Ontwikkeling van preset-detectie voor Fratoria DICOM-RT Studio

Afstudeerverslag

DE HAAGSE HOGESCHOOL

Naam: Fabian Schneider
Stagebedrijf: Fratoria BV, Rotterdam
Datum: 05-10-2014
Plaats: Den Haag

Inhoudsopgave

Voorwoord

Dit verslag is het eindverslag naar aanleiding van mijn afstudeerstage bij Fratoria BV tussen 1-4-2014 tot en met 6-10-2014. In het verslag wordt het ontwerpen en ontwikkelen van een module voor preset-detectie voor Fratoria DICOM-RT Studio besproken.

Lezers die vooral geïnteresseerd zijn in het proces en de planning worden verwezen naar hoofdstuk 4.

Alle werkzaamheden worden besproken in hoofdstuk 5.

Ik wil Frank Gescher en Iaroslav Smychliaev graag bedanken voor de stageplek die zij mij hebben aangeboden. Ook wil ik Maarten van Oosterhout mijn begeleiders van de Haagse Hogeschool, John Smeets en Henk van den Bosch, zeer bedanken voor hun hulp tijdens mijn afstudeer periode.

Samenvatting

Dit verslag is geschreven naar aanleiding van de afstudeeropdracht “Ontwikkeling van preset-detectie voor Fratoria DICOM-RT Studio” die is uitgevoerd bij Fratoria BV. DICOM-RT Studio is een applicatie voor het bekijken van radiotherapie-oncologie behandelplannen in 2D en 3D. Deze behandelplannen bestaan uit CT of MR beelden en worden opgeslagen met behulp van de DICOM-RT standaard. Voor het tonen van objecten wordt gebruik gemaakt van de VTK library.

Het in 3D tonen van behandelplannen brengt enkele problemen met zich mee. Één daarvan is dat CT en MR beelden alleen bestaan uit grijstinten, waardoor het voor de gebruiker lastig is om het verschil te zien tussen de verschillende weefsels. Om het makkelijker te maken het verschil te zien tussen weefsels maakt DICOM-RT Studio gebruik van color mapping om de grijstinten te converteren naar kleur. Zo kan een bepaalde grijswaarde gekleurd worden, waardoor de locatie van al het weefsel met de bijbehorende grijswaarde snel zichtbaar is. Dezelfde techniek wordt ook gebruikt voor ondoorzichtigheid. Zo kunnen bepaalde weefsels compleet doorzichtig worden gemaakt, zodat de focus ligt op de belangrijke weefsels.

De opgeslagen color mapping en ondoorzichtbaarheids instellingen worden binnen DICOM-RT studio presets genoemd. Presets gebruiken vaste grijswaardes om kleuren en ondoorzichtbaarheids waardes aan te koppelen.

Tussen behandelplannen kunnen de grijswaardes echter verschillend zijn. Twee verschillende type scanners zullen niet beschikken over dezelfde range van grijswaardes. Hierdoor zijn presets niet altijd bruikbaar met meerdere datasets.

Voor deze afstudeeropdracht is er gekozen voor Rapid Application Development omdat er zo veel kennis kan worden opgedaan door middel van prototypes. Ook kunnen de verschillende oplossingen zo worden uitgetest terwijl de risico’s worden geminimaliseerd. De risico’s en eisen zijn in kaart gebracht.

Allereerst is er onderzoek gedaan naar de werking van VTK en ITK. Hierna zijn in increment 1 twee prototypes ontwikkeld voor het testen van VTK, ITK en transfer functies. Prototype 1 was een mislukte poging om gelijk binnen DICOM-RT Studio te werken.
Bij prototype 2 is besloten om eerst te beginnen in een los project en deze later te verwerken in DICOM-RT Studio. Na wat ervaring op te hebben gedaan met VTK en ITK zijn er nog twee prototypes ontwikkeld. Prototype 3 werkt op basis van een pieken en dalen algoritme, dat de pieken en dalen binnen een histogram gebruikte als herkenningspunten. Deze herkenningspunten werden gebruikt als basis om de preset op te schalen. Na enkele tests bleek dat deze techniek niet werkte bij MR beelden en te onnauwkeurig voor CT waardoor er een nieuwe oplossing is gezocht.
Prototype 4 werkt op basis van “Otsu’s Method”, een thresholding techniek die de overgang van lucht naar zacht weefsel en van zacht weefsel naar bot kan detecteren in een histogram. Deze locaties op het histogram worden dan gebruikt als herkenningspunten om de preset op te schalen. De techniek bleek na tests zeer succesvol en is in increment 2 ingebouwd in DICOM-RT Studio.

Verklarende woordenlijst

DICOM	Digital Imaging and Communications in Medicine. Een standaard voor het opslaan van medische gegevens waaronder CT en MR beelden.
DICOM-RT	Een extensie van de DICOM standaard die Radiotherapie modules toevoegt waaronder RT image, RT Dose, RT Structure Set, RT Plan en RT Treatment record.
Scalar	Een variabele die de grijswaarde van een punt binnen het beeld bevat. Een DICOM image bestaat uit meerdere scalars. Color mapping en Transfer functions zijn gebaseerd op deze waardes.
ZZP-er	Zelfstandige zonder personeel
IGPS	Image Guided Patient Setup
Weefsel	Specifieke organen, tumoren, botten
Color mapping	Het koppelen van scalar waarde aan kleur.
Transfer Functions	Een functie die scalar waardes linkt aan visuele eigenschappen zoals kleur en ondoorzichtigheid.
Dataset	Een verzameling van DICOM beelden die kan worden geladen in “Fratoria DICOM-RT Studio”
GUI	Graphical User Interface
VTK	The Visualisation Toolkit (http://www.vtk.org/)
ITK	Insight Segmentation and Registration Toolkit (http://itk.org/)
Preset	Verzameling opgeslagen scalar waardes met bijbehorende kleur of ondoorzichtigheidswaarde die kunnen worden gebruikt als input voor Transfer functies.

1. Inleiding

Dit verslag is geschreven naar aanleiding van de afstudeeropdracht bij Fratoria BV. Het beschrijft het gehele afstudeertraject inclusief evaluatie.

Fratoria BV levert een eigen softwarepakket genaamd DICOM-RT Studio, een programma dat het mogelijk maakt om radiotherapie-oncologie behandelplannen in 2D en 3D te visualiseren. Voor de gebruiker is het belangrijk om verschillende weefsels zo goed mogelijk te kunnen onderscheiden. Hiervoor gebruikt Fratoria zogenaamde presets om kleur en ondoorzichtigheid te geven aan bepaalde delen van de scan. Deze presets zijn echter gebaseerd op de grijswaardes waaraan zij kleur of ondoorzichtigheid moeten geven. Omdat grijswaardes kunnen verschillen per CT of MR beeld moeten presets vaak worden aangepast per behandelplan. Dit is een tijdrovend proces en vergt veel kennis van de achterliggende visualisatie techniek.

Dit verslag beschrijft de ontwikkeling van een techniek die herkenningspunten binnen een histogram gebruikt voor het correct schalen van presets, waardoor de gebruiksvriendelijkheid wordt verhoogd en zo tijd bespaart.

Het verslag begint met achtergrondinformatie over Fratoria, DICOM-RT Studio en de gebruikte libraries. Daarna wordt ingegaan op de opdracht, analyse en de verschillende incrementen. Als laatsten volgen de conclusie, aanbevelingen en evaluatie.

2. Achtergrond

2.1 Bedrijfsbeschrijving:

2.1.1 Fratoria

Fratoria BV is een Nederlands particulier bedrijf dat software ontwikkelt voor radiotherapie-oncologie. Eén van de belangrijkste producten van Fratoria is: “Fratoria DICOM-RT Studio”. In het kort is dit een software pakket dat de gebruiker de mogelijkheid biedt om radiotherapie-oncologie behandelplannen in combinatie met CT en MRI beelden in 3D te visualiseren. De eindgebruikers van deze software zijn artsen, natuurkundigen, therapeuten en onafhankelijke software leveranciers. Ook levert Fratoria ondersteuning aan klanten wanneer er vragen of problemen zijn.

Fratoria is officieel gevestigd in Rotterdam. Typerend voor het bedrijf is dat het een “Virtual corporation” is. Dit betekent dat er geen werknemers op de loonlijst staan. De werknemers werken vanuit hun eigen locatie en communicatie verloopt grotendeels elektronisch of telefonisch; via Skype en email. Fysieke ontmoetingen vinden meestal plaats in het geval er software moet worden geïnstalleerd of getest, bij de klant of binnen het bedrijf zelf.

Fratoria is een klein bedrijf. Er zijn gemiddeld zes tot zeven ZZP-ers aan het bedrijf verbonden. Zelfs de directeur is een ZZP-er. Fratoria maakt verder gebruik van stagiaires van de HHS en TU-Delft.

Ondersteuning wordt in de eerste plaats geleverd door Iaroslav Smychliaev (Directeur Fratoria). Waar nodig worden klanten doorgestuurd naar de ontwikkelaar die verantwoordelijk is voor het product. De meeste klanten hebben support-contracten en krijgen een paar keer per jaar updates voor de software.

Het bedrijf is ISO 13485 en ISO 9001 gecertificeerd. Dit betekent dat het bedrijf voldoet aan internationale standaarden op het gebied van kwaliteit.

2.1.2 Geschiedenis

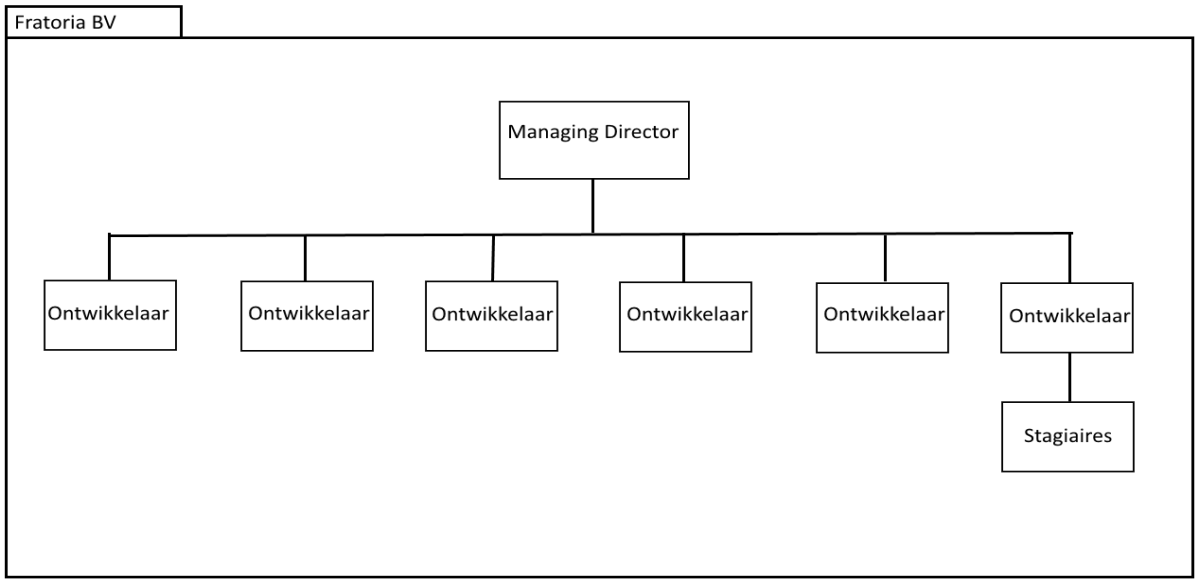
Fratoria is in 2006 opgericht door ca. zes professionals, onder wie een radiotherapeut, klinisch fysicus, investeerder en een computerdeskundige. De laatste is werkzaam als Managing Director.

De oprichters hadden verschillende commerciële ideeën die ze wilden uitvoeren. Gedurende de eerste twee jaar van het bestaan van het bedrijf zijn deze voorstellen gepresenteerd aan bedrijven, zoals Siemens, IBM en verschillende klinieken. Uit deze contacten is de “Fratoria DICOM-RT Studio” voortgekomen.

Daarna zijn er andere projecten van de grond gekomen, zoals Image Guided Patient Setup (IGPS) in samenwerking met Universitair Medisch Centrum Oslo en Elekta. IGPS is een hulpmiddel voor het positioneren van patiënten voor bestraling. Zo kan worden verzekerd dat de patiënt de correcte dosis krijgt op precies de juiste locatie. IGPS is sinds 2012 in gebruik bij verschillende ziekenhuizen in Scandinavië.

2.1.3 Organisatiestructuur

De directeur en medeoprichter van Fratoria is Iaroslav Smychliaev. Het bedrijf kenmerkt zich door een platte organisatie, waarbij ZZP-ers en stagaires worden ingezet.



Afbeelding 1: Organogram Fratoria.

De stagebegeleider vanuit het bedrijf, die tevens ontwikkelaar is, is Maarten van Oosterhout.
Hij is vanaf het eerste moment als ZZP-er betrokken bij Fratoria, soms met pauzes van enkele maanden in verband met andere werkzaamheden.

2.1.4 De markt

De markt, waarin Fratoria zich bevindt, is relatief stabiel. Het gaat om een kleine markt en de concurrentie is erg groot. Ze zullen niet gemakkelijk naar een concurrent overstappen wanneer er eenmaal een leverancier is gekozen. Dit komt ook omdat de klanten over het algemeen een grote binding hebben met de leverancier. Fratoria heeft de ambitie om zijn positie op de thuismarkt te verstevigen en zich verder uit te breiden in Europa.

2.2 DICOM-RT (Digital Imaging and Communications in Medicine)

DICOM-RT is een uitbreiding op de DICOM 3.0 standaard, die ondersteuning bevat voor radiotherapie. De standaard is uitgebracht door de National Electrical Manufacturers Association (NEMA) in 1993 en biedt onder andere ondersteuning voor CR, MR, CT, Ultrasound en Angiographie.

