Sander van Reek
30350
3-7-2012

Deformatiebepalingen in MRI- en ultrageluidafbeeldingen

THRIJSWIJK Haagse Hogeschool
Erasmus MC Universitair Medisch Centrum Rotterdam
elastiX

Samenvatting

Bij MRI en ultrageluidafbeeldingen, gemaakt van een bloedvat bij een varken, is berekend hoe het bloedvat deormeerd bij een verandering in druk. Hierbij is gebruikt gemaakt van het softwarepakket Elastix, waarbij eerst met behulp van tests met een geïdealiseerde geometrie is bepaald welke instellingen nodig zijn. Hiermee is berekend he de pixels verplaatsen. Uit dit verplaatsingsveld wordt vervolgens berekend hoe de pixels vervormen.

Dit deformatieveld kan vergeleken worden met het deformatieveld dat is berekend met een eindige-elementenmethode die wordt gebruikt om het risico van het scheuren van een plaque in te schatten. Hiermee kan deze methode worden gevalideerd.

Inhoudsopgave

1 Inleiding

Deze scriptie beschrijft de afstudeeropdracht over de bepaling van vervormingen binnen bloedvaten bij verschillende drukken. Eerst zal als achtergrondinformatie worden beschreven waar de afstudeeropdracht wordt uitgevoerd, wat het doel van de opdracht is en in detail wat de opdracht is. Hierna zal de werking van een van de gebruikte softwarepakketten worden beschreven. Vervolgens zullen de uitwerking en resultaten van het project worden uitgelegd.

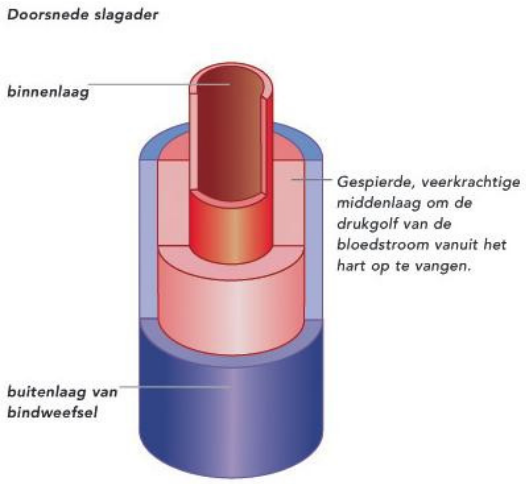
1.1 Bedrijfsinformatie

Het Erasmuc MC is een universitair medisch centrum in Rotterdam. Het is voortgekomen uit een fusie van het Dijkzigtziekenhuis, het Sophia kinderziekenhuis, de Daniel den Hoed kliniek en de Faculteit der Geneeskunde en Gezondheidswetenschappen van de Erasmus Universiteit Rotterdam.

De afstudeeropdracht wordt uitgevoerd bij de afdeling Biomedical Engineering. Deze afdeling is onderdeel van het Thoraxcenter, en richt zich op de oorzaak, diagnose en behandeling van cardiovasculaire aandoeningen, met name aderverkalking in bloedvaten.

1.2 Opbouw bloedvat

Een bloedvat is opgebouwd uit 3 lagen. Dit zijn van buiten naar binnen de adventitia, media en intima. De adventitia bestaat voornamelijk uit bindweefsel waarmee het bloedvat aan omringende organen vast zit. De media bestaat uit spierlagen en collageen voor stevigheid en elastine voor elasticiteit. De intima bestaat uit endotheel cellen en is de binnenbekleding van de vaatwand. Het lumen is het binnenste van het bloedvat, waar het bloed doorheen stroomt. Een schematische voorstelling[1] van een bloedvat is te zien in onderstaand figuur.

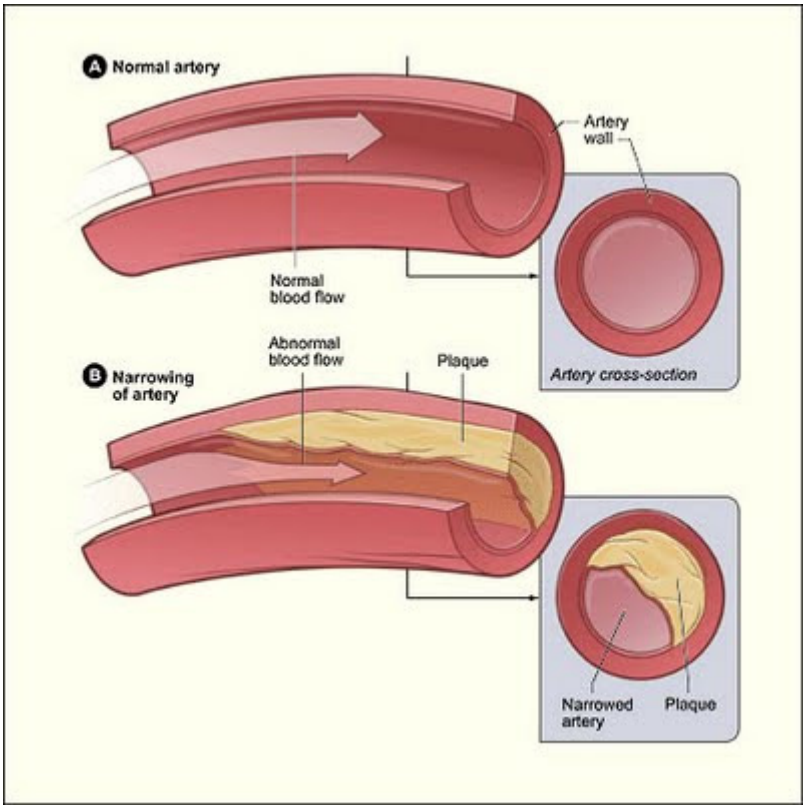


Figuur 1: Doorsnede van een slagader.

1.3 Plaque en Atherosclerose

Atherosclerose is een ziekte waarbij vetten in de bloedvatwand worden afgezet. Ontstekingscellen proberen deze vetten op te nemen, maar sterven hierbij en worden in de bloedvatwand opgenomen en vormen een plaue. Hierbij wordt vooral de intima aangetast[2]. Deze plaques bestaan voornamelijk uit vetten, ontstekingscellen, collageen weefsel en calcium. De lipide kern is het deel van de plaque met voornamelijk ontstekingscellen[3]. Een voorbeeld hiervan is te zien in onderstaande afbeelding[4].

Tussen het lumen en de lipide kern is een dunne laag van collageen en spiercellen. Dit is de cap. Door de bloeddruk en de stroming van het bloed door het lumen worden er krachten op deze cap uitgevoerd. Als deze krachten te groot worden scheurt de cap. De rode bloedcellen en bloedplaatjes komen in aanraking met de lipide kern waardoor een stolling ontstaat. Deze stolling kan de stroming in het bloedvat blokkeren met een hartinfarct of herseninfarct als gevolg.



Figuur 2: Schematische voorstelling van een bloedvat zonder en een bloedvat met plaque.

1.4 Opdrachtomschrijving

Van een bloedvat met plaques van een varken zijn Magnetic Resonance Imaging (MRI) en ultrageluidafbeeldingen gemaakt. Deze afbeeldingen zijn gemaakt met verschillende interne druk op dezelfde locatie. De opdracht is om uit deze afbeeldingen de vervormingen in het bloedvat als gevolg van het verschil in interne druk te bepalen. Deze vervormingen kunnen worden gebruikt om theoretische modellen voor de bepaling van de mechanische spanningen in de bloedvatwand te valideren. Dit wordt gedaam met een eindige-elementenmethode om het risico van het scheuren van een plaque in te schatten.

1.5 Plan van aanpak

De vervormingen kunnen worden bepaald met het softwarepakket Elastix. Om bekend te raken met deze software is als eerste gewerkt met een symmetrische geïdealiseerde geometrie. Hierop zijn verschillende bewerkingen uitgevoerd zoals een kleine verplaatsing, draaiing en vervorming. Er is gekeken hoe goed Elastix verplaatsingen van pixels kan bepalen. Hierbij is ook bepaald wat de effecten zijn van de verschillende instellingen in Elastix, welke zijn beschreven in hoofdstuk 2. Het doel was om de juiste instellingen te kunnen gebruiken voor MRI- en ultrageluidafbeeldingen.

Vervolgens zijn de MRI en ultrageluidafbeeldingen bekeken. Hierbij was het doel om de vervormingen in de afbeeldingen te bepalen; en te analyseren hoe deze afhangen van de interne druk in het bloedvat.

2 Elastix

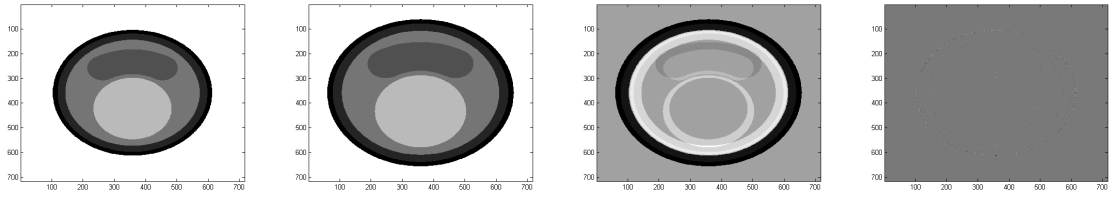
Om de vervormingen te berekenen wordt gewerkt met het softwarepakket Elastix. Dit is een programma dat afbeeldingen vergelijkt. Het is geschreven om medische afbeeldingen te vergelijken, maar kan ook voor andere toepassingen worden gebruikt. Er zijn veel instellingen die bepalen hoe het afbeeldingen worden vergeleken. De belangrijkste instellingen zullen hier worden beschreven.

2.1 Registratie

Door twee afbeeldingen te nemen, en hierbij per pixel de grijswaarden van de ene afbeelding van de andere afbeelding af te trekken, wordt een registratie van de afbeeldingen gemaakt. Hoe beter de afbeeldingen op elkaar lijken, hoe kleiner de grijswaarden van de registratie zijn.

Door één van de afbeeldingen te verplaatsen of vervormen vóór de registratie wordt gedaan, veranderen de grijswaarden van de registratie. Door verschillende transformaties uit te voeren kan de registratie worden geoptimaliseerd. Hierbij wordt de grijswaarde van de registratie geminimaliseerd. Als deze optimale registratie bekend is, zijn dus ook de nodige transformaties bekend. Uit deze transformaties kan bepaald worden hoe de afbeeldingen van elkaar verschillen. In de onderstaande afbeelding is het verschil te zien in de registratie voor en na de optimalisatie.

Een van de bestanden die Elastix als uitvoer geeft is de resultaatsafbeelding. Dit is de tweede, of 'moving' afbeelding die dusdanig bewerkt wordt, dat deze zo goed mogelijk overeenkomt met de eerste, of 'fixed' afbeelding. Met MATLAB kan dit resultaat gecontroleerd worden door de fixed afbeelding van de resultaatsafbeelding af te trekken.



Figuur 3: Voorbeeld van de optimalisatie van de registratie. De meest linkse afbeelding is de fixed afbeelding. Rechts hiervan is de moving afbeelding. De derde afbeedling is de registratie voordat er optimalisatie is toegepast. De meest rechtse afbeelding is de registratie na de optimalisatie.

Een ander bestand dat Elastix als uitvoer geeft, is het verplaatsingsveld. Dit is een matrix van getallen waarin is aangegeven hoe elke pixel in de tweede afbeeldingen moet worden verplaatst om de resultaatsafbeelding te krijgen. Met MATLAB kan dit verplaatsingsveld worden gevisualiseerd om een duidelijk beeld te krijgen in welke richting en met welke mate de pixels verplaatsen, en dus hoe de twee afbeeldingen van elkaar verschillen. Ook kan hieruit de vervorming van de pixels berekend worden, en daarmee het deformatieveld.

2.2 Transformaties

Er zijn verschillende transformatiemodellen die gebruikt kunnen worden in Elastix. Deze kunnen worden opgedeeld in rigide transformaties en niet rigide transformaties. Rigide transformaties verplaatsen de hele afbeelding. Niet rigide transformaties vervormen delen van de afbeelding. In figuur 4 en 5 zijn deze transformaties schematisch weergegeven.

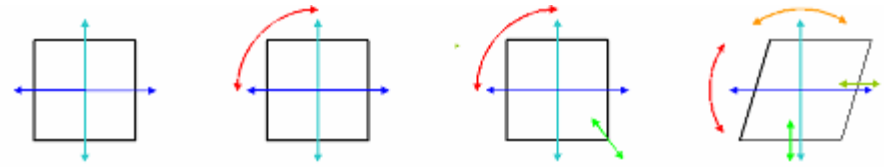
2.2.1 Rigide transformaties

De eenvoudigste rigide transformatie is de TranslationTransform. Deze kijkt alleen naar translaties. Het voordeel van een transformatie die niet naar bijvoorbeeld rotatie kijkt is dat als al bekend is dat er geen rotaties zijn, er hierdoor ook geen onnauwkeurigheid op kan treden.

Bij de EulerTransform worden zowel de translatie als de rotatie bepaald.

De SimilarityTransform bepaalt de translatie, rotatie en verandering in schaal.

De AffineTransform bepaalt de verplaatsing, rotatie en verandering in schaal van de afbeelding. Hierbij wordt de rotatie en verandering in schaal in de x- en y-richting apart bepaald, met als gevolg dat er ook vervormingen in de afbeelding kunnen zijn.



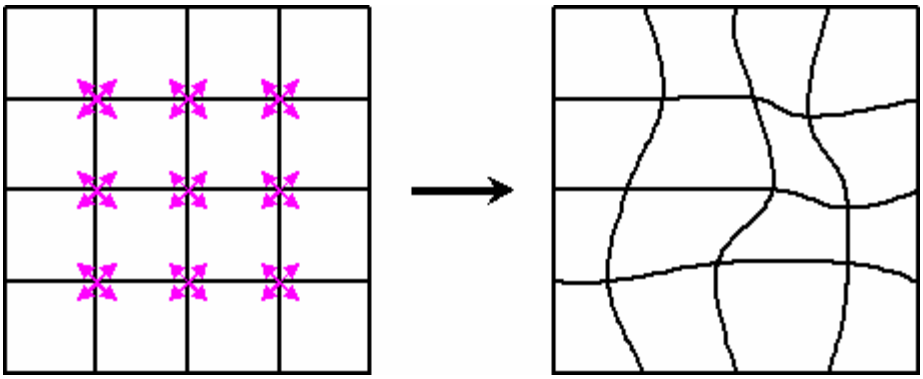
Figuur 4: Rigide transformaties. De vierkanten stellen de afbeelding voor en de pijlen geven de transformaties aan. Van links naar rechts zijn de TranslationTransform, de EulerTransform, de SimilarityTransform en de AffineTransform weergegeven.

2.2.2 Niet rigide transformaties

Niet rigide transformaties kijken naar verschillen binnen de afbeelding. Er wordt hierbij van uit gegaan dat de afbeeldingen niet ten opzichte van elkaar zijn verschoven. Als er grote rigide transformaties aanwezig zijn is het dus belangrijk om deze te bepalen voordat er naar niet rigide transformaties wordt gekeken.

De BSplineTransform maakt een vast raster met knopen. Door deze knopen te gebruiken voor splines, en vervolgens de knopen te bewegen, kunnen vloeiende vervormingen gevonden worden.

De SpineKernelTransform maakt gebruik van door de gebruiker opgegeven knopen in plaats van een vast raster.



Figuur 5: Niet rigide transformatie. Het grote vierkant geeft de afbeelding aan. De kleine vierkanten geven de locaties van de knopen aan. De pijlen geven de verplaatsingen van de knopen aan. Het rechter vierkant is een voorbeeld van de vervorming binnen een afbeelding als gevolg van de verplaatsing van de knopen.

2.3 Parameters

Om de gewenste resultaten te krijgen is het belangrijk om Elastix de juiste berekeningen uit te laten voeren. Dit wordt gedaan door de instellingen in de parameterbestanden te wijzigen. Behalve de transformatieparameter die bepaalt welke transformatie wordt uitgevoerd, zijn er veel parameters die bepalen hoe dit wordt gedaan.

2.3.1 Registratie

Elastix kan de afbeeldingen meerdere malen vergelijken. Hierbij wordt voor de eerste vergelijking de resolutie van de afbeeldingen sterk omlaag gehaald. Voor elke opeenvolgende vergelijking wordt de resolutie verhoogd, zo dat de laatste vergelijking de oorspronkelijke resolutie gebruikt. De standaard parameter hierbij is MultiResolutionRegistration. Hierbij worden voor elke resolutie dezelfde parameters gebruikt. Bij MultiResolutionRegistrationWithFeatures kunnen voor elke resolutie andere parameters, zoals de interpolatie en metriek, gebruikt worden.

2.3.2 Interpolatie

Interpolatie wordt gebruikt bij niet rigide transformaties. Dit is om de deformatie in de gebieden tussen de knopen te bepalen.

De eenvoudigste interpolatie is de NearestNeighborInterpolator. Hierbij wordt elke pixel op dezelfde manier verplaatst als de dichtstbijzijnde knoop.

De RayCastInterpolator is een lineaire interpolatie. De pixels worden verplaatst afhangend van de verplaatsing in de vier omringende knopen.

De BSplineInterpolator maakt gebruik van splines tot de derde orde om de deformatie te bepalen. Deze interpolatie geeft vloeiende vervormingen. Het resultaat hiervan zal over het algemeen het beste overeen komen met de werkelijke verplaatsing.

2.3.3 Metriek

De metriek is de mate van overeenkomst die het registratiealgoritme probeert te optimaliseren. Er zijn hierbij verschillende mogelijkheden. Welke het beste resultaat geeft, hangt af van de toepassing.

Bij de AdvancedMattesMutualInformationMetric wordt voor beide afbeeldingen een dichtheiddistributie gemaakt met behulp van histogrammen[5]. Dit is een methode om ruis te onderdrukken zodat de vorm van een afbeelding bepaald kan worden. Vervolgens wordt de informatie die in beide distributies gelijk is gebruikt voor de bepaling van de deformaties.

AdvancedMeanSquaresMetric bepaalt het verschil in de waarde van de pixels en kwadrateert deze. Dit is een vrij simpele methode, waarbij de registratie het beste is als de som van alle kwadraten zo klein mogelijk is.

AdvancedNormalizedCorrelationMetric maakt een correlatie tussen de twee afbeeldingen, en normaliseert deze met de autocorrelaties van de afbeeldingen. Deze genormaliseerde correlatie is gedefinieerd als:

$$NC = \frac{\sum_z f(x) * m(x + u(x, p))}{\sqrt{\sum_z f(x)^2 * m(x + u(x, p))^2}}$$

Hierbij is x een pixel, f de fixed afbeelding, m de moving afbeelding en $u(x, p)$ de verplaatsing van x gebaseerd op de transformatieparameters.

De NormalizedMutualInformationMetric gebruikts net als de AdvancedMattesMutual-InformationMetric een dichtheidsdistributie. Het verschil tussen deze metrieke stelsels is dat bij de NormalizedMutualInformationMetric de afbeeldingen worden genormaliseerd voor er naar overeenkomsten wordt gekeken.

2.3.4 ImagePyramids

De ‘ImagePyramid’ is de manier waarop de lagere resolutie afbeeldingen worden bepaald. Er zijn hierbij 3 verschillende opties.

De MultiResolutionShrinkPyramidImageFilter verlaagt de resolutie door een deel van de pixels van de hogere resolutie afbeelding te nemen als lage resolutie afbeelding. Er worden geen technieken gebruikt om voor vloeiende overgang te zorgen.

De MultiResolutionGaussianSmoothingPyramidImageFilter vervaagt de afbeelding. De afbeelding wordt hierbij niet kleiner, maar er gaan wel details verloren.

