambert* equation.

$$N = N_0 \cdot e^{-\int\limits_0^x \mu(x')\,dx'} \tag{2.2}$$

2.4.2 Filtered back-projection

The initial computed tomography (CT) scanner, as introduced by Godfrey Hounsfield in the early 1970s, was based on the two dimensional projection-slice theorem. This theorem states that the one-dimensional Fourier transform of the parallel projection of a function $f(x, y)$ is equal to a slice of the two-dimensional Fourier transform of function $f(x, y)$, whereby the direction of the slice is perpendicular to the direction of the projection. *E.g.*, when we project along the y-axis (see figure 2.6), the projection can be written as:

$$p(x) = \int\limits_{-\infty}^{\infty} f(x, y)\,dy \tag{2.3}$$

The Fourier transform of $f(x, y)$ is

$$F(k_x, k_y) = \int\limits_{-\infty}^{\infty}\int\limits_{-\infty}^{\infty} f(x, y)\,e^{-2\pi i(xk_x + yk_y)}\,dx\,dy \tag{2.4}$$

The slice $s(k_x)$ is then

$$\begin{aligned} s(k_x) = F(k_x, 0) &= \int\limits_{-\infty}^{\infty} \int\limits_{-\infty}^{\infty} f(x,y)\, e^{-2\pi i (x k_x)}\, dx\, dy \\ &= \int\limits_{-\infty}^{\infty} \left[\int\limits_{-\infty}^{\infty} f(x,y)\, dy \right] e^{-2\pi i (x k_x)}\, dx \\ &= \int\limits_{-\infty}^{\infty} p(x)\, e^{-2\pi i (x k_x)}\, dx \end{aligned} \tag{2.5}$$

and that is just the Fourier transform of $p(x)$. The projection-slice theorem can also easily be extended to higher dimensions.

The set of projections along straight lines is known as the Radon transform:

$$\begin{aligned} \mathcal{R}[f](\alpha, s) &= \int\limits_{-\infty}^{\infty} f(x(t), y(t))\, dt \\ &= \int\limits_{-\infty}^{\infty} f\left(t \cdot (\sin\alpha, -\cos\alpha) + s \cdot (\cos\alpha, \sin\alpha)\right)\, dt \end{aligned} \tag{2.6}$$

A computationally efficient inversion algorithm for the two-dimensional Radon transform is the so-called filtered back-projection, introduced by Feldkamp, Davis and Kress [32]. This algorithm takes the projections $\mathcal{R}[f](\alpha, s)$ as input, applies a ramp filter to them, and ‘smears’ the filtered projections back over their lines to produce an image. According to this algorithm, the reconstructed value f at any given location $\vec{x}$ can be expressed as

$$f(\vec{x}) = \frac{1}{4\pi} \int_0^{2\pi} p^*(\alpha, (\cos\alpha, \sin\alpha) \cdot \vec{x})\, d\alpha \tag{2.7}$$

whereby p^* denotes the ramp filtered projections.

2.4.3 Cone-beam reconstruction

In order to produce a complete and exact reconstruction from cone-beam projections, the trajectory of the C-arm has to satisfy the condition that its orbit intersects with every plane through the reconstruction space [33, 34]. These trajectories, however, are not very practical in a clinical setting. Therefore approximating three dimensional reconstruction techniques have been developed, using semi-circular trajectories of at least 180° [35], see figure 2.7. In order to compensate for the variations in the angular speed of the C-arm, every projection p^* in equation 2.7 is pre-multiplied by a weighting factor f. This factor is obtained for a projection i by taking the angle α

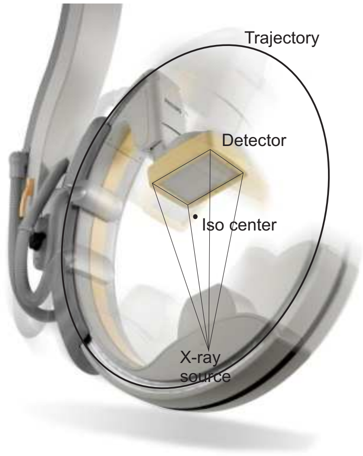


Figure 2.7: *Circular C-arm trajectory for cone beam reconstruction.*

between the normal $\vec{n}$ of the previous projection $i-1$ and the following one $i+1$, and dividing it by the overall angular range of the trajectory [36].

$$\alpha_i = \angle(\vec{n}_{i-1}, \vec{n}_{i+1})$$

$$f_i = \frac{\alpha_i}{\frac{1}{N}\sum\limits_{i=1}^{N} \alpha_i} \tag{2.8}$$

Since the late 1990s these techniques have been applied to commercial II-based C-arm systems [36–38]. Due to the geometrical distortions and the image signal response properties of the II along with the limited sampling of the rotational trajectory the 3D reconstruction was mainly limited to objects with high contrast in radiopacity, see figure 2.8. In case of 3D imaging of the vasculature the high contrast cone beam reconstructions are commonly called ‘three-dimensional rotational angiography (3D-RA)’ and for reconstructions of non-vascular structures the term ‘three-dimensional rotational X-ray (3D-RX)’ is often used.

The image intensifier suffers from a pincushion like deformation of the image, and is also sensitive for influences from external magnetic fields (such as the earth magnetic field). With the introduction of the flat detector these geometric deformations have been resolved. The ability to image also low contrast structures has significant clinical benefits. Bone tissues possess X-ray attenuation values up to 2000 Hounsfield units (HU), and iodine contrast medium can even reach 3000 HU. The attenuation value for air is -1000 HU, for fat around -50 HU, for water 0 HU and for soft-tissue around 40 HU. A fresh bleeding in the brain lies in the range from 10 to 60 HU, which means that a considerable improvement in contrast resolution is needed to provide a meaningful image in this range. The imaging of low contrast structures through cone beam reconstruction on flat detector C-arm systems has become available amongst others by the following improvements [39, 40]:

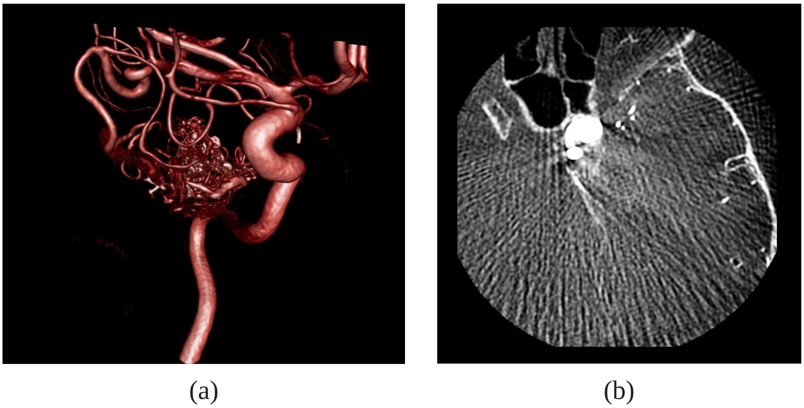

(a) (b)

Figure 2.8: *(a) 3D cone beam reconstruction of a brain Arteriovenous Malformations (AVM). The blood vessels can be made visible by the intra-vascular injection of iodine contrast medium. (b) A slice from a cone beam reconstruction, using an II based C-arm system. The high contrast structures, such as the skull and the vasculature, filled with iodine contrast medium, are well visible. The structures with low radiopacity, such as the soft-tissue structures and air, contain a lot of noise.*

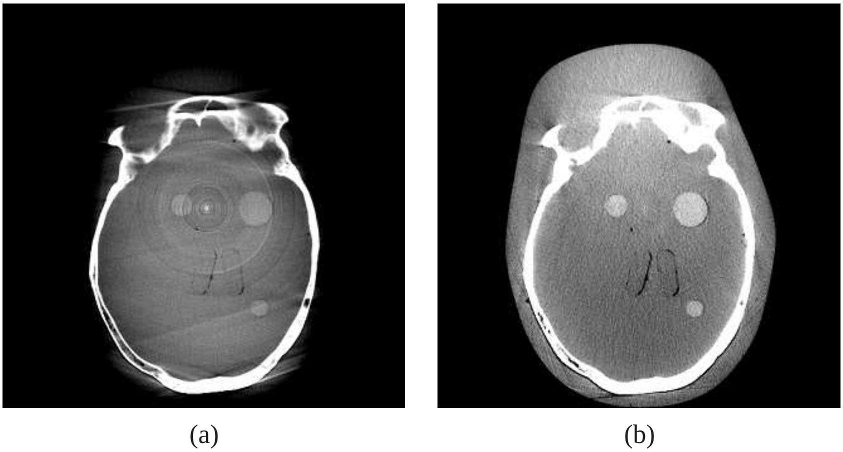

(a) (b)

Figure 2.9: *(a) Soft-tissue reconstruction of a head phantom without calibration. (b) With calibration.*

- Maintaining a constant voltage and current on the X-ray tube, and thus producing the same X-ray spectrum throughout the semi-circular movement.
- Acquiring more images (between 300 and 620 images) during the semi-circular trajectory.
- Calibrating the flat detector; For every detector pixel the gain offset and the linear and non-linear behavior is measured individually in a calibration proce-

dure. During image acquisition the measured signals are corrected, using the calibration data, see figure 2.9.

- Estimating the pixel intensity for unobstructed radiation (only traversal through air) of the detector pixels.

- X-ray radiation does not only travel in a straight line; a fraction is scattered by the imaged materials (such as the patient). The contribution of this scattered radiation to the image is estimated and subtracted from the measured image.

Chapters 3 and 4 deal with the efficient visualization of (amongst others) 3D angiographic datasets that have been obtained by the described cone-beam reconstruction technique. The clinical applications presented in chapters 9 and 10 are based on the registration of a pre-interventional dataset with a cone-beam reconstruction that has been obtained peri-interventionally. Especially the fact that the cone-beam reconstruction can be performed peri-interventionally with the same equipment that is being used to perform the minimally invasive procedure is of great clinical value, since it reliefs the patient from being transported to a CT or MR scanner and saves valuable procedure time.

Part I

Fused Visualization

The intra-operative setting and dynamics are very different from a diagnostic environment, which reflects on the requirements that are imposed on intra-operative visualization. The viewer of intra-operative images is typically not sitting behind a desktop workstation, but mostly standing at the table side were the patient lies, which limits the possibilities to interact with the workstation considerably. Also his primary focus is of course on the patient and the treatment course, rather than on the images. Even though the intra-operative images are undoubtedly of the greatest importance and very often elementary to the procedure, especially for minimally invasive treatment, they are also just one of the many stimuli that are presented to the physician during the intervention. Also of importance is the often limited time that is available to the physician to interpret the intra-operative images, especially in stress situations.

The factors mentioned above lead to the demand for visualizations that are easy to interpret and manipulate, without compromising on the visualization of the clinically relevant aspects. This rules out *e.g.*, the popular visualization methods for fused data sets, such as side-to-side or checkerboard visualization of 2D cross-sections, since they require too much interaction and are too time consuming to interpret. Rather a 3D visualization is needed. This is especially challenging for fused visualizations of multiple data sets, since the overwhelming amount of data tends to clutter the image, making it difficult to understand at a single glance, and slows down the rendering, which hinders the interactive manipulation.

In order to overcome these hurdles, the following sections describe fast volume rendering (chapter 3), our approach to reach an easy to interpret fused visualization of vascular, soft-tissue and live X-ray data (chapter 4), and the interactive visualization of volumetric data on autostereoscopic displays (chapter 5).

Chapter 3

Fast Volume Rendering

This chapter is an extended revision of the following paper:

- Daniel Ruijters and Anna Vilanova. Optimizing GPU Volume Rendering. *Journal of WSCG,* Volume 14, Number 1-3, January 2006, Pilzen (Czech Republic), pp. 9-16

3.1 Introduction

New developments in medical imaging modalities, numerical simulations, geological measurements, *etc*. lead to ever increasing sizes in volumetric data. The ability to visualize and manipulate the 3D data interactively is of great importance in the analysis and interpretation of the data. The interactive visualization of such data is challenging, since the frame rate is heavily depending on the amount of data to be visualized. Inherently, the demand for faster visualization methods is always existing, in spite of hardware innovations.

An established method for fast and interactive volume rendering on consumer hardware is GPU-based texture slicing [1–8]. Although this approach performs very well compared to CPU-based algorithms, due to the benefits from the parallelism available in the GPU pipeline, it can be accelerated significantly by taking into account the various bottlenecks that are encountered in the graphics hardware. Every individual bottleneck has a different optimal data chunk size and data throughput. In this chapter, a novel approach to accelerate GPU-based volume rendering is presented, allowing to tailor and balance the load on the individual bottlenecks to reach an optimal exploitation of the graphics hardware power.

3.2 Related work

The first rendering methods using the 3D texture capabilities of the graphics hardware were proposed by Cullip and Neumann [3], Akeley [1] and Cabral *et al*. [2]. Essentially these techniques consist of drawing polygons, which slice the volume in a back to front order. The data set is mapped as texture information on the polygons using tri-linear interpolation. The successive polygons are blended into the existing image.

Bricking is a technique to divide the volume data set into chunks, called bricks [41, 42]. It can be employed to deal with data sets exceeding the available texture memory. The bricks have then a size that is equal to or smaller than the size of the texture memory, and are loaded sequentially from main memory into the texture memory while rendering. However, this leads to significantly lower frame rates, since the bus architecture, connecting the graphics hardware to the main memory and CPU, proves to be a major bottleneck. Tong *et al.* [43] propose a bricking technique that allows skipping empty regions. Their method, however, requires new textures to be generated for every change of the transfer function, which is time consuming for very large data sets.

Texture compression can help to fit the entire volume in the main memory, and to alleviate the bus bottleneck. However, all presently available compression methods supported by graphics hardware (S3TC, FXT1, DXT1, VTC, *etc.*) are limited to lossy 8-bit RGB(α) compression, which make them unsuitable for the compression of the (often 12- or 16-bit) scalar values found in medical data, and therefore they are not used here. Further, Meissner *et al.* [6] show that the lossy compression algorithms severely reduce the image quality. Wavelet compression, as proposed by Guthe *et al.* [44] is a promising technique, but there not all parts of the volume are rendered at the highest resolution.

Not rendering all parts of the volume in the highest resolution possible is a way to reach higher frame rates, as demonstrated by LaMar *et al.* [45], Weiler *et al.* [46], Boada *et al.* [47] and Guthe *et al.* [44]. This is particularly suited to increase the render speed for perspective projections in a small view port, focusing on a detail of the volume. However, orthogonal projections of the entire volume in high resolution view ports, as is common in medical applications, can only profit from this technique at the cost of the image quality.

Space-skipping and space-leaping are techniques to accelerate volume rendering, that originate from ray-casting methods, see *e.g.*, Levoy [9], Zuiderveld *et al.* [10] and Yagel and Shi [11]. It is based on skipping empty parts of the volume. The idea of space-skipping can be applied to texture-mapping volume rendering as has been shown by Westermann and Sevenich [12].

The octree is an established multi-level data structure when dealing with voxel volumes, which has been used in numerous different applications. *E.g.*, Srinivasan *et al.* [48] apply an octree structure in volume rendering. Orchard and Möller [49] demonstrated the benefits of using adjacency information in splatting volume rendering.

Parker *et al.* have combined bricking and multi-level data structures to accelerate CPU-based iso-surface ray-tracing of volume data sets on multi-processor platforms and clusters [50, 51]. Grimm *et al.* have applied a two-staged space skipping, based on bricking and octrees, combined with gradient caching, to CPU-based ray-casting [52].

Roettger *et al.* [7] describe a GPU-based pre-integrated texture-slicing including advanced lighting. The authors also describe a GPU-based ray-tracing approach with early ray termination. Krüger and Westermann [8] propose a method to accelerate volume rendering based on early ray termination and space-skipping in a GPU-based ray-casting approach. The space-skipping addresses the rasterization bottleneck, using a single octree level only.

Here, some of the techniques cited above are combined to accelerate GPU volume rendering on a single workstation, using off-the-shelf hardware. Often it is found that acceleration of volume rendering has been treated as a singular problem to solve. The approach presented here rather focuses on the individual bottlenecks that are encountered while performing volume rendering, and tailor the different techniques to address specifically those bottlenecks.

3.3 Volume Rendering

Volume Rendering (also known as Direct Volume Rendering) is a method for visualizing volumetric data. The volumetric data assigns optical properties, such as color and opacity, to every point in the continuous three dimensional space. The Volume Rendering process then consists of following the traversal of rays of light through this three dimensional space, see figure 3.1. This is done by evaluating the volume rendering equation along the ray, as described by Kajiya [53]:

$$i = \int_{0}^{\infty} c(x) \cdot e^{-\int_{0}^{x} \tau(x')\, dx'} dx \tag{3.1}$$

Here i represents the resulting color of a ray, $c(x)$ is the emitted color at location x, and $\tau(x)$ the light absorbtion at a particular location.

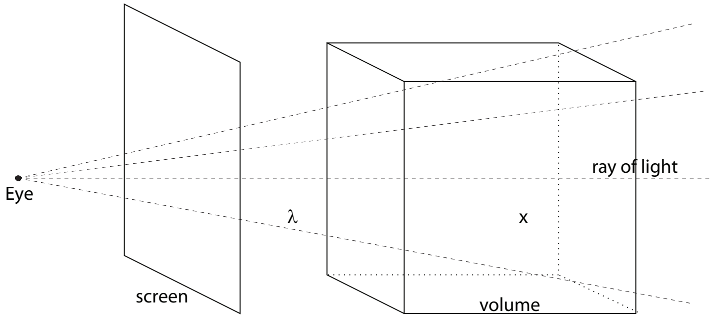


Figure 3.1: *An illuminated scene; a number of rays of light pass through the volume on the screen.*

The volume rendering equation can be approximated by the following summation [5]:

$$i = \sum_{n=0}^{N} (\alpha_n \, c_n \cdot \prod_{n'=0}^{n} (1 - \alpha_{n'})) \tag{3.2}$$

whereby α_n denotes the opacity of the volume at a given sample n, and c_n the color at the respective sample.

This summation can be broken down in N iterations over the so-called over-operator [54], whereby the rays are traversed in a back to front order:

$$C_{n+1} = \alpha_n \cdot c_n + (1 - \alpha_n) \cdot C_n \tag{3.3}$$

Here C_n denotes the intermediate value for a given ray. After N iterations, C_N represents the final color of that particular ray. N should be chosen such that every voxel is at least sampled once (we use two samples per voxel). Standard alpha blending, offered by DirectX or OpenGL, can be used to implement the over-operator.

The summation can also be evaluated from front to back by using the under-operator:

$$\begin{aligned} C_{m+1} &= (1 - A_m) \cdot \alpha_m \cdot c_m + C_m \\ A_{m+1} &= (1 - A_m) \cdot \alpha_m + A_m \end{aligned} \tag{3.4}$$

Since the ray is traversed in the opposite direction, when comparing to the over-operator, index m corresponds to $N - n - 1$ for c_m and α_m. Again, after N iterations $C_{m=N}$ represents the final color of the particular ray. When neglecting discretization issues, the over- and under-operator should deliver the same result for any given ray. It should be noted that when A_m goes to 1, any consequent $c_{m'}$ and $\alpha_{m'}$ with $m' > m$ do not contribute to the ray color anymore. A ray is then said to be saturated when A_m approximates 1. In early ray-termination, this effect is exploited to stop evaluating samples that do not contribute to the final image, and in this way the computation time is reduced.

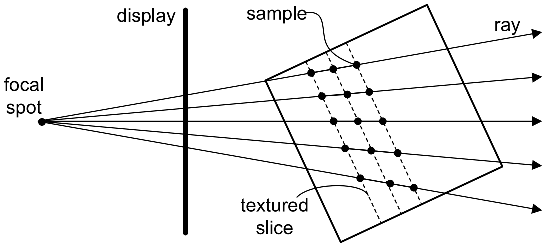
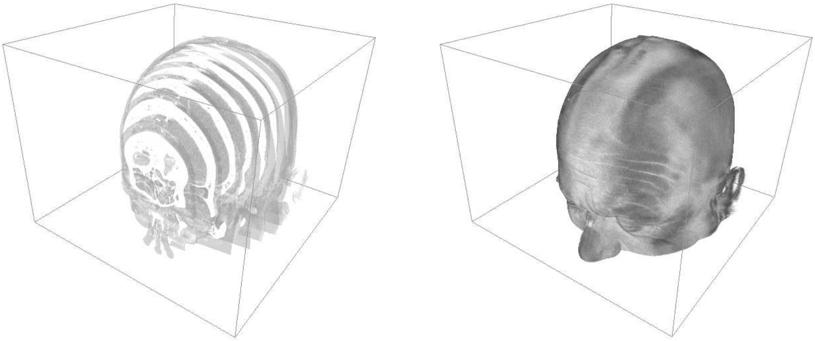


Figure 3.2: *Volume rendering involves the evaluation of the volume rendering equation along the rays, passing through the pixels of the display. The usage of textured slices means that the rays are not evaluated sequentially. Rather for a single slice the contribution of the sample points to all rays is processed.*

Equation (3.3) can be evaluated for all pixels in the frame buffer simultaneously, by using a set of N textured slices, containing the slab data, see figures 3.2 and 3.3. In iteration n, the textured slice n is then blended into the frame buffer, under the appropriate translation, rotation and perspective. Whereby the slices are processed in a back-to-front order, from the perspective of the viewer. After each iteration, all pixels in the frame buffer represent their respective C_{n+1} value.[55].



(a) (b)

Figure 3.3: *(a) A volume rendered data set, with large intervals between the textured slices. (b) The same volume rendered data set, with a small distance between the textured slices.*

3.4 Interpolation

When this method is applied to voxel data, the discrete voxel data samples have to be mapped to a continuous optical parameter description in three dimensional space. The optical parameters consist of a color and opacity component. The color component c is typically expressed as a red, green and blue tuple (RGB), and the opacity as a single value α. There are essentially two methods for mapping the discrete scalar voxel samples to the continuous optical parameter description:

- The scalar voxel data is interpolated to a continuous scalar description, using some kind of interpolation function, such as nearest neighbour, linear or cubic interpolation. Then a transfer function is applied, mapping the scalar voxel range to optical properties, *i.e.*, color and opacity (*e.g.*, in the form of a lookup table).

- The transfer function can also first be applied to the scalar discrete voxel samples. Then the interpolation to the continuous three dimensional space is performed to the optical parameters.

The voxel data consist of a three dimensional array on a uniform grid containing discrete values. The array can be regarded as a set of weighted dirac impulses, at regular intervals (the interval is constant in each direction, but might be different for each individual axis). Shannon's theorem states that if the original signal was bandwidth limited, and it was sampled with at least twice the highest frequency that was present in the original signal (Nyquist rate), it is possible to exactly reconstruct the original signal. In order to perform such an optimal reconstruction the set of sampled data (array of dirac impulses) has to be convolved with the sinc function ($f(x) = \sin(x)/x$) (see figure 3.4C). For a more in depth discussion see *e.g.*, [56].

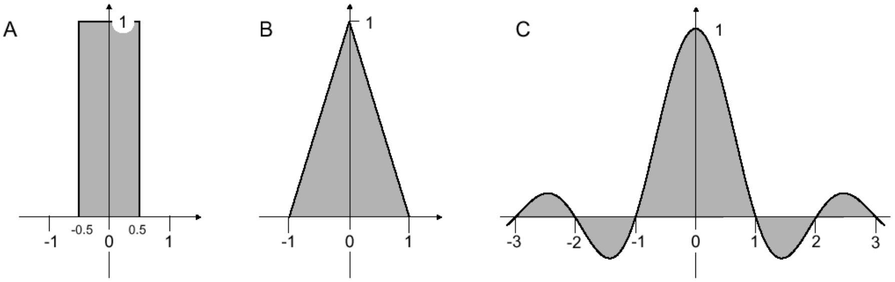


Figure 3.4: *One dimensional versions of the reconstruction filters: (A) the rect filter, (B) the hat filter and (C) the sinc filter*

Unfortunately a convolution with the sinc function cannot be performed on the graphics hardware, and therefore would cost too much performance. Two convolution kernels can be used on the graphics hardware: the rect function, and the hat function. A convolution with the rect kernel (see figure 3.4A) corresponds to nearest neighbour interpolation, and a convolution with the hat kernel (see figure 3.4B) corresponds to trilinear interpolation. Modern consumer graphics hardware provides very efficient nearest neighbour, bi- and tri-linear interpolation. Tri-linear interpolation provides clearly a better image quality, as figure 3.5 shows. However also tri-linear interpolation produces artefacts, since the corresponding Fourier transform is not bandwidth limited and higher order frequencies lead to aliasing. An even better result can be achieved using cubic interpolation [57, 58], which also can be GPU accelerated [13, 59].

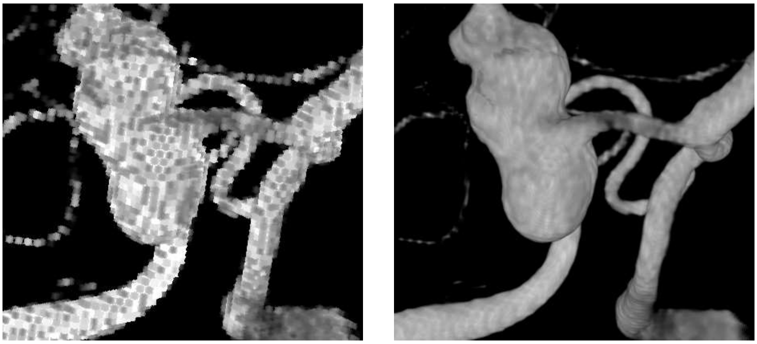


Figure 3.5: *Nearest neighbour and tri-linear interpolation*

3.5 Pre- versus post-lookup

When voxel data originates from techniques like MR, CT or 3D rotational angiography, the subject of interest is typically to be found within a certain range of voxel

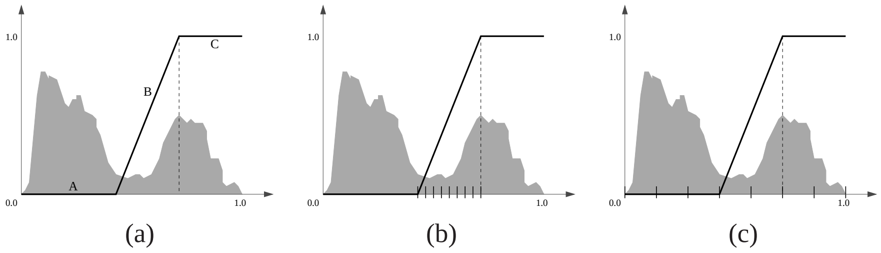


Figure 3.6: *(a) Histogram and transfer function, (b) distribution of discrete values for pre-lookup, and for (c) post-lookup*

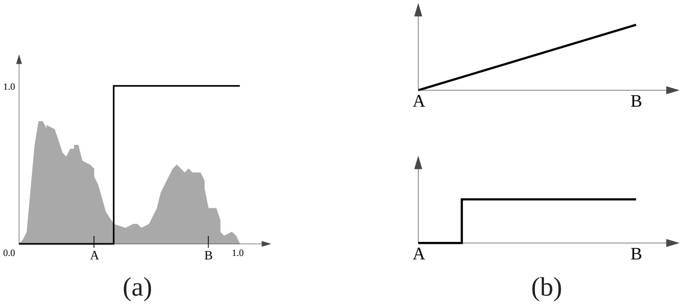
values. A transfer function could make this range visible and hide others, and thus make the subject of interest visible. A transfer function might also enhance or suppress certain properties of the image (*e.g.*, contours enhancement).

The transfer function can be performed before interpolating the volume (see section 3.4), which is called pre-lookup, or after the interpolation, known as post-lookup. Pre-lookup simply involves looping over all voxels and performing the transfer function for every voxel value. Post-lookup is somewhat more complicated. It means that whenever an interpolated value is used in the volume rendering process, first the original voxel values are taken and interpolated (using *e.g.*, nearest neighbour or trilinear interpolation), and then the interpolated value is the input for the transfer function.

Figure 3.6a contains an example of a transfer function. The grey graph represents the histogram of the voxel values present in a certain volume. The solid black line represents the transfer function. In this particular case all voxels with a value within the range of the first part (A) are mapped to 0, for the middle part (B) a linear lookup is performed and for the last part (C) all values are mapped to 1. Figure 3.6b demonstrates for a pre-lookup, how the discrete dynamic range is distributed over the middle part. Figure 3.6c shows that for a post-lookup the discrete dynamic range is distributed over the entire range of the function (After all, in that case the transfer function is only performed after the interpolation step). It might be obvious that if the bit depth of the voxel data is higher than the bit depth of the transfer function, the pre-lookup method has a richer dynamic range, meaning that it uses a bigger set of visible voxel values. However the post-lookup method will produce more accurate spatial results.

Why does the post-lookup method produce more accurate spatial results? Assume a binary transfer function like in figure 3.7a. Now consider a linear interpolation between two adjacent voxel values, one has value A in figure 3.7a, the other has value B. If the pre-lookup method was used, value A will be mapped to 0, value B to 1, and the linear interpolation will produce a gradual (linear) transition from 0 to 1 for the space between the voxels, as is shown in figure 3.7b top.

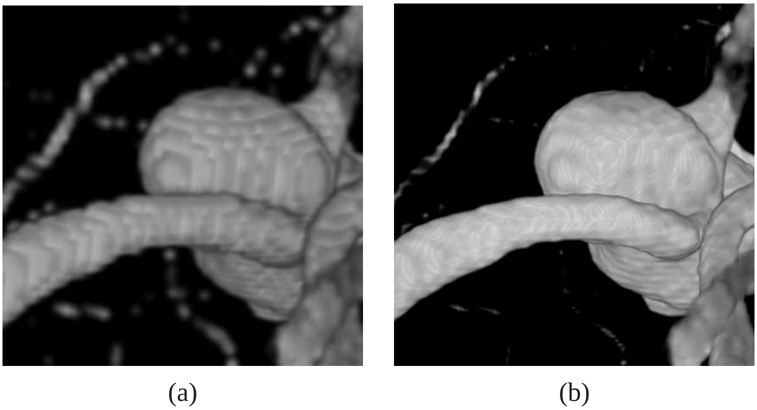
For the post-lookup method first a linear interpolation from A to B will be performed, and then on the interpolated values the transfer function will be applied. This will result in a binary transition, whereby the boundary will be close to the voxel with



(a) (b)

Figure 3.7: *(a) A binary transfer function transfer function, (b) pre-lookup (top) and post-lookup (bottom) interpolation between two adjacent voxel values*

value A, as depicted in figure 3.7b bottom. This is the result we would expect.



(a) (b)

Figure 3.8: *A volume displayed with the same transfer function using (a) the pre-lookup and (b) the post-lookup method*

As mentioned earlier, the post-lookup method is performed after interpolation. In practice that means that post-lookup is performed in the rasterizing step in the render pipeline (see section 3.6). The interpolation as well as the post-interpolative transfer function in the form of a lookup table can be easily performed by a GPU program. Note that pre-lookup and post-lookup can be used in addition to each other.

3.6 Bottlenecks

The rendering pipeline is the general process flow that is being used to depict virtual three dimensional scenes. Such a scene consists of flat geometrical primitives, such as points, lines, triangles and polygons. The rendering pipeline can be implemented in hardware to a various degree [60].

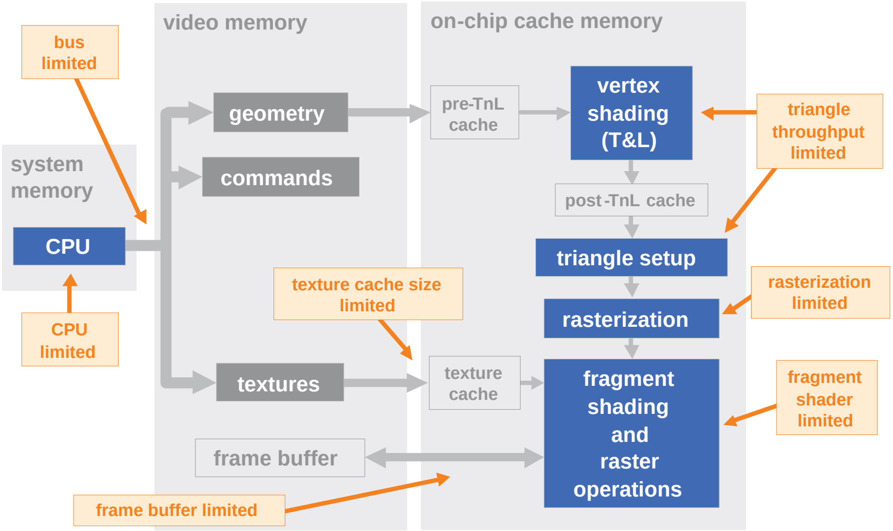


Figure 3.9: *The graphics hardware pipeline and its bottlenecks (adapted from [61]), light grey: memory units, dark grey: data structures, blue: processing units, red: bottlenecks.*

The process of rendering an image involves traversing all primitives in order to transform their scene coordinates to camera coordinates. A lighting model is performed on the primitives, and the results are stored as color per vertex. The next step is rasterization of the primitives. Rasterization converts the above mapped primitives into fragments. Fragments correspond to pixel locations in the frame buffer, and contain some properties such as color, texture coordinates, and depth (z buffer). A fragment is one-to-one associated with a pixel. Before being placed into the frame buffer, each fragment may be subjected to a series of tests and modifications. These include stencil test, depth test, and blending. Finally, the two dimensional image that has been rendered in the frame buffer, is displayed on the screen.

Although textures might have three dimensions (thus volume data), the described rendering pipeline does not allow the direct rendering of volumetric objects. All geometrical primitives are flat. The rendering pipeline merely offers the possibility to calculate an intersection with a flat primitive in the volume data, using interpolation.

Figure 3.9 illustrates the graphics pipeline, employed for GPU-based volume rendering [61]. Here the most important points in the pipeline that result in a bottleneck are discussed.

The bus - The volume data has to be transferred over the bus from the system memory into the graphics card memory. Since this is the slowest part of the entire pipeline, these transfers have to be as few as possible.

Triangle throughput - The triangle throughput is mainly limited by the vertex shading and triangle setup phase. A straight forward implementation of texture-mapping volume rendering would involve only few triangles, but techniques for space-skipping may increase the amount of triangles considerably. If the triangle count becomes too high, this will become a limiting factor for the frame rate.

Rasterization - When performing volume rendering based on texture slicing, the vast majority of the pixels on the screen are accessed multiple times. Space-skipping techniques may be used to reduce the amount of pixels to be accessed, but this also increases the triangle count.

Texture cache size - Texture lookup is one of the more time consuming operations performed during the rasterization step. When the texture fits in the cache, these lookup operations will be faster.

Fragment shader - Fragment shader programs impact the duration of the rasterization step. Simple fragment programs, such as applying a lookup table, generally do not limit the frame rate, however more complex operations, such as specular lighting [6, 7], multi-dimensional transfer functions [62] or pre-integrated rendering [4, 5, 7], can form a bottleneck. Especially fragment programs that perform multiple texture lookups (*e.g.*, on-the-fly gradient calculation for specular lighting) are relatively slow.

3.7 Method

When performing volume rendering usually only a fraction of all voxels actually contribute to the final image, since a relatively small amount of voxels are of interest and another amount of them are occluded. Furthermore, datasets are often sparse. In 3D medical data sets (obtained by *e.g.*, ultrasound, CT, MR or rotational angiography [63, 64]) the anatomical structures of interest encapsulated in the data sets occupy only a part of the total data. Typically 5% to 40% of all voxels contain visible data, and even highly filled CT or MR data sets rarely exceed 55%. Especially vascular data sets can be marked as sparse data sets, since vessels, due to their tubular form, occupy only a small percentage of the volume (1% to 8%).

This chapter seeks to reach the maximum benefit in exploiting skipping void parts of the volume (space-skipping). The novelty that is introduced lies in dividing the space-skipping in two stages; a course division using bricking (figure 3.10a) and a finer one using octrees (figure 3.10b). These steps are based on an analysis of the bottlenecks encountered in the graphics pipeline when performing texture-mapping volume rendering. The first stage, bricking, is chopping the volume in so called texture bricks. The bricks are loaded into the video memory, to serve as data for the volume rendering algorithm, which is executed by the GPU. The bricks address the bus- and texture cache size-bottleneck. To further alleviate the load on the fragment shaders, early ray termination is applied to each brick additionally. This benefits especially highly-filled data sets. The second stage is employing an octree within each brick. The octrees address the rasterization bottleneck. As will be demonstrated, the two stages have to be balanced, because lifting one bottleneck may overload another bottleneck (*e.g.*, rasterization bottleneck versus triangle throughput bottleneck).

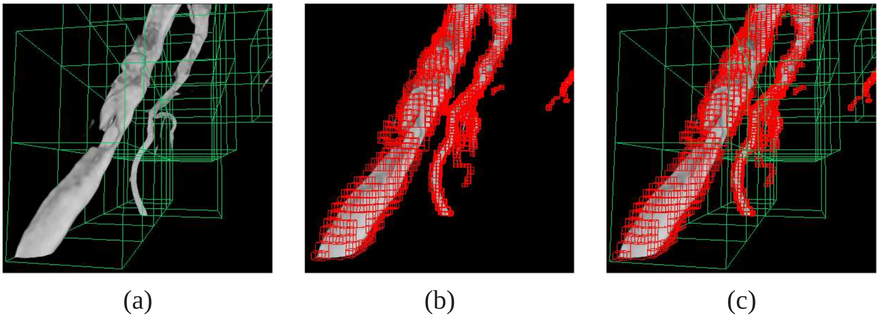


Figure 3.10: *The same volume fragment, rendered with (a) bricking cubes visible, (b) octree cubes visible (note the various cube sizes) and (c) both bricking and octree cubes visible.*

The role of the transfer function in volume rendering is to map the scalar voxel information to the optical properties (*i.e.*, color and opacity) [62]. The above described approach is implemented such that the flexibility to change the transfer function at run-time is preserved. This offers the possibility to focus on different scalar ranges in the volume, without lengthy calculations. To accomplish this, the unmodified scalar voxel values are stored in the brick textures, and a fragment shader program is used, to lookup the RGBα values after interpolation of the scalar voxel values has been performed. Since the octrees depend on the visibility of the data, and thus on the transfer function, they have to be recomputed when the transfer function changes. This can be done on the fly, as will be explained below.

3.8 Bricking

As mentioned in section 3.2, the voxel volume can be divided into chunks, called bricks, in order to cope with voxel data sets sizes exceeding the size of the texture memory of the graphics hardware. Note that these bricks contain the original scalar values of the voxel volume, thus the values before applying the transfer function. This enables us to change the transfer function on the fly, since a transfer function change does not require creating new textures.

To obtain a correct interpolation at the bricks' boundaries it is necessary that the data held by adjacent bricks overlap. The overlap depends on the convolution kernel used for interpolation [57], and should correspond to $(kernelsize - 1)$. For nearest neighbor interpolation that means that no overlap is needed, since the width of the kernel is one. For tri-linear interpolation the overlap should be one voxel in every direction (for other kernels the overlap may even be larger). Pre-integrated rendering [4, 5, 7] or the on-the-fly calculation of gradients require the overlap to be increased by another voxel in every direction. For bricks of b^3 voxels and an overlap of n voxels, the memory overhead is approximately $(3n/b) \cdot 100\%$.

The bricks are loaded into the video memory as 3D textures. Many graphics cards require 3D texture sizes to be a power of 2 in every direction. If the volume dimensions do not divide evenly into brick dimensions, either an additional layer of partially empty bricks should be added in each direction, or smaller rest-bricks should be used.

When the amount of data in the textures exceeds the available texture memory, textures are swapped between the main memory and the texture memory. If a requested brick is not resident in the texture memory, it is loaded from the main memory, replacing resident textures [60]. In most OpenGL implementations resident textures are swapped out on a Least Recently Used (LRU) base.

Traditionally bricking in texture based rendering is used to be able to render data sets which exceed the size of the texture memory of the graphics hardware. The bricks are then chosen to be as large as possible, and they are sequentially loaded from the main memory into the texture memory. This implies that for each frame the entire volume data is transferred over the bus.

In the presented approach, however, considerably smaller brick sizes are chosen. The smaller the brick size is, the bigger is the chance of bricks being completely void after applying the transfer function, and void bricks do not need to be drawn. Therefore, once they are swapped out of the texture memory, they are never reloaded into the texture memory, and thus the bus bottleneck is alleviated.

Bricking is even applied to volumes that completely fit into the texture memory to improve data locality, which will result in less cache trashing on the graphics card [65–67]. On the other hand smaller bricks could introduce a larger overhead due to the overlap needed for interpolation. Thus the optimal brick size needs to be defined depending on the available texture memory, optimal texture size (see section 3.6), nature of the data set, overhead due to overlap, and the constraints posed by the graphics hardware.

3.9 Early ray termination

To be able to perform early ray termination at all, the volume has to be traversed in a front-to-back order. This can be done by evaluating the volume rendering integral in discrete steps, using the under operator, see equation 3.4.

Before a brick is rendered, early ray termination is applied to its destination pixels. This is tested by executing a fragment shader program, while drawing a solid bounding box around the brick with back face culling switched on. The fragment shader program writes the maximal value in the depth buffer for saturated rays [7, 8]. When slicing the brick texture the early z-test will prevent any fragment operations to be executed for those rays, reducing the load on the rasterization and fragment shader bottlenecks. Early ray termination is only performed once per brick, and not more often (*e.g.*, for every octree node or every sample) because the overhead involved (changing fragment shaders, performing the test) would otherwise annihilate the benefits.

3.10 Octree

By not rendering the void bricks, the load on the rasterization bottleneck is already reduced. The load is even further reduced by applying octrees. Every brick possesses its own octree. Every octree node corresponds to a cuboid part of the voxel volume, which can be divided into eight parts, corresponding to the child nodes (see figure 3.11). The octree is kept in main memory. It only describes the geometry of the visible data. The actual voxel data is to be found in the brick textures.

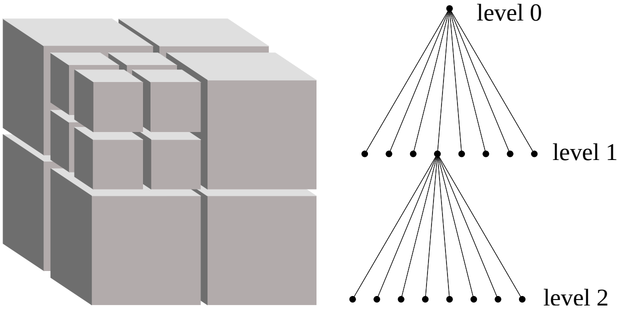


Figure 3.11: *An octree division, and its tree*

For tri-linear interpolation, let a cell be defined as a cube with adjacent voxel values assigned to its eight corners. For every position within the cell an intensity value is defined as the tri-linear interpolation of the corner values. Therefore a cell can only be completely void if its eight corner values are completely transparent $(\alpha = 0)$ after applying the transfer function. This definition can easily be extended to any given interpolation kernel, by setting the size of a cell to $(kernelsize - 1)^3$.

Every octree node carries a variable describing the ratio r of visible data to total data within its cube. At the final level of the octree, every node represents uniquely one cell, and is considered either completely filled $(r = 1)$ or void $(r = 0)$. Every higher octree level nodes ratio can be calculated by averaging the ratios of its children. This calculation only needs to be performed when the transfer function has changed.

Rendering an image means that the bricks have to be processed in a front to back order. For each brick the respective octree is traversed, starting with its parent node. Depending on its ratio r there are three ways to process a node:

$r = 0$**:** The node is completely void. It is not drawn at all, and is not traversed any further.

$0 < r < threshold$**:** The nodes children will be traversed, and to each child node this strategy will be applied recursively.

$r \geq threshold$**:** The node is drawn completely. It is not traversed any further.

If the $threshold$ is set to 1, exactly all filled cells will be drawn, and no void cells. However, that would lead to a lot of tiny cubes at the boundaries of the visible data structures, and thus the load on the triangle throughput bottleneck becomes too high.

Therefore a lower $threshold$ should be chosen to allow some degree of void data to be drawn.

Further, an octree level l is specified at which nodes lower in the hierarchy are not traversed anymore. At this level, any node that is not void will be drawn completely. This strategy is applied to prevent that the overhead of traversing the octree completely down to the leaves becomes bigger than the time that is won by skipping the empty parts. Additionally, this approach enables us to update the octree on the fly, when the transfer function changes. The octrees nodes at level l contain the minimum and maximum voxel value that is represented by the voxel samples in their corresponding cubes. This is constant data, independent from the transfer function. When the transfer function changes, only the visibility of the octree nodes higher than level l needs to be computed again. However, this set of nodes is only a fraction of the amount of voxels, and therefore this recalculation can be performed very quickly.

When traversing a node, its children have to be sorted in a front to back order. Since there are eight children, it would seem that there are $8! = 40320$ ways to arrange the children. But since the arrangement along the three perpendicular axes is the same for all children, there remain $2^3 = 8$ possible orders. When a node is to be drawn, the cuboid box corresponding to this node is sliced, and the slices are rasterized and blended into the previously drawn slices. The slices can be axis-aligned or viewport-aligned. For the most straight-forward form of volume rendering, the brick texture is interpolated on every slice, taking its position in the brick into account, and after interpolation the transfer function is applied. However, it is also possible to perform more sophisticated forms of volume rendering on the slices, like pre-integrated volume rendering or include specular lighting [6, 7].

The octree is generated and traversed on the CPU. Its purpose is to lower the workload on the graphics pipeline, and thus the GPU. The octree reduces the time that the GPU spends on processing data which never contribute to the final image. The actual volume rendering algorithm, as well as interpolation, the post-interpolative transfer function, and optionally, specular lighting, is being performed by the GPU.

3.11 Results

The described approaches have been tested with several different graphics cards: the nVidia QuadroFX 3400 (256MB on board memory), the ATi FireGL X1 (128MB), and the 3DLabsWildcat 7110 (256MB). With each card the volume in figure 3.12b has been rendered, using the same lookup table settings. The volume data concerned the iliac arteries, acquired through 3D rotational angiography. Since contrast media was injected into the vessels, the vessels could easily be classified using the transfer function. Only 3% of the voxels in this volume contain visible data. All results have been obtained using a view port of 800^2 pixels and the sample rate for the volume rendering equation was set to 1.5 samples per voxel.