DICOM is een standaard voor het omgaan met, opslaan, printen en verzenden/ontvangen van medische beelden op een netwerk zowel als een file format. Het biedt de mogelijkheid om scanners, servers, desktop pc's, printers en andere hardware te integreren via een PACS (Picture Archiving and Communication System). Beelden opgeslagen binnen PACS zijn op deze manier toegankelijk voor alle andere apparaten binnen het netwerk.

Een DICOM data object bestaat uit meestal twee onderdelen. De "tags" of "header" die het object identificeren en omschrijven, en de pixel data. De pixel data kan bestaan uit meerdere standaards waaronder JPEG, JPEG Lossless, JPEG2000 en Run-length encoding. Meestal is dit echter raw binary data.

2.3 VTK/ITK

The Visualization Toolkit is een open-source, cross-platform software pakket, dat ontwikkeld is door Kitware Inc. VTK is een uitgebreide wrapper voor OpenGL en is ontwikkeld voor beeldverwerking en het tonen van 3D-objecten. Het tonen van objecten binnen VTK werkt met een Visualization pipeline, die bestaat uit meerdere onderdelen (Zie afbeelding 2):

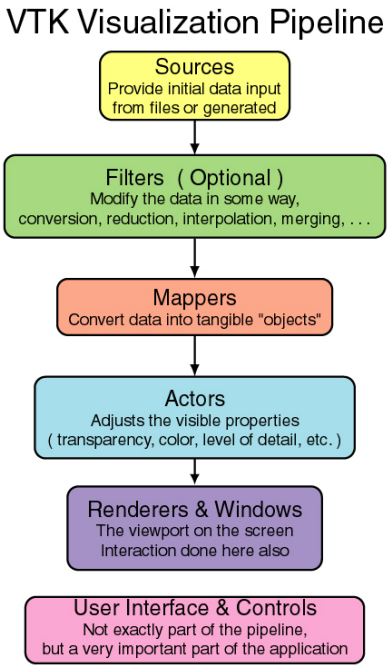
De bron: Dat is in het geval van dit project een set DICOM-RT bestanden.

Filters: Deze bewerken de data die ze ontvangen met bijvoorbeeld een median filter. Hierna sturen ze de data door naar een andere filter of mapper.

Mappers: Deze zetten de ontvangen data om naar 3D objecten, die binnen een renderer kunnen worden geladen.

Actors: Dit zijn de 3D-objecten. Via de actor kunnen onder andere de kleur, ondoorzichtigheid, rotatie en locatie van het object worden aangepast.

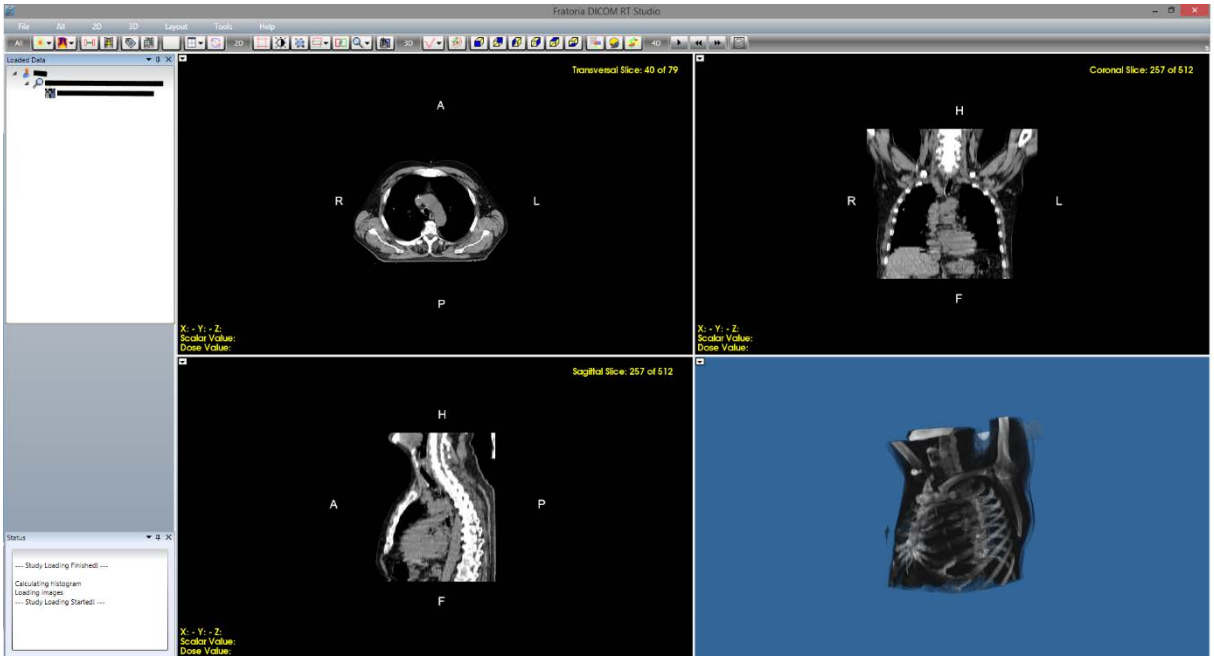
Renderers en windows: Deze tonen de actors daadwerkelijk op het scherm. Ook de besturing wordt hier opgevangen.



Afbeelding 2: VTK Visualization pipeline [9]

Een andere library, die binnen DICOM-RT Studio wordt gebruikt, is ITK. ITK is een ander open-source cross-platform pakket, dat ontwikkeld is door Kitware en Insight software consortium. Het is ontwikkeld voor beeld segmentatie en -registratie en kan gebruikt worden voor bijvoorbeeld het isoleren van objecten binnen een scan.

2.4 Fratoria DICOM-RT Studio



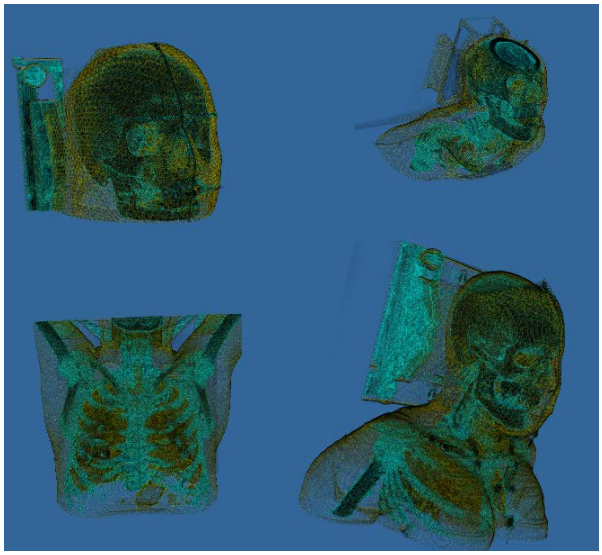
Afbeelding 3: Een screenshot van Fratoria DICOM-RT Studio.

Fratoria DICOM-RT Studio is een applicatie die het voor de gebruiker mogelijk maakt om radiotherapie-oncologie behandelplannen in 3D te bekijken in combinatie met CT of MRI beelden. Een behandelplan, gecombineerd met CT of MRI, wordt in dit verslag een “dataset” genoemd.

Het programma rendert op basis van de 2D CT of MRI beelden een 3D model. Dit 3D model kan met verschillende tools worden bewerkt. Zo bestaat de optie om de kleur van bepaalde scalar ranges aan te passen.

Scalar is de naam die wordt gebruikt voor een punt binnen een gerenderde dataset. Dit punt heeft een waarde die gelijk is aan de grijswaarde van dat punt binnen het object. Deze scalar-waardes kunnen door middel van transferfuncties worden gekoppeld aan kleuren, waardoor er een gekleurd 3D object ontstaat. Deze transferfuncties kunnen vervolgens binnen de applicatie worden opgeslagen voor hergebruik. Binnen de applicatie worden dit presets genoemd.

2.5 Datasets



Afbeelding 4: Vier dataset beelden van verschillende onderdelen van het menselijk lichaam

Een dataset bestaat uit de behandelplannen van een persoon gecombineerd met CT of MR beelden. Deze gegevens zijn opgeslagen volgens de DICOM-RT standaard.

Voor deze afstudeerperiode levert Fratoria meerdere datasets. Deze datasets zijn deels geanonimiseerd, hierdoor moet er een NDA worden ondertekend.

Er zullen zowel CT als MR datasets worden geleverd. Dit is belangrijk, omdat beiden zullen moeten werken met het uiteindelijke algoritme. Ook is het belangrijk om scans te hebben van verschillende onderdelen van het lichaam. Hiermee kan het algoritme optimaal worden getest.

3. Opdracht

3.1 Probleemstelling

Het bekijken van medische 3D-beelden is een krachtige visualisatietechniek, die gebruikt wordt in bijvoorbeeld moderne radio- en oncologische applicaties, waarbij specifieke technieken worden toegepast om verschillende soorten weefsel, zoals bloedvaten, spieren en bot, te tonen.

Bestaande applicaties voor radiotherapie hebben echter erg gelimiteerde 3D rendering-mogelijkheden. Ze focussen zich grotendeels op 2D beelden, terwijl er ook wat 3D oppervlakte rendering, zoals de huid van de patiënt worden gedaan. 3D visualisatie kan aanzienlijk worden verbeterd om artsen meer inzicht te geven in de anatomie van de patiënt.

Fratoria DICOM-RT Studio gebruikt voor het tonen van verschillende weefsels “presets”. Dit zijn opgeslagen transferfunctiepunten die de kleur en ondoorzichtigheid bepalen van de getoonde dataset. Deze presets moeten echter per dataset handmatig worden aangemaakt. Het aanmaken van deze presets is vaak tijdrovend en vereist veel technisch inzicht in de visualisatie techniek.

3.2 Opdrachtomschrijving

De opdracht is tweedelig. De eerste opdracht is het ontwikkelen van een methode die presets automatisch aanpast, waardoor deze niet meer per dataset hoeven te worden aangemaakt. Dit ontwerp moet vervolgens worden uitgewerkt en geïmplementeerd in DICOM-RT Studio.

De tweede opdracht is het optimaliseren van de techniek die Fratoria gebruikt voor het renderen van merged volumes. Aan het eind van de eerste opdracht was er echter vanwege de moeilijkheidsgraad van de eerste opdracht geen tijd meer over om de tweede opdracht te voltooien.

3.3 Risico's

Als een van de eerste stappen moet er een risico-analyse worden gemaakt. Deze bevat de mogelijke scenario's waarin het project zou kunnen falen.

Nr	Risico	Kans van optreden
1	Te weinig tijd voor het voltooien van de opdracht	-/+
2	Te weinig kennis aanwezig voor voltooiing	-
3	Algoritme blijkt niet functioneel na implementatie	-

3.4 Op te leveren producten

Voortgaand aan deze afstudeerstage is er een plan van aanpak geleverd. Hierin staat de werkwijze beschreven, als ook de technieken en methoden die gebruikt zijn en een requirements analyse. Verder zullen alle ontwerpen die tijdens deze periode zijn gemaakt en de broncode worden opgeleverd. Tot slot zal er een handleiding worden opgeleverd die de aanpassingen aan DICOM-RT Studio beschrijft.

3.5 Gemaakte afspraken

3.5.1 Communicatie

De communicatie met Fratoria zal grotendeels elektronisch verlopen. Voor vragenlijsten of het opleveren van documentatie gaat email gebruikt worden. Voor korte vragen of feedback is Skype beschikbaar.

Contact met school gaat via de mail of telefonisch. Vaste contactmomenten worden bevestigd via de mail, waarna vergaderingen en besprekingen op school zullen plaatsvinden.

3.6 Werkwijze

Om ervoor te zorgen dat het project goed wordt uitgevoerd, is het belangrijk om een duidelijke ontwikkelmethode te kiezen. Dit zorgt voor een goede structuur en maakt het project overzichtelijk. Binnen Fratoria wordt er geen vaste ontwikkelmethode gebruikt. Hierdoor kan er een ontwikkelmethode kan worden gekozen die goed bij het project past.

Door het gebrek aan kennis van VTK en ITK aan het begin van deze afstudeeropdracht is er gekozen voor Rapid Application Development (RAD). Zo kunnen de verschillende onderdelen van de VTK pipeline zowel als de algoritmes afzonderlijk worden ontwikkeld en getest. De kennis die wordt vergaard tijdens het ontwikkelen van deze onderdelen kan vervolgens worden toegepast op het eindproduct.

Aangezien het belangrijkste deel van dit project het ontwikkelen van een algoritme is, wordt deze ontwikkeld in een apart prototype. Dit prototype zal dan dienen als *proof of concept* voor het uiteindelijke product. Het tweede increment zal dienen voor de implementatie van het algoritme binnen Fratoria DICOM-RT Studio.

De risico's kunnen aanzienlijk worden verlaagd door de meest risicovolle onderdelen onder te brengen in verschillende prototypes. Dit zorgt ervoor dat de ervaring die wordt opgedaan tijdens de ontwikkeling optimaal wordt benut.

3.7 Planning

3.7.1 Initiële planning

In bijlage 1 is de initiële planning te zien. Deze bevat geen data omdat het erg moeilijk was in te schatten hoelang elke fase zou duren.

Na het schrijven van het plan van aanpak zal er onderzoek worden gedaan naar de werking van VTK en ITK. Hierna zal er worden gekeken naar de code en user interface van de huidige versie van Fratoria DICOM-RT Studio. Hierbij zal vooral worden gelet op de werking van presets en het gebruik van transferfuncties.

