DELFT SPLINE SYSTEMS

Afstudeerverslag

DpDental

03/07/2012

Opdrachtgever:	**Delft Spline Systems** **Lex Lennings**
Student:	**Jelko Sluijs**
Studentnummer:	**1187270**
Begeleider:	**Leo van Moergestel**

Afstudeerverslag van het DpDental project voor de studie Technische Informatica aan de Hogeschool Utrecht

Samenvatting

Het afstudeerproject DpDental voor het bedrijf Delft Spline Systems gaat over het uitbreiden van het programma DeskProto, zodat het gebruikt kan worden als onderdeel van een Dentaal Systeem. Het is uitgevoerd door Jelko Sluijs een student Technische Informatica aan de Hoge School Utrecht, de student is al enkele jaren werkzaam bij het bedrijf Delft Spline Systems.

Als eerste is er onderzocht welke features horen bij een dentaal systeem, dit is gedaan door middel van een bezoek aan een tandtechnisch laboratorium. In overleg met de opdrachtgever is er een keuze gemaakt van features die geïmplementeerd moesten worden. Naast de selectie van de features moest er ook rekening met de wensen van de opdrachtgever worden gehouden. Communicatie tussen de student en de opdrachtgever kon eenvoudig plaatsvinden, omdat zij werken in dezelfde kamer.

Het project maakt deel uit van een groter geheel - de aanpassing van een bestaand programma, een wens van de opdrachtgever was om de bestaande software zo min mogelijk te veranderen, daarbij moest er geprobeerd worden om zoveel mogelijk bestaande functionaliteit te gebruiken. Nadat de definitie van het project duidelijk was kon er een plan bedacht worden om de software te ontwerpen. Als aanpak is het project verdeeld in kleinere deelprojecten die zoveel mogelijk onafhankelijk van elkaar zijn, om ze aan het einde van het project aan elkaar te kunnen knopen als één geheel. Aan elke deelproject is er afwisselend gewerkt. Nadat er een feature was geïmplementeerd werd er een build van de software gemaakt, zodat de opdrachtgever het kon testen, vervolgens kon hij via Mantis (een bug-tracking-systeem) feedback gegeven.

De grootste uitdagingen van het project waren het berekenen van een automatische contourlijn om een geometrie en het plaatsen van een support op een vrij te kiezen positie. Dit is uiteindelijk allemaal goed gelukt, echter de wensen voor de automatische contour kwamen niet precies overeen met het doel voor het DpDental project. Hierdoor zijn een paar verkeerde keuzes gemaakt, deze conclusie kon echter pas achteraf gemaakt worden.

Bij de dentale versie van DeskProto hoort ook een grafische 3D weergave van een DpDental project. In DeskProto zijn 3D weergaven gerealiseerd door middel van OpenGL, omdat de student al bij het bedrijf werkte, is er bij dit project vanuit gegaan dat de student al vertrouwd was met OpenGL. In dit verslag zullen dan bijvoorbeeld geen problemen worden behandeld, die ontstonden bij het maken van een DpDental project 3D weergave.

Inhoudsopgave

1. Achtergronden

1.1 Inleiding

Dit afstudeerverslag beschrijft het project DpDental voor het bedrijf Delft Spline Systems. Het bedrijf Delft Spline Systems is gestart in 1985 en gericht op de verkoop en ontwikkeling van 3D-CAM software (het pakket DeskProto). Het is een klein bedrijf, momenteel werken er 3 mensen: een CAD-CAM specialist tevens eigenaar en twee ontwikkelaars. Het bedrijf is gevestigd in Utrecht (Tuindorp Oost), dat is dicht bij de Hogeschool Utrecht.
De afkorting CAD-CAM staat voor 'Computer Aided Manufacturing–Computer Aided Design', het ontwerpen en vervaardigen van producten met behulp van een computer.

1.2 Wat is Deskproto

DeskProto is een 3D-CAM Software pakket voor Windows gericht op het snel vervaardigen van modellen (Rapid Prototyping). Rapid Protoyping is het converteren van een 3D-CAD model naar een fysiek model, dit wordt gedaan op 2 verschillende manieren namelijk: door materiaal toe te voegen of door materiaal te verwijderen. DeskProto is gericht op de laatste manier - een model maken door materiaal te verwijderen. Dit heet CNC frezen – computergestuurd frezen. Op de website www.deskproto.com onder support zijn video`s hierover te vinden.

De eerste versie van DeskProto kwam in 1995 uit, de tweede versie is van de grond af aan opnieuw opgezet en kwam in 1998 uit. De structuur die toen is opgezet, is vandaag de dag nog steeds hetzelfde. DeskProto is een C++ MFC applicatie, die OpenGL gebruikt voor het tekenen van CAD modellen. Een CAD-model of geometrie is een virtueel 3D model gemaakt met CAD ontwerpsoftware zoals bijvoorbeeld Autocad.

Met het programma kan een DeskProto project worden aangemaakt. Een project bestaat uit Parts en Operations. Een Part legt vast welk deel van een model gefreesd moet worden, in een Operation kan met parameters worden ingesteld hoe een Part gefreesd moet worden. Als het te frezen werkstuk is vastgelegd kan het programma freesbanen berekenen. Een verzameling van freesbanen wordt opgeslagen in een NC file en kan dan naar een CNC freesmachine worden gestuurd.

1.3 Dentaal Systeem

Tegenwoordig kan ook het vervaardigen van een prothese voor in de mond (kroon of brug) door middel van CNC frezen. Dit wordt gedaan in tandtechnisch laboratoria die een dentaal CAD-CAM systeem bezitten. Een onderdeel van dat systeem is een 3D-CAM programma. Van deze software wordt momenteel het grootste gedeelte van die markt gedomineerd door enkele dure pakketten. Omdat het zo duur is hebben enkel de grote bedrijven de mogelijkheid om deze dienst aan te bieden. Het is de bedoeling dat er straks met DeskProto een goedkoop alternatief beschikbaar komt, zodat het ook mogelijk wordt voor de kleinere tandtechnisch laboratoria om deze techniek toe te passen.

Om dit mogelijk te maken zijn er eerst enkele aanpassingen nodig voor DeskProto. Hiervoor is dus het volgende project bedacht: DpDental – dentale versie van DeskProto.

2. Probleemstelling

2.1 Vervaardigen van een prothese

Om uit te zoeken welke aanpassingen nodig zijn voor DeskProto voordat het gebruikt kan worden als CAM pakket bij dentale systemen, ben ik met een collega op bezoek geweest bij Williams-Benelux - een tandtechnisch laboratorium. Tijdens dit bezoek zijn we rondgeleid en is er uitgelegd hoe het proces voor het vervaardigen van kronen met behulp van CNC frezen in elkaar steekt.

Een patiënt met tandbederf gaat eerst naar een tandarts. Als de tandarts besluit dat er een kroon nodig is wordt de tand of kies geslepen in een stompje. Vervolgens wordt er van het gebit een afdruk gemaakt. Dit kan door de patiënt te laten happen in een soort klei. Hiervan wordt dan een afgietsel gemaakt in gips. Het gipsmodel wordt door de tandarts verstuurd naar een tandtechnisch laboratorium.

De bedoeling is dat er op het stompje een kapje (kroon) of brug geplaatst gaat worden. Een brug (figuur 1.) wordt gebruikt indien er sprake is van een afwezige tand. En dient als constructie voor een kunsttand tussen twee stompjes, waarbij op elk stompje een kroon komt met daartussen de kunsttand. Het vervaardigen van een kapje of brug gebeurt in een tandtechnisch laboratorium.

Fig. 1 - Een brug

In de eerste stap, het CAD gedeelte, wordt van het gipsmodel een digitale versie gemaakt door middel van een 3D scanner. Er zijn meerdere mogelijkheden voor dit proces, bijvoorbeeld door met een speciale scanner direct in de mond scannen. Vervolgens wordt met programmatuur een 3D-model van een kapje gemaakt. Het kan ook voorkomen dat er van een derde rechtstreeks een 3D-model wordt aangeleverd. Bij de tweede stap moet het kapje worden uitgefreesd – het CAM gedeelte.

Als materiaal wordt een ronde schijf van zirkonium gebruikt, ook wel een blank genoemd. Zirkonium is erg kostbaar, daarom is het van belang dat er zo min mogelijk van wordt verspild. Tevens heeft het materiaal de eigenschap dat het gaat krimpen indien het wordt gesinterd. Sinteren is een chemisch proces waarbij materiaalkorrels van bijvoorbeeld zirkonium op een temperatuur worden gebracht waarop ze nog net niet smelten. Hierdoor groeien de contactpunten aan elkaar waardoor het contactoppervlak groter wordt, wat resulteert in heel hard materiaal. Naast de hardheid is het materiaal ook weefsel vriendelijk, wat het uitermate geschikt maakt voor mondprothesen. Een ander materiaal wat bijvoorbeeld ook wordt gebruikt voor tandprothesen is porselein, of het ook geschikt is om te frezen is onduidelijk.

Voordat er wordt begonnen met het frezen van één of meer kapjes, is het taak om de 3D-modellen van de kapjes zo efficiënt mogelijk bij elkaar te plaatsen binnen de blank, zodat er zo weinig mogelijk materiaal van wordt verspild bij het frezen. Tevens krijgt elk kapje peilers waarop het stabiel kan staan tijdens het sinteren. En er worden ‘support-blocks’ toegevoegd, deze zijn nodig om ervoor te zorgen dat tijdens het frezen het werkstuk op zijn plek blijft zitten. Het belangrijkste gedeelte van het frezen is de preparatielijn, dit is het grensvlak rondom het kapje welke zo precies mogelijk moet aansluiten op het afgeslepen gedeelte van de tand of kies.

Bij de laatste stap, het sinteren, gaan de kapjes een oven in. Dan moeten de gefreesde kapjes op de peilers staan, zodat ze stabiel blijven tijdens dit proces. Doordat het materiaal gaat krimpen tijdens het sinteren, moet een model, voordat het wordt gefreesd, juist geschaald worden op basis van de krimpfactor. Elke blank heeft weer zijn eigen krimpfactor. Voor zover het CAD-CAM aspect, na het sinteren is de kroon nog niet klaar. Hoe het proces voor het vervaardigen van een kroon verder verloopt is onduidelijk.

Tegenwoordig is de tegenhanger van frezen - 3D printen - door nieuwe ontwikkelingen veel in het nieuws. Je zou je misschien afvragen of het niet eenvoudiger is om die techniek toe te passen. Dit is momenteel onmogelijk, doordat de techniek is beperkt door het gebruik van bepaalde materialen - het is niet mogelijk om zirkonium te printen. Wellicht dat daar in de toekomst verandering in komt.

2.2 Probleemanalyse

Op basis van het voorgaande verhaal is een lijst gemaakt met features die bij de software van een dentaal systeem horen. De lijst is nog niet absoluut, het zou best kunnen zijn dat er in de toekomst features bijkomen of vervallen. Dit zal pas duidelijk worden zodra de software is uitgetest door meerdere tandtechnici.