Since the optimal brick size is mainly determined by the properties of the texture memory (see section 3.8) and the optimal octree limit is primarily used to balance the rasterization load and the triangle throughput (see sections 3.6 and 3.10) they can be considered to be fairly orthogonal variables. Therefore their optimum can be found

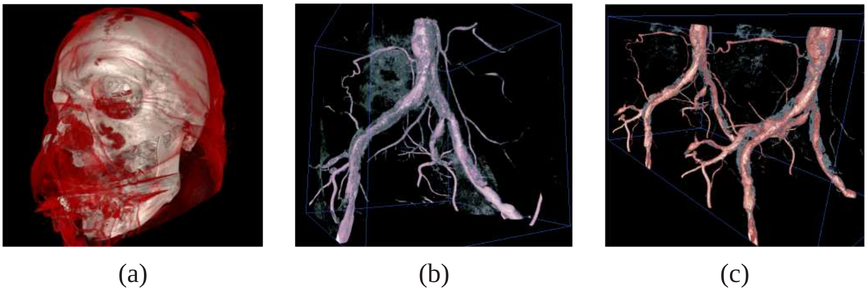

(a) (b) (c)

Figure 3.12: *Test volumes: (a)* 512^3 *volume, used for testing early ray termination, (b) vascular* 512^3 *volume, (c) gigabyte volume of* $642 \times 642 \times 1284$ *voxels, generated by duplicating a large 3D-RA volume.*

by varying one variable, while keeping the other one constant.

On each graphics card the test volume was rendered with different brick sizes, see figure 3.13, while the octree limit was set to 8^3 voxels. The ATi FireGL X1 and the 3DLabs Wildcat 7110 clearly show that their optimal brick size is considerably smaller than their largest possible brick size. The nVidia QuadroFX 3400 does not benefit from the bricking for the 256MB test volume. However, also this card clearly profits from the bricking for the sparse 1GB volume in figure 3.12c: the optimal brick size is then 64^3 voxels, with an average frame rate of 37 fps, while for 256^3 bricks only a mere 3.1 fps is reached.

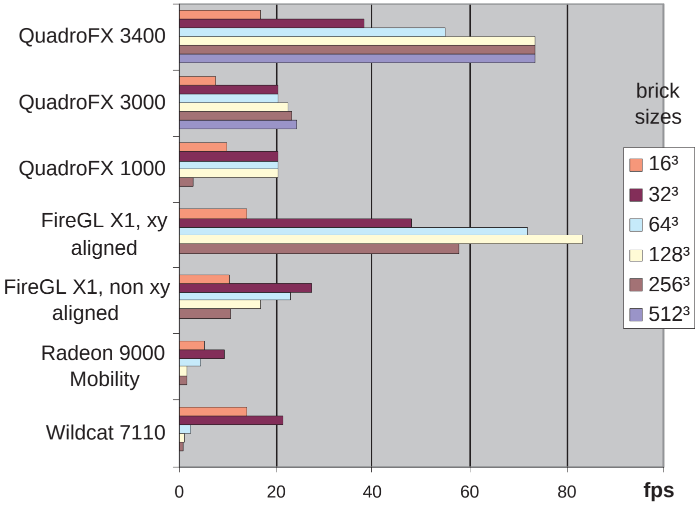


Figure 3.13: *Performance using different brick sizes.*

The performance of the ATi FireGL X1 depends heavily on the sampling direction of the bricks, because the ATi card treats the 3D textures as a stack of 2D slices. When the bricks are traversed in the x or y direction, the slices are accessed rather linear, and the performance is much better than when they are traversed in the z direction. It is inevitable to traverse in the z direction, when the viewing direction and the z-axis of the textures differ more than $45°$. This effect can be reduced by alternating the orientation of the textures for each consecutive brick [42]. Especially striking is the fact that the optimal brick size and octree limit is different for each sampling direction. When sampled in the xy-plane direction larger bricks benefit from linear traversal, while in other directions smaller bricks benefit from less cache trashing. In figures 3.13 and 3.14 this fact is illustrated by the performance measurement when sampling was aligned to the xy-plane, and when not.

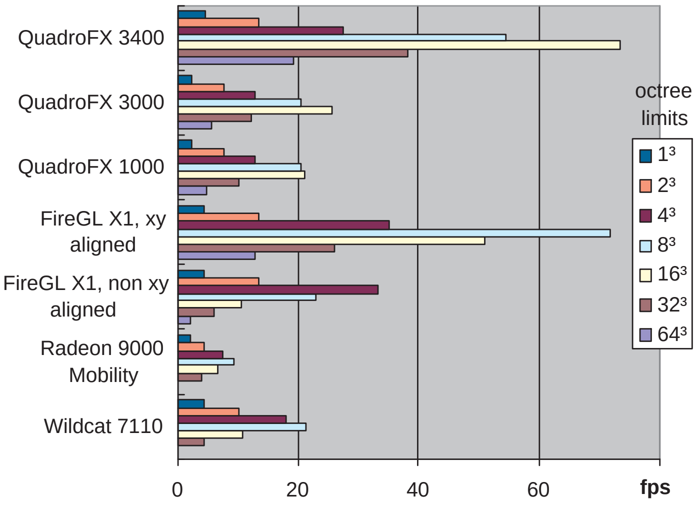


Figure 3.14: *Performance using different octree limits.*

Further, the volume was rendered with a fixed brick size of 64^3 voxels and variable octree limits (the octree limit is the smallest octree cube allowed). Not every octree branch reaches this limit, see section 3.10. Figure 3.14 unsurprisingly shows that there is an optimum octree size for every graphics card. Smaller octree limits lead to too much CPU overhead and triangle count, and larger octrees to too much rasterization overhead. The 64^3 octree level corresponds to not using any octrees at all, only bricking.

Table 3.1 shows the acceleration achieved, using the volume in figure 3.12b, with an optimal combination of brick size and octree depth for each particular graphics card versus the same GPU volume rendering routines applied without any bricking or octrees at all. Since early ray termination does not provide any performance gain for sparse data sets, it was not used on this volume.

Early ray termination was tested on the QuadroFX 3400 using the volume in figure 3.12a. GPU volume rendering without optimizations yielded 2.2 fps, using 64^3 bricks and 8^3 octree limits 5.2 fps were reached, and with additionally early ray

Graphics card	a Optimized	b Non-optimized	a/b
nVidia QuadroFX 3000 AGP	25.5 fps	2.2 fps	11.6
nVidia QuadroFX 3400 PCIx	73.5 fps	9.6 fps	7.66
ATi FireGL X1, xy aligned	83.3 fps	0.23 fps	362
ATi FireGL X1, non xy aligned	27.4 fps	0.23 fps	119
ATi Radeon 9000 mobility	9.35 fps	0.26 fps	36.0
3DLabs Wildcat 7110	21.3 fps	0.38 fps	56.1

Table 3.1: *Average frame rates reached when using (a) best combination of bricking and octrees, (b) GPU rendering without bricking or octrees.*

termination switched on, the average frame rate was 16.1 fps.

Since the rendering primarily depends on the graphics card, replacing *e.g.*, a Xeon 3.0GHz by a Xeon 1.7GHz delivered approximately the same performance figures. The only part which is bounded by the CPU and main memory performance is building a new octree after the transfer function has been changed. For a volume consisting of 512^3 voxels (16 bit per voxel, 256MB for the entire volume), rendered with a brick size of 64^3 voxels and an octree limit of 8^3 voxels, building all new octrees for the entire 512^3 volume took 6.5 milliseconds on the Xeon 1.7GHz and 3.5 milliseconds on the Xeon 3.0GHz machine.

3.12 Conclusions

In this chapter, an approach to accelerate GPU-based volume rendering was presented. The approach consisted of a two staged space-skipping and early ray termination, and was tailored to lift the various bottlenecks encountered in the graphics pipeline.

In the first stage, the entire volume is chopped into bricks, and from these bricks 3D textures are created. Empty bricks are never drawn, nor kept in the video memory, and therefore the bus bottleneck is relieved. The optimal brick size depends on the nature of the data (there should be a reasonable chance that there are bricks which are completely void), the available texture memory, the texture cache size and the overhead introduced by brick overlap. Since the brick textures content does not depend on the transfer function, they need to be created only once for static data.

The octrees, which form the second stage, focus on skipping data that is not visible after applying the transfer function. In this way the rasterization bottleneck is addressed. To prevent too much overhead to be introduced, a certain amount of void data per octree box is allowed, and there is a limit to the granularity of the octree boxes. The optimal octree parameters are determined by the weight of the rasterization phase (*i.e.*, are there complex fragment shader programs involved, *etc*.) and the trade-off between less rasterization operations and more triangles (triangle throughput bottleneck). Since the octree depends on the transfer function, it has to be recalculated when the transfer function changes.

In this chapter it has been shown how the individual bottlenecks have been addressed by a two-folded approach. First the bus bottleneck and texture cache size has been addressed by bricking, and consequently the rasterization bottleneck has been addressed by the octrees. The rasterization and fragment shader bottleneck were further lifted by employing early ray termination. The results show that the parameters can be optimized for different graphics cards. Since the transfer function only leads to recalculating the octrees, and not reloading the bricks, it can also be changed quickly and interactively.

The graphics industry are introducing more powerful hardware at an impressive pace. However developments in medical imaging modalities are equally impressive, resulting in larger volume data sets. Which means that in the foreseeable future the techniques that were presented here will preserve their benefits.

Chapter 4

Fusion of Vascular, Soft-tissue and X-ray data

This chapter is based on the following papers:

- Daniel Ruijters, Drazenko Babic, Bart M. ter Haar Romeny, and Paul Suetens. Silhouette Fusion of Vascular and Anatomical Data. *Poceedings of IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI'06)*, April 2006, Washington DC (USA), pp. 121-124. doi:10.1109/ISBI.2006.1624867
- Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. 3D Multi-modality Roadmapping in Neuroangiography. *Proceedings of SPIE - Volume 6509, Medical Imaging 2007: Visualization and Image-Guided Procedures*, February 2007, San Diego (USA), pp. 65091F. doi:10.1117/12.708474
- Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. Real-time integration of 3-D multimodality data in interventional neuroangiography. *Journal of Electronic Imaging*, Volume 18, Issue 3, July-September 2009. doi:10.1117/1.3222939

4.1 Introduction

In this chapter it is described how data coming from multiple modalities can be represented in a single fused image. We apply the fusion to vessels that were segmented from a 3DRA acquisition and soft-tissue information in a voxel dataset, which typically originates from a CT or MR scan. The live 2D X-ray fluoroscopy video stream is merged into this combined bi-modal representation. Taking into account the intra-procedural usage of this visualization, we aim at creating an image that can be rendered in real-time and is easy to interpret, while containing all important clinical aspects of the rendered data. The described techniques are not restricted to this particular application or modalities, though, but can also be applied in other situations.

Image fusion is the domain of combining two (or more) datasets in a combined visualization. In the state of the art, regarding the fusion of volumetric datasets, two classes of algorithms can be distinguished:

3D Compositing, whereby structures further from the viewer are obscured by closer ones. This includes surface rendering techniques, such as iso-surface rendering

[68] and mesh rendering [69], as well as direct volume rendering of multiple datasets [70–74] and combining surface and volume rendering [75–77]. The advantage of this class of algorithms is the fact that the spatial relationship and topology between the datasets is very clear. Obscured structures can be made visible by using a high level of transparency, though this tends to clutter the resulting image with information. Another way to visualize obscured structures is hiding a part of one of the datasets, *e.g.*, by using clip planes or clipping volumes.

Overlaying is the other class, whereby selected internal structures influence the resulting image, regardless whether they are obscured. In its simplest form this means that the resulting image is composed of the combination of the separate 2D projection of both datasets [78], using *e.g.*, blending, but also the combination of volume rendering techniques with Maximum Intensity Projection [79] can be considered to be a member of this class. The benefit of this method is the fact that no (relevant) information is hidden because of occlusion. On the downside, however, the representation of the topology of the data sets is lost.

We intend to use the best of both classes, by combining them in a hybrid approach, without cluttering the result with an overload of data, and thus yielding an image that is easy to interpret. Our fused visualization is achieved by presenting a subset of the soft-tissue dataset and the live fluoroscopy data in a single image. In order to obtain such an image, a soft-tissue dataset (*e.g.*, a CT or MR dataset) is registered with a 3DRA reconstruction, see chapter 6. Such a representation allows a direct correlation of the position of the live data in the fluoroscopic video stream and the multiple 3D datasets. The fast visualization, achieved by using off-the-shelf graphics hardware, is an integral part of the method.

4.2 Method

4.2.1 Volume and mesh blending

In order to render a fused image, first the triangulated mesh, representing the vessels, is rendered in the frame buffer. Simultaneously the depths of the triangles are written in the z-buffer. Consequently, a slab out of the voxel dataset is mixed into the scene using direct volume rendering. The position, orientation and thickness of the slab can be altered by the user. The slab is rendered by evaluating the direct volume rendering equation for each pixel in the view port. To mix the triangulated mesh and the direct volume rendering, we test the z-buffer at each iteration of the over operator (equation 3.3). If the z-buffer test shows that, for a particular pixel, the position of the present sample of the ray is further away from the viewer than the triangle in the frame buffer, the frame buffer remains unchanged. The first sample that lies closer to the viewer will take the present value of the frame buffer as input, which was written by rendering the triangulated mesh, see figure 4.1. In this way the color of the mesh is blended into the volume rendering equation at the appropriate place. An example of such a blended image can be found in figure 4.5b.

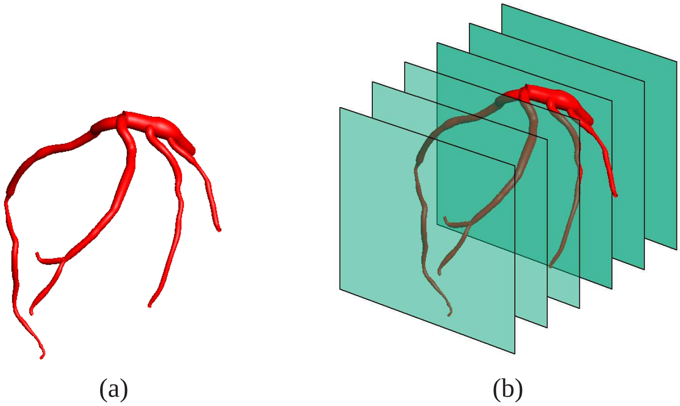

(a) (b)

Figure 4.1: *(a) To fuse meshes and voxel data, first the mesh is rendered to the frame and z-buffer. (b) Then the volume rendering equation 3.2 is evaluated by drawing textured slices in a back-to-front order. As long as the slice information lies behind the mesh, the z-buffer test will prevent the mesh pixels to be overwritten. When the slices are in front of a mesh pixel, the mesh color will be fed into the volume rendering equation.*

The mesh, representing the vessels that were segmented from an intra-operative 3DRA dataset, and the soft-tissue MR or CT data will typically not be contained in the same coordinate space. The framework presented in appendix A can be applied directly during the visualization of the mesh and textured slices, which makes a resampling of the slab with the soft-tissue data to the grid of the 3DRA data unnecessary, leading to a better image quality [70].

4.2.2 Silhouette overlaying

In order to show relevant internal structures that are occluded by the volume rendered data, we optionally overlay this data with the silhouette of the mesh. The silhouette only shows the outline of the mesh structures, and therefore it barely obscures any previously rendered parts of the image. On the other side it indicates the shape of the occluded mesh. Furthermore, the interface of the occluded and visible part of the mesh allow to locate the silhouette within the 3D scene. An example of this silhouette rendering is presented in figure 4.2.

The silhouette render technique we apply is based on the method described by Raskar and Cohen [80]. In order to render a silhouette, first the front faces (triangles with a normal pointing to the viewer) of the mesh are rendered to the z-buffer only. This can be achieved by drawing the triangles with a completely transparent color, or by switching the color mask off. Consequently the wire frame of the back faces (triangles with a normal pointing away from the viewer) of the mesh are rendered, but this time with a solid color (red in our case), a line thickness larger than 1 pixel (2 pixels in our case), and z-test enabled.

This process will lead to lines only being drawn where the front facing and back facing triangles meet, and thus to a silhouette of the mesh. Again the whole silhouette

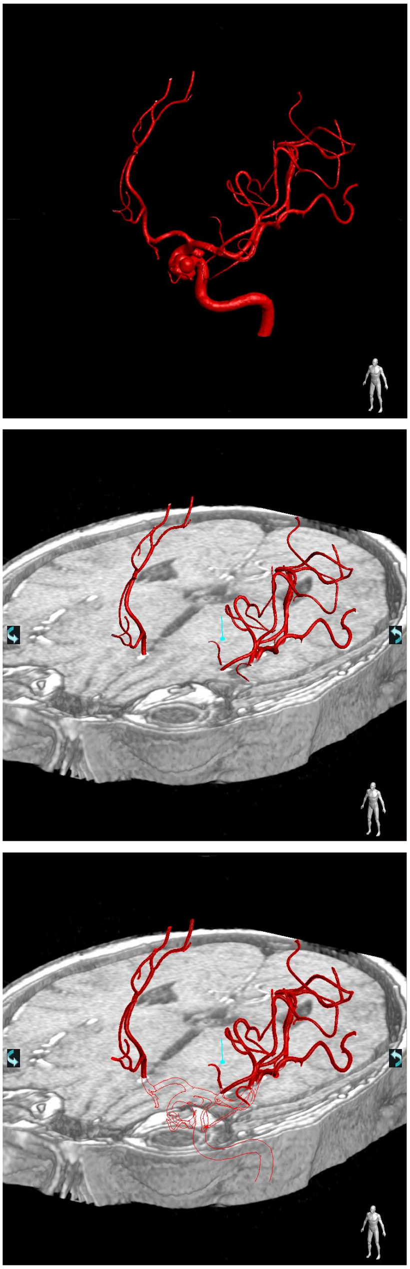
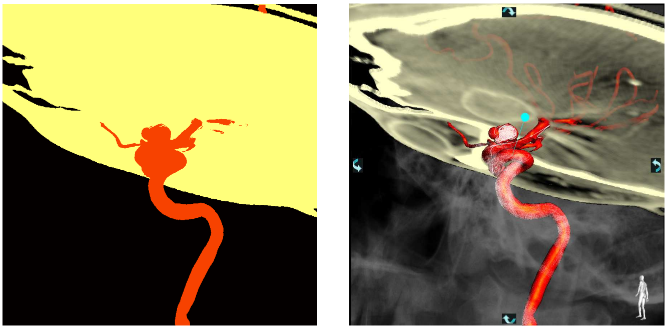


Figure 4.2: *The silhouette rendering allows to visualize the occluded part of a scene and to present the relation with the contextual data, without cluttering the image. Top image: cerebral vessels, segmented from the 3D-RA dataset. Middle image: the cerebral vessels, combined with a volume rendered slab out of a MR dataset, which obscures the aneurysm. Bottom image: the segmented vessels and MR slab, overlaid with the silhouette of the vessels.*

rendering can be easily implemented to employ the graphics hardware using either the DirectX or OpenGL API.

4.2.3 Blending with the 2D X-ray image

For the interpretation of the fused image it is beneficial to process parts of the X-ray fluoroscopy data differently, depending on their underlying 3D information. In order to achieve this, it is necessary to establish which kind of 3D data is contained in each pixel. Here this information is obtained by employing the stencil buffer (for a description of the stencil buffer, see [60]). For every pixel in the frame buffer that is filled by the mesh, a constant value S_1 is written to the stencil buffer. Also while rendering the voxel data slab, a stencil buffer operation is defined to write a constant S_2 to every pixel that receives a color value from the direct volume rendering process, with $\alpha > 0$. The S_1 labels can be overwritten by S_2 during this operation, see figure 4.3.



(a) (b)

Figure 4.3: *(a) The mesh and the voxel slab write different values to the stencil buffer. (b) The stencil buffer can then be applied to process the fluoroscopy image during rendering.*

Finally the current fluoroscopy image is blended into the frame buffer, which is done in multiple passes. The action that is performed on a given pixel in a certain pass, is determined by the value in the stencil buffer. S_1 in the stencil buffer means that the vessel tree is depicted in that pixel, S_2 corresponds to the soft-tissue data. If the stencil buffer is empty at a certain pixel position, then that particular pixel has not been filled with any information yet (background). Since the S_1, S_2 and empty regions are addressed individually, different blending and image processing operations can be performed to these regions (compare figures 4.4a and 4.4b). For instance, a spatial sharpening to enhance small details, and a temporal smoothing to reduce noise can be applied to the vessel region.

The fluoroscopy data that overlays the background, can contain some anatomical landmarks, which are relevant to the physician. The most important part of the fluoroscopy image, though, is to be found inside the vessel region, since the movement of the endovascular devices is supposed to be contained within this region. This hierarchy is reflected in the intensity and filtering of the fluoroscopy data stream. The

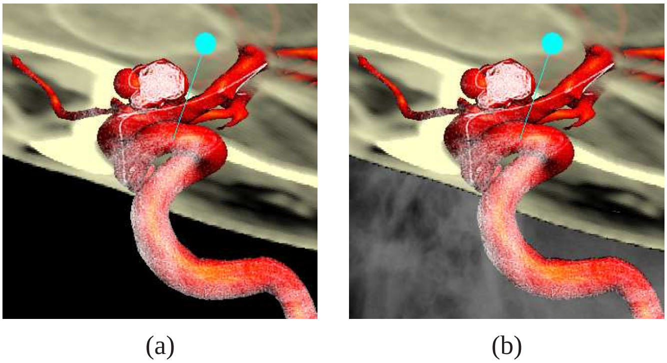

(a) (b)

Figure 4.4: *(a) In the first fluoroscopy overlay pass, the pixels that are labeled* S_1 *(vessel) in the stencil buffer, are treated. In this case, a sharpening filter was applied to the fluoroscopy data, before they were blended with the frame buffer content. (b) In the second pass, the pixels that were labeled as background in the stencil buffer, are processed. The fluoroscopy data is written without being sharpened, and the intensity is reduced.*

fluoroscopic information that overlays the soft-tissue slab could be suppressed, to reduce cluttering of information in this region.

4.3 Results and discussion

The augmented visualization, consisting of a mesh extracted from a 256^3 voxel 3DRA dataset, a Volume Rendered slab from a $256^2 \cdot 198$ voxel CT dataset and the fluoroscopy image stream, can be displayed at an average frame rate of 38 frames per second. All figures were measured on a Xeon 3.6 GHz machine with 2 GB of memory, and an nVidia QuadroFX 3400 graphics card with 256 MB of memory, using the datasets that are depicted in figure 4.5.

The benefit of the described method is the fact that it allows to visualize multiple partially overlapping datasets in a single fused image, while still preserving an easy interpretation of the data topology, morphology and the relations between the different datasets. Especially during clinical interventions it is very important that the information conveyed by the image can be understood at a single glance. By using the capabilities of the GPU and the efficient volume rendering techniques introduced in chapter 3, the fused visualization can be rendered at interactive frame rates, which is particularly important when real-time data is part of the fused image.

The main restriction of the fusing technique is the limitation of blending a single opaque mesh with a single volume rendered dataset. Brecheisen *et al*. [77] describe a powerful application of depth peeling to combine multiple volume rendered datasets, together with polygon surfaces (*i.e.*, meshes) and clipping volumes. However, how to combine this approach with the bricking and octrees of chapter 3 is currently an unsolved question. The absence of these accelerating techniques, together with the overhead introduced by the depth peeling presently still lead to frame rates that are unattractive for usage during clinical interventions.

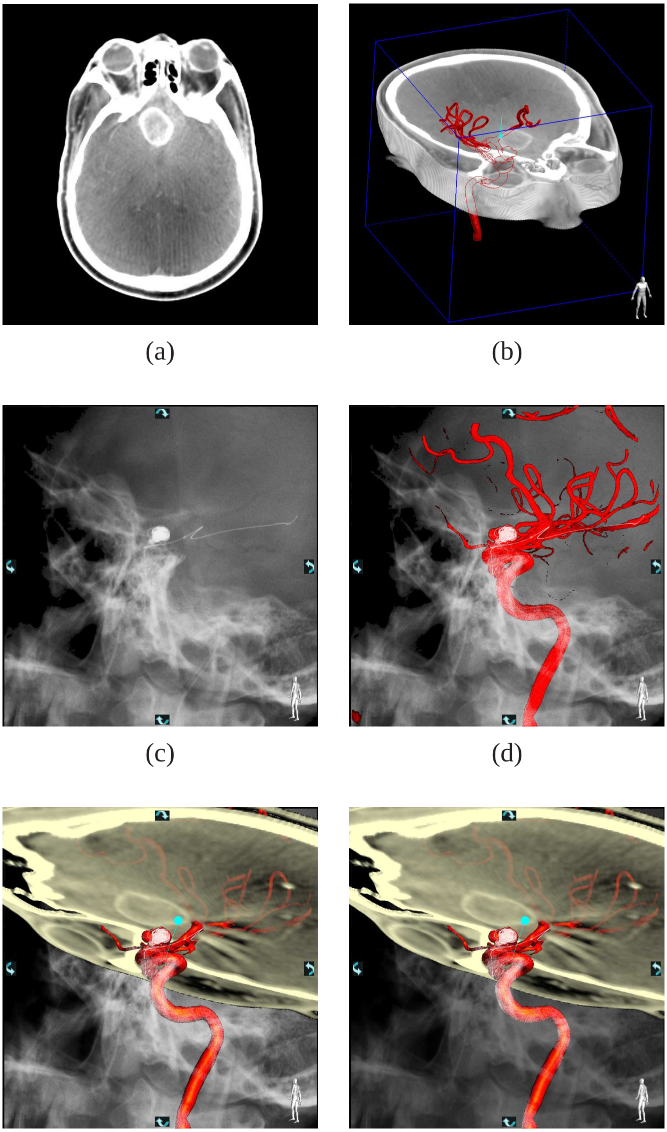


Figure 4.5: *(a) A CT image, clearly showing a tumor, (b) The CT dataset, registered with the 3DRA dataset, (c) a single frame from the fluoroscopy image stream, (d) the fluoroscopy image mixed with the vessel tree from the 3DRA dataset, (e) the fluoroscopy image, the 3DRA vasculature and a slab from the CT data, (f) the fluoroscopy image outside the 3DRA vessel tree is darkened.*

Chapter 5

Autostereoscopic visualization

This chapter is an extended revision of the following papers:

- Daniel Ruijters. Integrating Autostereoscopic Multi-View Lenticular Displays in Minimally Invasive Angiography. *In Proc. MICCAI 2008 workshop on Augmented Environments for Medical Imaging and Computer-Aided Surgery (AMI-ARCS),* September 10, 2008, New York (USA), pp. 87-94
- Daniel Ruijters. Dynamic Resolution in GPU-Accelerated Volume Rendering to Autostereoscopic Multiview Lenticular Displays. *EURASIP Journal on Advances in Signal Processing,* Volume 2009, Article ID 843753, 8 pages, 2009. doi:10.1155/2009/843753

5.1 Introduction

Stereoscopic images present a view on a 3D scene that adds a sensation of depth by showing slightly different images to the left and right eye of an observer. The additional depth impression enables a natural interpretation of the 3D data. Principally there are two approaches for conveying a stereoscopic image: time multiplexing and spatial multiplexing of two or more views. *Auto*stereoscopic displays allow a stereoscopic view of a 3D scene without the use of any additional external aids, such as goggles. Though two views are enough to create the impression of depth (after all, we have only two eyes), offering more views has the advantage that the viewer is not restricted to a fixed sweet spot, since there is a range of positions where the viewer will be presented with a stereoscopic visualization. As a consequence, multiple viewers can look at the same stereoscopic screen, without wearing goggles. Furthermore it is possible to ‘look around’ an object, when moving within the stereoscopic range, which aids the depth perception.

Multiview autostereoscopic displays can be regarded as three-dimensional light field displays [81, 82] (or four-dimensional, when also considering time). The dimensions are described by the parameters (x, y, ϕ), whereby x and y indicate a position on the screen and ϕ indicates the angle in the horizontal plane in which the light is emitted. The light is further characterized by its intensity and its color.

Present technological solutions to autostereoscopic displaying typically use either lenticular lenses or parallax barriers to achieve the stereoscopic effect. The parallax barriers displays use a grid that is placed at a small distance from the screen pixels, and blocks a different set of pixels when viewed from a different direction. The multiview lenticular display uses a sheet of lenses to spatially multiplex the views [83], and typically offers four to fifteen spatially sequential images. The advantage of the lenticular technique over the parallax barriers approach is the fact that all light produced by the screen is also emitted.

In this chapter we examine the autostereoscopic display in the context of medical applications. Especially during clinical interventions, interactive manipulation and high resolution visualization of medical data are two important and sometimes conflicting requirements. We discuss the properties of the lenticular autostereoscopic display with respect to perceived resolution, and we explore a dynamic balancing of interactive frame rates and highest possible resolution in direct volume rendered data.

5.2 State of the art

In 1838 Sir Charles Wheatstone developed a device, called the stereoscope, which allowed the left and the right eye to be presented with a different image (illustration or photograph), in order to create an impression of depth. Matusik and Pfister [84] presented a comprehensive overview of the various systems for stereoscopic visualizations, that have been developed over the time. The development of *auto*stereoscopic display devices, presenting stereoscopic images without the use of glasses, goggles or other viewing aids has seen an increasing interest since the 1990s [85–87].

The advancement of large high resolution LCD grids, with sufficient brightness and contrast, has brought high quality multiview autostereoscopic lenticular displays within reach [88]. A number of publications have investigated the image quality aspects of autostereoscopic displays. Seuntiëns *et al*. [89] have discussed the perception quality of lenticular displays as a function of white noise. Konrad and Agniel [90] describe the Fourier domain properties of the lenticular display, and they propose a pre-filtered sample approach. The effect of light that ought to be contributed to one particular view leaking into other views, which is called crosstalk, has been quantitatively investigated by Braspenning *et al*. [91] and Boev *et al*. [92].

The range of viewing positions, allowing the perception of a stereoscopic image, is mainly determined by the number of views offered by the display. Further, a higher resolution per view leads to less artifacts and improves the image quality. The required resolution of the LCD pixel grid can be established as the number of views times the resolution per view. Clearly, fulfilling both requirements demands very high resolution LCD pixel grids, which means that an enormous amount of pixel data has to be rendered and transferred to the display.

Several publications describe how the GPU can be employed to extract the data stream for the lenticular display from a 3D scene in an effective manner. Kooima *et al*. [93] present a two-pass GPU based algorithm for two-view head-tracked parallax barrier display. First the views for the left and the right eye are rendered, and in the subsequent pass they are interweaved. Domonkos *et al*. [94] describe a two-pass approach, dedicated for iso-surface rendering. In the first pass they perform the geometry calculations on the pixel-shader for every individual pixel, and in the second pass the shading is performed. Hübner and Pajarola [95] describe a GPU-based single-pass multiview volume rendering, varying the direction of the cast rays depending on their location on the lenticular screen.

The previous GPU-based approaches were dedicated render methods, working on the native resolution of the lenticular LCD grid. We present an approach that decouples the render resolution from the native LCD grid resolution, allowing lower resolutions, when higher frame rates are demanded.

5.3 The multiview lenticular display

The multiview lenticular display device consists of a sheet of cylindrical lenses (lenticulars) placed on top of an LCD in such a way that the LCD image plane is located at the focal plane of the lenses [96]. The effect of this arrangement is that LCD pixels located at different positions underneath the lenticulars fill the lenses when viewed from different directions, see figure 5.1. Provided that these pixels are loaded with suitable stereo information, a 3D stereo effect is obtained, in which the left and right eye see different, but matching information. The screen we used offered nine distinct angular views, but our method is applicable to any number of views.

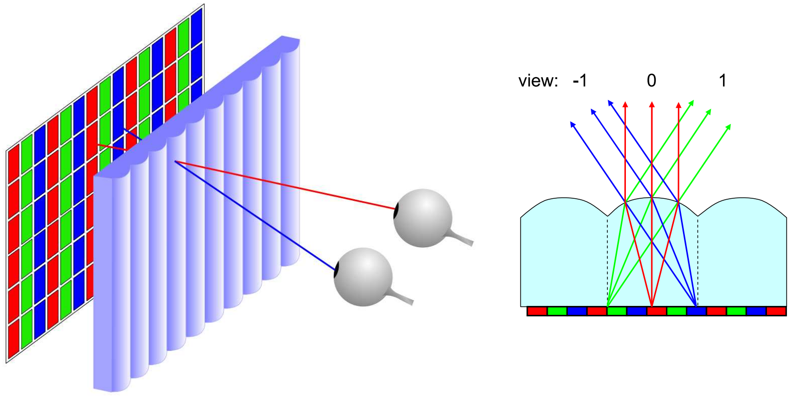


Figure 5.1: *The light of the sub-pixels is directed into different directions by the sheet of lenticular lenses.*

The fact that the different LCD pixels are assigned to different views (spatial multiplex), leads to a lower resolution per view than the resolution of the LCD grid [97]. In order to distribute this reduction of resolution over the horizontal and vertical axis, the lenticular cylindrical lenses are not placed vertically and parallel to the LCD column, but slanted at a small angle [83]. The resulting assignment of a set of LCD pixels, which is specified by the manufacturer, is illustrated in figure 5.2. Note that the red, green and blue color channels of a single pixel are depicted in different views.

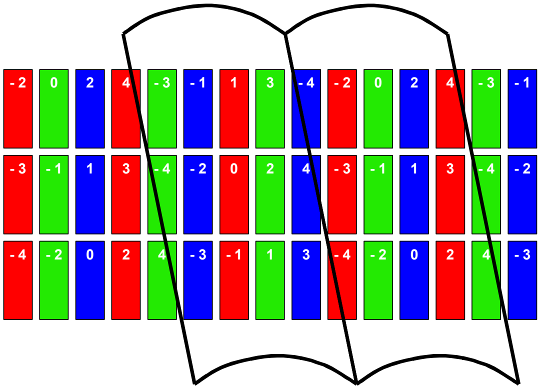


Figure 5.2: *The cylindrical lenses depict every sub-pixel in a different view. The numbers in the sub-pixels indicate in which view they are visible.*

5.4 The different angular views

We propose a two pass algorithm: First the individual views from the different foci positions are separately rendered to an orthogonal grid. In the second pass, the final output signal has to be resampled from the views to a non-orthogonal grid in the compositing phase (see figure 5.3). The processing power of the GPU is harvested for both passes. In order to maintain an acceptable frame rate, the resolution of the views can be changed dynamically.

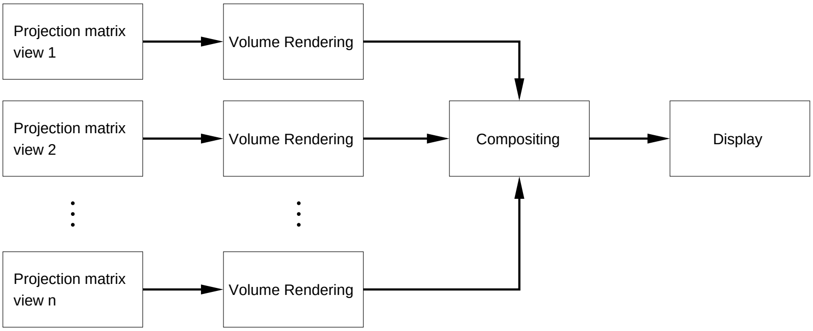


Figure 5.3: *The process of rendering for the lenticular display. Optionally, the rendering of the n individual views can be done in parallel.*

The frustums that result from the different focal spots, are illustrated in figure 5.4. The viewing directions of the frustums are not parallel to the normal of the screen, except for the center one. Therefore the corresponding frustums are asymmetric [98]. A world coordinate (x, y, z) that is perspectively projected, using such an asymmetric frustum, leads to the following view port coordinate $v(x, y)$:

$$v(x,y) = \left(\frac{(x - n \cdot d) \cdot f}{f - z} + n \cdot d, \ \frac{y \cdot f}{f - z} \right) \tag{5.1}$$

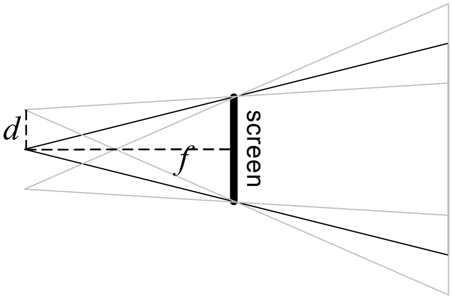


Figure 5.4: *The frustums resulting from three different view points.*

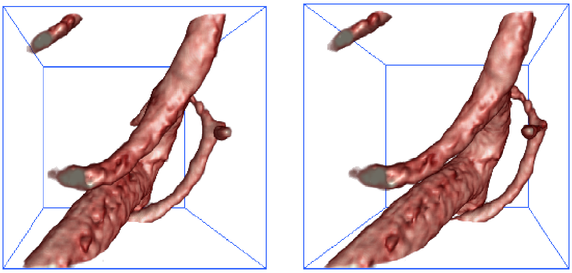


Figure 5.5: *The same scene rendered from the most left and most right view point.*

Whereby f denotes the focal distance, n the view number and d the distance between the view cameras. All parameters should be expressed in the same metric (*e.g.*, millimeters), and the origin is placed in the center of the view port.

Figures 5.4 and 5.5 illustrate the process of rendering the scene from focal spot positions with an offset to the center of the screen. After the projection matrix has been established based on equation 5.1, the scene has to be rendered for that particular view, using the techniques from chapter 3. All views are stored in a single texture, which we call *texture1*. In OpenGL, the views can be placed next to each other in horizontal direction, using the `glViewport` command. The location of a pixel in view n in *texture1* can be found as follows:

$$\vec{t} = \left(\frac{1}{2} + \frac{n}{\tilde{N}} + \frac{2p_x - 1}{2\tilde{N}}, \quad p_y \right) \tag{5.2}$$

whereby $\vec{t}$ denotes the normalized texture coordinate, $\vec{p}$ the normalized pixel coordinate within the view, and $\tilde{N}$ the total number of views. The view index n is here assumed to be in the range $\left[-\frac{\tilde{N}-1}{2}, \frac{\tilde{N}-1}{2}\right]$, as is used in *e.g.*, figure 5.2.

5.5 Resolution considerations

The maximum information density that can be conveyed by the lenticular display per view, is determined by the way the pixels of the LCD grid are refracted by the lenticu-

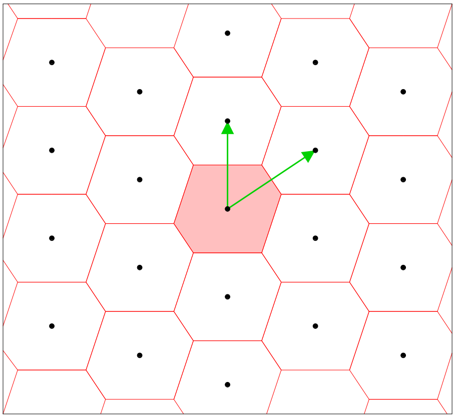


Figure 5.6: *A lattice (black dots) and corresponding Voronoi cells (red). The green vectors compose a possible basis V for this lattice. The Voronoi cell of a given lattice point is the set of points in $\mathbb{R}^N$ that are closer to this particular lattice point than to any other lattice point.*

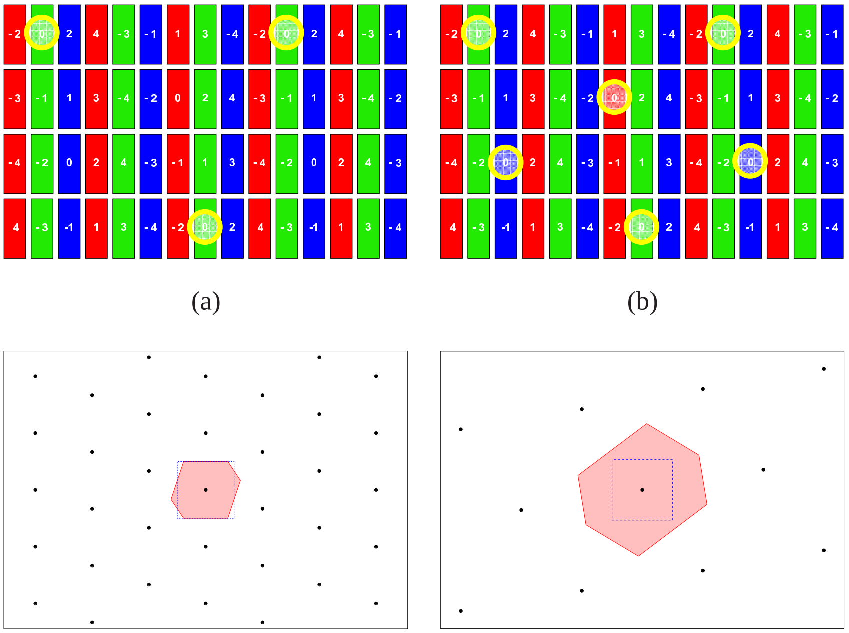
lar lenses. In modern lenticular displays, the lens array is slanted under a slight angle, which affects the distribution of the set of pixels that are diverted to a particular viewing angle. In figure 5.7a it is shown how the green sub-pixels, visible from the middle viewing position (view 0), are distributed over the LCD grid. Though the allocation of the sub-pixels over the grid is regular, it is not orthogonal. The sampling theory of multidimensional signals, described by Dubois [99], can be used to examine the frequency range that can be transmitted by a certain non-orthogonal grid. Especially the maximum view port size that does not lead to aliasing is of interest. When the resolution of the view port is too high, the compositing undersamples the view, and aliasing occurs. Though such views can be low-pass filtered to prevent aliasing, it is preferable to render them immediately at the optimal resolution, in order to keep the load on the scarce processing resources as low as possible.

The set of sub-pixels that are refracted to the same angular view can be considered to form a lattice. Let the vectors $\{\vec{v_1}, \vec{v_2}, ..., \vec{v_N}\}$ form a basis, not necessarily orthogonal, of $\mathbb{R}^N$. Then *lattice* $\Lambda \subset \mathbb{R}^N$ is defined as a set of discrete points in $\mathbb{R}^N$, formed by all linear combinations of vectors $\vec{v_1}, \vec{v_2}, ..., \vec{v_N}$ with integer coefficients.

In order to perform a Fourier transform of a signal, sampled on a lattice, the reciprocal lattice is required. The *reciprocal lattice* Λ^* of lattice Λ is defined as the set of vectors $\vec{y}$, such that $\vec{y} \cdot \vec{x}$ is an integer for all $\vec{x} \in \Lambda$. Let V be the matrix, whose columns are the representation of the basis vectors $\vec{v_1}, \vec{v_2}, ..., \vec{v_N}$ in the standard orthonormal basis for $\mathbb{R}^N$. Then matrix W, containing the basis vectors of the reciprocal lattice Λ^*, is determined by $W^T V = I$, with I being the $N \cdot N$ identity matrix.

The *Voronoi cell* of a lattice is defined as the set of all points in $\mathbb{R}^N$ closer to origin $\vec{0}$, than to any other lattice point, see figure 5.6. The basis V for a given lattice is not unique (*i.e.*, a lattice Λ can be described by several different basis matrices V). However, any basis for a certain lattice Λ delivers the same unique Voronoi cell.

Let the Fourier transform of a continuous multi-dimensional signal $u_c(\vec{x})$ with



-2	0	2	4	-3	-1	1	3	-4	-2	0	2	4	-3	-1
-3	-1	1	3	-4	-2	0	2	4	-3	-1	1	3	-4	-2
-4	-2	0	2	4	-3	-1	1	3	-4	-2	0	2	4	-3
4	-3	-1	1	3	-4	-2	0	2	4	-3	-1	1	3	-4

(a)

-2	0	2	4	-3	-1	1	3	-4	-2	0	2	4	-3	-1
-3	-1	1	3	-4	-2	0	2	4	-3	-1	1	3	-4	-2
-4	-2	0	2	4	-3	-1	1	3	-4	-2	0	2	4	-3
4	-3	-1	1	3	-4	-2	0	2	4	-3	-1	1	3	-4

(b)

(c)

(d)

Figure 5.7: *(a) The LCD pixel grid and the view that is associated with each sub-pixel. The green sub-pixels that are diverted to view 0 are circled. (b) All sub-pixels that are diverted to view 0 are circled, independent from their color. (c) The reciprocal lattice of the green sub-pixels for view 0. The Voronoi cell of the reciprocal lattice is indicated in pink. In blue the Nyquist frequency of the 1/3 orthogonal grid is indicated. Since the Voronoi cell does not cover the complete Nyquist frequency range, slight aliasing in the higher frequencies might occur. (d) The reciprocal lattice of the sub-pixel configuration of view 0, ignoring their color. Since the Nyquist frequency range (blue) is contained within the Voronoi cell (pink), there is no aliasing in the intensity image.*

$\vec{x} \in \mathbb{R}^N$ be defined as:

$$U_c(\vec{f}) = \int_{\mathbb{R}^N} u_c(\vec{x}) e^{-j2\pi\vec{f}\cdot\vec{x}} d\vec{x}, \qquad \vec{f} \in \mathbb{R}^N \tag{5.3}$$

The Fourier transformation of signal u_c sampled on lattice Λ is periodical, with lattice Λ^* as periodicity [99]:

$$U(\vec{f}) = \frac{1}{|\det V|} \sum_{\vec{r}\in\Lambda^*} U_c(\vec{f}+\vec{r}) \tag{5.4}$$

Consequently, if a signal that is not bandwidth limited within the Voronoi cell of lattice Λ^*, is sampled on lattice Λ, spectral overlap (*i.e.*, aliasing) occurs.