De rest van de opdracht is ingedeeld in twee incrementen. In het eerste increment zullen meerdere prototypes worden ontwikkeld om de verschillende elementen te testen. Elk prototype zal de volgende fases doorlopen: Analyse, ontwerp, implementatie en testen. In increment 2 vindt het ontwerp, de ontwikkeling en het testen van de definitieve versie plaats.

Het verloop van de tweede opdracht is identiek.

3.7.2 Uiteindelijke verloop

Na het derde prototype werd het duidelijk dat er meer tijd ging zitten in de eerste opdracht dan origineel was voorzien. De oorzaak hiervoor was de tijd die het kostte om bekend te worden met VTK, zowel als het onderzoek en de implementatie van de verschillende technieken. Hierdoor werd ervoor gekozen de tweede opdracht te schrappen. Dit zorgde voor meer tijd in increment 2.

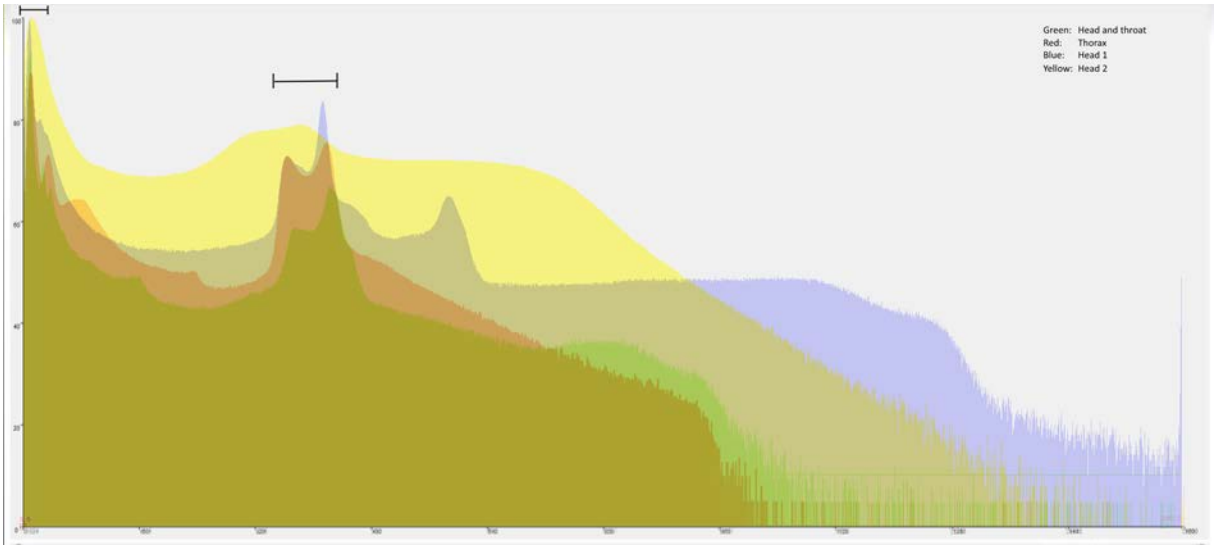
De nieuwe planning is te zien in bijlage 2.

4. Analyse oplossingen

4.1 Oplossingen

Om specifieke weefsels binnen een dataset te tonen moeten deze automatisch worden herkend en op het scherm worden getoond. Na het lezen van 'The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics 3rd Edition.' [3], online onderzoek en gesprekken met Maarten van Oosterhout zijn er vier oplossingen overgebleven. De drie oplossingen die uiteindelijk zijn weggevallen worden besproken in dit hoofdstuk.

4.1.1 Pieken en dalen



Afbeelding 5: Vier overlappende histogrammen

Op afbeelding 5 zijn vier overlappende histogrammen te zien, zoals die worden gebruikt binnen DICOM-RT Studio voor het instellen van transfer functies (6.2.2). Deze histogrammen zijn van verschillende datasets, waaronder die van een thorax, twee schedels en een thorax met schedel.

Zoals te zien komen de scans op verschillende punten overeen, aangegeven door de zwarte lijnen. Na verder onderzoek komen deze locaties overeen met de dichtheid van lucht en zacht weefsel (spier, organen).
Het idee achter de pieken en dalen oplossing is om deze overeenkomende punten automatisch op te sporen en te gebruiken als herkenningspunten. Op basis van deze herkenningspunten kan dan een preset worden geschaald naar de juiste positie.

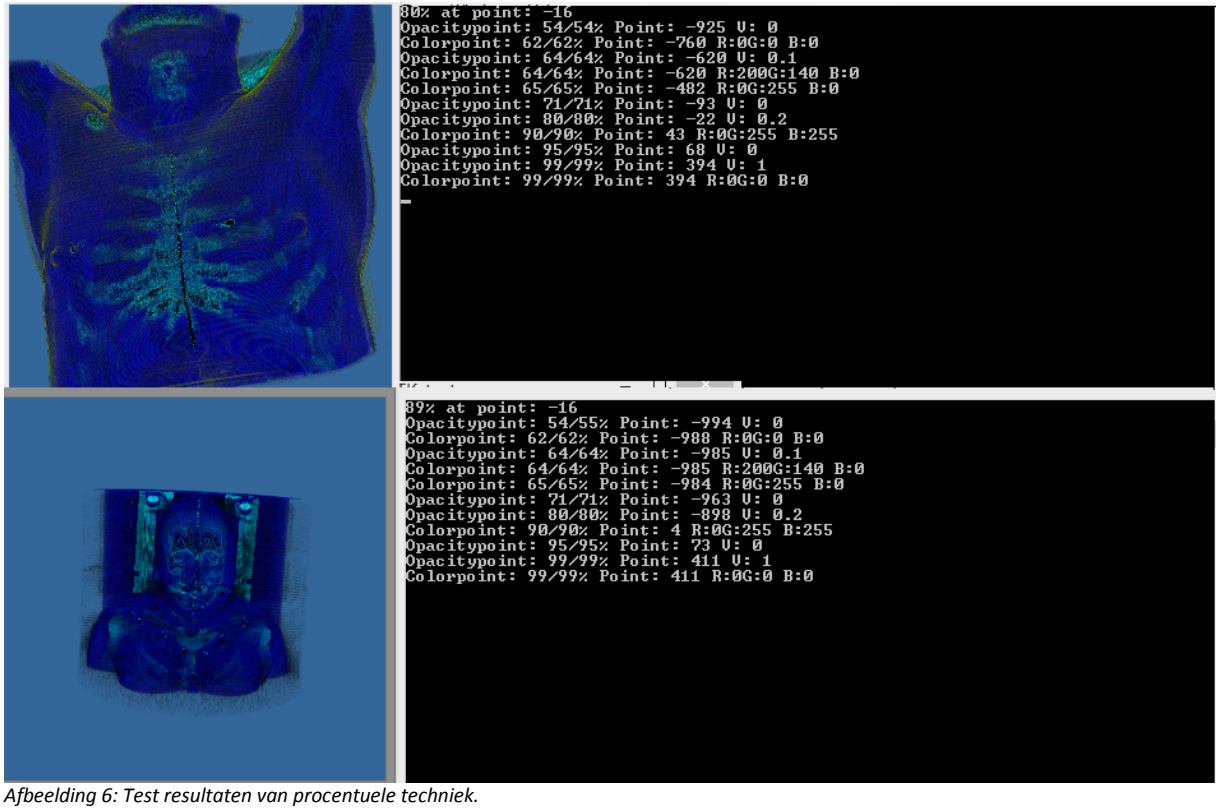
De pieken- en dalenoplossing is getest in prototype 3. Het bleek dat deze techniek niet nauwkeurig genoeg werkte en te vaak foute resultaten gaf. Tevens bleek de oplossing niet te werken met MR datasets.

4.1.2 Region-growing

Het idee achter de region-growingoplossing is dat er automatisch een scalar-punt binnen de dataset wordt geselecteerd. Er wordt dan naar de scalar punten om dit punt heen gekeken om te bepalen of deze bij hetzelfde object horen. Dit gaat door totdat het hele object is geselecteerd. Dit object zou dan als zijn eigen actor in kleur worden getoond op het scherm.
Na een gesprek met Maarten van Oosterhout bleek dit geen geschikt idee, omdat de scalar-waardes op de grens van weefsels te dicht bij elkaar liggen. Dit zorgt ervoor dat het algoritme de grenzen niet detecteert, waardoor het grotere gebieden selecteert dan de bedoeling is. Ook is region-growing een techniek die wordt gebruikt voor segmentatie, waardoor deze niet geschikt is voor deze opdracht.

4.1.3 Procentueel

De procentuele oplossing is gebaseerd op de hypothese dat de procentuele hoeveelheid scalarpunten, die een weefsel of stof binnen een dataset opneemt, hetzelfde is. Zo zou lucht de eerste 50% van scalarpunten op zich nemen. Na een paar kleine aanpassingen aan prototype 3 kon deze theorie snel getest worden.



Afbeelding 6: Test resultaten van procentuele techniek.

Op afbeelding 6 is een test te zien van deze techniek. De “54/55%” geeft aan op welk punt de color of opacity point moet worden aangemaakt, en waar deze uiteindelijk is aangemaakt. De waarde representeert de hoeveelheid scalar punten die lager zijn dan de preset scalar waarde.
Zoals te zien bij punt “62/62%” verschillen de scalar waardes (Point -760 en Point: -988) meer dan 200 punten. Dit is zeer onnauwkeurig; een normaal verschil bij deze datasets zou rond de 10 punten moeten liggen. Hierdoor was snel geconcludeerd dat deze techniek niet zou werken.

4.2 Gekozen oplossing: Otsu

Na het testen van prototype 3 en de pieken- en dalenoplossing is er opnieuw onderzoek gedaan naar verschillende herkenningsmethoden. Eén van de hierbij gevonden methoden is een thresholdingtechniek, genaamd de Otsu-methode.

Afbeelding 7: Links een zwartwit afbeelding, rechts dezelfde afbeelding waar de otsu formule op gedraaid is. [8]

De otsu methode kan het perfecte punt vinden om een afbeelding op te splitsen. Op afbeelding 7 is een afbeelding te zien met een enkele threshold die het plaatje opdeelt in zwart en wit. De threshold die gebruikt is, is gegenereerd door de otsu methode.

De hypothese achter deze oplossing is dat de otsu formule, indien uitgevoerd op een dataset, meerdere thresholds zou kunnen genereren. Deze thresholds worden dan geplaatst op de optimale splitsingspunten, wat in het geval van een dataset zou moeten inhouden dat deze worden geplaatst op de punten waar lucht overgaat in huid, en waar huid overgaat in bot.

De thresholds die gegenereerd worden door de otsu-formule kunnen dan worden gebruikt als herkenningspunten voor de presets. Ook zouden de punten zelf als preset kunnen werken door de punten direct in een transferfunctie te zetten. Zo kunnen huid, bot en lucht worden geselecteerd. Er kan dan nog een otsu-berekening worden uitgevoerd op het geselecteerde weefsel, waarna deze weer wordt opgedeeld in thresholds. Deze techniek is gebruikt in prototype 4 en de uiteindelijke implementatie.

Een gedetailleerde uitleg van het onderliggende algoritme kan worden gevonden in het artikel van Dr. Andrew Greensted [6].

5. Increment 1

5.1 Requirements analyse

Tijdens de requirementsfase worden de door de klant gestelde eisen achterhaald en verdeeld in functionele en niet-functionele eisen.

5.1.1 Functionele requirements

Voor de prototypes van increment 1 zijn de volgende functionele eisen opgesteld.

- DICOM Data laden
- 3D scan tonen
- Transfer functies gebruiken voor tonen van actor
- Histogram berekenen
- Vinden van herkenningspunten in histogram
- Preset baseren op herkenningspunten

De eerste drie eisen zullen vooral om ervaring op te doen met VTK en het werken met DICOM data, deze zijn voortgekomen uit features van de huidige DICOM-RT Studio.
De laatste drie punten zijn voortgekomen uit mogelijke oplossingen voor de opdracht.

5.1.2 Niet-Functionele requirements

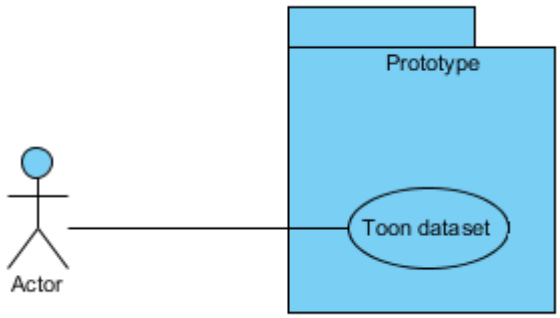
Tijdens de gesprekken met de opdrachtgever zijn er ook enkele niet-functionele eisen naar voren gekomen.

- Prototype in C++
- Minimale laadtijd

Aangezien de achterliggende ActiveX code van DICOM-RT Studio is geschreven in C++ is het slim om de prototypes in dezelfde taal te schrijven. Zo kan de code direct worden overgezet wanneer deze goed blijkt te zijn.

Voor de gebruiksvriendelijkheid en omdat berekeningen snel veel tijd kunnen kosten, is het belangrijk dat er een minimale laadtijd in gedachten wordt gehouden.

5.1.3 Use cases



Afbeelding 8: Use case prototypes.

De use case voor de prototypes is vrij simpel (afbeelding 8). De gebruiker kan namelijk alleen de dataset laten tonen. Bij de start wordt automatisch een dataset geladen, waarna er een preset op wordt toepast.