- Specificeren van zirkoniumschijf (blank)
- Meerdere CAD-modellen beheren (positioneren, oriënteren en schalen)
- Support-bridge toevoegen op vrij te kiezen plaatsen
- Het te frezen gebied moet automatisch een freeformlijn rondom de geometrie zijn
- Peilers plaatsen (optioneel)
- Preparatielijn frezen (optioneel)

Naast de features die bij de software van een dentaal systeem horen zijn er ook nog de wensen van de opdrachtgever voor DeskProto. Deze zijn een gebruiksvriendelijke wizard en een instelbaar freesplan. Een wizard waarbij een gebruiker de mogelijkheid heeft om de voorgaande features toe te passen. De wizard moet dan alle frees technische instellingen automatisch toepassen op basis van een instelbaar freesplan.

- Dental Wizard
- Instelbaar freesplan (optioneel)

2.2.1 Specificeren van zirkoniumschijf

Specificeren van blanks, elke blank zal worden gespecificeerd aan de hand van bepaalde eigenschappen onder andere de krimpfactor, afmetingen en nummer of naam ter identificatie. Dit is vereist om CAD-modellen te positioneren en juist te schalen, zodat het de juiste afmeting heeft na het sinteren en om te onthouden waar al is gefreesd.

2.2.2 Meerdere CAD-modellen beheren

Het moet mogelijk worden om meerdere CAD-modellen van kapjes te positioneren en te oriënteren binnen een blank. Hierbij moet rekening worden gehouden of er op een positie al gefreesd is of niet. Het oriënteren is van belang om zoveel mogelijk ondersnijding te voorkomen: ondersnijding ontstaat indien de frees niet al het materiaal kan verwijderen waardoor er geen juist model(prototype) ontstaat.
Het oriënteren moet op een gebruiksvriendelijke manier omdat het bestemd is voor tandtechnici die weinig of niets weten van CNC frezen.

2.2.3 Support bridges

In DeskProto is de functionaliteit voor ondersteuning van support-blocks beperkt, momenteel moeten de supports evenwijdig aan de X, Y of Z as georiënteerd zijn, en is er een maximum van vier. Deze beperking moet worden opgeheven.

2.2.4 Automatische freeformlijn
Het te frezen gebied moet worden begrensd door een freeformlijn. Deze lijn is een, equidistante lijn er omheen (d.w.z. steeds op gelijke afstand van de buitencontour: namelijk de freesdikte). Deze lijn is ook nodig om de kronen binnen de blank te kunnen positioneren.

2.2.5 Peilers plaatsen
Mogelijkheid voor het plaatsen van peilers. Peilers zorgen ervoor dat het model stabiel kan staan tijden de sinterfase. Dit zou als een intelligente support-block kunnen, deze variant krijgt een dynamische lengte afhankelijk van de afstand tussen het model en het materiaal.

2.2.6 Preparatielijn frezen
Omdat de preparatielijn zo belangrijk is wil men deze vaak afzonderlijk nafrezen, daardoor komt het vaak voor dat er een afzonderlijke CAD-file van de preparatielijn is geleverd. Het zou ook mogelijk zijn om deze automatisch te bepalen. De preparatielijn is dan gedefinieerd als een 3D-polyline. Momenteel is er geen ondersteuning voor 3D-polylines, de wens is om dat in de toekomst te ondersteunen.

2.2.5 Dental Wizard
In de wizard moet stap voor stap een DeskProto project worden gemaakt dat geschikt is om kapjes te frezen uit zirkonium. Het doel daarbij is zo eenvoudig mogelijk, een gebruiker moet met zo min mogelijk kennis over frezen de stappen van de wizard kunnen doorlopen.

2.2.6 Instelbaar freesplan
In samenwerking met een CAD-CAM specialist zal er een typisch freesplan worden bedacht dat in de meeste gevallen uitkomst biedt. Denk onder andere aan voor- en nafrezen freeskeuze etc. De bedoeling is dat de wizard bij het aanmaken van een DeskProto project het freesplan overneemt. Dit is wellicht eenvoudig te realiseren met huidige functionaliteit – de DeskProto defaults.

2.3 Doelstellingen

Het hoofddoel is om DeskProto uit te breiden zodat het gebruikt kan worden als onderdeel van een Dentaal Systeem.

subdoelstellingen:

- Een goede tool voor tandtechnici, dit zal hoofdzakelijk bepaald worden door de gebruiksvriendelijkheid van de software plus de kwaliteit van de freesbanen.

- Het penetreren van een nieuwe markt voor DeskProto, prijs en eenvoud zijn de sleutelwoorden waar DeskProto zich van moet gaan onderscheiden ten opzichte van de concurrentie.

    - Het zou gebruikt kunnen worden door zelfstandige tandartsen, zodat zij niet afhankelijk hoeven te zijn van een tandtechnisch laboratorium.
    - Met een beperkt budget het mogelijk maken om toch prothesen te ontwikkelen, denk bijvoorbeeld aan ontwikkelingslanden.

3. Opdracht

De opdracht omvat het uitbreiden van het programma DeskProto met een deel van de features zoals die in de probleemanalyse zijn opgesomd. Er is een keuze gemaakt uit de lijst van features, die minimaal nodig zijn om het te gebruiken als CAM software voor dentale systemen. Na overleg met de opdrachtgever is besloten om de volgende lijst met features te implementeren.

- Specificeren van blank
- Meerdere CAD-modellen positioneren en oriënteren
- Support bridges met oriëntatie op vrij te kiezen plaatsen
- Automatische contour
- Dental Wizard ontwerpen

De wizard zal het eindproduct zijn van de opdracht. Hierin zullen de genoemde features toegankelijk zijn voor de gebruiker. De wizard moet de volgende opties hebben:

- Keuze voor een zirkonium blank (een nieuwe of waarvan reeds een deel gebruikt)
- Laden van CAD-model (kroon of brug)
- Het CAD-model kunnen positioneren binnen een blank
- Het CAD-model kunnen oriënteren
- Supports met een oriëntatie op vrij te kiezen plaatsen
- Optie om de freesbanen te berekenen

4. Aanpak

Als aanpak is het project verdeeld in kleinere deelprojecten die zoveel mogelijk onafhankelijk van elkaar zijn, om ze aan het einde van het project aan elkaar te kunnen knopen als één geheel. Aan elke deelproject is er afwisselend gewerkt. Nadat er een feature was geïmplementeerd werd er een build van de software gemaakt, zodat de opdrachtgever het kon testen, vervolgens kon hij via Mantis (een bug-tracking-systeem) feedback geven.

De volgorde die aanvankelijk was bedacht bij de planning is uiteindelijk niet de volgorde geworden bij het implementeren. Dit kwam voornamelijk door de technische uitdagingen. Omdat van te voren niet duidelijk was hoe een bepaald probleem zou worden opgelost, kon er geen goed plan gemaakt worden.
Een aantal hindernissen hebben een belangrijke invloed gehad op het tot stand komen van het eindresultaat:

- *Gebruik van hoofdscherm* - Momenteel is er een probleem met de huidige structuur van DeskProto, waardoor het redelijk complex is geworden om het hoofdscherm te gebruiken. Al was dit niet gelukt was er misschien voor gekozen om een apart dialoog te ontwerpen voor het tekenen van een DpDental Project.

- *Snelheid contour algoritme* - De snelheid van het contour algoritme voor het bepalen van de automatische freeformlijn had invloed op het verloop van het project. Al had het algoritme heel langzaam geweest, was er waarschijnlijk een totaal andere wizard gekomen. Hiervoor zijn een aantal snelheidstesten gedaan.

- *Oriënteren van een CAD-model* - Al had de huidige code voor het oriënteren niet bruikbaar geweest, hadden er waarschijnlijk tekstvakken in de wizard gekomen waarmee een oriëntatie kon worden ingesteld.

Gebruikte tools
Omdat het gaat om een aanvulling voor DeskProto – het project maakt deel uit van een groter geheel. Lag het voor de hand om dezelfde tools te gebruiken als de tools die gebruikt worden voor DeskProto.

- *Visual Studio* - De Integrated Development Environment (IDE) die gebruikt is voor het programmeren is Microsoft Visual Studio 2011, omdat de gebruikte programmeertaal C++ is hoort daarbij ook nog de Visual C++ compiler bij.

- *Subversion* - Als versiebeheersysteem wordt een tool genaamd Subversion gebruikt, deze tool legt de historie vast van elke wijziging die plaatsvindt aan de codebase. Met deze tool is het ook mogelijk om een zogenaamde branch te maken, een vertakking van de huidige codebase. Voor het DpDental project zijn alle code wijzigingen op een branch uitgevoerd. Zie in bijlage 2 bij Subversion log een historie van alle wijzigingen.

- *Mantis* - Een bug-tracking systeem, hierin worden de historie en status van problemen (features of bugs) geregistreerd. Vergelijkbaar met een Scrum product backlog. Als er een probleem is opgelost door de programmeur kan de status van een bug in Mantis aangepast worden. In een overzicht kan de opdrachtgever zien van welke problemen terugkoppeling wordt verwacht.

- *Enterprise Architect* - Een UML-tool van Sparx Systems die is gebruikt voor het maken van UML diagrammen, bv. de klassendiagrammen.

- *Paint.NET* - Een gratis tekentool gebruikt voor het maken van icons.

5. Beschrijving eindresultaat

In dit hoofdstuk wordt het eindresultaat beschreven op een relatief eenvoudige wijze om een beeld te krijgen van de verschillende aspecten van het DpDental project, in het volgende hoofdstuk zal een technische beschrijving volgen. Het eindproduct – de dental wizard, bestaat uit twee delen namelijk: een dialoog en een DpDental view. Het dialoog bestaat uit een drietal pagina`s die een gebruiker stap voor stap begeleiden bij het maken van een DpDental project. De DpDental view is een grafische weergave van een DpDental project.

Zie figuur 2. voor een eenvoudig klassendiagram van het DpDental model, in bijlage 2. bij paragraaf DpDental Model staat de uitgebreide versie.

5.1 Dental View

Het is mogelijk voor de gebruiker om de view te wijzigen net zoals bij een camera. Er zijn drie camera functies: *Pan*, *Rotate* en *Zoom*. Alle functies worden aangestuurd door middel van de muis. Met de *Pan* functie kan het beeld verschoven worden, dit kan door de muiswiel knop ingedrukt te houden. De *Rotate* functie wordt geactiveerd door de linkermuisknop ingedrukt te houden en de zoomfunctie gaat door middel van het op en neer bewegen van de muiswiel.

Naast het wijzigen van de view wordt het scherm(de view) ook gebruikt voor interactie met het DpDental project. Denk bijvoorbeeld aan het positioneren van een CAD-model binnen een blank.