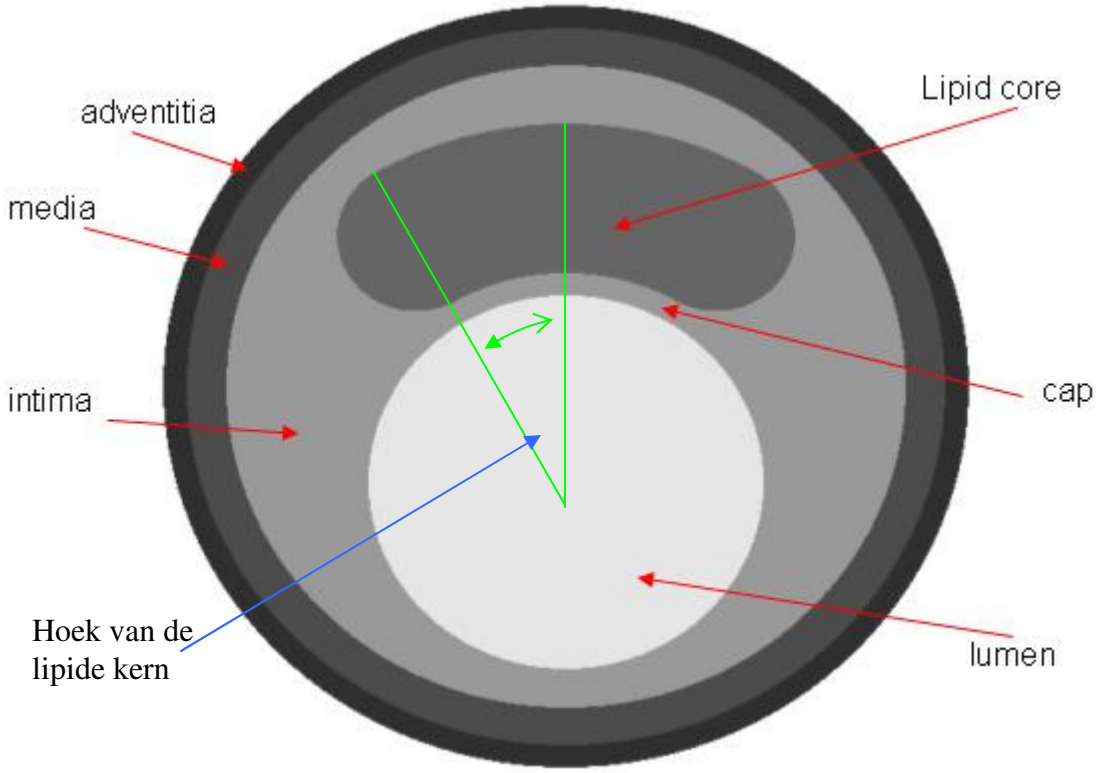
De RecursiveMultiResolutionPyramidImageFilter vervaagt de afbeelding per stap, om deze vervolgens te verkleinen door maar een deel van de pixels te nemen.

2.3.5 Overige parameters

De parameters die hiervoor zijn beschreven bepalen welke registraties worden uitgevoerd. Enkele van deze parameters hebben ook optionele parameters, waarmee in meer detail aangegeven kan worden hoe deze parameters werken. Hierbij gaat het onder andere over de afstand tussen de knopen voor de BSpline transformatie, het aantal resoluties waarbij de registratie wordt uitgevoerd of het aantal maal dat de deformatie wordt verbeterd.

3 Geïdealiseerde geometrie

De uitvoer van Elastix hangt af van de vele verschillende instellingen. Om te bepalen welke instellingen goed werken om de gewenste informatie te krijgen, zal eerst getest worden met afbeeldingen op basis van zelf opgelegde vervormingen van een geïdealiseerde geometrie[6]. Aangezien hierbij alle informatie van te voren bekend is, kan de uitvoer van Elastix goed worden vergeleken met de verwachte uitvoer.



Figuur 6: Symmetrisch geïdealiseerde basisgeometrie van de doorsnede van een bloedvat met lipide kern.

3.1 Symmetrisch geïdealiseerde basisgeometrie

In MATLAB is een script geschreven om afbeeldingen te maken gebaseerd op symmetrische basisgeometrie van een bloedvat. Met dit script kunnen de vormen en afmetingen van de verschillende onderdelen worden gevarieerd. Ook kan hierbij de kwaliteit van de afbeeldingen worden veranderd door onder andere de toevoeging van ruis. Dit script staat in bijlage 6. Een voorbeeld van de geometrie is te zien in bovenstaand figuur.

De basiswaarden voor de groottes van de verschillende onderdelen staan tabel 1.

Tabel 1: Basis waarden voor de afmetingen van de onderdelen in de basis geometrie van de doorsnede van een bloedvat.

Onderdeel	Basiswaarde [mm]
Lumen (straal)	1.25
Cap (dikte)	0.05
Lipide kern (dikte)	1.20
Lipide kern (hoek)	30°
Intima (straal)	2.15
Media (dikte)	0.25
Adventitia (dikte)	0.15

3.2 Optimale kwaliteit

De eerste testen zijn met afbeeldingen met zowel geïdealiseerde geometrie als met hoge resolutie en hebben geen ruis of artefacten. De gebruikte resolutie is 512 bij 512 pixels.

3.2.1 Rigide verplaatsing

De eerste test is om een afbeelding enkele pixels te verplaatsen. Hieruit kan bepaald worden hoe exact Elastix zulke verplaatsingen kan bepalen.

Bij een verplaatsing van 1 tot 20 pixels in de x-richting, y-richting, en zowel x- als y-richting is de nauwkeurigheid van de verplaatsing bepaald.

Tabel 2: Nauwkeurigheid in affiniteit bij translaties in de positieve x-richting. De grootste fout is de grootste afwijking in berekende verplaatsing ten opzichte van de werkelijke verplaatsing.

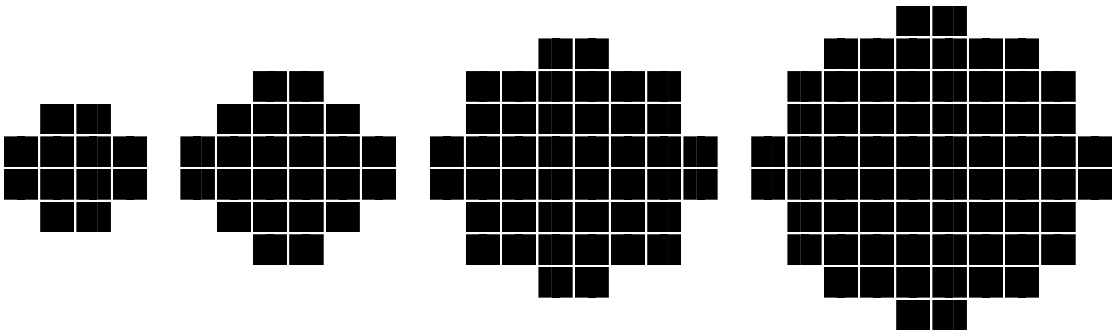
Translatie in x-richting [pixels]	Grootste fout in x-richting [pixels]	Grootste fout in y-richting [pixels]
1.0000	0.0150	0.0441
2.0000	0.0055	0.0187
4.0000	0.0159	0.0250
6.0000	0.0098	0.0291
8.0000	0.0387	0.0187
10.0000	0.0140	0.0427
12.0000	0.0077	0.0243
16.0000	0.0286	0.0193
20.0000	0.0041	0.0272

De resultaten voor verplaatsing in de positieve x-richting staan in bovenstaande tabel en in figuur 29 in bijlage 1. Hierin is duidelijk te zien dat de uiteindelijke afwijking onafhankelijk is van de oorspronkelijke verplaatsing. De grootste afwijking is minder dan 0.05 pixels. De afwijking bij verplaatsingen in andere richtingen komen hier ook mee overeen.

Ook wordt er bij elke verplaatsing een kleine verandering in schaal en rotatie gevonden. In de x-richting wordt er een verandering in schaal gevonden van -0.002% tot +0.004%. In de y-richting wordt er een verandering in schaal gevonden van +0.001% tot +0.007%. Er worden rotaties gevonden tot 0.005°.

3.2.2 Homogene deformatie

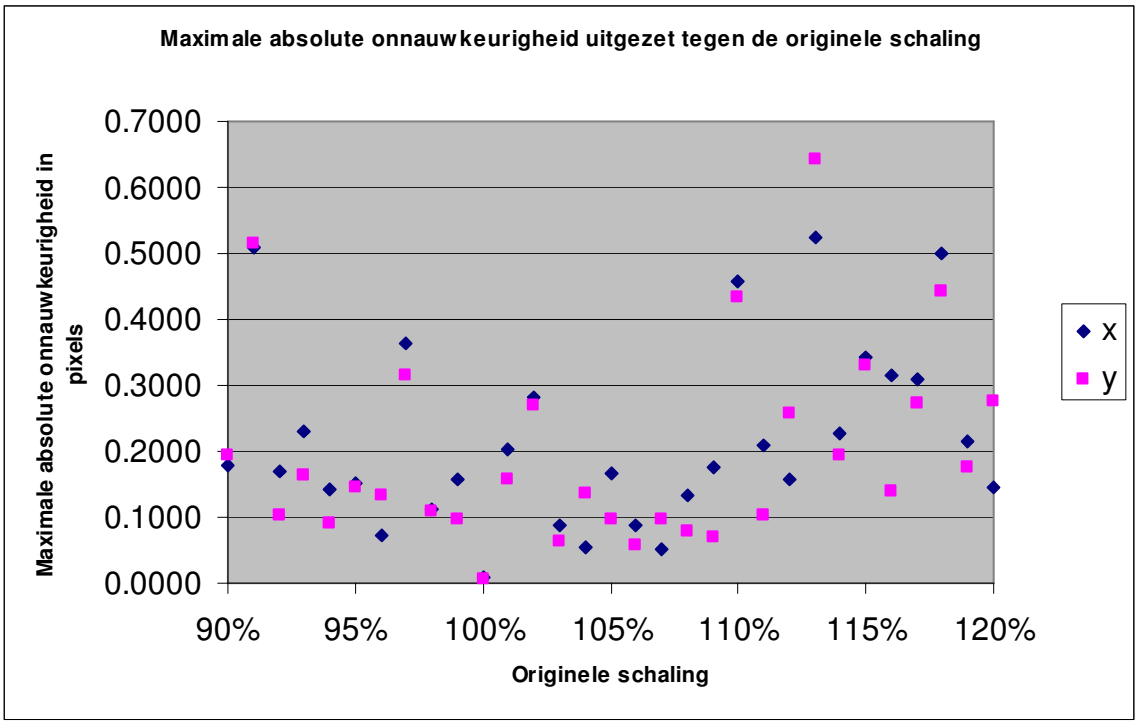
De tweede test is om de afbeelding te vergroten of verkleinen, om ook hierbij de nauwkeurigheid te bepalen. Hierbij is het belangrijk om er rekening mee te houden dat er vervormingen in de afbeelding op kunnen treden bij overgangen tussen materialen. In de gesimuleerde afbeeldingen wordt per pixel bepaald welk materiaal het voorstelt, en dus welke kleur het moet zijn. Bij een verandering in de schaal van de afbeelding zal niet elke rand op elk punt een geheel aantal pixels verschuiven, met als gevolg dat de er lokale afwijkingen met een grootte tot 0,5 pixels op kunnen treden. Een voorbeeld van deze afwijking in opgelegde vervorming is te zien in figuur 7. Deze afwijkingen kunnen invloed hebben op de berekeningen in Elastix en dus op de uiteindelijke onnauwkeurigheid.



Figuur 7: Vervorming van een cirkel bij verandering in straal.

De afbeelding voor het bloedvat met geïdealiseerde geometrie is geschaald met waarden van 90% tot 120%. Vervolgens zijn elk van deze afbeeldingen vergeleken met de oorspronkelijke afbeelding. Hieruit is bepaald hoe groot het verschil is tussen de werkelijke verplaatsingen en de door Elastix bepaalde verplaatsingen. Het resultaat hiervan is te zien in figuur 8.

Er is duidelijk te zien dat met een paar uitzonderingen alle afwijkingen kleiner zijn dan 0,5 pixels. De gemiddelde afwijking door berekende verplaatsing, rotatie en verandering in schaal is 0,21 pixels. De onnauwkeurigheid in alleen de verandering in schaal heeft een gemiddelde waarde van 0,025% die overeenkomt met een waarde tot 0,1 pixels. De grotere onnauwkeurigheid tussen bepalingen in verandering in schaal ten opzichte van bepaling in verplaatsing kan worden verklaard door de lokale afwijkingen bij de overgang van materialen omdat de grijswaarden herberekend moeten worden.



Figuur 8: De maximale absolute onnauwkeurigheid in de bepaling van de verandering in schaal van afbeeldingen. De onnauwkeurigheid is uitgezet tegen de originele verandering in schaal. De onnauwkeurigheid in x- en y-richting zijn apart uitgezet.

3.2.3 Combinatie van translatie en homogene deformatie

Bij de test om zowel translatie als homogene deformatie te combineren komen de resultaten overeen met die van alleen homogene deformatie. De onnauwkeurigheid is voor elke test gemiddeld 0,3 pixels en maximaal 0,7 pixels. De onnauwkeurigheid als gevolg van translatie is niet significant ten opzichte van de onnauwkeurigheid als gevolg van de verandering in schaal.

3.3 Lagere kwaliteit

Vervolgens kan de invloed worden bekeken als dezelfde testen worden herhaald, maar met lagere kwaliteit afbeeldingen. In de praktijk zal de resolutie niet altijd hoog zijn, en er is altijd ruis aanwezig.

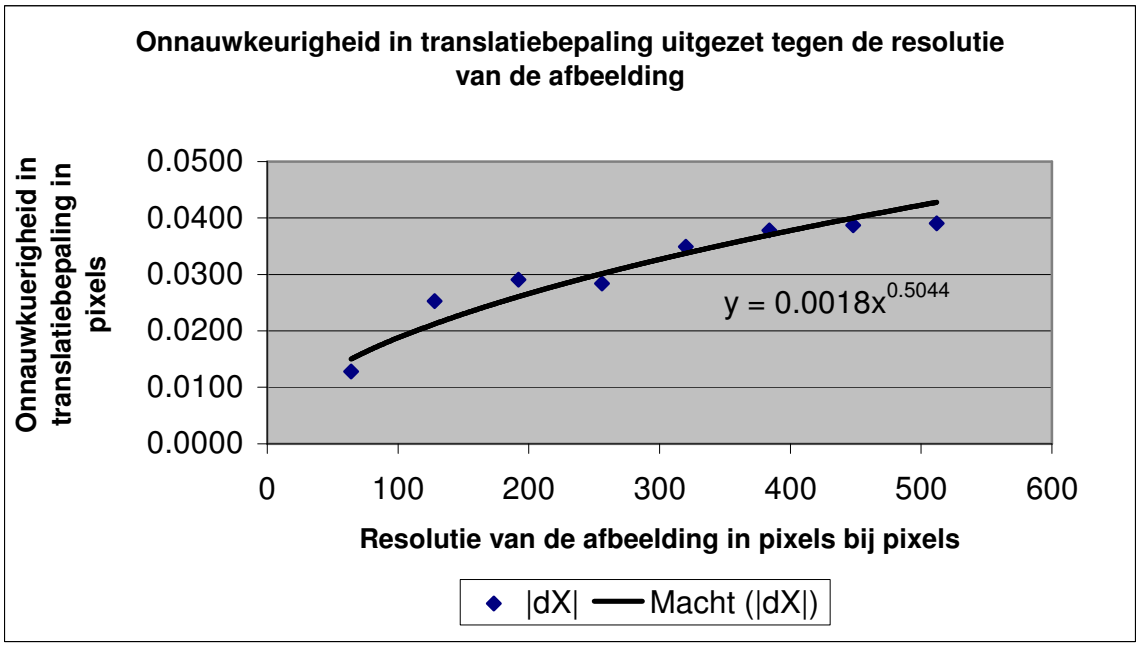
3.3.1 Resolutie

Voor de test met lagere resolutie afbeeldingen wordt gekeken naar de nauwkeurigheid in translatiebepaling. De laagste resolutie is 64 bij 64 pixels, met een verplaatsing van 1 pixel in de x-richting. De hogere resoluties zijn veelvouden hiervan. De relatieve verplaatsing is bij elke afbeelding hetzelfde.

Tabel 3: Onnauwkeurigheid in translatiebepaling bij verschil in resolutie.

Resolutie $R \times R$ [pixel bij pixel]	Werkelijke translatie Δx_r [pixel]	Berekende translatie Δx_c [pixel]	$\frac{\lvert \Delta x_r - \Delta x_c \rvert}{\sqrt{R}}$ [pixel^{½}]
64 x 64	1,0000	1,0128	0,00160
128 x 128	2,0000	2,0253	0,00224
192 x 192	3,0000	2,9709	0,00210
256 x 256	4,0000	4,0284	0,00178
320 x 320	5,0000	5,0349	0,00195
384 x 384	6,0000	6,0378	0,00193
448 x 448	7,0000	6,9613	0,00183
512 x 512	8,0000	7,9610	0,00172

De resultaten hierbij staan in bovenstaande tabel en in onderstaand figuur. Uit de berekende waarden kan worden afgeleid dat de onnauwkeurigheid in translatiebepaling omgekeerd evenredig is met de wortel van de resolutie[7].



Figuur 9: Onnauwkeurigheid in translatiebepaling uitgezet tegen de resolutie van de afbeelding.

3.3.2 Ruis

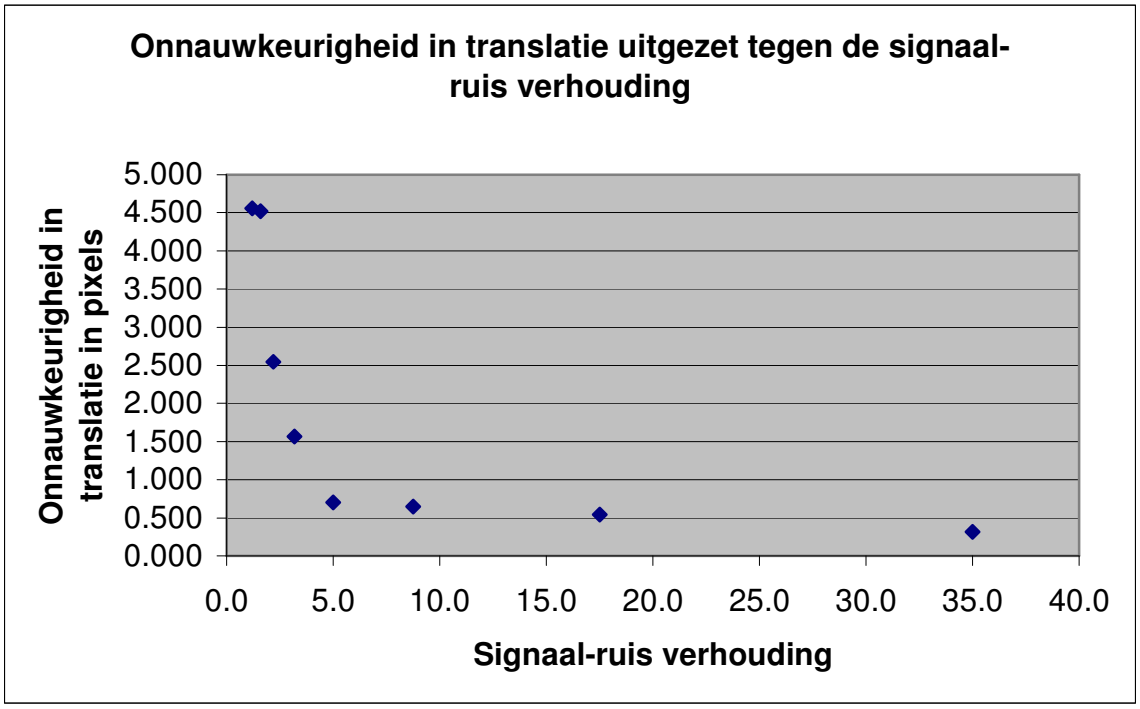
Voor de test met ruis is de afbeelding met hoge resolutie gemaakt, en verschoven over 5 pixels. Zonder ruis heeft elke pixel een grijswaarde tussen de 0 en 1, waarbij de gemiddelde grijswaarde 0,7 is. De ruis verandert de grijswaarde per pixel bij een willekeurige waarde tot het ruisniveau.