The sampling that occurs in the compositing phase can be examined, considering only one monochromatic primary color (red, green or blue), or can be evaluated for all colors together, see figure 5.7. The basis matrices V of the sample lattice can be established by taking two (non-linearly dependent) vectors between adjacent lattice points. The LCD pixel distance is used as a metric, which means that two neighboring sub-pixels (*e.g.*, red and green) have a distance of $\frac{1}{3}$ pixel. *E.g.*, for the color-independent lattice (figure 5.7b), we take the vectors $\vec{v_1} = (\frac{5}{3}, -1)^T$ and $\vec{v_2} = (\frac{4}{3}, 1)^T$. This delivers the following basis matrices V and their reciprocals W^T:

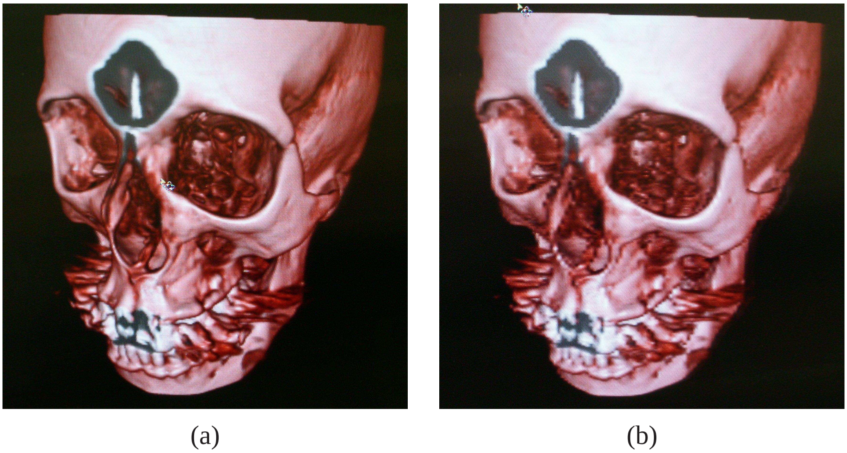
$$\begin{aligned} V_{mono} &= \begin{pmatrix} 3 & -1 \\ 0 & -3 \end{pmatrix} & V_{color} &= \begin{pmatrix} \frac{5}{3} & \frac{4}{3} \\ -1 & 1 \end{pmatrix} \\ W_{mono} &= \frac{1}{9}\begin{pmatrix} 3 & 0 \\ -1 & -3 \end{pmatrix} & W_{color} &= \frac{1}{9}\begin{pmatrix} 3 & 3 \\ -4 & 5 \end{pmatrix} \end{aligned} \tag{5.5}$$

The individual views are rendered on an orthogonal grid, and the Voronoi cell of an orthogonal lattice is a simple rectangle. The maximum resolution that can be visualized on the lenticular screen can be examined by fitting this Nyquist frequency rectangle range of the orthogonal grid on the Voronoi cell of the reciprocal lattice of the lenticular sample grid.

A logical choice for the resolution of the individual views, for a lenticular screen with 9 views, seems to be $\frac{1}{3}$ of the LCD pixel grid resolution in both directions. After all, this represents the same amount of information: 9 views with each $\frac{1}{3} \cdot \frac{1}{3} \cdot$ the amount of pixels of the LCD grid. We call this the 1/3 orthogonal grid. The Nyquist frequency rectangle of this resolution has been depicted on top of the Voronoi cell of the reciprocal lattice of the lenticular sample grid in figure 5.7. Looking at a single primary color channel (in figure 5.7a the green sub-pixels are used, but the lattice is the same for red and blue), it can be noted that the rectangle is not completely encapsulated within the Voronoi cell. This means that for monochromatic red, green and blue images there is a slight undersampling in certain directions, and aliasing might occur in the higher frequencies. If the lenticular lattice for a single view is considered, regardless of the colors of the sub-pixels, then the rectangle is completely contained within the Voronoi cell, see figures 5.7b and d. This implies that for grey colored images there is no aliasing when only the intensities are considered, but there might be some aliasing between the colors. In practise this behaviour resembles color dithering for real-world images. High frequent primary-colored structures (such as thin lines) may suffer from slight visible aliasing artifacts, though.

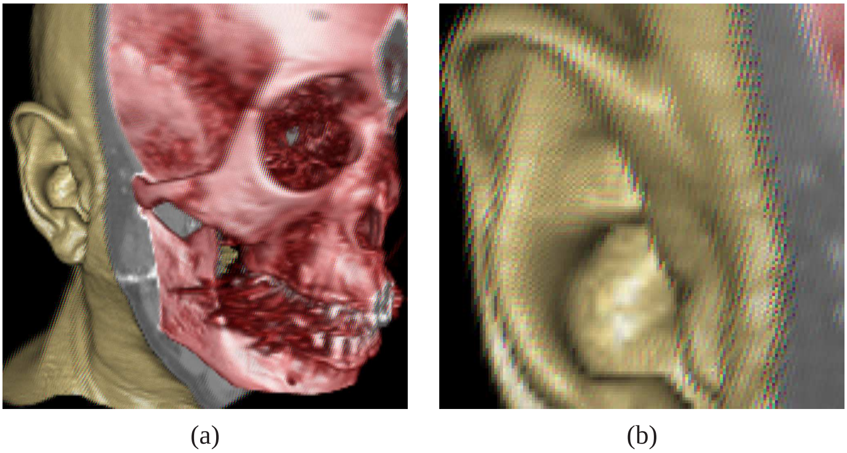
5.6 Dynamic resolution

As long as there are sufficient processing resources available, we use the 1/3 orthogonal grid as resolution of our views. This resolution provides a good trade-off between maximum detail and minimum aliasing, as described above. When the frame rate falls below a pre-defined threshold, the resolution of the individual views can be lowered, see figure 5.8. For sake of simplicity we use the same resolution for all views that contribute to a particular frame, but there is no technical reason imposing this. The resolution of a view can simply be changed by setting the view port to the desired



(a) (b)

Figure 5.8: *(a) The lenticular screen has been photographed, to show how a view is being displayed, rendered at the 1/3 orthogonal grid resolution. (b) The same view, but sampled at 0.375 · the resolution of the view in figure a). Though the downsampling is visible, the effect is less strong than might be expected. This can be contributed to the fact that the displaying process possesses a low-pass filter character, due to effects like crosstalk.*



(a) (b)

Figure 5.9: *(a) The raw output signal that is send to the lenticular display. Please note that the Moiré-like structures are not artifacts, but can be contributed to the interweaved sub-pixels, belonging to different views. (b) A zoomed fragment of the left image.*

size. The size of the off-screen buffer containing *texture1* is not changed; it is always kept at the maximum size needed.

Of course, lowering the view resolution does not guarantee that the desired minimum frame rate is achieved. This is mostly determined by the major bottlenecks in the 3D scene [55], see chapter 3. In cases where the major bottleneck is determined by the fragment throughput, the frame rate scales very well with the view port

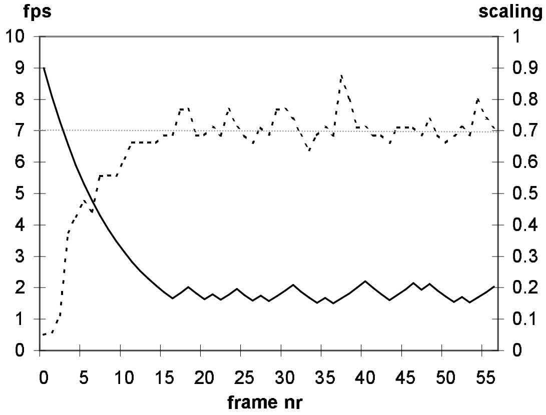


Figure 5.10: *Dashed line: frame rate in frames per second (fps). Solid line: view resolution scaling. The horizontal axis represents the consecutive frame numbers.*

size, and may increase significantly. When *e.g.*, the vertex throughput is the most important bottleneck, the frame rate is largely independent of the view port size.

Lower resolution views, correspond to smaller Nyquist rectangles in the frequency domain. For lower resolutions, the rectangle typically fits in the Voronoi cell of figure 5.7c, which implies that the view is oversampled by the compositing process. This corresponds to low-pass filtering the view at maximum resolution, which means that reducing dynamically the view resolution does not lead to aliasing artifacts, but merely to loss of detail. These details can be regained when the scene content is more static, and there is sufficient time to render the scene at high resolution.

To composite the final image, which will be displayed on the lenticular screen, the red, green and blue component of each pixel has to be sampled from a different view (see figure 5.2). The view number stays fixed all the time for each sub-pixel. Therefore this information is pre-calculated once, and then put in a static texture map, called *texture0*.

In the compositing phase, all the pixels in the output image are parsed by a GPU program. For each normalized pixel coordinate $\vec{p}$ in the output image, *texture0* will deliver the view numbers n that have to be sampled for the red, green and blue components. The respective views are then sampled in *texture1* according equation 5.2, using bi-linear interpolation, delivering the appropriate pixel value, see figure 5.9.

5.7 Results

Figure 5.10 shows the adaptive adjustment of the view resolution. The minimum desired frame rate was set to 7 frames per second in this case, which corresponds to rendering 63 views per second, since the lenticular display requires nine views to compose one frame. The measurements were performed using volume rendering of the data set depicted in figure 5.8, and involved advanced lighting. It consisted of $256^2 \cdot 200$ voxels (25 MB), while the output signal comprised $1600 \cdot 1200$ pixels. The

Volume description	Frames per second	Views per second
3DRX foot, $256^2 \cdot 200$ voxels (25 MB)	52.4	471
CT head, $512^2 \cdot 256$ voxels (128 MB)	19.2	173
3DRA vascular (sparse), 512^3 voxels (256 MB)	21.3	192
4D cardiac CT, 3 phases of $512^2 \cdot 333$ voxels (totally 510 MB)	13.6	123

Table 5.1: *The performance of GPU-accelerated volume rendering when generating images for the lenticular screen.*

resolution of the views was the resolution of the 1/3 orthogonal grid, multiplied by the scaling factor (right vertical axis) in both the x- and y-direction.

In order to characterize the performance of the GPU-accelerated volume rendering and compositing, several data sets were rendered at the 1/3 orthogonal grid resolution to an output window of 800^2 pixels. Nine views were rendered per frame, and the view size was 264^2 pixels. Table 5.1 shows the frame rates that were measured, using different datasets. All measurements were obtained, using a 2.33 GHz Pentium 4 system, with 2 GB RAM memory, and an nVidia QuadroFX 3500 with 256 MB on board memory as graphics card. It becomes clear that in order to achieve acceptable and interactive overall frame rates, a substantial number of views have to be rendered per second. Here we benefit considerably from the GPU acceleration and sophisticated optimization techniques that were described in chapter 3.

5.8 Clinical setup

We applied the presented approach to visualize intra-operatively acquired 3D data sets on a Philips 42” lenticular screen, which was mounted in the operation room (OR), see figure 5.11. The screen possesses an LCD panel consisting of 1920×1080 pixels (HD resolution), which are refracted into nine distinct views by the lenticular lenses. The orientation of the depicted 3D data set can follow in realtime the orientation of an X-ray C-arc system, which means that on the lenticular display the 3D data set is visualized from the same viewing angle as the viewing incidence on the patient in the real-time X-ray image, see section 6.6. This approach allows to reduce the X-ray radiation, since the physician can choose the optimal orientation to acquire X-ray images without actually radiating. Further it improves the interpretation of the live projective 2D X-ray image, which is presented on a separate display, since the 3D data on the stereoscopic screen (which is in the same orientation) gives a proper depth impression through the stereovision of the lenticular screen. The fact that the clinician is not limited to a single sweet spot (a single viewing location where the stereoscopic effect can be perceived), makes the multi-view display particularly suitable for this environment, since the clinical intervention demands that an operator can be positioned freely in the range close to the patient.

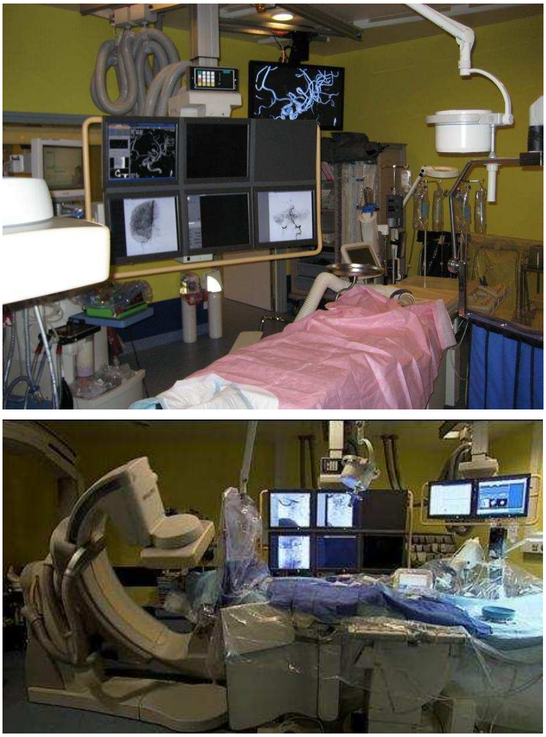


Figure 5.11: *Top: the 42” lenticular screen (top right display) in the operating room. Bottom: the X-ray C-arc system in a clinical intervention.*

The ability to visualize and manipulate the 3D data interactively is of great importance in the analysis and interpretation of the data. Interactivity, in this context, means that the frame rates of the visualization are sufficient to provide direct feedback during user manipulation (such as rotating the scene). When the visualization’s frame rate is too low manipulation becomes very cumbersome. Five frames per second are often used as a required minimum frame rate.

Especially the cerebral vasculature consists of many curved vessels, see figure 5.12. From a single X-ray image it is impossible to interpret the curvature perpendicular to the viewing plane (the curvature in the z-direction of the image). But even looking at a 3D rendered image on a 2D plane (*i.e.*, conventional monitor), it is often very difficult to estimate the in-plane curvature without rotating the vessel tree. Rotating a 3D scene is not a problem when sitting behind a desktop computer, but during a clinical intervention the performing clinician is primarily occupied with the medical procedure, and interaction with sterile computer input devices (which are available in the OR) is an additional task, demanding focus. The stereoscopic image allows to interpret the 3D shape, including the in-plane curvature, in a single glance without any additional input interaction, and therefore reduces the mental stress on the clinician during the intervention.

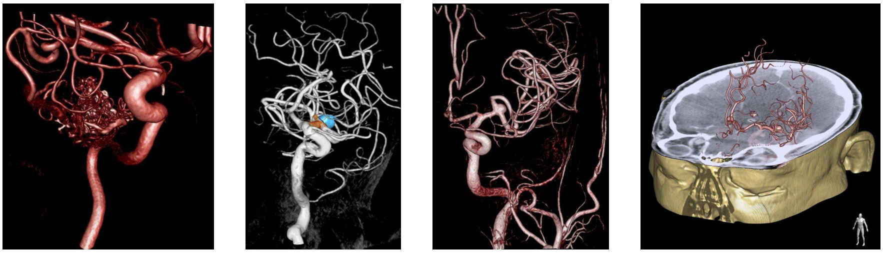


Figure 5.12: *Volume rendered vascular 3DRA images. From left to right: brain arteriovenous malformations, virtual stenting and aneurysm (blue), another neuro vessel tree with aneurysm, and vasculature mixed with soft-tissue data.*

5.9 Conclusions

In this chapter a method for accelerated rendering to multiview lenticular displays has been presented. Due to the GPU-acceleration, together with the adaptive adjustment of the intermediate view resolution, interactive frame rates can be reached, which allows intuitive manipulation of the rendered scene. Since both the volume rendering and the compositing take place on the graphics hardware, the requirements for the other components of the PC system are rather modest. Thus the realization of the proposed high performance system can be very cost effective, apart from the costs of the lenticular screen.

Feedback from clinicians indicates that there certainly is a perspective for clinical added value. Orthopedic surgery is suggested as another application area that could benefit from the multi-view stereoscopic display. For better integrated usage, the display should be mounted on the same ceiling suspension with the other (2D) displays. Also the integration of the live fluoroscopy image in the 3D scene would be highly appreciated. Both the fact that the clinicians do not need to wear any additional glasses, and are not limited to a sweet spot, as well as the fact that large data sets can be manipulated interactively, make this method very suitable for a clinical interventional environment. In order to give an impression of the added value of the depth perception provided by the stereoscopic 3D effect, figures 5.13 and 5.14 show anaglyph images of a carotid artery.

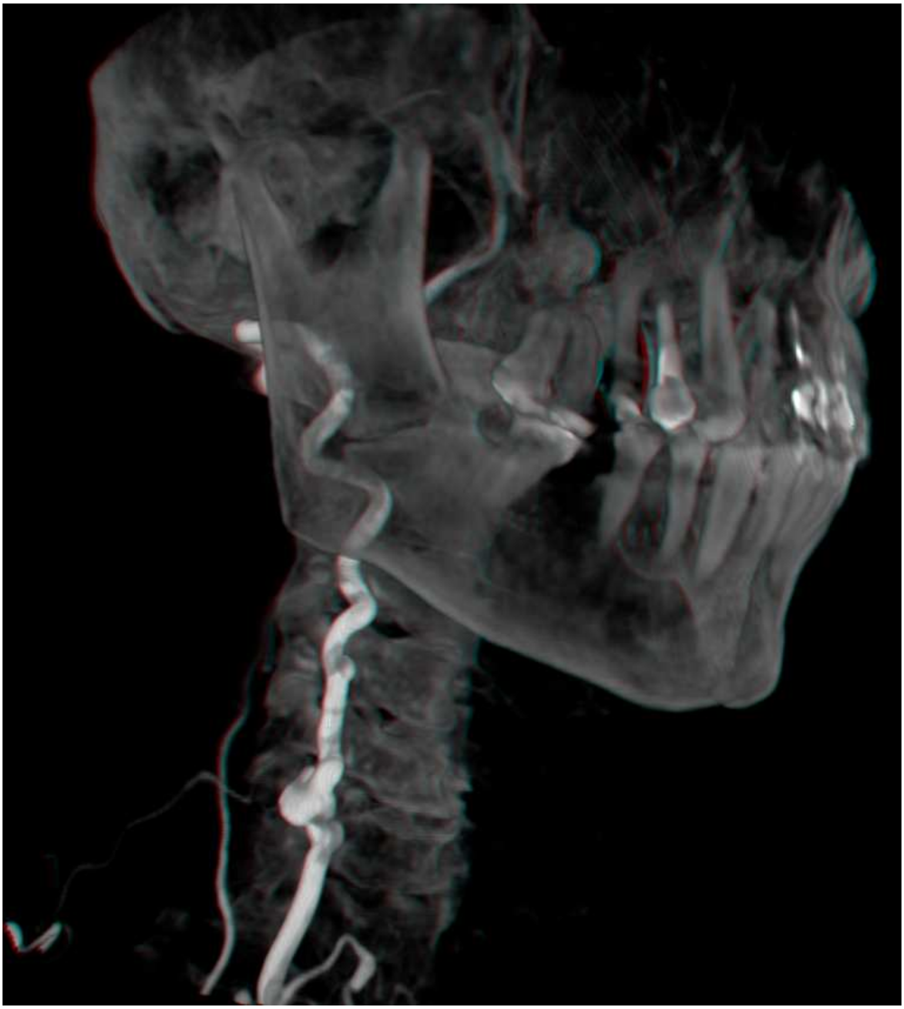


Figure 5.13: *Anaglyph image of a 3DRA reconstruction showing a aneurysms in a carotid artery. The stereovision effect is obtained by using anaglyph glasses: the left eye glass should be red and the right eye blue.*

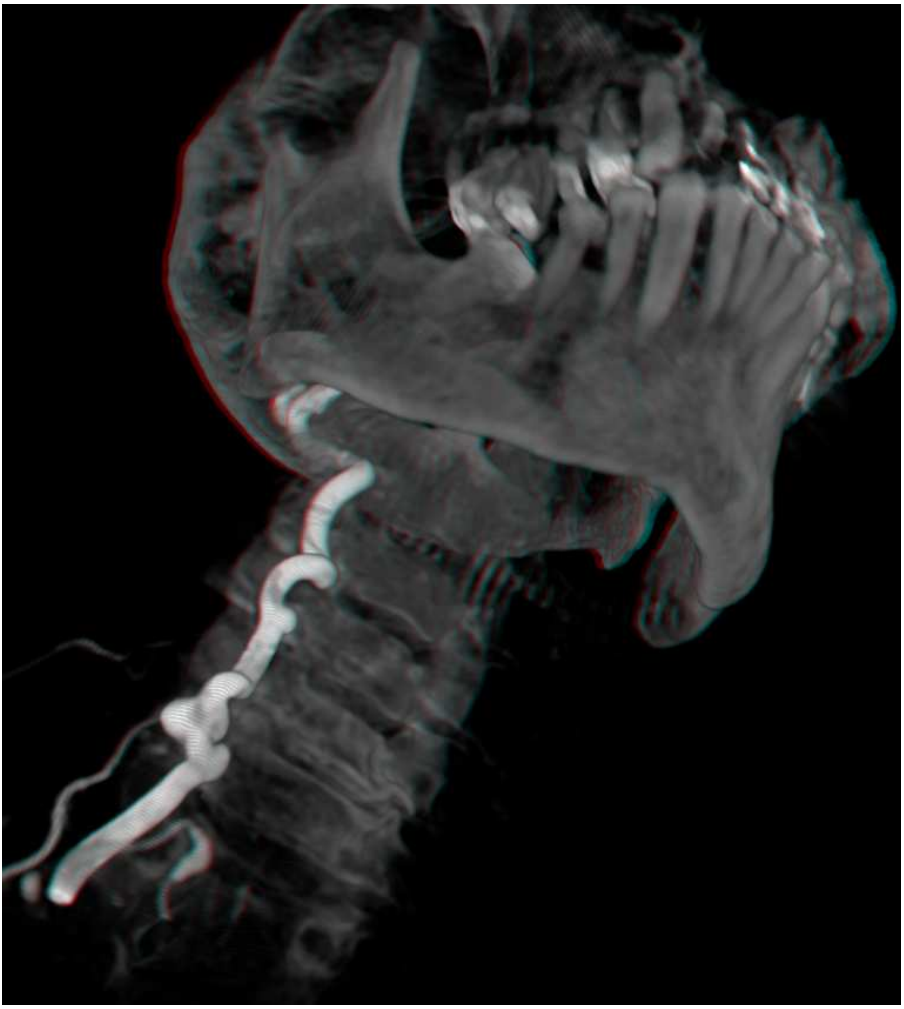


Figure 5.14: *Anaglyph presentation of the same dataset as in figure 5.13, but shown from a different angle.*

Part II

Intra-interventional Registration

Registration is the process of spatially aligning two image datasets, such that the corresponding anatomical morphology in both datasets overlaps. The application of image registration during a clinical intervention imposes constraints on the algorithms and their possibilities to interact with the user.

The major restriction on the algorithm is the available computation time. In spite of Moore's law, the computation time for registration algorithms tends to be rather considerable and ranges from minutes to hours. For usage during clinical interventional treatment, this amount of time is not available.

Many registration methods benefit from interaction with the user. They either need user input to perform the registration task, or benefit from user driven initialization, which can shorten their computation time. The possibilities for user interaction during treatment, however, are limited. The clinician is typically standing in the intervention room at the patient's table side, and input devices are often less accurate and not always easy to use (*e.g.*, due to sterility requirements) than compared to desktop usage. Also, the display is frequently one or more meters away or at an awkward angle. Furthermore, the clinician is focussed on the clinical procedure and there is only limited time and attention for computer interaction.

The registration methods which were developed and applied took these constraints into consideration. In the following chapters first the relevant state of the art is introduced (chapter 6), a fast approach to registration algorithms by using the graphics hardware is described in chapter 7, and finally chapter 8 introduces a method especially designed for vascular 2D-3D registration.

Chapter 6

Registration algorithms

6.1 Introduction

The objective of a registration algorithm is to establish a spatial mapping between two image datasets. Typically, one of the datasets is assigned to be the reference data and is not modified, while the other dataset is handed the role of floating data, and is spatially manipulated to match the structures in the reference data. Most registration algorithms consist of three components;

- A spatial transformation, delivering a mapping between the coordinate space of the floating image and the coordinate space of the reference image.

- A similarity measure, indicating the quality of a given spatial transformation by expressing the resemblance of the reference data and the spatially transformed floating data in quantifiable terms.

- An optimization algorithm, which iteratively searches the optimum of the similarity measure. The search space consists of the multi-dimensional control variables of the spatial transformation.

It should be noted that this decomposition does not apply to all registration approaches known in the literature. Demons or optical flow algorithms, for example, integrate the deformation and optimization components in a single combined scheme [100, 101].

The image data may originate from different imaging modalities, and is not restricted to only two dimensional images, but may also refer to higher dimensional data (*e.g.,* 3D or 4D). It is also possible to register images of different dimensionalities, which results in a spatial mapping that is a one-to-many mapping. In this work we focus on 3D-3D registration of multi-modal data and 2D-3D registration between 2D X-ray projection images and 3D voxel data, whereby we pay especially attention to the calculation time of the algorithms.

6.2 Spatial transformation

6.2.1 Affine transformations

Spatial transformations can be divided into two classes; affine and non-affine transformations. Affine transformations consist of a linear transformation and a translation, and can be expressed by:

$$\vec{x} \mapsto A\vec{x} + \vec{b} \tag{6.1}$$

In homogeneous coordinates the affine transformation can be captured in a single matrix (for 3D space, a 4×4 matrix):

$$\begin{bmatrix} \vec{y} \\ 1 \end{bmatrix} = \begin{bmatrix} A & \vec{b} \\ 0, \ldots, 0 & 1 \end{bmatrix} \begin{bmatrix} \vec{x} \\ 1 \end{bmatrix} \tag{6.2}$$

An important property of affine transformations is the fact that there are relatively few parameters to describe the transformation, which allows affine registrations to be calculated relatively fast, compared to non-affine registrations, due to the limited dimensionality of the parameter space. Since all spatial elements (*i.e.*, pixels or voxels) undergo the same transformation, the computation of an affine registration is relatively robust. After all, a false transformation leads to a discorrespondence for (almost) all spatial elements.

Rigid transformation is the most commonly used sub-class of affine transformations. Rigid transformations only consist of a rotation and translation, and thus do not possess any scaling or skewing. Rigid transformations represent the non-deforming displacement of subjects.

6.2.2 Non-affine transformations

Non-affine (or elastic) transformations are applied to describe local deformations. It is possible to divide non-affine transformations into three subclasses:

- Mesh based transformations. A mesh of non-uniformly distributed control points is established. Typically the control points are placed on some features that are extracted from the floating image, *e.g.*, gradients, ridges, segment boundaries, *etc*. The mesh is then projected on the reference image, and the control points are manipulated according to some criterium (*e.g.*, optimizing the similarity measure, or finding corresponding patterns in the neighborhood of the control point). The deformation of the image elements between the control points is then driven by the displacement of the control points. Most commonly the mesh is triangulated, and the elements within a triangle (or tetrahedron in 3D) are linearly interpolated.

- Uniform grid based transformations. A uniform grid of control points drive a linear combination of a class of basis functions [102]. Common choices for the basis functions are orthonormal wavelet or Fourier bases [103], thin plate spline models using radial basis functions [104], elastic body splines [105], or B-splines [106, 107]. The latter have the advantage of local support, which allows a reduction of computation time. The advantage of the uniform grid approach is the fact that there is no dependency of the robust extraction of feature points. The disadvantage is the fact that in order to describe fine deformation structures, a huge number of control points is needed. There are, however, techniques that allow to 'switch off' control points dynamically during the registration (*e.g.*, when the derivative of the similarity measure is very low for a certain control point), and thus reduce the dimensionality of the parameter space, see *e.g.*, [108].

- Direct mapping. These methods map each element in the floating image directly on the reference image [100, 101]. In every iteration a displacement is established for each image element of the floating image, and then typically a regularizer is applied to the displacement field. Omitting the regularization would yield an ill-posed problem [102].

6.3 Similarity measure

The similarity measure indicates how well the content of the reference image and the spatially transformed floating image overlap. Similarity measures can be grossly subdivided into two classes: intensity driven approaches and feature driven approaches. As a rule of thumb feature driven approaches are faster than intensity based methods, since the amount of information needed to describe the features is usually significantly smaller than the amount of original image information. Feature driven approaches are usually targeted at registering a specific anatomical structure. Their typical weak points are the dependency on robust extraction of the desired features from the image, and their reduced suitability for non-affine registration due to lack of per-point information, especially outside the feature areas. In intensity based approaches, information of the entire image space contributes to the similarity measure, which aids in developing reliable non-affine registration methods.

The choice of the similarity measure depends very much on the subject to be registered (*i.e.*, which anatomy, region of interest, *etc*.), and on the types of images (*i.e.*, modality, dimensionality, *etc*.) to be used. Popular intensity driven similarity measures are sum of squared differences (SSD) and mutual information (MI) [109].

An important aspect for intensity driven similarity measures is image interpolation. Images (in any dimensionality) are composed of a set of discrete spatial elements. However, the spatial transformation is typically defined as a continuous mapping $\mathbb{R}^N \mapsto \mathbb{R}^M$, with N and M denoting the dimensionality of the floating and reference image space, respectively. This means that in the floating image intensities 'in-between' the discrete spatial element positions have to be determined. According to the Shannon-theorem, any bandwidth limited signal can be reconstructed from a set of discrete samples using `sinc` interpolation, provided the sample rate fulfills the Nyquist criterion. Unfortunately, this interpolation is computationally very expensive. Unser *et al*. has demonstrated that higher order B-spline interpolation forms a good alternative [110]. In section 7.2 it is discussed how this can be mapped efficiently on the processing capabilities of the GPU.

In SSD for all spatial elements in the reference image, the difference between intensities of the reference element and the corresponding element in the spatially transformed floating image are calculated. The sum of the squares of these differences, divided by the number of processed elements, yields the SSD similarity measure. Since SSD is very sensitive to large intensity differences, it is primarily suited for single-modality registration. The least-squares form of SSD makes it computationally very attractive for optimizers that use the derivatives of the similarity measure and assume a quadratic form, such as Quasi-Newton and Levenberg-Marquardt optimizers.

MI is a concept known from information theory, and expresses the statistical relation between two given sets of ordered data, and can be formulated in terms of entropy. The entropy of a random variable X, consisting of a number of events x, with probability distribution $p(x)$, is $H(X) = -\sum_x p(x)\log(p(x))$. The entropy is maximal when the probability is uniformly distributed, *i.e.*, each event has equal probability. Its minimal value of zero is reached when $p(x) = 1$ for one event and zero for the others.

MI can be expressed, using the joint entropy $H(A, B; \tau)$ and marginal entropies $H(A)$ and $H(B; \tau)$, whereby A denotes the reference image, B the floating image, and τ the spatial transformation:

$$\begin{aligned} I(A, B; \tau) &= H(A) + H(B; \tau) - H(A, B; \tau) \\ &= \sum_{a \in A} \sum_{b \in B} p(a, b; \tau) log \left(\frac{p(a, b; \tau)}{p(a)p(b; \tau)} \right) \end{aligned} \tag{6.3}$$

The joint probability $p(a, b; \tau)$ can be easily obtained by determining the joint histogram of the reference image and the transformed floating image. The joint histogram is a two dimensional histogram, whereby the one axis represents the intensities of the reference image and the other axis the intensities of the floating image. The joint histogram bins represent the occurrence of their particular intensity pair for the entire set of spatial positions in the image space. In order to be robust for noise and to speedup calculations, the histogram bins usually represent a range of intensities. Commonly the whole intensity range of an image is divided into 8 to 64 evenly spaced bins.

In its simplest form, it is not possible to determine the derivative of the MI similarity measure with respect to the spatial transformation parameters. In order to overcome this limitation several approaches have been proposed; The Parzen window distributes intensities over several adjacent bins [111], while the (generalized) partial volume approach pairs the spatial elements in the reference image to range of elements in the floating image according to a chosen kernel [109, 112]. Dirk Loeckx has demonstrated that all mentioned variations of MI can be captured in a single comprehensive framework, which can be expressed in a single formula [113]. This framework also makes it insightful how the derivative can be obtained, and why it is not available for the simplest form of MI.

The iterative closest point (ICP) algorithm or a variation thereof is being used in many feature-based methods. ICP relies on minimizing the sum of minimal distances between the feature points in the reference and projected image. It can be efficiently calculated by determining the distance transformation of the features in the reference image as a preprocessing step. The ICP is then obtained in the iterations by projecting the transformed features of the floating image on this distance transformation, and summing the sampled distances [114].

6.4 Optimization

Finding the parameter set that delivers the optimal transformation according to the chosen similarity criterion can be a challenging task. The dimensionality of the parameter space can be enormous when there are many control points, and even rigid registration already possesses a six dimensional parameter space. In this huge space there are often many local optima. Establishing the global optimum within the time frame that is available during interventional treatment is not a possibility given current computation resources (even when using a workstation cluster). Therefore local optimization strategies are used. The global optimum (or a reasonable approximation) can be found by a local optimization strategy, provided the initialization of the parameter configuration is within a sufficient monomodal range of the similarity function.

The local optimization strategy can advance much faster when analytical derivatives of the similarity measure are available. Gradient descent, Quasi-Newton and Levenberg-Marquardt are examples of such optimizers [115]. An example of an optimization strategy that does not require derivative information is Powell's method [116]. This method searches the parameter space by performing a line search in every direction of an orthonormal basis in each iteration. The basis can be adapted between the iterations. It may be obvious that this approach takes considerably more time, but is also somewhat less likely to get trapped in small local optima.

Another group of strategies that do not demand derivative information are the probabilistically based algorithms, such as simulated annealing [117] or controlled random search [118]. These methods sample the parameter space in each iteration around an intermediate set of best samples according to a stochastically driven strategy. These types of algorithms are even more robust to local optima than Powell's method, but their results are not necessarily reproducible. Furthermore, they typically need a lot of samples before the optimum is found with a reasonable accuracy. Stochastic algorithms can be implemented to perform a local search, starting from an initial configuration, or a global search with a random initialization.

Independent from the chosen optimization strategy, a good approach has proven to be the following: start the registration with a number of iterations in low resolution with few control points to find large deformations, and gradually refine the registration by moving to higher resolutions and more control points. The low resolution images represent a coarse scale, and are very suitable to find large magnitude low frequency deformations. Also the iterations in low resolution can be computed relatively fast. Unser *et al*. have shown how the image pyramids that contain the image in different resolutions can be efficiently obtained [119].

6.5 Validation

The objective of validation is the assessment that the registration algorithm fulfills its clinical purpose [120]. The validation of the algorithm should always be conducted in the context of its application. Many boundary conditions are determined by the application; Which imaging modalities are being used? What is the typical spatial resolution, dimensionality, signal to noise ratio, field of view, *etc*., of the images? Which anatomy is being imaged? Is manual interaction with an expert user an option? Which computation times are acceptable? Which type and amount of non-ridged deformations are to be expected? *Etc*., *etc*., *etc*. When the context of the application has been established, a set of representative datasets has to be gathered. Usually, for initial development a rather limited database is being used, but for proper validation a representative database with sufficient variability is needed. There are several aspects that are evaluated during the validation of a registration algorithm. The properties that most commonly are investigated are: effectiveness, robustness and duration (computation time) [121].

The ground truth is the transformation that perfectly describes the spatial relation between two image datasets. The effectiveness of an algorithm describes how close the algorithm can approach the ground truth. It can be evaluated qualitatively by overlaying the reference image data with the transformed floating data, and visually inspecting the result for deviations. The effectiveness can be investigated quantitatively by the residual error, which measures the deviation of the transformation yielded by the registration algorithm from the ground truth. Since the ground truth usually is unknown, a gold standard is used, approximating the ground truth as closely as possible. A gold standard can be obtained by *e.g.*, using simulated data, using expert identified landmarks [122], or recording extrinsic data using a controlled environment (*e.g.*, fiducial markers or optically tracked probes) [120], *etc*. For rigid transformations the residual error can be expressed as the translational and rotational difference between the registration result and the gold standard. For elastic registration it can be quantified by the root mean square (RMS) error between all points in the registration and gold standard deformation field:

$$e = \sqrt{\frac{1}{|V|}\int_V |\tau(x) - \tilde{\tau}(x)|^2 \, dx} \tag{6.4}$$

whereby e expresses the RMS error, V the region of interest, τ the spatial deformation yielded by the registration, and $\tilde{\tau}$ the gold standard.

A quantitative indicator for the robustness of a registration algorithm is its capture range. It is defined as the range of initial poses of the floating data that still deliver a registration of sufficient quality. Or in other words: how far can the floating image be initialized from the ground truth, without causing the registration algorithm to fail. When a gold standard is available, a successful registration can be defined as a residual error smaller than a predetermined threshold. In the absence of a gold standard the success of a registration can be judged by an expert.

Within this work, several validation experiments were conducted. The intrinsic inaccuracies that are introduced in GPU-based elastic registration are described in section 7.2.3, and the associated computation times are investigated in section 7.4.2. Chapter 8 discusses the effectiveness, robustness and duration of vesselness-based registration. The capture range and computation times are evaluated using real clinical data, while the residual error is measured using simulated data to obtain a gold standard. Chapter 9 explores the robustness and clinical feasibility of intra-operative registration of 3DRA to CT data and 3DRA to MR data.

6.6 Machine-based 2D-3D Registration

6.6.1 Introduction

In the previous sections the registration was driven by the image content, using an image-based registration algorithm. There are numerous image-based 2D-3D registration methods known in the literature for registering fluoroscopy images to either CT or MR images [123–129]. These algorithms, however, take a considerable amount of time to compute. Further they need a sufficient number of landmarks to be present in the 2D fluoroscopy image, which is not necessarily always the case (*e.g.*, due to the absence of contrast medium).

In this section, a fundamentally different approach is presented: machine based registration. With the introduction of motorized calibrated C-arm X-ray angiography, 3D reconstruction of the vasculature came within reach. Since such 3DRA datasets are obtained with the same apparatus as the 2D fluoroscopy data, it is possible to calculate a spatial mapping, based on the state of the geometry (viewing incidence angles, source-detector distance, *etc*.) and calibration data, provided that there was no patient motion between the acquisition of the 3DRA data and fluoroscopy data [130–132]. This method also allows to obtain a registration, when there are insufficient landmarks present in the images (*e.g.*, due to the absence of iodine contrast medium in the fluoroscopy images). A further advantage of machine-based registration is the fact that it can be computed in realtime. Machine-based registration and image based 2D-3D registration have been compared by Baert *et al*. [133]. They concluded that pre-calibrated machine based registration is highly accurate as long as there is no patient motion, and a registration error of less than 0.5 mm was observed in their experiments. Image based registration, though slightly less accurate, proved to be more robust for (limited) patient motion. A method for determining the C-arm incidence based on tracking a fiducial was proposed by Jain *et al*. [134], who reported a mean accuracy of 0.56 mm in translation (standard deviation: 0.33 mm) and 0.33° in rotation (standard deviation: 0.21°), using a fiducial of $3 \times 3 \times 5$ cm. We, however, do not use any fiducials. We only use the information concerning the geometry state, as is provided by the C-arm system.

6.6.2 Calibration

In a CT gantry the X-ray tube and detector array are rotating in a large rigid ring. This leads to a stable, easily measurable and predictable relative position of the X-ray tube and detector array, a prerequisite for tomographic reconstructions. The X-ray

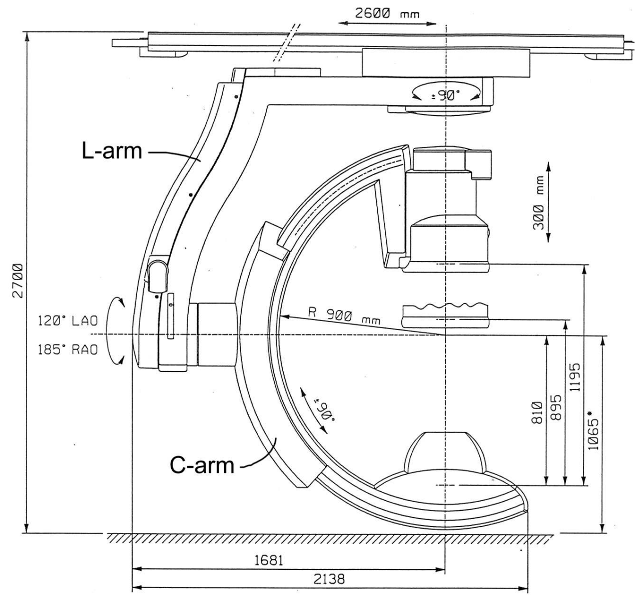


Figure 6.1: *The mechanical components of the X-ray C-arm.*

C-arm is of a completely different mechanical construction with only a few joints connecting the different parts, see figure 6.1. Especially the fact that load of the weight of the C-arm (including the X-ray tube and detector) relative to the L-arm differs depending on its angulation and rotation leads to variation in the bending of the C-arm. Luckily the bending for a given pose proves to be geometrically very reproducible (sub millimeter).

In order to perform 3D tomographic reconstructions using C-arm equipment (see section 2.4.3), calibration methods have been developed to determine the actual position of the X-ray tube and detector for a given set of mechanical propeller, roll and L-arm angles [135, 136]. The original calibration methods were developed for the image intensifier systems, see section 2.3, and corrected also the pincushion deformation that was caused by the image intensifier tube. Modern flat detector systems do not suffer from this image deformation, and the calibration only needs to correct for mechanical bending due to gravity.

The parameters that are measured in the calibration procedure are the deviation of the focal spot location, the deviations of the iso-center location and orientation. All parameters are measured for varying C-arm poses. The focal spot location is determined by mounting a grid on the detector at a fixed distance. This grid contains a number of bronze markers. The deviation from the expected position determines the actual location of the focal spot [136].

The location and orientation of the iso-center axes are obtained by placing a regular polyhedron (dodecahedron) with marker balls on the corners at the iso-center of the C-arm, and performing a circular movement with the C-arm. During the movement X-ray images are obtained, and the center of the marker balls in the images

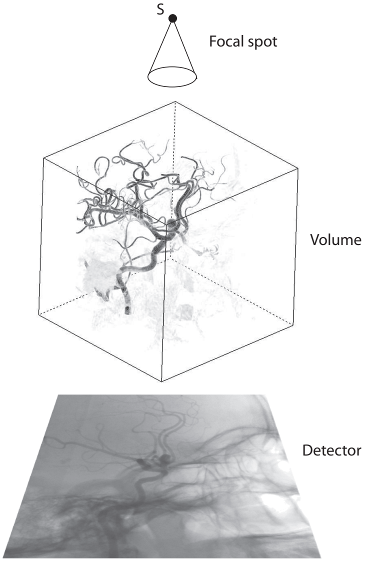


Figure 6.2: *The projection of a point in 3D space on the detector grid is determined by the position of the focal spot and the position and orientation of the detector.*

are detected. The detection of the markers, which are distributed nearly isotropically in the image, is a fairly easy task, since they absorb considerably more X-ray dose than any other object on the image. On the detected markers the geometrical shape of the dodecahedron is fitted, using an L2-error norm [136]. All calibration steps are fully automatic, and need only to be performed at installation and after system disturbances.

6.6.3 Projection

A common part in 2D-3D registration algorithms is the projection of a point in 3D space on the X-ray detector plane. In order to perform this projection, a 4×4 matrix M is defined, such that $\vec{p} = M \cdot \vec{v}$, whereby $\vec{p}$ and $\vec{v}$ are homogenous coordinates. Vector $\vec{v}$ is then a coordinate in the 3D CT space, and vector $\vec{p}$ a coordinate on the detector grid (the z value of $\vec{p}$ is simply disregarded).

Understanding the projection in mathematical detail is best accomplished by considering the components of the transformation chain separately. Matrix M can be decomposed into two matrices:

$$M = P \cdot R, \tag{6.5}$$

whereby P is the perspective transformation defined by the position of the focal spot (f_x, f_y, f_z), the position of the center of the detector (c_x, c_y, c_z), and the detector

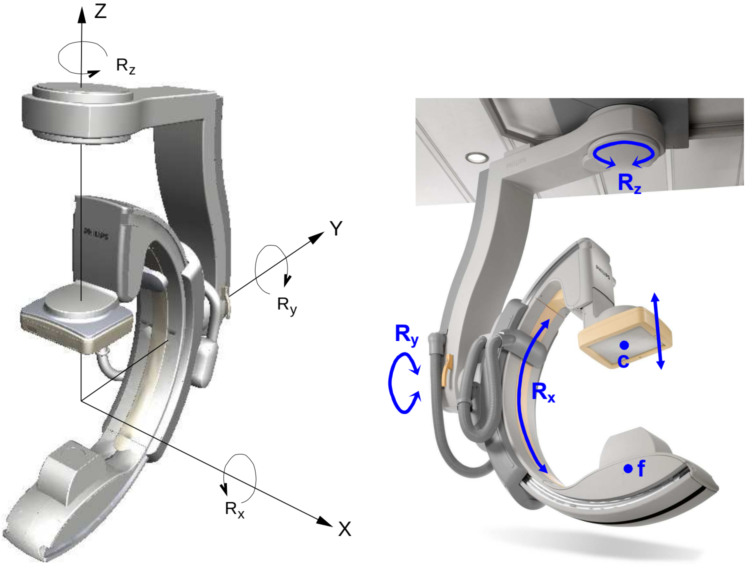


Figure 6.3: *The X-ray angiography C-arm system's geometry, and its degrees of freedom.*

dimensions (d_x, d_y), see figure 6.2.

$$P = \begin{pmatrix} 2/d_x & 0 & 0 & 2(f_x - c_x)/d_x \\ 0 & 2/d_y & 0 & 2(f_y - c_y)/d_y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/(f_z - c_z) & 1 \end{pmatrix} \tag{6.6}$$

Matrix R describes the viewing incidence of the X-ray C-arm geometry, and is determined by the L-arm (R_z), the propeller (R_y) and the roll rotation (R_x) of the C-arm, and can be expressed as $R = R_x \cdot R_y \cdot R_z$, see figure 6.3. Note that the order of the matrix multiplications is given by the mechanics of the C-arm system.

$$R = R_x \cdot R_y \cdot R_z =$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos\gamma & -\sin\gamma & 0 & 0 \\ \sin\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The angles α, β and γ are corrected for gravity effects, using the calibration data, as explained in the previous section. Now that matrix M is established, expressing the relation between the 3D space and the detector space, we can project any given position immediately on the detector grid.