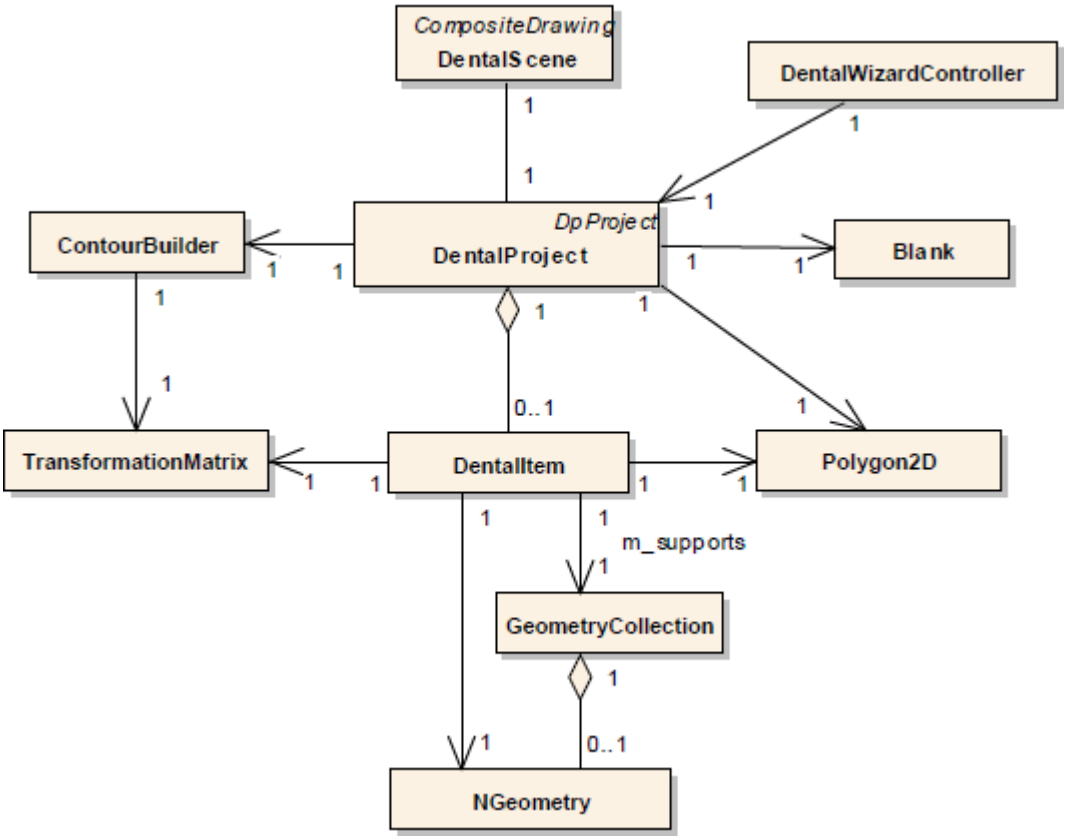


Fig. 2 – DpDental model

5.2 Blank page

Als een gebruiker de wizard start in DeskProto begint hij op de eerste pagina – de blank pagina. Op de blank pagina (figuur 3.) heeft de gebruiker twee opties: hij kan door middel van de combobox een al eerder gebruikte blank selecteren, of de gebruiker kan een nieuwe blank specificeren. Afhankelijk van de keuze worden respectievelijk de controls voor een nieuwe blank of een bestaande in- of uitgeschakeld.

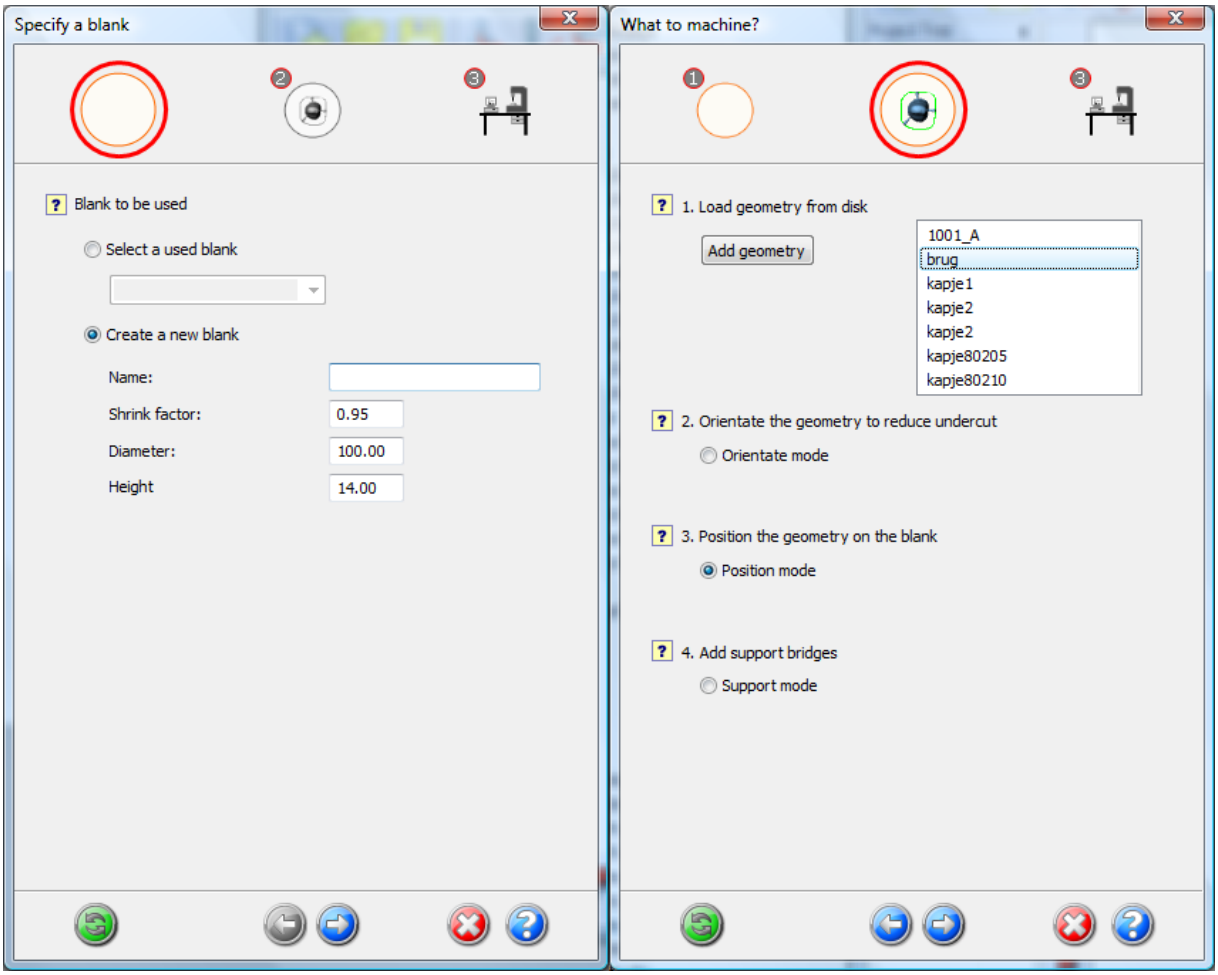


Fig. 3 - Pagina 1 en 2 van de wizard

Er zijn vier tekstvelden die de gebruiker kan invullen voor het specificeren van een blank. Ten eerste de naam, dit veld heeft verder geen invloed op de werking van het programma, en dient puur ter identificatie. Met het tweede veld de krimpfactor kan de gebruiker aangeven in wat voor mate het freesmodel krimpt als het de sinteroven in gaat. Om ervoor te zorgen dat het uiteindelijke resultaat niet te klein is, berekent de software een schaalfactor en past die toe op het CAD-model. In het derde en vierde veld kan de gebruiker de dimensies van de blank vastleggen: de diameter in mm. en hoogte in mm.

Zodra de gebruiker zeker weet voor blank hij wil gaan gebruiken, kan hij naar de volgende pagina gaan.

5.3 Material page

Op de tweede pagina kan de gebruiker vastleggen welk deel van de blank gefreesd mag worden. Dit gaat door middel van het toevoegen en instellen van één of meerdere DentalItems. Een DentalItem is een projectitem van een DpDental project.
Elk DentalItem bestaat uit de volgende onderdelen (figuur 4.):

- CAD-model
- Positie
- Oriëntatie
- Contour
- Eén of meer supports

De eerste stap is om een CAD-model te laden van de schijf (een file met een STL, DXF of VRML extensie), dit kan door middel van de knop '*Add geometry*'. Zodra de file succesvol is geladen wordt er een DentalItem toegevoegd aan het project en wordt er automatisch rondom het CAD-model een contour berekend. De contour zal het te frezen gebied begrenzen. Op de view van het hoofdscherm zal een DentalItem verschijnen en in de listbox naast de '*Add geometry*' knop wordt aan de lijst een nieuwe item toegevoegd. Deze is gelijk aan de naam van de CAD-file.

In de listbox is het mogelijk om een DentalItem te selecteren, in de view is dit aangeduid door een contour rood te maken. Daarnaast is het ook mogelijk om via de muis een item te selecteren, dit gaat door middel van een linkermuisklik op een item.

De tweede stap is het instellen van de oriëntatie. Als de radiobutton is ingesteld op '*Orientate mode*' kan de gebruiker oriënteren door: op een punt te klikken binnen de contour van het geselecteerde DentalItem en de muis te slepen met de linkermuisknop ingedrukt. De geometrie wordt dan ten opzichte van de blank gedraaid.
Als de gebruiker de linker muisknop loslaat wordt de nieuwe oriëntatie opgeslagen en wordt de contour opnieuw berekend. De contour is een freeformlijn rondom de geometrie van het bovenaanzicht. Indien de oriëntatie is aangepast wijzigt ook het bovenaanzicht, daarom moet de contour opnieuw worden bepaald als de oriëntatie wijzigt, dit is tevens ook de reden waarom na het laden van een CAD-model het oriënteren als eerste moet.

De volgende stap is het positioneren van een DentalItem binnen de blank, dit gaat op een vergelijkbare wijze als bij het oriënteren: Stel de radiobutton in op '*position mode*', klik op een punt binnen de contour van het geselecteerde item en sleep de muis met de linkermuisknop ingedrukt. Tijdens het slepen wordt de positie van de contour automatisch geüpdate, dit is mogelijk omdat de contour van de geometrie hetzelfde blijft, enkel de positie wijzigt, hierdoor is het updaten van de contour heel eenvoudig.

De laatste stap van de material page is het toevoegen van één of meer supports, stel de modus in op '*add support mode*' en klik op een vlakje van het geselecteerde DentalItem. Op die positie zal een support worden toegevoegd, die loodrecht op het vlakje zal staan, met een lengte die automatisch wordt bepaald. De lengte van de support zal automatisch worden begrensd door de contour.

De gebruiker zal de voorgaande stappen moeten herhalen voor elke CAD-file die hij wil gaan frezen. Hierbij heeft hij zelf de keuze in wat voor volgorde, echter het meest logische is om de volgorde aan te houden zoals hierboven staat beschreven.

Nadat de gebruiker de blank zo efficiënt mogelijk heeft ingedeeld met DentalItems kan hij naar de volgende pagina gaan.