Tabel 4: Onnauwkeurigheid in translatiebepaling bij verschil in ruisniveau.

Ruisniveau [-]	S/N [-]	Onnauwkeurigheid in de berekende translatie ΔX [pixels]	Onnauwkeurigheid in de berekende translatie ΔY [pixels]
0,020	35	-0,224	0,229
0,040	18	0.387	-0,382
0,080	8,8	0,457	0,460
0,140	5,0	-0,501	0,492
0,220	3,2	-1,489	0,489
0,320	2,2	-2.492	0,514
0,440	1,6	-4,490	0,497
0,580	1,2	-4,529	0,507

Uit de gegevens van bovenstaande tabel is op te maken dat ruis een grote invloed heeft op de nauwkeurigheid van de berekeningen. Een kleine hoeveelheid ruis geeft een onnauwkeurigheid tot een halve pixel. Zoals in figuur 10 te zien is, gaat bij een signaalruisverhouding kleiner dan 5 de nauwkeurigheid sterk omlaag. Bij een signaalruisverhouding kleiner dan 2 wordt de translatie niet meer gevonden en blijft alleen de onnauwkeurigheid die ook de y-richting te zien is over.

Wel valt op dat de onnauwkeurigheid hierbij een afwijking in dezelfde richting geeft bij vrijwel elk ruisniveau. Dit duidt op een systematische onnauwkeurigheid, zoals te verwachten is bij berekeningen met een theoretisch model als uitgangspunt.



Figuur 10: Onnauwkeurigheid in berekende translatie uitgezet tegen de signaal-ruis verhouding bij een translatie van 5 pixels.

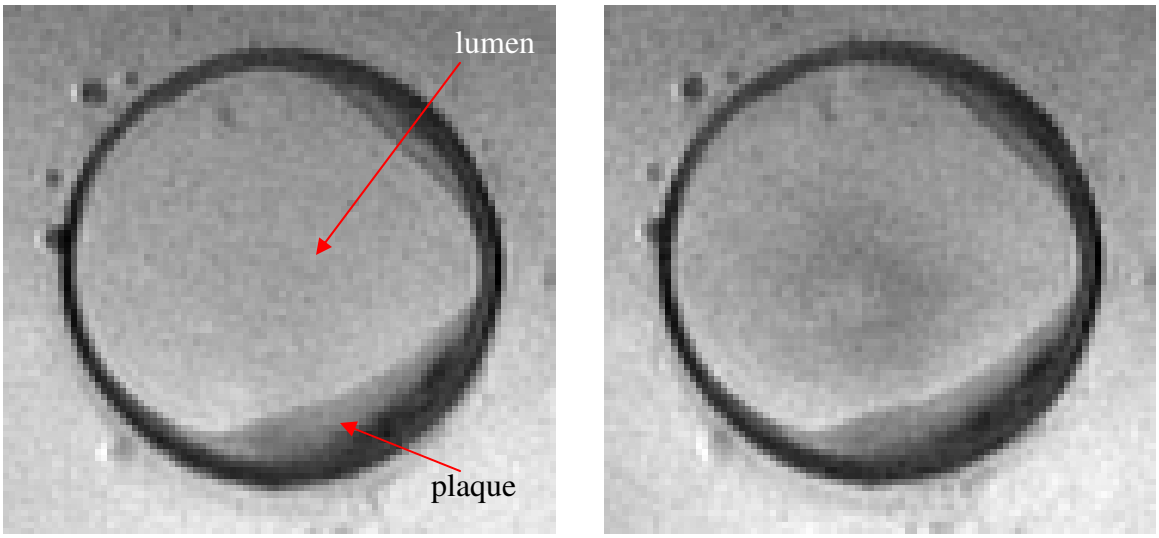
4 MRI

Bij een eerder project zijn MRI-afbeeldingen gemaakt van een atherosclerotisch bloedvat van een varken. Deze afbeeldingen zijn gemaakt bij een interne druk van 10,7 kPa (80 mmHg), 13,3 kPa (100 mmHg) en 16,0 kPa (120 mmHg) hoger dan de externe druk. Deze waarden komen overeen met de standaard bloeddruk.

4.1 Originele afbeeldingen

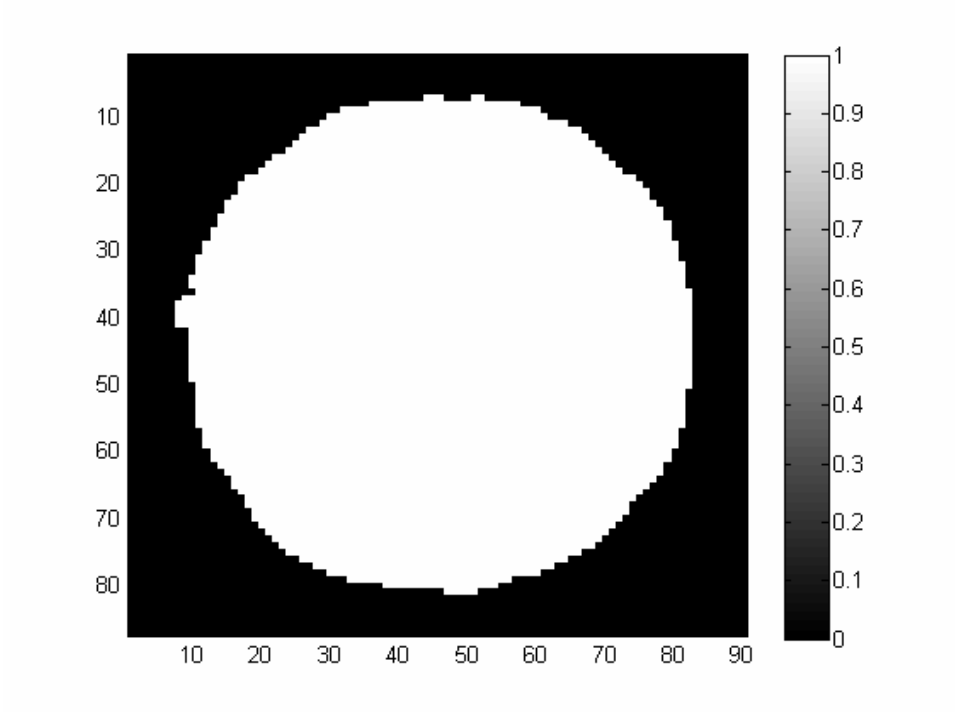
Na de testen met geïdealiseerde geometrie kan bepaald worden of deze resultaten ook bij MRI afbeeldingen behaald kunnen worden. Deze afbeeldingen hebben een lage resolutie. De MRI afbeeldingen die hierbij worden gebruikt hebben een resolutie van ongeveer 80 bij 80 pixels voor het deel van de afbeelding met het bloedvat, waarbij 1 pixel overeen komt met 48,8 µm. Ook bevatten deze afbeeldingen ruis en artefacten. Dit kan een grote invloed hebben op de berekende deformatievelden.

Als eerste test hierbij worden twee MRI afbeeldingen gebruikt van een gedeelte van het bloedvat met weinig plaque. De afbeeldingen zijn te zien in figuur 11. Bij deze test wordt gebruik gemaakt van een masker zodat er geen afwijkingen zijn als gevolg van het omringende materiaal. Dit masker is een binaire afbeelding die aangeeft welke pixels gebruikt mogen worden voor de berekeningen, en is te zien in figuur 12.



Figuur 11: MRI afbeeldingen van een bloedvat met weinig plaque, met een druk van 10,7 kPa (links) en 16,0 kPa (rechts)

Er zijn enkele luchtbellen te zien buiten het bloedvat. Deze zijn een gevolg van de opstelling waarbij de afbeeldingen zijn gemaakt. Er is geen zichtbaar verschil tussen de media en adventitia. Het lumen en intima zijn wel goed zichtbaar.

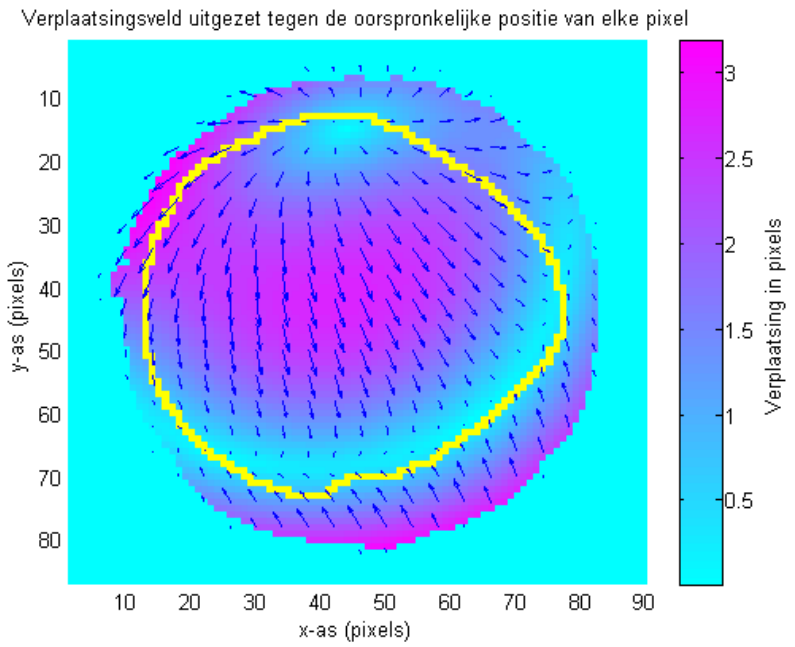


Figuur 12: Een masker voor gebruik in Elastix. Alleen de pixels die op dezelfde locatie liggen als de witte pixels in het masker worden gebruikt voor de deformatiebepaling.

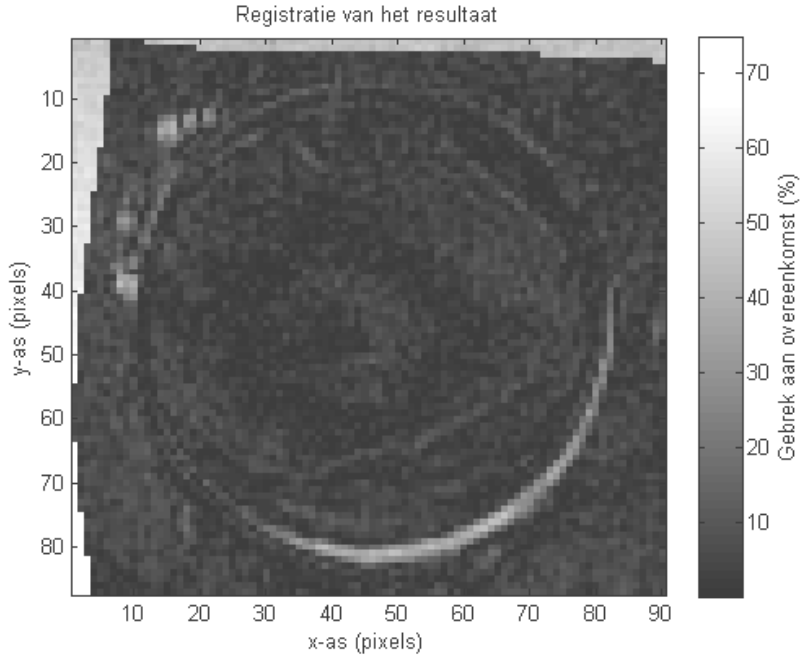
Figuur 13 is het berekende verplaatsingsveld bij de twee MRI afbeeldingen. De meeste pijlen wijzen naar de buitenkant van de afbeelding. Deze pijlen geven de richting van de verplaatsing aan; het lijkt er dus op dat het lumen groter is geworden.

In figuur 14 is de registratie van dit resultaat te zien. De lichtere pixels rechts onderin de afbeeldingen geven aan dat de registratie daar niet goed is gelukt. De lichtere pixels links en bovenin liggen buiten het gebruikte masker. Hier is de deformatie niet berekend, dus is te verwachten dat deze niet klopt.

Door figuur 13 en 14 te vergelijken is te zien dat de verplaatsing van de intima in het verplaatsingsveld niet klopt. En om de vergroting van het lumen te bepalen moet alleen op de verplaatsing van de randen van het lumen worden gelet. De verplaatsingen binen het lumen hebben een grootte van minder dan 3 pixels, en hebben daarmee geen invloed op de grootte van het lumen. Het lumen lijkt naar boven te zijn uitgezet, maar met een verplaatsing van minder dan 0,3 pixels is dit niet significant.



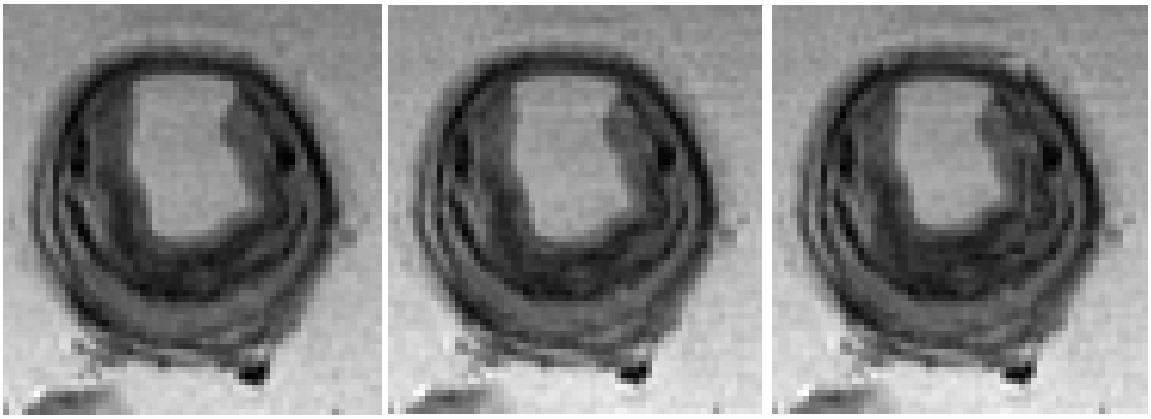
Figuur 13: Verplaatsingsveld uitgezet tegen de oorspronkelijke positie van elke pixel. De kleur van de achtergrond en de lengte van de pijlen geven de grootte van de verplaatsing aan. De richting van de pijlen geeft de richting van de verplaatsing aan. Deze verplaatsing geeft aan hoe de eerste afbeelding moet worden bewerkt om overeen te komen met de tweede afbeelding. De gele rand geeft de contour van het lumen aan.



Figuur 14: Registratie van het resultaat. De registratie is de resultaatsafbeelding met de eerste (fixed) afbeelding er vanaf getrokken. Dit laat zien hoe goed de registratie is gelukt. De lichtere kromme rechts onderin de afbeelding geeft aan dat de registratie bij de rand van de media niet goed is gelukt.

4.2 Handmatige deformatie

Bij de MRI afbeeldingen zijn de zichtbare deformaties zeer klein. Om een beter beeld te krijgen van de kwaliteit van het deformatieveld is de test herhaald, maar met een handmatig aangebrachte grotere deformatie in de tweede afbeelding. Hierbij is voor een deel van het bloedvat met veel plaque gekozen.

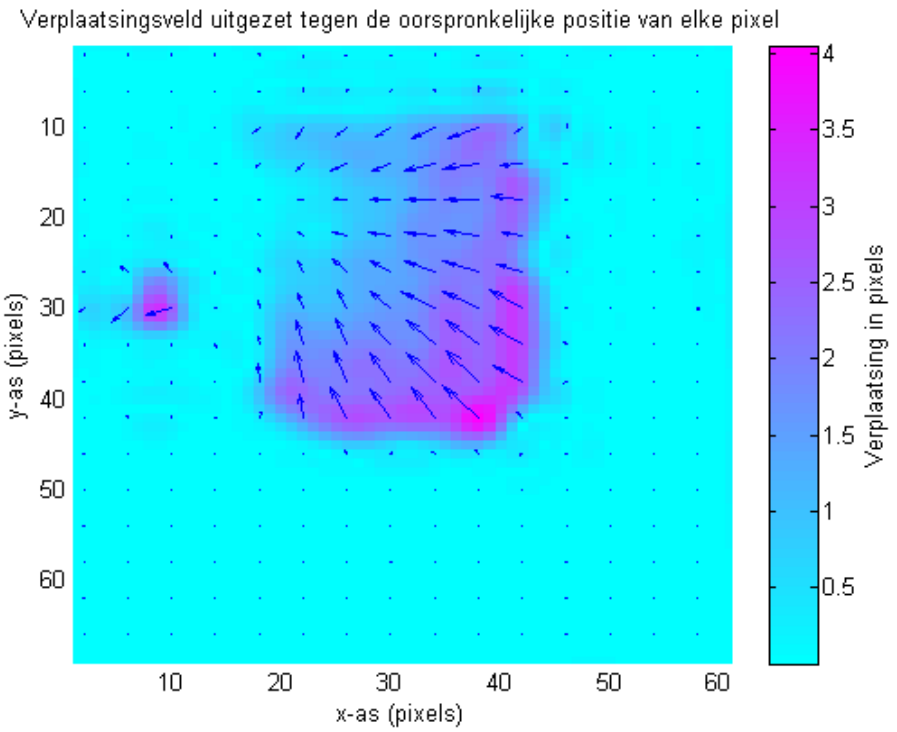


Figuur 15: MRI afbeeldingen van een bloedvat met plaque. Druk binnen het bloedvat van 13,3 kPa (links), 16,0 kPa (midden) en 16,0 kPa met het lumen 10% verkleind (rechts).

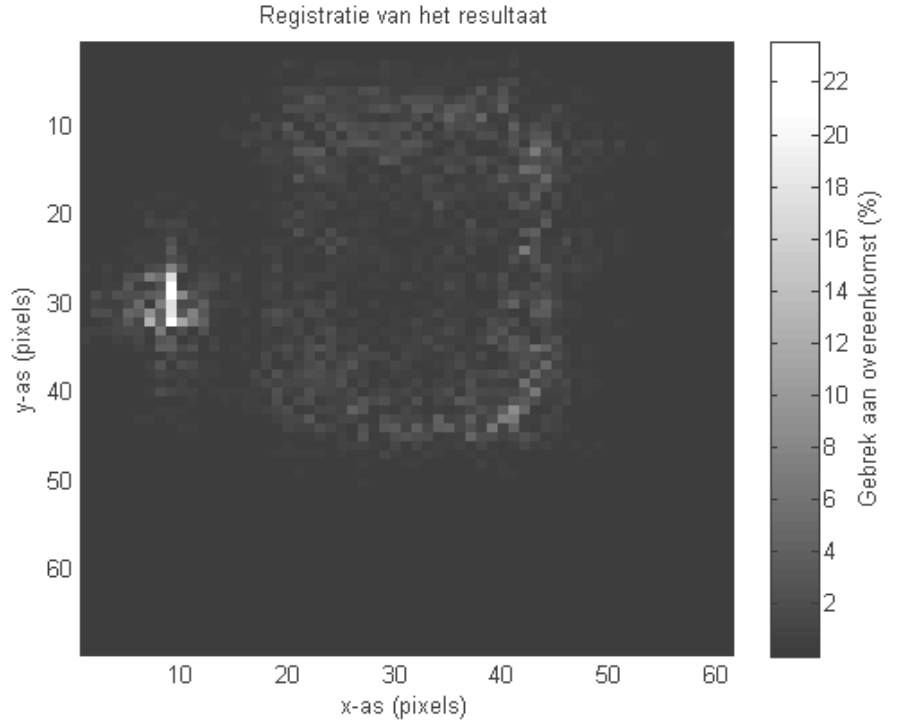
Bij deze afbeeldingen is het lumen te zien als het lichtere gedeelte binnen het bloedvat. De lipide kern is het donkere gedeelte onder het lumen. De eerste twee afbeeldingen zijn gemaakt bij een druk van 13,3 kPa (100 mmHg) en 16,0 kPa (120 mmHg). Voor de derde afbeelding zijn de afmetingen van het lumen 10% kleiner gemaakt. De lege pixels aan de rand van het lumen zijn opgevuld met een kopie van de omringende pixels.