Chapter 7

GPU-acceleration in Elastic Image Registration

This chapter is partially based on the following papers:

- Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Efficient GPU-Accelerated Elastic Image Registration. *In Proc. Sixth IASTED International Conference on BIOMEDICAL ENGINEERING (BioMed),* February 13-15, 2008, Innsbruck (Austria), pp. 419-424
- Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Accuracy of GPU-based B-Spline Evaluation. *In Proc. Tenth IASTED International Conference on COMPUTER GRAPHICS AND IMAGING (CGIM),* February 13-15, 2008, Innsbruck (Austria), pp. 117-122
- Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Efficient GPU-Based Texture Interpolation using Uniform B-Splines. *Journal of Graphics Tools,* Volume 13, Number 4, pp. 61-69, 2008

7.1 Introduction

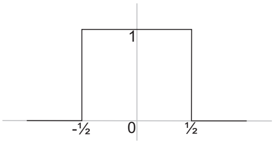
The advantage of elastic intra-patient image registration over rigid registration is the fact that it can take local deformation of anatomical structures into account. A cubic B-spline based deformation field is sufficiently smooth to model local elastic displacements of anatomical structures (*e.g.*, organs or breast) [107, 137]. However, the application of elastic registration during interventional treatment is still seriously limited by the considerable computation time, which is determined by the very large parameter space of the elastic deformation.

An approach to reduce the computation time, without changing the essential algorithm, is the employment of the vast computation power of the modern off-the-shelf GPU hardware. Though the overall computation power of the GPU nowadays surpasses the capabilities of the CPU, its performance does not scale equally well for any type of algorithm. In the literature there are several publications dealing with GPU-based elastic registration [138–140], using a piece-wise linear deformation field. We propose a GPU-driven cubic B-spline deformation field, which yields a smoother warping, and therefore can be considered to be a more realistic model for organic deformations.

Further we discuss how the capture range of the elastic registration can be enlarged. It is well known [141] that derivative-based optimizers (*e.g.*, quasi-Newton-like optimizers) only evolve to the correct solution if the initial position in the parameter space is sufficiently close to the optimum. Our approach to elastic registration lends itself very nicely to use derivative information from larger scale-spaces [142]. This allows the optimization process to take information of a larger neighbourhood into account, and therefore is less prone to get stuck in a local optimum.

7.2 Uniform B-spline interpolation

Uniform spline-based interpolation was introduced by Schoenberg [143] and has been described exhaustively by Unser [110]. The starting point for any degree of the B-spline function forms the B-spline basis of degree 0, also known as the box function. We use the variant of the B-spline function that is centered around the origin, which is chosen since its symmetry can be exploited within the GPU program:



$$\beta_0(x) = \begin{cases} 1, & |x| < \frac{1}{2} \\ \frac{1}{2}, & |x| = \frac{1}{2} \\ 0, & |x| > \frac{1}{2} \end{cases} \tag{7.1}$$

Any subsequent B-spline basis of degree n can be obtained by the recursive convolution of the box function with the B-spline basis of degree $n-1$:

$$\beta_n(x) = \beta_0(x) * \beta_{n-1}(x), \qquad n \geq 1 \tag{7.2}$$

The derivative of the B-spline basis function can easily be obtained by:

$$\frac{\delta\beta_n(x)}{\delta x} = \beta_{n-1}\left(x + \tfrac{1}{2}\right) - \beta_{n-1}\left(x - \tfrac{1}{2}\right) \tag{7.3}$$

Which means that the derivative of a B-spline function of degree n, is a B-spline function of degree $n-1$. Further it can be concluded that the B-spline function of degree n has a non-zero derivative up to the n-th order, which is a indicator for the 'smoothness' of the function.

The integral of the B-spline basis function of degree n can be expressed as:

$$\int_{-\infty}^{x} \beta_n(x)\,dx = \sum_{k=0}^{+\infty} \beta_{n+1}\left(x - \frac{1}{2} - k\right) \tag{7.4}$$

Spline-based interpolation at a given position $x \in \mathbb{R}$ can be written as:

$$s(x) = \sum_{k \in \mathbb{Z}} c(k)\beta_n(x-k) \tag{7.5}$$

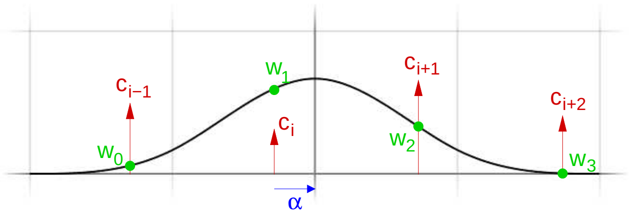


Figure 7.1: *Cubic B-spline interpolation. The image coefficients c are multiplied by the weights $w_n(\alpha)$. The weights are determined by the fractional amount α of the present coordinate, and the B-spline basis function β_3. Index i is the integer part of the coordinate.*

Or in words: the interpolated value s at a given position x is the summation of the shifted central B-spline β_n, weighted by the B-spline coefficients $c(k)$, which are located on a uniform (regular) grid.

Since B-splines have limited support, the amount of coefficients $c(k)$ that play a role in the interpolation at position x are quite moderate. It should be pointed out that $c(k) = s(k)$ is only the case for the 0*th* and 1*st* order B-spline (corresponding to nearest neighbour and linear interpolation). The coefficients for the cubic B-spline can be efficiently obtained, using a causal and anti-causal filter (see [110]).

The 0*th* (nearest-neighbour), 1*st* (linear) and 3*rd* (cubic) order B-spline are most popular. The 0*th* and 1*st* order B-spline can be evaluated very rapidly, and do not need any change of sampled values. However often they do not produce a result that is sufficiently close to natural signals. The cubic B-spline is sufficiently smooth, while its support is still quite local (its width is 4), which is favourable for the cost of the interpolation. Since the deformation of organs and other anatomical structures is typically rather smooth, we chose the cubic B-spline to model our deformation field.

7.2.1 Cubic B-spline interpolation

Evaluating cubic B-spline interpolation for any given position involves the weighted addition of the four adjacent coefficients (see figure 7.1), which allows equation 7.5 to be rewritten as:

$$\begin{aligned} s_3(i+\alpha) &= w_0(\alpha)\cdot c(i-1) + w_1(\alpha)\cdot c(i) + \\ &\quad w_2(\alpha)\cdot c(i+1) + w_3(\alpha)\cdot c(i+2) \end{aligned} \tag{7.6}$$

whereby the weights w depend on the fractional amount α of the present coordinate, and on the cubic B-spline basis function. More specifically:

$$\begin{aligned} w_0(\alpha) &= \beta_3(-\alpha-1) \\ w_1(\alpha) &= \beta_3(-\alpha) \\ w_2(\alpha) &= \beta_3(1-\alpha) \\ w_3(\alpha) &= \beta_3(2-\alpha) \end{aligned} \tag{7.7}$$

7.2.2 GPU-accelerated cubic B-spline evaluation

Sigg and Hadwiger [13] have described how cubic B-spline interpolation can be performed efficiently by the GPU. Their method is based on decomposing the cubic interpolation into 2^N weighted linear interpolations, instead of 4^N weighted nearest neighbor interpolations, whereby N denotes the dimensionality. Since linear interpolations are hardwired on the graphics hardware, they can be performed much faster than addressing the corresponding set of nearest neighbor lookups.

The basic idea can be understood by considering 1D linear interpolation, which can be expressed as follows:

$$s_1(i+\alpha) = (1-\alpha)\cdot s_0(i) + \alpha \cdot s_0(i+1) \tag{7.8}$$

with $i \in \mathbb{N}$ being the integer part of the interpolation coordinate and $\alpha \in \mathbb{R}$ being the fractional part in the range $[0,1]$. Building on this equation, the weighted addition of two neighbouring samples can be rewritten to be expressed as a weighted linear interpolation:

$$a \cdot s_0(i) + b \cdot s_0(i+1) = (a+b)\cdot s_1(i + \tfrac{b}{a+b}) \tag{7.9}$$

Using equation 7.9, equation 7.6 can be decomposed into two weighted linearly interpolated lookups:

$$\begin{gathered} s_3(i+\alpha) = g_0 \cdot c_1(i+h_0) + g_1 \cdot c_1(i+h_1) \\[2ex] \begin{aligned} g_0 &= w_0 + w_1 \\ g_1 &= w_2 + w_3 \\ h_0 &= (w_1/g_0) - 1 \\ h_1 &= (w_3/g_1) + 1 \end{aligned} \end{gathered} \tag{7.10}$$

whereby c_1 expresses linear interpolation between the cubic B-spline coefficients.

This scheme can easily be extrapolated to the N-dimensional case, whereby $g_{\vec{j}} = \prod g_{j_k}$, and $\vec{h}_{\vec{j}} = \sum \vec{e}_k \cdot h_{j_k}$, with k denoting the axis and $\vec{e}_k$ the basis vector. For 3D cubic interpolation 64 nearest neighbour lookups can be replaced by 8 linear interpolations. On modern GPUs that leads to a considerable performance gain.

Sigg and Hadwiger put g_0, h_0 and h_1 as a function of α in a 1D lookup texture (g_1 is redundant), and use this texture to obtain the variables g and h in the GPU program. They suggest using an RGB texture, consisting of 128 samples of 16-bit accuracy, and using linear filtering between the samples. For 3D interpolation this approach involves three lookups in this texture, and from the resulting parameters the eight coordinates for the linear interpolations are calculated.

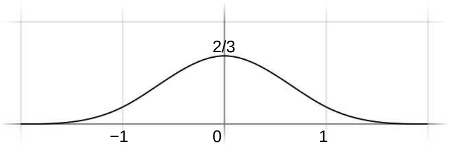
The lookup table distributes the cubic interpolation into two parts in the programming code: the GPU part that performs the actual interpolation, and the CPU part that creates the lookup table. Furthermore, the lookup table is one of the sources of imprecision, since for any value between its entries linear interpolation is used. Therefore, we explore the on-the-fly calculation of the weights on the GPU, reducing source code complexity, and improving the precision.

Equation 7.10 shows that the variables g and h are a function of the B-spline weights w obtained in equation 7.7. Since the B-spline is composed of piece-wise polynomials, it would appear that a GPU implementation would involve a number of undesirable conditional statements, leading to a considerable slowdown of the GPU program. However, the conditional statements can be avoided, since the determination of the weights is facilitated by the fact that w_0 is always located in the first quadrant of the cubic B-spline, w_1 always in the second, *etc*. Since the cubic B-spline (as well as its derivatives) consist of a single equation per quadrant (see figure 7.2), the following equations for the set of weights can be established:

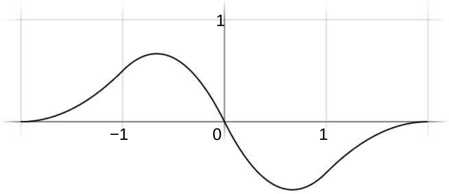
$$\begin{aligned}
w_0(\alpha) &= \tfrac{1}{6} \cdot (1-\alpha)^3 \\
w_1(\alpha) &= \tfrac{2}{3} - \tfrac{1}{2}\,\alpha^2 \cdot (2-\alpha) \\
w_2(\alpha) &= \tfrac{2}{3} - \tfrac{1}{2}(1-\alpha)^2 \cdot (1+\alpha) \\
w_3(\alpha) &= \tfrac{1}{6} \cdot (\alpha)^3
\end{aligned} \tag{7.11}$$

After the weights have been established, the variables g and h can be calculated, using equation 7.10. The CUDA code [144] below illustrates this process for the 2D case. It should be noted that the code can be ported very easily to *e.g.*, Cg [145], the OpenGL Shading Language or DirectX HLSL.

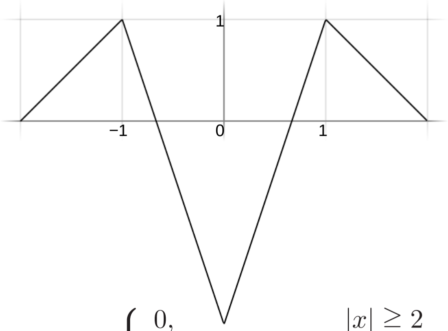
```
__device__ float interpolate_bicubic(texture tex, float x, float y)
{
   // transform the coordinate from [0,extent] to [-0.5, extent-0.5]
   float2 coord_grid = make_float2(x - 0.5, y - 0.5);
   float2 index = floor(coord_grid);
   float2 fraction = coord_grid - index;

   float2 one_frac = 1.0 - fraction;
   float2 one_frac2 = one_frac * one_frac;
   float2 fraction2 = fraction * fraction;
   float2 w0 = 1.0/6.0 * one_frac2 * one_frac;
   float2 w1 = 2.0/3.0 - 0.5 * fraction2 * (2.0-fraction);
   float2 w2 = 2.0/3.0 - 0.5 * one_frac2 * (2.0-one_frac);
   float2 w3 = 1.0/6.0 * fraction2 * fraction;

   float2 g0 = w0 + w1;
   float2 g1 = w2 + w3;
   // h0 = w1/g0 - 1, move from [-0.5, extent-0.5] to [0, extent]
   float2 h0 = (w1 / g0) - 0.5 + index;
   float2 h1 = (w3 / g1) + 1.5 + index;

   // fetch the four linear interpolations
   float tex00 = tex2D(tex, h0.x, h0.y);
   float tex10 = tex2D(tex, h1.x, h0.y);
   float tex01 = tex2D(tex, h0.x, h1.y);
   float tex11 = tex2D(tex, h1.x, h1.y);

   // weigh along the y-direction
   tex00 = lerp(tex01, tex00, g0.y);
   tex10 = lerp(tex11, tex10, g0.y);

   // weigh along the x-direction
   return lerp(tex10, tex00, g0.x);
}
```



$$\beta_3(x) = \begin{cases} 0, & |x| \geqq 2 \\ \frac{1}{6} \cdot (2 - |x|)^3, & 1 \leqq |x| < 2 \\ \frac{2}{3} - \frac{1}{2}|x|^2 \cdot (2 - |x|), & |x| < 1 \end{cases}$$



$$\beta_3'(x) = \begin{cases} 0, & |x| \geqq 2 \\ \frac{1}{2}x^2 + 2x + 2, & -2 < x \leqq -1 \\ -\frac{3}{2}x^2 - 2x, & -1 < x \leqq 0 \\ \frac{3}{2}x^2 - 2x, & 0 < x \leqq 1 \\ -\frac{1}{2}x^2 + 2x - 2, & 1 < x < 2 \end{cases}$$



$$\beta_3''(x) = \begin{cases} 0, & |x| \geqq 2 \\ 2 - |x|, & 1 \leqq |x| < 2 \\ 3 \cdot |x| - 2, & |x| < 1 \end{cases}$$

Figure 7.2: *The cubic B-spline, and its 1st and 2nd order derivative. Note that for all, there is a single equation per quadrant. We use the variant of the B-spline function that is centered around the origin, since this allows us to exploit its symmetry in the GPU programs.*

7.2.3 Accuracy and performance

In table 7.1 the deviation from the expected interpolated value is given for the lookup table based cubic interpolation method and the on-the-fly method. The error is defined as the normalized pixel intensity calculated by the GPU program, minus the intensity calculated by the CPU using double floating point precision. The root mean square of the errors was calculated for 512^2 pixels. The on-the-fly method is both more accurate and faster. However, on older graphics hardware (before 2007) the on-the-fly approach is slightly slower than the lookup table method, while still being more accurate.

Table 7.1: *Accuracy and timing of cubic interpolation with and without using a lookup table. All measurements were obtained on an nVidia GeForce 9800 GTX.*

Method	RMS	Time (ms)
Lookup table	$9.39 \cdot 10^{-5}$	0.96
On-the-fly	$8.58 \cdot 10^{-5}$	0.74

GPU-based interpolation suffers from another precision issue. When *e.g.*, an 8-bit texture is filtered, most people would expect that first the neighboring texture knots are queried, cast to floating point, and then weighted and added. This is, however, not the case; the texture knots are first weighted and added, and then cast to floating point, which limits the precision to the least significant bit of the texture data format [59], as is illustrated in figure 7.3. As a consequence, higher accuracies can only be obtained by using larger texture words, and thus at the cost of texture memory consumption.

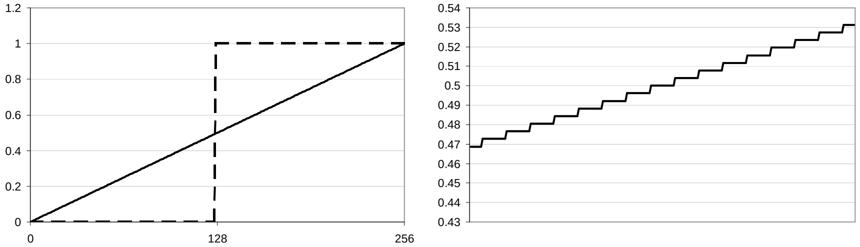


Figure 7.3: *The left graph shows linear interpolation between 0 and 1/65535 using a 16-bit integer texture (dashed) and a 16-bit floating point texture (solid). The right graph zooms in on the solid line, showing the limited precision of the fixed point texture coordinates.*

A further precision issue of the linear texture interpolation is caused by the fact that the accuracy of the texture coordinates is limited to a fixed point format with 8 bits of fractional value [146]. This means that there are only 254 discrete coordinate positions between two texture knots, as shown in the zoomed graph in figure 7.3, which especially is of interest when the knots are far apart (*e.g.*, in a B-spline deformation field for elastic registration). The mentioned texture interpolation accuracy

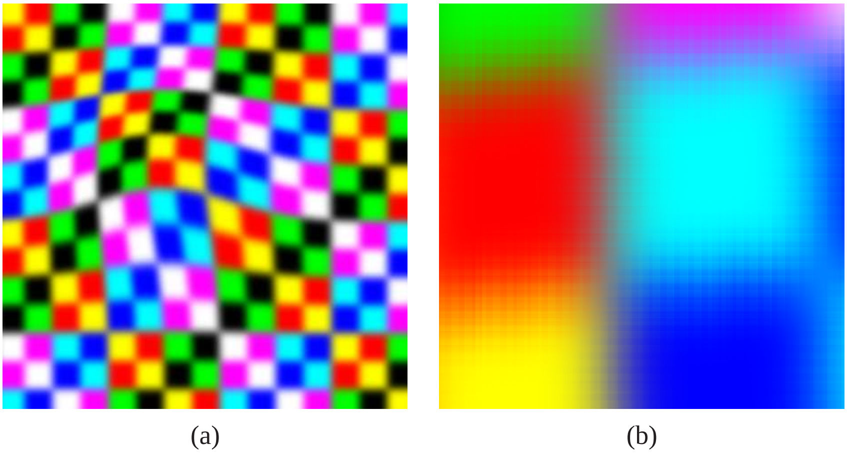

(a) (b)

Figure 7.4: *(a) A GPU-based B-spline deformed image. (b) A zoomed part of the left image. The artifacts are clearly visible: the transition between the blocks should be smooth, whereas it is jerky.*

effects are the cause for the deviations of the on-the-fly method in table 7.1 and is illustrated in figure 7.4.

Performance measurements of 3D cubic B-spline interpolation, using a CUDA implementation of the on-the-fly method on an nVidia GeForce 9800 GTX, reached $356 \cdot 10^6$ cubic interpolations per second. As a reference, a straightforward CUDA implementation using 64 nearest neighbour lookups delivered $93.6 \cdot 10^6$ cubic interpolations per second, and simple tri-linear interpolation delivered $486 \cdot 10^6$ linear interpolations per second. Cubic interpolation was also implemented to run on the CPU. On an Intel Xeon 5140 2.33 GHz a straightforward implementation delivered $0.45 \cdot 10^6$ cubic interpolations per second and a multi-threaded SSE implementation managed $10.3 \cdot 10^6$ cubic interpolations per second.

Since the tri-cubic approach uses eight tri-linear interpolations per cubic interpolation, a slowdown of factor eight could be expected. The cubic interpolation scores much better than this, which can be explained by the fact that the mentioned eight linear interpolations are spatially very close to each other, and the data, therefore, is still locally present in the texture cache. This favorable performance aspect, together with the compact code, makes the cubic B-spline interpolation an attractive solution for fast and high quality interpolation on the GPU.

7.3 GPU-accelerated Elastic Registration

7.3.1 Similarity measure

The similarity measure E used in intensity based registration algorithms is a function of the reference image A and the floating image B, which is deformed to the coordinate space of the reference image:

$$E = E(A, B^\tau) \tag{7.12}$$

whereby B^τ represents the deformed floating image. Let $\vec{i}$ be a position in the reference image space, and the function $\vec{\tau}(\vec{i})$ be the deformation of the reference image coordinate system to the floating image coordinate system. Obviously B^τ and B are connected as follows:

$$B^\tau(\vec{i}) = B(\vec{\tau}(\vec{i})) \tag{7.13}$$

In this chapter we will restrain ourselves to the class of algorithms, in which the similarity measure can be expressed as a sum of contributions per spatial element (pixel for 2D, voxel for 3D, *etc.*). Sum of Squared Differences (SSD) and Cross-Correlation (CC) are examples of members of this class. This class generally can be written as follows:

$$\begin{aligned} E = \quad & \tfrac{1}{\|I\|} \sum_{\vec{i} \in I} e(A(\vec{i}),\, B^\tau(\vec{i})) = \\ & \tfrac{1}{\|I\|} \sum_{\vec{i} \in I} e\left(A(\vec{i}),\, B(\vec{\tau}(\vec{i}))\right) \end{aligned} \tag{7.14}$$

Here e denotes the contribution to the similarity measure per spatial element, and $\vec{i} \in I \subset \mathbb{Z}^N$ represents the set of N-dimensional discrete spatial positions (*i.e.*, pixel or voxel positions in the image).

The deformation $\vec{\tau}$ is driven by a set of parameters $\vec{c}_j$. It is this set of parameters that is manipulated by the iterative optimization algorithm. In order to obtain a better prediction of parameters used in the next iteration, the Jacobian matrix, containing the partial derivatives of the similarity measure to the parameter space $\delta E / \delta c_{j,k}$ is required [147], with index k denoting the axis. The partial derivative can be decomposed into the following product [107]:

$$\frac{\delta E}{\delta c_{j,k}} = \frac{1}{\|I\|} \sum_{\vec{i} \in I} \frac{\delta e(\vec{i})}{\delta B^\tau(\vec{i})} \left. \frac{\delta B(\vec{x})}{\delta x_k} \right|_{\vec{x} = \vec{\tau}(\vec{i})} \frac{\delta \tau_k(\vec{i})}{\delta c_{j,k}} \tag{7.15}$$

7.3.2 Deformation field

Similar to Kybic and Unser [107], we use a B-spline driven deformation field. The deformation field then can be described by the following equation:

$$\vec{\tau}(\vec{i}) = \vec{i} + \sum_{j \in I_c} \vec{c}_j \cdot \beta_n(\vec{i}/\vec{h} - j) \tag{7.16}$$

The deformation for position $\vec{i}$ is given by $\vec{\tau}(\vec{i})$. The set of control points $\vec{c}_j$, which drive the deformation, is denoted by $I_c \subset \mathbb{Z}^N$. Vector $\vec{h}$ represents the spacing of the control points, which is required to be integer. Since $\vec{i}$ is added to the sum, the identity deformation corresponds to all control points being zero. $\beta_n(\vec{i})$ is the N-dimensional tensor product of an uniform B-spline function, whereby n indicates the degree of the B-spline.

$$\frac{1}{6 \cdot 2} \cdot \begin{array}{|c|c|c|} \hline -1 & 0 & 1 \\ \hline -4 & 0 & 4 \\ \hline -1 & 0 & 1 \\ \hline \end{array} \qquad \frac{4}{81 \cdot 9} \cdot \begin{array}{|c|c|c|c|c|} \hline -2 & -6 & 0 & 6 & 2 \\ \hline -15 & -45 & 0 & 45 & 15 \\ \hline -27 & -81 & 0 & 81 & 27 \\ \hline -15 & -45 & 0 & 45 & 15 \\ \hline -2 & -6 & 0 & 6 & 2 \\ \hline \end{array}$$

Figure 7.5: *The* $3 \cdot 3$ *and* $5 \cdot 5$ *derivative kernel in x-direction, based on the 1st order derivative of the cubic B-spline (see fig. 7.2). Larger kernels can be used to obtain derivatives in higher scales.*

Similarity measure	Contribution per pixel	Derivative
SSD	$e(\vec{i}) = (A(\vec{i}) - B^{\tau}(\vec{i}))^2$	$\delta e / \delta B^{\tau} = 2 \cdot (A(\vec{i}) - B^{\tau}(\vec{i}))$
CC	$e(\vec{i}) = A(\vec{i}) \cdot B^{\tau}(\vec{i})$	$\delta e / \delta B^{\tau} = A(\vec{i})$

Table 7.2: *Sum of Squared Differences (SSD) and Cross-Correlation (CC) similarity measures, and their derivative with respect to the deformed image.*

7.3.3 Derivatives

As can be understood from equation 7.16, the derivative of the deformation field $\delta\tau_k(\vec{i})/\delta c_{j,k}$ simply is a constant term: $\beta_n(\vec{i}/\vec{h} - j)$. Since the control points are evenly spaced, a fixed template of width $n \cdot h$ can be pre-computed to express this derivative. During the calculation of the derivative, the template is then shifted over the image, depending on index j.

In contrast to [107], we do not obtain the derivative of the deformed floating image analytically. We rather use an image based approach, employing a convolution with Sobel-like kernels, which approximates the Gaussian derivative. Such a convolution can be very efficiently implemented to run on the GPU.

The usage of kernels also allows us to determine the derivative at different scales, by scaling the B-spline derivative: $\beta_3'(x/s)$. Employing a higher scale allows to increase the capture range of the optimization algorithm, since the derivative is based on a wider spatial range [106]. In order to obtain a derivative of the similarity measure that is fully based on a different scale, the floating and reference image should be Gaussian blurred. Our first tests show, however, that merely basing $\delta B(\vec{x})/\delta x_k$ on a larger scale, by using bigger derivative kernels (see figure 7.5) already results in an enlarged capture range.

The derivative of the first multiplicand in equation 7.15 depends on the used similarity measure. In table 7.2 the derivatives for SSD and CC are given.

7.4 GPU implementation

7.4.1 Similarity measure & derivatives

The GPU implementation of the similarity measure and the first order derivatives works as follows: for every voxel in the reference image a thread is started, and its contribution to the similarity measure and derivatives is calculated. In the thread the corresponding location in the deformed floating data is obtained by adding the cubic B-spline driven deformation field to the thread's voxel coordinate, see equation 7.16. Hereby, we can make efficient use from the fact that a cubic B-spline lookup can be decomposed into 8 linearly weighted interpolations, rather than 64 nearest neighbor lookups, which is much faster on the GPU [13, 59], see sections 7.2.2 and 7.2.3.

When the deformed coordinate has been established, the voxel intensities of the reference and floating datasets are fetched, and the similarity measure contribution of the thread can be established, see equation 7.14. The gradient of the floating dataset and its intensities are stored in a single texture with four entries per voxel. In this way the interpolated lookup at the deformed coordinate immediately yields the intensity and the gradient of the floating data $\delta B(\vec{x})/\delta x_k$ at this particular location. It should be noted that this gradient image is static during the optimization process, and therefore needs to be calculated only once for the entire registration procedure. The gradient image can easily be obtained on the GPU in a pre-processing step by convolving the floating image with Sobel-like derivative kernels of size 3^N for every axis direction. For a multi-scale approach the 3^N kernels can be replaced by larger kernels of size $(2 \cdot n + 1)^N$ (see figure 7.5), and the neighbourhood that has to be sampled, should be enlarged correspondingly. The sampling of the neighbourhood, especially for large kernels, can be further optimized using the principle described in equation 7.9.

The derivative of the similarity measure to the control points consists of three multiplicands, see equation 7.15. Two of those can be established for each GPU thread; the gradient of the floating data $\delta B(\vec{x})/\delta x_k$ with $\vec{x} = \vec{\tau}(\vec{i})$, and the derivative of the similarity measure to the voxel space $\delta e/\delta B^{\tau}$. The similarity measure $e(\vec{i})$ and first order derivatives contributions are stored in an intermediate 3D data array for each thread. The following pseudo CUDA code encapsulates the first pass for SSD:

```
__global__ void sim_kernel(float4* output, int3 h)
{
  int3 coord = thread.coord;
  float3 coordf = make_float3(coord) + 0.5f;
  float3 offset = interpolate_bspline(deform_coeffs, coordf / h);
  float3 deform = coordf + offset;

  float4 floating = tex3D(flt_img, deform);  //gradient.xyz, image.w
  float reference = tex3D(ref_img, coordf);

  float diff = reference - temp.w;
  //gradient, pre-multiplied by derivative of sim. meas.
  output[coord].xyz = 2 * diff * floating.xyz;
  output[coord].w = diff * diff;  //sim. meas.
}
```

In the second pass, the first order derivatives $\delta E/\delta c_{j,k}$ are calculated by multiplying a subset of the previously stored derivative data with the B-spline weights $\beta_3(\vec{i}/\vec{h}-j)$. The B-spline weights are constant, and can be decomposed into a tensor product of three pre-computed 1D arrays of width $4 \cdot h_k$. The second pass is illustrated by the following pseudo CUDA code:

```
__global__ void coeffs_kernel(float4* output, int3 h)
{
  int3 coord = thread.coord;
  int3 start = (coord-2) * h;
  int3 end = 4 * h;  //support of the cubic b-spline

  float4 temp = make_float4(0,0,0,0);
  for (int z = start.z; z < end.z; z++)
  for (int y = start.y; y < end.y; y++)
  for (int x = start.x; x < end.x; x++)
  {
    float tensor = tex1D(tx, x) * tex1D(ty, y) * tex1D(tz, z);
    float4 inter = tex3D(intermediate_img, start+(x,y,z));
    temp.xyz += tensor * inter.xyz;
    temp.w += inter.w;
  }
  output[coord] = temp;
}
```

7.4.2 Results

In order to characterize the calculation time of the proposed algorithm, the GPU implementation was compared to a straightforward single threaded CPU implementation and a multi-threaded SSE optimized CPU version. For all versions we used the approach that is introduced in the previous section, with loop-unrolling applied to the inner for-loop of the second pass. We used a 2.33 GHz quad-core Intel Xeon with 2 GB memory and an NVIDIA GeForce GTX 260 with 896 MB memory to perform our measurements. The reference and floating data was obtained by deforming a CT dataset according to a B-spline field of 16^3 randomly determined control points in the range [-8, 8] and adding some white noise.

We measured the time to obtain the similarity measure and first order derivatives by performing a quasi-Newton driven optimization in 40 iterations, and averaging the time per iteration. In order to bring the figures in the same range for different dataset sizes (ranging from 32^3 voxels to 256^3 voxels) we divided the time per iteration by the number of voxels in the reference and floating dataset, see figure 7.6. It can be concluded that the time per voxel depends somewhat on the amount of control points, and not very much on the dataset size.

On our quad-core machine the multi-threaded SSE algorithm performed best when using four threads (figure 7.7 left), which we used for all our other measurements. The speedup factor of the GPU version compared to the other two implementations is illustrated in figure 7.7 right. The boxplots in figure 7.8 were obtained by comparing the time per iteration for similar datasets sizes and amount of control points. They show the median, and the distribution of the speedup factors.

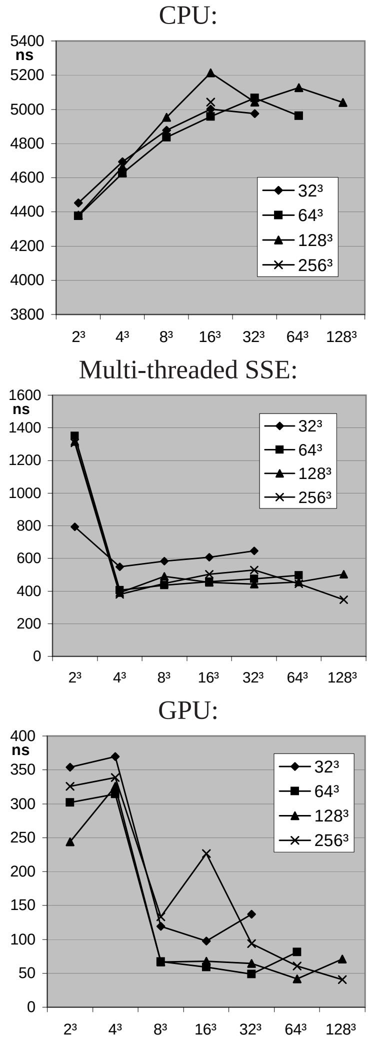


Figure 7.6: *The graphs show the amount of time (in nanoseconds) that is spend per voxel, when calculating the similarity measure and first order derivatives for a given transformation. The y-axis indicates the time, and the x-axis the amount of B-spline transformation control points. The lines in the graphs correspond to different dataset sizes. The top graph shows the measurements for the straightforward CPU version, the middle graph represents the multi-threaded SSE implementation, and the bottom graph the GPU version. Note that the scale and range of the y-axis is different. The measurements for* e.g., 128^3 *control points show a performance improvement for the GPU of a factor 100 over the CPU implementation.*

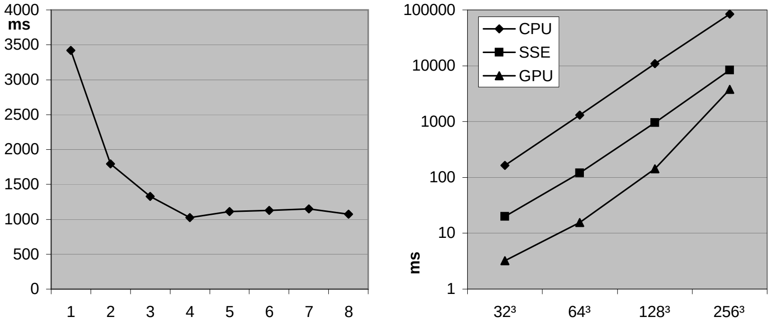


Figure 7.7: *The left graph shows the calculation time per iteration (y-axis, in milliseconds) for the SSE implementation, using different amount of threads (x-axis). Four threads provide the maximal use of processing resources at the least amount of overhead on the quad-core machine. The right graph represents the calculation time per iteration (in milliseconds, logarithmic scale) for different dataset sizes, using* 16^3 *B-spline control points, for the three implementations.*

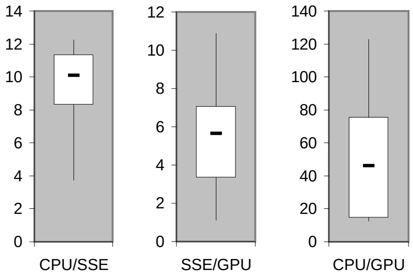


Figure 7.8: *The boxplots show the speedup factor distribution when comparing the various implementations. The left one represents the speedup factor distribution of the multi-threaded SSE implementation over the single-threaded straightforward CPU version. The middle shows the speedup of the GPU over the multi-threaded SSE version. The right boxplot illustrates the speedup of the GPU over the single-threaded straightforward CPU version.*

When we dissected the time per iteration into the time used for the first and second pass (see table 7.3), we discovered that the GPU version spends considerably more time in the second pass than in the first pass.

7.4.3 Discussion

In practise, a good approach has proven to start the registration in low resolution with few control points to find large deformations, and to gradually refine the registration by moving to higher resolutions and more control points [148]. Let us consider *e.g.*, a registration that first performs 20 iterations at a resolution of 64^3 with 8^3 control points, then 10 iterations at 128^3 with 16^3 control points, and finally 5 iterations at 256^3 with 32^3 control points. The straightforward CPU version would take 329 sec-

Implementation	Pass 1	Pass 2	Overhead
SSE	536.8 ms	474.2 ms	3.1 ms
GPU	9.6 ms	128.8 ms	1.5 ms

Table 7.3: *Distribution of the time per iteration over the passes, using datasets of* 128^3 *voxels and* 16^3 *control points.*

onds (5.5 minutes) to perform this registration, the multi-threaded SSE version costs 31.2 seconds, and the GPU implementation takes 7.4 seconds. Five minutes is unacceptable for many interventional and surgical applications, 31.2 seconds becomes an issue when the registration has to be performed multiple times (to compensate for progressively deforming of the brain), while 7.4 seconds is quite acceptable.

7.5 Conclusions

This chapter described how intensity based elastic registration algorithms, using a B-spline deformation model, can be implemented efficiently to run on the GPU. The various aspects of an efficient and accurate approach to cubic B-spline deformation on the GPU were discussed, and the accuracy issues that may arise when the GPU is used for this task were examined. Further, it was demonstrated how the similarity measure, as well as its derivative, can be calculated by the GPU, using a two-pass solution. Also it was indicated how a multi-scale approach of the derivative can help to enlarge the capture range, when employing quasi-Newton like optimizers.

In order to characterize the calculation time of the proposed algorithm, the GPU implementation was compared to a straightforward single threaded CPU implementation and a multi-threaded SSE optimized CPU version. As test data eight different cone-beam CT datasets of the head of patients with either arterio-venous malformations or aneurysms were used. The time to obtain the similarity measure and first order derivatives was measured by performing a quasi-Newton driven optimization in 40 iterations, and averaging the time per iteration. In order to bring the figures in the same range for different dataset sizes we divided the time per iteration by the number of voxels in the datasets.

It can be concluded that the time *per voxel* depends somewhat on the amount of control points, and not very much on the dataset size. On average a speedup factor of 50 compared to the straightforward CPU implementation and a factor of 5 with respect to the multi-threaded SSE version was reached. When these performance figures are projected on a realistic calculation scenario, we can conclude that the straightforward CPU implementation is too slow for habitual application during surgery. The multi-threaded SSE approach is suitable for singular use during the intervention, while the GPU version is considered fast enough for multiple usage to correct for progressive deforming of the treated anatomy.

Chapter 8

Vesselness-based 2D-3D Registration

This chapter is an extended revision of the following paper:

- Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Vesselness-based 2D-3D registration of the coronary arteries. *International Journal of Computer Assisted Radiology and Surgery,* Volume 4, Number 4, June 2009, pp. 391-397. doi:10.1007/s11548-009-0316-z

8.1 Introduction

In this chapter we propose a new method for registering the coronary vessel tree in intra-operative X-ray angiography images to a 3D model of the coronary vasculature, which has been obtained from a pre-operative CTA dataset. When such a registration has been established, a fused visualization of the real-time X-ray image stream and the 3D CTA data can be displayed, which is very useful for guidance of intra-vascular devices, such as catheters, during the minimal invasive treatment of coronary artery disease (CAD). Especially for chronic total occlusion (CTO) of a coronary artery, this procedure has great clinical benefit, since the occluded part of the artery, which is practically invisible in the X-ray image, still can be depicted in the CTA dataset. The objective of the 2D-3D registration algorithm is to find a spatial mapping between the 2D and the 3D images.

8.2 Related work

2D-3D image registration has a number of clinical applications, such as radiotherapy planning and verification [149–152], surgery planning and guidance [123, 153–155], and minimal invasive vascular treatment in coronary artery [127], peripheral [123, 125, 156] and neuro-interventions [157–162].

Most algorithms for 2D-3D image registration can be classified as either intensity-based or feature-based. Intensity-based methods [123, 125, 129, 153, 159, 163–170] directly use the pixel and voxel values to calculate a similarity measure, and require no or little segmentation. Feature-based methods [124, 151, 155, 158, 161, 162, 171–175] are based on a segmentation of landmark features in the images. Once this segmentation has been obtained, the registration step can be performed quite fast. The segmentation, however, is not always trivial or robust, and erroneous segmentations can lead to erroneous registrations.

Due to their tubular structure, vessels occupy a relatively small fraction of the image, which especially poses a hurdle in intensity-based image registration. Therefore feature-based registration has received particular interest for vascular 2D-3D registration. Many feature-based methods are based on the iterative closest point (ICP) approach [15], which relies on minimizing the sum of minimal distances between the feature points in the reference and projected image. Fitzgibbon [114] has shown that the distance transform can be used in ICP-like registration, in order to improve its efficiency. This approach has been applied [161, 162] to register the neuro-vasculature by segmenting the vessel tree in the 2D and the 3D image and applying a stochastic optimization strategy.

8.3 Method

8.3.1 Spatial mapping

In section 6.6 it has been shown how the three dimensional space is projected on the detector image of an X-ray C-arm. This projection could be expressed in a single matrix M. In order to bring a CT dataset into this three dimensional space, we need to establish the relation between the CT frame of reference and the C-arm space, and to incorporate this relation into the projection matrix M'.

The transformation from the frame of reference of the CT dataset to the iso-centric X-ray coordinate frame can be decomposed into two a-priori known relations and an unknown part. The DICOM header of the CT data already tells us how the patient was oriented with respect to the CT data. We also know how the patient is oriented with respect to the X-ray equipment (*e.g.*, head first, nose up). Furthermore, the patient is typically positioned to have the center of the anatomy of interest (*e.g.*, the coronary arteries) approximately at the iso-center of the X-ray C-arm equipment. The accuracy of this information is quite limited, but it allows for a coarse initialization of the spatial mapping and reduces the search space of the registration process. The fine tuning is represented by the unknown part, and it is the objective of our registration algorithm to find a more precise mapping.

Matrix O expresses the a-priori information. The rotational part of this matrix is extracted from the DICOM information of the CT data. The translational part is established such that the center of the coronary vessel model in the CT dataset corresponds to the iso-centric origin of the X-ray coordinate frame.

Matrix T is the rigid registration matrix, which is manipulated during the optimization process. It should be noted that we do not perform a calibration of the X-ray equipment to correct for deviations of parameters delivered by the system, since these deviations are found to be smaller than the deformations that occur due to the cardiac and respiratory motion. Furthermore we rely on the registration process to correct also for the system inaccuracies.

The complete transformation M' of a point in the frame of reference of the CT data to the X-ray detector space is an extension of equation 6.5 and can be expressed by:

$$M' = P \cdot T \cdot R \cdot O \tag{8.1}$$

We deliberately put matrix T between matrices P and R, because in this way its axes are aligned with the axes of the X-ray detector. This is of importance, since the translation perpendicular to the detector (z-axis of the detector) only leads to perspective zoom, which is very difficult to estimate accurately for deforming structures such as the coronary arteries of a beating heart. Therefore, we rather do not change this z-translation in the image-based registration process.

8.3.2 Vesselness filter

The vesselness filter plays a central role in the registration method that is being described in this chapter. The vesselness filter expresses the likelihood that a particular pixel can be contributed to a vessel-like structure. As such it can be regarded as a fuzzy segmentation. In our experiments we apply a multi-scale vesselness filter, as proposed by Frangi [14], which will be briefly described in this section. Note however that our similarity measure is not restricted to this particular vesselness filter, but can use any filter that enhances the vascular structures and suppresses any other structure in the image.

The vesselness filter seeks to enhance tube-like structures in the image. These structures are identified using the eigenvalues of the Hessian matrix $\mathcal{H}$. This matrix, which contains the second order derivatives of the image L, can be written for the 2D case as:

$$\mathcal{H} = \begin{pmatrix} \frac{\delta^2 L}{\delta x^2} & \frac{\delta^2 L}{\delta x \delta y} \\ \frac{\delta^2 L}{\delta y \delta x} & \frac{\delta^2 L}{\delta y^2} \end{pmatrix} \tag{8.2}$$

Now let λ_1 and λ_2 be the eigenvalues of the Hessian matrix $\mathcal{H}$, and let them be ordered such that $|\lambda_1| \leq |\lambda_2|$. When $|\lambda_1|$ is small and $|\lambda_2|$ large, a ridge is encountered, which is a good indicator for a tube-like structure. When $|\lambda_1|$ and $|\lambda_2|$ are of similar magnitude, the encountered structure is a ‘blob’, which is not a property of vessel structures. To capture these properties, the ‘blobness’ is defined as:

$$\mathfrak{R}_{\mathfrak{B}} = \frac{\lambda_1}{\lambda_2} \tag{8.3}$$

Furthermore, the ‘structureness’ of the image is taken into account, which is defined as:

$$\mathcal{S}_t = ||\mathcal{H}||_F = \sqrt{\lambda_1^2 + \lambda_2^2} \tag{8.4}$$

This term produces a high response when there are a lot of image features present, and a low response for areas with few features (background).

Using these properties, the vesselness is defined as:

$$\mathcal{V} = \begin{cases} 0, & \lambda_2 \geq 0 \\ \exp\left(-\frac{{\mathfrak{R}_{\mathfrak{B}}}^2}{2\beta^2}\right)\left(1-\exp\left(-\frac{\mathcal{S}_t^2}{2c^2}\right)\right), & \lambda_2 < 0 \end{cases} \tag{8.5}$$

The parameters β and c can be tuned to change the sensitivity of the filter.