Ter verduidelijking van de use case is het volgende scenario gedefinieerd:

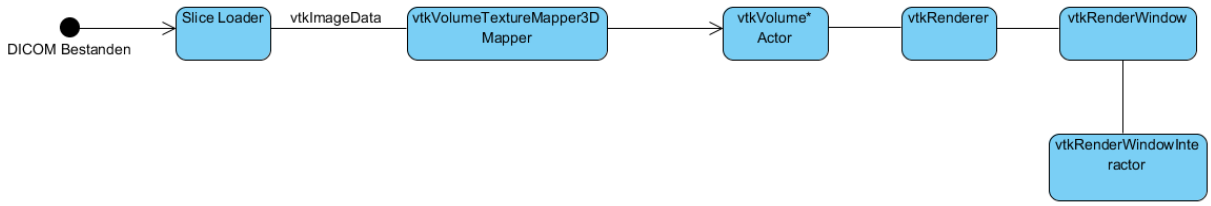
Omschrijving	De dataset kan in 3D worden getoond.
Preconditie	Er is nog geen dataset geladen
Stappen	- DICOM datasets worden ingeladen naar vtkImageData - Histogram van de vtkImageData wordt berekend - Herkenningspunten worden berekend op basis van het histogram - vtkImageData wordt omgezet tot vtkVolume - Preset wordt geladen - vtkVolume wordt getoond door vtkRenderer
Postconditie	3D object getoond op het scherm.

5.2 Analyse

In deze fase wordt bepaald wat de volgende onderzoeksvraag is. Aan het begin van elk prototype wordt gekeken welke vraag ermee moet worden beantwoord. De functionele eisen die moeten worden getest word hier ook bepaald. Zo is er een duidelijk doel voor ogen. Er wordt ook gekeken naar de vorige prototypes en welke leermomenten daaruit voort kwamen.

Prototype	Onderzoeksvraag	Functionele Eisen
1	Hoe werkt een VTK Pipeline?	- DICOM Data laden - 3D scan tonen
2	Hoe werken transfer functions en color mapping?	- Transfer functies gebruiken voor tonen van actor
3	Hoe werkt een algoritme gebaseerd op pieken en dalen binnen een histogram?	- Histogram berekenen - Vinden van herkenningspunten in histogram - Preset baseren op herkenningspunten
4	Werkt een algoritme gebaseerd op de Otsu-formule beter?	- Histogram berekenen - Vinden van herkenningspunten in histogram - Preset baseren op herkenningspunten

5.2.1 VTK pipeline

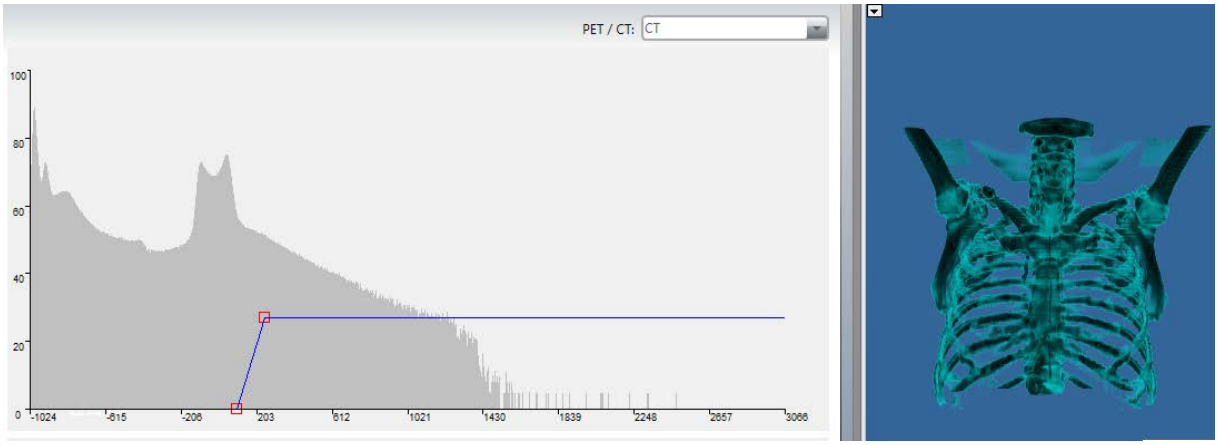


Afbeelding 9: VTK-Pipeline binnen DICOM-RT Studio

De VTK pipeline begint bij de DICOM bestanden. Deze worden geladen door de slice loader. Dit is een functie die de bestanden omzet in een vtkImageData object. Dit vtkImageData object bevat scalars en vectors voor de verschillende punten in de dataset. Om deze data om te zetten naar een object die de renderer kan tonen, wordt gebruik gemaakt van een vtkVolumeTextureMapper3D.

Deze mapper produceert aan de hand van de vtkImageData een vtkVolume. Deze kan dan samen met transferfuncties en color mapping worden aangepast, waarna de vtkRenderer en vtkRenderWindow het resultaat tonen op het scherm.

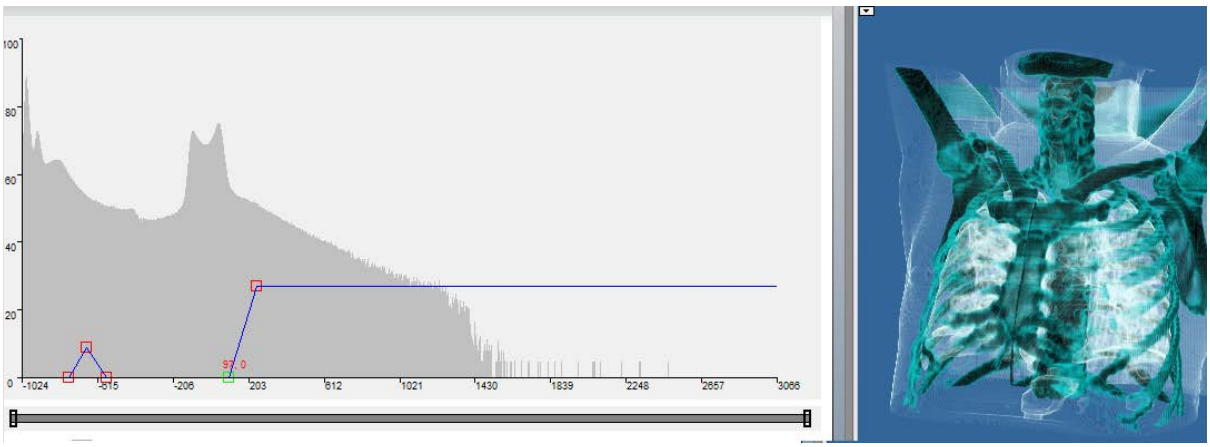
5.2.2 Transfer functions en color mapping



Afbeelding 10: Screenshot van de DICOM-RT ondoorzichtbaarheids instellingen.

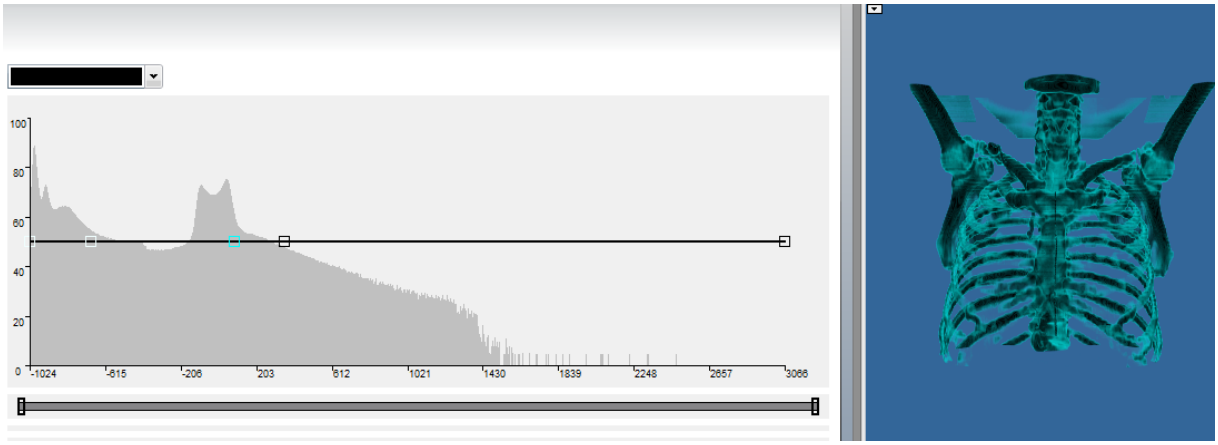
Transferfuncties werken door middel van punten. Op afbeelding 10 is een voorbeeld zichtbaar van een transferfunctie voor ondoorzichtigheid die wordt uitgevoerd op een dataset. Links is het histogram te zien van de rechter dataset. Het histogram bestaat uit twee assen. De X-as geeft de scalar waarde weer. En de Y-as geeft de mate van ondoorzichtigheid van dat punt aan.

De transferfunctie die hier te zien is bevat 2 punten. Punt 1 op scalar waarde 97 met als ondoorzichtbaarheidswaarde 0 en punt 2 met scalar waarde 289 en ondoorzichtbaarheidswaarde 52. De transfer functie zorgt dat de scalars de correcte ondoorzichtbaarheidswaarde krijgen. De punten tussen deze waardes in krijgen bijbehorende waardes. De twee punten zorgen er zo samen voor dat alleen objecten met de dikte van die rond bot worden getoond.



Afbeelding 11: Screenshot van de DICOM-RT ondoorzichtbaarheids instellingen.

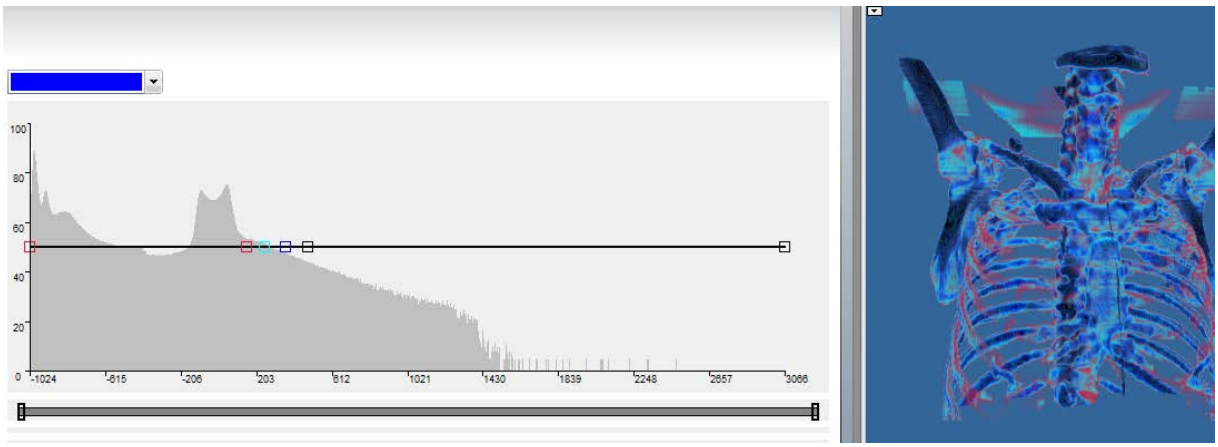
Met de toevoeging van drie punten kan de transferfunctie zo worden aangepast dat ook huid en zacht weefsel worden getoond.



Afbeelding 12: Screenshot van de DICOM-RT kleur instellingen.

Dezelfde transferfuncties worden ook gebruikt voor color mapping. Op afbeelding 12 is de color mapping transferfunctie te zien, die gebruikt is op de eerdere dataset. Het histogram is hetzelfde, echter in plaats van punten met ondoorzichtbaarheidswaardes staan er punten met RGB waardes. Deze dataset bevat drie kleuren: wit, blauw en zwart. De transferfunctie koppelt deze net als voorheen aan de bijbehorende scalar waardes.

Ook zorgt deze ervoor dat scalar waardes tussen twee kleuren in de juiste kleur krijgen.



Afbeelding 13: Screenshot van de DICOM-RT kleur instellingen.

Wanneer er meerdere kleuren dicht bij elkaar worden gezet, wordt het verschil in dikte veel duidelijker dan voorheen.

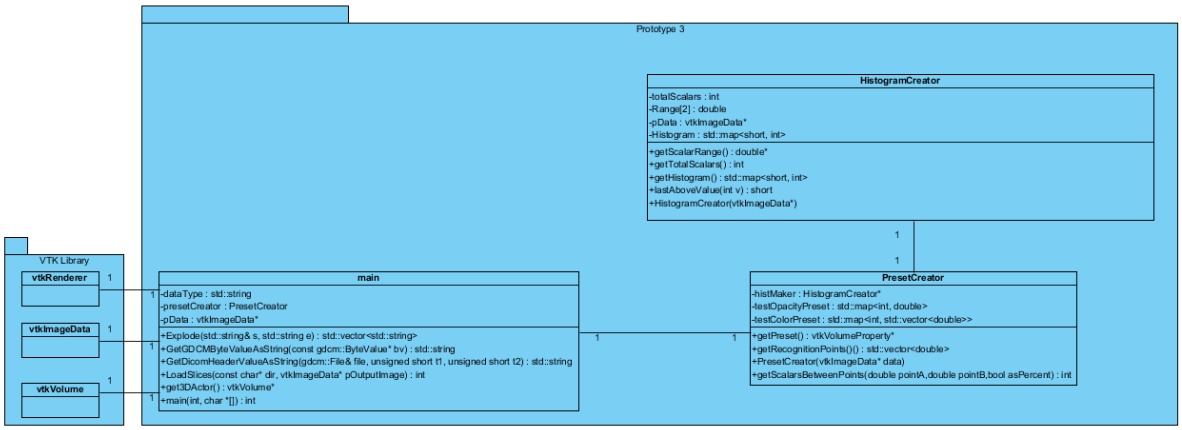
5.3 Ontwerp

De ontwerpen van prototypes 3 en 4 zijn hier terug te vinden. De diagrammen bevatten drie klassen: main, HistogramCreator en PresetCreator. Hieronder een lijstje van de klassen met de bijbehorende taken uit het scenario op 6.1.3. De klassendiagrammen van prototype 1 en 2 zijn niet toegevoegd omdat deze gelijk zijn aan main.