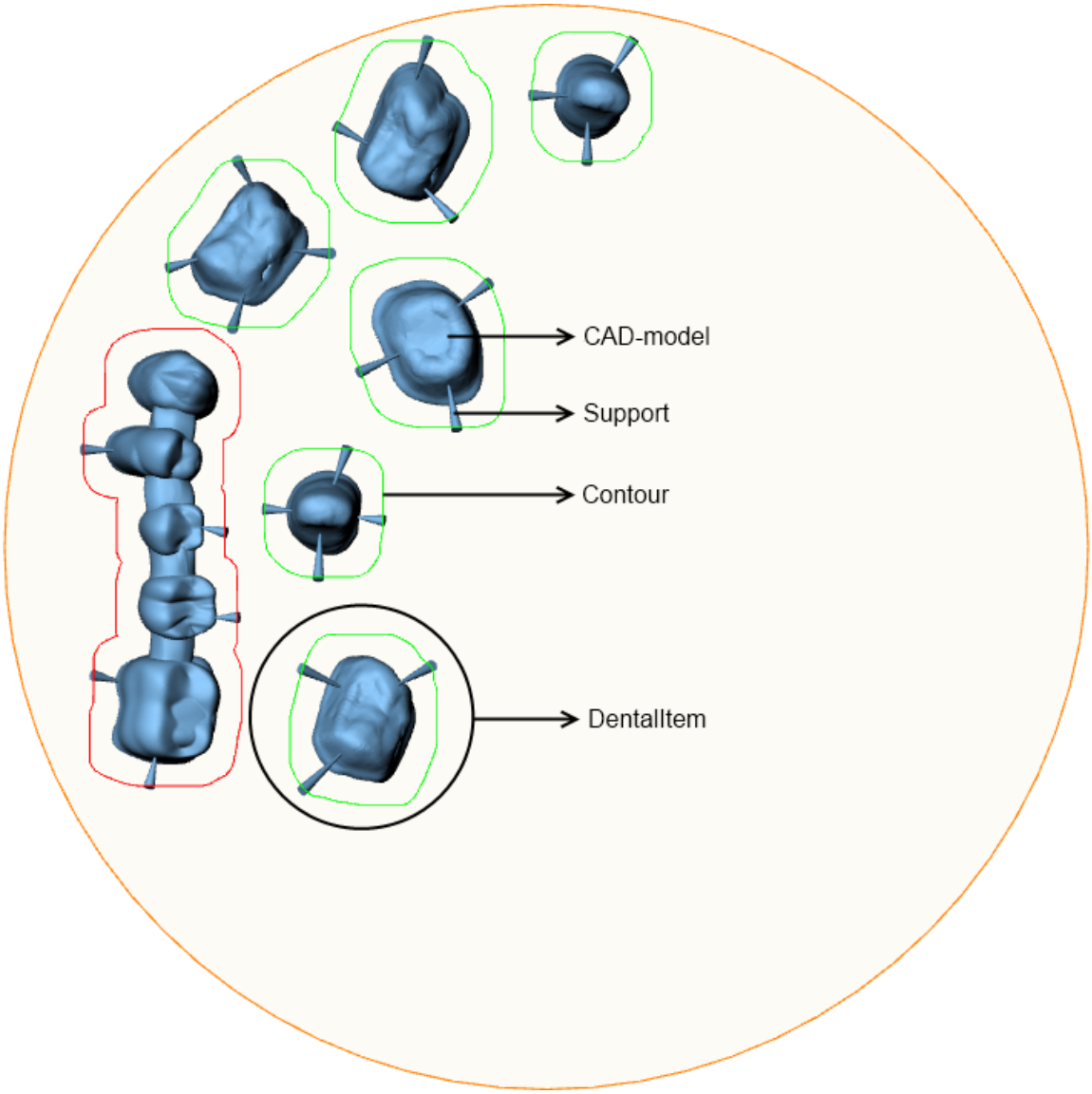


Fig. 4 - Dental view

5.4 Machine page

Zodra de gebruiker naar de volgende pagina gaat wordt het DpDental project omgezet naar een regulier DeskProto project en wordt de DpDental view vervangen door de normale view van DeskProto. (figuur 5.) Het toepassen van alle relevante freesinstellingen die in de meeste gevallen uitkomst zal bieden vindt plaats bij het omzetten.

Op de pagina heeft de gebruiker als eerste de optie om aan te geven van welke DentalItems hij de freesbanen wilt berekenen. Vervolgens kan hij met het knopje ´*calculate*´ de freesbanen berekenen. Afhankelijk van wat de gebruiker wil kan hij ze direct naar een freesmachine sturen of eerst bewaren op de schijf als een NC-file, om ze later naar een machine te sturen.

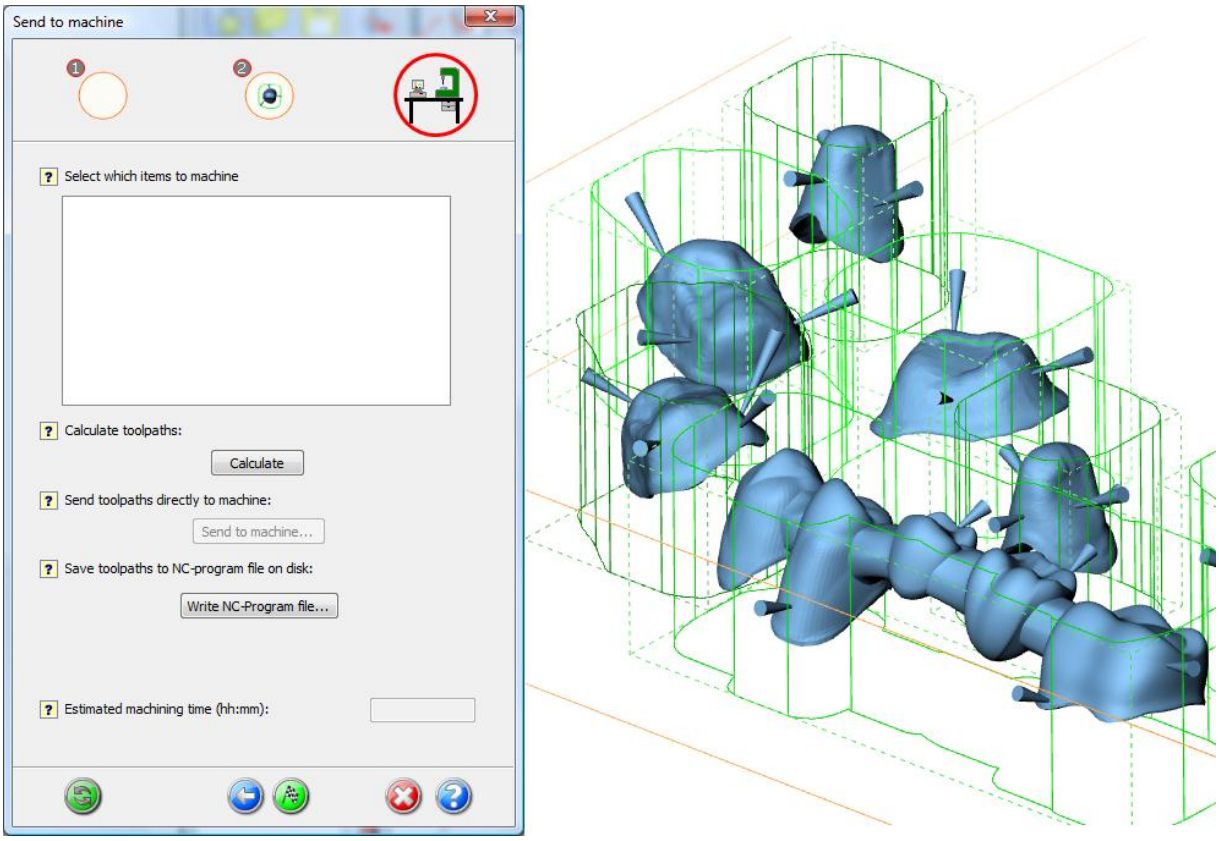


Fig. 5 - Page 3 en normale view

6. Technische beschrijving

In dit hoofdstuk volgt een technische beschrijving, om inzicht te krijgen in de werking van de gemaakte programmatuur. Enkel de onderwerpen met de grootste technische uitdaging voor het tot stand komen van het eindresultaat zullen worden behandeld.

6.1 SRT-Matrix

De positie, oriëntatie en grootte van een geometrie object, wordt beschreven door middel van een transformatiematrix, een serie matrix vermenigvuldigingen. Dit wordt voor meerdere doeleinden gebruikt:

- Transformeren van geometrie objecten
- Aanpassen van Viewpoint (dient als camera bij een 3D weergave)
- Wegschrijven naar de projectfile
- Tekenen van de geometrie objecten

Met de term geometrie object wordt een CAD-model voor een DentalItem of Support bedoeld. Een geometrie object bestaat uit een verzameling van driehoeken, een driehoek wordt weer beschreven door drie punten en een normaal vector (een eenheidsvector die loodrecht op de driehoek staat), om een object te transformeren worden alle punten vermenigvuldigd met een transformatiematrix.

Door de matrices altijd in een vaste volgorde te vermenigvuldigen, is het mogelijk om ze op een uniforme manier te behandelen. De volgorde die wordt aangehouden is schaling, rotatie en translatie. Ook wel een SRT-matrix genoemd. De schaal component bepaald de grootte van een geometrie object, de rotatie component de oriëntatie en de positie wordt bepaald door de translatie component.

```
/// <summary>calculates a matrix from all values in this transformset</summary>
TransformationMatrix TransformSet::calculateMatrix()
{
    TransformationMatrix m;
    m.translate(m_xTranslation, m_zTranslation, m_zTranslation);
    m.rotz(m_zRotation * DEG2RAD);
    m.roty(m_yRotation * DEG2RAD);
    m.rotx(m_xRotation * DEG2RAD);
    m.scale(m_xScale, m_yScale, m_zScale);
    return m;
}
```

Fig. 6 - Code voor het berekenen van een SRT-matrix

In het code voorbeeld (figuur 6.) is te zien hoe een SRT-matrix wordt berekend. Om een transformatiematrix te berekenen die eerst schaalt, roteert en dan transleert is het van belang dat de matrices in de omgekeerde volgorde worden vermenigvuldigd. Dit lijkt erg verwarrend, maar het heeft te maken met het feit dat matrix vermenigvuldigingen niet commutatief zijn. Bij de bronvermeldingen staan verwijzingen naar artikelen op internet die uitleggen hoe een translatie, rotatie of schaal matrix wordt berekend en hoe je deze kan vermenigvuldigen.

De oriëntatie van een geometrie object bestaat uit een combinatie van rotatie matrices, één om de x-as, y-as en z-as. Een combinatie van rotaties om de x-as, y-as en z-as kan worden beschouwd als één rotatiematrix. Doordat het mogelijk is om een combinatie als één te beschouwen wordt het werken met oriëntaties veel eenvoudiger in tegenstelling tot het beschrijven van een oriëntatie door een rotatie om de x-as, y-as en z-as.

Door middel van matrix decompositie is het mogelijk om de scalaire, rotatie en translatie componenten af te leiden. De klasse TransformSet (figuur 7.) heeft deze functionaliteit ingekapseld. De methode *decomposeMatrix* ontleed een matrix en slaat de componenten op in de member variabelen van TransformSet. (figuur 8.)

Met de klasse TransformSet is het ook mogelijk om de componenten afzonderlijk in te stellen en vervolgens een matrix te berekenen met de methode *calculateMatrix*.