To take the various sizes of the vessels into account, the filter is applied in several scales. The different scales s are obtained by defining the differentiation used for the Hessian matrix at each pixel location $\vec{i}$ as a convolution with derivatives of a Gaussian of variable width:

$$\frac{\delta}{\delta x} L(\vec{i}, s) = L(\vec{i}) * \frac{\delta}{\delta x} G(\vec{i}, s) \tag{8.6}$$

with the two-dimensional Gaussian at scale s defined as:

$$G(\vec{i}, s) = \frac{1}{2\pi s^2} e^{-\frac{||\vec{i}||^2}{2s^2}} \tag{8.7}$$

The overall vesselness is then assembled from the vesselness at different scales:

$$\mathcal{V}(\vec{i}) = \max_s \left(\mathcal{V}(\vec{i}, s)\right) \tag{8.8}$$

8.3.3 Similarity measure

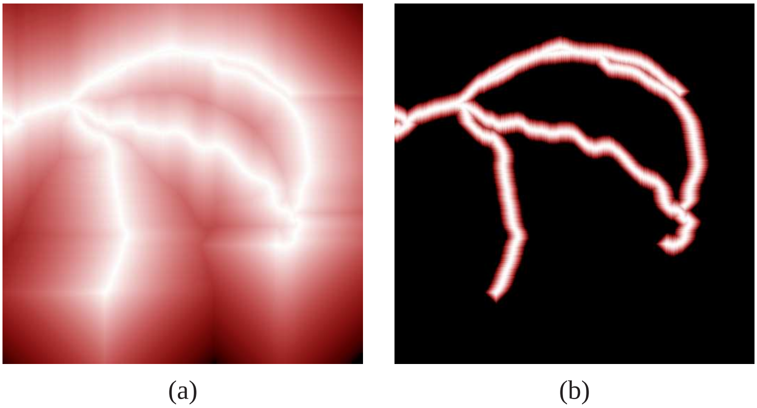
In order to obtain a 3D model of the coronary vessels, they are segmented in the pre-operative 3D CTA datasets. There are good algorithms available for this task (*e.g.*, [176–179]). Furthermore, it is not necessary to exclude manual interaction, since this segmentation can be performed pre-operatively, when there is less time-pressure and stress. A reliable and robust segmentation of the 2D X-ray angiography images can be more challenging, because of the projective nature of these images. Also, due to the intra-operative acquisition of the X-ray images manual interaction or correction is not desirable. To overcome these limitations we introduce a method which does not require an explicit segmentation of the 2D X-ray image.

We perform a distance transform (DT) on the projected 3D model in each iteration of the optimization process. The fact that the DT is calculated in every iteration differs from DT-based ICP, where the points of the 3D model are projected on a static DT of the segmented 2D image. It is necessary to recalculate the DT, because the pose of the 3D model changes in each iteration. The Distance Transform computes for each pixel position $\vec{i}$ the distance to the nearest feature point $\vec{q}$ in a set of feature points Ω, which is the set of projected 3D points in our case:

$$DT(\vec{i}) = \min_{\vec{q} \in \Omega} ||\vec{q} - \vec{i}|| \tag{8.9}$$

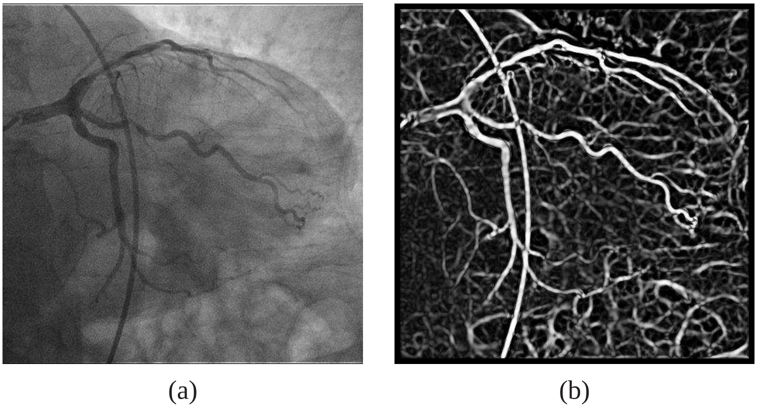
To achieve a rapidly declining distance weighting function $\mathcal{D}$ that yields only a high response close to the feature points, the squared DT is subtracted from a constant value c (see figure 8.1):

$$\mathcal{D}(\vec{i}) = \max\left(0,\ c - DT(\vec{i})^2\right) \tag{8.10}$$



(a) (b)

Figure 8.1: *(a) Inverted Distance Transform of the projected coronary vessels in the 3D CT dataset. (b) Inverted Squared Distance Transform, see equation 8.10.*



(a) (b)

Figure 8.2: *(a) X-ray image of the coronary arteries. (b) Vesselness transform of the X-ray image.*

A vesselness filter $\mathcal{V}$, as described in section 8.3.2, is applied to the 2D X-ray image (see figure 8.2). Our similarity measure can then be expressed as the sum of the product of the distance weighting function $\mathcal{D}$ and the vesselness $\mathcal{V}$ over all pixels in the image:

$$S = \sum_{\vec{i} \in I} \mathcal{D}(\vec{i}) \cdot \mathcal{V}(\vec{i}) \tag{8.11}$$

8.3.4 Optimization strategy

The search space, consisting of the multi-dimensional control variables of the spatial mapping, is determined by two degrees of translational and three degrees of rotational freedom (rigid registration). The process of projecting 3D data on a 2D plane implies a considerable reduction of information. As a result there are many incorrect transformations that yield a relatively good similarity measure (*e.g.*, projecting not corresponding vessel branches on each other), and form a local optimum in the search space.

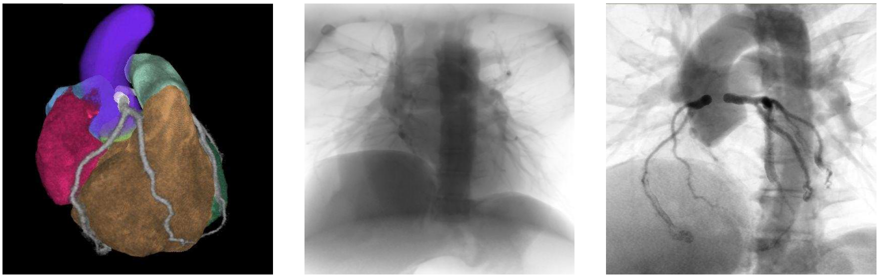
We use a stochastic optimization approach, since such approaches are less likely to get stuck in a local optimum. The used optimization strategy uses a population of samples in the search space. In every iteration of the optimization algorithm the n best samples are taken, and they each create m new samples. The m 'children' of a sample are randomly generated according to a Gaussian normal distribution. The standard deviation σ of the normal distributed random samples is multiplied with a reduction factor r for each iteration, since we assume that the global optimum is closer as we progress.

The initial σ can differ for each dimension of the search space. In our case we use a significantly smaller σ for the three rotation variables than for the three translation variables, since we can perform already a quite good estimation for the rotation of the 3D model, based on the DICOM information of the CT data, and the viewing incidence of the X-ray C-arm system.

8.4 Results

We evaluated the presented similarity measure with respect to accuracy and capture range, comparing it against ICP-based registration. As optimization strategy a standard Powell optimizer and the stochastic optimization strategy described in section 8.3.4 were used.

The accuracy and capture range was assessed using simulated data. In order to do this, the coronary arteries were segmented from a real cardiac CT data set, as well as the heart mask. From this CT dataset a Digitally Reconstructed Radiograph (DRR) was constructed, which simulates an X-ray projection of the CT data. In angiographic X-ray images the contrast medium is injected intra-vascularly, while the cardiac CT images are obtained with intra-venously administered contrast medium. Therefore, the DRR was generated with different X-ray attenuation coefficients assigned to the different segments, see figure 8.3. The registration process is then started with a given offset translation and rotation. The advantage of the simulated data is the fact that the



(a) (b) (c)

Figure 8.3: *(a) Segmented cardiac CT dataset. (b) A Digitally Reconstructed Radiograph (DRR) of the same dataset. (c) A DRR, using different X-ray attenuation coefficients per segment, simulating X-ray angiography.*

gold standard transformation is known, and therefore the error of the registration process can be quantified, see table 8.1.

Using the simulated data, we established a maximum capture range of 14.1 mm translation and 5.2° rotation for the ICP-Powell combination, whereby the capture range is defined as the set of initial translations and rotations with respect to the gold standard that still yield a successful registration. The vesselness-Powell combination delivered a capture range of 43.8 mm and 22.1° respectively, and the vesselness-stochastic combination reached 71.1 mm and 20.3°. The average calculation time for the ICP method was only 82 ms, while vesselness-Powell combination took 2.7 seconds and the vesselness-stochastic combination calculated for 11.0 seconds.

We further investigated the capture range using clinical data from four pairs of 2D X-ray images and 3D coronary vessel trees, segmented from cardiac CT data. It should be mentioned that it is impossible to establish an objective gold standard for such real world data, especially since the cardiac phase might differ somewhat for the 2D and 3D images of the pair. Therefore we proceeded in the following way: A large number (about 30 per dataset pair) of registrations were started from different starting positions (translation and rotation). The resulting transformation was then labelled either as ’successful’ or ’erroneous’ by an expert. The largest successful registration in sense of translated distance and rotated angle was taken as a measure for the capture range, see table 8.2.

Sim.meas. Optimizer	ICP Powell	Vesselness Powell	Vesselness stochastic
Translation (mm)	$\bar{x} = 1.44, \sigma = 1.50$	$\bar{x} = 0.42, \sigma = 0.12$	$\bar{x} = 0.54, \sigma = 0.47$
Rotation	$\bar{x} = 1.31°, \sigma = 1.00°$	$\bar{x} = 0.70°, \sigma = 0.77°$	$\bar{x} = 1.06°, \sigma = 1.00°$

Table 8.1: *Residual error of a successful registration, measured using simulated data. The same set of initial transformations was used for all methods.*

Sim.meas. Optimizer	ICP Powell	ICP stochastic	Vesselness Powell	Vesselness stochastic
Pair 1	0.0 mm, 0.0°	7.8 mm, 8.72°	83.3 mm, 21.7°	76.9 mm, 40.3°
Pair 2	21.8 mm, 17.5°	33.1 mm, 16.6°	68.7 mm, 31.9°	62.0 mm, 41.5°
Pair 3	12.9 mm, 25.2°	11.2 mm, 14.5°	49.7 mm, 23.8°	60.8 mm, 37.3°
Pair 4	12.1 mm, 11.0°	20.2 mm, 15.2°	30.8 mm, 30.3°	71.0 mm, 51.7°

Table 8.2: *The maximum capture range was established using clinical datasets.*

8.5 Discussion

In this chapter a novel feature-driven 2D-3D registration method has been introduced. This method is based on the iterative stochastic optimization of our similarity measure, which relies on the 3D coronary vessel model, obtained from a cardiac CT dataset, and a 2D X-ray image of the coronary arteries. The similarity measure is obtained by applying a vesselness filter to the 2D image, and then weighting it with a function based on the squared distance transform of the projected 3D vasculature.

It has been demonstrated that this similarity measure outperforms the Iterative Closest Point (ICP) method, both in sense of capture range and reliability. This can mainly be contributed to the fact that we do not perform an explicit binary segmentation of the vessel structures in the 2D X-ray image. This segmentation was rather trivial in many other publications, mainly dealing with Digital Subtraction Angiography (DSA) images (*e.g.,* [174, 175]), which are not available for the coronary arteries due to the heart and respiratory motion. In our approach an explicit segmentation is avoided by using directly the vesselness image in our similarity measure. Furthermore, the squared distance transform guarantees that only vessel structures close to the projected centerlines contribute to the similarity measure, while it is wide enough to maintain a large capture range.

It has been shown that the stochastic optimization approach enlarges the capture range, since it is not likely to get stuck in a local optimum far from the global optimum. Future work might include evaluating other stochastic global optimization strategies, as *e.g.*, proposed by Kennedy and Eberhart [180].

The test results have shown that the proposed registration approach can be calculated rather quickly, yielding calculation times similar to distance transform based ICP. The efficiency is reached by the limited extent of the squared distance transform, which can be calculated within far less iterations than a regular distance transform. The multi-scale vesselness filter, which is rather expensive to calculate, only needs to be obtained once for the entire registration process, and therefore does not pose a significant bottleneck.

When we started our search for a 2D-3D registration approach for the coronary arteries, we quickly abandoned intensity-based methods, because of their limited capture range for registration of vessel structures (intensity-based methods work best when there are large overlapping landmark areas, which is not a property of the vasculature). After initially disappointing results with feature-based registration, using a binary segmentation of the vessels in the 2D image, we developed the novel and robust approach sketched in this chapter. Our test results show that it performs very well for the task of 2D-3D registering of the coronary vessel tree.

Part III

Clinical Applications

Chapter 9

Real-time 3D Multimodality Fusion in Neuroangiography

This chapter is partially based on the following papers:

- Daniel Ruijters, Marijke Vermeer, Anna Vilanova, and Paul Suetens. Robustness of Mutual Information Based Intra-Operative Registration. *First Annual Symposium of the IEEE/EMBS Benelux Chapter*, December 7-8, 2006, Brussels (Belgium), pp. 171-174
- Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. 3D Multi-modality Roadmapping in Neuroangiography. *Proceedings of SPIE - Volume 6509, Medical Imaging 2007: Visualization and Image-Guided Procedures*, February 2007, San Diego (USA), pp. 65091F. doi:10.1117/12.708474
- Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. Real-time integration of 3-D multimodality data in interventional neuroangiography. *Journal of Electronic Imaging*, Volume 18, Issue 3, July-September 2009. doi:10.1117/1.3222939

9.1 Introduction

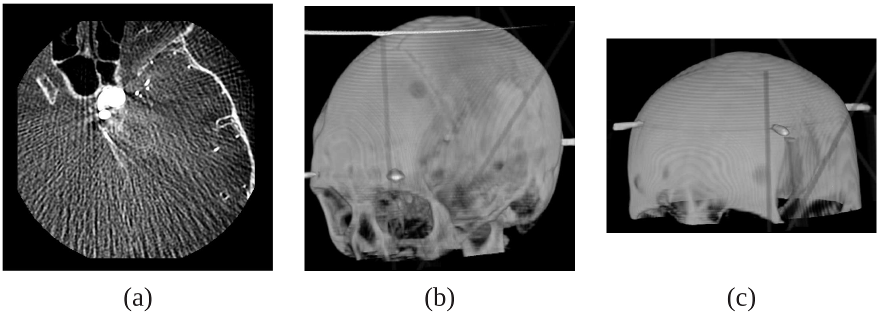
To the present date, the fluoroscopic image with the live information about endovascular interventional devices, and soft-tissue images (such as CT or MR) are visualized on separate displays. This means that the clinician has to perform a mental projection of the position of the endovascular device on the soft-tissue data. It may be clear that a combined display of this information is of great advantage, since it reliefs the clinician of performing this task. Furthermore, a fused image allows more precise navigation of the endovascular devices, since these devices are visualized together with pathologies and contextual information, present in the soft-tissue data. In order to provide the maximum benefit of such an augmented image, the live fluoroscopy data and the soft-tissue data have to be combined in real-time, with low latency and a sufficient frame rate (15 or 30 frames per second, depending on the acquisition mode). Since the visualization is targeted at the usage during an intervention, it should not only be fast, but also easy to interpret and the manipulation of the image should be interactive and easy to use.

9.2 Method

9.2.1 Pre-processing

Our approach relies on the acquisition of a 3DRA dataset at the beginning of the intervention. The 3DRA dataset is co-registered to a soft-tissue dataset, such as CT or MR, which has been obtained prior to the intervention (*e.g.*, for diagnostic purposes). Using 3D image registration during interventional treatment poses a number of constraints on the registration algorithm. Especially, the calculation time of the algorithm has to be limited, since the result of the registration process is to be used during the intervention. In order to reduce the calculation time, the GPU is employed to accelerate the registration algorithm [181, 182].

3DRA reconstructions may have a very high spatial resolution (a voxel can be as small as 0.1 mm), but tend to be rather noisy in the dynamic range, see section 2.4. To reduce the sensibility to noise we use a limited number (16-32) of grey level bins for the 3DRA dataset. As a result of the limited dynamic range, the vessels, bones and sinuses are the only structures that are well delineated, and can serve as landmarks. The registration process is primarily determined by the facial structures, such as the eye sockets, the nose, the sinuses, etc. It is therefore of importance that such structures are contained both in the 3DRA dataset, as well as the soft-tissue dataset, see figure 9.1.



(a) (b) (c)

Figure 9.1: *(a) A slice out of a 3DRA dataset, showing the limited dynamic range. The visible anatomy are the sinuses, the skull, and a contrast medium filled aneurysm. (b) A CT dataset, containing the facial structures. (c) A CT dataset, missing a major part of the facial structures, which hinders the registration process.*

Since we focus on cerebral applications, and there are only limited elastic transformations of the anatomical structures within the head, we can use a rigid registration (*i.e.*, only a global translation and rotation, see chapter 6). Rigid registration further has the property that it can be calculated relatively robust and fast. Typically, a registration algorithm consists of a multi-dimensional similarity measure, indicating the quality of a given spatial mapping, and an optimization algorithm, which searches the optimum (maximum or minimum, depending on the measure) of the similarity measure. The search space consists of the control variables of the similarity measure, which are in the case of rigid registration: translation in the x-, y- and z-direction, and rotation around the x-, y- and z-axis. We use Mutual Information as similarity measure, as described by Maes *et al*. [109], because it performs very well on intermodality registration and does not demand any a-priori knowledge of the datasets. In order to further limit the calculation time, we employ the Powell algorithm [116] as optimizer, which is a so-called local optimizer. Local optimization algorithms are generally faster than global optimizers, but they do not guarantee that the overall optimum is found. To assure that the correct optimum is found, the image-based registration is preceded by an optional rough manual registration, which is to be performed by the clinician. Note that this pre-processing step has to be performed only once.

A further pre-processing step forms the creation of a triangulated mesh, representing the vessel tree. In order to obtain such a mesh, the vessels are segmented in the 3DRA volume, which is a fairly easy task since the iodine contrast medium absorbs more X-ray than any other substance present in the dataset. From the segmented data a mesh is extracted by applying the marching cubes algorithm [183].

9.3 Clinical use

In order to visualize the data the techniques described in chapter 4 are being used. The availability of the live fluoroscopy image stream, combined with the vasculature segmented from the 3DRA dataset and the registered soft-tissue (CT or MR) dataset, during the intervention is of great clinical relevance. The combination of the fluoroscopy image with the 3DRA vessel tree provides the advantage that the guide wire and catheter position can be located with respect to the vessel tree, without additional contrast injection (see figure 4.5d), while the C-arm position and the X-ray source to detector distance can be altered freely [131]. Even during *e.g.*, rotations of the C-arm, the machine-based 2D-3D registration will always be up to date, see section 6.6. The additional visualization of the soft-tissue data allows to correlate the position of the guide wire and catheter to pathologies which are only visible in the soft-tissue data. Especially the fact that this information is available in real-time makes it very suitable for navigation.

The slab with the soft-tissue data can be moved, its width can be changed and its orientation can be rotated freely, to visualize different parts of the anatomical dataset. In this way the optimal view of a certain pathology can be determined. The implementation of the rendering, running on the GPU offers interactive speed throughout.

The integration 3D multi-modality data can be used in the following treatments:

- Navigation to the optimal position for intra-arterial particle injection in endovascular embolization of intracranial neoplastic tissue, and arteriovenous malformation (AVM) treatment, prior to stereotactic radiation surgery.

- Navigation to the optimal position for intracranial stenting in cases where aneurysms are pressing on surrounding eloquent and motoric brain tissue.

- Navigation in the vessel portions to be embolized in *e.g.*, hemorrhagic stroke.

- Navigation in the vessel segments where thrombolytic therapy should be applied in *e.g.*, ischemic stroke or vascular vasospasms.

9.4 Results

9.4.1 Robustness

In order to validate the applicability of our registration approach in the clinical practice, we investigated the capture range of the GPU-accelerated automatic registration algorithm, using clinical data. In this context we defined the capture range as the extent of the parameter search space that can serve as start position for the optimizer, and still evolves to a correct spatial transformation between the datasets, see chapter 6. If this extent is too small, the manual pre-registration becomes too cumbersome and time-consuming to be performed during an intervention.

First we determined a gold standard transformation for every dataset pair. This was done by manually defining a starting position that was sufficiently close to the correct transformation, and then let the registration algorithm run. The results were then visually inspected, to assure that the transformation was indeed correct. All gold standard transformations were of sub-voxel accuracy.

To establish the range of the search space where the algorithm behaves robustly, we made the following assumption: if a registration process, started from a translation in a certain direction, evolves to the gold standard transformation, each registration attempt from a smaller translation in the same direction is also assumed to lead to the gold standard transformation, *i.e.*, the capture range is convex without any holes. Hereby two transformations were considered to be the same if each of the components of the rotation matrix differ less than a particular δ_R (we used δ_R = 0.05), and the translation differs less than δ_T (we used δ_T = 0.5 mm).

Based on this assumption, the robust translation extent was determined, using an approach, similar to a binary search [184]; The gold standard transformation was applied to the datasets, and one dataset was translated in a certain direction $\vec{d}$, with $|\vec{d}|$ being of unit length. If performing the registration process indeed leads to the gold standard transformation, the process was repeated with the translation vector doubled. If not, the translation vector was halved. This process was continued until a bounding interval (b_1, b_2), with $b_1 < b_2$, was found, whereby a translation of b_1 still was within the capture extent, and b_2 not. Then, iteratively a new limit $b = (b_1 + b_2)/2$ was tested. If a registration started from a translation with vector $b \cdot \vec{d}$ evolved to the gold standard transformation, b was within the capture range, and b_1 was set to b for the next iteration. Otherwise b_2 was set to b. In this way the accuracy of the boundary of the capture range was doubled (the uncertainty was halved) in every iteration.

The iterative process was continued until the boundary of the capture range was found with an accuracy of 5 mm. Using this method, the robust translation range was determined for every patient in 14 distinct directions (see figure 9.2). A similar scheme was used to determine the robust rotation extent around the x-, y- and z-axes in both directions. The robust rotation range was determined with a precision of $1°$.

The capture range with respect to translation and rotation were determined for

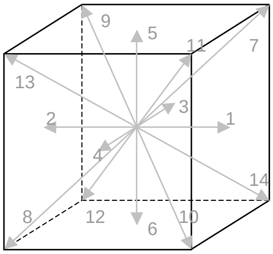


Figure 9.2: *The translation of the datasets was tested in all 14 depicted directions.*

dataset pairs obtained from 11 patients; 7 patients with a 3DRA-CT dataset pair, and 4 patients with a 3DRA - MR pair. 88% of the CT datasets can be registered correctly when the registration process is started within 30 mm translation to the gold standard transformation with the 3DRA dataset, see figure 9.3. 67% manage to robustly register within 50 mm translation. The results we obtained are comparable, or slightly better than published by Stancanello *et al*. [185]. The results of starting the registration process with the datasets rotated to each other, is illustrated in figure 9.4. 88% still of the CT datasets can be registered correctly to the 3DRA dataset when the rotation is 20°, 74% when the rotation is 30°.

The results for the 3DRA-MR dataset pairs are shown in figure 9.5. Unfortunately not all MR datasets fulfilled the criteria that were described in section 9.2.1 (not enough landmark regions present, slices too far apart). However, more than 60% of the registration attempts still succeed when the translation is 10 mm.

The accuracy of the calibrated machine-based 2D-3D registration was measured on five Philips Allura C-arm X-ray angiography systems. The registration was least accurate at the corners of the 3DRA reconstruction volume. The maximal deviation of the 2D fluoroscopy image and the projected 3DRA image was 0.4 mm at the corners of the reconstruction volume, and the average deviation at this location was 0.2 mm, which is well within the clinically acceptable range.

9.4.2 Computation time

The GPU implementation of the Mutual Information based registration algorithm takes less than 8 seconds to register the 3DRA dataset and the soft-tissue dataset in the pre-processing step. The extraction of the mesh that represents the vessels, the another pre-processing step, costs 300 ms. Overall it can be concluded that these figures are very acceptable and do not hinder the interventional procedure, especially since the pre-processing step has to be performed only once.

Given a certain set of viewing incidence angles, it takes a mere 1.5 μs to calculate the matrix, which expresses the 2D-3D registration between the 3DRA dataset and the fluoroscopy image. It is important that this part can be calculated in real-time, since it should be updated on-the-fly when the geometry pose of the X-ray C-arm system changes. The augmented visualization, consisting of a mesh extracted from a 256^3 voxel 3DRA dataset, a Volume Rendered slab from a $256^2 \cdot 198$ voxel CT dataset and the fluoroscopy image stream, can be displayed at an average frame rate

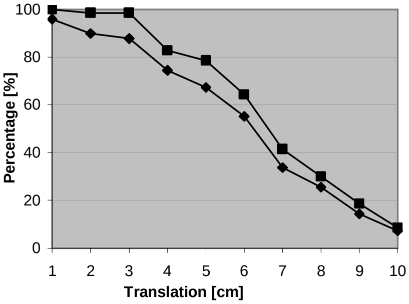


Figure 9.3: *The percentage of 3DRA-CT dataset pairs that can be registered correctly, for a given initial translation. The upper line shows the results if the two most difficult to register patients are not taken into account. The lower line indicates the results for all patients.*

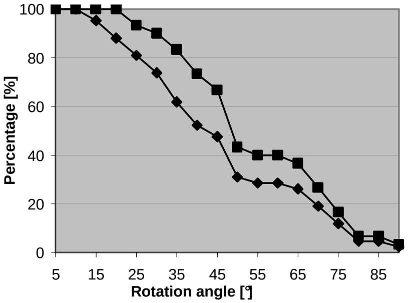


Figure 9.4: *The percentage of 3DRA-CT dataset pairs that can be registered correctly, for a given initial rotation. Upper line: without the two most difficult to register patients. Lower line: the results for all patients.*

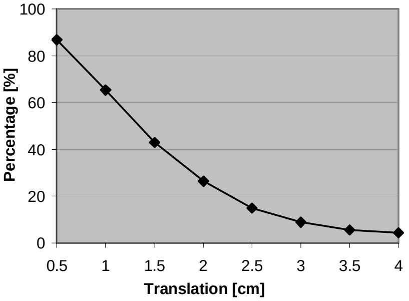


Figure 9.5: *The percentage of 3DRA-MR dataset pairs that can be registered correctly, for a given initial translation.*

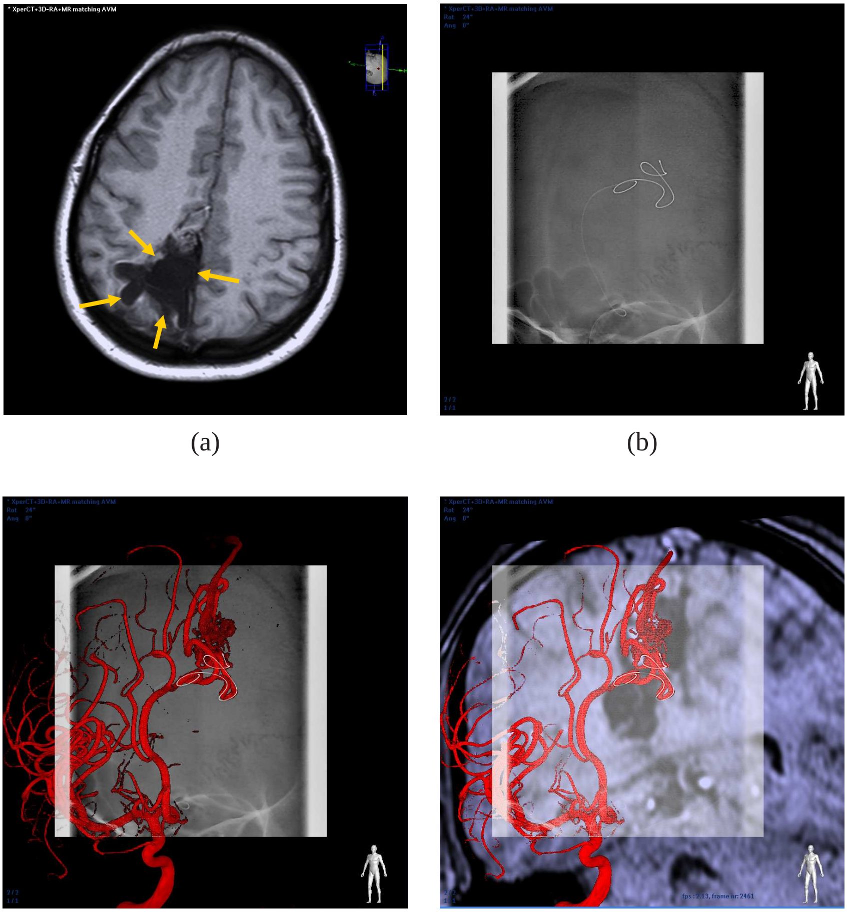
of 38 frames per second. All figures were measured on a Xeon 3.6 GHz machine with 2 GB of memory, and a nVidia QuadroFX 3400 graphics card with 256 MB of memory, using the datasets that are depicted in figure 4.5.

9.5 Discussion

Being able to see the live fluoroscopy image within the context of the 3D vasculature and soft-tissue information is of great clinical relevance. The combination of the fluoroscopy image with the 3DRA vessel tree adds value, since the guide wire and catheter position can be located with respect to the vessel tree without additional contrast injection (see figure 9.6b and 9.6c), while the C-arm position and the X-ray source to detector distance can be altered freely. Even during rotations of the C-arm, the machine-based 2D-3D registration will always be up to date. The clinical interest of this so called 3D-roadmapping has been described before [131]. The additional visualization of the soft-tissue data, allows correlating the position of the guide wire and catheter to anatomical information and pathologies which are only visible in the soft-tissue data. The fact that this information is available in real-time, makes it especially suitable for navigation.

The addition of soft-tissue visualization to the 3D-roadmapping technique, and especially high-quality MR datasets, brings extra information that may be important for the operators decision making and increase safety during the procedure as well as shorten the operating time. In embolisations of brain arteriovenous malformations (b-AVMs) or intracranial tumors using liquid adhesives or particles, the exact position of the catheter tip is crucial. The obvious goal is to embolise the pathological structures and avoid spilling over to normal vessel supplying normal brain tissue. The complicated vessel anatomy can in these situations be difficult to comprehend and the 3D multimodality roadmapping may in such instances prove to be of great value, especially since the 3D volume is possible to freely rotate with controls located at the interventional table. The technique may also be of great assistance for targeting areas of a b-AVM that are to be partially embolised thereby avoiding so-called piece-meal embolisation, as well as for avoiding high risk treatment close to eloquent areas of the brain. The exact position for delivery may also be important for intra-arterial delivery of other compounds i.e. cytostatic agents for tumors, growth factors for stroke and degenerative brain disorders, a field that at the moment is largely developing and growing.

The morphological MR or CT dataset holds the soft-tissue structures relevant to the procedure as well as some pathological processes that may not be visible in the 3DRA or fluoroscopy data. The most relevant parts of the soft-tissue data can be visualized by choosing a slab (see figure 9.6), whose location, orientation and thickness can be interactively altered by the operator at any time. Alternatively, it is possible to select a representation of the soft-tissue data, whereby an octant, quarter, or half is cut open (see figure 9.7). The location and orientation of the intersection can be interactively changed. The 3D-3D registration, which was calculated in the first pre-processing step, is applied to the position of the soft-tissue data.



(a) (b)

(c) (d)

Figure 9.6: *(a) An MR image, showing an AVM and impacted brain tissue, indicated by the yellow arrows, (b) the live fluoroscopy image without contrast medium shows the guide wire, but does not reveal its relation to the vasculature and the soft-tissue, (c) the fluoroscopy image mixed with the vessel tree from the 3DRA dataset adds the vascular context to the live data, (d) the fluoroscopy image, the 3DRA vasculature and a slab from the MR data. The MR slab is positioned parallel to the view port at the guide wire tip.*

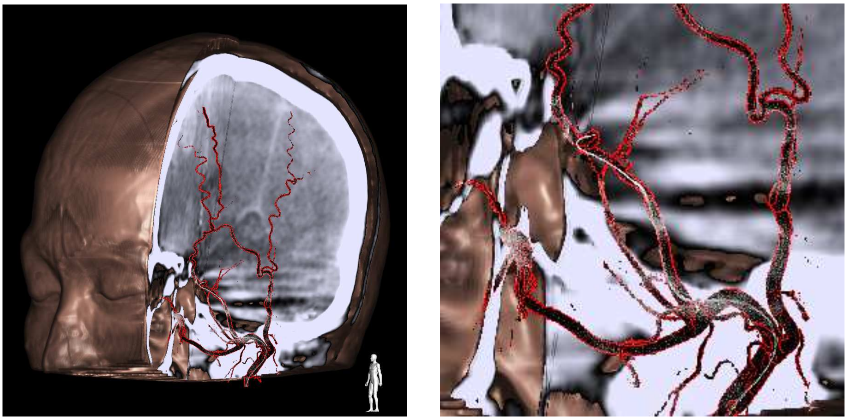

(a) (b)

Figure 9.7: *(a) A quarter is cut out of a soft-tissue dataset, while the 3DRA vessels are overlayed with the live fluoroscopy information, (b) a zoomed fragment of the left image, showing the micro guide wire.*

9.6 Conclusions

In this chapter the application of fusing real-time fluoroscopy, 3DRA data and soft-tissue data into a combined image, and its usage within neuro-endovascular procedures has been presented. The combination of the fluoroscopic image with the 3DRA vessel tree, known as 3D-roadmapping, offers the advantage that the spatial relationship between the endovascular device and the surrounding vessel morphology can be determined, without additional contrast injection, while the position of the C-arm geometry can be altered freely. The method is especially targeted at the use in minimally invasive vascular procedures, and distinguishes itself in the fact it adds contextual information to the fluoroscopy images and 3D vasculature.

The steps necessary to achieve this visualization have been described. First an image-based registration of the 3DRA dataset and the soft-tissue dataset has to be performed. We have demonstrated that the capture range is sufficient for interventional usage, and that due to the acceleration by the graphics hardware, the calculation time is very limited. The machine-based registration between the fluoroscopy image and the 3DRA data only depends on the geometry incidence angles, the X-ray source to detector distance and the calibration data. It can be easily calculated in real-time. Also we described how the visualization can be implemented to employ the possibilities of modern off-the-shelf graphic cards, allowing real-time display of the registered data with the live fluoroscopy image stream. Further possible clinical applications have been identified, and it has been demonstrated how the presented method can be employed in those applications.

The strength of the described approach lies in its real-time nature, which is primarily achieved by the on-the-fly 2D-3D registration, and the GPU-accelerated fused visualization. The interactive real-time aspect contributes to the 3D perception of the anatomy and pathologies during an intervention. The clinical feedback has been very positive; the 3D roadmapping technique is considered a valuable method for accurate navigation and helps to reduce x-ray dose and use of harmful iodine contrast agent [131, 186]. A possible disadvantage of the present method is the fact that patient motion will render the 2D-3D registration to be invalid. Therefore future work could combine machine-based registration with image-based registration to correct for patient motion.

Chapter 10

Multimodal Registration in Needle Guidance

This chapter is based on the following papers:

- Daniel Ruijters, Laurent Spelle, Jacques Moret, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. XperGuide: C-arm Needle Guidance. *In Proc. European Congress of Radiology - ECR 2008;* Vienna (Austria), C-591, March 7-11, 2008, *European Radiology,* Volume 18, Supplement 1, February 2008, p. 459. doi:10.1007/s10406-008-0003-0
- Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. Frame-less C-arm Needle Guidance. *MICCAI 2008 Workshop on Needle Steering: Recent Results and Future Opportunities,* September 6, 2008, New York (USA)
- Laurent Spelle, Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Jeremy Guillermic, and Jacques Moret. First clinical experience in applying XperGuide in embolization of jugular paragangliomas by direct intratumoral puncture. *International Journal of Computer Assisted Radiology and Surgery,* Volume 4, Number 6, November 2009, pp. 527-533. doi:10.1007/s11548-009-0370-6

10.1 Introduction

Paragangliomas, also known as glomus tumors, are highly vascularized neoplasms of neural crest origin that arise from the glomus cells, which are chemoreceptor organs in the walls of blood vessels that have a role in regulating blood pressure and blood flow. Glomus cells are located in aortic bodies near the aortic arch and the carotid bodies, situated close to the bifurcation of the carotid arteries. The glomus cells are a part of the paraganglion system composed of the extra-adrenal paraganglia of the autonomic nervous system, derived from the embryonic neural crest. Paragangliomas are most frequently located in the abdomen (85%) and the thorax (12%), and only 3% are found in the head and neck region. Glomus tumors are multiple in 25% of patients, and are usually considered benign. However, in about 3% of cases they are malignant and have the ability to metastasize [187–189].

Glomus tumors can be treated by surgical excision, radiation therapy, or a combination of those. Especially for large tumors, surgical removal is often associated with substantial intraoperative bleeding rate, due to their vascular nature [188, 190–194].

In order to reduce the intraoperative blood loss, preoperative transarterial embolization has proven to be beneficial [195–199]. However, in many cases the devascularization remains incomplete because of the extensive angioarchitecture and considerable arteriovenous shunting of the lesions. Therefore, direct percutaneous puncture and the injection of acrylic glue or cyanoacrylate has been described as an effective alternative [200–205].

In this chapter we describe a novel approach to the planning of the puncture trajectory, and the interventional needle guidance. Our method relies on C-arm fluoroscopy for the real-time guidance, while we also intend to integrate soft-tissue information, in order to use an optimal path. Since the proposed method does not rely on a stereotactic frame or markers, the strain on the patient is reduced, and the procedure duration shortened [206, 207].

10.2 Methods and materials

10.2.1 Procedural technique

Prior to patient puncturing, the optimal needle paths are drawn on a preoperative computed tomography (CT) dataset. Determination of the optimal needle trajectory is initiated by marking the ultimate needle point, located in the lesion center (figure 10.1). A line is drawn in the 3D patient space towards the skin boundary, continuously checking whether it traverses any vital anatomical structures or impenetrable bones.

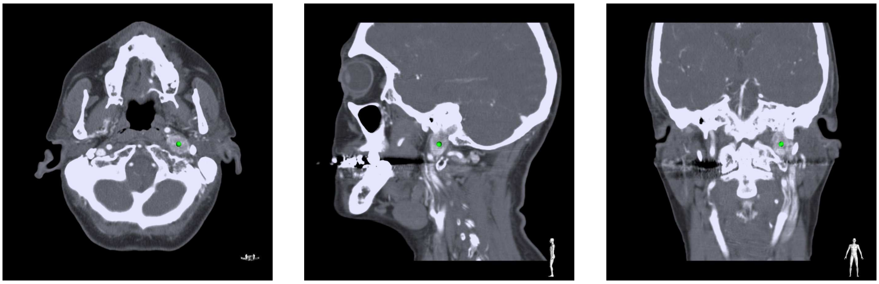
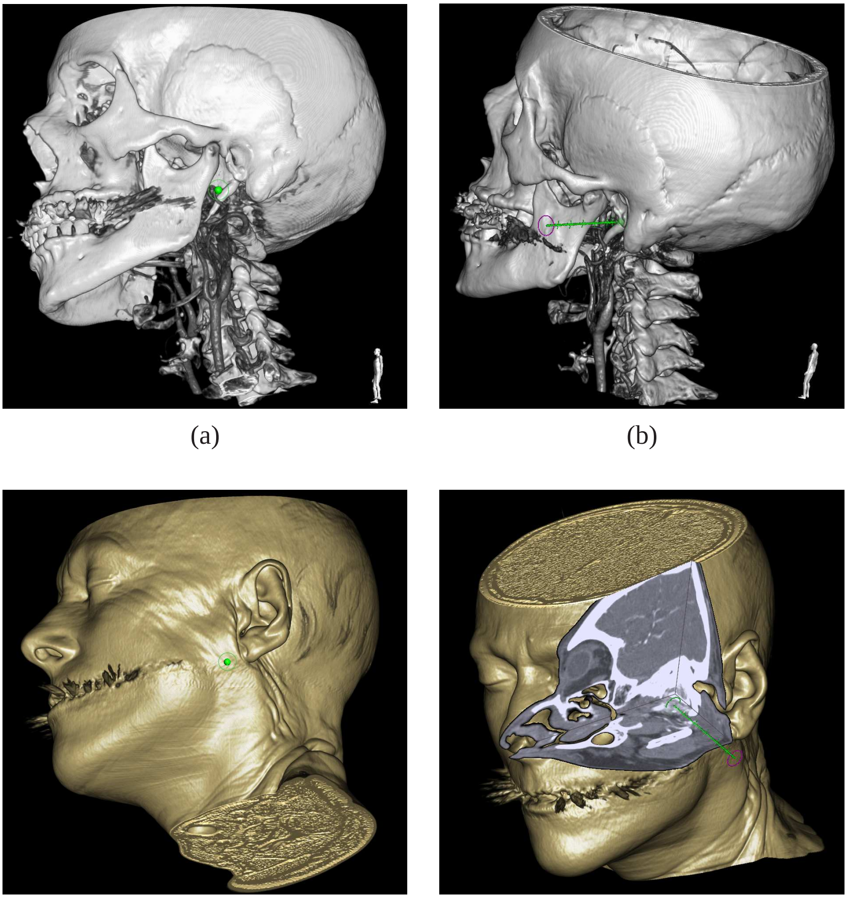

(a) (b) (c)

Figure 10.1: *The target point (green) is marked in the glomus tumor. (a) Axial view. (b) Sagittal view. (c) Coronal view.*

When the line is defined, the puncture point located on the patient skin is defined as the entry point of the virtual trajectory (figure 10.2). The inspection of the line is performed by doing soft tissue stacking perpendicular to the line's spatial location (oblique cross reformat stack). Multiple trajectories can be stored in this way. This planning phase is meant to be performed ahead of the intervention execution or periprocedurally in the case additional lesion access is needed.

At the beginning of the intervention a 3D soft-tissue cone-beam CT dataset (Philips Allura XperCT; Best, the Netherlands) is acquired with the C-arm X-ray system, see



(a) (b) (c) (d)

Figure 10.2: *(a) To establish the path to the target point a 3D view on the skull is used, in order to find a straight path without penetrating any bone tissue. (b) A planned path can be investigated from any orientation. (c) This view permits to view the entry point on the skin. (d) An octant through the planned trajectory is cut out, allowing to inspect the soft-tissue along the path.*

section 2.4. Consequently, the preoperative CT dataset is co-registered to the peri-operative cone-beam CT according to the Mutual Information criterion [109]. Since the C-arm system is used to obtain the cone-beam CT data, as well as the 2D fluoroscopy data, the relation between their respective coordinate systems is inherently known, as long as there is no patient motion. As a consequence, the image-based registration of the CT and peri-operative cone-beam CT datasets also registers the CT and C-arm coordinate systems, see section 6.6.

After the automatic registration has been completed and validated by the physician, the path vector is sent to the C-arm, and the geometry viewing incidence is steered to be tangent to the planned path: the entry view. Since this view is tangent to the needle trajectory, the path is foreshortened to a single point. When the needle is positioned at the entry position and its orientation is tangent to the fluoroscopy image, it can be inserted (figure 10.3). The C-arm viewing incidence is then steered to be perpendicular to the planned path: the progression view. In this orientation, the needle can be navigated along the planned trajectory.

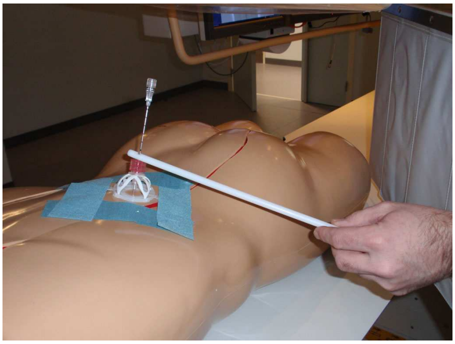


Figure 10.3: *The needle orientation is adjusted under fluoroscopy guidance to insert the needle in the back of a phantom. The physician has to take care to prevent direct X-ray radiation to his/her hands.*

The live fluoroscopy image is overlaid with the planned needle trajectory and fused with an oblique slice of the soft-tissue data, perpendicular to the viewing incidence and passing through the target point, using the method presented in chapter 4. The overlay image is real-time updated for any change in viewing incidence (L-arm angle, rotation, angulation), field of view, and source-image distance [206]. The entry view is compensated for parallax distortion. The projection of the planned path and soft-tissue information is aided considerably by the fact that modern C-arm systems use flat X-ray detectors, which do not possess any pincushion deformation of the image, contrary to their image intensifier predecessors.

The entry view and progression view steps are repeated for all planned puncture paths. The views can be selected at table side. Optionally, new paths can be planned during the intervention. After the insertion, a new cone-beam CT can be acquired and registered to verify the needle position with regard to the soft-tissue structures and anatomical landmarks.

10.2.2 Patients and materials

Two patients with a jugular paraganglioma tumor were selected for treatment according to the described method. Embolization by needle puncture was preferred over surgical excision because of the surgical treatment related difficulties: highly vascularized tumor tissue and the associated trauma. The patients were treated with percutaneous intratumoral injection of cyanoacrylate in order to embolize the lesion [205]. Each puncture was performed under high-quality X-ray roadmapping (Philips Allura Xper FD20; Best, the Netherlands). The treatment was performed under general anesthesia, which considerably reduces the risk of patient motion. Patient motion introduced in the course of the procedure would lead to misalignment of the fused image data. Catheter angiography was used to visualize the tumor location and to confirm the successful embolization of the capillary lesion network. Figure 10.4 shows examples of pre- and post-embolization vasculature. No additional imaging techniques, such as ultrasound, were used.

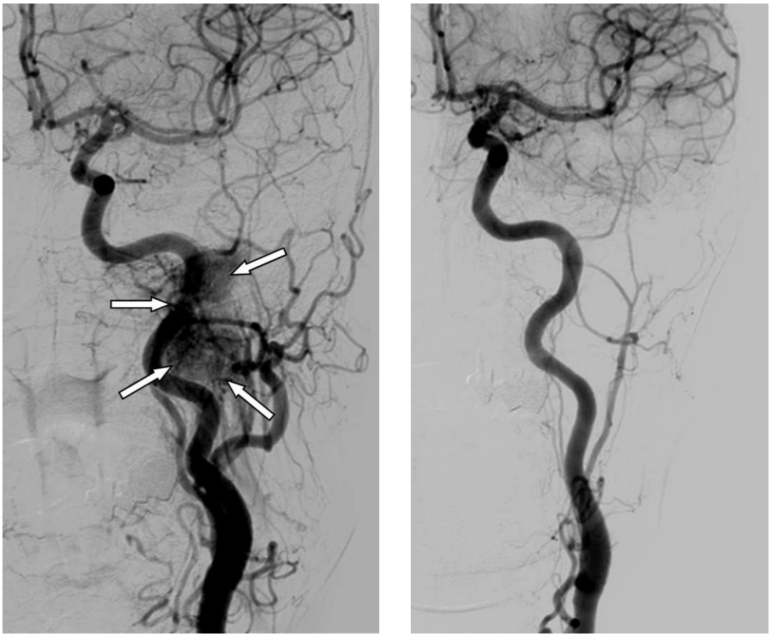

(a) (b)

Figure 10.4: *Endovascularly injected contrast medium shows the vascularization of glomus tumor (a) before, and (b) after embolization in DSA images. The tumor is indicated by the white arrows.*

For both patients two needle trajectories were planned using a preoperative CT angiography scan (16-slice Siemens Somatom Sensation, data sets consisted of 256 and 271 slices respectively of 512^2 pixels, voxel size: $0.42 \cdot 0.42 \cdot 0.70$ mm^3, H50s filter, arterial phase).