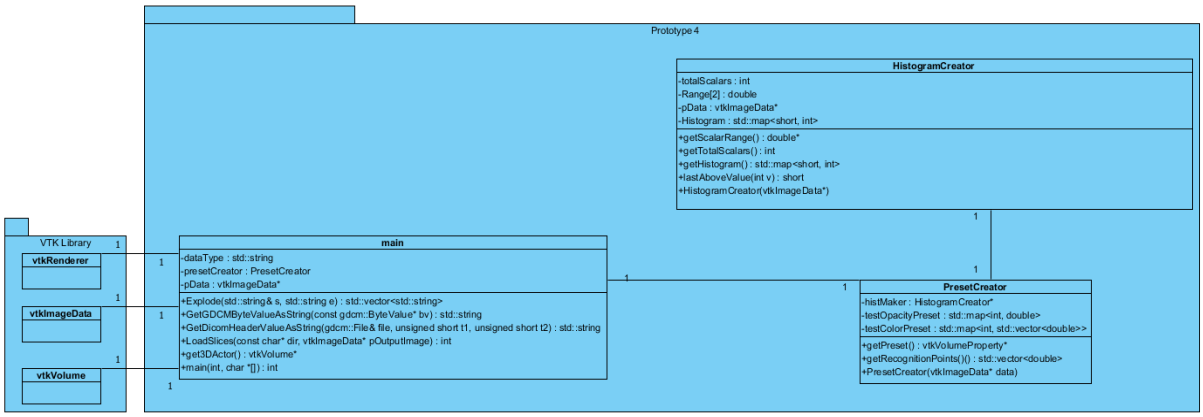
Main	- DICOM dataset worden ingeladen - 3D object wordt getoond
HistogramCreator	- Histogram van de dataset wordt berekend
PresetCreator	- Herkenningspunten worden berekend op basis van het histogram. - Preset wordt geladen

Omdat het laden van de dataset zowel als het tonen van het 3D object in het eindproduct worden afgehandeld door DICOM-RT Studio, worden deze verwerkt in de main klasse van het prototype.

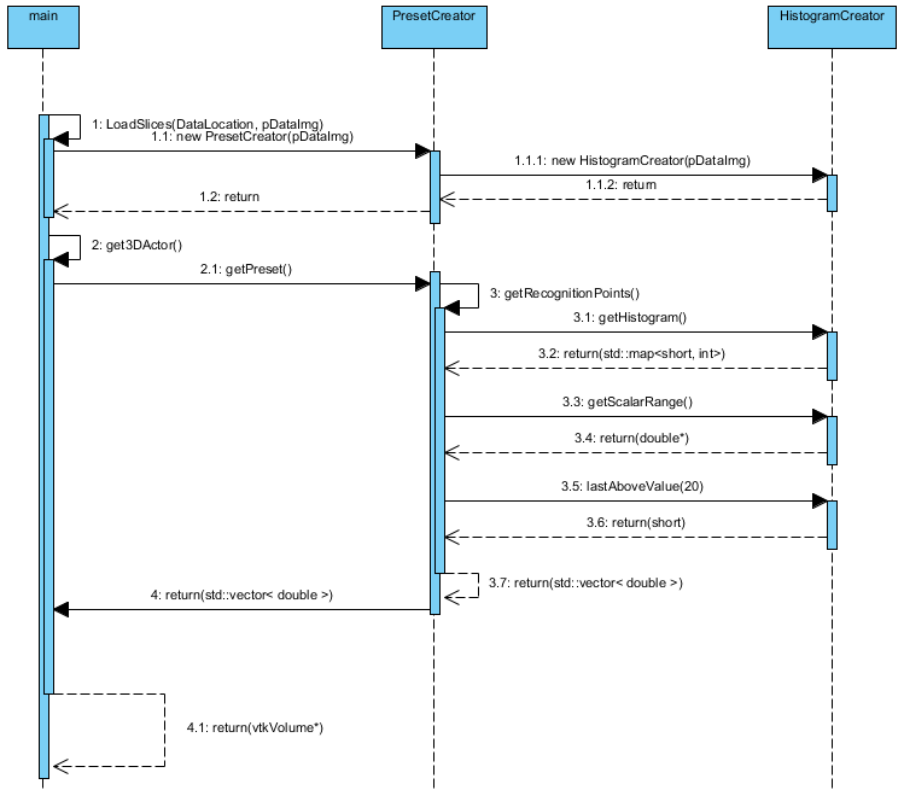
De PresetCreator zal alle berekeningen doen voor de herkenningspunten, zowel als het maken van de preset. De HistogramCreator berekent een histogram gebaseerd op de dataset.



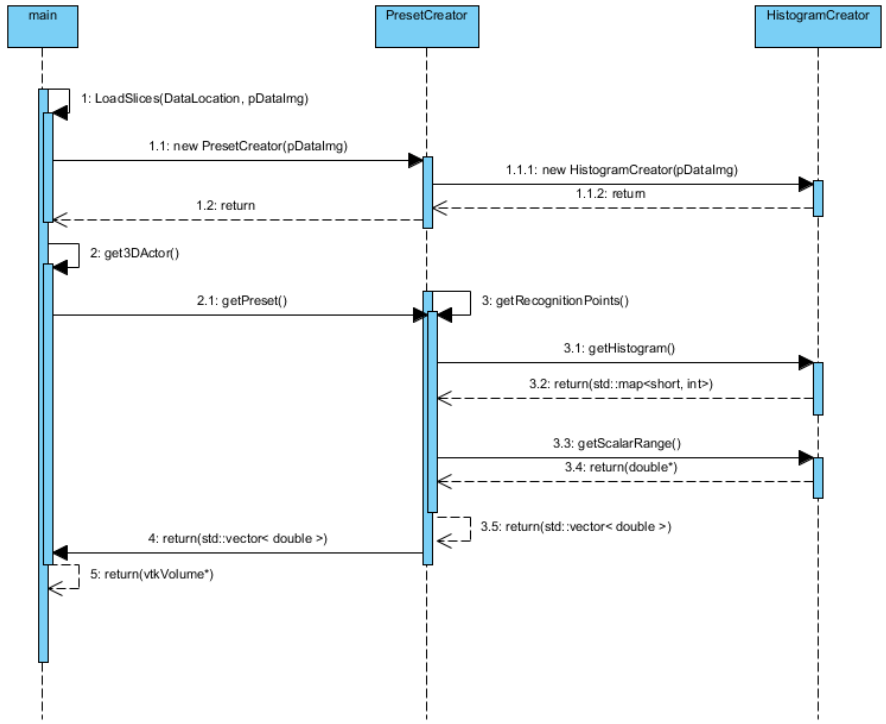
Afbeelding 14: Klassendiagram Prototype 3.



Afbeelding 15: Klassendiagram Prototype 4



Afbeelding 16: Sequentiediagram Prototype 3



Afbeelding 17: Sequentiediagram Prototype 4.

5.4 Implementatie

Tijdens de implementatie fase worden de gemaakte ontwerpen omgezet in code. Hiervoor is gebruik gemaakt van Visual Studio 2008 Professional.

De eerste functies die werden geïmplementeerd, waren die voor het laden van DICOM bestanden en het tonen van 3D objecten. Het laden van DICOM bestanden wordt gedaan door code die al bestond in DICOM-RT Studio.

Hieronder is de code te zien van de get3DActor() functie.

```
vtkVolume* get3DActor(vtkImageData* pData){

//Declaratie
	vtkVolumeTextureMapper3D* Mapper = vtkVolumeTextureMapper3D::New();
	vtkVolume* VolumeActor = vtkVolume::New();

	PresetCreator presetCreator = PresetCreator(pData);

	//link Volume Property aan Actor
	VolumeActor->SetProperty(presetCreator.getPreset());
	//link pDataImg aan de Mapper
	Mapper->SetInput(pData);
	//Link mapper aan actor
	VolumeActor->SetMapper(Mapper);

	return VolumeActor;

}
```

De mapper vtkVolumeTextureMapper3D neemt de vtkImageData en zet deze om naar een vtkVolume. Dit volume krijgt via setProperty en presetCreator.getPreset() de color mapping en ondoorzichtbaarheidswaardes van de juiste preset toegekend. Om de juiste color mapping en ondoorzichtbaarheidswaardes te bepalen moet eerst een histogram worden berekend. Dit wordt gedaan door de HistogramCreator binnen de constructor.

```
HistogramCreator::HistogramCreator(vtkImageData* data){
              Histogram = new std::map<short, int>;
              pData=data;

              Range[0]=pData->GetScalarRange()[0];
              Range[1]=pData->GetScalarRange()[1];

              int maxX=data->GetDimensions()[0];
              int maxY=data->GetDimensions()[1];
              int maxZ=data->GetDimensions()[2];

              totalScalars=maxX*maxY*maxZ;

              for(int i=0; i<maxZ; i++){
                     for(int j=0; j<maxY; j++){
                            for(int k=0; k<maxX; k++){
short* scalarvalue=static_cast<short*>(data->GetScalarPointer(k,j,i));
                                          (*Histogram)[scalarvalue[0]]++;
                            };};};};
```

In prototype 3 wordt zodra de HistogramCreator is aangemaakt er een std::map<short,int > aangemaakt. Deze zal dienen als histogram. Hierna wordt het totaal aantal scalars berekend door middel van (X-as * Y-as * Z-as). Vervolgens wordt elke scalar waarde binnen de vtkImageData doorlopen en wordt de bijbehorende index waarde van de std::map opgeteld.

In prototype 4 wordt dit anders aangepakt. Het histogram wordt nogsteeds aangemaakt binnen HistogramCreator, echter wordt er voor het berekenen van het histogram gebruik gemaakt van een `itk::Statistics::ScalarImageToHistogramGenerator`. Deze histogramgenerator is nodig aangezien de Otsu functie een ITK::HistogramType verwacht als input. Hoewel het mogelijk is om ITK::HistogramTypes handmatig in te vullen was het effectiever om de ITK histogram generatorgenerator te gebruiken.

Voor het gebruiken van de histogramgenerator zijn meerdere typedefs nodig voor het definieren van de verschillende objecten.

```
typedef short InputPixelType;
typedef itk::Image< InputPixelType,  3 >   InputImageType;
typedef itk::Statistics::ScalarImageToHistogramGenerator< InputImageType >
ScalarImageToHistogramGeneratorType;
typedef ScalarImageToHistogramGeneratorType::HistogramType HistogramType;
typedef itk::OtsuMultipleThresholdsCalculator< HistogramType > CalculatorType;
```

De `itk::Statistics::ScalarImageToHistogramGenerator` is een klasse van ITK die gebruik maakt van ITK imagestypes. Aangezien er gebruik wordt gemaakt van VTK voor laden en tonen van de datasets moet er gebruik worden gemaakt van de volgende functies.

```
void ConnectPipelines(VTK_Exporter* exporter, ITK_Importer importer) en void
ConnectPipelines(ITK_Exporter exporter, VTK_Importer* importer)
```

Deze functies zijn geleverd door Fratoria en bieden de mogelijkheid om ITK en VTK pipelines aan elkaar te verbinden.

Na het genereren van het histogram kan de Otsu-berekening worden gedaan. Deze berekening vindt plaats binnen de HistogramCreator klasse en kan aangeroepen worden met de functie getOtsu();

```
CalculatorType::OutputType HistogramCreator::getOtsu(){
	cout << "Start Otsu" <<endl;
	CalculatorType::Pointer calculator = CalculatorType::New();

	calculator->SetNumberOfThresholds(3);

	calculator->SetInputHistogram(histogram->GetOutput());

	try
	{
		calculator->Update();
	}
	catch( itk::ExceptionObject & excp )
	{
		std::cerr << "Exception thrown " << excp << std::endl;
	}

	cout << "Eind Compute" <<endl;

	return  calculator->GetOutput();
};
```

De getOtsu() functie berekent de Otsu thresholds en returned de output. De functie wordt binnen de prototypes aangeroepen door de getRecognitionPoints() functie van de presetCreator, deze wordt zelf aangeroepen door de functie getPreset.

```
std::vector< double > PresetCreator::getRecognitionPoints(){
	std::vector< double > Result;
	CalculatorType::OutputType thresholdVector = histMaker->getOtsu();

	CalculatorType::OutputType::const_iterator iterator =
thresholdVector.begin();

	for(;iterator < thresholdVector.end();iterator++){
		Result.push_back(static_cast<double>((*iterator)));
	}
	cout<<"TestResult: "<<Result[0]<<" "<<Result[2]<<endl;
	return Result;
};
```

Binnen getRecognitionPoints() wordt het resultaat van de Otsu formule doorlopen. De waardes worden dan omgezet naar een std::vector<double>. De herkenningspunten worden binnen getPreset() gebruikt om de scalar waardes te bepalen van de color en opacity punten.

```
vtkVolumeProperty* PresetCreator::getPreset(){
	vtkVolumeProperty* VolumeProp = vtkVolumeProperty::New();
	vtkPiecewiseFunction* Opacity = vtkPiecewiseFunction::New();
	vtkColorTransferFunction* Color = vtkColorTransferFunction::New();

	Color->SetColorSpaceToRGB();
	Color->RemoveAllPoints();
	Opacity->RemoveAllPoints();

	double minRange=histMaker->getScalarRange()[0];
	double maxRange=histMaker->getScalarRange()[1];

	std::vector< double > Points = getRecognitionPoints();
	short pointDifference = std::abs(Points[2]-Points[0])/(short)100;


	for(std::map<int,double>::iterator it = testOpacityPreset.begin() ; it !=
testOpacityPreset.end(); ++it){
		Opacity->AddPoint(Points[0]+(pointDifference*it->first),it->second);

	}

	for(std::map<int,std::vector<double>>::iterator it = testColorPreset.begin()
	; it != testColorPreset.end(); ++it){

	Color->AddRGBPoint(Points[0]+(pointDifference*it->first),it->second[0] /
255.0,it->second[1] / 255.0,it->second[2] / 255.0);

}

	Color->Build();

	VolumeProp->SetScalarOpacity(0,Opacity);
	VolumeProp->SetColor(Color);
	VolumeProp->SetInterpolationTypeToLinear();

	return VolumeProp;
};
```

Nadat de ColorPreset en Opacity zijn aangemaakt worden deze toegevoegd aan VolumeProp. Deze wordt vervolgens binnen de get3DActor functie toegevoegd aan de volume actor.