Er is geen verschil te zien in de grootte van het lumen tussen de eerste en tweede afbeelding. Ook het berekende verplaatsingveld laat geen significant verschil zien bij de rand van het lumen.

Er wordt gekeken naar het verschil tussen de tweede en derde afbeelding. Aangezien hierbij alleen een rechthoekig gedeelte, met hierin het lumen, verschilt tussen de afbeeldingen, wordt ook alleen daar verplaatsing verwacht. In figuur 16 is dit duidelijk zichtbaar. Wel is ook een ongewenste deformatie verder naar links zichtbaar. Deze verplaatsing is ook zichtbaar in de registratie in figuur 17.



Figuur 16: Verplaatsingsveld tussen de tweede en derde afbeelding uit figuur 15.

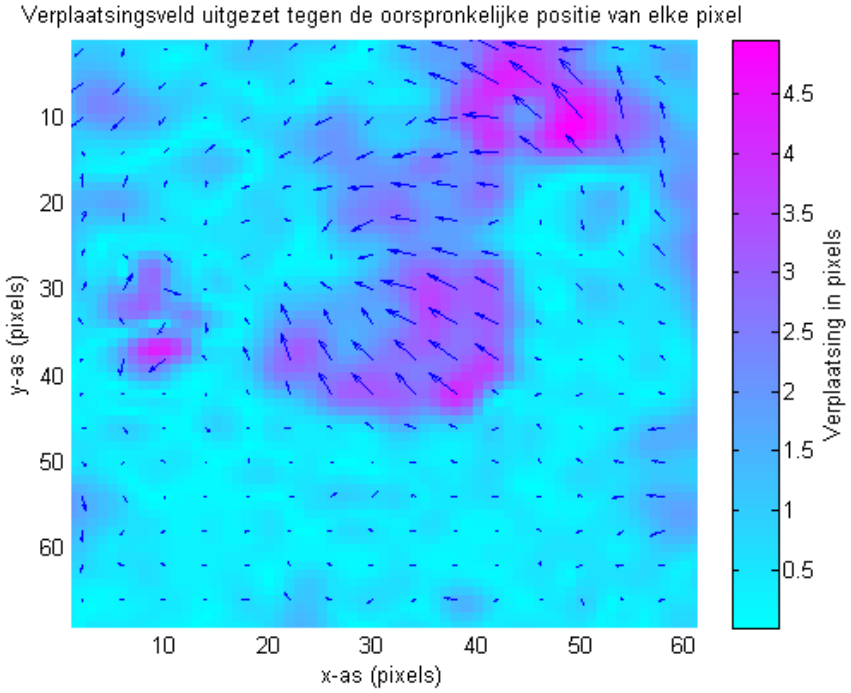


Figuur 17: Registratie van het resultaat tussen de tweede en derde afbeelding uit figuur 15.

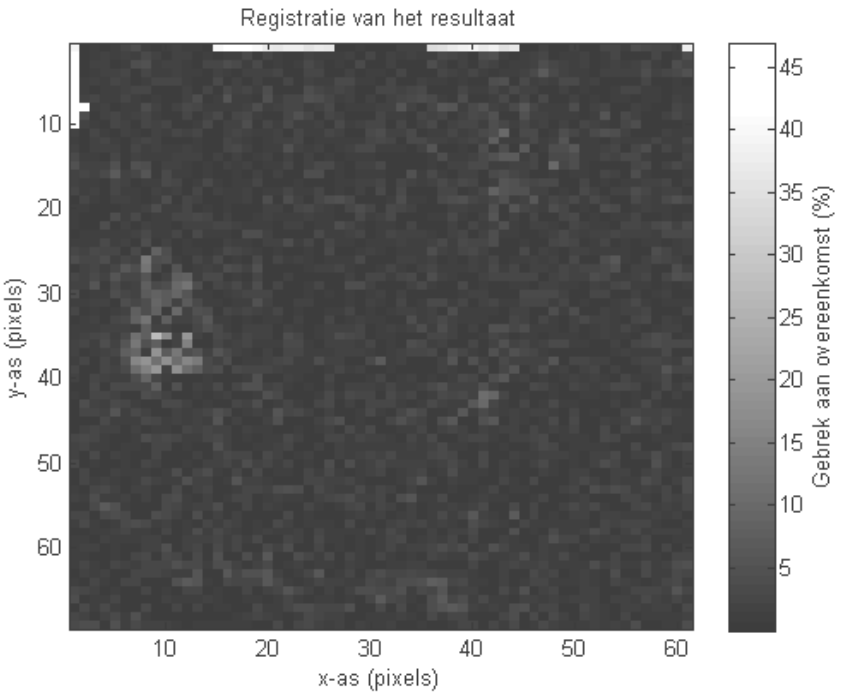
Aangezien de handmatig aangebrachte deformatie goed te zien, is lijken er geen problemen op te treden als gevolg van het contrast in de afbeeldingen of de lage resolutie. Vervolgens kan dus de deformatie worden bekeken tussen de eerste en derde afbeelding uit figuur 15. Dit is dus een vergelijking tussen twee verschillende MRI-afbeeldingen, waarbij voor één van de afbeeldingen een extra deformatie is aangebracht.

Dit resultaat is te zien in figuur 18 en 19. De handmatig aangebrachte deformatie is duidelijk zichtbaar in het verplaatsingsveld. Er is een grotere verplaatsing zichtbaar rechtsboven. Dit is een gevolg van de handmatig aangebrachte deformatie, en is zichtbaar als een lichtere driehoek in de derde afbeelding in figuur 15.

Hieruit blijkt dat ook ruis en kleine artefacten geen groot effect hebben op het verplaatsingsveld.



Figuur 18: Verplaatsing tussen de eerste en derde afbeelding uit figuur 15.



Figuur 19: Registratie van het resultaat tussen de eerste en derde afbeelding uit figuur 15.

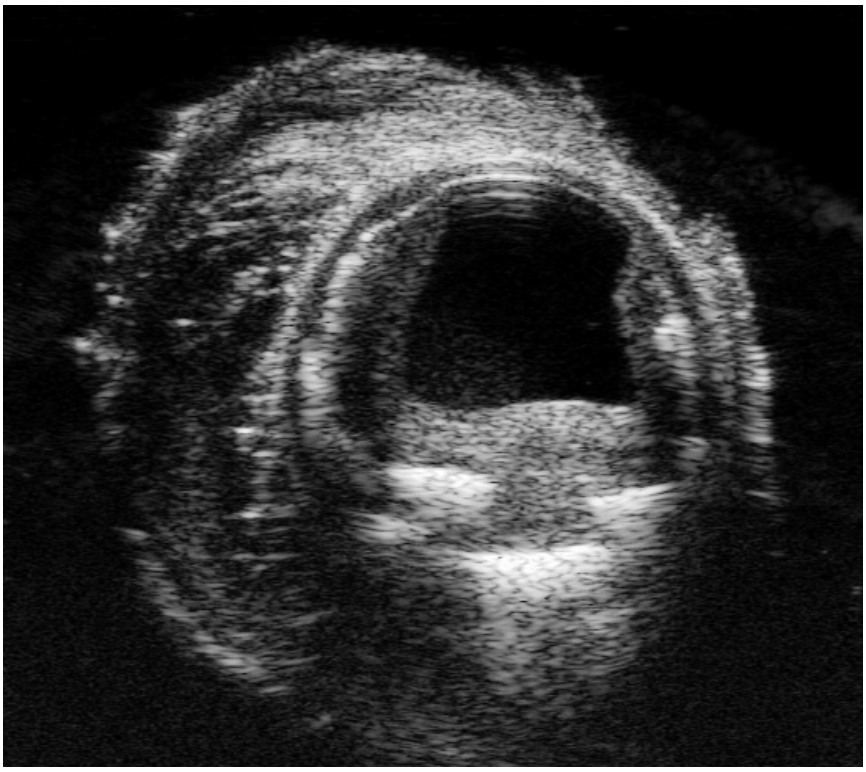
Uit deze resultaten kan geconcludeerd worden dat de gemaakte MRI-afbeeldingen niet geschikt zijn om verplaatsingsvelden te berekenen. Er zijn geen zichtbare vervormingen bij een verandering in druk. Aangezien dit wel wordt verwacht, lijkt het er op dat de interne druk bij de MRI-experimenten niet klopt met de ingestelde waarden.

5 Ultrageluid

Van dezelfde bloedvaten van het varken waar MRI afbeeldingen zijn gemaakt, zijn ook ultrageluidafbeeldingen gemaakt. De techniek die hierbij wordt gebruikt heet echografie. Ultrasoon geluid wordt door het bloedvat gestuurd. Bij overgangen tussen verschillende materialen wordt een deel van het geluid gereflecteerd. Ook is de geluidssnelheid afhankelijk van de dichtheid van het materiaal. Uit de snelheid van het gereflecteerde geluid en de tijd tussen het uitzenden en opvangen van het geluid kan een beeld worden gevormd van de materialen in een bloedvat.

5.1 Ultrageluidafbeeldingen

De gebruikte ultrageluidafbeeldingen hebben een hogere resolutie dan de MRI afbeeldingen. Deze varieert van 17 tot 26 µm per pixel. Het nadeel van ultrageluid is dat plaque niet goed te onderscheiden is van gezond weefsel. Ook kan er geen informatie worden verkregen over materialen die achter een hard materiaal zoals calcium zit.



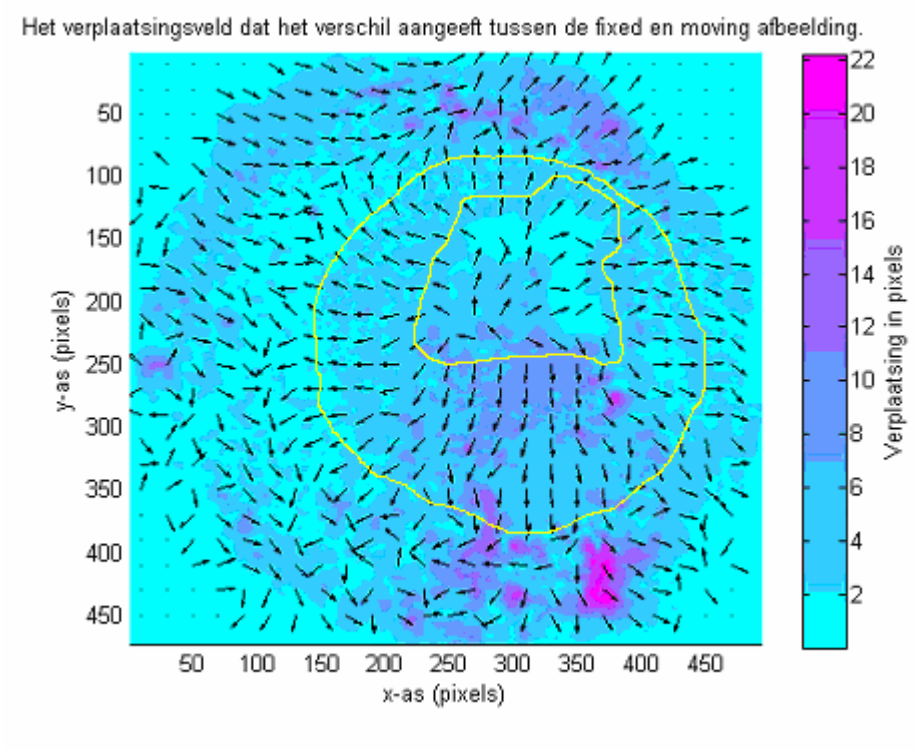
Figuur 20: Ultrageluidafbeelding van een bloedvat bij een interne druk van 10,7 kPa.

De bovenstaande afbeelding is van een doorsnede van een bloedvat. Het lumen is in deze afbeelding goed zichtbaar als het donkere gedeelte in het midden van de afbeelding. De intima, media en adventitia zijn zichtbaar als een respectievelijk donkere, lichte en donkere rand. Buiten de adventitia is loshangend weefsel. De drie lichtere plekken onder het lumen zijn locaties met calcium.

5.2 Resultaten

Voor de eerste test met ultrageluidafbeeldingen zijn twee afbeeldingen op dezelfde locatie genomen. Deze afbeeldingen zijn bij een interne druk van 10,7 kPa (80 mmHg) en 13,3 kPa (100 mmHg). De afbeeldingen zijn te zien in bijlage 3.

Het berekende verplaatsingsveld is te zien in figuur 18. Hierin is duidelijk te zien dat het lumen groter wordt, en dat alles om het lumen heen naar buiten beweegt bij de toename in druk binnen het lumen.



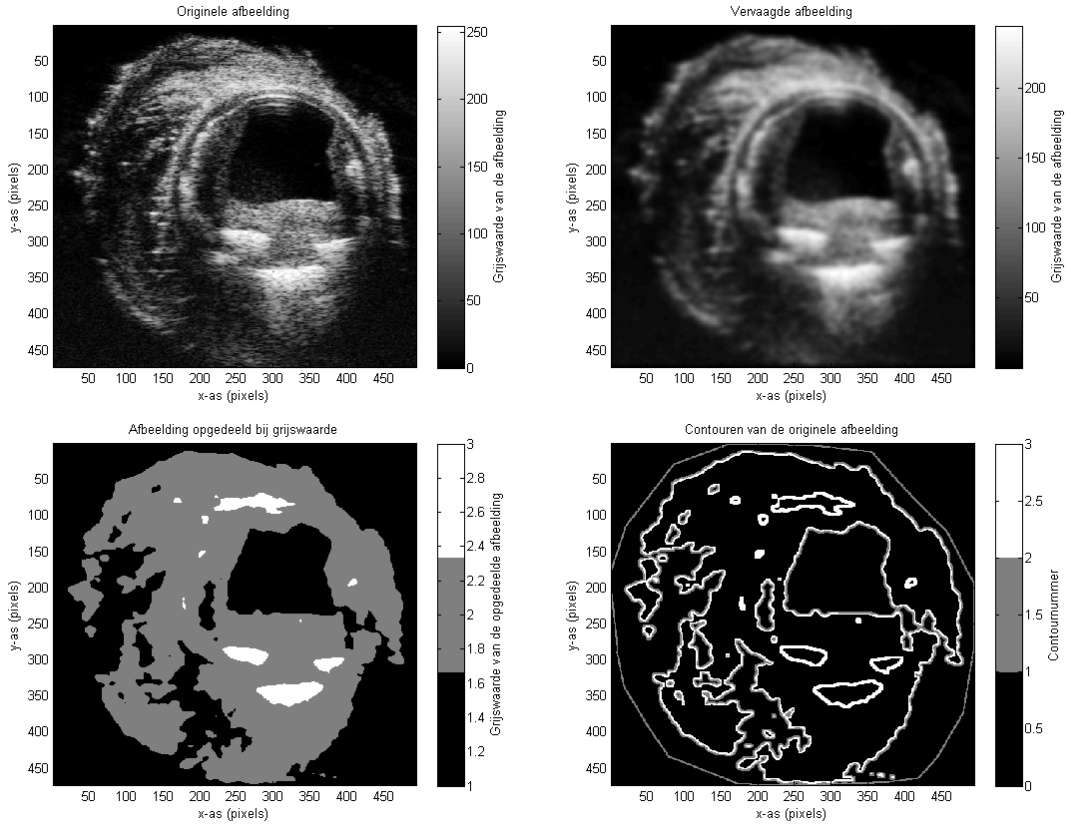
Figuur 21: Het verplaatsingsveld tussen twee ultrageluidafbeeldingen. De kleur geeft de grootte van de verplaatsing aan, en de richting van de pijlen geeft de richting van de verplaatsing aan. De gele randen zijn handmatig getekende contouren van het lumen en het adventitia.

Dezelfde berekeningen zijn ook gedaan voor de verplaatsingen bij een druk van 2,7 kPa naar 5,3 kPa, van 5,3 kPa naar 8,0 kPa en van 8,0 kPa naar 10,7 kPa. De verplaatsingsvelden hierbij staan in bijlage 4.

5.2.1 Automatische contouren

Om bij elk van de ultrageluidafbeeldingen de grootte van het lumen te bepalen is een script geschreven in Matlab®. Dit is onderdeel van het script dat de resultaten van Elastix kan lezen en visualiseren en staat in bijlage 7. Dit deel van het script vervaagt de afbeelding om de effecten van ruis tegen te gaan. Vervolgens wordt de afbeelding opgedeeld gebaseerd op de grijswaarden van de pixels. De grenswaarden hierbij worden door de gebruiker opgegeven. Vervolgens kunnen de overgangen hierbij worden bepaald,

met de contouren van de afbeelding als resultaat. In onderstaand figuur is dit proces zichtbaar gemaakt.



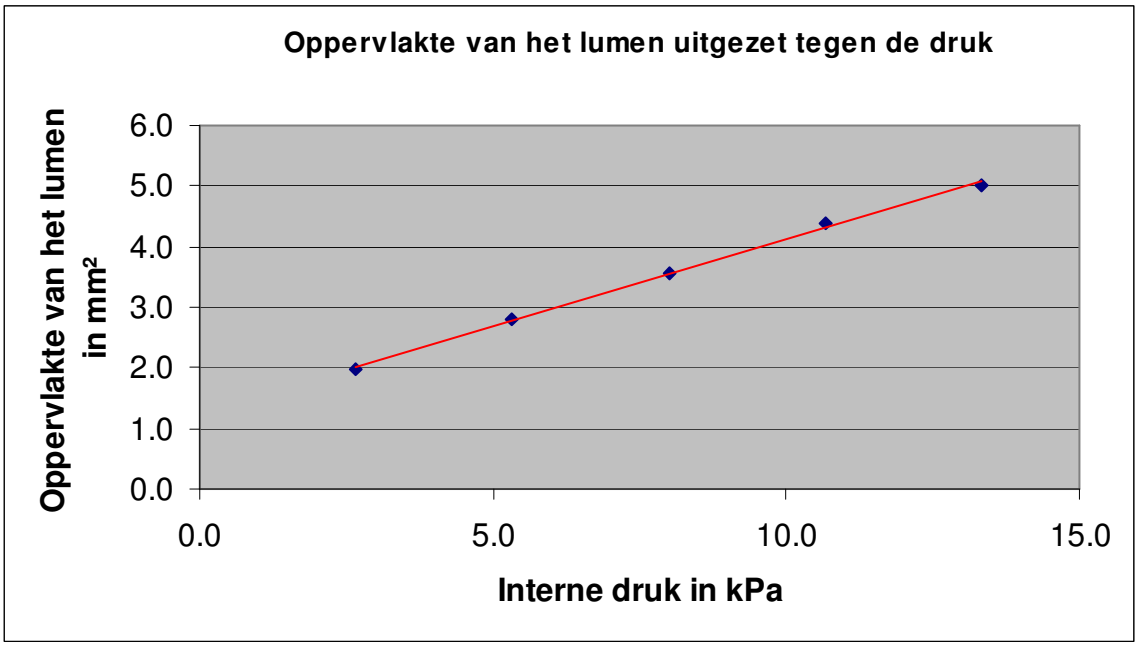
Figuur 22: Het proces om contouren van een afbeelding te bepalen. Linksboven is de originele afbeelding. Rechtsboven is de vervaagde afbeelding. Linksonder is de afbeelding opgedeeld bij grijswaarde, met de grenswaarden op 30 en 180. Rechtsonder zijn de contouren van de afbeelding.