10.3 Results

The registration with the peri-operative cone-beam CT reconstruction took less than 8 seconds, due to the efficient calculation of the Mutual Information criterion by employing the processing power of the graphics hardware (figure 10.5). Maeda et al. have shown that in phantom studies a target point can be reached with a gap of 3.8 ± 1.9 mm [207]. For the two patients it proved to be possible to guide the needle within 5 mm of the planned path, using the fluoroscopy fused with soft-tissue visualization (figure 10.6). For the first patient (female, 63 years) one additional path was planned during the intervention in order to maximally embolize the tumor, and for the second patient (female, 64 years) three additional trajectories were planned.

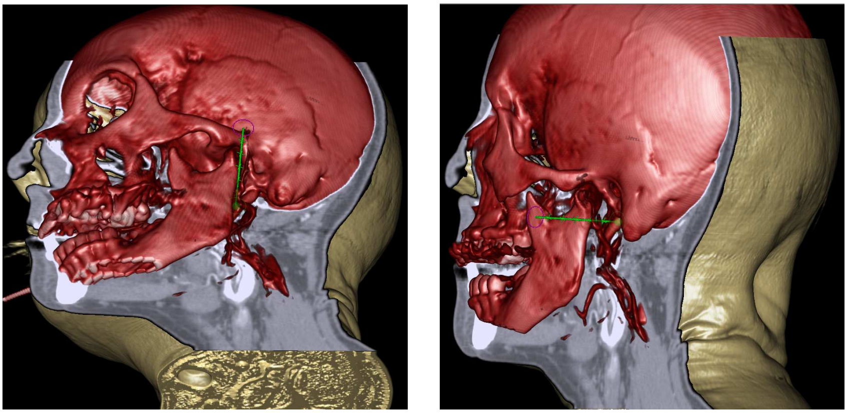

(a) (b)

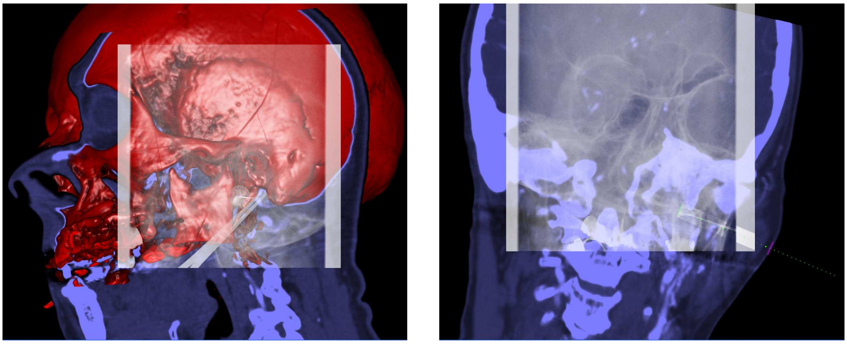
Figure 10.5: *The registered CT data (yellow) and the cone-beam CT data (red), together with the planned path. (a) Left oblique view. (b) Posterior oblique view.*

As embolic agent the currently available Onyx 18, a nonadhesive liquid embolic agent comprised of 6% EVOH copolymer dissolved in dimethylsulfoxide, was used. To puncture, a 22-gauge spinal needle (Terumo; Tokyo, Japan) was employed.

Using the described XperGuide technique allowed to steer the embolization needle with a higher confidence to the planned target locations for injection of the embolic agent and reduced the risk of puncturing the carotid artery. The availability of the real-time position of the needle over the planned needle path (and any deviations) and the anatomical landmarks in the CT dataset reduces the need for intermediate angiography, and therefore reduces the use of iodine contrast medium and X-ray dose compared to traditional fluoroscopy guided direct puncturing. No post procedure complication was established during the one year checkup.

10.4 Discussion

Ultrasound guidance is considered as the first line imaging technique while performing needle punctures. However, due to the presence of the massive occipital skull



(a) (b)

Figure 10.6: *(a) Entry point view, showing the real-time fluoroscopy image (inner white square overlaid image), the soft-tissue (blue), the skull (red), and the bull's eye target point. The needle is being positioned for entry. When the needle is foreshortened to a single point at the bull's eye it can be inserted. (b) Progression view, showing the real-time fluoroscopy image, the soft-tissue and the planned path. Any vertical deviation from the path can be monitored. In-plane deviations can be checked by switching back to the entry point view.*

base bone and ultrasound interference with the bony anatomy, other imaging modalities are used for guidance in the head and neck region, such as static CT images, CT fluoroscopy, X-ray fluoroscopy, or optionally stereotactic navigation. All mentioned approaches possess their limitations; CT based procedures are limited by the patient access area within the gantry. Additionally, the needle path that can be planned and tracked is restricted to the axial planes, imaged by the CT modality. Static CT images further lack real time feedback. Another option is X-ray fluoroscopy, which produces less X-ray dose and offers fewer restrictions in patient access compared to CT fluoroscopy. However, this modality does not provide any soft-tissue information.

The fluoroscopy navigation overlaid with the planned path, as proposed in this chapter, has been shown to be an accurate tool for needle guidance. The procedure is performed in the angio lab, using C-arm fluoroscopy. No additional navigation equipment, special devices or special needles are required, which means that there is no necessity to invest in additional specialized and unfamiliar equipment and training, delivering a cost-efficient procedure. The fact that the presented method does not use any stereotactic frame or markers reduces the strain on the patient and facilitates the work flow management. The procedure can be carried out more efficiently, compared to CT guidance.

The described technique offers the advantage over traditional angiographic X-ray guided punctures that the needle is accurately inserted along a path, which was planned on a three-dimensional soft-tissue dataset. Furthermore, the soft-tissue data, as well as the planned needle path, are visualized together with the real-time fluoroscopic image of the needle that is being inserted. The presence of this combined information increases the confidence during guidance and allows for a more accurate deliverance of the embolic agent at the destination location in the tumor.

The patient pose differed between the preoperative CT and the fluoroscopy guided intervention, in order to obtain optimal access to the planned trajectory proximate to the ear (figure 10.7), but this did not form a complicating factor. The registration step was not hindered by the difference in pose, and conveyed the planned paths and CT soft-tissue information into the coordinate system of the C-arm. The needle accessibility of an intracranial location, however, can be limited by the topology of the skull.

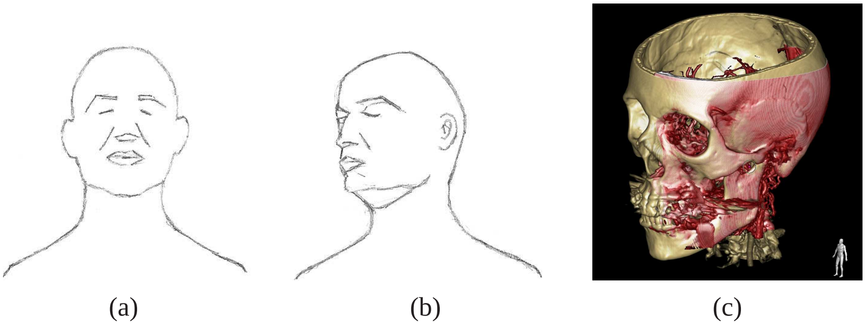


Figure 10.7: *(a) Patient pose in the CT scan. (b) Patient pose on the C-arm table; the head is tilted to gain better accessibility to the needle entry position near the ear. (c) Registered CT data (yellow) and C-arm generated cone-beam CT data (red). The registration can be performed without any manual initialization or interaction.*

10.5 Conclusions

We present a method for planning and guiding needle insertion by combining X-ray C-arm fluoroscopy and 3D soft-tissue information. The entry point view and the progression view together allow a complete assessment of the present needle position with regard to the planned trajectory. The fusion with the soft-tissue dataset incorporates information that is missing in the fluoroscopy image in a readily accessible manner.

Since all the involved equipment is already available in the angio suite, and there are no additional constraints to the pre-interventional CT acquisition, the described method can be easily and cost-efficiently incorporated. From the feedback from the clinical users from various hospitals it can be concluded that the described method provides a higher degree of confidence during the procedure, because the planning, pre-interventional soft-tissue data and the live fluoroscopic tracking of the needle is accurately presented in a fused image. The procedure is considered to be easy to use when the hospital staff is adequately trained. First clinical experience in applying the proposed guidance in the percutaneous embolization of paragangliomas by intratumoral needle injection of an embolic agent has been obtained. The procedure is considered to be sufficiently accurate, successful and aids in reducing the procedural time.

Chapter 11

Multimodal Registration for Coronary Artery Disease Interventions

This chapter is partially based on the following papers:

- Daniel Ruijters, Niels H. Bakker, Onno Wink, Bart M. ter Haar Romeny, and Paul Suetens. CT TrueView - Cardiac CT in the Cathlab. *In Proc. Annual Symposium of the IEEE/EMBS Benelux Chapter,* December 6-7, 2007, Heeze (the Netherlands), pp. 38-41
- Daniel Ruijters, Niels H. Bakker, Onno Wink, Bart M. ter Haar Romeny, and Paul Suetens. Integrating CT in Minimally Invasive Treatment of the Coronary Arteries. *In Proc. European Congress of Radiology - ECR 2008;* Vienna (Austria), C-200, March 7-11, 2008, *European Radiology,* Volume 18, Supplement 1, February 2008, p. 378. doi:10.1007/s10406-008-0003-0
- Joel A. Garcia, Shyam Bhakta, Joseph Kay, Kak-Chen Chan, Onno Wink, Danny Ruijters, and John D. Carroll. On-line multi-slice computed tomography interactive overlay with conventional X-ray: A new and advanced imaging fusion concept. *International Journal of Cardiology,* Volume 133, Issue 3, April 17, 2009, pp. e101-e105.
  doi:10.1016/j.ijcard.2007.11.049
- Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Vesselness-based 2D-3D registration of the coronary arteries. *International Journal of Computer Assisted Radiology and Surgery,* Volume 4, Number 4, June 2009, pp. 391-397. doi:10.1007/s11548-009-0316-z
- Onno Wink, Harvey S. Hecht, and Daniel Ruijters. Coronary Computed Tomographic Angiography in the Cardiac Catheterization Laboratory: Current Applications and Future Developments. *Cardiology Clinics, Advances in Coronary Angiography,* edited by S. J. Chen and J. D. Carroll, Volume 27, Issue 3, August 2009, pp. 513-529.
  doi:10.1016/j.ccl.2009.04.002

11.1 Introduction

Coronary Artery Disease (CAD) is a condition in which plaque builds up inside the coronary arteries, which supply the myocardium (heart muscle) with oxygen-rich blood. Plaque consists of fat, cholesterol, calcium, and other substances found in the blood. When plaque builds up in the arteries, the condition is called atherosclerosis.

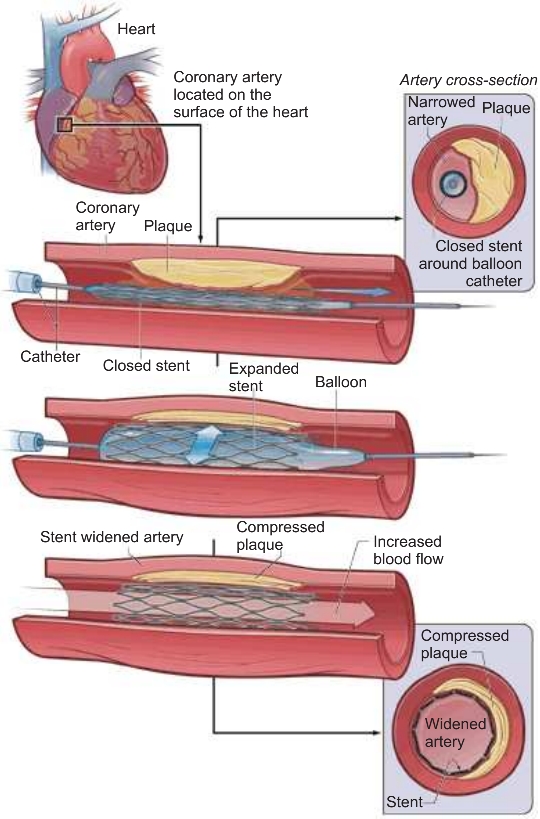


Figure 11.1: *Stent delivery to a stenotic artery by percutaneous catheterization [208].*

The obstructed blood supply to the myocardium can lead to fatigue, shortness of breath and chest pain (known as angina). It also increases the likelihood that blood cloths will form in the arteries. A blood cloth or the acute rupture of plaque can partially or completely block the blood flow. A sudden complete obstruction may lead to an acute myocardial infarction (heart attack). Over time, CAD can weaken the heart muscle and lead to heart failure and arrhythmias [208]. While the symptoms and signs of coronary artery disease are noted in the advanced state of the disease, most individuals with coronary artery disease show no evidence of disease for decades as the disease progresses. CAD is the leading cause of death worldwide. The treatment options are: medication, coronary artery bypass surgery, or percutaneous intervention (figure 11.1).

A Chronic Total Occlusion (CTO) is defined as an artery that has been completely occluded for more than 30 days. Medical therapy (*e.g.*, cholesterol lowering medications, beta-blockers, nitroglycerin, calcium antagonists, *etc*.) is partially efficacious, but rarely completely eliminates either the symptoms or the objective evidence of the ischemia. Surgical treatment involving Coronary Artery Bypass Grafting (CABG) is effective as long as the distal target vessel is anatomically suitable for insertion of a bypass graft. The limitations of the bypass surgery are well known and include risk of surgical mortality, and significant expense.

Another treatment option is Percutaneous Coronary Intervention (PCI). This minimally invasive, less costly procedure is accomplished by using conventional guidewire techniques to slowly 'poke' through the occlusion. The time spent to recanalize a chronic total occlusion is estimated to be between 5 minutes and several hours with an average time of about 30 minutes. Percutaneous intervention of CTOs accounts for 10% to 15% of all angioplasties; however, after successful recanalization, there is an increased rate of subsequent restenosis and reocclusion compared with nonocclusive stenoses. Although several randomized trials demonstrated the efficacy of stent implantation over balloon-only angioplasty, even with stents there remains a significant rate of both restenosis (32% to 55%) and reocclusion (8% to 12%). Chronically occluded coronary arteries account for approximately 20-30% of the documented coronary disease encountered in coronary catheterization labs today [209–212].

The lack of anterograde blood flow in the totally occluded vessel segment, however, prevents the angiographic visualization using catheter injected iodine contrast medium during angioplasty. In case there is no or little calcified plaque present, there are no landmarks that indicate the location of the occluded vessel during fluoroscopic navigation and treatment of the diseased vessel segment. The absence of anterograde flow has been reported as one of the factors that can lead to procedural failure [213].

Due to the fact that contrast medium is injected intravenously, Computed Tomography (CT) visualizes both anterograde and retrograde filled vessel segments. Furthermore, because of its high contrast resolution, CT can also visualize soft plaques sections. As a result it is possible to identify also the occluded vessel segments in the CT reconstruction. In this chapter we describe the clinical experience with integrating pre-interventional coronary CT angiography and peri-interventional X-ray fluoroscopy in the interventional treatment of CAD. The CTA and X-ray images are presented in a fused visualization, using the techniques of Chapter 4, in order to guide the navigation and deployment of intravascular devices through the diseased coronary arteries, especially for CTO cases.

11.2 Methods and materials

11.2.1 Pre-interventional acquisition and analysis

Recent years have seen considerable advancements in the detailed imaging of the cardiac anatomy using CT reconstructions. The newer 256-slice (Brilliance iCT, Philips Healthcare) and 320-slice (Aquilion One, Toshiba Medical Systems) CT scanners are able to cover the whole heart volume in one or two rotations, which leads to considerable gains in temporal and spatial resolution and reduces motion artifacts. These developments especially enable high-quality coronary imaging with a significant reduction in radiation dose [214]. Prospectively gated axial imaging for coronary CT angiography (or "step and shoot"), with radiation applied only during the middiastolic coronary rest phase, has demonstrated radiation dose savings of greater than 75% compared with the traditional technique of helical retrospective gating while

Table 11.1: *Correlation of 64-slice computed tomography and invasive angiography for the detection of greater than 50% coronary obstruction.*

Study	n	Sensitivity	Specificity	Negative Predictive Value
Leschka *et al*. [221]	67	94%	97%	99%
Leber *et al*. [222]	59	73%	97%	99%
Mollet *et al*. [223]	52	99%	95%	99%
Raff *et al*. [224]	70	86%	95%	98%
Ropers *et al*. [225]	82	95%	93%	99%
Fine *et al*. [226]	66	95%	96%	95%
Weighted average		90%	95%	98%

maintaining image quality [215–217] and with a diagnostic accuracy greater than 96% [218, 219]. Currently, coronary CT angiography is most renowned for its negative predictive value, *i.e.*, for its ability to rule out the presence of (severe) coronary stenosis, see table 11.1 [220].

The fused visualization requires that the CT and X-ray image data are registered with the objective to find the spatial mapping between both datasets. In order to perform the automatic registration of the CT data and X-ray fluoroscopy images (see Chapter 8) and the intra-interventional visualization, the coronary arteries need to be segmented from the CT data. There is a vast amount of literature on this subject, see *e.g.*, [176–179]. The extraction of the coronary arteries from the cardiac CT data is generally started after the whole heart segmentation has been performed because it can provide clues to the location of the coronary arteries. The first step consists of extracting the tubular structures in the raw CT data. To facilitate this extraction process a separate representation is often generated in which the vessel-like structures are highlighted. An example of such a “vesselness” [14] filtered image is shown in figure 11.2 The coronary artery tree is then traced in the vessel-enhanced data starting at the ostia, which could have been located during the preceding whole heart segmentation. The segmentation algorithm tries to follow the enhanced structures until it reaches the distal end of the vessel or the signal- or contrast-to-noise reaches a threshold. Most commercially available applications are capable of extracting the major coronary arteries, but let the operator provide the appropriate labels. It is always possible to edit and extend the automatically found centerlines or to trace the entire vessels manually.

The segmentation of the coronary arteries in the CT data allows a detailed analysis of the vessel lumen, stenosis and associated plaque tissue. Highly calcified regions can be marked to show the extent of the lesion. The lesion diameter can be quantified, however the derived percent diameter stenosis is usually rounded to the encompassing quartile (*e.g.*, 50%-75%) to cope with the limitations of the spatial resolution. The often tortuous nature of the vessels does not allow them to be captured in their entirety in a single cut through the volume, even with a large slab size. This can be overcome

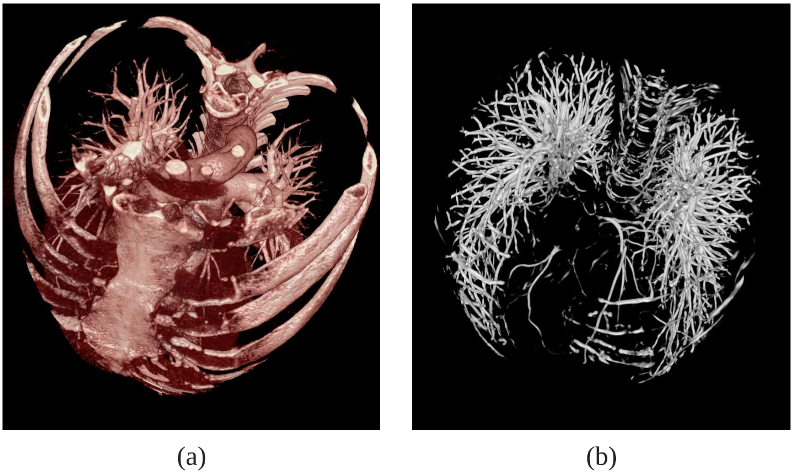

(a) (b)

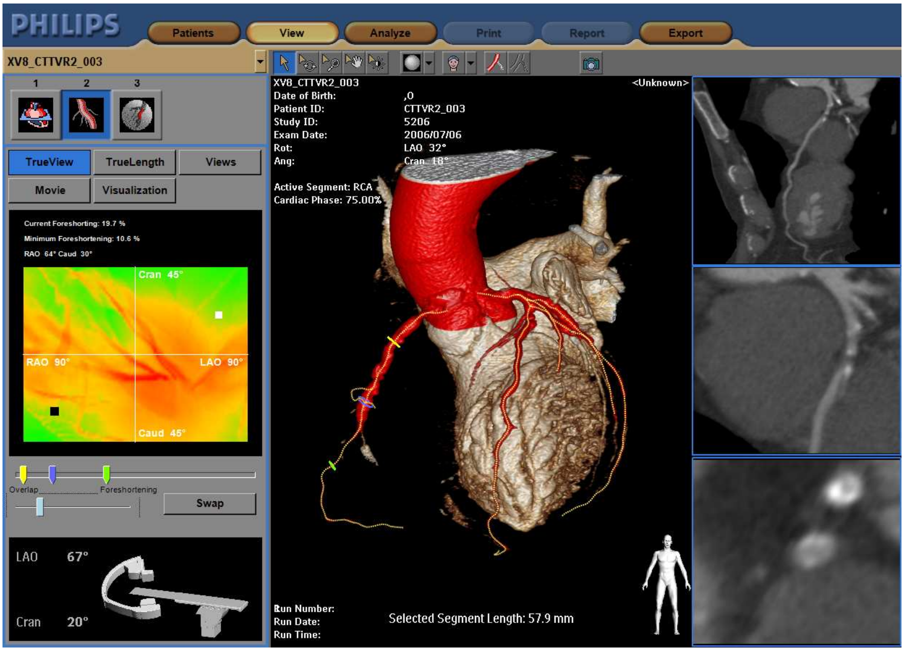
Figure 11.2: *Example of a vessel-enhanced image. (a) A volume rendering of the original CT dataset. (b) The same dataset after a vesselness filter was applied. The major coronary arteries are visible in the center of the volume.*

by using a curved Multi-Planar Reformat (MPR) to enable the visualization of the entire course of the vessel, along with providing true distance measurements that are not subject to foreshortening. The curved MPR is based on fitting a curved plane through the centerline of a selected segmented artery.

11.2.2 Peri-interventional use

Because the ability for in-room manipulation of the CT data and their derived information by the interventionalist during the diagnostic and PCI portion is limited, it is important to show the essentials of the CT information during the critical portions of the procedure. Figure 11.3 shows the segmented CT data as it is being displayed in the cathlab. It is essential that the physician has the possibility to operate the system directly from the patient table side. The operator has the opportunity to select the coronary artery of interest and define the vessel segment of interest from the table side. In the lower left panel the C-arm configuration is shown that corresponds with the current viewing angle of the CT dataset. The orientation of the rendered views can be coupled to follow the C-arm geometry viewing incidence in real-time.

Spatial foreshortening is the distortion of geometrical structures (*e.g.*, vessels) when depicted at an angle (figure 11.11). Foreshortening of the vessel geometry in X-ray images makes it difficult to asses their true length, and therefore it is preferable to select X-ray projection views that have minimal foreshortening for the vessel segments of interest. In order to assist the physician in selecting C-arm views with least foreshortening and vessel overlap, an optimal view map is generated (figure 11.4) [227]. The vertical axis of this map represents the angulation of the C-arm system, and the horizontal axis the rotation. The color of a point on the optimal view map represents the amount of foreshortening of the selected coronary segment, and



XV8_CTTVR2_003

XV8_CTTVR2_003
Date of Birth: ,0
Patient ID: CTTVR2_003
Study ID: 5206
Exam Date: 2006/07/06
Rot: LAO 32°
Ang: Cran 18°

Active Segment: RCA
Cardiac Phase: 75.00%

Current Foreshorting: 19.7 %
Minimum Foreshortening: 10.6 %
RAO 64° Caud 30°

LAO 67°
Cran 20°

Run Number:
Run Date:
Run Time:

Selected Segment Length: 57.9 mm

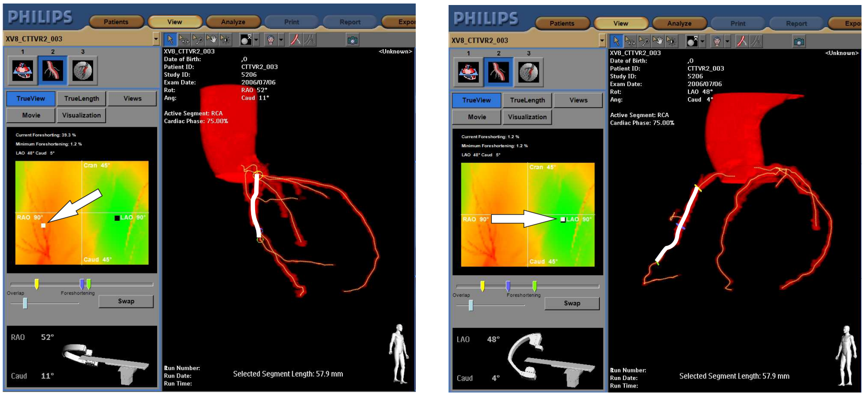
Figure 11.3: *In-room presentation of the segmented CT data. In the middle the aorta trunk, the coronary arteries (red) and in this case the left ventricle are shown. The curved MPRs at the right correspond to the selected vessel segment.*

its likelihood to overlap with other vessel branches. Based on this optimal view map a series of C-arm angles may be chosen to be used during the intervention. These angles may be different than the routine views commonly used by the cardiologist. The C-arm can be steered to the selected angles, using the principles described in section 6.6 and appendix A.

After a registration between the segmented CT data and a selected X-ray image has been performed, as discussed in chapter 8, the live X-ray fluoroscopy image stream can be displayed fused to the CT data in real-time, as is shown in figure 11.5. Ultimately, the cardiac phase of the CT image would be matched to the cardiac phase of the radiographic data. Currently the CT images are shown in a preselected static cardiac phase; the coronary arteries in the X-ray image display a periodic motion around the coronary arteries, segmented from the CT data.

11.3 Results

A typical example of CTO revascularization using image fusion is demonstrated in figure 11.6. The catheter coronary angiography reveals the vessel cutoff, whereas the superimposed CT images from a similar viewing angle demonstrate the occluded segment and the remainder of the left circumflex artery, which is filled by collaterals. Successful restoration of flow was accomplished by the antegrade technique [210],



XV8_CTTVR2_003
Current Foreshorting: 39.3 %
Minimum Foreshortening: 1.2 %
LAO 48° Caud 5°
RAO 52°
Caud 11°
XV8_CTTVR2_003
Date of Birth: ,0
Patient ID: CTTVR2_003
Study ID: 5206
Exam Date: 2006/07/06
Rot: RAO 52°
Ang: Caud 11°
Active Segment: RCA
Cardiac Phase: 75.00%
Run Number:
Run Date:
Run Time:
Selected Segment Length: 57.9 mm

XV8_CTTVR2_003
Current Foreshorting: 1.2 %
Minimum Foreshortening: 1.2 %
LAO 48° Caud 5°
LAO 48°
Caud 4°
XV8_CTTVR2_003
Date of Birth: ,0
Patient ID: CTTVR2_003
Study ID: 5206
Exam Date: 2006/07/06
Rot: LAO 48°
Ang: Caud 4°
Active Segment: RCA
Cardiac Phase: 75.00%
Run Number:
Run Date:
Run Time:
Selected Segment Length: 57.9 mm

(a) (b)

Figure 11.4: *The optimal view map is based on the vessel segment indicated by the white line. The white arrow points to the spot on the optimal view map that corresponds to the current viewing angle. (a) This suboptimal viewing angle leads to a considerable amount of foreshortening and a lot of overlapping vessels. (b) This viewing angle delivers least foreshortening, while the vessel segment of interest does not overlap with other branches.*

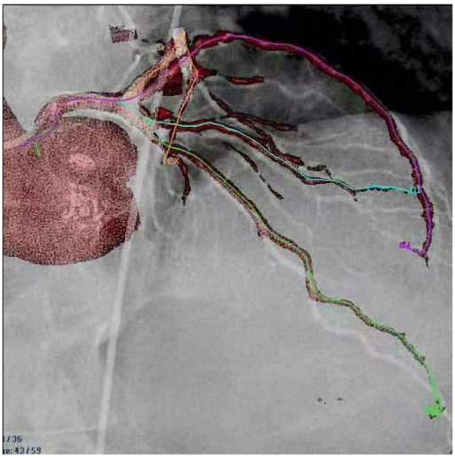


Figure 11.5: *A fused visualization of the coronary arteries, segment in a 3D CT dataset (red), and 2D X-ray angiography image (grey). The combined visualization allows the correlation of the vessels in the CT data (and pre-interventional annotations) and the peri-interventional X-ray by the observer.*

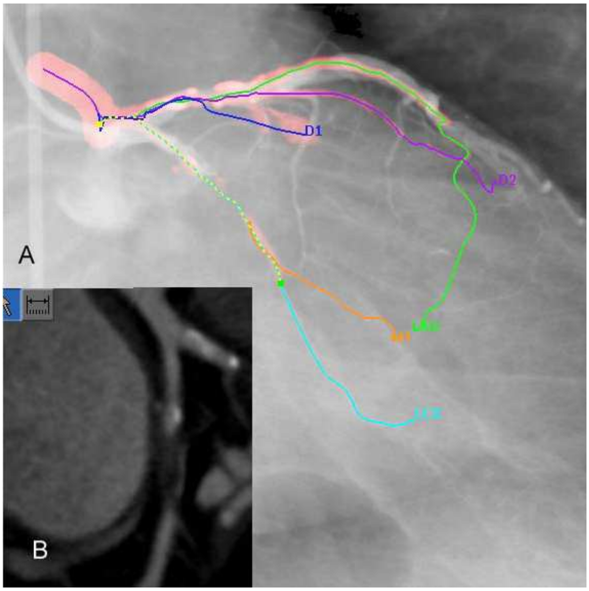
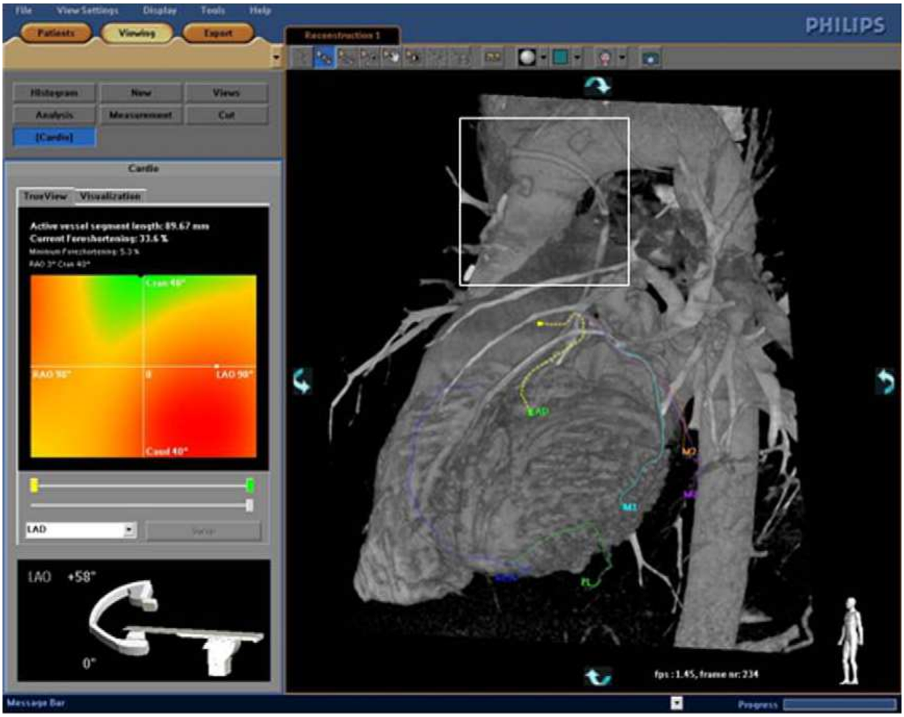


Figure 11.6: *(a) Fusion of CT (red) and X-ray images (grey) for navigation in a CTO vessel. The catheter injected contrast medium (white) does not enter the left circumflex (LCX), whereas the path is still visible from the CT data. (b) The corresponding curved MPR, showing the CTO and the retrograde filled continuation of the vessel.*

which means that the catheter was progressed in the direction of the bloodstream.

11.3.1 Case report

A 40-year-old male with a complex cardiovascular history presented with atypical anginal symptoms. Four years back he presented with a brain abscess which required drainage. He subsequently developed aortic valve endocarditis and underwent aortic valve replacement, ascending aorta replacement, and reimplantation of his coronary arteries. He reportedly had an intraoperative left anterior descending artery (LAD) injury which required the placement of a saphenous vein graft (SVG) to the LAD. No preoperative coronary angiography was performed. Soon after discharge he presented to another hospital with an acute coronary syndrome and positive cardiac biomarkers. A 90-percent lesion of his proximal right coronary artery (RCA) was stented. A week later, the patient presented with chest pain and positive cardiac biomarkers. A repeat intervention of the RCA was performed and due to a distal edge dissection at the previously placed stent another stent was implanted and overlapped with the previous stent. Given his complicated cardiovascular history and presentation the decision was made to perform a coronary CTA. The CTA suggested 50-75% in-stent restenosis of the mid-RCA stents and prompted invasive coronary angiography, which revealed that the RCA had only mild irregularities with widely patent stents. The LAD and the left circumflex artery had only mild luminal changes. The graft to the



Active vessel segment length: 89.67 mm
Current Foreshortening: 33.6 %
Minimum Foreshortening: 5.3 %
RAO 3° Cran 40°

Figure 11.7: *Computed tomography showing the saphenous vein graft (SVG) location in an unusual superior and anterior position due to a surgical repair of the aorta. Inside the white box note the origin of the SVG in relation to the aorta and the proximity to the innominate artery. The centerline prediction of the position of the coronary arteries in the left anterior oblique position is shown. LAD = left anterior descending artery; RCA = right coronary artery; M1 = first obtuse marginal; M2 = second obtuse marginal.*

LAD was not visualized as in prior procedures despite aortography in two planes. Competitive flow along the mid-LAD suggested a patent SVG. The overlay of the CT image (figure 11.7) on X-ray (figures 11.8 and 11.9) successfully allowed for the cannulation of the vein graft to the LAD, which, due to the distorted anatomy of the aorta, was in an unusually high and anterior location (figure 11.10) [228]. The fusion of the CT data with the live fluoroscopy image stream provides more confidence and a higher accuracy during the navigation and deployment of the intravascular devices. Especially for complex pathologies and anatomical deformation the roadmap provided by the fused CT morphology is highly valuable.

11.4 Discussion

The views with least foreshortening and overlap with other vessel branches with respect to a chosen vessel segment can be planned before or during the intervention, using the information in the optimal view map (see figure 11.4). Being able to se-

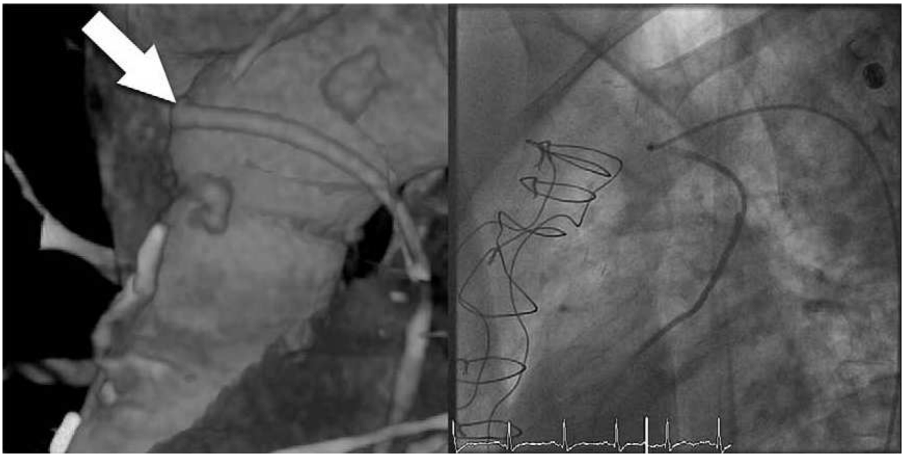


Figure 11.8: *Magnified image of the ostium of the vein graft (arrow) by CT (left) and the X-ray angiography acquired from the same viewing angle (right).*

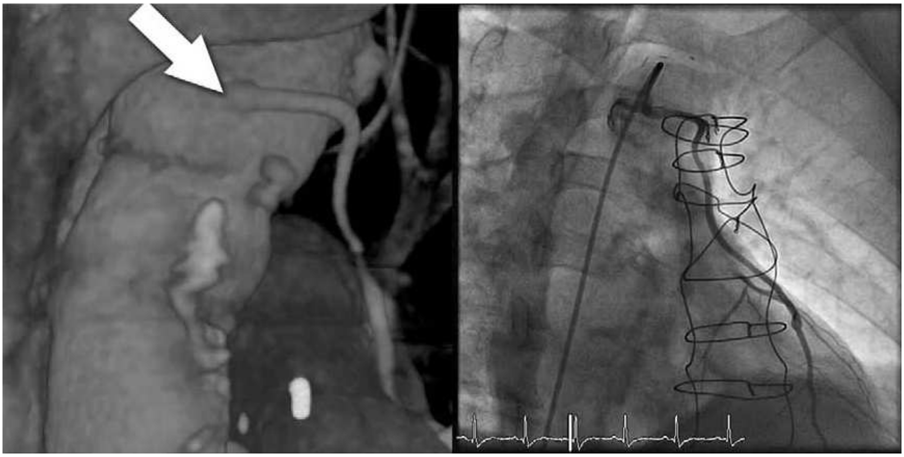


Figure 11.9: *Rendering of the ostium (arrow) of the SVG and the X-ray angiogram from a different angle in the CT (left) and the corresponding X-ray viewing angle (right).*

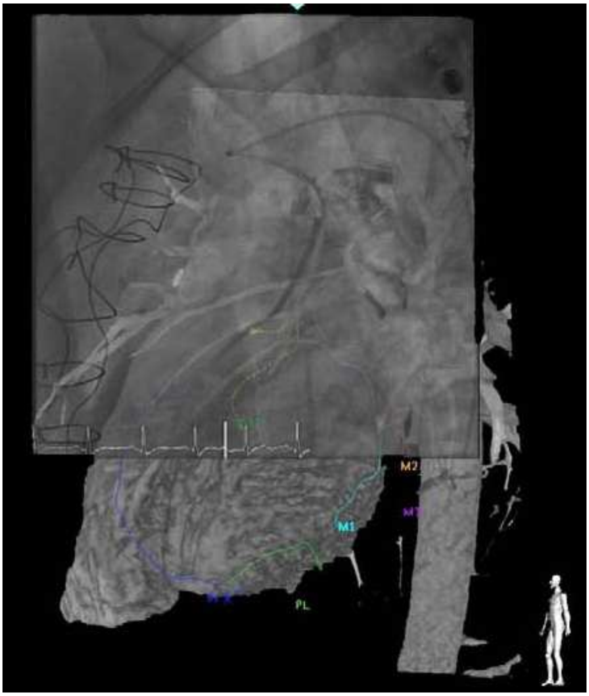


Figure 11.10: *Computed tomography background with a live X-ray overlay showing the CT derived position of the SVG and the subsequent successful cannulation and injection under fluoroscopy.*

lect the most appropriate view immediately helps to save contrast agent and radiation dose, and also provides more optimal views for best positioning and deployment of the intravascular devices, such as stents. Suboptimal views, leading to foreshortening (figure 11.11), may partially account for the suboptimal sensitivity and specificity of coronary angiography [229]

Coupling the in-room presentation of the segmented coronary CT data and associated curved MPR representations with the viewing incidence of the C-arm geometry in real-time provides information of the current viewing angle without administration of additional contrast medium and X-ray dose.

Image fusion of CT and live X-ray fluoroscopy has several clinical applications, such as radiotherapy planning and verification, surgery planning and guidance, and minimally invasive vascular treatment in peripheral and neurovascular interventions. The application of image fusion to coronary interventions is challenged by cardiac and respiratory motion, and only recently efforts in this domain have been reported [127, 214, 228, 230]. Although the live X-ray images of the coronary arteries display a periodic movement around the static CT image, it is still considered valuable for navigation purposes. Especially for CTO cases, where the vessel distal to the occlusion is completely hidden in the X-ray image, the fused image provides a useful road map.

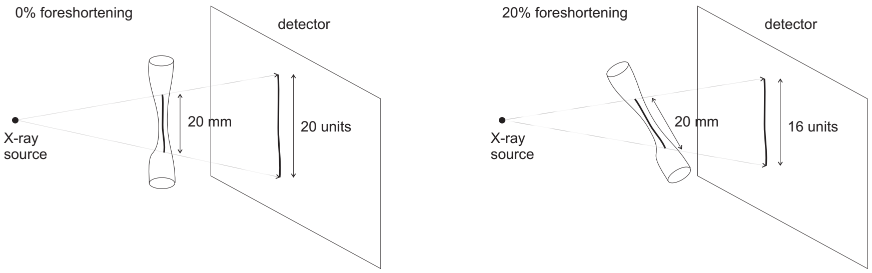


Figure 11.11: *Perpendicular orientation of the X-ray beam to a vessel segment or lesion of interest results in minimal distortion of the projection image (left). Vessel foreshortening results in suboptimal projection images that misrepresent the true length of a lesion or vessel segment (right). Adapted from [229]*

11.5 Conclusions

It is to be expected that with the improvements being achieved in coronary CT angiography with respect to increasing temporal and spatial resolution at a decreasing radiation dose, CT will gain a more prominent role during diagnosis of Coronary Artery Disease (CAD), especially for the more complex cases. Along with improvements in the CT imaging techniques, the tools for automatic analysis become ever more sophisticated. When this detailed information is available at the diagnostic stage, the desire to integrate it in the treatment course is a logical step. In this chapter first clinical experiences with image fusion during the treatment of CAD have been reported. We have embedded the registration algorithm, presented in chapter 8 in an integrated clinical application that allows to fuse the live X-ray fluoroscopy images and the diagnostic CT data in a single fused image. The system can be operated from the patient table side in an intuitive manner. First clinical feedback has been positive, since the CT data helps the physician during guidance and deployment of the intravascular devices.

Chapter 12

Conclusions

It was the objective of this thesis to combine multiple image datasets into a coherent fused visualization to guide minimally invasive treatment, assuring usable and cognitively adequate interaction and interpretation by the clinician. In this work this goal has been filled in by focussing on practical technical solutions, without loss of general applicability. In this context a number of fundamental concepts have been developed that enable the integration of multimodal data in image guided interventions and therapy (IGIT). Fast volumetric visualization and registration of multimodal data has been explored in order to achieve the realtime integrated fused image guidance during interventional treatment. The developed methods have been validated in a number of concrete clinical applications, which serve to demonstrate the general applicability of the presented concepts.

The methods that have been presented in this thesis can be divided into two main categories: visualization techniques and registration techniques. For the fast visualization of volumetric datasets a double space-skipping hierarchy has been developed. This double hierarchy has been based on the analysis of the GPU hardware pipeline. Each hierarchal level addressed a bottleneck in this pipeline and can be tailored to optimally leverage its data throughput. Due to this approach, the maximal rendering performance can be reached without sacrificing any image quality. Especially for visualization during image-guided interventions it is of the greatest importance that image rendering, which is fused with real-time acquired clinical data, is instantaneous and allows for fluid and interactive manipulation.

Several practical approaches for fusing volumetric and X-ray projection data have been presented here. Overlaying the silhouette of the 3D datasets allows to indicate the shape of obscured parts of a dataset, while maintaining an image that is easy to interpret at the same time. The use of the stencil buffer allows to process the real-time projection image differently, depending on the underlying 3D data. These methods enable the presentation of easy-to-read images during the intervention without occupying any significant additional processing resources.

Autostereoscopic displays provide depth perception without any external aids, such as goggles. Their added value in the intervention room concerns the intuitive interpretation of 3D data. As a consequence of this improved 3D presentation there is a reduced need for interactive manipulation with such data, which provides the clinician with more time for other tasks. The rendering of images for such autostereoscopic screens demands a lot of computation resources, since the 3D scene has to be depicted for each of the views that are emitted by the autostereoscopic display. In this work it has been shown how the GPU can applied to efficiently render to such displays and how the frame rates and the resolution can be balanced when the processing resources are scarce.

Concerning the subject of registration, the application of the GPU has been investigated to accelerate the process of elastic image registration. Elastic registration typically requires considerable computation times, and thus its acceleration would aid the utilization during clinical procedures. The improvements in computation times amounted up to a factor 50, using the proposed method. We investigated the precision and performance aspects of the GPU in the efficient evaluation of uniform cubic B-splines, which are employed in the registration process. Especially for clinical applications it is essential to know the (im)precision of the used algorithms and to assess its impact on the clinical results.

A novel similarity measure has been developed for the purpose of registering coronary arteries, which were imaged using CT and the X-ray C-arm. The similarity measure does not require a segmentation of the live interventional X-ray image. It uses the vesselness filter to enhance the vessel structures in the X-ray image [14]. The similarity is then obtained by calculating the dot product of the distance transform of the projected CT vessel centerlines with the response of the vesselness filter. Our validation tests proved that it performes robustly and accurately for the given task.

The clinical applications that were developed and investigated in the context of this thesis concerned minimally invasive treatment using the X-ray C-arm. All applications used the fast volume rendering and data fusion that were introduced in chapters 3 and 4. The first clinical application that has been described dealt with the guidance and roadmapping of the catheter in the treatment of arteriovenous malformations (AVM). For this purpose a pre-interventional MR dataset, clearly depicting the AVM, was registered with a peri-interventional 3D-RA reconstruction using a GPU-accelerated mutual information criterion. The real-time fluoroscopy image stream was then overlaid on the fused 3D data, using the machine-based registration described in section 6.6. The roadmap information provided by the 3D-RA vasculature allows to reduce the amount of harmful contrast agent and provides more insight in the 3D topology of the vessel tree. The fused MR delivers additional information regarding the AVM location, affected tissue and feeding vessels.