5.5 Testen

Na de implementatie fase zijn er verschillende tests uitgevoerd om te kijken of het product presteerde zoals mocht worden verwacht. De eis voor de tests is dat de gegenereerde preset zoveel mogelijk visueel lijkt op de preset waarop deze gebaseerd is als mogelijk. In bijlage 11.3 is één test te zien met positieve resultaten. De afbeelding toont een test van prototype 3 waarin herkenningspunten en opacity punten werden vergeleken. De betekenis van de tekst onder de datasets is als volgt:

"Top found at": Een plek waar een piek is gevonden in het histogram.
"Highest points found": De twee hoogst gevonden pieken
"minRange maxRange": De minimum en maximum scalar waarde
"Opacity": De plek waar een opacity punt is aangemaakt gebaseerd op de highest points

Te zien is dat op deze datasets de gevonden herkenningspunten zeer overeen komen. De geteste preset toont huid geel, terwijl het botten licht blauw tot zwart kleurt. Na de positieve resultaten met de eerste test van CT beelden met prototype 3, werd er getest met MR beelden. Prototype 3 werkte echter veel slechter met MR beelden dan met CT, waardoor de scans (bijlage 11.4) er erg verschillend uitzien.

Bijlage 11.5 bevat afbeeldingen van de tests met de otsu methode. De afbeeldingen zijn opgedeeld in Originele preset methode links, en de met de otsu methode berekende preset aan de rechterkant. De bovenste 2 afbeeldingen zijn CT afbeeldingen en de onderste MR. De preset die gebruikt wordt hoort zacht weefsel doorzichtig wit te tonen, terwijl het hardere weefsel, zoals bot, toont in het lichtblauw tot zwart. De preset is origineel gemaakt op een andere dataset dan de vier op de afbeelding. Dit is belangrijk omdat de resultaten van de presets identiek zijn op degene waarop ze zijn aangemaakt.

Zoals te zien in de eerste dataset zorgt de otsu preset voor een betere selectie van botten. Maar het duidelijkste resultaat is te zien bij de MR datasets. Waar eerst bijna de gehele dataset gezien werd als bot, wordt nu zacht weefsel wel herkend.

6. Increment 2

6.1 Requirements analyse:

Tijdens de requirements fase worden de door de klant gestelde eisen achterhaald en verdeeld in functionele en niet-functionele requirements. Bij de requirements van increment 2 is geen rekening meer gehouden met leerdoelen zoals bij increment 1. Door middel van de prototypes en meerdere gesprekken met de opdrachtgever en begeleider zijn enkele nieuwe eisen tot stand gekomen.

6.1.1 Functionele requirements

- Het programma moet met zowel CT als MR werken.
- Het moet nieuwe type preset kunnen laden en opslaan

Tijdens de ontwikkeling van de prototypes werd duidelijk dat er goed gelet moest worden op de compatebiliteit met zowel CT als MR. Dit omdat het scalar type van CT “short” is, terwijl MR “unsigned short” is.

De nieuwe presets gebruiken andere waardes dan de presets die nu worden gebruikt. Hierdoor moet er ruimte worden gemaakt voor deze nieuwe waardes binnen de settings van DICOM-RT Studio.

De eisen van 6.1.1 gelden hier ook.

6.1.2 Niet-Functionele requirements

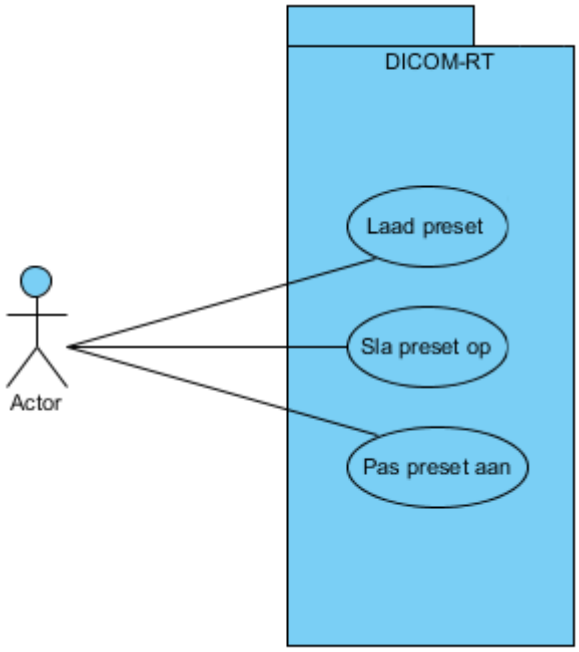
- Onderdeel van DICOM-RT studio:
- VTK/ITK
- User interface in C#, ActiveX in C++.
- Vasthouden aan het format van Fratoria.

Het eindproduct moet deel uitmaken van DICOM-RT studio, eventueel in simpele demonstratievorm. Er zal gebruik worden gemaakt van de VTK en ITK library’s voor het tonen en bewerken van 3D objecten. Voor compatibiliteit zal er gebruik worden gemaakt van VTK versie 5.10 en ITK versie 4.4.0.

De user interface van DICOM-RT Studio is geschreven in C# terwijl de achterliggende ActiveX code in C++ is geschreven. Verder moet het format aansluiten bij die van Fratoria.

De eisen van 6.1.2 gelden hier ook.

6.1.3 Use cases



Afbeelding 18: Usecase uiteindelijke implementatie.

Ter verduidelijking van de use case is het volgende scenario gedefinieerd.

Omschrijving	Preset kan worden ingeladen
Preconditie	Er is een dataset geladen
Stappen	- Presets worden opgehaald uit appSettings (LoadPresets) - Gebruiker navigeert naar 3D-Settings - Gebruiker selecteert een preset (cmbPresetName_SelectionChanged()) - Transfer functies en color mapping presets worden toegepast op actor - Actor wordt gerenderd
Postconditie	Preset is geladen

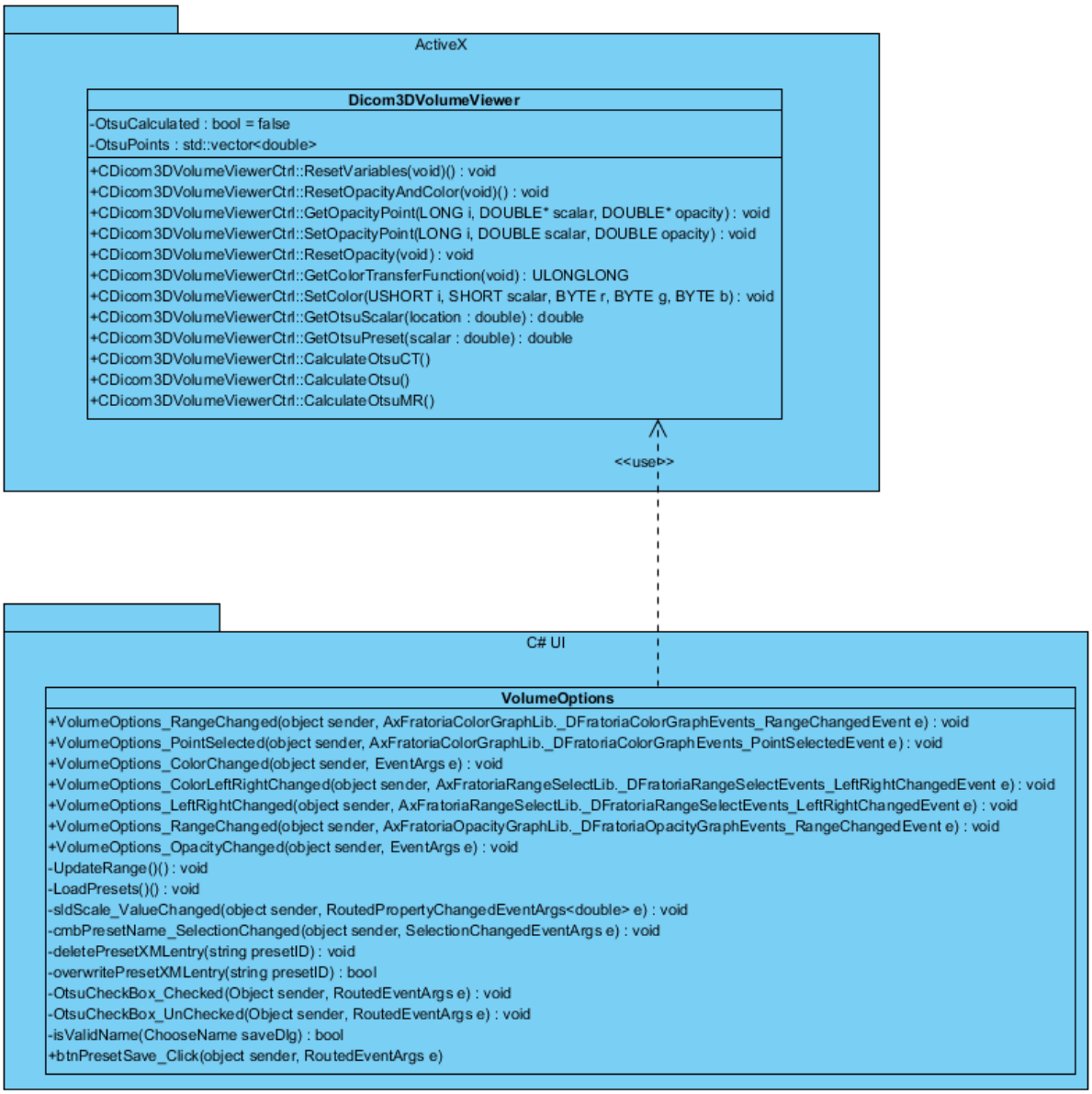
Omschrijving	Preset kan worden opgeslagen
Preconditie	Er is een dataset geladen
Stappen	- Gebruiker gaat naar 3D-Settings - Gebruiker drukt op opslaan (btnPresetSave_Click) - Gebruiker vult een naam in voor de preset (isValidName) - Transfer functie punten worden geschreven naar customConfig.xml
Postconditie	Preset is toegevoegd aan customConfig.xml

Omschrijving	Preset kan worden bewerkt
Preconditie	Er is een dataset geladen
Stappen	- Gebruiker navigeert naar 3D-Settings - Gebruiker past huidige preset punten aan - Gebruiker drukt op opslaan (btnPresetSave_Click) - Gebruiker vult zelfde naam in voor preset (isValidName) - Transfer functie punten worden geschreven naar customConfig.xml
Postconditie	Preset is aangepast en opgeslagen in customConfig.xml

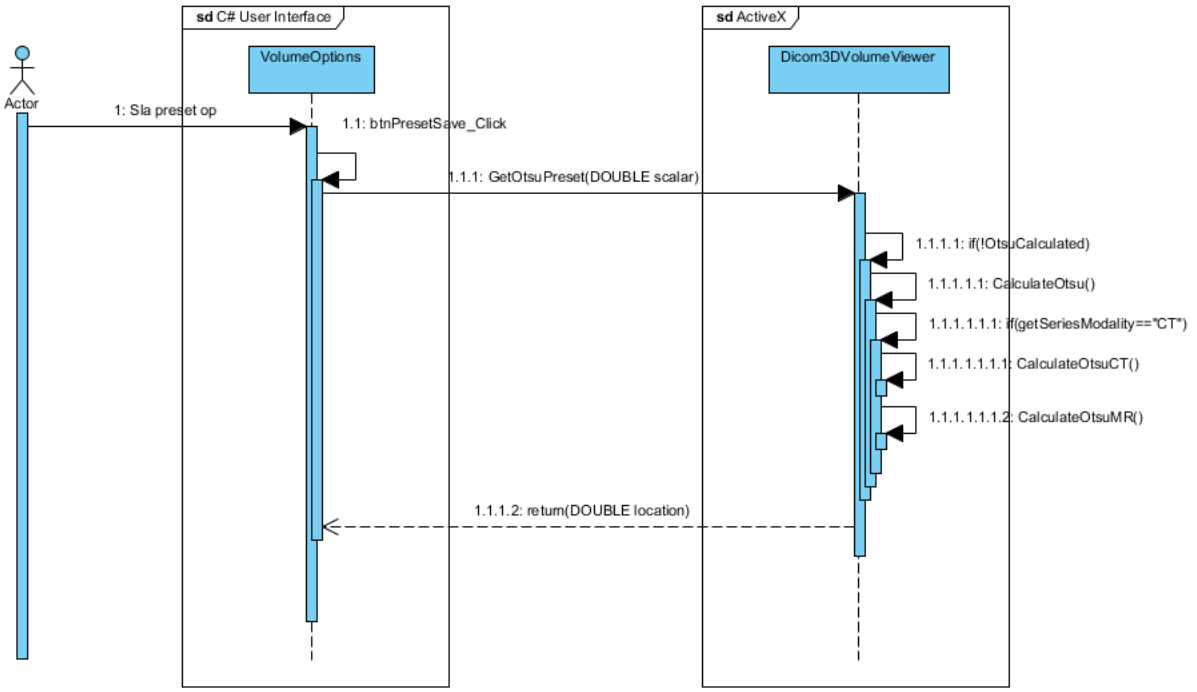
6.2 Ontwerp

Het huidige systeem van Fratoria bestaat uit een C# frontend en een C++ backend. VolumeOptions is de klasse die hoort bij het 3D settings menu binnen Fratoria DICOM-RT Studio. Hierin zal de user interface worden geplaatst.