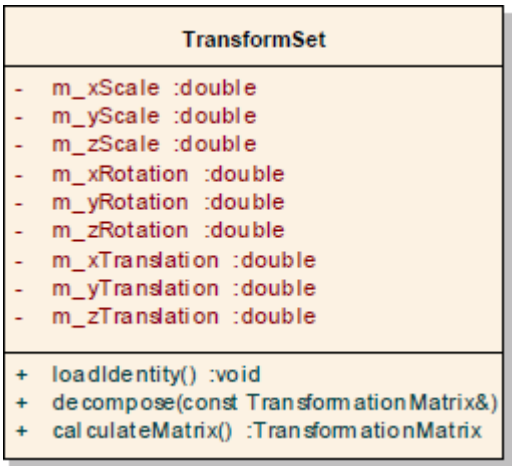


Fig. 7 - De klasse TransformSet

```
/// <summary>
/// decomposes the matrix to transform parameters (Matrix decomposition)
/// extracts the scalar, translation and rotation components from a SRT matrix
/// </summary>
void TransformSet::decomposeMatrix(const TransformationMatrix& matrix)
{
    // scaling
    m_xScale = matrix.getColumnVector(0).mag();
    m_yScale = matrix.getColumnVector(1).mag();
    m_zScale = matrix.getColumnVector(2).mag();
    // translation
    Point3D translation = matrix.getColumnVector(3);
    m_xTranslation = translation.x;
    m_yTranslation = translation.y;
    m_zTranslation = translation.z;
    // rotation
    TransformationMatrix orientation = matrix.getRotationMatrix();
    m_xRotation = orientation.xAngle() * RAD2DEG;
    m_yRotation = orientation.yAngle() * RAD2DEG;
    m_zRotation = orientation.zAngle() * RAD2DEG;
}
```

Fig. 8 - Code om een SRT-matrix te ontleden

Doordat de transformatiematrices op een uniforme manier worden behandeld plus de mogelijkheid om ze te ontleden door middel van decompositie, kan relatief eenvoudig de grootte, oriëntatie en positie van een geometrie object worden aangepast.

6.1.1 Probleem met middelpunt

Werken met SRT-matrices kan pas uniform indien de geometrie gecentreerd is op het nulpunt. Dit is belangrijk bij het roteren, anders draait de positie mee, en dit is in de meeste gevallen ongewenst. Dit gaf een probleem bij sommige CAD-modellen die niet waren gecentreerd, deze stonden al op een positie ingesteld. Vaak is dit opzettelijk omdat het de positie is waar de gebruiker wil frezen. Met andere woorden er is al een keer een translatie transformatie over de punten van de geometrie heen gegaan, hierdoor klopt de volgorde niet. Om alle CAD-modellen op dezelfde manier te behandelen wordt na het inlezen, indien nodig de geometrie eerst gecentreerd. Om de originele positie toch te kunnen gebruiken wordt de translatie die nodig was voor het centeren gebruikt als default voor de SRT-matrix.

6.2 Support toevoegen

Een kegelvormige geometrie van 1x1x1 mm (gepositioneerd bij het nulpunt). wordt van schijf geladen en gebruikt als basis voor elke support. Vervolgens wordt er een SRT-matrix opgesteld die de grootte, oriëntatie en positie van de support beschrijft.
Het algoritme voor het toevoegen van een support bestaat uit twee delen: het bepalen van de oriëntatie en het bepalen van de lengte. In bijlage 2, paragraaf 3.2 'toevoegen van supports' is de code te zien die gebruikt wordt voor het toevoegen van een support.

Als de gebruiker in de view op een driehoek klikt van de geometrie, wordt eerst de cursorpositie omgerekend naar een objectcoördinaat. Dat punt zal tevens ook als basis dienen voor de positie van de support - de translatie van de SRT-matrix. Het converteren van een muiscursor positie naar een objectcoördinaat wordt o.a. gedaan door gebruik te maken van een OpenGL functie: *gluUnproject*. Hoe deze functie precies werkt valt buiten de grenzen van dit verslag.

6.2.1 Bepalen van de oriëntatie

Het idee is dat de support loodrecht op het vlakje wordt geplaatst, hiervoor is de normaalvector van de driehoek nodig om een rotatiematrix te berekenen. Eerst wordt uitgezocht op welke driehoek is geklikt. Met het objectcoördinaat worden alle driehoeken van de geometrie doorlopen om te testen of deze in de driehoek ligt. Om te testen of een punt in een driehoek ligt wordt een functie *'isInsideTriangle'* gebruikt. Als de driehoek is gevonden, kan met de functie *'CalculateRotationMatrixFromNormal*' een rotatiematrix worden berekend. De functie berekend aan de hand van de normaalvector en een vector die omhoog wijst twee rotaties: één om de x-as en één om de z-as.

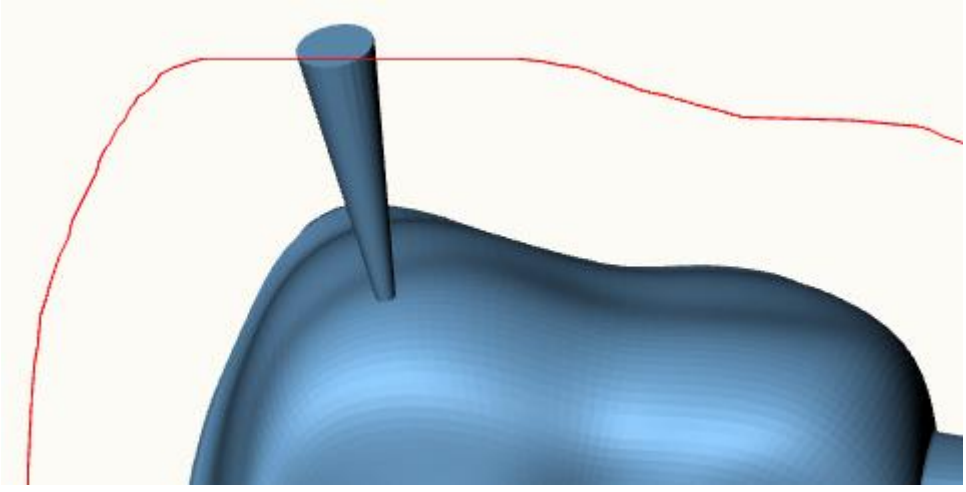


Fig. 9 - Een support

6.2.2 Bepalen van de lengte

De lengte van de support wordt automatisch begrensd door de contour. Om de lengte automatisch te bepalen moet er een snijpunt gevonden worden op de contour. Omdat de punten van een contour in 2D coördinaten zijn en die van een geometrie in 3D, zijn er extra stappen nodig om een snijpunt te vinden. Als het snijpunt is gevonden kan met de formule voor het bepalen van de lente van een vector, de lengte voor de support worden bepaald.

Denkbeeldige lijn bedenken

Als eerste wordt er een denkbeeldige lijn bedacht die van het vlakje tot aan een punt buiten de geometrie loopt. De eerste punt van de lijn zal de XY-waarde zijn in objectcoördinaten van het punt waar met de muis was geklikt. Het tweede punt van de lijn moet een punt zijn die buiten de contour ligt, dit wordt berekend met behulp van vector berekeningen.

Snijpunt zoeken

Met de denkbeeldige lijn wordt een functie in de klasse Polygon2D aangeroepen '*findIntersections'*, waarmee all snijpunten met de contour gevonden worden.

Kortste snijpunt vinden

Indien er meer dan één snijpunt is gevonden, wordt het snijpunt gebruikt die het dichtst bij het geklikte punt ligt.

Punt op de z-as bepalen

Als het snijpunt is gevonden kan door middel van de stelling van Pythagoras en de tangens formule een punt op de z-as worden bepaald.

6.3 Contour generatie

Er zijn bij het automatisch genereren van een contourlijn twee factoren van belang, deze zijn: nauwkeurigheid en snelheid. Voor het dentaal systeem is het belangrijk dat de contour snel gegenereerd kan worden. De nauwkeurigheid is niet belangrijk, omdat de primaire functie enkel het begrenzen van het te frezen gebied is.

Er is enorm veel literatuur te vinden omtrent contourlijnen, het helemaal goed uitwerken ervan zou al een afstudeerproject op zichzelf zijn. Daarom is er eerst gekeken naar bestaande libraries. Er zijn verschillende uitgeprobeerd, maar ze kosten allemaal teveel processorcapaciteit.

Een idee was om bijvoorbeeld de driehoeken van een geometrieobject plat te beschouwen (de z-waarde negeren), en deze via een Polygoon union functie aan elkaar te koppelen. Dit leverde wel een heel nauwkeurig resultaat op, echter dit duurde veels te lang, bij een wat grotere geometrie kan het al gauw een paar uur duren.

Zelf had ik bedacht om de geometrie te tekenen op een bitmap en vervolgens met een Vision techniek genaamd edge detection de randen te vinden. Dit idee is verder uitgewerkt, uiteindelijk is het algoritme een soort mix geworden van: een Vision techniek met gebruik van huidige code.

6.3.1 Beschrijving contour algoritme

In deze paragraaf wordt de werking van het contour algoritme globaal uitgelegd. In de volgende paragraaf wordt er dieper op ingegaan. In hoofdlijnen bestaat het contour algoritme uit een viertal stappen.

1. Tekenen van de geometrie op een offscreen buffer
2. Contourlijn bepalen aan de hand van een bitmap
3. Tekenen van geometrie plus contourlijn met een bepaalde dikte
4. Offset lijn bepalen aan de hand van een bitmap

De eerste stap is het tekenen van de geometrie op een plaatje in de kleuren zwart en wit. Zie figuur 10. Hiervoor wordt een OpenGL offscreen buffer voor gebruikt. Dit is gerealiseerd met de OpenGL pbuffer extensie(PixelBuffer). In DeskProto wordt de pbuffer extensie ook gebruikt voor een printfunctie. Nadat er op de offscreen buffer is getekend wordt de pixel data uitgelezen, er is dan in feite een bitmap. Vervolgens wordt er een boolean array met dezelfde dimensies als de bitmap gemaakt. Deze wordt dan gevuld met enen en nullen, voor elke zwarte pixel komt er een één en voor elke witte pixel een nul.

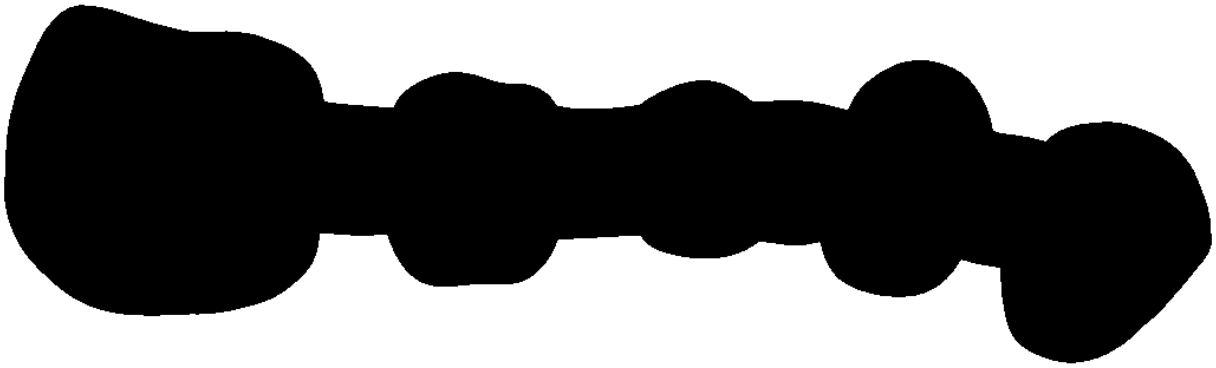


Fig. 10 - Bitmap van eerste stap

De boolean array wordt vervolgens gebruikt als input voor de klasse GridContourer, dit is een bestaande klasse in DeskProto die wordt gebruikt om een contour-only toolpath te berekenen. Wat deze klasse in feite doet is het aan elkaar knopen van alle grenspunten. Om zo een contour te krijgen. Zie groene lijn in figuur 11.