A further clinical application that has been researched addressed the percutaneous embolization of skull base paragangliomas (glomus tumors) through direct needle punctures. The needle path was planned on a pre-interventional CT dataset, which was registered with a peri-interventional cone-beam CT reconstruction of the patient's head. This registration also brought the treatment planning into the frame of reference of the C-arm system. The fused 3D data, together with the planned needle trajectory, could then be visualized together with the live fluoroscopy images. The insertion point and needle orientation were determined and guided by the real-time C-arm images, fused with the planned data. Also the progression of the needle insertion was monitored from a viewing angle that was determined by the planned path. It proved to be possible to execute the clinical procedure in an accurate and safe fashion, using the described techniques.

The final clinical application that has been described concerned the use of CT data for roadmapping purposes during the treatment of chronic total occlusion (CTO) of the coronary arteries. For this procedure the coronary arteries were segmented in a pre-interventional CT dataset. The segmented arteries then were registered with peri-interventional X-ray angiography images, using the vesselness-based method, defined in chapter 8. During the advancement and deployment of the intra-vascular devices the live X-ray image stream is overlaid on the static segmented CT data. This procedure proved to aid the navigation for the vessel segment distal to the occlusion, and increased the confidence during the guidance of the intra-vascular devices.

A large amount of coordinate spaces with dynamically changing spatial relations is inherent to the integration of multiple image datasets during a clinical intervention. We have introduced a coordinate space framework, described in appendix A, that allows to administrate any number of such coordinate spaces in a transparent and maintainable manner. The utilization of such a framework eases the conception of new applications for minimally invasive procedures, and reduces the risk of programming errors. As such, it contributes to the target of developing techniques that are generally applicable to image guided interventions. The framework has proven its added value in several large software projects at Philips Healthcare.

All algorithms were implemented in C++ code and embedded in clinical prototypes that could be operated by the clinical staff. The prototype software packages have been clinically investigated at the following hospitals: the Karolinska Institutet in Stockholm, Sweden, the Fondation Rotchild Hospital in Paris, France, the Centre Cardiologique de Nord in Saint-Denis, France, the Institut Cardiovasculaire Paris Sud in Massy, France, the Lenox Hill Hospital in New York, USA, the University of Colorado Hospital in Denver, USA, the Royal Hallamshire Hospital in Sheffield, UK, and the National Taiwan University Hospital in Fooyin, Taiwan. During several medical conferences live interventional treatments have been broadcast from various hospitals, using these prototypes; The coronary artery fusion software has been used during live cases at the European Congress of Percutaneous Cardiovascular Radiology (EuroPCR) of 2006 and 2007 by the Institut Cardiovasculaire Paris Sud in Massy, France, and during the Transcatheter Cardiovascular Therapeutics (TCT) conference in 2006 by the University of Colorado Hospital, Denver, USA. The needle guidance and multi-modality 3D roadmapping applications have been used during live cases at the Live Interventional Neuroradiology & Neurosurgery Course (LINNC) of 2007 and 2009 by the Fondation Rotchild Hospital in Paris, France. The described technologies have been integrated into commercially available solutions (Philips Allura 3D-RA, sold over 500 products, and Philips Allura XperGuide), which have been installed in hospitals all over the world.

12.1 Outlook and future work

The advancement of image guided interventions and therapy (IGIT) has just begun. In the future there will be a higher volume of procedures, more complex procedures, and more image integration. The ever increasing computing power will enable more complex algorithms to compute within time frames that are acceptable for interventional use. Computing devices will have more parallel capacities than today's hardware. In this sense, the trend of designing algorithmic solutions that can harness this parallel power, as has been done in this thesis by using the GPU, will continue. It is to be expected that the solutions that have been described in this thesis are just the beginning of whole families of new and advanced image processing methods that can be used in interventional treatment.

For volume visualization the primary future developments with most added value for interventional image guidance lie in the area of simultaneously interactive ray-casting of multiple volume datasets. In this thesis it has already been shown how fast volume rendering can be combined with surface rendering of other datasets, see Chapter 4. The state of the art already describes direct volume rendering of multiple datasets, but at the cost of a significant performance penalty and typically sacrificing the visualization of topological relationships as presented in this thesis. Future algorithms and hardware may address these issues. In the field of fast registration algorithms there are many directions that can be explored. Robust model and feature-based registration, registering more than two datasets simultaneously, and the exploitation of massive parallelism are just a few examples. The application of the presented methods to a large patient population in the context of a clinical trial would allow to quantify the merits in terms of efficacy, radiation dose reduction, iodine contrast medium used, and clinical outcome.

Furthermore there are still many clinical interventional treatments that can benefit from the presented techniques. Multimodal image guidance for applications such as endovascular aneurysm repair (EVAR) when treating an abdominal aortic aneurysm (AAA) or the embolization or ablation of liver tumors, elastic image registration of functional image data during neuro-surgery, and the registration of planning data for guidance during the treatment of structural heart disease are only some of the image guided therapies that could benefit from advanced peri-interventional registration and interactive fused visualization. The results from the clinical applications described in this thesis are very encouraging and the path that has been followed can be expanded to improve the clinical outcome for many other interventional treatments.

Bibliography

[1] Kurt Akeley. Reality engine graphics. In *Proceedings of SIGGRAPH'93*, volume 27, pages 109–116, 1993.

[2] Brian Cabral, Nancy Cam, and Jim Foran. Accelerated volume rendering and tomographic reconstruction using texture mapping hardware. In *Symposium on Volume visualization*, pages 91–98. ACM Press, 1994.

[3] Timothy J. Cullip and Ulrich Neumann. Accelerating volume reconstruction with 3D texture hardware. Technical Report TR93-027, 1993.

[4] Klaus Engel and Thomas Ertl. Interactive high-quality volume rendering with flexible consumer graphics hardware. In *Eurographics '02 - State of the Art Report*, 2002.

[5] Klaus Engel, Martin Kraus, and Thomas Ertl. High-quality pre-integrated volume rendering using hardware-accelerated pixel shading. In *Proceedings of the Eurographics workshop on Graphics hardware*, pages 9–16. ACM Press, 2001.

[6] Manfred Meissner, Stefan Guthe, and Wolfgang Strasser. Interactive lighting models and pre-integration for volume rendering on PC graphics accelerators. In *Proceedings Graphics Interface*, pages 209–218, 2002.

[7] Stefan Roettger, Stefan Guthe, Daniel Weiskopf, Thomas Ertl, and Wolfgang Strasser. Smart hardware-accelerated volume rendering. In *VisSym'03: Proceedings of the symposium on Data Visualisation*, pages 231–238, 2003.

[8] Jens Krueger and Rüdiger Westermann. Acceleration techniques for GPU-based volume rendering. In *Proceedings IEEE Visualization*, 2003.

[9] Marc Levoy. Efficient ray tracing of volume data. *ACM Transactions on Graphics*, 9(3):245–261, 1990.

[10] Karel Zuiderveld, A. H. J. Koning, and Max A. Viergever. Acceleration of ray-casting using 3D distance transform. In *Proceedings of Visualization in Biomedical Computing II*, pages 324–335, 1992.

[11] Roni Yagel and Zhouhong Shi. Accelerating volume animation by space-leaping. In *IEEE Visualization*, pages 62–69, Oct 1993.

[12] Rüdiger Westermann and Bernd Sevenich. Accelerated volume ray-casting using texture mapping. In *Proceedings IEEE Visualization 2001*, pages 271–278. IEEE Computer Society, 2001.

[13] Christian Sigg and Markus Hadwiger. Fast third-order texture filtering. In Matt Pharr, editor, *GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation*, pages 313–329, 2005.

[14] Alejandro F. Frangi, Wiro J. Niessen, Koen L. Vincken, and Max A. Viergever. Multiscale vessel enhancement filtering. In *Proceedings MICCAI'98*, pages 130–137, London, UK, 1998. Springer-Verlag.

[15] Paul J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Mar 1992.

[16] Wilhelm Röntgen. Über eine neue Art von Strahlen. *Sitzungsberichte der physikalisch-medizinischen Gesellschaft zu Würzburg*, (9):132–142, 1895.

[17] Eduard Haschek and Otto Lindenthal. A contribution to the practical use of photography according to röntgen. *Wien Chir Wochenschr*, (9):63, 1896.

[18] Bernhard Meier. Percutaneous coronary intervention: a review. *Medica Mundi*, 50(1):26–34, 2006.

[19] Werner Forssmann. Die sondierung des rechten Herzens. *Klinische Wochenschrift*, 8(45):2085–2087, 1929.

[20] Werner Forssmann. Über Kontrastdarstellung der Höhlen des lebenden rechten Herzens und der Lungenschlagader. *Münchener Medizinische Wochenschrift*, (78):489–492, 1931.

[21] Misty M. Payne. Charles Theodore Dotter. *Texas Heart Institute Journal*, 28(1):28–38, 2001.

[22] Andreas R. Grüntzig. Transluminal dilatation of coronary-artery stenosis [letter]. *Lancet*, 1:263, 1978.

[23] Heidar Arjomanda, Zoltan G. Turi, Daniel McCormick, and Sheldon Goldberg. Percutaneous coronary intervention: Historical perspectives, current status, and future directions. *American Heart Journal*, 146(5):787–796, Nov 2003.

[24] Ulrich Sigwart, Jacques Puel, V. Mirkovitch, F. Joffre, and L. Kappenberger. Intravascular stents to prevent occlusion and restenosis after transluminal angioplasty. *New England Journal of Medicine*, 316:701–706, 1987.

[25] René Aarnink and Volker Rasche. Mobile C-arm systems. *Medica Mundi*, 50(1):19–25, 2006.

[26] Arnold R. Cowen, Stephen M. Kengyelics, and A. Giles Davies. Solid-state, flat-panel, digital radiography detectors and their physical imaging characteristics. *Clinical Radiology*, (63):487–498, 2008.

[27] Tom J. C. Bruijns, Robert F. Bury, Falko Busse, Andrew G. Davies, Arnold R. Cowen, Walter Ruetten, and Hans Reitsma. Technical and clinical assessments of an experimental flat dynamic x-ray image detector system. In *Proceedings SPIE Medical Imaging*, pages 324–335, 1999.

[28] Robert F. Bury, Atnold R. Cowen, Andrew G. Davies, P. Hawkridge, A. J. C. Bruijns, and Eric von Reth. Initial technical and clinical evaluation of a new universal image receptor system. *European Radiology*, 10(12):1983–1987, 2000.

[29] Hækan Geijer amd Karl-Wilhelm Beckman, Torbjörn Andersson, and Jan Persliden. Image quality vs radiation dose for a flat-panel amorphous silicon detector: a phantom study. *European Radiology*, 11(9):1704–1709, 2001.

[30] Tom J. C. Bruijns, Raoul J. M. Bastiaens, Bart Hoornaert, Eric von Reth, Falko Busse, Volker K. Heer; Thierry Ducourant, Arnold R. Cowen, Andrew G. Davies, and Francois Terrier. Image quality of a large-area dynamic flat detector: comparison with a state-of-the-art II/TV system. In *Proceedings SPIE Medical Imaging*, pages 332–343, 2002.

[31] Jens Wiegert. *Scattered radiation in cone-beam computed tomography: analysis, quantification and compensation*. PhD thesis, University of Technology Aachen (RWTH), 2007.

[32] Lee A. Feldkamp, L. Craig Davis, and J. W. Kress. Practical conebeam algorithm. *Journal of the Optical Society of America A*, 1(6):612619, June 1984.

[33] Rolf Clack and Michel Defrise. Cone-beam reconstruction by the use of Radon transform intermediate functions. *Journal of the Optical Society of America A*, 11(2):580–585, 1994.

[34] Hermann Schomberg. Complete source trajectories for C-arm systems and a method for coping with truncated cone-beam projections. In *Proceedings of the 2001 International Meeting on Fully Three-Dimensional Image Reconstruction in Radiology and Nuclear Medicine*, pages 221–224, Pacific Grove, CA, USA, Oct 2001.

[35] Michael Grass, Th. Köhler, and Roland Proksa. 3D cone-beam CT reconstruction for circular trajectories. *Physics in Medicine and Biology*, (45):329–347, 2000.

[36] Michael Grass, Reiner Koppe, Erhard Klotz, Roland Proksa, Michael H. Kuhn, Hans Aerts, John op de Beek, and Richard Kemkers. Three-dimensional reconstruction of high contrast objects using C-arm image intensifier projection data. *Computerized Medical Imaging and Graphics*, (23):311–321, 1999.

[37] Rebecca Fahrig, Allan J. Fox, Stephen Lownie, and David W. Holdsworth. Use of a C-arm system to generate true three-dimensional computed rotational angiograms: preliminary in vitro and in vivo results. *American Journal of Neuroradiology*, (18):1507–1514, 1997.

[38] Nassir Navab, Ali Bani-Hashemi, Mariappan S. Nadar, Karl Wiesent, Peter Durlak, Thomas Brunner, Karl Barth, and Rainer Graumann. 3D reconstruction from projection matrices in a C-arm based 3D-angiography system. In *MICCAI ’98: Proceedings of the First International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 119–129, London, UK, 1998. Springer-Verlag.

[39] Niels J. Noordhoek, Peter G. van de Haar, and Jan Timmer. Direct comparison of commercially available C-arm CT to multislice CT image quality. In *Proceedings RSNA*, Chicago, USA, Nov 2006.

[40] Willi A. Kalender and Yiannis Kyriakou. Flat-detector computed tomography (FD-CT). *European Radiology*, (17):2767–2779, 2007.

[41] George Eckel. *OpenGL Volumizer Programmer's Guide*. Silicon Graphics, Inc, 1998.

[42] Daniel Weiskopf, Manfred Weiler, and Thomas Ertl. Maintaining constant frame rates in 3D texture-based volume rendering. In *Computer Graphics International (CGI)*, pages 604–607, 2004.

[43] X. Tong, W. Wang, W. Tsang, and Z. Tang. Efficiently rendering large volume data using texture mapping hardware. In *Joint Eurographics - IEEE TCVG Symposium on Visualization (VisSym)*, pages 121–132, 1999.

[44] Stefan Guthe, Michael Wand, Julius Gonser, and Wolfgang Strasser. Interactive rendering of large volume data sets. In *Proceedings IEEE Visualization*, pages 53–60, 2002.

[45] Eric LaMar, Bernd Hamann, and Kenneth I. Joy. Multiresolution techniques for interactive texture-based volume visualization. In *Proceedings IEEE Visualization*, pages 355–361, 1999.

[46] Manfred Weiler, Rüdiger Westermann, Chuck Hansen, Kurt Zimmerman, and Thomas Ertl. Level-of-detail volume rendering via 3D textures. In *Proceedings of Volume Visualization and Graphics Symposium 2000*, pages 7–13, 2000.

[47] Imma Boada, Isabel Navazo, and Roberto Scopigno. Multiresolution volume visualization with a texture-based octree. *The Visual Computer*, 17:185–197, 2001.

[48] Rajagopalan Srinivasan, Shiaofen Fang, and Su Huang. Rendering by template-based octree projection. In *Proceedings of the 8th Eurographics Workshop on Visualization in Scientific Computing*, pages 155–163. Eurographics, 1997.

[49] Jeff Orchard and Torsten Möller. Accelerated splatting using a 3D adjacency data structure. In *Proceedings Graphics Interface*, pages 191–200. Canadian Information Processing Society, 2001.

[50] Steven Parker, Peter Shirley, Yarden Livnat, Charles Hansen, Peter-Pike Sloan, and M. Parker. Interacting with gigabyte volume datasets on the origin 2000. In *The 41st Annual Cray User's Group Conference*, 1999.

[51] David E. DeMarle, Steven Parker, Mark Hartner, Christiaan Gribble, and Charles Hansen. Distributed interactive ray tracing for large volume visualization. In *IEEE Symposium on Parallel and Large-Data Visualization and Graphics*, pages 87–94, 2003.

[52] Sören Grimm, Stefan Bruckner, Armin Kanitsar, and Eduard Gröller. Memory efficient acceleration structures and techniques for CPU-based volume raycasting of large data. In *IEEE Symposium on Volume Visualization and Graphics*, pages 1–8, 2004.

[53] James T. Kajiya. The rendering equation. *Computer Graphics, Proceedings of SIGGRAPH'86*, 20(4):143–150, 1986.

[54] Thomas Porter and Tom Duff. Compositing digital images. *Computer Graphics*, 18(3):253–259, 1984.

[55] Daniel Ruijters and Anna Vilanova. Optimizing GPU volume rendering. *Journal of WSCG'06*, 14(1-3):9–16, 2006.

[56] Jim Blinn. Return of the jaggy. *IEEE Computer Graphics & Applications*, 9(2):82–88, 1989.

[57] Stephen R. Marschner and Richard J. Lobb. An evaluation of reconstruction filters for volume rendering. In *Proceedings IEEE Visualization*, pages 100–107, 1994.

[58] Matthias Hopf and Thomas Ertl. Accelerating 3D convolution using graphics hardware. In *Proceedings IEEE Visualization*, pages 471–474, 1999.

[59] Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Accuracy of GPU-based B-spline evaluation. In *Computer Graphics and Imaging (CGIM)*, pages 117–122, Feb 2008.

[60] Dave Shreiner, Mason Woo, Jackie Neider, and Tom Davis. *OpenGL Programming Guide: The Official Guide to Learning OpenGL (red book)*. Addison-Wesley Pub Co, 4 edition, 2003.

[61] Cyril Zeller. Balancing the graphics pipeline for optimal performance. `http://developer.nvidia.com/`, 2002.

[62] Joe Kniss, Gordon Kindlmann, and Charles Hansen. Interactive volume rendering using multi-dimensional transfer functions and direct manipulation widgets. In *Proceedings IEEE Visualization*, pages 255–262. IEEE Computer Society, 2001.

[63] Richard Kemkers, John op de Beek, and Hans Aerts. 3D-rotational angiography: first clinical applications. In *Proceedings in Computer Assisted Radiology and Surgery*, pages 182–187. Elsevier Science, 1998.

[64] Jos C. van den Berg. Three-dimensional rotational angiography. *Endovascular Today*, Mar 2003.

[65] Ziyad S. Hakura and Anoop Gupta. The design and analysis of a cache architecture for texture mapping. In *ISCA: 24th annual International Symposium on Computer architecture*, pages 108–120, 1997.

[66] Michael Cox, Narendra Bhandari, and Michael Shantz. Multi-level texture caching for 3D graphics hardware. In *Proceedings ISCA*, pages 86–97, 1998.

[67] Homan Igehy, Matthew Eldridge, and Kekoa Proudfoot. Prefetching in a texture cache architecture. In *Eurographics Workshop on Graphics Hardware*, pages 133–142, 1998.

[68] Bharath Modayur, Rex Jakobovits, K. Maravilla, George Ojemann, and James Brinkley. Evaluation of a visualization-based approach to functional brain mapping. In *Proceedings Annual Fall Symposium, American Medical Informatics Association*, pages 429–433, Oct 1997.

[69] Kevin P. Hinshaw, Andrew V. Poliakov, Eider B. Moore, Richard F. Martin, Linda G. Shapiro, and James F. Brinkley. Shape-based cortical surface segmentation for visualization brain mapping. *NeuroImage*, (16):295–316, Jun 2002.

[70] Karel J. Zuiderveld and Max A. Viergever. Multi-modal volume visualization using object-oriented methods. In *Symposium on Volume Visualization, ACM SIGGRAPH*, pages 59–66, 1994.

[71] Peter Hastreiter and Thomas Ertl. Integrated registration and visualization of medical image data. In *Proceedings Computer Graphics Interface (CGI.98)*, pages 78–85, Jun 1998.

[72] David R. Nadeau. Volume scene graphs. In *Proceedings IEEE Symposium on Volume Visualization*, pages 49–56, 2000.

[73] Isabel Harb Manssour, S. S. Furuie, Luciana Porcher Nedel, and Carla M. Dal Sasso Freitas. A framework to visualize and interact with multimodal medical images. In *Proceedings VG.01*, pages 385–398. Springer-Verlag, Jun 2001.

[74] Isabel Harb Manssour, S. S. Furuie, Silvia Delgado Olabarriaga, and Carla M. Dal Sasso Freitas. Visualizing inner structures in multimodal volume data. In *Proceedings SIBGRAPI 2002*, Oct 2002.

[75] Arie Kaufman, Roni Yagel, and Daniel Cohen. Intermixing surface and volume rendering. In *3D Imaging in Medicine: Algorithms, Systems, applications*, pages 217–227. Springer-Verlag, Jun 1990.

[76] E. Ruth Johnson and Charles E. Mosher Jr. Integration of volume rendering and geometric graphics. In *Proceedings CH Volume Visualization Workshop*, pages 1–7. ACM Press, May 1989.

[77] Ralph Brecheisen, Anna Vilanova i Bartroli, Bram Platel, and Bart ter Haar Romeny. Flexible GPU-based multi-volume ray-casting. In *Proceedings Vision, Modeling, and Visualization (VMV)*, page 10, 2008.

[78] Xiaoping Hu, Kim K. Tan, David N. Levin, Simranjit G. Galhotra, Charles A. Pelizzari, George T. Y. Chen, Robert N. Beck, Chin-Tu Chen, and Malcolm D. Cooper. Volumetric rendering of multimodality, multivariable medical data. In *Proceedings CH Volume Visualization Workshop*, pages 45–49. ACM Press, May 1989.

[79] Dirk Vandermeulen, Peter Plets, Steven Ramakers, Paul Suetens, and Guy Marchal. Integrated visualization of brain anatomy and cerebral blood vessels. In *Workshop on Volume Visualization, ACM SIGGRAPH*, pages 39–46, Oct 1992.

[80] Ramesh Raskar and Michael Cohen. Image precision silhouette edges. In *Proceedings Symposium on Interactive 3D Graphics*, pages 135–140. ACM Press, April 1999.

[81] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings SIGGRAPH 96*, pages 31–42, 1996.

[82] Aaron Isaksen, Leonard McMillan, and Steven J. Gortler. Dynamically reparameterized light fields. In *Proceedings SIGGRAPH 2000*, pages 297–306, Jul 2000.

[83] Cees van Berkel, David W. Parker, and Anthony R. Franklin. Multiview 3D LCD. In *Proceedings SPIE - Stereoscopic Displays and Virtual Reality Systems III*, volume 2653, pages 32–39, April 1996.

[84] Wojciech Matusik and Hanspeter Pfister. 3D TV: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes. *ACM Transactions on Graphics*, 23(3):814–824, 2004.

[85] Michael Halle. Autostereoscopic displays and computer graphics. *ACM Computer Graphics*, 31(2):58–62, May 1997.

[86] Neil A. Dodgson. Autostereoscopic 3D displays. *Computer*, 38(8):31–36, 2005.

[87] Levent Onural, Thomas Sikora, Jörn Ostermann, Aljoscha Smolic, M. Reha Civanlar, and John Watson. An assessment of 3DTV technologies. In *Proceedings NAB Broadcast Engineering*, pages 456–467, 2006.

[88] Anthony Vetro, Wojciech Matusik, Hanspeter Pfister, and Jun Xin. Coding approaches for end-to-end 3D TV systems. In *Picture Coding Symposium (PCS)*, December 2004.

[89] Pieter J. H. Seuntiëns, Ingrid E. J. Heynderickx, Wijnand A. IJsselsteijn, Paul M. J. van den Avoort, Jelle Berentsen, Iwan J. Dalm, Marc T. M. Lambooij, and Willem Oosting. Viewing experience and naturalness of 3D images. In *Proceedings SPIE Optics East, IT 107 Three dimensional TV, Video, and Display IV, Vol. 6016*, pages 43–49, 2005.

[90] Janusz Konrad and Philippe Agniel. Subsampling models and anti-alias filters for 3-D automultiscopic displays. *IEEE Transactions on Image Processing*, 15(1):128–140, 2006.

[91] Ralph Braspenning, Eric Brouwer, and Gerard de Haan. Visual quality assessment of lenticular based 3D displays. In *Proceedings 13th European Signal Processing Conference (EUSIPCO)*, September 2005.

[92] Atanas Boev, Atanas Gotchev, and Karen Egiazarian. Crosstalk measurement methodology for auto-stereoscopic screens. In *Proceedings 3DTV Conference*, pages 1–4, May 2007.

[93] Robert L. Kooima, Tom Peterka, Javier I. Girado, Jinghua Ge, Daniel J. Sandin, and Thomas A. DeFanti. A GPU sub-pixel algorithm for autostereoscopic virtual reality. In *Proceedings IEEE Virtual Reality Conference*, pages 131–137, Mar 2007.

[94] Balázs Domonkos, Attila Egri, Tibor Fóris, Tamás Juhász, and László Szirmay-Kalos. Isosurface ray-casting for autostereoscopic displays. In *Proceedings of the WSCG*, pages 31–38, January 2007.

[95] Thomas Hübner and Renato Pajarola. Single-pass multi-view volume rendering. In *Proceedings IADIS International Computer Graphics and Visualization*, July 2007.

[96] Cees van Berkel. Image preparation for 3D-LCD. In *Proceedings SPIE - Stereoscopic Displays and Virtual Reality Systems VI*, volume 3639, pages 84–91, May 1999.

[97] Neil A. Dodgson. Autostereo displays: 3D without glasses. In *EID: Electronic Information Displays*, 1997.

[98] Damien Maupu, Mark H. Van Horn, Susan Weeks, and Elizabeth Bullit. 3D stereo interactive medical visualization. *IEEE Computer Graphics and Applications*, 25(5):67–71, Sept.-Oct. 2005.

[99] Eric Dubois. The sampling and reconstruction of time-varying imagery with application in video systems. *Proceedings of the IEEE*, 73(4):502–523, April 1985.

[100] Jean-Philippe Thirion. Image matching as a diffusion process: an analogy with Maxwells demons. *Medical Image Analysis*, 2(3):243–260, 1998.

[101] Steven Haker, Lei Zhua, Allen Tannenbaum, and Sigurd Angenent. Optimal mass transport for registration and warping. *International Journal of Computer Vision*, 60(3):225–240, 2004.

[102] Ingwer C. Carlsen, Astrid Franz, Sven Kabus, T. Netsch, and Vladimir Pekar. Elastic medical image registration - EMIR literature survey and basic application requirements. Technical Report PFLH-Report 1677/2003, Philips Research, Nov 2003.

[103] Yali Amit. A nonlinear variational problem for image matching. *SIAM Journal on Scientific Computing*, 15(1):207–224, 1994.

[104] Mike Fornefett, Karl Rohr, and H. Siegfried Stiehl. Radial basis functions with compact support for elastic registration of medical images. *Image and Vision Computing*, 19(1-2):87–96, 2001.

[105] Jan Kohlrausch, Karl Rohr, and H. Siegfried Stiehl. A new class of elastic body splines for nonrigid registration of medical images. In *Proceedings Workshop Bildverarbeitung für die Medizin*, pages 164–168, 2001.

[106] Philippe Thévenaz, Urs E. Ruttimann, and Michael Unser. Iterative multi-scale registration without landmarks. In *Proceedings International Conference Image Processing*, volume 3, pages 228–231, Oct 1995.

[107] Jan Kybic and Michael Unser. Fast parametric elastic image registration. *IEEE Transactions on Image Processing*, 12(11):1427–1442, Nov 2003.

[108] Julia A. Schnabel, Daniel Rueckert, Marcel Quist, Jane M. Blackall, Andy D. Castellano-Smith, Thomas Hartkens, Graeme P. Penney, Walter A. Hall, Haiying Liu, Charles L. Truwit, Frans A. Gerritsen, Derek L. G. Hill, and David J. Hawkes. A generic framework for non-rigid registration based on non-uniform multi-level free-form deformations. In *MICCAI'01*, pages 573–581, Oct 2001.

[109] Frederik Maes, A. Collignon, Dirk Vandermeulen, Gug Marchal, and Paul Suetens. Multimodality image registration by maximization of mutual information. *IEEE Transactions on Medical Imaging*, 16(2):187–198, 1997.

[110] Michael Unser. Splines: A perfect fit for signal and image processing. *IEEE Signal Processing Magazine*, 16(6):22–38, Nov 1999.

[111] Philippe Thévenaz and Michael Unser. Optimization of mutual information for multiresolution image registration. *IEEE Transactions on Signal Processing*, 9(12):2083–2099, Dec 2000.

[112] Hua-Mei Chen and Pramod K Varshney. Mutual information-based CT-MR brain image registration using generalized partial volume joint histogram estimation. *IEEE Transactions on Medical Imaging*, 22(9):1111–1119, Sep 2003.

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

[114] Andrew W. Fitzgibbon. Robust registration of 2D and 3D point sets. In *Proceedings British Machine Vision Conference BMVC*, volume II, pages 411–420, Sep 2001.

[115] Philip E. Gill, Walter Murray, and Margaret H. Wright. *Practical Optimization*. Academic Press, London, 1981.

[116] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 1992.

[117] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[118] W. L. Price. Global optimization by controlled random search. *Journal of Optimization Theory and Applications*, 40(3):333–348, 1983.

[119] Michael Unser, Akram Aldroubi, and Murray Eden. The L_2-polynomial spline pyramid. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):364–379, April 1993.

[120] Pierre Jannin, Christophe Grova, and Calvin R. Maurer, Jr. Model for defining and reporting reference-based validation protocols in medical image processing. *International Journal of Computer Assisted Radiology and Surgery*, 1(2):1001–1015, 2006.

[121] Dimitri P. Solomatine. Two strategies of adaptive cluster covering with descent and their comparison to other algorithms. *Journal of Global Optimization*, 14(1):55–78, 1999.

[122] J. Micheal Fitzpatrick and Jay B. West. The distribution of target registration error in rigid-body point-based registration. *IEEE Transactions on Medical Imaging*, 20(9):917–927, Sep 2001.

[123] Jürgen Weese, Graeme P. Penney, Paul Desmedt, T. M. Buzug, Derek L. G. Hill, and David J. Hawkes. Voxel-based 2-D/3-D registration of fluoroscopy images and CT scans for image-guided surgery. *IEEE Transactions on Information Technology in Biomedicine*, 1(4):284–293, 1997.

[124] Alan Liu, Elizabeth Bullit, and Stephen M. Pizer. 3D/2D registration via skeletal near projective invariance in tubular objects. In *MICCAI '98: Proceedings of the First International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 952–963, London, UK, 1998. Springer-Verlag.

[125] Graeme P. Penney, Philipp G. Batchelor, Derek L. G. Hill, David J. Hawkes, and Jürgen Weese. Validation of a two- to three-dimensional registration algorithm for aligning preoperative ct images and intraoperative fluoroscopy images. *Medical Physics*, 28(6):1024–1031, 2001.

[126] Everine B. van de Kraats, Graeme P. Penney, Dejan Tomaževič, Theo van Walsum, and Wiro J. Niessen. Standardized evaluation methodology for 2D-3D registration. *IEEE Transactions on Medical Imaging*, 24:1177–1190, 2005.

[127] Guy-Anna Turgeon, Glen Lehmann, Maria Drangova, David Holdsworth, and Terry Peters. 2D-3D registration of coronary angiograms for cardiac procedure planning. *Medical Physics*, 32(12):3737–3749, 2005.

[128] Dejan Tomaževič, Boštjan Likar, and Franjo Pernuš. 3-D/2-D Registration by Integrating 2-D Information in 3-D. *IEEE Transactions on Medical Imaging*, 25(1):17–27, 2006.

[129] Guoyan Zheng, Xuan Zhang, and Lutz-Peter Nolte. A class of novel point similarity measures based on MAP-MRF framework for 2D-3D registration of X-Ray Fluoroscopy to CT images. In *Proceedings IEEE Int. Symp. Biomed. Imaging (ISBI): From Nano to Macro*, pages 438–441, 2006.

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

[131] Michael Söderman, Drazenko Babic, Robert Homan, and Tommy Andersson. 3D roadmap in neuroangiography: technique and clinical interest. *Neuroradiology*, 47:735–740, 2005.

[132] Sébastien Gorges, Erwan Kerrien, Marie-Odile Berger, Jérémie Pescatore, René Anxionnat, and Luc Picard. Model of a vascular C-arm for 3D augmented fluoroscopy in interventional radiology. In *Proceedings MICCAI'05*, pages 214–222, 2005.

[133] Shirley A. M. Baert, Graeme P. Penney, Theo van Walsum, and Wiro J. Niessen. Precalibration versus 2D-3D registration for 3D guide wire display in endovascular interventions. In *Proceedings MICCAI'04*, pages 577–584, 2004.

[134] Ameet Kumar Jain, Tabish Mustafa, Yu Zhou, Gregory S. Chirikjian, and Gabor Fichtinger. FTRAC - a robust fluoroscope tracking fiducial. *Medical Physics*, 32(10):3185–3198, 2005.

[135] Anne Rougée, Catherine L. Picard, Yves L. Trousset, and Cyril Ponchut. Geometrical calibration for 3D X-ray imaging. In *Proceedings SPIE, Image Capture, Formatting, and Display*, volume 1897, pages 161–169, San Diego, USA, 1993.

[136] Reiner Koppe, Erhard Klotz, John Op de Beek, and Hans Aerts. 3D vessel reconstruction based on rotational angiography. In *Proceedings CAR*, pages 101–107, Berlin, Germany, 1995.

[137] Daniel 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):712–721, Aug 1999.

[138] Grzegorz Soza, Michael Bauer, Peter Hastreiter, Christopher Nimsky, and Günter Greiner. Non-rigid registration with use of hardware-based 3D Bézier functions. In *Proceedings MICCAI'02*, pages 549–556, 2002.

[139] Robert Strzodka, Marc Droske, and Martin Rumpf. Image registration by a regularized gradient flow. a streaming implementation in DX9 graphics hardware. *Journal on Computing*, 73(4):373–389, Nov 2004.

[140] Alexander Köhn, Johann Drexl, Felix Ritter, Matthias König, and Heinz-Otto Peitgen. GPU accelerated registration in two and three dimensions. In *Proceedings Bildverarbeitung für die Medizin*, pages 261–265, Mar 2006.

[141] Derek L. G. Hill, Philipp G. Batchelor, Mark Holden, and David J Hawkes. Medical image registration. *Physics in Medicine and Biology*, 46:R1–R45, 2001.

[142] Ulrich Clarenz, Marc Droske, and Martin Rumpf. Towards fast non–rigid registration. In *Inverse Problems, Image Analysis and Medical Imaging, AMS Special Session Interaction of Inverse Problems and Image Analysis*, volume 313, pages 67–84, 2002.

[143] Isaac J. Schoenberg. Contributions to the problem of approximation of equidistant data by analytic functions. *Quarterly of Applied Mathematics*, 4(1):pp. 45–99 and 112–141, 1946.

[144] Ian Buck. GPU computing: Programming a massively parallel processor. In *Code Generation and Optimization, CGO'07*, page 17, Mar 2007.

[145] William R. Mark, R. Steven Glanville, Kurt Akeley, and Mark J. Kilgard. Cg: A system for programming graphics hardware in a C-like language. *ACM Transactions on Graphics*, 22(3):896–907, 2003.

[146] *NVIDIA CUDA Compute Unified Device Architecture: Programming Guide, Appendix D.2 Linear Filtering.* 2008.

[147] Sven Kabus, Thomas Netsch, Bernd Fischer, and Jan Modersitzki. B-spline registration of 3D images with Levenberg-Marquardt optimization. In *Proceedings SPIE Medical Imaging*, pages 304–313, 2004.

[148] Tauseef Rehman, Gallagher Pryor, John Melonakos, and Allen Tannenbaum. Multi-resolution 3D nonrigid registration via optimal mass transport on the GPU. In *Proceedings Computational Biomechanics for Medicine-II, MICCAI*, pages 122–132, 2007.

[149] J. Bijhold. Three-dimensional verification of patient placement during radiotherapy using portal images. *Medical Physics*, 20(2):347–356, 1993.

[150] Kenneth G. A. Gilhuijs, Peter J. H. van de Ven, and Marcel van Herk. Automatic three-dimensional inspection of patient setup in radiation therapy using portal images, simulator images, and computed tomography data. *Medical Physics*, 23:389–399, 1996.

[151] Martin J. Murphy. An automatic six-degree-of-freedom image registration algorithm for image-guided frameless stereotaxic radiosurgery. *Medical Physics*, 24:857–866, 1997.

[152] Sébastien Clippe, David Sarrut, Claude Malet, Serge Miguet, Chantal Ginestet, and Christian Carrie. Patient setup error measurement using 3-D intensity-based image registration techniques. *International Journal of Radiation Oncology Biology Physics*, 56(1):259–265, 2003.

[153] Louis Lemieux, R. Jagoe, David R. Fish, Neil D. Kitchen, and David G. T. Thomas. A patient-to-computed-tomography image registration based on digitally reconstructed radiographs. *Medical Physics*, 21:1749–1760, 1994.

[154] Stéphane Lavallée, Jocelyne Troccaz, Pascal Sautot, Bruno Mazier, Philippe Cinquin, Philippe Merloz, and Jean-Paul Chirossel. Computer-assisted spinal surgery using anatomy-based registration. In *Computer-Integrated Surgery: Technology and Clinical Applications*, pages 425–449, Cambridge, MA, 1996. MIT Press.

[155] André Guéziec, Peter Kazanzides, Bill Williamson, and Russell H. Taylor. Anatomy-based registration of CT-scan and intraoperative X-ray images for guidance of a surgical robot. *IEEE Transactions on Medical Imaging*, 17(5):715–728, 1998.

[156] Marcel Breeuwer, John P. Wadley, Hubrecht L. T. de Bliek, Johannes Buurman, Paul Desmedt, Paul M. C. Gieles, Frans A. Gerritsen, N. L. Dorward, N. D. Kitchen, B. Velani, D. G. T. Thomas, Onno Wink, Jan D. Blankensteijn, Bert C. Eikelboom, W. P. Th. M. Mali, Max A. Viergever, Graeme P. Penney, Ronald P. Gaston, Derek L. G. Hill, Calvin R. Maurer Jr., David J. Hawkes, Frederik Maes, Dirk Vandermeulen, Rudi Verbeeck, Paul Suetens, Georg Schmitz, Thorsten M. Buzug, Cristian Lorenz, Jürgen Sabczynski, Jürgen Weese, W. Zylka, and M. H. Kuhn. The easi project-improving the effectiveness and quality of image-guided surgery. *IEEE Transactions on Information Technology in Biomedicine*, 2(3):156–168, 1998.

[157] Erwan Kerrien, Marie-Odile Berger, Eric Maurincomme, Laurent Launay, Régis Vaillant, and Luc Picard. Fully automatic 3D/2D subtracted angiography registration. In *MICCAI '99: Proceedings of the Second International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 664–671, London, UK, 1999. Springer-Verlag.

[158] Yasuyo Kita, Dale L. Wilson, and J. Alison Noble. Real-time registration of 3D cerebral vessels to X-ray angiograms. In *MICCAI '98: Proceedings of the First International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 1125–1133, London, UK, 1998. Springer-Verlag.

[159] John H. Hipwell, Greame P. Penney, Robert A. McLaughlin, Kawal Rhode, Paul Summers, Tim C. Cox, James V. Byrne, J. Alison Noble, and David J. Hawkes. Intensity-based 2-D-3-D registration of cerebral angiograms. *IEEE Transactions on Medical Imaging*, 22(11):1417–1426, 2003.

[160] Jacques Feldmar, Grégoire Malandain, Nicholas Ayache, Sara Fernández-Vidal, Eric Maurincomme, and Yves Trousset. Matching 3D MR angiography data and 2D X-ray angiograms. In *CVRMed-MRCAS '97: Proceedings of the First Joint Conference on Computer Vision, Virtual Reality and Robotics in Medicine and Medial Robotics and Computer-Assisted Surgery*, pages 129–138, London, UK, 1997. Springer-Verlag.

[161] Charles Florin, James Williams, Ali Khamene, and Nikos Paragios. Registration of 3D angiographic and X-ray images using sequential monte carlo sampling. *Computer Vision for Biomedical Image Applications*, 3765:427–436, 2005.

[162] Hari Sundar, Ali Khamene, Chenyang Xu, Frank Sauer, and Christos Davatzikos. A novel 2D-3D registration algorithm for aligning fluoroscopic images with pre-operative 3D images. In *SPIE Medical Imaging*, Feb 2006.

[163] Ravi Bansal, Lawrence H. Staib, Zhe Chen, Anand Rangarajan, Jonathan Knisely, Ravinder Nath, and James S. Duncan. Entropy-based dual-portal-to-3D-CT registration incorporating pixel correlation. *IEEE Transactions on Medical Imaging*, 22(1):29–49, 2003.

[164] Wolfgang Birkfellner, Joachim Wirth, Wolfgang Burgstaller, Bernard Baumann, Harald Staedele, Beat Hammer, Niels Claudius Gellrich, Augustinus Ludwig Jacob, Pietro Regazzoni, and Peter Messmer. A faster method for 3D/2D medical image registration - a simulation study. *Physics in Medicine and Biology*, 48(16):2665–2679, 2003.

[165] Lisa M. Gottesfeld Brown and Terrance E. Boult. Registration of planar film radiographs with computed tomography. In *MMBIA '96: Proceedings of the 1996 Workshop on Mathematical Methods in Biomedical Image Analysis (MMBIA '96)*, pages 42–51, Washington DC, USA, 1996. IEEE Computer Society.

[166] Dotan Knaan and Leo Joskowicz. Effective intensity-based 2D/3D rigid registration between fluoroscopic X-ray and CT. In *Proceedings MICCAI'03*, volume 2878 of *Lecture Notes in Computer Science*, pages 351–358. Springer, 2003.

[167] Harel Livyatan, Ziv Yaniv, and Leo Joskowicz. Gradient-based 2-D/3-D rigid registration of fluoroscopic X-ray to CT. *IEEE Transactions on Medical Imaging*, 22(11):1395–1406, 2003.

[168] Graeme P. Penney, Jürgen Weese, J. A. Little, Paul Desmedt, Derek L. G. Hill, and David J. Hawkes. A comparison of similarity measures for use in 2D-3D medical image registration. *IEEE Transactions on Medical Imaging*, 17(4):586–595, 1998.

[169] Dejan Tomaževič, Boštjan Likar, Tomaž Slivnik, and Franjo Pernuš. 3-D/2-D registration of CT and MR to X-ray images. *IEEE Transactions on Medical Imaging*, 22(11):1407–1416, 2003.

[170] Lilla Zöllei, W. Eric L. Grimson, Alexander Norbash, and William M. Wells III. 2D-3D rigid registration of X-ray fluoroscopy and CT images using mutual information and sparsely sampled histogram estimators. In *Proceedings IEEE Conference on Computer Vision Pattern Recognition*, pages 696–703, 2001.

[171] Jacques Feldmar, Nicholas Ayache, and Fabienne Betting. 3D-2D projective registration of free-form curves and surfaces. *Computer Vision and Image Understanding*, 65:403–424, 1997.

[172] Stéphane Lavallée and Richard Szeliski. Recovering the position and orientation of free-form objects from image contours using 3-D distance maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:378–390, 1995.

[173] Ali Hamadeh, Stéphane Lavallée, and Philippe Cinquin. Automated 3-dimensional computed tomography and fluoroscopic image registration. *Computer Aided Surgery*, 3:11–19, 1998.

[174] K. K. Lau and Albert C. S. Chung. A global optimization strategy for 3D-2D registration of vascular images. In *Proceedings 17th British Machine Vision Conference, BMVC*, volume 2, pages 489–498, Edinburgh, UK, Sep 2006.

[175] Sabine Mollus, Jördis Lübke, Andreas J. Walczuch, Heidrun Schumann, and Jürgen Weese. Model-to-image based 2D-3D-registration of angiographic data. In *Proceedings SPIE Medical Imaging: Image Processing*, volume 6914, pages 69142S–1–69142S–12, San Diego, USA, 2008.

[176] Silvia D. Olabarriaga, Marcel Breeuwer, and Wiro J. Niessen. Minimum cost path algorithm for coronary artery central axis tracking in CT images. In *Proceedings MICCAI'03*, pages 687–694. Springer-Verlag, 2003.

[177] Alexandre Bousse, C. Boldak, Christine Toumoulin, Guanyu Yanga, Soizic Laguitton, and Dominique Boulmier. Coronary extraction and characterization in multi-detector computed tomography. *ITBM-RBM*, 27(5-6):217–226, 2006.

[178] Pascal Fallavollita and Farida Cheriet. Optimal 3D reconstruction of coronary arteries for 3D clinical assessment. *Comput Med Imaging Graph*, 32(6):476–487, 2008.

[179] Daniel Mueller and Anthony Maeder. Robust semi-automated path extraction for visualising stenosis of the coronary arteries. *Computerized Medical Imaging and Graphics*, 32(6):463–475, 2008.

[180] James Kennedy and Russell C. Eberhart. Particle swarm optimization. In *Proceedings IEEE International Conference on Neural Networks*, volume 4, pages 1942–1948, 1995.

[181] Matthias Tessmann, Christian Eisenacher, Frank Enders, Marc Stamminger, and Peter Hastreiter. GPU accelerated normalized mutual information and B-spline transformation. In *Proceedings Eurographics Workshop on Visual Computing for Biomedicine (EG VCBM)*, pages 117–124, 2008.

[182] Ramtin Shams and Nick Barnes. Speeding up mutual information computation using NVIDIA CUDA hardware. In *Proceedings Digital Image Computing: Techniques and Applications (DICTA)*, pages 555–560, Washington, DC, USA, 2007. IEEE Computer Society.

[183] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3-D surface construction algorithm. *ACM Computer Graphics*, 21(4):163–169, 1987.

[184] Donald E. Knuth. *The Art of Computer Programming, Volume 3: Sorting and Searching, Section 6.2.1: Searching an Ordered Table*. Addison-Wesley, 1997.