Bij elk van de ultrageluidsafbeeldingen zijn de contouren bepaald. Voor elke afbeelding zijn dezelfde hoeveelheid vervaging en dezelfde grenswaarden gebruikt. Het is belangrijk om consistent te zijn bij deze bepaling, zodat de relatieve onnauwkeurigheid voor elke afbeelding gelijk is.

5.2.2 Bruikbaarheid van de ultrageluidafbeeldingen

Met deze contouren kan voor elke afbeelding de oppervlakte van de doorsnede van het lumen worden bepaald. Het aantal pixels binnen de contour rond het lumen kan worden bepaald. Bekend is dat een lengte van 511 pixels overeen komt met 9 mm. Hiermee kan de oppervlakte van het lumen in mm^2 worden berekend. Deze is uitgezet tegen de druk binnen het lumen in figuur 23. Er is duidelijk te zien dat het oppervlakte recht evenredig is met de druk.

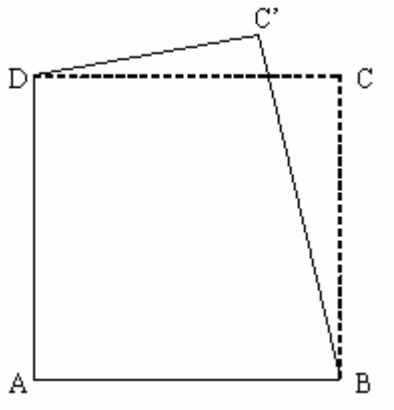
Ook de verplaatsing van de drie plekken met calcium is consistent. Uit de ultrageluidafbeeldingen kunnen de verplaatsingsvelden dus goed worden bepaald.



Figuur 23: Oppervlakte van de doorsnede van het lumen uitgezet tegen de interne druk.

5.3 Deformatiebepaling

Bij de eindige-elementenmethode, welke gevalideerd moet worden met behulp van de meetgegevens, wordt de principiële rek berekend. Dit zijn waarden die aangeven hoe elke pixel wordt uitgerekt, niet hoe elke pixel verplaatst. Het is dus nodig om de berekende verplaatsingsvelden om te zetten naar deformatievelden.



Figuur 24: Schematische voorstelling van de locaties van 4 pixels. A, B, C en D geven de locatie aan van de vier pixels voor de deformatie. C' geeft de locatie aan van de pixel C na de deformatie.

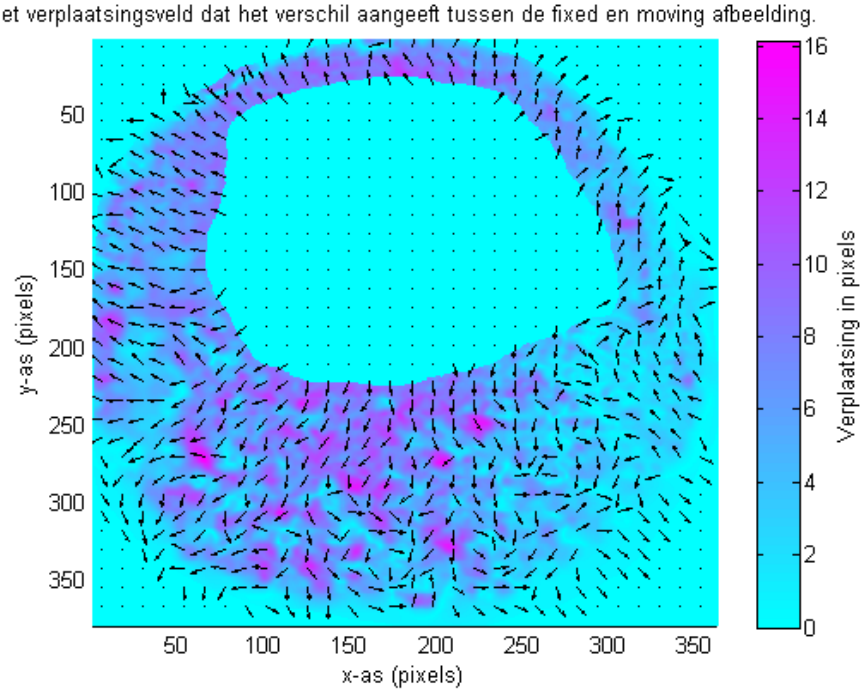
Stel A, B, C en D in bovenstaand figuur zijn de middenpunten van 4 pixels en vormen een vierkant met zijden van lengte 1. Als een pixel wordt verplaatst verandert de afstand

tussen de pixels, en daarmee vervormt de afbeelding waar de pixels een onderdeel van zijn. Dit is weergegeven in bovenstaande afbeelding, waarbij pixel C verplaatst naar C'.

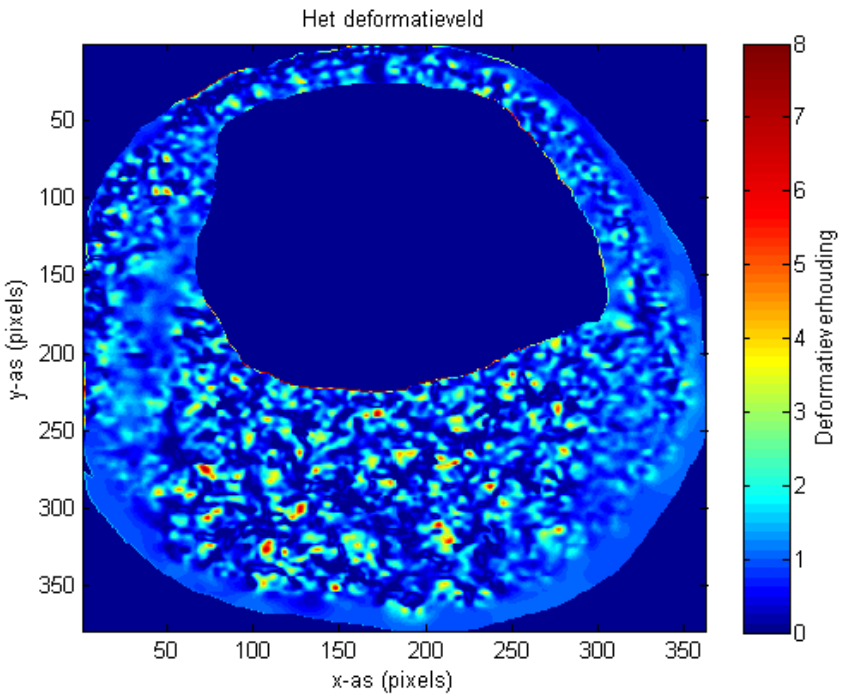
De deformatiematrix die de vervorming van het gebied tussen de middelpunten van deze pixels aangeeft kan als volgt worden berekend:

$$F = \begin{bmatrix} 1 + x_{C'} - x_D & x_{C'} - x_B \\ y_{C'} - y_D & 1 + y_{C'} - y_B \end{bmatrix}$$

De rechtse Cauchy-Green deformatie tensor[8] is gedefinieerd als $C = F \cdot F^T$. Deze geeft het kwadraat van lokale veranderingen in afstand als gevolg van deformatie aan. De twee eigenwaarden van deze tensor geven het kwadraat van de minimale en maximale principiële rek aan. Als de verandering in positie van de pixels bekend is, kan de principiële rek worden berekend door de wortel te nemen van de eigenwaarden van de deformatie tensor. De kleinste positieve waarde is de minimale principiële rek, en de grootste positieve waarde is de maximale principiële rek. Door de minimale principiële rek te vermenigvuldigen met de maximale principiële rek, kan de deformatie worden berekend. Een waarde van 1 geeft hierbij aan dat er geen deformatie is. Waarden kleiner dan 1 geven een verkleining aan en waarden groter dan 1 geven een vergroting aan.



Figuur 25: Het verplaatsingsveld dat het verschil aangeeft tussen ultrageluidafbeeldingen gemaakt bij 5,3 kPa en 10,7 kPa.



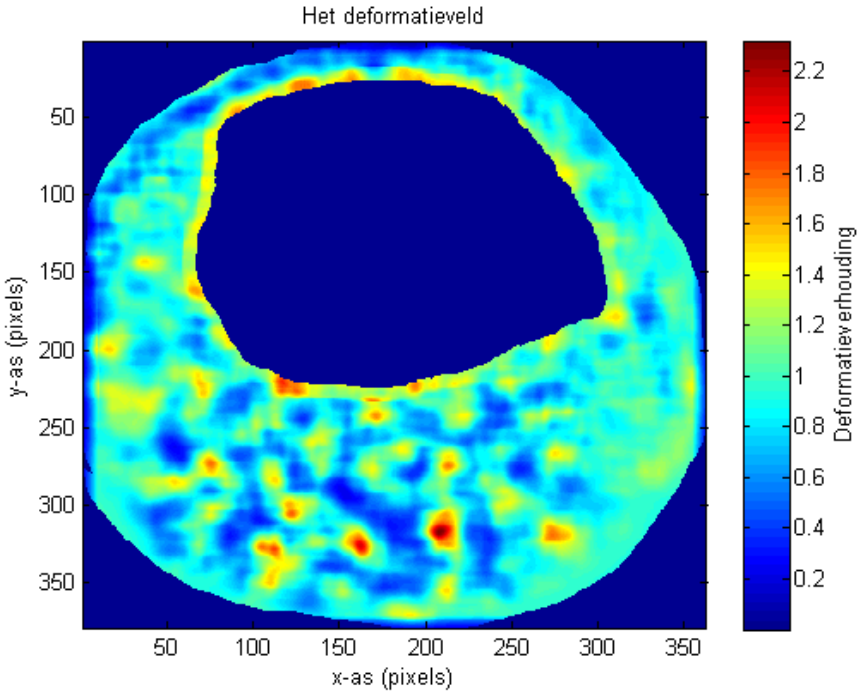
Figuur 26: Het deformatieveld berekend uit het verplaatsingsveld van ultrageluidafbeeldingen bij 5,3 kPa en 10,7 kPa.

Figuur 25 is het verplaatsingsveld dat is berekend aan de hand van twee ultrageluidafbeeldingen. Met de gegevens van dit verplaatsingsveld is het deformatieveld in afbeelding 26 berekend. De effecten van ruis zijn hierin duidelijk zichtbaar. Om de deformaties zichtbaar te maken is het nodig om de effecten van de ruis te verminderen.

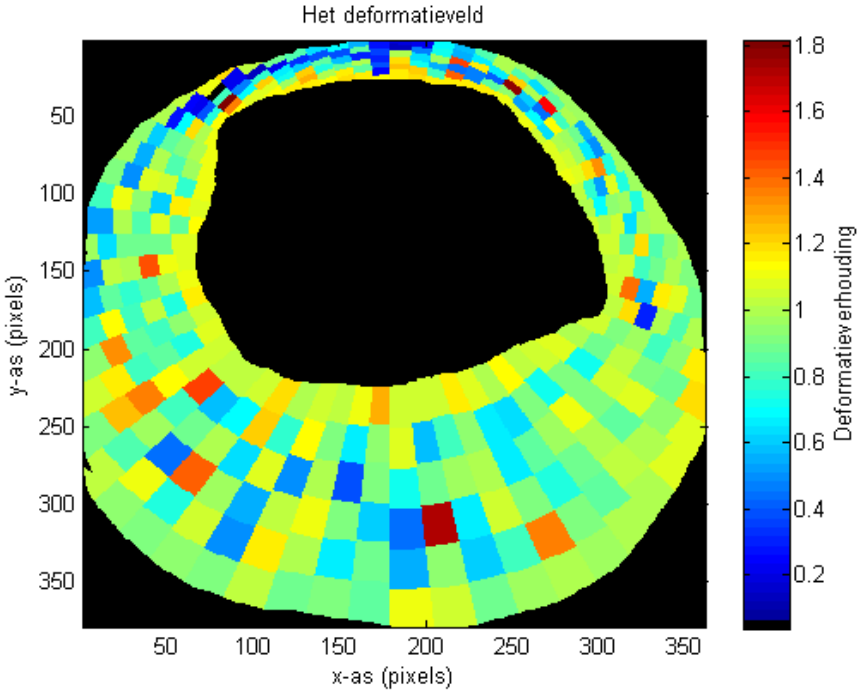
Bij afbeelding 27 is een filter gebruikt om de ruis te verminderen. Bij dit filter wordt voor elk punt het gemiddelde genomen van de omringende punten. Aangezien ruis stochastisch is, terwijl de deformaties afhangen van de materiaaleigenschappen en de krachten die hierop werken, is dit een effectief filter tegen ruis zonder de globale vorm van de informatie te veranderen.

Bij afbeelding 28 zijn de gegevens opgedeeld in segmenten. Elk segment geeft de gemiddelde waarde weer van de gegevens die binnen dit segment vallen.

In beide afbeeldingen is zichtbaar dat de deformatie rond het lumen groter is dan 1. Dit betekent dat de materialen hier worden uitgerekt als de druk binnen het lumen groter wordt. In de plaque onder het lumen is de gemiddelde deformatie kleiner dan 1. Hier worden de materialen samengedrukt.



Figuur 27: Het deformatieveld berekend uit het verplaatsingsveld van ultrageluidafbeeldingen bij 5,3 kPa en 10,7 kPa. Bij deze afbeelding is een filter toegepast om de ruis te verminderen.



Figuur 28: Het deformatieveld berekend uit het verplaatsingsveld van ultrageluidafbeeldingen bij 5,3 kPa en 10,7 kPa. Bij deze afbeeldingen zijn de gegevens opgedeeld in segmenten, waarbij elk segment de gemiddelde waarde weergeeft van de gegevens binnen het segment.

6 Onnauwkeurigheidsanalyse

De nauwkeurigheid van de deformatievelden hangen af van verschillende factoren.

Elastix maakt enkele veronderstellingen om het verplaatsingsveld te berekenen. Welke veronderstellingen dit zijn hangt af van de gebruikte instellingen, maar elk zorgt voor een systematische onnauwkeurigheid in het resultaat. Ook de kwaliteit van de gebruikte afbeeldingen, vooral de hoeveelheid ruis en artefacten, heeft een effect op de nauwkeurigheid.

6.1 Afbeeldingkwaliteit

Zoals is bepaald met afbeeldingen met een geïdealiseerde geometrie, is de berekende verplaatsing niet exact. Elastix bepaalt de verplaatsing door vele verplaatsingen uit te proberen, en geeft het resultaat met de beste overeenkomst als uitvoer. Door meer verplaatsingen uit te proberen, kan de nauwkeurigheid worden vergroot, met als nadeel dat de benodigde rekentijd evenredig omhoog gaat.

Bij afbeeldingen met een hogere resolutie is de nauwkeurigheid groter dan bij afbeeldingen met een lagere resolutie. Deze systematische onnauwkeurigheid is een gevolg van de ImagePyramid methode. Bij lage resolutie afbeeldingen gaat meer informatie verloren als de resolutie omlaag wordt gehaald dan bij hoge resolutie afbeeldingen. Hierdoor kan de verplaatsing minder nauwkeurig worden bepaald.

Ruis heeft een grote invloed op het berekende verplaatsingsveld. In hoofdstuk 3.3.2 is aangetoond dat Elastix een signaal-ruis verhouding van ten minste 5:1 nodig heeft om verplaatsingen te kunnen berekenen.

Ook artefacten in de afbeeldingen zorgen voor onnauwkeurigheid. Shadowing, een MRI-artefact waarvan een voorbeeld te zien is in figuur 39 in bijlage 5, is een artefact waarbij de afbeelding meerdere keren te zien is, met een kleine verschuiving tussen de beelden. Microscopisch kleine luchtbellen in het materiaal waarbij MRI of ultrageluidafbeeldingen zijn gemaakt geven een verstoring in de afbeelding die vele malen groter is dan de luchtbel. Een voorbeeld hiervan is te zien in figuur 40 in bijlage 5. Harde materialen zoals calcium verstoren ultrageluidafbeeldingen. Deze weerkaatsen als het geluid, waardoor er geen beeld kan worden gevormd van de materialen achter het harde materiaal. Elk van deze artefacten verstoort de originele afbeelding, waardoor er op de locaties met een verstoring het verplaatsingsveld niet kan worden berekend.

6.2 Artefacten door Elastix

Verder zijn er enkele artefacten die door Elastix worden veroorzaakt.

Als er meerdere pixels bij elkaar dezelfde grijswaarde hebben, zoals in het lumen, kan Elastix niet zien of er hierbij verplaatsing is. Elke poging om de verplaatsing te bepalen zal hetzelfde resultaat hebben. Elastix geeft hierbij een willekeurige verplaatsing als resultaat, welke dus niet altijd klopt. Dit artefact kan worden vermeden door gebruik te maken van een masker. Door het gebied met de pixels met dezelfde grijswaarde met het masker weg te halen zal dit gebied niet in de berekeningen mee worden genomen, en zal in plaats daarvan dezelfde verplaatsing als de pixels net rond dit gebied worden gebruikt.

Bij de bepaling van de verplaatsing van de pixels aan de rand van de afbeelding maakt Elastix gebruik van een door de gebruiker ingestelde waarde. Deze waarde geeft aan wat de grijswaarde van pixels buiten de afbeelding is. Waar deze waarde niet overeen komt met de grijswaarden aan de rand van de afbeelding komen deze zichtbaar terug in de registratie. Een voorbeeld hiervan zijn de witte pixels aan de rand van figuur 16. Ook dit artefact kan worden vermeden door gebruik te maken van een masker. In dit geval moet het masker de pixels aan de rand van de afbeelding verbergen.

Als derde is er de onnauwkeurigheid als gevolg van de afstand tussen de knopen voor de B-splines. Bij een grote afstand worden sterke plaatselijke veranderingen niet goed berekend, aangezien splines juist vloeiende lijnen zijn. Bij een kleine afstand tussen de knopen zijn de verplaatsingen die gevonden kunnen worden niet altijd realistisch. Hierdoor kan de spline een vorm aannemen die op een sinusgolf lijkt, met een grote afwijking in het uiteindelijke resultaat als gevolg.

6.3 Onnauwkeurigheid in het deformatieveld

Het deformatieveld wordt berekend aan de hand van het verplaatsingsveld. Elke onnauwkeurigheid in het verplaatsingsveld vergroot de onnauwkeurigheid in het deformatieveld.

Door de deformatiematrix te vereenvoudigen van $F = \begin{bmatrix} 1+x_{C'}-x_D & x_{C'}-x_B \\ y_{C'}-y_D & 1+y_{C'}-y_B \end{bmatrix}$ naar $F = \begin{bmatrix} 1+a & b \\ c & 1+d \end{bmatrix}$ voor het deformatieveld wordt berekend, is de formule overzichtelijker. De deformatie op één punt is dan

$$Def = \sqrt{-\frac{1}{4}\cdot\left(Z+a^2+b^2+c^2+d^2+2\cdot a+2\cdot d+2\right)\cdot\left(Z-a^2-b^2-c^2-d^2-2\cdot a-2\cdot d-2\right)}$$

, met

$$Z = \sqrt{\left(a^2+b^2+c^2+d^2+2\cdot a\cdot d-2\cdot c\cdot d+4\cdot a+4\cdot d+4\right)\cdot\left(a^2+b^2+c^2+d^2+2\cdot c\cdot d-2\cdot a\cdot d\right)}$$

Door deze formule partieel te differentiëren kan de onnauwkeurigheid van het deformatieveld in dat punt worden berekend.

$$\Delta Def = \left|\frac{\partial Def}{\partial a}\right|\cdot\Delta x+\left|\frac{\partial Def}{\partial b}\right|\cdot\Delta x+\left|\frac{\partial Def}{\partial c}\right|\cdot\Delta x+\left|\frac{\partial Def}{\partial d}\right|\cdot\Delta x$$

$$\Delta Def = 4\cdot\Delta x\cdot\left(\left|(1+d)Q\right|+\left|c\cdot Q\right|+\left|b\cdot Q\right|+\left|(1+a)Q\right|\right), \text{ met}$$

$$Q = \frac{1+a+d+a\cdot d-c\cdot b}{\sqrt{G+2\cdot H\cdot K}\cdot\sqrt{G-2\cdot H\cdot K}}.$$

Hierbij geldt:

$$G = 2\cdot\left(a^2+b^2+c^2+d^2\right)+4\cdot(1+a+d)$$

$$H = \sqrt{a^2+b^2+c^2+d^2+2\cdot(a\cdot d-c\cdot b)+4\cdot(1+a+d)}$$

$$K = \sqrt{a^2+b^2+c^2+d^2+2\cdot(c\cdot b-a\cdot d)}$$

7 Conclusie

Elastix kan met zowel de geïdealiseerde geometrie, MRI afbeeldingen en ultrageluidafbeeldingen alle verplaatsingen goed bepalen. Bij grotere oppervlakten met gelijke grijswaarden ontstaan artefacten, maar dit kan verholpen worden door gebruik te maken van maskers of door alleen naar de verplaatsingen bij contouren te kijken.