Fig. 11 - Contour na eerste stap

Om de contourlijn met offset te bepalen worden de eerste stappen herhaald. Echter nu wordt na het tekenen van de geometrie in het zwart de al eerder bepaalde contourlijn er overheen getekend, ook in het zwart. Deze lijn wordt getekend in een bepaalde dikte afhankelijk van de diameter van de frees. Het lijkt dan alsof er met een dikke stift langs de rand van de geometrie is gegaan. In figuur 12. is dat geïllustreerd, echter voor het voorbeeld is de lijn getekend in het groen.

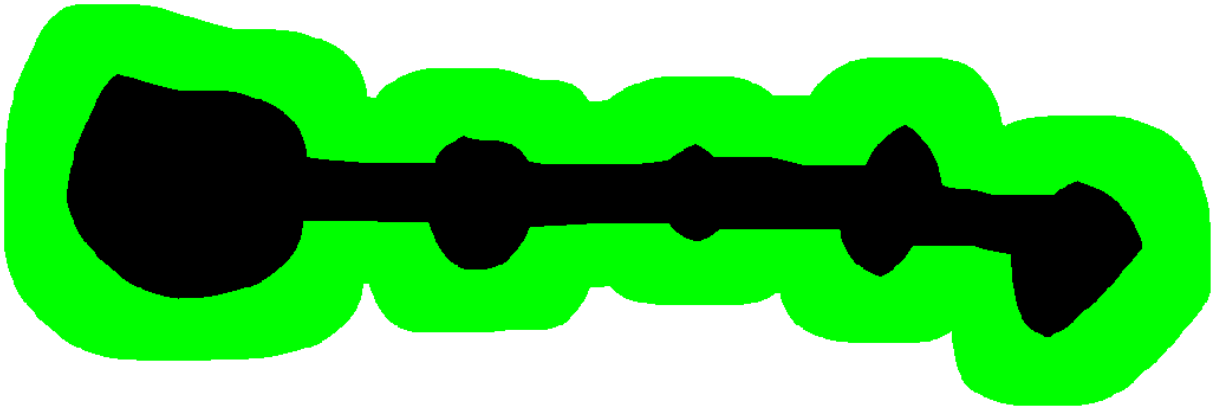


Fig. 12 - Contour met offset

Bij de laatste stap wordt weer een boolean array gemaakt en vervolgens met de GridContourer worden de grenspunten weer aan elkaar geknoopt. Echter nu plus een optimalisatie stap genaamd: smoothing en stripping. In figuur 13. is het resultaat te zien. Hier is de geometrie op een normale manier getekend met daarom heen de contourlijn met offset. In dit voorbeeld is de offset 1 mm.

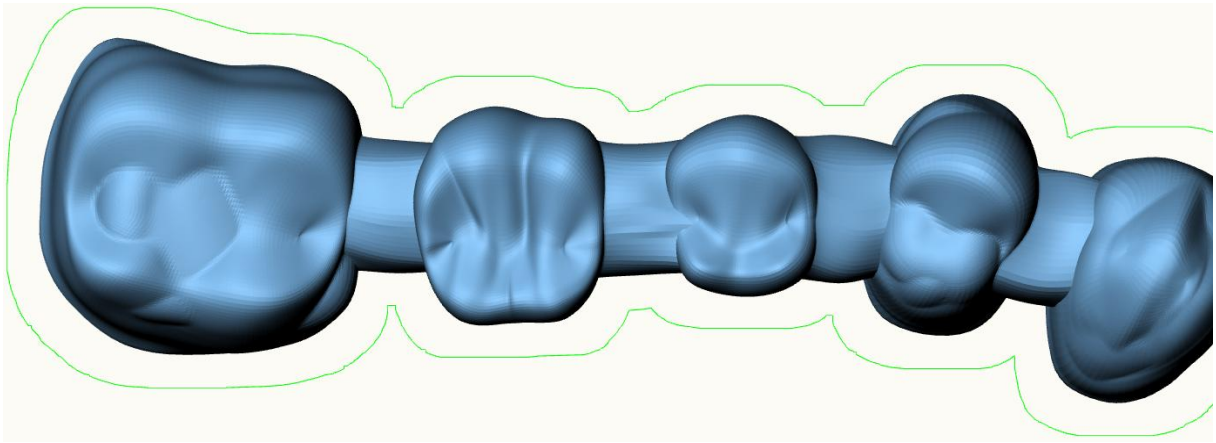


Fig. 13 - Resultaat

6.3.2 Problemen contour algoritme

In deze paragraaf worden enkele problemen genoemd, die tijdens het implementeren van het algoritme ontstonden.

Zo efficiënt mogelijk de pixelbuffer gebruiken

Er is gekozen voor een standaard dimensie van 1024 bij 1024 pixels. Om de 1024 bij 1024 pixels zo goed mogelijk op te vullen plus ervoor te zorgen dat er niet buiten wordt getekend, moet de geometrie bij het tekenen juist geschaald worden.
Het berekenen van de schaalfactor wordt berekend aan de hand van een boundingbox.

Terugrekenen van pixelcoördinaten naar objectcoördinaten

Nadat alle punten zijn geconverteerd naar een contour, moeten deze nog omgezet worden naar objectcoördinaten. Dit gebeurt door alle punten te converteren met een matrix. De matrix wordt berekend door:

Matrix = inverse modelMatrix * inverse projectieMatrix

Converteren van punten naar een polygoon

Dit gebeurt met de klasse GridContourer. Een probleem met het gebruik van de GridContourer was dat er soms geen gesloten contour kon worden gevormd. Dit had te maken met de oorsprong van de GridContourer, om dit op te lossen is bij het tekenen van de geometrie ervoor gezorgd dat er altijd een rand witte pixels overblijft.

Tekenen van de offsetlijn

Het tekenen van een 2D polygoon met dikke lijnen in OpenGL bleek nog vrij gecompliceerd te zijn. Het probleem is dat OpenGL een vrij primitieve API is. Het tekenen van dikke lijnen laat gaten achter – zie figuur 14. en 15.

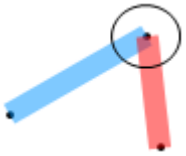


Fig. 14 - tekenprobleem

Om dit op te lossen worden bij de eerste stap de punten nog niet gestript en wordt bij het tekenen van de offsetlijn in plaats van lijnen punten getekend, op die manier worden alle gaten gevuld.

Fig. 15 - Illustratie van tekenprobleem

Een ander probleem die om de hoek kwam kijken, ging om de dikte van de lijn die getekend moest worden. Stel dat de offset 2 mm. is, hoeveel pixels moeten er dan ingekleurd worden voordat de 2 mm. is bereikt. De mm. naar pixel conversie gebeurt door de breedte van het model (welke in mm. is) te delen door de breedte van de pixelbuffer. Of indien de hoogte groter is dan de hoogte van het model delen door de hoogte van de pixelbuffer.

Door dus de offset in mm. te vermenigvuldigen met de conversiefactor, krijg je de juiste lijndikte in pixels die dan aan de functie *glLineWidth* kan worden door gegeven. Echter de functie *glLineWidth* is zwaar verouderd, er wordt niet gegarandeerd dat de opgegeven waarde in pixels precies het aantal pixels omslaat. Wat dat precies in de praktijk gaat betekenen is nog onduidelijk, er zal met verschillende typen videokaarten van verschillende fabrikanten getest moeten worden, dit zal typisch in de beta test periode plaatsvinden.

RenderContext switch

Om naar een OpenGL pixelbuffer te tekenen dient er een aparte rendercontext aangemaakt te worden. En er moet een pixelbuffer worden aangemaakt. Voordat er dan getekend kan worden moet deze eerste actief gemaakt worden. Een contextswitch kost relatief veel processortijd, om te voorkomen dat er telkens een contextswitch plaatsvindt, zijn er in de klasse twee methoden *begin* en *end* gemaakt. In de methode *begin* wordt de huidige actieve context onthouden en op non actief gezet, vervolgens wordt de context die geassocieerd is met de pixelbuffer actief gemaakt. Vervolgens kan er naar getekend worden. Zie ook het sequence diagram en klassendiagram in bijlage 2. paragraaf 1.4.

7. Implementatieplan

Ten eerste moet de huidige software verder uitgebouwd worden voordat de software publiceerbaar is. Daarbij moet er gekeken worden naar de overige features die horen bij een dentaal systeem. Vragen zoals: Welke features zijn nog eenvoudig toe te voegen? Welke features moeten nog geïmplementeerd worden en welke zijn wenselijk, maar niet noodzakelijk? Ten tweede moet er een testperiode plaatsvinden, in die periode worden de freesmodellen en de werking van de software getest.

Hieronder volgt een lijst met aandachtspunten van de overige features voor een dentaal systeem:

- Instelbaar freesplan
    - In overleg met tandtechnici bepalen van freesinstellingen die in de meeste gevallen uitkomst kan bieden (denk o.a. freeskeuze, nauwkeurigheid, strategie)
    - Zou dit op een generieke manier mogelijk zijn, zodat het ook bruikbaar is voor de bestaande wizards?
- Onthouden waar al is gefreesd
    - Hoe kan dat het beste worden geïmplementeerd? (bv. materiaalbibliotheek, vlaggetjes per DentalItem, etc.)
- Plaatsen van peilers
    - Uitzoeken of er ook bedrijven zijn die bij het sinteren van het model geen peilers gebruiken
    - Er zal in plaats van een kegelvormige geometrie waarschijnlijk een cilindervormige geometrie nodig zijn
    - Niet loodrecht op een vlakje plaatsen maar evenwijdig aan de z-as
    - Eventueel opvullen van ruimte met geometrie die zal ontstaan tussen het vlakje en het raakvlak van de support
    - Kan waarschijnlijk eenvoudig toegevoegd worden door gebruik te maken van de huidige support code
- Preparatielijn frezen (3D-polyline frezen)
    - Welke wijzigingen zijn nodig om een 3D-polyline te frezen?
    - Hoe belangrijk is het frezen van de preparatielijn?
- Bepalen preparatielijn
    - Hoe complex is het bepalen van de preparatielijn eigenlijk?
    - Is het interessant voor DeskProto om dit automatisch te kunnen, zou zo`n algoritme ook nog interessant zijn voor andere toepassingen binnen DeskProto?

Hieronder volgt een lijst met aandachtspunten voor het uitbouwen van het huidige systeem:

- De tooltip teksten toevoegen
- De handleiding uitbreiden
- Vertalingen toevoegen
- Plaatjes voor de wizard dialoog ontwerpen
- Validaties toevoegen (denk bv. aan overlapping van DentalItems, zijn items binnen de Blank?)
- Hover effecten bij het wijzigen van een DentalItem (aanpassen cursor)
- Voortgangsdialogen bij functies die langer dan 5 seconden duren
- Mogelijkheid voor verwijderen van een DentalItem
- Verdere integratie met huidige user interface (bv. gebruik van knoppen die in de toolbar staan, bijwerken statusbar)

Hieronder volgt een stappenplan voor het testen van de software en freesresultaten.