[185] Joseph Stancanello, Carlo Cavedon, Paolo Francescon, Pietro Cerveri, Giancarlo Ferrigno, Federico Colombo, and Stefano Perini. Development and validation of a CT-3D rotational angiography registration method for AVM radiosurgery. *Medical Physics*, 31(6):1363–1371, Jun 2004.

[186] Chung-Jung Lin, Raphael Blanc, Frédéric Clarençon, Michel Piotin, Laurent Spelle, Jérémy Guillermic, and Jacques Moret. Overlying fluoroscopy and preacquired CT angiography for road-mapping in cerebral angiography. *AJNR American Journal of Neuroradiology*, October 2009.

[187] Dana Erickson, Yogish C. Kudva, Michael J. Ebersold, Geoffrey B. Thompson, Clive S. Grant, Jon A. van Heerden, and William F. Young Jr. Benign paragangliomas: clinical presentation and treatment outcomes in 236 patients. *Journal of Clinical Endocrinology & Metabolism*, 86:5210–5216, 2001.

[188] René van den Berg, G. Rodesch, and P. Lasjaunias. Management of paragangliomas: clinical and angiographic aspects. *Intervental Neuroradiology*, 8:127–134, 2002.

[189] Ponnandai Somasundar, R. Krouse, R. Hostetter, R. Vaughan, and T. Covey. Paragangliomas: a decade of clinical experience. *Journal of Surgical Oncology*, 74(4):286–290, 2000.

[190] Peter Patetsios, Dennis R. Gable, Wilson V. Garrett, Jeffrey P. Lamont, Joseph A. Kuhn, William P. Shutze, Harry Kourlis, Bradley Grimsley, Gregory J. Pearl, Bertram L. Smith, C. M. Talkington, and Jesse E. Thompson. Management of carotid body paragangliomas and review of a 30-year experience. *Annals of Vascular Surgery*, 16(3):331–338, 2002.

[191] Manfred Muhm, Peter Polterauer, Wolfgang Gstöttner, Andreas Temmel, Bernd Richling, Gerhard Undt, Bruno Niederle, Michael Staudacher, and Herbert Ehringer. Diagnostic and therapeutic approaches to carotid body tumors. Review of 24 patients. *Archives of Surgery*, 132(3):279–284, 1997.

[192] John P. Leonetti, Joseph J. Donzelli, Fred N. Littooy, and Brian P. Farrell. Perioperative strategies in the management of carotid body tumors. *Otolaryngology Head and Neck Surgery*, 117(1):111–115, 1997.

[193] James S. Brown. Glomus jugulare tumors revisited: a ten-year statistical follow-up of 231 cases. *The Laryngoscope*, 95(3):284–288, 1985.

[194] Joseph C. Sniezek, Alain N. Sabri, and James L. Netterville. Paraganglioma surgery: complications and treatment. *Otolaryngologic Clinics of North America*, 34(5):993–1006, 2001.

[195] Mark S. Persky, Avi Setton, Yasunari Niimi, Jonathan Hartman, Douglas Frank, and Alex Berenstein. Combined endovascular and surgical treatment of head and neck paragangliomasa team approach. *Head & Neck*, 24(5):423–431, 2002.

[196] Andreas Gruber, Gerhard Bavinzski, Monika Killer, and Bernd Richling. Preoperative embolization of hypervascular skull base tumors. *Minimally Invasive Neurosurgery*, 43(4):62–71, 2000.

[197] Tapani Tikkakoski, Jukka Luotonen, Sami Leinonen, Topi Siniluoto, Outi Heikkilä, Markku Päivänsalo, and Kalevi Hyrynkangas. Preoperative embolization in the management of neck paragangliomas. *The Laryngoscope*, 107(6):821–826, 1997.

[198] Terrence P. Murphy and Derald E. Brackmann. Effects of preoperative embolization on glomus jugulare tumors. *The Laryngoscope*, 99(12):1244–1247, 1989.

[199] Anton Valavanis. Preoperative embolization of the head and neck: indications, patient selection, goals, and precautions. *AJNR American Journal of Neuroradiology*, 7(5):943–952, 1986.

[200] Laurent Pierot, Anne Boulin, Lina Castaings, Frédéric Chabolle, and Jacques Moret. Embolization by direct puncture of hypervascularized ORL tumors. *Annales d'otolaryngologie et de chirurgie cervico faciale*, 111(7):403–409, 1994.

[201] Alfredo Casasco, Denis Herbreteau, Emmanuel Houdart, Bernard George, P. Tran Ba Huy, D. Deffresne, and Jean-Jacques Merland. Devascularization of craniofacial tumors by percutaneous tumor puncture. *AJNR American Journal of Neuroradiology*, 15(7):1233–1239, 1994.

[202] John C. Chaloupka, Sundeep Mangla, Daniel C. Huddle, Toni C. Roth, Sanchayeeta Mitra, Douglas A. Ross, and Clarence T. Sasaki. Evolving experience with direct puncture therapeutic embolization for adjunctive and palliative management of head and neck hypervascular neoplasms. *The Laryngoscope*, 109(11):1864–1872, 1999.

[203] Alfredo Casasco, Emmanuel Houdart, Alessandra Biondi, Harish S. Jhaveri, Denis Herbreteau, Armand Aymard, and Jean-Jacques Merland. Major complications of percutaneous embolization of skull-base tumors. *AJNR American Journal of Neuroradiology*, 20:179–181, 1999.

[204] Marco Leonardi, C. Barbara, L. Simmonetti, R. Giardino, N. Nicoli Aldini, M. Fini, L. Martini, L. Masetti, M. Joechler, and F. Roncaroli. Glubran 2: a new acrylic glue for neuroradiological endovascular use. *Interventional Neuroradiology*, 8:245–250, 2002.

[205] Daniel Giansante Abud, Charbel Mounayer, Goetz Benndorf, Michel Piotin, Laurent Spelle, and Jacques Moret. Intratumoral injection of cyanoacrylate glue in head and neck paragangliomas. *AJNR American Journal of Neuroradiology*, 25:1457–1462, 2004.

[206] John M. Racadio, Drazenko Babic, Robert Homan, John W. Rampton, Manish N. Patel, Judy M. Racadio, and Neil D. Johnson. Live 3D guidance in the interventional radiology suite. *American Journal of Roentgenology*, 189:W357–W364, 2007.

[207] Noboru Maeda, Keigo Osuga, Hiroki Higashihara, Kaname Tomoda, and Hironobu Nakamura. A novel conebeam CT guided interventions by XperGuide: Accuracy and feasibility in a phantom model. *Journal of Vascular and Interventional Radiology*, 19(2, Supplement):S90, Feb 2008.

[208] U.S. department of health & human services, national heart lung and blood institute (NHLBI). `http://www.nhlbi.nih.gov/health/`.

[209] Angela Hoye, Kengo Tanabe, Pedro A. Lemos, Jiro Aoki, Francesco Saia, Chourmouzios Arampatzis, Muzaffer Degertekin, Sjoerd H. Hofma, Georgios Sianos, Eugene McFadden, Willem J. van der Giessen, Pieter C. Smits, Pim J. de Feyter, Ron T. van Domburg, and Patrick W. Serruys. Significant reduction in restenosis after the use of sirolimus-eluting stents in the treatment of chronic total occlusions. *Journal of the American College of Cardiology*, 43(11):1954–1958, Jun 2004.

[210] George D. Dangas, Roxana Mehran, Martin B. Leon, and Jeffrey W. Moses. *Handbook of Chronic Total Occlusions*. Informa, London, UK, 2007.

[211] Kean H. Soon, Joseph B. Selvanayagam, Nicholas Cox, Anne-Maree Kelly, Kevin W. Bell, and Yean L. Lim. Percutaneous revascularization of chronic total occlusions: Review of the role of invasive and non-invasive imaging modalities. *International Journal of Cardiology*, 116:1–6, 2007.

[212] David M. Safley, John A. House, Steven P. Marso, J. Aaron Grantham, and Barry D. Rutherford. Improvement in survivial following successful percutaneous coronary intervenion of corornary chronic total occlusions: Variability by target vessel. *JACC: Cardiovascular Interventions*, 1(3):295–302, 2008.

[213] Nico R. Mollet, Angela Hoye, Pedro A. Lemos, Filippo Cademartiri, Georgios Sianos, Eugene P. McFadden, Gabriel P. Krestin, Patrick W. Serruys, and Pim J. de Feyter. Value of preprocedure multislice computed tomographic coronary angiography to predict the outcome of percutaneous recanalization of chronic total occlusions. *American Journal of Cardiology*, 95:240–243, 2005.

[214] Onno Wink, Harvey S. Hecht, and Daniel Ruijters. Coronary computed tomographic angiography in the cardiac catheterization laboratory: Current applications and future developments. *Cardiology Clinics, Advances in Coronary Angiography*, 27(3):513–529, Aug 2009.

[215] James P. Earls, Elise L. Berman, Bruce A. Urban, Charlene A. Curry, Judith L. Lane, Robert S. Jennings, Colin C. McCulloch, Jiang Hsieh, and John H. Londt. Prospectively gated transverse coronary CT angiography versus retrospectively gated helical technique: improved image quality and reduced radiation dose. *Radiology*, 246(3):742–753, 2008.

[216] Nobuhiko Hirai, Jun Horiguchi, Chikako Fujioka, Masao Kiguchi, Hideya Yamamoto, Noriaki Matsuura, Toshiro Kitagawa, Hiroki Teragawa, Nobuoki Kohno, and Katsuhide Ito. Prospective versus retrospective ECG-gated 64-detector coronary CT angiography: assessment of image quality, stenosis, and radiation dose. *Radiology*, 248(2):424–430, Jun 2008.

[217] Oliver Klass, Martin Jeltsch, Sebastian Feuerlein, Horst Brunner, Hans-Dieter Nagel, Matthew J. Walker, Hans-Juergen Brambs, and Martin H. K. Hoffmann. Prospectively

gated axial CT coronary angiography: preliminary experiences with a novel low-dose technique. *European Radiology*, 19(4):829–836, Apr 2009.

[218] Hans Scheffel, Hatem Alkadhi, Sebastian Leschka, André Plass, Lotus Desbiolles, Ivo Guber, Tobias Krauss, Jürg Grünenfelder, Michele Genoni, Thomas F. Luescher, Borut Marincek, and Paul Stolzmann. Low-dose CT coronary angiography in the step-and-shoot mode: diagnostic performance. *Heart*, 94:1132–1137, 2008.

[219] Takao Maruyama, Masanori Takada, Toshiaki Hasuike, Atsushi Yoshikawa, Eiji Namimatsu, and Tohru Yoshizumi. Radiation dose reduction and coronary assessability of prospective electrocardiogram-gated computed tomography coronary angiography. comparison with retrospective electrocardiogram-gated helical scan. *Journal of the American College of Cardiology*, 52(18):1450–1455, 2008.

[220] Harvey S. Hecht and Gary Roubin. Usefulness of computed tomography guided percutaneous coronary intervention. *American Journal of Cardiology*, 99:871–875, 2007.

[221] Sebastian Leschka, Hatem Alkadhi, André Plass, Lotus Desbiolles, Jürg Grünenfelder, Borut Marincek, and Simon Wildermuth. Accuracy of MSCT coronary angiography with 64-slice technology: first experience. *European Heart Journal*, 26(15):1482–1487, 2005.

[222] Alexander W. Leber, Andreas Knez, Franz von Ziegler, Alexander Becker, Konstantin Nikolaou, Stephan Paul, Bernd Wintersperger, Maximilian Reiser, Christoph R. Becker, Gerhard Steinbeck, and Peter Boekstegers. Quantification of obstructive and nonobstructive coronary lesions by 64-slice computed tomography: a comparative study with quantitative coronary angiography and intravascular ultrasound. *Journal of the American College of Cardiology*, 46:147–154, 2005.

[223] Nico R. Mollet, Filippo Cademartiri, C. A. G. van Meigham, G. Runza, Eugene P. McFadden, T. Baks, Patrick W. Serruys, G. P. Krestin, and Pim J. de Feyter. High-resolution spiral computed tomography coronary angiography in patients referred for diagnostic conventional coronary angiography. *Circulation*, 112:2318–2323, 2005.

[224] Gilbert L. Raff, Michael J. Gallagher, William W. ONeill, and James A. Goldstein. Diagnostic accuracy of noninvasive coronary angiography using 64-slice spiral computed tomography. *JACC Journal of the American College of Cardiology*, 46:552–557, 2005.

[225] Dieter Ropers, Johannes Rixe, Katharina Anders, Axel Küttner, Ulrich Baum, Werner Bautz, Werner G. Daniel, and Stephan Achenbach. Usefulness of multidetector row spiral computed tomography with 64- x 0.6-mm collimation and 330-ms rotation for the noninvasive detection of significant coronary artery stenoses. *American Journal of Cardiology*, 97(3):343–348, 2006.

[226] Jeffrey J. Fine, Christie B. Hopkins, Nicol Ruff, and F. Carter Newton. Comparison of accuracy of 64-slice cardiovascular computed tomography with coronary angiography in patients with suspected coronary artery disease. *American Journal of Cardiology*, 97(2):173–174, 2006.

[227] Onno Wink, Richard Kemkers, S. James Chen, and John D. Carroll. Intra-procedural coronary intervention planning using hybrid 3-dimensional reconstruction techniques. *Academic Radiology*, 10(12):1433–1441, dec 2003.

[228] Joel A. Garcia, Shyam Bhakta, Joseph Kay, Kak-Chen Chan, Onno Wink, Danny Ruijters, and John D. Carroll. On-line multi-slice computed tomography interactive overlay with conventional X-ray: A new and advanced imaging fusion concept. *International Journal of Cardiology*, 133(3):e101–e105, Apr 2009.

[229] Nathan E. Green, S.-Y. James Chen, Adam R. Hansgen, John C. Messenger, Bertron M. Groves, and John D. Carroll. Angiographic views used for percutaneous coronary interventions: A three-dimensional analysis of physician-determined vs. computer-generated views. *Catheterization and Cardiovascular Interventions*, 64:451–459, 2005.

[230] Ariel Roguin, Sobhi Abadi, Ahuva Engel, and Rafael Beyar. Novel method for real-time hybrid cardiac CT and coronary angiography image registration: visualising beyond luminology, proof-of-concept. *EuroIntervention*, 4(5):648–653, Mar 2009.

[231] International Standards Organization. The virtual reality modeling language. *ISO/IEC*, 14772(1), 1997.

[232] D. Wang, Ivan Herman, and Graham J. Reynolds. The open inventor toolkit and the PREMO standard. *Computer Graphics Forum*, 16(4):159–175, 1997.

[233] John Rohlf and James Helman. IRIS performer: a high performance multiprocessing toolkit for real-time 3D graphics. In *Proceedings ACM SIGGRAPH*, pages 381–395, 1994.

[234] Henry A. Sowizral, Kevin C. Rushforth, and Michael F. Deering. *The Java 3D API specification*. Addison-Wesley, 1998.

[235] Dirk Reiners, Gerrit Voss, and Johannes Behr. OpenSG basic concepts. In *Proceedings OpenSG 2002 Symposium*, 2002.

[236] OSG. Open scene graph. `http://www.openscenegraph.org/`.

[237] nVidia. NVSG. `http://developer.nvidia.com/object/nvsg_home.html`.

[238] Arvind Sekar and Arthur H. Lee. Defining the universe: Creating a data model for a geospatial data repository. In *Proceedings SIWs Spring Simulation Interoperability Workshop*, 2004.

[239] Ralph M. Toms and Paul A. Birkel. Choosing a coordinate framework for simulations. In *Proceedings SISO Fall Simulation Interoperability Workshop*, 1999.

[240] Conrad Weisert. Point-extent pattern for dimensioned numeric classes. *ACM SIGPLAN notices*, 32(11):17–20, 1997.

[241] Erich Gamma, Richard Helm, Ralph Johson, and John Vlissides. *Design Patterns Elements of reusable object-oriented software*. Addison-Wesley Publishing Company, 1995.

[242] Jack E. Bresenham. Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4(1):25–30, 1965.

Publications

Book chapter

1. Onno Wink, Harvey S. Hecht, and Daniel Ruijters. Coronary Computed Tomographic Angiography in the Cardiac Catheterization Laboratory: Current Applications and Future Developments. *Cardiology Clinics, Advances in Coronary Angiography,* edited by S. J. Chen and J. D. Carroll, Volume 27, Issue 3, August 2009, pp. 513-529. doi:10.1016/j.ccl.2009.04.002

International Journal

1. Laurent Spelle, Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Jeremy Guillermic, and Jacques Moret. First clinical experience in applying XperGuide in embolization of jugular paragangliomas by direct intratumoral puncture. *International Journal of Computer Assisted Radiology and Surgery,* Volume 4, Number 6, November 2009, pp. 527-533. doi:10.1007/s11548-009-0370-6

2. Gert Schoonenberg, Raoul Florent, Pierre Lelong, Onno Wink, Daniel Ruijters, John Carroll, and Bart ter Haar Romeny. Projection-based motion compensation and reconstruction of coronary segments and cardiac implantable devices using rotational X-ray angiography. *Medical Image Analysis,* Volume 13, Issue 5, October 2009, pp. 785-792. doi:10.1016/j.media.2009.07.005

3. Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. Real-time integration of 3-D multimodality data in interventional neuroangiography. *Journal of Electronic Imaging,* Volume 18, Issue 3, July-September 2009. doi:10.1117/1.3222939

4. Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Vesselness-based 2D-3D registration of the coronary arteries. *International Journal of Computer Assisted Radiology and Surgery,* Volume 4, Number 4, June 2009, pp. 391-397. doi:10.1007/s11548-009-0316-z

5. Joel A. Garcia, Shyam Bhakta, Joseph Kay, Kak-Chen Chan, Onno Wink, Danny Ruijters, and John D. Carroll. On-line multi-slice computed tomog-

raphy interactive overlay with conventional X-ray: A new and advanced imaging fusion concept. *International Journal of Cardiology,* Volume 133, Issue 3, April 17, 2009, pp. e101-e105. doi:10.1016/j.ijcard.2007.11.049

6. Daniel Ruijters. Dynamic Resolution in GPU-Accelerated Volume Rendering to Autostereoscopic Multiview Lenticular Displays. *EURASIP Journal on Advances in Signal Processing,* Volume 2009, Article ID 843753, 8 pages, 2009. doi:10.1155/2009/843753

7. Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Efficient GPU-Based Texture Interpolation using Uniform B-Splines. *Journal of Graphics Tools,* Volume 13, Number 4, pp. 61-69, 2008

8. Daniel Ruijters and Anna Vilanova. Optimizing GPU Volume Rendering. *Journal of WSCG,* Volume 14, Number 1-3, January 2006, Pilzen (Czech Republic), pp. 9-16

9. Petr Dokládal, Isabelle Bloch, Michel Couprie, Daniel Ruijters, Raquel Urtasun, and Line Garnero. Topologically controlled segmentation of 3D magnetic resonance images of the head by using morphological operators. *Pattern Recognition,* Volume 36, Number 10, 2003, pp. 2463-2478. doi:10.1016/S0031-3203(03)00118-3

International Journal Abstract

1. Daniel Ruijters, Niels H. Bakker, Onno Wink, Bart M. ter Haar Romeny, and Paul Suetens. Integrating CT in Minimally Invasive Treatment of the Coronary Arteries. *In Proc. European Congress of Radiology - ECR 2008;* Vienna (Austria), C-200, March 7-11, 2008, *European Radiology,* Volume 18, Supplement 1, February 2008, p. 378. doi:10.1007/s10406-008-0003-0

2. Daniel Ruijters, Laurent Spelle, Jacques Moret, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. XperGuide: C-arm Needle Guidance. *In Proc. European Congress of Radiology - ECR 2008;* Vienna (Austria), C-591, March 7-11, 2008, *European Radiology,* Volume 18, Supplement 1, February 2008, p. 459. doi:10.1007/s10406-008-0003-0

3. Daniel Ruijters, Niels H. Bakker, Onno Wink, Mani Vembar, Guy Lavi, Bart M. ter Haar Romeny, and Paul Suetens. Integrating CT in Minimally Invasive Treatment of the Coronary Arteries. *1st Annual Scientific Meeting of the Society of Cardiovascular Computed Tomography,* July 2006, Washington DC (USA), *International Journal of Cardiovascular Imaging,* Volume 22, Suppl. 1, 2006, p. 12. doi:10.1007/s10554-006-0001-z

International Conference: Published in Proceedings

1. Svitlana Zinger, Luat Do, Daniel Ruijters, and Peter H. N. de With. iGLANCE: Interactive Free Viewpoint for 3D TV. *In Proc. 3D Stereo MEDIA 2009,* December 1-3, 2009, Liège (Belgium), 4 pages.

2. Daniel Ruijters and Svitlana Zinger. IGLANCE: Transmission to Medical High Definition Autostereoscopic Displays. *In Proc. 3DTV-CONFERENCE 2009: The True Vision - Capture, Transmission and Display of 3D Video,* May 4-6, 2009, Potsdam (Germany), 4 pages. doi:10.1109/3DTV.2009.5069626

3. Svitlana Zinger, Daniel Ruijters, and Peter H. N. de With. iGLANCE project: free-viewpoint 3D video. *In Poster Proc. 17th International Conference on Computer Graphics, Visualization and Computer Vision (WSCG),* February 2009, Pilzen (Czech Republic), pp. 35-38

4. Daniel Ruijters. Integrating Autostereoscopic Multi-View Lenticular Displays in Minimally Invasive Angiography. *In Proc. MICCAI 2008 workshop on Augmented Environments for Medical Imaging and Computer-Aided Surgery (AMI-ARCS),* September 10, 2008, New York (USA), pp. 87-94

5. Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Efficient GPU-Accelerated Elastic Image Registration. *In Proc. Sixth IASTED International Conference on BIOMEDICAL ENGINEERING (BioMed),* February 13-15, 2008, Innsbruck (Austria), pp. 419-424

6. Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. GPU-Accelerated Digitally Reconstructed Radiographs. *In Proc. Sixth IASTED International Conference on BIOMEDICAL ENGINEERING (BioMed),* February 13-15, 2008, Innsbruck (Austria), pp. 431-435

7. Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Accuracy of GPU-based B-Spline Evaluation. *In Proc. Tenth IASTED International Conference on COMPUTER GRAPHICS AND IMAGING (CGIM),* February 13-15, 2008, Innsbruck (Austria), pp. 117-122

8. Daniel Ruijters, Niels H. Bakker, Onno Wink, Bart M. ter Haar Romeny, and Paul Suetens. CT TrueView - Cardiac CT in the Cathlab. *In Proc. Annual Symposium of the IEEE/EMBS Benelux Chapter,* December 6-7, 2007, Heeze (the Netherlands), pp. 38-41

9. Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. 3D Multi-modality Roadmapping in Neuroangiography. *Proceedings of SPIE - Volume 6509, Medical Imaging 2007: Visualization and Image-Guided Procedures,* February 2007, San Diego (USA), pp. 65091F. doi:10.1117/12.708474

10. Daniel Ruijters, Marijke Vermeer, Anna Vilanova, and Paul Suetens. Robustness of Mutual Information Based Intra-Operative Registration. *First Annual*

Symposium of the IEEE/EMBS Benelux Chapter, December 7-8, 2006, Brussels (Belgium), pp. 171-174

11. Daniel Ruijters, Jeroen Terwisscha van Scheltinga, Bart M. ter Haar Romeny, and Paul Suetens. Design Pattern for Multi-Modality Coordinate Spaces. *10th Philips Software Conference*, November 2006, Veldhoven (the Netherlands), 8 pages

12. Daniel Ruijters. GPU-Accelerated Volume Rendering to 3D Lenticular Displays. *IEEE Workshop on "Content Generation and Coding for 3D-Television"*, June 2006, Eindhoven (the Netherlands), IEEE Benelux Chapter on Consumer Electronics, ISBN 90-386-2062-4 Vol. 1, 3 pages

13. Daniel Ruijters, Drazenko Babic, Bart M. ter Haar Romeny, and Paul Suetens. Silhouette Fusion of Vascular and Anatomical Data. *Poceedings of IEEE International Symposium on Biomedical Imaging: From Nano to Macro (ISBI'06),* April 2006, Washington DC (USA), pp. 121-124.
doi:10.1109/ISBI.2006.1624867

International Conference: Abstract

1. Daniel Ruijters, Drazenko Babic, Robert Homan, Peter Mielekamp, Bart M. ter Haar Romeny, and Paul Suetens. Frame-less C-arm Needle Guidance. *MICCAI 2008 Workshop on Needle Steering: Recent Results and Future Opportunities,* September 6, 2008, New York (USA)

2. Daniel Ruijters, Bart M. ter Haar Romeny, and Paul Suetens. Digitally Reconstructed Radiographs using the Graphics Hardware. *In Proc. Annual Symposium of the IEEE/EMBS Benelux Chapter,* December 6-7, 2007, Heeze (the Netherlands), p. 91

Patent Applications

1. D. Ruijters. Imaging Method for Sampling a Cross-Section Plane in a Three-Dimensional (3D) Image Data Volume. WO/2009/022283, International Filing Date: 11.08.2008

2. D. Ruijters, N. H. Bakker, and R. Homan. Coupling the Viewing Direction of a Blood Vessel's CPR View with the Viewing Angle on this 3D Tubular Structure's Rendered Voxel Volume and/or with the C-Arm Geometry of a 3D Rotational Angiography Device's C-Arm System. WO/2009/019640, International Filing Date: 31.07.2008

3. D. Ruijters. Fused Perfusion and Functional 3D Rotational Angiography Rendering. WO/2008/059417, International Filing Date: 09.11.2007

4. D. Ruijters and W. Lie. Method of Performing Tableside Automatic Vessel Analysis in an Operation Room. WO/2008/047266, International Filing Date: 09.10.2007
5. D. Ruijters, D. Babic, R. Homan, and P. Mielekamp. Determining Tissue Surrounding an Object being inserted into a Patient. WO/2007/113705, International Filing Date: 15.03.2007
6. D. Ruijters and P. Mielekamp. Silhouette Blend Rendering of Anatomical Structures. WO/2007/054863, International Filing Date: 02.11.2006

Curriculum Vitae

Daniel Ruijters has received his engineering degree at the University of Technology Aachen (RWTH), Germany, and performed his master thesis project at the École Nationale Supérieure des Télécommunications (ENST) in Paris, France. Since 2001 he is employed by Philips Healthcare. He started at the Cardio/Vascular X-ray software development department, and since 2004 he works at the Cardio/Vascular Innovation department, where he develops clinical prototypes in the area of 3D cardiovascular image processing and visualization. He is a senior scientist since 2007. Amongst others, he created multi-modal navigation software for minimally invasive vascular and cardiac treatment, and he developed clinical applications for the 3D monitor. His primary research interest areas are medical image processing, 3D visualization, image registration, fast algorithms and hardware acceleration.

Daniel has acted as session chair during the 2006 WSCG conference and the 2008 IASTED Conference on Computer Graphics and Imaging (CGIM), and was invited for the panel discussion of the 2008 MICCAI workshop on Augmented Environments for Medical Imaging and Computer-Aided Surgery (AMI-ARCS). He served as reviewer for the 2008 MICCAI High Performance Computing workshop, IEEE Transactions on Visualization and Computer Graphics, and European Radiology.

Appendix A

A Coordinate Space Framework

This chapter is based on the following paper:

- Daniel Ruijters, Jeroen Terwisscha van Scheltinga, Bart M. ter Haar Romeny, and Paul Suetens. Design Pattern for Multi-Modality Coordinate Spaces. *10th Philips Software Conference*, November 2006, Veldhoven (the Netherlands), 8 pages

A.1 Introduction

There are numerous applications with a large number of geometrical coordinate spaces (also known as coordinate systems or frame-of-reference), such as interactive 3D graphics and modelling applications, algorithms handling complex dynamic mechanical structures, coordinate transformations for astronomical or geodetic purposes (such as GPS), and many others. The relationships between the different coordinate spaces can be static or dynamic, bijective or many-to-one, continuous or contain discontinuities, and be affine or non-affine.

In the biomedical arena multiple coordinate spaces are encountered when dealing with multi-modal image registration and fusion, but also when considering the mechanical parts of an imaging system, or when establishing the geometrical relation between an image (*e.g.*, coming from an ultrasound probe) and the patient. Especially the dynamic integration of multiple image datasets from different sources at run time leads to an explosion of different coordinate spaces. Sometimes coordinate spaces that are very similar (*e.g.*, integer voxel coordinates at the center or at the corner of the voxel extent) lead to erroneous assumptions by programmers that are often undiscovered and persistent system inaccuracies are the consequence.

When the amount of coordinate spaces is large, the code dealing with coordinate transformations may become complex and prone to errors. To overcome these problems, we developed a software framework, to transparently and robustly deal with multiple coordinate spaces. This framework especially aims at severely reducing the chance of making false assumptions, and reducing the complexity of the code.

The framework can be applied for the transformation of data between related spaces of ordered n-tuples, such as vector spaces. It might be of interest to point out that these spaces are not necessarily Euclidian. For instance, a voxel space (a vector space representing voxel indices), whereby the voxels are not cubic (which is very common in *e.g.*, CT and MR volumes) is *not* Euclidian. Strictly speaking, the framework can also be applied to spaces of ordered n-tuples that are not vector spaces, *i.e.*, spaces whose elements cannot be linearly combined, such as manifolds. The only criterion is that a mapping exists between the spaces.

A.2 Related work

An intuitive hierarchical data structure for managing the relations between different coordinate systems is the scene graph. A scene graph is a collection of nodes in a tree structure. The nodes in a scene graph represent a coordinate space. Often a node is associated with a spatial object, such as a voxel dataset or a mesh in a biomedical application, or *e.g.*, an engine, a door or a steering wheel in an automotive modeling application. The edges or links between the nodes contain their spatial relationship. A node may have many children but only a single parent. The spatial transformation between any two nodes in the scene graph can be established by concatenating all spatial transformations on the path between them. A property of the tree hierarchy is the fact that a change of the spatial transformation of a node to its parent, automatically affects the children of that node in the same way. Parent nodes, therefore, act as compound objects to their children, which can then be moved, transformed, selected, *etc*. as easily as a single object.

Scene graphs are used in numerous graphics applications, toolkits, modelling and programming languages, such as VRML [231], Open Inventor [232], OpenGL Performer [233], Java 3D [234], Open SG [235], Open Scene Graph [236], nVidia NVSG [237] and many others. Where VRML is only able to describe the scene graph, the others also provide some means to transform points from one coordinate system to another. The transformations are, however, still very much driven by the user of these tools, *i.e.*, it is his responsibility to keep track of which data are in which coordinate space, and whether and how they should be transformed.

Zuiderveld and Viergever [70] describe an Object-Oriented approach aimed at integrated visualization of multiple volumetric datasets, which also deals with coordinate systems. However, they chose to leave the coordinate transformations to the responsibility of the user of their framework. Nadeau [72] presents volume scene graphs, a structure for composing scenes containing volumetric data sets, where the scene graph is used to transform coordinates from world to image space. Other transformations, though, are not directly provided by his framework.

In geospatial applications [238, 239] it often is desirable to express points in different geometry systems, such as geocentric, heliocentric or local coordinate systems. For reasons of efficiency or simplicity it can be desirable to express coordinates in flat earth or spherical earth coordinate systems, and for accuracy ellipsoid or geoid coordinate space may be required. The software paradigm in this chapter can be used to easily query data in the desired coordinate system.

There is a significant difference between a point and an extent (the distance between two points), as Weisert [240] points out. This difference is particularly of importance when transforming data from one coordinate system to another one. For example, translations do not affect the values of an extent, but they do affect the coordinates of a point (a ruler of 30 cm retains its size of 30 cm when we move it from Eindhoven to Leuven).

The here presented framework can be regarded as a software design pattern for dealing with a large number of coordinate spaces. A general overview of design patterns is offered by Gamma et al. ("the gang of four") [241].

A.3 Design basics

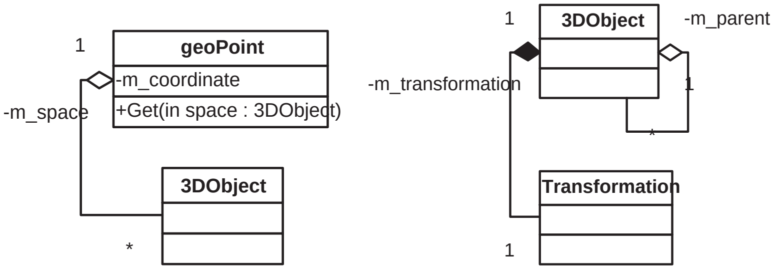
We define "geometry classes" as a set of object classes, describing basic geometrical entities, such as points, vectors, lines, angles, planes, *etc*. To identify them easily, we use the prefix *"geo"*. Instances of these classes are generalized under the term "geometry objects". It is our objective to easily query them in any given coordinate space. A further important class in our framework is the *3D object*. A *3D object* is a node in the scene graph and contains a number of geometry objects to describe its spatial properties. Any spatial entity that can be drawn should be derived from the *3D object* class. But there can be also abstract *3D objects* that do not draw anything, but merely represent an abstract frame of reference.

One of the first observations we make, is the fact that any *3D object*, which can be found in a scene graph, implicitly defines its own coordinate space. Consider a traditional *3D object* (*e.g.*, a table), which is located in a parent space (*e.g.*, a room). The object has a translation, which corresponds to the coordinate of the origin of the object expressed in its parent space. Further it can have a rotation, which corresponds to a rotation of its axes with regard to the axes of the parent space, and a scaling. In fact we have just described a rigid transformation between two coordinate spaces.

In our software paradigm, an abstract coordinate space is an instance of a *3D object*, and therefore every instance of a class, inherited from the *3D object* class, always defines its own coordinate space.

Further, we establish that the coordinates of a point are always defined in a coordinate space. This may be a trivial observation. In an application with many coordinate spaces, though, treating a datum in the wrong coordinate space is one of the most common causes for bugs, and may be difficult to track when coordinate spaces are similar or related. The same considerations are of course true for instances of other geometry classes, *e.g.*, normals, lines, planes, *etc*. Therefore we provide all geometry classes, with a reference to the coordinate space they are internally defined in (see figure A.1a). In this way it is virtually impossible to 'assume' a wrong coordinate space.

The internal coordinate space of an instance of such a class is defined at construction of the instance, and stays fixed during the lifetime of the instance. This is particularly of importance when the relations between the coordinate spaces are dynamic. Imagine, for instance, a camera, which moves with respect to the depicted scene; the relation between the camera coordinate space and scene coordinate space



(a) (b)

Figure A.1: *(a) the geoPoint class has internal coordinate values, and a reference to its internal coordinate space, (b) every 3DObject contains a reference to its parent, and a spatial transformation with regard to its parent.*

is then dynamic. It matters whether the coordinates of a point are defined in camera space (*e.g.*, relative to the view port corners), or in scene coordinates (*e.g.*, relative to the position of an object in the scene).

A.4 Transforming space

The *Transformation* class describes the spatial mapping between an instance of a *3D object* and its parent in the scene graph (see figure A.1b). This is an abstract class, and specific transformation classes, such as affine transformations, are inherited from this class.

The *Transformation* class possesses virtual functions to transform coordinates from its owner space to the parent of its owner, and vise versa (this approach is similar to the Visitor design pattern [241]). These functions return a boolean to indicate whether the requested transformation could be performed. In this way also many-to-one relations could be implemented; the function corresponding to the one-to-many direction would then always return false. Further, transformations that are only valid for a certain sub-space can use this mechanism, since they would return false for points outside the sub-space. Similar virtual functions are available for transforming all other geometry objects, like vectors, matrices, plane equations, *etc*.

A rigid transformation can be described by translation and rotation only. In the case of an affine transformation (of which the rigid transformation is a sub class) the virtual transformation functions can be implemented by multiplying homogeneous coordinates (x, y, z, w), representing points or vectors, with the 4*4 transformation matrix (or inverted matrix, for the inverse transformation). A typical implementation for elastic (non-affine) transformations could use spline interpolation, driven by a volumetric mesh.

Since no assumptions are being made about the type of transformation, the various relations in a scene graph might be of a different kind (*e.g.*, affine and non-affine transformations could be found in the same scene graph, to depict for instance underwater scenes).

A.5 Querying geometry objects

One of the most essential functions that any geometry class has in our framework, is the *Get* function (see figure A.1a). The *Get* function allows to query a geometry object with respect to a given coordinate space. The *Get* function of *e.g.*, the *geoPoint* class takes a reference to a coordinate space as input parameter, and returns the coordinate values of the point with respect to the passed coordinate space.

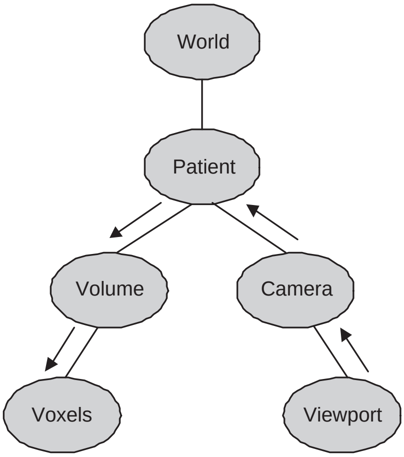


Figure A.2: *Traversing the scene graph.*

If the passed reference to a coordinate space equals the internal space of a geometry object, its internal values are simply returned. If they are not equal, the internal values are transformed from the internal coordinate space to the destination one, passed as input parameter. In order to do this, the scene graph is traversed, delivering the path from the internal coordinate space to the destination one. The transformations between the intermediate coordinate spaces in the path are then applied to the geometry object. If a transformation returns false (thus it is not possible to transform the geometry object over that node), or if no path exists between the internal and destination space (*i.e.*, they are not in the same scene graph), an exception is thrown.

Four rigid transformations of a coordinate, as is shown in figure A.2, take 2.7 μs on a Pentium IV 3.0 GHz machine.

A.6 Traversing the scene graph

When the coordinate space that is passed to the *Get* function of a geometry object differs from the internal one, the scene graph has to be traversed. In order to perform this efficiently two arrays with references to the nodes in the scene graph are built. The ascending array starts with the internal coordinate space of the geometry object. Iteratively the parent of the last node in the array is added, until the top node is reached. The descending array starts with the destination coordinate space, and also here parents are added until the top node is reached. Now it is checked whether the last node (the top node) in both arrays is the same. If this is not the case, meaning that the internal and destination coordinate spaces are not in the same scene graph, an exception is thrown.

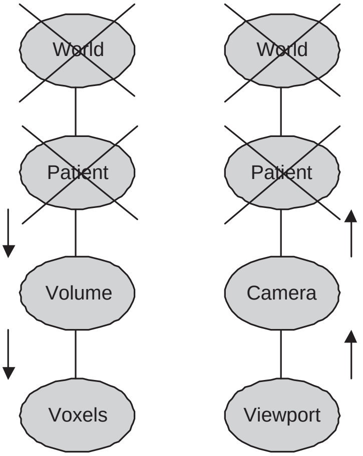


Figure A.3: *Building the path that represents the traversal of the scene graph.*

After the check, the nodes at the end of both arrays are removed if they are the same. This is repeated until the ends are different, see figure A.3. The remaining nodes now form the path that has to be traversed. The data of the geometry object is consequently transformed by the nodes in the ascending array, starting with the first node in the array. Then the data is transformed by the nodes in the descending array, calling the inverse transformation functions. This array is parsed starting from the end, see figure A.3. Note that the order of this algorithm is only determined by the height of the scene graph, not by its width.

A.7 Operator overloading

Another important feature is the fact that we overloaded the operators of the *geoPoint* and *geoVector* classes. Note that two points cannot be added together, but a point and a vector can be added, delivering a new point [240]. By overloading the operators, we can even add points and vectors, which are internally expressed in a different coordinate system.

For instance let us consider the position of an object in the scene, expressed by an instance of the *geoPoint* class, with the world coordinate system as internal space (expressed in *e.g.*, millimeters), and a *geoVector* instance, expressing a mouse movement, with the view port coordinate system as internal space (in pixel coordinates). Suppose we want to translate the object by the mouse movement. The corresponding code could be as simple as `objPos = objPos + mouseMove;` as is shown in the following code:

```
void Translate(geoPoint& objPos, const geoVector& mouseMove)
{
    objPos = objPos + mouseMove;
}
```

The function *Translate* illustrates the code that a user of the Coordinate Space Framework would write. Note that the internal coordinate spaces of the variables in this expression typically will be different. The *operator+* function shows how the framework deals with this code internally. The `mouseMove` variable is queried in the coordinate space of the `objPos` variable:

```
geoPoint geoPoint::operator+ (const geoVector& vec) const
{
    return geoPoint(m_coordinates + vec.Get(m_space), m_space);
}
```

The overloaded *operator+* of the *geoPoint* class will take care that `mouseMove` is transformed to the coordinate space of `objPos`. The *Get* member function of the *geoVector* class will transform the internal values of the `vec` variable from its own internal space to the internal space of the `objPos` variable, and then the two can be added without any problems. This feature leads to very powerful and simple code, as illustrated in the *Translate* function, since the code expresses what you want to achieve conceptually, instead of expressing all kinds of difficult coordinate transformations.

A.8 Real life examples

A.8.1 Mouse click on a voxel volume

Take an iso-surface rendered voxel volume, and suppose we want to determine which surface voxel lies under the mouse cursor at a mouse click. In order to do so, a line through the cursor position (viewing ray) has to be intersected with the iso-surface. This line is defined by the cursor position and the camera normal, in the case of a parallel projection, and by the cursor position and the camera focus point in the case of a perspective projection. Using the presented framework, it is no problem to define a line from two points which are constructed in different coordinate systems. After the line has been defined, it can be easily obtained in voxel space, using the *Get* function. Then it should be passed to a 3D variant of Bresenham's algorithm [242], to deliver the intersection point.

A.8.2 Defining points of interest

Let's consider an application with two windows next to each other, in order to view two registered multi-modality volumetric data sets. In one view a slice of the reference data set is displayed, while in the other the corresponding interpolated surface

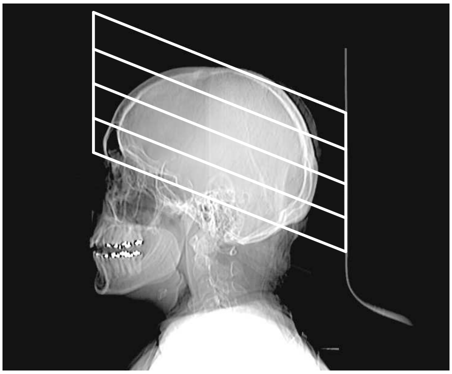


Figure A.4: *Planning the location of the CT slices, with tilted gantry. The gantry is tilted to avoid radiating the eyes, while capturing a maximum of relevant anatomical data.*

through the other data set is shown (non-affine registration). A mouse click marks a point of interest in one data set. The point can be constructed as follows: `geoPoint mousePnt(viewport1.x, viewport1.y, 0);` Drawing the corresponding point in the other view is as simple as: `Plot(mousePnt.Get(viewport2));`, assuming that the *Plot* function is a library function that takes pixel coordinates as input.

Note that the *Get* function transforms the `mousePnt` coordinates first from viewport1 (pixels) to world coordinates (millimeters), which is a rigid transformation. Then the coordinates are transformed from world coordinates to the frame of reference of volume2, which is a non-affine transformation, executed by a different transformation class. Finally the coordinates are transformed from the frame of reference to viewport2, which is a rigid transformation again, delivering the pixel coordinates of the point to be plotted. This complete procedure remains hidden for the user of the framework.

A.8.3 Gantry-tilt CT volumes

CT volumes, which have been acquired with a tilted gantry, produce a voxel space with non-orthogonal axes (see figure A.4). Typically such volumes are resampled on a orthogonal grid for volumetric visualization, leading to loss of image quality. However, it is possible to encapsulate the shearing (skew) that is introduced by the non-orthogonality in an affine transformation (*e.g.,* expressed in a 4*4 matrix). In this way the data can be depicted without resampling, using our framework.

For instance, a ray-cast algorithm could define the viewing rays in camera space, and feed them to the interpolator. The interpolator queries the respective rays in voxel space, and the *Get* function takes care of the transformation from camera to world space to voxel space. The latter step involves the shear operation.

A.8.4 Follow camera orientation

Suppose we want one single object in the 3D scene always to be presented with the same side to the camera. This object, however, should be positioned and scaled according to its location in the scene (*i.e.*, moving camera could change only the rotation of the object, but not its translation or scaling). To solve this task, we can define to `geoVector` instances in camera space, representing the x- and y-axis of the camera. For the x-axis this can look like:

```
geoVector x_camera(1,0,0,cameraSpace);
```

Now we will rotate the object such that its x-axis will point in the direction of the camera x-axis. To do so we query the camera x-axis in the coordinate space of the object: `x_camera.Get(objectSpace);` The nice thing is that this produces the orientation of the camera x-axis in the object space instantaneously, no matter how many nodes there are between the camera and the object in the scene graph. The vector still has to be normalized, and then the dot product between this vector and the object x-axis (which is simply (1,0,0)) delivers the cosine of the angle that we should rotate. The cross product delivers the rotation axis. The same procedure can be followed to orient the y-axis correctly.

A.9 Conclusions

In this chapter we have introduced a generic software solution for a flexible and transparent framework for handling multiple coordinate spaces. The proposed framework is especially powerful when the number of coordinate spaces is large and their relations are dynamic, such as is *e.g.*, the case in multi-modality medical applications.

The complexity of dealing with multiple coordinate spaces lies in the transformation between the individual spaces. The strength of the proposed framework is the fact that these transformations are maintained at a single spot, and in the rest of the code no awareness of these transformations is needed. The resulting code expresses conceptually what the programmer wants to achieve, instead of expressing all kinds of difficult coordinate transformations.

In the case that the actual values of a geometry object are needed with respect to a certain coordinate space, these can only be obtained by explicitly passing the desired coordinate space to the *Get* operation. This severely reduces the chance of ‘assuming’ a wrong coordinate space, one of the most common causes of bugs in such applications. If a transformation is needed from the internal coordinate space to the requested one, the transformation is performed automatically, and hidden from the user of the function call.

It is worth mentioning that the described framework has been successfully implemented in two medium and three large scale software projects.