De verplaatsingen bij de MRI-afbeeldingen zijn veel kleiner dan bij ultrageluidafbeeldingen. Het lijkt er daardoor op dat de interne druk bij de MRI-experimenten niet klopt met de ingestelde waarden.

De ultrageluidafbeeldingen kunnen goed worden gebruikt om de verplaatsingsvelden te berekenen. Hieruit kan vervolgens het deformatieveld worden berekend. Door dit deformatieveld te vergelijken met resultaten uit de eindige-elementenmethode kan deze methode worden gevalideerd.

8 Literatuurlijst

[1] *Het bloedvatenstelsel.* Binnengehaald 24 mei 2012 van http://www.natuurinformatie.nl/nnm.dossiers/natuurdatabase.nl/i002120.html

[2] Ron Waksman, Patrick W. Serruys, Johannes Schaar. *Vulnerable Plaque*, 2nd edition. Informa healthcare.

[3] Mamoo Nakamura, MD, David P. Lee, MD, Alan C. Yeung, MD Division of Cardiovascular Medicine, Stanford University Medical Center, Stanford CA. *Identification and Treatment of Vulnerable Plaque*. Coronary Artery Restenosis, vol 5, suppl 2, 2004.

[4] *What is Cholesterol?* Binnengehaald 7 juni 2012 van http://redpacketsg.blogspot.nl/2011/08/what-is-cholesterol.html

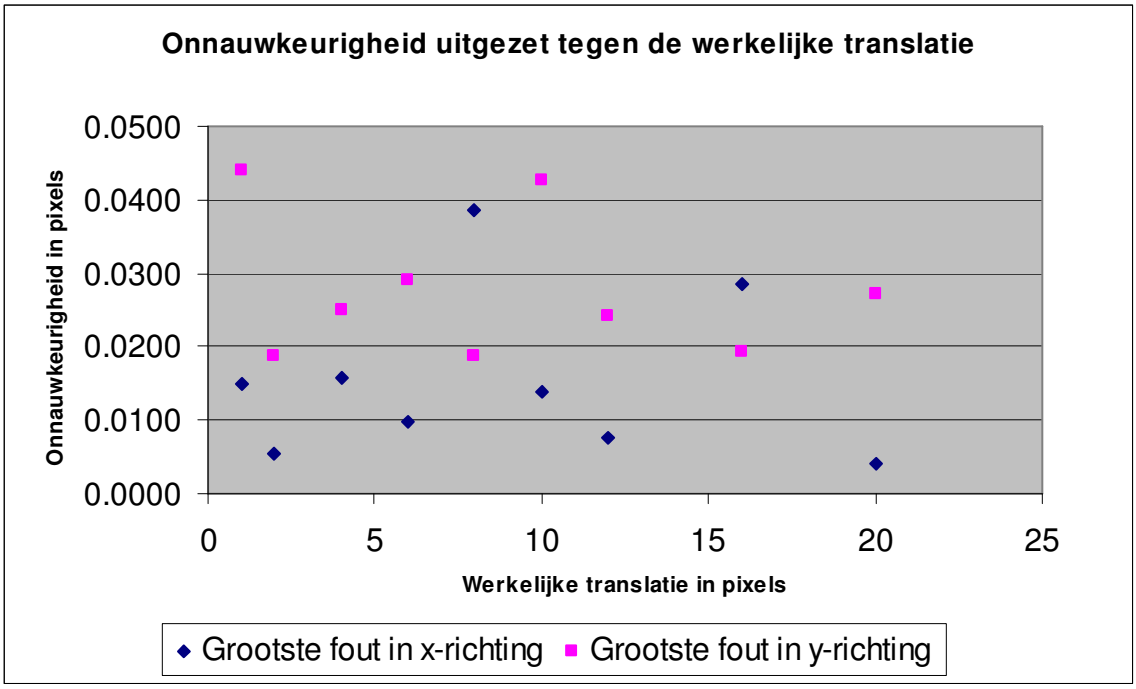
[5] *Parzen window*. Binnengehaald 1 mei 2012 van http://en.wikipedia.org/wiki/Parzen_window

[6] A. C. Akyildiz et al (2011)*, Effects of intima stiffnes and plaque morphology on peak cap stress*. Biomedical Engineering Online, 2011, 10:25.

[7] Weisstein, Eric W. *"Disk Point Picking."* From *MathWorld*--A Wolfram Web Resource. http://mathworld.wolfram.com/DiskPointPicking.html

[8] *Finite strain theory*. Binnengehaald 21 juni 2012 van http://en.wikipedia.org/wiki/Finite_strain_theory

Bijlage 1: Onnauwkeurigheid in affiniteit



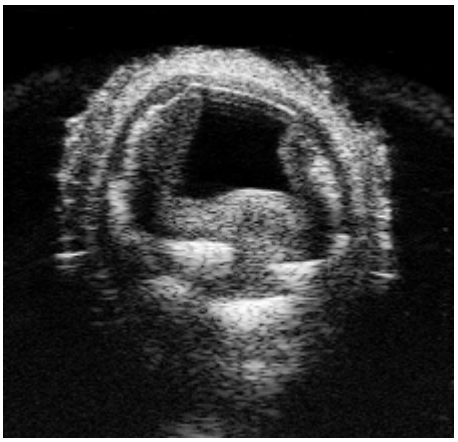
Figuur 29: Onnauwkeurigheid in translatiebepaling bij een geïdealiseerde geometrie uitgezet tegen de werkelijke translatie. De onnauwkeurigheid is onafhankelijk van de werkelijke translatie.

Bijlage 2: Onnauwkeurigheid in verandering in schaal

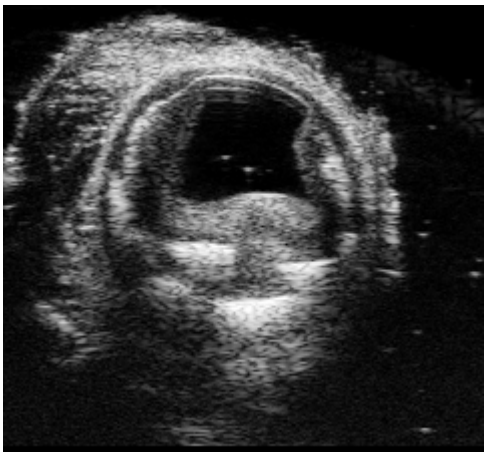
Tabel 5: De berekende onnauwkeurigheid bij bepaling van de verandering in schaal in een afbeelding. De waarden in de kolommen voor berekende rotatie zijn een maat voor de verplaatsing in de x- en y-richting relatief tot de afstand in respectievelijk de y- en x-richting tot het middelpunt van de afbeelding. De maximale onnauwkeurigheid is de grootste afstand een pixel kan bewegen gebaseerd op de waarden voor berekende verandering in schaal, rotatie en translatie ten opzichte van de werkelijke verplaatsing van de pixels. De relatieve fout is de maximale onnauwkeurigheid in diagonale richting ten opzichte van de gemiddelde werkelijke verplaatsing per pixel.

Originele verandering in schaal	Berekende verandering in schaal		Berekende rotatie		Berekende translatie		Maximale onnauwkeurigheid		Relatieve fout
%	x	y	x	y	x	y	x	y	%
90%	90.014%	90.026%	0.00009	-0.00021	-0.0996	-0.0244	0.1799	0.1952	0.029%
91%	91.006%	91.014%	-0.00119	0.00124	-0.0653	-0.0200	0.5102	0.5148	0.079%
92%	92.011%	92.010%	-0.00009	0.00015	-0.0988	-0.0108	0.1707	0.1017	0.021%
93%	93.017%	93.019%	-0.00040	0.00023	-0.0289	-0.0140	0.2317	0.1639	0.030%
94%	94.013%	94.008%	-0.00016	0.00008	-0.0384	-0.0335	0.1430	0.0909	0.018%
95%	95.007%	95.008%	0.00021	-0.00026	-0.0531	-0.0224	0.1527	0.1452	0.022%
96%	95.997%	96.019%	0.00001	-0.00011	-0.0579	-0.0250	0.0729	0.1338	0.016%
97%	97.032%	97.021%	0.00052	-0.00058	-0.0643	-0.0317	0.3633	0.3141	0.049%
98%	98.017%	98.013%	0.00002	-0.00014	-0.0441	-0.0121	0.1110	0.1091	0.016%
99%	99.021%	99.004%	-0.00010	-0.00011	-0.0473	-0.0420	0.1586	0.0975	0.019%
100%	100.000%	99.999%	-0.00001	0.00000	-0.0064	0.0002	0.0087	0.0061	0.001%
101%	100.995%	100.992%	-0.00052	0.00034	-0.0006	-0.0049	0.2035	0.1574	0.025%
102%	102.008%	102.009%	-0.00059	0.00063	0.0429	0.0106	0.2818	0.2701	0.038%
103%	103.010%	103.009%	-0.00012	-0.00006	0.0103	0.0066	0.0885	0.0625	0.010%
104%	104.002%	104.010%	-0.00012	-0.00024	-0.0033	0.0127	0.0550	0.1356	0.014%
105%	105.013%	105.015%	-0.00022	0.00007	0.0412	0.0191	0.1678	0.0973	0.018%
106%	106.007%	106.006%	-0.00017	0.00002	0.0005	0.0299	0.0868	0.0591	0.010%
107%	107.004%	107.004%	-0.00011	-0.00006	0.0015	0.0590	0.0528	0.0960	0.010%
108%	108.001%	107.992%	-0.00031	0.00002	-0.0194	0.0439	0.1320	0.0788	0.014%
109%	109.013%	109.013%	-0.00025	0.00005	0.0403	0.0058	0.1765	0.0710	0.017%
110%	110.016%	110.008%	-0.00098	0.00093	0.0484	0.0735	0.4562	0.4337	0.057%
111%	111.019%	111.012%	-0.00037	0.00006	0.0123	0.0384	0.2101	0.1042	0.021%
112%	112.018%	112.020%	0.00016	-0.00034	0.0369	0.0618	0.1567	0.2564	0.026%
113%	113.018%	113.005%	0.00126	-0.00159	0.0089	0.0541	0.5251	0.6424	0.073%
114%	114.027%	114.019%	-0.00024	0.00021	0.0467	0.0505	0.2265	0.1929	0.026%
115%	115.020%	115.001%	0.00067	-0.00078	0.0316	0.0480	0.3421	0.3310	0.041%
116%	116.024%	116.016%	-0.00045	0.00010	0.0663	0.0471	0.3139	0.1399	0.029%
117%	117.012%	117.023%	-0.00051	0.00039	0.0824	0.0509	0.3091	0.2735	0.035%
118%	118.010%	118.010%	-0.00102	0.00090	0.1014	0.0867	0.5014	0.4429	0.056%
119%	119.025%	119.013%	-0.00015	-0.00022	0.0691	0.0502	0.2139	0.1771	0.023%
120%	120.014%	120.027%	-0.00005	-0.00028	0.0770	0.0769	0.1459	0.2748	0.026%

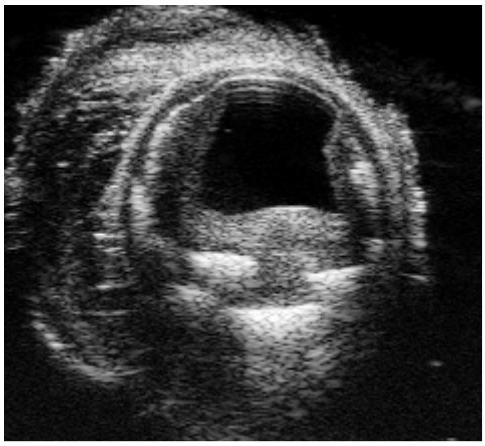
Bijlage 3: Ultrageluidafbeeldingen



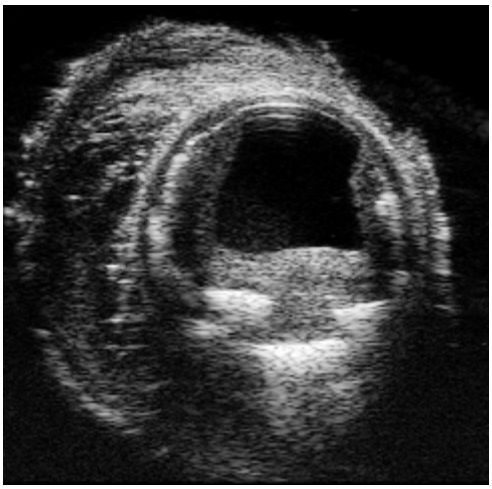
Figuur 30: Ultrageluidafbeelding bij een druk van 2,7 kPa (20 mmHg)



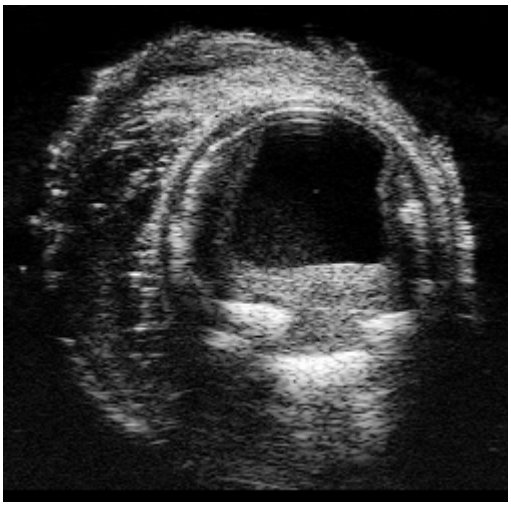
Figuur 31: Ultrageluidafbeelding bij een druk van 5,3 kPa (40 mmHg)



Figuur 32: Ultrageluidafbeelding bij een druk van 8,0 kPa (60 mmHg)

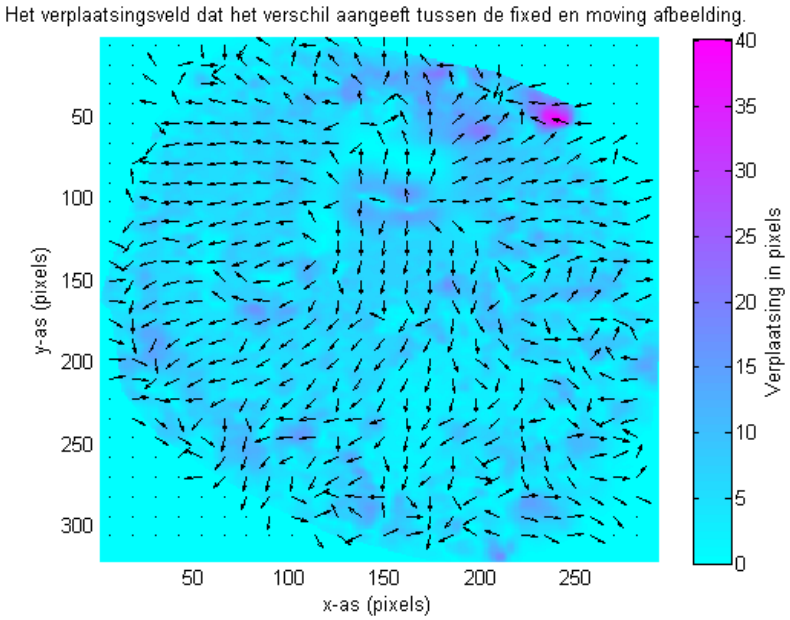


Figuur 33: Ultrageluidafbeelding bij een druk van 10,6 kPa (80 mmHg)

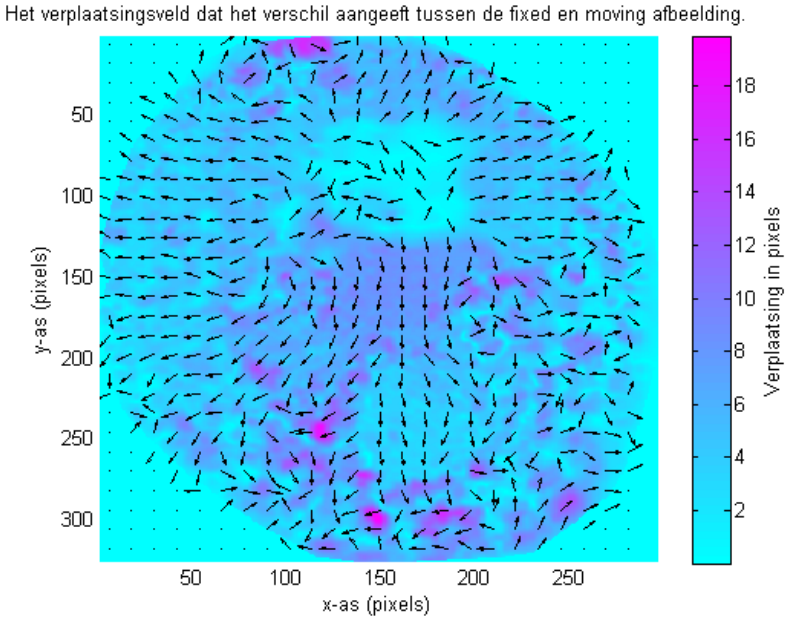


Figuur 34: Ultrageluidafbeelding bij een druk van 13,3 kPa (100 mmHg)

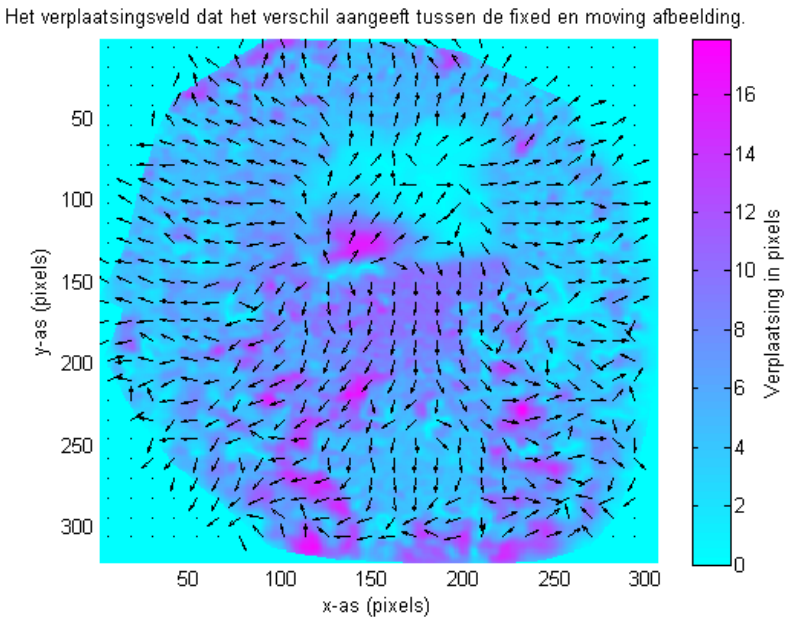
Bijlage 4: Ultrageluid verplaatsingsvelden



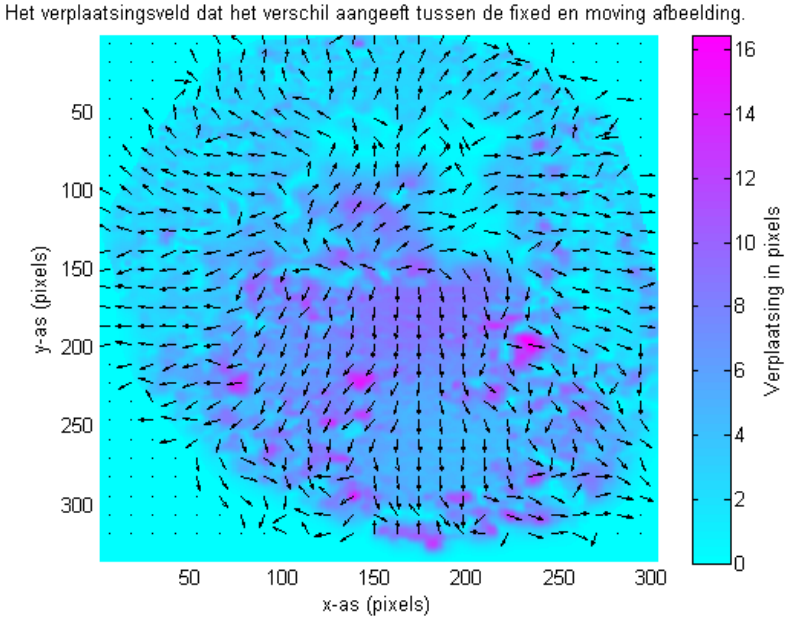
Figuur 35: Verplaatsingsveld dat het verschil aangeeft tussen de ultrageluidafbeeldingen bij 2,6 kPa en 5,3 kPa.