1. Lokaal testen van de software
2. Lokaal testen van freesresultaten d.m.v. de simulatie
3. Testen van de freesresultaten met materiaal voor prototype`s (in het jargon Tooling Board)
4. Testen van de freesresultaten op zirkonium
5. Testen van de software en freesresultaten door beta testers

Verdere aandachtspunten:

- Wordt de DpDental edition een aparte build? Indien dat niet het geval is moet de code op de aparte Subversion branch worden samengevoegd met de huidige codebase
- Moet er een aparte installatie komen? (bv. met een DpDental voorbeeld project)

8. Evaluatie

Contour algoritme
Aanvankelijk was het het doel om een zo goed mogelijke algoritme te maken voor het genereren van een contour, zodat het ook gebruikt kan worden bij de berekening van freesbanen. Dit was echter een misvatting, want het doel van de contour voor het DpDental systeem is enkel het begrenzen van het freesgebied. Daarbij maakt het niet zoveel uit hoe nauwkeurig de lijn is. Natuurlijk wil je automatisch een zo goed mogelijk algoritme verzinnen, als je zo`n soort probleem gaat oplossen. Echter de problemen die allemaal spelen bij het verzinnen van een algoritme voor een perfecte contour worden gauw over het hoofd gezien. Echter door deze fout is hierover wel meer inzicht gekregen. Voor DeskProto kunnen het beste twee algoritmes komen voor het generen van een contour. Een algoritme voor het afbakenen van het freesgebied en een algoritme voor het frezen. Momenteel is het algoritme een mix van de twee, het huidige algoritme kan gerefactored worden, zodat het optimaal is voor het begrenzen van een freesgebied.

Support
Over het plaatsen van een support ben ik heel erg tevreden, momenteel vindt de oriëntatie automatisch plaats – loodrecht op een vlak van de geometrie. Echter in sommige gevallen is dit ongewenst bijvoorbeeld als de support te steil omhoog steekt. Dit zou opgelost kunnen worden door de gebruiker na het plaatsen de mogelijkheid te geven om handmatig bij te stellen. Dit kan weer resulteren in een ander probleem, indien een support niet loodrecht op een vlak is geplaatst. Er is dan geen volledige hechting met het raakvlak, het ervoor zorgen dat er een betere hechting is, zou kunnen leiden tot meer ondersteuning. Dit zou bijvoorbeeld opgelost kunnen worden door de support een stuk in de geometrie te schuiven, of door geometrie te genereren aan de support die de ruimte opvult.

Oriënteren van DentalItem
Persoonlijk ben ik nog niet heel erg tevreden over het met de muis roteren van een DentalItem. Momenteel wordt dezelfde code gebruikt voor het roteren van de camera in DeskProto, het roteert een DentalItem nog niet altijd zoals je zou verwachten. Daarnaast is het ook handig als je een zijaanzicht en een onderaanzicht ziet tijdens het oriënteren.
Bij het onderaanzicht zou dan de optie ‘Show downward faces’ gebruikt kunnen worden, die tekent alle driehoekjes waarvan de normaalvector naar beneden wijst in een andere kleur. Het geeft aan waar er allemaal ondersnijding plaatsvindt.

Bronvermeldingen

Delft Spine Systems:
www.deskproto.com
Tandtechnische informatie:
http://www.dentalcomb.nl/behandeling.html
OpenGL website:
www.opengl.org
OpenGL RedBook:
http://www.glprogramming.com/red/
glLineWidth documentatie:
http://www.opengl.org/sdk/docs/man/xhtml/glLineWidth.xml
pbuffer extensie:
http://www.opengl.org/registry/specs/ARB/wgl_pbuffer.txt
TransformatieMatrices:
http://en.wikipedia.org/wiki/Transformation_matrix
Rotatiematrices:
http://en.wikipedia.org/wiki/Rotation_matrix
Feature extraction:
http://en.wikipedia.org/wiki/Feature_extraction

Dictaat: Wiskunde voor computergraphics

Boek: Computer graphics for Java Programmers
Auteur: Leender Ammeraal, ISBN: 0471981427

Bijlage 1 - Plan van aanpak

Bijlage 2 - Klasse diagrammen en code voorbeelden

Bijlage 1
Plan van aanpak

Inhoudsopgave

1. Achtergronden

1.1 Inleiding

Dit document beschrijft de afstudeeropdracht DpDental voor het bedrijf Delft Spline Systems. Delft Spline Systems is een klein bedrijf dat 3D CAM software ontwikkelt (het pakket DeskProto). Momenteel werken er 3 mensen: een CAD-CAM specialist tevens eigenaar en twee ontwikkelaars waarvan één part-time. Het is gevestigd in Utrecht (Tuindorp Oost), dat is dicht bij de Hogeschool Utrecht.

1.2 Wat is Deskproto

DeskProto is een 3D CAM Software pakket gericht op het snel vervaardigen van modellen (Rapid Prototyping). Rapid Protoyping is het converteren van een 3D CAD model naar een fysiek model, dit wordt gedaan op 2 verschillende manieren namelijk: door materiaal toe te voegen of door materiaal te verwijderen. DeskProto is gericht op de laatste manier - een model maken door materiaal te verwijderen. Dit heet CNC frezen – computergestuurd frezen. Op de website www.deskproto.com onder support zijn video`s hierover te vinden.

Met het programma kan een DeskProto-project worden aangemaakt. Een project bestaat uit Parts en Operations. Een Part legt vast welk deel van een model gefreesd moet worden, in een Operation kan met parameters worden ingesteld hoe een Part gefreesd moet worden. Als het te frezen werkstuk is vastgelegd kan het programma freesbanen berekenen. Een verzameling van freesbanen wordt opgeslagen in een NC file en kan dan naar een CNC freesmachine worden gestuurd.

1.3 Dentaal Systeem

Tegenwoordig kan ook het vervaardigen van een prothese voor in de mond (kroon of brug) door middel van CNC frezen. Dit wordt gedaan in tandtechnisch laboratoria die een dentaal CAD/CAM systeem bezitten. Een onderdeel van dat systeem is een 3D CAM programma. De software voor deze markt wordt momenteel gedomineerd door enkele dure pakketten. Omdat het zo duur is hebben enkel de grote bedrijven de mogelijkheid om deze dienst aan te bieden. Het is de bedoeling dat er straks met DeskProto een goedkoop alternatief beschikbaar komt, zodat het ook mogelijk wordt voor de kleinere tandtechnisch laboratoria om deze techniek toe te passen.

Om dit mogelijk te maken zijn er eerst enkele aanpassingen nodig voor DeskProto. Hiervoor is dus het volgende project bedacht: DpDental – Dentale versie van DeskProto. Samengevat is het doel van het afstudeerproject: alle vereiste aanpassingen uitzoeken en ontwikkelen voor DeskProto die nodig zijn zodat het gebruikt kan worden in tandtechnisch laboratoria.

2. Probleemstelling

2.1. Vervaardigen van een prothese

Om uit te zoeken welke aanpassingen nodig zijn voor DeskProto voordat het gebruikt kan worden als CAD/CAM pakket bij dentale systemen, ben ik met een collega op bezoek geweest bij Williams-Benelux - een tandtechnisch laboratorium. Tijdens dit bezoek zijn we rondgeleid en is er uitgelegd hoe het proces voor het vervaardigen van kronen met behulp van CNC frezen in elkaar steekt.

Een patiënt met tandbederf gaat eerst naar een tandarts. Als de tandarts besluit dat er een kroon nodig is wordt de tand of kies geslepen in een stompje. Vervolgens wordt er van het gebit een afdruk gemaakt. Dit kan door de patiënt te laten happen in een soort klei. Hiervan wordt dan een afgietsel gemaakt in gips. Vervolgens stuurt de tandarts het gipsmodel van het gebit naar een tandtechnisch-laboratorium.

De bedoeling is dat er op het stompje een kapje (kroon) of brug geplaatst gaat worden. Een brug wordt gebruikt indien er sprake is van een afwezige tand. En dient als constructie voor een kunsttand tussen 2 stompjes, waarbij op elk stompje een kroon komt met daartussen de kunsttand. Het vervaardigen van een kapje of brug gebeurt in een tandtechnisch-laboratorium.

In de eerste stap, het CAD gedeelte, wordt van het gipsmodel een digitale versie gemaakt door middel van een 3D scanner. Er zijn meerdere mogelijkheden voor dit proces, bijvoorbeeld door met een speciale scanner direct in de mond scannen. Vervolgens wordt met programmatuur een virtueel 3D-model van een kapje gemaakt. Het kan ook voorkomen dat er van een derde rechtstreeks een 3D-model wordt aangeleverd. Bij de tweede stap moet het kapje worden uitgefreesd – het CAM gedeelte. Als materiaal wordt een ronde schijf van Zirkonium gebruikt, ook wel een blank genoemd. Wat wel belangrijk om te weten is dat Zirkonium erg kostbaar is, daarom is het van belang dat er zo min mogelijk van wordt verspild.
Tevens heeft het materiaal de eigenschap dat het gaat krimpen indien het wordt gesinterd. Sinteren is een chemisch proces waarbij materiaalkorrels van bv. Zirkonium op een temperatuur worden gebracht waarop ze nog net niet smelten. Hierdoor groeien de contactpunten aan elkaar waardoor het contactoppervlak groter wordt, wat resulteert in heel hard materiaal.

Voordat er wordt begonnen met het frezen van één of meer kapjes, is het taak om de 3D-modellen van de kapjes zo efficiënt mogelijk bij elkaar te plaatsen binnen de blank, zodat er zo weinig mogelijk materiaal van wordt verspild bij het frezen. Tevens krijgt elk kapje peilers waarop het stabiel kan staan tijdens het sinteren. En er worden ‘Support-blocks’ toegevoegd, deze zijn nodig om ervoor te zorgen dat tijdens het frezen het werkstuk op zijn plek blijft zitten. Het belangrijkste gedeelte van het frezen is de preparatielijn, dit is het grensvlak rondom het kapje welke zo precies mogelijk moet aansluiten op het afgeslepen gedeelte van de tand of kies.

Bij de laatste stap, het sinteren, gaan de kapjes een oven in. Dan moeten de gefreesde kapjes op de peilers staan, zodat ze stabiel blijven tijdens dit proces. Doordat het materiaal gaat krimpen tijdens het sinteren, moet een model, voordat het wordt gefreesd, juist geschaald worden met de krimpfactor. Elke blank heeft weer zijn eigen krimpfactor. Voor zover het CAD-CAM aspect, na het sinteren is de kroon nog niet klaar want, er is enkel een kapje, het kapje wordt dan later verder verwerkt. Hoe dat proces werkt valt buiten de grenzen van dit verslag.

2.2. Verwachte aanpassingen

Hieronder volgt een lijst van de verwachte aanpassingen die nodig zijn aan DeskProto om het geschikt te maken als Dentaal systeem.