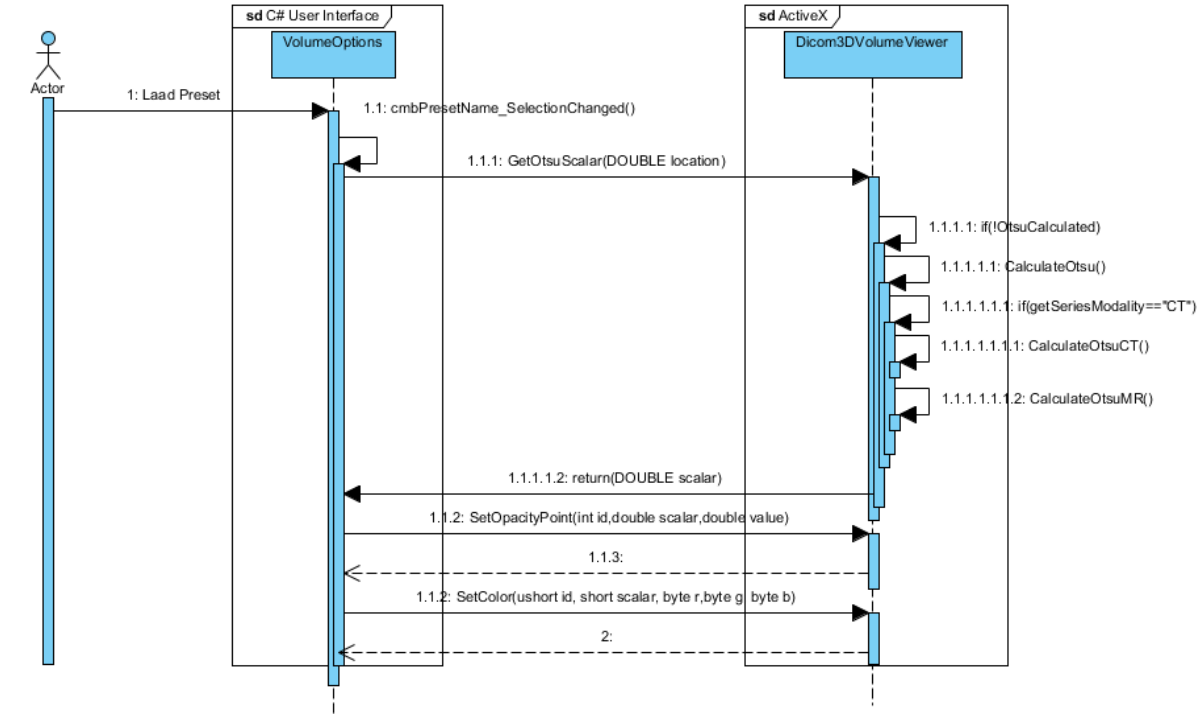
De code voor het backend zal worden geplaatst in Dicom3DVolumeViewer. Dit is de klasse waar vanuit de getoonde 3D volumes kunnen worden bewerkt. Vanuit hier kunnen transferfuncties worden toegepast of kan de camera worden bewogen.



Afbeelding 19: Klassendiagram van de ActiveX en C# User interface componenten.



Afbeelding 20: Sequentie diagram opslaan preset.



Afbeelding 21: Sequentie diagram voor het laden van een preset.

6.3 Implementatie

De code van de uiteindelijke implementatie van increment 2 is grotendeels hetzelfde als die van prototype 4 van increment 1.

Één van de verschillen is dat de klassen PresetCreator en HistogramCreator zijn verdwenen. Vanwege de complexiteit van het toevoegen van klassen in ActiveX is er samen met Maarten besloten om de functies te verwerken in de bestaande klassen VolumeOptions en Dicom3DVolumeViewer.

VolumeOptions is onderdeel van de gebruikers interface. Binnen deze klasse worden de Presets geladen, aangepast en opgeslagen. Alle achterliggende functies zoals setColor en setOpacityPoint voor het instellen van de transfer functies en color mapping zijn verwerkt in Dicom3DVolumeViewer.

Een andere aanpassing is de vervanging van getOtsu() door CalculateOtsuCT() en CalulateOtsuMR(). Vanwege de vele typedefs die nodig zijn voor het berekenen van de otsu thresholds is ervoor gekozen 2 verschillende functies te gebruiken. Zo blijft de code netter en wordt er minder gebruik gemaakt van "If(CT) {}else{}". CalculateOtsuCT en CalculateOtsuMR worden automatisch aangeroepen door CalculateOtsu().

```
void CDicom3DVolumeViewerCtrl::CalculateOtsu(void)
{	AFX_MANAGE_STATE(AfxGetStaticModuleState());

	if(_pDataSet->GetSeriesModality()=="CT"){
		CalculateCT();
	}
	else
	{
		CalculateMR();	}}
```

De scalar waardes en preset waardes worden ook anders berekend dan binnen de prototypes. In plaats van direct in getPreset Volumeproperties te genereren wordt er ingehaakt op de bestaande presets.

Met een checkbox in het menu kan de waarde van UseOtsu op true of false worden gezet. Wanneer een preset wordt geladen en UseOtsu op true staat moet de scalar waarde worden berekend. Deze waarde wordt berekend door middel van de functie:

```
DOUBLE CDicom3DVolumeViewerCtrl::GetOtsuScalar(DOUBLE location)
{
	AFX_MANAGE_STATE(AfxGetStaticModuleState());

	if(!OtsuCalculated){
		CalculateOtsu();
	}
	return (OtsuResult[0]+(OtsuPointDifference*(location)));
}
```

Wanneer een preset moet worden opgeslagen is de preset waarde nodig. Deze wordt berekend door gebruik te maken van de volgende functie:

```
DOUBLE CDicom3DVolumeViewerCtrl::GetOtsuPreset(DOUBLE scalar)
{	AFX_MANAGE_STATE(AfxGetStaticModuleState());

	if(!OtsuCalculated){
		CalculateOtsu();
	}
	return ((scalar-OtsuResult[0])/OtsuPointDifference);}
```

6.4 Testen

De meeste testen zijn uitgevoerd tijdens increment 1. In increment 2 lag de focus bij het testen van het opslaan en laden van presets. Ook moest worden gekeken of het algoritme correct draaide.

Dit is getest door een preset aan te maken en deze toe te passen op verschillende datasets (Bijlage 11.5). Zowel het opslaan als laden van presets verliep zonder problemen.

7. Conclusie en Aanbevelingen

7.1 Conclusie

De afstudeeropdracht is gebaseerd op het feit dat presets maken voor datasets veel kennis vergt van de gebruiker en dat de weergave van de beelden niet duidelijk genoeg is in zwart-wit. Ook was het maken van presets per dataset een tijdrovend process. Het ontwikkelen van een programma dat de presets automatisch berekend zorgt ervoor dat presets niet meer per dataset aangemaakt hoeven te woden.

Uit onderzoek is gebleken dat met behulp van de Otsu-tresholdingtechniek, die de thresholds tussen verschillende weefsels berekent, er een goed onderscheid kan worden gemaakt tussen de verschillende weefsels.

Zo is de gebruiksvriendelijkheid verbeterd terwijl de userinterface niet is veranderd. De uiteindelijke methode is echter niet uitgebreid getest, deze tests zullen nog wel uitgevoerd moeten worden door Fratoria.

7.2 Aanbevelingen

Voor het verbeteren van het eindproduct heb ik de volgende aanbevelingen:

1. Er zou gekeken kunnen worden waar de Otsu-berekening het beste kan worden uitgevoerd. Op dit moment wordt deze uitgevoerd wanneer een preset wordt gekozen en de checkbox voor het gebruik van Otsu aanstaat. Dit zou ook kunnen gebeuren wanneer een dataset wordt geladen. Laden duurt dan langer, maar er is maar één moment waarop dit gebeurt waardoor de totale laadtijd minder is.

2. Ook worden de herkenningspunten opnieuw berekend elke keer als er wordt gewisseld van dataset. Deze zouden beter kunnen worden opgeslagen tot het programma wordt afgesloten. Ook dit verbeterd de laadtijd.

3. Een derde aanbeveling is om de besturing van de camera binnen DICOM RT Studio aan te passen. Deze besturing werkt goed voor een joystick, maar een optie om te wisselen naar besturing voor muis zou prettig zijn. In mijn prototypes gebruikte ik vtkInteractorStyleRubberBand3D voor de besturing. Dit verhoogd gebruikersvriendelijkheid wanneer gebruik word gemaakt van een muis.

4. Aangezien de nieuwe methode niet uitgebreid getest is, zouden er enkele tests kunnen worden uitgevoerd om de daadwerkelijke gebruiksvriendelijkheid en prestatie te testen.

8. Evaluatie

8.1 Zelfreflectie

Tijdens de afstudeerperiode heb ik naar mijn mening de meeste gestelde leerdoelen behaald. Het enige waaraan niet voldaan is, is het ontwerpen van een mens-machine interface (C11). Het bleek voor de gebruiksvriendelijkheid namelijk bevoordelijk om de interface hetzelfde te laten.

Ik ben tevreden over het resultaat. Ik heb veel geleerd over het renderen van beelden met VTK en ITK en ik ben veel beter geworden in het gebruiken van libraries binnen Visual Studio. Naast het gebruik van libraries heb ik ook geleerd te werken met ActiveX controls. Dat had ik voorheen nog nooit gebruikt.

Ook heb ik geleerd dat ik het erg fijn vind om contact te hebben met mensen tijdens het werk.
Tijdens deze stage heb ik volledig vanuit huis gewerkt en verliep het contact met Fratoria via de mail en Skype. Hoewel de vrijheid fijn was, maakte het op afstand werken het wel lastiger om demonstraties te geven, vragen te stellen over code of om uitleg te krijgen. Het afgezonderd werken is niet in alle opzichten ideaal te noemen.

9. Bronvermelding

[1] Elling, Rien. Andeweg, Bas. De Jong, Jaap. Swankhuisen, Christine. (2011) 'Rapportage-techniek: Schrijven voor lezers met weinig tijd 4e Druk.' [Boek] ISBN: 978-90-01-79478-1.

[2] Schroeder, Will. Martin, Ken. Lorensen, Bill. (2002) 'The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics 3rd Edition.' [Boek] ISBN: 1-93093407-6.

[3] Kitware (2012) 'VTK 5.10.0 Documentation' [Documentatie] http://www.vtk.org/doc/release/5.10/html/

[4] Kitware (2013) 'Insight Toolkit 4.4.0' [Documentatie] http://www.itk.org/Doxygen44/html/index.html

[5] Electa (Zonder datum) 'What is DICOM?' [Artikel] http://www.elekta.com/healthcare-professionals/products/elekta-software/what-is-dicom.html.

[6] Greensted, Andrew (17 juni 2010) 'Otsu Thresholding' [Artikel] http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html

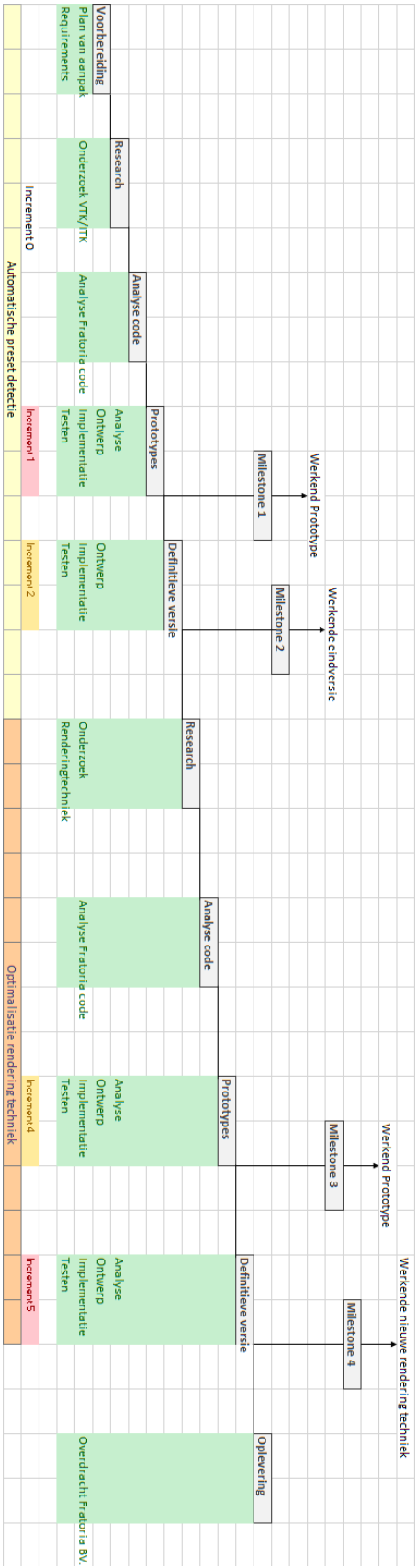
[7] Onbekend (28 september 2014) 'Rapid application development' [Artikel] https://en.wikipedia.org/wiki/Rapid_application_development

[8] Onbekend (3 september 2014) 'Otsu's Method' [Artikel] https://en.wikipedia.org/wiki/Otsu's_method