Figuur 36: Verplaatsingsveld dat het verschil aangeeft tussen de ultrageluidafbeeldingen bij 5,3 kPa en 8,0 kPa.

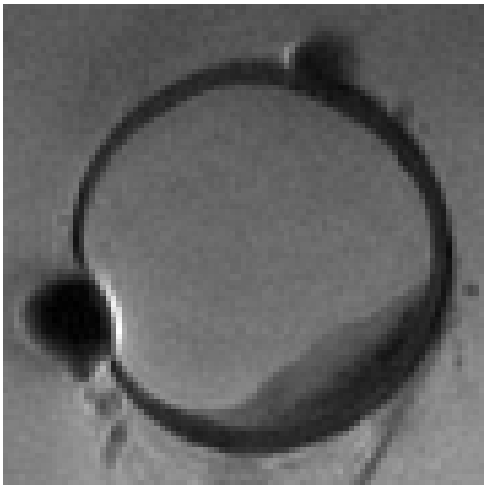


Figuur 37: Verplaatsingsveld dat het verschil aangeeft tussen de ultrageluidafbeeldingen bij 8,0 kPa en 10,6 kPa.

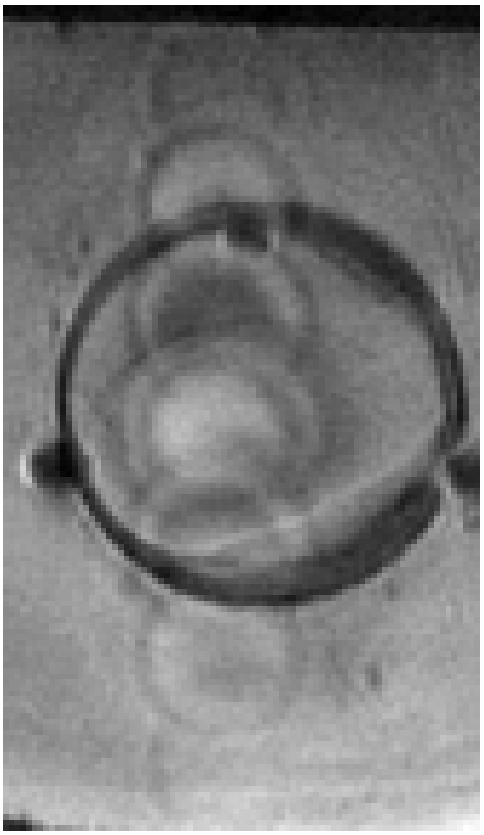


Figuur 38: Verplaatsingsveld dat het verschil aangeeft tussen de ultrageluidafbeeldingen bij 10,6 kPa en 13,3 kPa.

Bijlage 5: MRI artefacten



Figuur 39: MRI afbeelding met microbubbels. Luchtbellen met een diameter van enkele micrometers, kleiner dan één pixel, geven een verstoring in het beeld die vele malen groter is.



Figuur 40: MRI afbeelding met een 'shadowing' artefact. Een deel van de afbeelding wordt vele malen herhaald weergegeven in de afbeelding.

Bijlage 6: MATLAB script: geïdealiseerde geometrie

```
clear
clc

% Set ImagePath and file to save image
SaveImage       = 0;          % 0 to only show the image on screen
ImagePath       = 'C:\Projects\Project 1B\Noise\004\';
ImageSubPath    = '001\';
ImageName       = 'moving';   % 'fixed' or 'moving'
UseMask         = 1;

% Effects (positive integers for smoothing and pixalize or 0 to disable)
RotateAngle     = 0.0;        % Degrees, rotates counter-clockwise
Smoothing       = 0;          % Positive integer, or 0 to disable
NoiseLevel      = 0.00;       % Positive float, or 0 to disable
Pixalize        = 0;          % Positive integer, or 0 to disable
ScalingFactor   = 1.00;       % Homogeneous deformation of entire image, 1.0 to disable
TranslationX    = 0;          % Translation of entire image in x-direction in pixels, 0
                                to disable
TranslationY    = 0;          % Translation of entire image in y-direction in pixels, 0
                                to disable

% Set resolution (square image)
ROI             = 1.1;
ImageSize       = floor(512 * ROI);

% ------------------------------------------------------------------------
% -------------------------- BASE IMAGE DATA -----------------------------
% ------------------------------------------------------------------------

% Scaling: 1 px = 10 um at ImageSize of 512 * ROI
Scale = ImageSize / 5.12 / ROI * ScalingFactor;                          % [px/mm]

% Area size (sizes in scale * mm, angles in radians)
LumenRadius         = Scale * 1.250;
IntimaRadius        = Scale * 2.150;
MediaThickness      = Scale * 0.250;
AdventitiaThickness = Scale * 0.150;
CapThickness        = Scale * 0.150;
LipidThickness      = Scale * 1.000;
LipidAngle          = 30.0 / 180.0 * pi;

% Area centers (center of image at ImageSize / 2, top-left at 0)
IntimaCenterX       = ImageSize / 2 + TranslationX; % Also center of Media and Adventitia
IntimaCenterY       = ImageSize / 2 + TranslationY; % Also center of Media and Adventitia
LumenCenterX        = ImageSize / 2 + TranslationX;
LumenCenterY        = ImageSize / 2 + TranslationY + Scale * 0.640;
LipidCenterX        = ImageSize / 2 + TranslationX;
LipidCenterY        = ImageSize / 2 + TranslationY + Scale * 0.640;

% Area shades (grayscale, 0.01 = black, 1.00 is white, 0 is transparant)
LumenColor      = 200 / 255;
IntimaColor     = 150 / 255;
MediaColor      = 090 / 255;
AdventitiaColor = 060 / 255;
LipidColor      = 120 / 255;

% ------------------------------------------------------------------------
% ---------------------- IMAGE AREA CALCULATIONS -------------------------
% ------------------------------------------------------------------------

% Meshgrid for pixel coordiates in the matrix
[rr cc] = meshgrid(1:ImageSize);
```

```
% Lumen, Intima, Media and Adventitia area
Lumen = LumenColor * (sqrt((rr-LumenCenterX) .^ 2 + (cc-LumenCenterY) .^ 2) <=
LumenRadius);
Intima = IntimaColor * (sqrt((rr-IntimaCenterX) .^ 2 + (cc-IntimaCenterY) .^ 2) <=
IntimaRadius);
Media = MediaColor * (sqrt((rr-IntimaCenterX) .^ 2 + (cc-IntimaCenterY) .^ 2) <=
IntimaRadius + MediaThickness);
Adventitia = AdventitiaColor * (sqrt((rr-IntimaCenterX) .^ 2 + (cc-IntimaCenterY) .^ 2)
<= IntimaRadius + MediaThickness + AdventitiaThickness);

% Lipid area
LipidRadius = LumenRadius + CapThickness;

LipidInner = (sqrt((rr-LipidCenterX) .^ 2 + (cc-LipidCenterY) .^ 2) <= LipidRadius);
LipidOuter = (sqrt((rr-LipidCenterX) .^ 2 + (cc-LipidCenterY) .^ 2) <= LipidRadius +
LipidThickness);
LipidRing = (LipidInner==0) .* (LipidOuter>0);

LipidAngleMap = atan2((rr - LipidCenterX), (LipidCenterY - cc));
LipidSemiRing = LipidRing .* (abs(LipidAngleMap) < LipidAngle);

LipidCircleY = (LipidRadius + LipidThickness * 0.5) * sin(pi * 0.5 - LipidAngle);
LipidCircleX = (LipidRadius + LipidThickness * 0.5) * cos(pi * 0.5 - LipidAngle);
LipidLeftCircle = (sqrt((rr-LipidCenterX-LipidCircleX) .^ 2 + (cc-
LipidCenterY+LipidCircleY) .^ 2) <= LipidThickness / 2);
LipidRightCircle = (sqrt((rr-LipidCenterX+LipidCircleX) .^ 2 + (cc-
LipidCenterY+LipidCircleY) .^ 2) <= LipidThickness / 2);

Lipid = LipidColor * min(LipidLeftCircle + LipidRightCircle + LipidSemiRing, 1);

% Add materials together. Later materials overwrite previous materials!
MaterialSum = ones(ImageSize);
MaterialSum = (Adventitia==0) .* MaterialSum + Adventitia; % Add adventitia
MaterialSum = (Media==0)      .* MaterialSum + Media;      % Add media
MaterialSum = (Intima==0)     .* MaterialSum + Intima;     % Add intima
MaterialSum = (Lumen==0)      .* MaterialSum + Lumen;      % Add lumen
MaterialSum = (Lipid==0)      .* MaterialSum + Lipid;      % Add lipid

% ------------------------------------------------------------------------
% ------------------------- SPECIAL EFFECTS ------------------------------
% ------------------------------------------------------------------------

% Mask
if UseMask
    if strcmp(ImageName, 'fixed')
        MaskName = 'fMask';
    else
        MaskName = 'mMask';
    end
    ImageMask = (MaterialSum < 1);
end

% Rotation
if RotateAngle ~= 0  % Skip rotation if angle is 0
    Rotated = 1 - imrotate(1 - MaterialSum, RotateAngle); % Rotate negative of image to
                                                            keep outside white
    RightBorder = round((size(Rotated,1) + ImageSize) / 2);
    LeftBorder = round((size(Rotated,1) - ImageSize) / 2 + 1);
    MaterialSum = Rotated(LeftBorder:RightBorder, LeftBorder:RightBorder);
end

% Smoothing
if Smoothing > 0 % Skip smoothing if 0
    ConvArea = ones(Smoothing * 2 + 1);                              % Smoothing area
    MaterialSum = conv2(MaterialSum, ConvArea, 'same');              % 2d convolution
    MaterialSum = MaterialSum / max(reshape(MaterialSum,[],1));      % Normalisation
end



% Gray noise
```

```
if NoiseLevel > 0                % Skip noise generation if level is 0
    Noise = random('normal',-NoiseLevel,NoiseLevel,ImageSize,ImageSize);
    MaterialSum = max(min(MaterialSum + Noise, 1), 0);
end

% Pixalisation
if Pixalize > 0                 % Skip pixalisation if voxel size equals pixel size
    Pixalize = 2^Pixalize;      % Only works with powers of 2

    % Pixalized coordinates
    pix = min(ceil((1:ImageSize) / Pixalize) * Pixalize - Pixalize/2, ImageSize);
    MaterialSum(1:ImageSize, 1:ImageSize) = MaterialSum(pix, pix);
end

% ------------------------------------------------------------------------
% ----------------------- SAVING DATA TO FILES ---------------------------
% ------------------------------------------------------------------------

% Save and show results
figure;
if SaveImage == 1
    if not(exist([ImagePath ImageSubPath], 'dir'))
        mkdir(ImagePath, ImageSubPath);
    end
    fullpath = [ImagePath ImageSubPath ImageName '.tif'];
    cmap = gray;
    IndI = ceil(MaterialSum * size(cmap, 1));
    imwrite(IndI, cmap, fullpath);
    imshow(fullpath);
    if UseMask
        imwrite(ImageMask, [ImagePath ImageSubPath '\' MaskName '.tif']);
    end
else
    imshow(MaterialSum)
end
```

Bijlage 7: MATLAB script: ImageCompare.m

Visualisatie van het verplaatsingsveld, berekening van automatische contouren, berekening van de grootte van het lumen en berekening en visualisatie van het deformatieveld.

```
clear
clc

% Directory to read the images and elastix data from
ImagePath           = 'C:\Projects\Project 3B\Experiment 3\slice 40\2\';

% Name and format of the fixed image
FixedImageName      = 'subfixed';
ImageFormat         = 'png';

% Image Settings
UseMask             = 1;     % 0 to not use a ImageMask even if it does exist
MinimumDisplacement = 0.000; % Remove all displacements that are smaller
MaximumDisplacement = 0.000; % Remove all displacements that are larger; 0 to disable
Pix                 = 12;    % Distance between displacement field arrows; must be an
                               even number
Scaling             = 1;     % Resize the image

% Automatic contours.
UseContours         = 0;      % Set to 1 to draw automatic contours
    ContourSmoothing    = 6;  % Size of smoothing area
    SmoothingSteps      = 3;  % Number of times smoothing is performed
    ContourBins         = [-1 30 180];       % Grayvalues to draw contours at
    ContourLineSize     = 2;  % Line thickness of the contours
    CalculateLumen      = 0;  % Set to 1 to calculate the size of the lumen

% Deformation field. Do not use contours at the same time.
DeformationNotDisplacement  = 1; % Set to 1 to show deformation field instead of
                                    displacement field
    Smoothing                   = 0; % Size of smoothing area to reduce noise; not needed
                                       when using segments
    SegmentDeformationData      = 0; % Number of radial segments to create, or 0 to not
                                       use segments.
        SegmentRows             = 1; % Number of segments to create per radial segment,
                                       must be at least 1.
    DrawLumenMask               = 1; % Set to 1 to use or create a second mask to cover
                                       the lumen area
        UseExistingLumenMask    = 1; % Set to 1 to use an existing lumen mask if one is
                                       aviable

% Other settings
ResultInSubplots            = 0;  % Set to 1 to have all images in a single figure window

%Save deformation images to a file if set to 1
SaveDeformationData     = 1;
OutputPath              = 'C:\Projects\Project 3B\Segmented\Experimemt 3\Slice 40\2\';


% --------------------------------------------------------------------
% ------------------------- LOAD DATA FILES --------------------------
% --------------------------------------------------------------------

% Load images
if strcmp(ImageFormat, 'dcm')
    FixedImage = double(dicomread([ImagePath FixedImageName '.dcm']));
else
    FixedImage = double(imread([ImagePath FixedImageName '.' ImageFormat]));
end
FixedImage = imresize(FixedImage, Scaling);

% Load ImageMask if the mask can be found
```

```
if UseMask
    if exist([ImagePath 'fMask.dcm'], 'file')
        ImageMask = double(dicomread([ImagePath 'fMask.dcm']));
        ImageMask = ImageMask / max(reshape(ImageMask, [], 1));
    elseif exist([ImagePath 'fMask.' ImageFormat], 'file')
        ImageMask = double(imread([ImagePath 'fMask.' ImageFormat]));
        ImageMask = ImageMask / max(reshape(ImageMask, [], 1));
    else
        UseMask = 0;
    end
end

% Turn coloured image into grayscale image
if size(FixedImage, 3) == 3
   FixedImage = (FixedImage(:,:,1) + FixedImage(:,:,2) + FixedImage(:,:,3))/3;
end

% Load header file for the displacement field (created by Elastix)
fid = fopen([ImagePath 'deformationField.mhd']);
header = textscan(fid, '%s', 'delimiter', '\n', 'whitespace', '');
fclose(fid);
header = header{:};
DimSize = regexp(header{10}, ' ', 'split'); % Get the image size
DimSizeX = str2double(DimSize{3});
DimSizeY = str2double(DimSize{4});
ElementSpacing = regexp(header{9}, ' ', 'split'); % Get the image spacing values
ElementSpacingX = str2double(ElementSpacing{3});
ElementSpacingY = str2double(ElementSpacing{4});

% Load displacement field (created by Elastix)
ImageSize = [DimSizeY, DimSizeX, 2];
precision = 'float';
offset = 0;
interleave = 'bip';
byteorder = 'ieee-le';
Transform = multibandread([ImagePath 'deformationField.raw'], ImageSize, precision,
offset, interleave, byteorder);
Transform(:,:,1) = Transform(:,:,1) / ElementSpacingX;
Transform(:,:,2) = Transform(:,:,2) / ElementSpacingY;

% Find and load ResultImage picture (created by Elastix)
for n = 8:-1:-1
    ResultImageName = ['Result.' num2str(n) '.raw'];
    if exist([ImagePath ResultImageName], 'file')
       break
    end
end
ResultImage = multibandread([ImagePath ResultImageName], [DimSizeY DimSizeX 1], 'double',
offset, interleave, byteorder);
ResultImageScale = mean(reshape(ResultImage, [], 1)) / mean(reshape(FixedImage, [], 1));

% ------------------------------------------------------------------------
% -------------------------- APPLY IMAGE TOOLS ---------------------------
% ------------------------------------------------------------------------

% Apply ImageMask
if UseMask
    % Cropping to only include ImageMask area
    [~, MaskMinX] = max(max(ImageMask));                    % Left coordinate
    [~, MaskMaxX] = max(max(fliplr(ImageMask)));
    MaskMaxX = size(ImageMask, 2) - MaskMaxX;               % Right coordiate
    [~, MaskMinY] = max(max(ImageMask, [], 2));             % Upper coordinate
    [~, MaskMaxY] = max(max(flipud(ImageMask), [], 2));
    MaskMaxY = size(ImageMask,1) - MaskMaxY;                % Lower coordinate

    % Crop ImageMask, fixed image, result image and displacement field
    ImageMask = ImageMask(MaskMinY : MaskMaxY, MaskMinX : MaskMaxX);
    FixedImage = FixedImage(MaskMinY : MaskMaxY, MaskMinX : MaskMaxX);
    ResultImage = ResultImage(MaskMinY : MaskMaxY, MaskMinX : MaskMaxX);
    Transform = Transform(MaskMinY : MaskMaxY, MaskMinX : MaskMaxX, :);
    % Get new image size
```

```
    DimSizeY = size(ImageMask, 1);
    DimSizeX = size(ImageMask, 2);

    % Apply the ImageMask
    Transform = Transform .* cat(3, ImageMask, ImageMask);
end

% Calculate and apply contours
if UseContours
    % Smoothing
    GrayImage = FixedImage;
    if ContourSmoothing > 0
        ConvArea = ones(ContourSmoothing) / (ContourSmoothing ^ 2);
        for s = 1:SmoothingSteps
            GrayImage = conv2(GrayImage, ConvArea, 'same');
        end
    end

    % Binning
    BinnedImage = zeros(size(GrayImage, 1), size(GrayImage, 2));
    for n = 1:size(ContourBins,2)
        BinnedImage = BinnedImage + (GrayImage > ContourBins(n)) * 1.0;
    end

    % Apply ImageMask
    if UseMask
        BinnedImage = BinnedImage .* ImageMask;
    end

    % Get contours
    ConvArea = ones(1 + 2 * ContourLineSize) / ((1 + 2 * ContourLineSize) ^ 2);
    SmoothedBinnedImage = conv2(BinnedImage, ConvArea, 'same');
    ContouredImage = abs(BinnedImage - SmoothedBinnedImage) > 1e-9;
    ContouredImage = ContouredImage .* BinnedImage;

    % Remove deformation information that is not on contours
    Transform = Transform .* cat(3, (ContouredImage>0), (ContouredImage>0));
end

% Get size of transformation
TransformSize = sqrt(abs(Transform(:,:,1)) .^ 2 + abs(Transform(:,:,2)) .^ 2);

% Remove small deformations and set upper limit on large deformations
TransformSize(TransformSize < MinimumDisplacement) = 0;
if MaximumDisplacement > 0
    TransformSize(TransformSize > MaximumDisplacement) = MaximumDisplacement;
end
Transform = Transform .* cat(3, (TransformSize ~= 0), (TransformSize ~= 0));

% Average deformation arrows over small areas
PixBinX = (1:floor(DimSizeX/Pix)) * Pix - Pix / 2;
PixBinY = (1:floor(DimSizeY/Pix)) * Pix - Pix / 2;
ConvArea = ones(Pix) / Pix^2;
TransformX = conv2(Transform(:,:,1), ConvArea, 'same');
TransformY = conv2(Transform(:,:,2), ConvArea, 'same');
TransformX = TransformX(PixBinY, PixBinX);
TransformY = TransformY(PixBinY, PixBinX);

% ------------------------------------------------------------------------
% --------------------- DISPLACEMENT TO DEFORMATION ----------------------
% ------------------------------------------------------------------------

if DeformationNotDisplacement

    if DrawLumenMask
        % Find and load existing lumen mask
        if UseExistingLumenMask
            if exist([ImagePath 'fLumenMask.png'], 'file')
                LumenMask = double(imread([ImagePath 'fLumenMask.png']));
                LumenMask = LumenMask / max(reshape(LumenMask, [], 1));
            else
```

```
            UseExistingLumenMask = 0;
        end
    end

    % Let user draw a new lumen mask
    if ~UseExistingLumenMask
        h=figure;
        imagesc(FixedImage);
        set(gcf, 'Position', get(0, 'Screensize'));
        colormap(gray);
        title('Draw the mask for the fixed image');
        InputMaskPoints = imfreehand;
        LumenMask = 1-(InputMaskPoints.createMask) * 1.0;
        close(h);

        % Save the lumen mask to a file
        imwrite(LumenMask, [ImagePath 'fLumenMask.png'], 'png');
    end

    % Apply the lumen mask
    Transform = Transform .* cat(3, LumenMask, LumenMask);

    % Calculate the center of the lumen mask
    SegmentCenter = regionprops(LumenMask==0, 'Centroid');
    SegmentCenter = SegmentCenter.Centroid;
else
    % Calculate the center of the image
    SegmentCenter = [DimSizeX / 2, DimSizeY / 2];
end

% Make a copy of the displacement field before overwriting the original
OldTransform = Transform;

% Deformation matrix
Transform(:, 2:DimSizeX, 1) = 1 + OldTransform(:, 2:DimSizeX, 1) - OldTransform(:,
    1:DimSizeX-1, 1);    % compare x of points with same y
Transform(2:DimSizeY, :, 2) = OldTransform(2:DimSizeY, :, 1) -
    OldTransform(1:DimSizeY-1, :, 1);        % compare x of points with same x
Transform(:, 2:DimSizeX, 3) = OldTransform(:, 2:DimSizeX, 2) - OldTransform(:,
    1:DimSizeX-1, 2);        % compare y of points with same y
Transform(2:DimSizeY, :, 4) = 1 + OldTransform(2:DimSizeY, :, 2) -
    OldTransform(1:DimSizeY-1, :, 2);    % compare y of points with same x

% Transpose in 3rd dimension
TransformTranspose = Transform;
% switch 2nd and 3rd collumn around
TransformTranspose(:, :, [2 3]) = TransformTranspose(:, :, [3 2]);
% Reshape dimensions from {y * x * 4} to {y * x * 2 * 2}
TransformTranspose = reshape(TransformTranspose, DimSizeY, DimSizeX, 2, 2);
Transform = reshape(Transform, DimSizeY, DimSizeX, 2, 2);

% Calculate right Caughy-Green tensor
TransformTranspose = Transform .* TransformTranspose;

% Eigenvectors
% Switch dimensions from {y by x by 2 by 2} to {2 by 2 by y by x}
TransformTranspose = permute(TransformTranspose, [3 4 1 2]);
d=zeros(DimSizeY, DimSizeX, 2);
% Calculate the eigenvalues for each location
for y = 1:DimSizeY
    for x = 1:DimSizeX
        d(y,x,:) = eig(TransformTranspose(:,:,y,x));
    end
end

% Sort the eigenvalues so the first collumn has the lowest values
d = reshape(d, [], 2); % Reshape to a list with 2 collumns         {xy+y by 2}
d = permute(d, [2 1]); % Reshape to a list with 2 rows             {2 by xy+y}
d = sort(d);           % Sort each collumn
d = permute(d, [2 1]); % Reshape back to a list with 2 collumns    {xy+y by 2}
d = reshape(d, DimSizeY, DimSizeX, 2);  % Reshape back to an array {x by y by 2}
```

```
    % Square root to get principal stretch
    Transform = real(d.^.5);


% Smoothing
    if Smoothing > 0
        % Square smoothing area
        ConvArea = ones(Smoothing * 2 + 1);
        for n = 1:2
            temp = Transform(:,:,n);

            % 2d convolution and normalisation
            temp = conv2(temp, ConvArea, 'same');
            temp = temp / sum(reshape(ConvArea,[],1));

            Transform(:,:,n) = temp;
        end
    end


    % Segmenting
    if SegmentDeformationData > 0

        % Angle for each radial segment
        SegmentAngle = 2 * pi / SegmentDeformationData;

        % Create a map showing all angles and distances from the center
        [rr cc] = meshgrid(1:DimSizeX, 1:DimSizeY);
        SegmentAngleMap = atan2((rr - SegmentCenter(1)), (SegmentCenter(2) - cc)) + pi;
        SegmentDistanceMap = sqrt((rr-SegmentCenter(1)).^2 + (cc-SegmentCenter(2)).^2);

        % Pre-allocate arrays
        SegmentEigenNaN = zeros(DimSizeY, DimSizeX);
        CutValues = zeros(SegmentDeformationData,2);

        for m = 1:SegmentDeformationData

            % Create a segment
            clear Segment
            Segment = (abs(SegmentAngleMap) >= (m-1) * SegmentAngle) .*
               (abs(SegmentAngleMap) <= m * SegmentAngle);

            % Make a copy of the image masks, and apply the segment to it
            ImageMaskArea = sum(reshape(ImageMask .* Segment, [], 1));
            LumenMaskArea = sum(reshape((1-LumenMask).* Segment, [], 1));

            % Calculate distances to split the segment at to make rows
            ImageMaskRadius  = (ImageMaskArea  / pi * SegmentDeformationData) ^ 0.5;
            LumenMaskRadius = (LumenMaskArea  / pi * SegmentDeformationData) ^ 0.5;
            SegmentRadius = ((1:(SegmentRows+1))-1) * (ImageMaskRadius - LumenMaskRadius)
               / SegmentRows + LumenMaskRadius;
            SegmentRadius(1) = 0;
            SegmentRadius(SegmentRows+1) = 32000;

            % Split segment into rows
            for n = 1:SegmentRows
                % Create the segment rows
                Segment = (abs(SegmentAngleMap) >= (m-1) * SegmentAngle) .*
                       (abs(SegmentAngleMap) <= m * SegmentAngle);
                Segment = Segment .* (SegmentDistanceMap >= SegmentRadius(n)) .*
                       (SegmentDistanceMap <= SegmentRadius(n+1));

                % Get deformation data that is on the segment row
                SegmentEigen = Transform .* cat(3, Segment, Segment);

                % Replace zeros with NaNs
                SegmentEigenNaN = SegmentEigen;
                SegmentEigenNaN(SegmentEigen==0) = NaN;

                % Get the mean of the segment
```

```
                SegmentMean = nanmean(reshape(SegmentEigenNaN, [], 2));
                CutValues(m,:) = SegmentMean;



% Fill an area with the mean values
                SegmentMean = cat(3, SegmentMean(1) * ones(DimSizeY, DimSizeX),
                      SegmentMean(2) * ones(DimSizeY, DimSizeX));

                % Replace values with same location as the segment with the mean value
                SegmentEigen(SegmentEigen == 0) = Transform(SegmentEigen == 0);
                SegmentEigen(SegmentEigen > 0) = SegmentMean(SegmentEigen > 0);
                Transform(cat(3, Segment, Segment) > 0) = SegmentEigen(cat(3, Segment,
                      Segment) > 0);
            end
        end
    end

    % Apply the masks again if needed
    if DrawLumenMask;
        Transform = Transform .* cat(3, LumenMask, LumenMask);
    end
    if UseMask;
        Transform = Transform .* cat(3, ImageMask, ImageMask);
    end

end

% ------------------------------------------------------------------------
% ------------------------- DISPAY EIGENVALUES ---------------------------
% ------------------------------------------------------------------------
if DeformationNotDisplacement
    hFig1 = figure;
    if ResultInSubplots
        % Make figure window full screen if all images are to be drawn in a
        % single figure window
        set(gcf, 'Position', get(0, 'Screensize'));
    end

    % Set the lowest value on the colormap to black
    cmap = colormap(jet);
    if SegmentDeformationData > 0
        Transform(Transform ==0) = NaN;
        MinTransform = min(reshape(Transform, [], 2)) * 0.95;
        Transform(1,1,1) = MinTransform(1);
        Transform(1,1,2) = MinTransform(2);
        cmap(1,:) = 0;
    end

    % Minimum principal stretch
    if ResultInSubplots
        subplot(2,2,1);
    end
    colormap(cmap)
    imagesc(real(Transform(:,:,1)));
    colorbar
    title('Minimum principal stretch')
    xlabel('x-axis (pixels)')
    ylabel('y-axis (pixels)')
    cblabel('Strech')

    % Maximum principal stretch
    if ResultInSubplots
        subplot(2,2,2);
    else
        hFig2 = figure;
    end
    colormap(cmap)
    imagesc(real(Transform(:,:,2)));
    colorbar
    title('Maximum principal stretch')
```

```
    xlabel('x-axis (pixels)')
    ylabel('y-axis (pixels)')
    cblabel('Strech')

    % Original image
    if ResultInSubplots
        subplot(2,2,3);
    else
        hFig3 = figure;
    end
    image(max(cat(3, FixedImage, FixedImage, FixedImage) / max(reshape(FixedImage, [],
        1)),0));
    title('Original image')
    xlabel('x-axis (pixels)')
    ylabel('y-axis (pixels)')
    cblabel('Gray value')

    % Multiplied image
    if ResultInSubplots
        subplot(2,2,4);
    else
        hFig4 = figure;
    end
    colormap(cmap)
    imagesc(real(Transform(:,:,1) .* Transform(:,:,2)));
    colorbar
    title('Deformation field')
    xlabel('x-axis (pixels)')
    ylabel('y-axis (pixels)')
    cblabel('Deformation')

    % Draw a line when image is not segmented
    if SegmentDeformationData == 0
        % Get a cut
        [x, y] = ginput(2);
        x = round(x);
        y = round(y);

        % Get the points on the line
        LineAngle = atan2(y(2) - y(1), x(2) - x(1)) / pi * 180 - 90;
        CutValuesMin = Transform(min(y) : max(y), min(x) : max(x), 1);
        CutValuesMinR = imrotate(CutValuesMin, LineAngle);
        CutValuesMinL = CutValuesMinR(:, max(1, round(size(CutValuesMinR,2) / 2 - 0.5)));
        CutValuesMax = Transform(min(y) : max(y), min(x) : max(x), 2);
        CutValuesMaxR = imrotate(CutValuesMax, LineAngle);
        CutValuesMaxL = CutValuesMaxR(:, max(1, round(size(CutValuesMaxR,2) / 2 - 0.5)));
        CutValues = cat(2, CutValuesMinL, CutValuesMaxL);

        % Draw the line
        hold on
        line(x, y, 'Color','r');
        hold off

        if ResultInSubplots
            subplot(2,2,4);
        else
            hFig4 = figure;
        end
        imagesc(real(CutValues));
        colorbar
        title('Minimum and maximum principal stretch over a line')
        xlabel('x-axis (pixels)')
        ylabel('y-axis (pixels)')
        cblabel('Stretch')
    end

    % Save images to files
    if SaveDeformationData

        % Create output directory if needed
        if not(exist(OutputPath, 'dir'))
```

```
            mkdir(OutputPath);
        end

        % Create output file
        fid = fopen([OutputPath 'EigenValues.txt'], 'w');

        % Write data to text file
        if SegmentDeformationData > 0
            fprintf(fid, '%d segments\r\nStarting at the bottom going clockwise\r\n\r\n',
               SegmentDeformationData);
        else
            fprintf(fid, 'x coordinate:  %6.0f\r\n', x(1));
            fprintf(fid, 'y coordinates: %6.0f %6.0f\r\n\r\n', y);
        end
        fprintf(fid, '%6.6f    %6.6f\r\n', CutValues.');
        fclose(fid);

        % Save imags
        if ResultInSubplots
            saveas(hFig1, [OutputPath 'EigenValues.png']);
        else
            saveas(hFig1, [OutputPath 'SegmentedMinimumPrincipalStress.png']);
            saveas(hFig2, [OutputPath 'SegmentedMaximumPrincipalStress.png']);
            saveas(hFig4, [OutputPath 'MultipliedPrincipalStress.png']);
        end
    end

else
% ------------------------------------------------------------------------
% --------------------- DISPAY DISPLACEMENT IMAGES -----------------------
% ------------------------------------------------------------------------
    figure;

    % Show contours
    if UseContours

        % Calculate the size of the lumen in pixels²
        if CalculateLumen
            % Draw contour image over fixed image
            axis([1, size(GrayImage, 2), 1, size(GrayImage, 1)])
            colormap(gray(max(reshape(FixedImage, [], 1))))
            colorbar;
            cbfreeze
            colormap(jet)
            set(gca, 'ydir', 'reverse');
            hold on
                image(max(cat(3, FixedImage, FixedImage, FixedImage) /
                      max(reshape(FixedImage, [], 1)),0));
                image(ContouredImage * max(reshape(FixedImage, [], 1)) /
                      max(reshape(ContouredImage, [], 1)),  'AlphaData', (ContouredImage
                      > 0) * 0.60);
            hold off
            freezeColors
            xlabel('x-axis (pixels)')
            ylabel('y-axis (pixels)')
            cblabel('Gray value')

            % Get user input for lumen location
            title('Select opposite corners of lumen ROI')
            LumenCorners = sort(round(ginput(2)));
            title('Select any point inside the lumen')
            LumenCenter = round(ginput(1));

            % Calculate lumen size
            LumenValue = BinnedImage(LumenCenter(2), LumenCenter(1));
            LumenCoordY = LumenCorners(1,2): LumenCorners(2,2);
            LumenCoordX = LumenCorners(1,1): LumenCorners(2,1);
            BinnedLumenImage = BinnedImage(LumenCoordY, LumenCoordX);
            LumenSize = sum(reshape(BinnedLumenImage == LumenValue, [], 1));

            % Refresh figure window
```

```
            close;
            figure;
        end

        % Draw contour image over fixed image
        if ResultInSubplots
            set(gcf, 'Position', get(0, 'Screensize'));
            subplot(2,2,4)
        end
        axis([1, size(GrayImage, 2), 1, size(GrayImage, 1)])
        colormap(gray(max(reshape(FixedImage, [], 1))))
        colorbar;
        cbfreeze
        colormap(jet)
        set(gca, 'ydir', 'reverse');
        hold on
            image(max(cat(3, FixedImage, FixedImage, FixedImage) /
               max(reshape(FixedImage, [], 1)),0));
            image(ContouredImage / max(reshape(ContouredImage, [], 1)) *
               max(reshape(FixedImage, [], 1)),  'AlphaData', (ContouredImage > 0) *
               0.60);
        hold off
        freezeColors
        xlabel('x-axis (pixels)')
        ylabel('y-axis (pixels)')
        title('Fixed image with automatic contours')
        cblabel('Gray value')
    elseif ResultInSubplots
        set(gcf, 'Position', get(0, 'Screensize'));
    end

    % Show deformation arrows on top of ResultImage picture
    if ResultInSubplots
        subplot(2,2,2)
    elseif UseContours
        figure;
    end
    axis([1, size(TransformSize, 2), 1, size(TransformSize, 1)])
    colormap('gray')
    set(gca, 'ydir', 'reverse');
    hold on
        imagesc(ResultImage, [0 max(reshape(ResultImage, [], 1))]);
        quiver(PixBinX, PixBinY, TransformX, TransformY, 'r');
    hold off
    freezeColors
    colorbar
    cbfreeze
    xlabel('x-axis (pixels)')
    ylabel('y-axis (pixels)')
    title('Resultimage with the displacement field')
    cblabel('Gray value')

    % Show registration (ResultImage minus FixedImage)
    if ResultInSubplots
        subplot(2,2,3)
    else
        figure;
    end
    if UseMask
        ResultImage = ResultImage .* ImageMask;
        FixedImage = FixedImage .* ImageMask;
    end
    colormap('jet')
    set(gca, 'ydir', 'reverse');
    imagesc(abs(ResultImage - FixedImage))
    ResultImageSum = sum(reshape(abs(ResultImage - FixedImage), [], 1));
    ResultImageMax = max(reshape(abs(FixedImage), [], 1));
    freezeColors
    colorbar
    cbfreeze
    xlabel('x-axis (pixels)')
```

```
    ylabel('y-axis (pixels)')
    title('Registration of the fixed and result image')
    cblabel('Difference in gray value between the images')

    % Show displacement field
    if ResultInSubplots
        subplot(2,2,1)
    else
        figure;
    end
    axis([1, size(TransformSize, 2), 1, size(TransformSize, 1)])
    colormap('cool')
    Arrowsize = sqrt(TransformX .^ 2 + TransformY .^ 2);
    set(gca, 'ydir', 'reverse');
    hold on
        imagesc(TransformSize)
        quiver(PixBinX, PixBinY, TransformX ./ Arrowsize, TransformY ./ Arrowsize, 0.5,
'black')
    hold off
    colorbar
    cbfreeze

    xlabel('x-axis (pixels)')
    ylabel('y-axis (pixels)')
    title('The displacement field which shows the difference between the fixed and moving
image.')
    cblabel('Displacement in pixels')
end
```

Bijlage 8: Originele opdrachtomschrijving

MR Imaging of atherosclerotic porcine iliac arteries
Plaque is een afzetting van kalk en vet in bloedvaten. Als zo'n afzetting loslaat, kan deze op een andere plek een verstopping veroorzaken. Dit kan een infarct als gevolg hebben.

Er wordt veel onderzoek gedaan naar plaque. Eén van de onderzoeken hierbij is de bepaling wanneer plaque loslaat of afbreekt. De afstudeeropdracht hoort tot dit onderzoek. Het doel hierbij is om te bepalen op welke locaties en bij welke druk plaque loslaat.

Bij een stukje bloedvat met plaque van een varken zijn onder andere MRI-afbeeldingen gemaakt waarbij verschillende drukken op de plaque zijn uitgeoefend. Door deze afbeeldingen en theoretische modellen te vergelijken kunnen verplaatsingen worden bepaald. Deze verplaatsingen, samen met materiaaleigenschappen van de verschillende materialen in plaque en de uitgeoefende druk leiden tot de bepaling van rek in het materiaal.

Als uiteindelijk resultaat van dit deel van het onderzoek zal uit onder andere MRI-afbeeldingen een inschatting kunnen worden gemaakt van bij welke druk plaque kapot gaat.

Eventueel zullen er ook berekeningen worden gedaan voor de bepaling van de druk in bloedvaten. Deze druk hangt onder andere af van de geometrie van de bloedvaten.