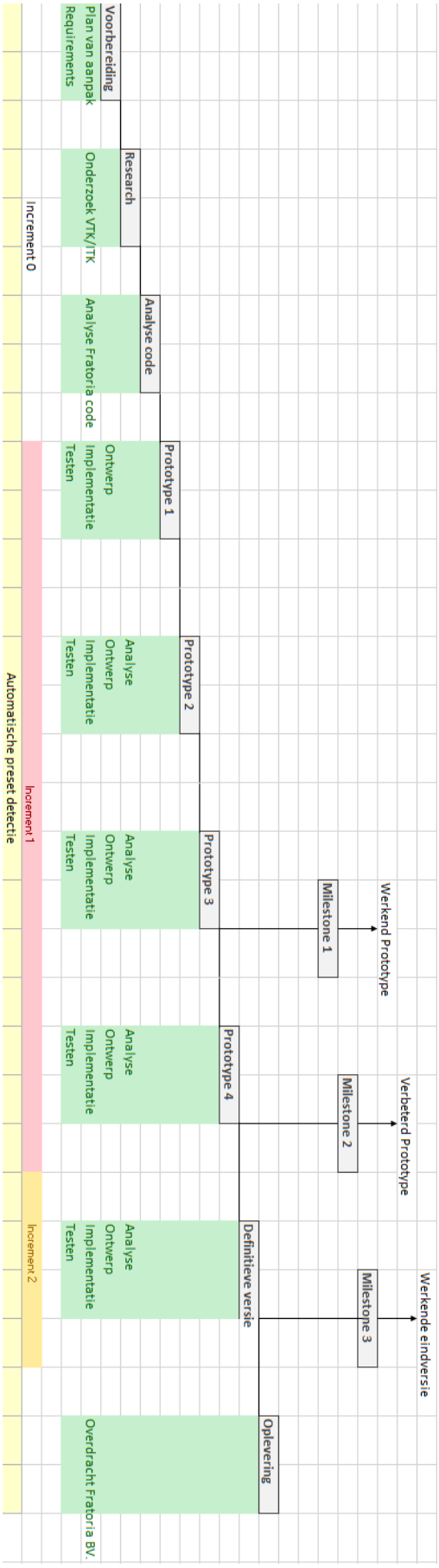
[9] John T. Bell (2004) 'Visualization Toolkit (VTK) Tutorial' [Artikel] http://www.cs.uic.edu/~jbell/CS526/Tutorial/Tutorial.html

10. Bijlagen

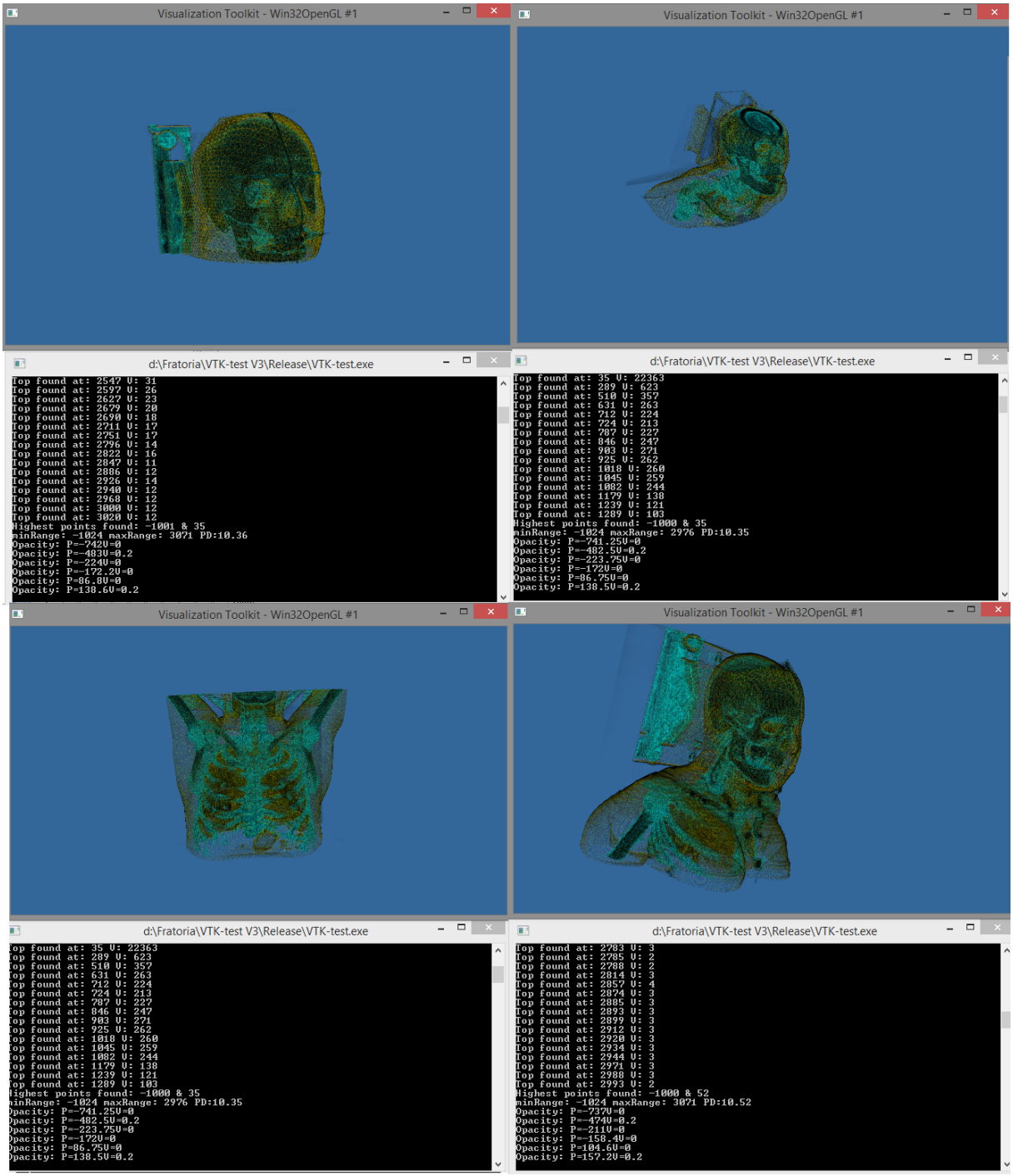
10.1 Originele planning



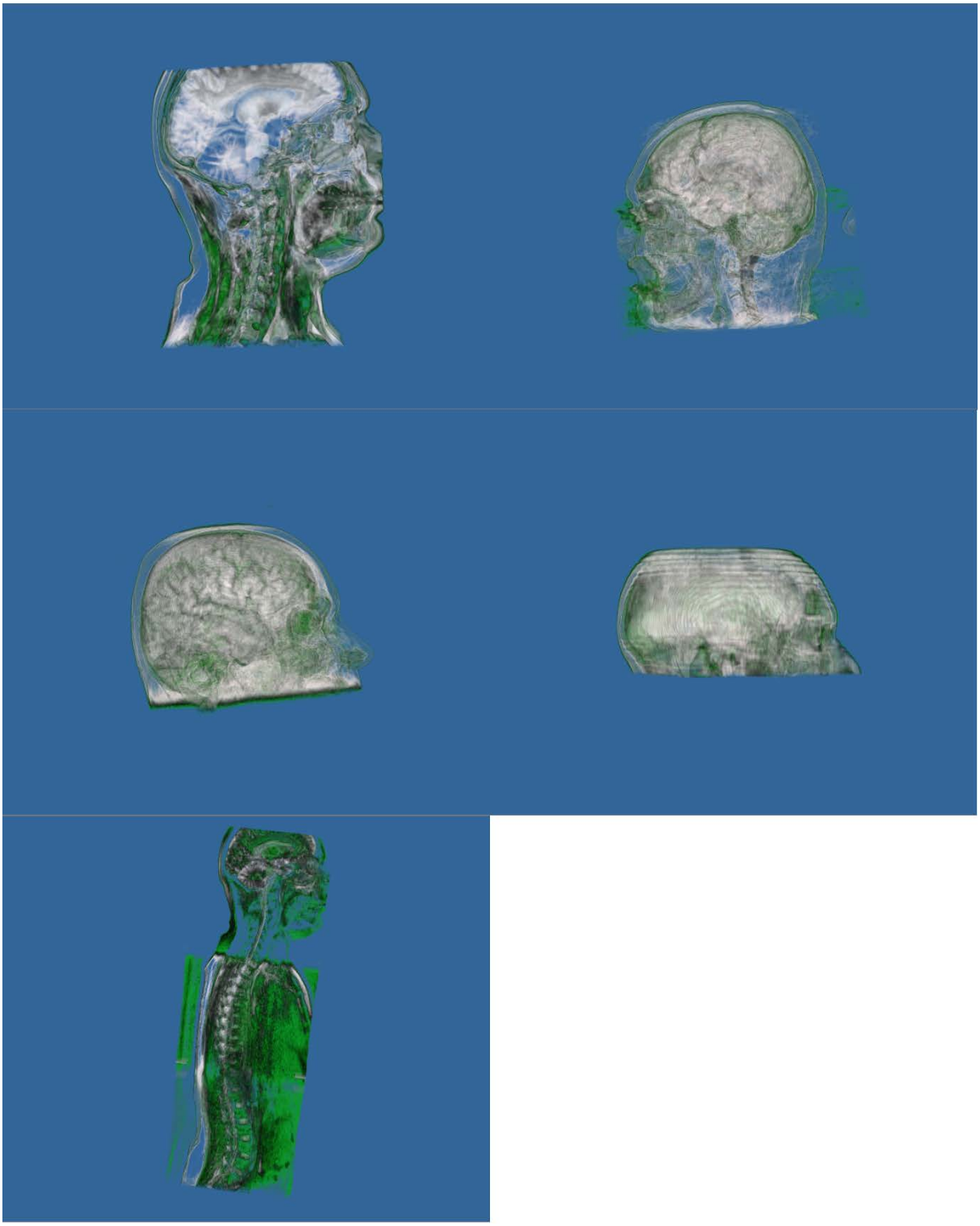
10.2 Uiteindelijke planning



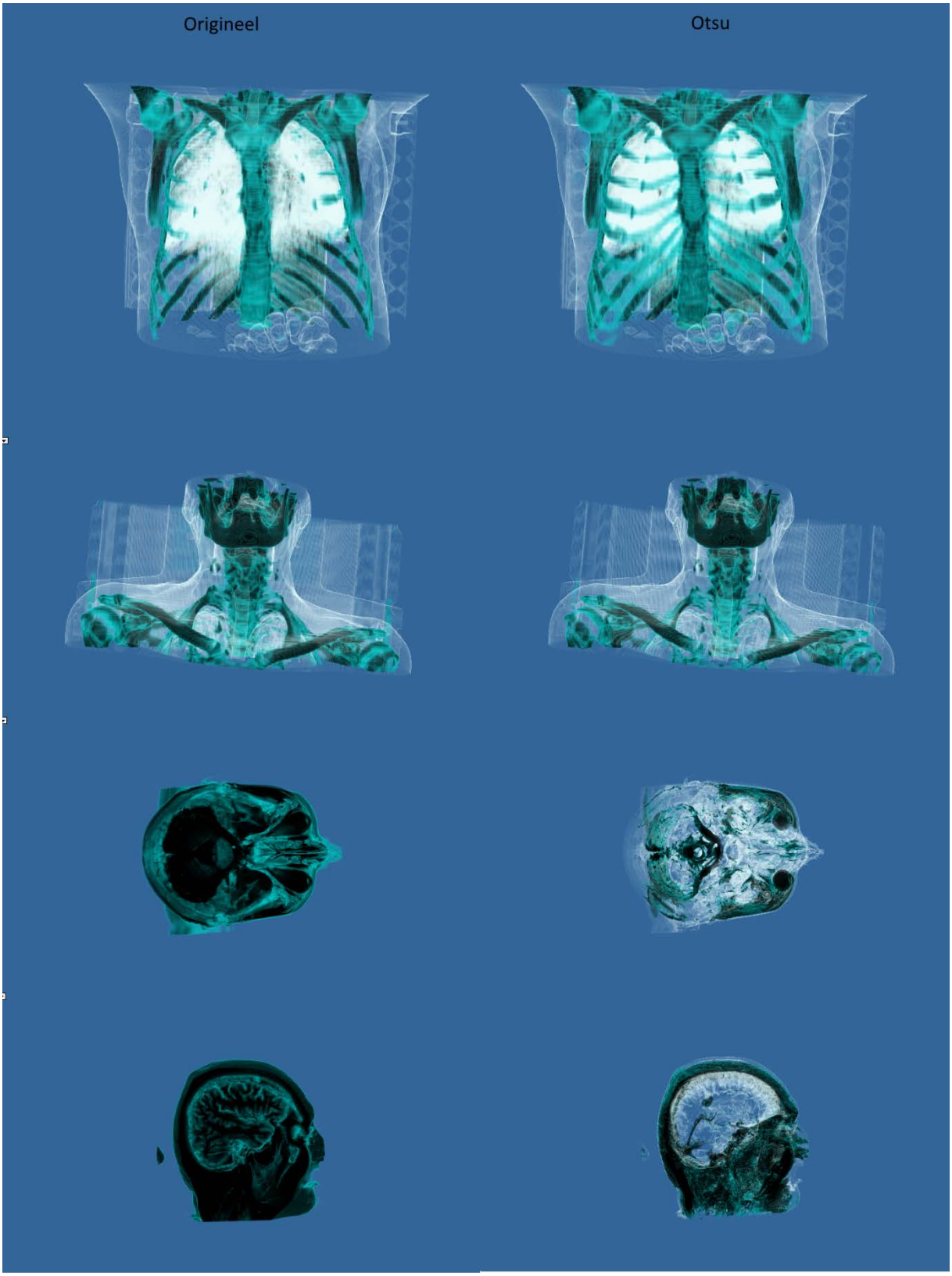
10.3 Testresultaten prototype 3



10.4 Testresultaten prototype 3 MR



10.5 Vergelijking Otsu met huidige systeem



10.6 Plan van aanpak

Zie “Plan van aanpak Fabian Schneider.docx”