- Specificeren van Zirkoniumschijf
- Meerdere STL files beheren (positioneren, oriënteren en schalen)
- Support bridge toevoegen op vrij te kiezen plaatsen
- Eventueel peilers toevoegen
- Het te frezen gebied moet automatisch een freeform lijn rondom de geometrie zijn
- Ontwerpen van een gebruikersvriendelijke wizard
- Frezen van preparatielijn (extra)

Het moge duidelijk zijn dat het gaat om aanvullingen voor een bestaand programma (DeskProto), waarin veel van de benodigde functionaliteit al aanwezig is. Bijvoorbeeld het berekenen van freesbanen, laden en tekenen van geometrie.

2.2.1. Specificeren van Zirkoniumschijf

Specificeren van blanks, elke blank zal worden gespecificeerd aan de hand van bepaalde eigenschappen onder andere de krimpfactor, afmetingen en nummer of naam ter identificatie. Dit is vereist om 3D modellen(Kapjes) te positioneren en juist te schalen, zodat het de juiste afmeting heeft na het sinteren en om te onthouden waar al is gefreesd.
Het specificeren van materiaal in DeskProto gaat door middel van een Part, momenteel kunnen er enkel rechthoekige parts gebruikt worden, dit moet worden aangepast zodat het ook mogelijk is om cilindervormige parts te gebruiken.

2.2.2. Meerdere STL files beheren

Het moet mogelijk worden om meerdere 3D modellen (STL files) van kapjes te positioneren en te oriënteren op een blank. Hierbij moet rekening worden gehouden of er op een positie al gefreesd is of niet. Het oriënteren is van belang om zoveel mogelijk ondersnijding te voorkomen. Dit moet op een gebruiksvriendelijke manier omdat het bestemd is voor tandtechnici die weinig of niets weten van CNC frezen.

2.2.3. Support bridges en peilers

In DeskProto is de functionaliteit voor ondersteuning van support blocks beperkt, momenteel moeten de supports evenwijdig aan de X, Y of Z as georiënteerd zijn.
Verder heeft DeskProto de beperking dat er maar maximaal 4 support blocks kunnen worden aangemaakt. Als extra kan er een intelligente support block worden bedacht, deze variant krijgt een dynamische lengte afhankelijk van de afstand tussen het model en het materiaal. Deze intelligente supportblock kan dan gebruikt worden als peiler.

2.2.4. Automatische freeformlijn

Het te frezen gebied moet worden begrensd door een freeform lijn. Deze lijn is een, equidistante lijn er omheen (d.w.z. steeds op gelijke afstand van de buitencontour: namelijk de freesdikte). Deze lijn is ook nodig om de kronen binnen de blank te kunnen positioneren. Begrenzen met een freeform contourlijn is al wel mogelijk in DeskProto, echter het automatisch genereren van deze lijn niet.

2.2.5. Wizard

Het is ook de bedoeling dat er een wizard komt(een serie dialoog vensters). In de wizard moet dan stap voor stap een DeskProto project worden gemaakt dat geschikt is om kapjes te frezen uit Zirkonium. In samenwerking met een CAD-CAM specialist zal er een typisch frees plan worden bedacht dat in de meeste gevallen uitkomst biedt. Denk onder andere aan voor- en na frezen freeskeuze etc. Daarbij moet het freesplan ook instelbaar zijn, maar dat valt buiten de opdracht.

2.2.6. Preparatielijn

Omdat de preparatielijn zo belangrijk is wil men deze vaak afzonderlijk nafrezen, daardoor komt het vaak voor dat er een afzonderlijke file van de preparatielijn is geleverd. Het zou ook mogelijk zijn om deze automatisch te bepalen. De preparatielijn is dan gedefinieerd als een 3D polyline. Momenteel is er beperkte ondersteuning voor 3D polyline`s, het moeilijke is om een freesbaan te berekenen waarbij de buitenkant van de frees precies de preparatielijn volgt. De preparatielijn frezen is voor deze opdracht een bonus feature vanwege de complexiteit en de beperkte hoeveelheid tijd die beschikbaar is.

2.3 Verwachte vereiste vakkennis

Hieronder wordt globaal aangegeven hoe DeskProto momenteel is opgebouwd, zodat er een indruk kan worden gekregen over de vereiste vakkennis. Er wordt voornamelijk bedoeld de vakkennis die nodig is om überhaupt effectief met de DeskProto code te werken. Omdat ik er al enige tijd werk, ben ik al bekend met de meeste technieken en de huidige codebase van DeskProto.

- De hoofdtaal waarmee DeskProto is geprogrammeerd is C++ (Er wordt gebruik gemaakt van de Microsoft Visual C++ compiler)
- Voor de user-interface wordt gebruik gemaakt van de Microsoft Foundation Classes (MFC), een klassenbibliotheek die het werken met de Windows API vereenvoudigt
- De UI van de wizard werkt met WTL (Windows Template Library), vergelijkbaar met MFC echter template based
- Het tekenen van 3D modellen in het programma is gerealiseerd met OpenGL (Open Graphics Language)
- Het beheren van de code wordt met een versie-beheersysteem genaamd Subversion gedaan
- Voor het bijhouden van features/bugs met terugkoppeling wordt een bug-tracking systeem gebruikt genaamd Mantis

3. Projectdefinitie

3.1 Projectgrenzen

Dit project omvat het uitbreiden van het programma DeskProto met de aanpassingen die in paragraaf 2.2 zijn opgesomd. Het frezen van de preparatielijn kan extra worden gedaan indien er tijd overblijft.

Voor de duidelijkheid worden er enkele kernpunten genoemd die buiten het project vallen:

- Berekenen van de freesbanen
- Laden en tekenen van geometrie
- Framework voor een wizard

Daarbij zou het kunnen voorkomen dat er van de huidige codebase stukken code gerefactored moeten worden voordat ze herbruikbaar zijn.

3.2 Uitgangspunten

- Alle wijzigingen aan de codebase worden gedaan op een aparte Subversion Branch, pas als alles goed werkt wordt er samengevoegd.
- De afstudeerder moet redelijke kennis hebben van het gebruik van OpenGL en het gebruik van 3D transformatie matrices. Er gaat simpelweg veel gebruik van gemaakt worden.
- De benodigde aanpassingen worden beschouwd als deelprojecten
- Voor het testen van deelprojecten wordt Mantis gebruikt om de voortgang te volgen.

3.3 Randvoorwaarden

- De layout van de wizard wordt pas goedgekeurd indien de opdrachtgever erover tevreden is.
- Het doel is dat na afloop van dit project de software gebruikt kan worden in tandtechnisch laboratoria

3.4 Uitdaging

De grootste uitdaging zal waarschijnlijk liggen bij het oplossen van de volgende punten:

1. Meerdere STL files beheren (positioneren, oriënteren en schalen)
2. Support bridge toevoegen op vrij te kiezen plaatsen
3. Het te frezen gebied moet automatisch een freeform lijn rond de geometrie zijn

Voor de eerste twee punten zal voornamelijk het vak Wiskunde voor computergraphics terugkomen. Omdat het positioneren, oriënteren en schalen voornamelijk wordt gerealiseerd door middel van transformatie matrices.

De moeilijkheid van het derde punt is het juiste algoritme vinden en toepassen zodat er een geschikte contourlijn kan worden bepaald. Een mogelijkheid zou het gebruik van vision technieken kunnen zijn.

4. Projectactiviteiten

Hieronder volgt een opsomming van de uit te voeren activiteiten, deze zijn onderverdeeld in drie fases.

Oriënterende fase:

- Onzekerheden wegnemen
- Plan van aanpak opstellen
- Afstudeercontract tekenen

Ontwikkel fase:

- Documenteren (Scriptie)
- Wizard ontwerpen en implementeren
- Automatische freeformlijn bepalen
- STL files oriënteren en positioneren
- Supports plaatsen en oriënteren
- Dental projectfile

Afrondings fase:

- Uitloop
- Presentatie voorbereiden
- Afstudeerzitting

5. Producten

Aan het einde van het afstuderen moet er in de applicatie DeskProto een Dental wizard beschikbaar zijn met de volgende opties:

- Keuze voor een zirkonium blank (nieuw, of waarvan reeds een deel gebruikt)
- Laden van een kroon of brug (STL), oriënteren die (roteren voor een optimale freesbaarheid) en positioneren binnen in de blank.
- Voeg support bridges toe: meer dan vier mogelijk, met een conusvorm, op vrij te kiezen plaatsen.
- Het te frezen gebied (Operation segment) moet automatisch een freeform lijn rond de geometrie zijn.
- Maak freesbanen, waarbij het aantal operations en de operation parameters automatisch worden ingesteld (freesplan).
- Save dit als DpDental project, met informatie over de blank en welke delen al zijn opgebruikt.

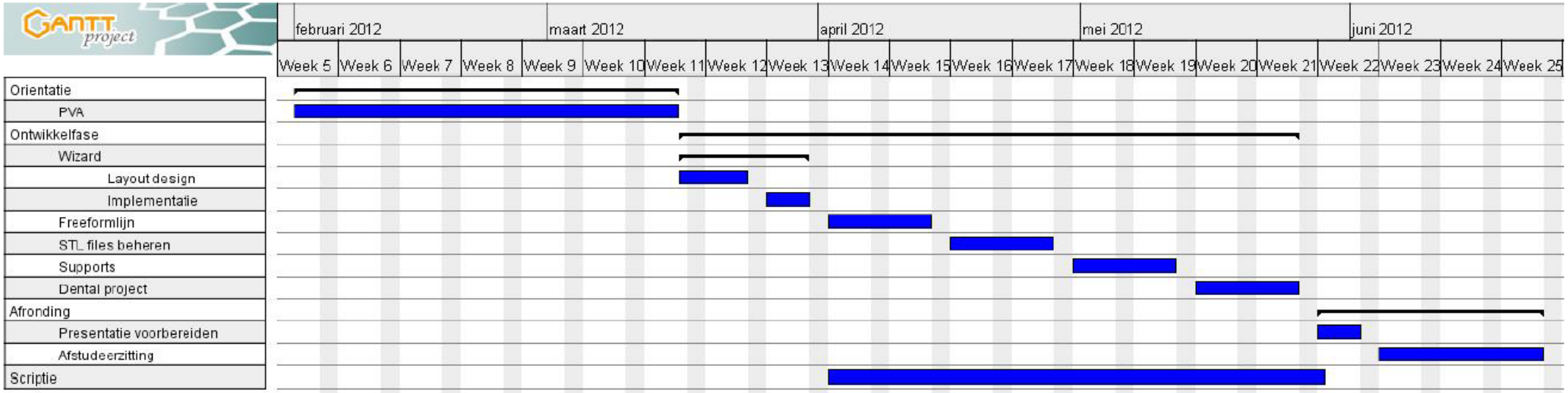
6. Betrokkenen

Begeleider:
Naam: De heer ir. Lex Lennings
Functie: CAD/CAM specialist
Mail: lennings@deskproto.com
Tel: 030-2965957

Docentbegeleider:
Naam: De heer Leo van Moergestel
Functie: Docent
Mail: leo.vanmoergestel@hu.nl
Tel: 06-10508351

Student:
Naam: Jelko Sluijs
Functie: Ontwikkelaar
Mail: jelko.sluijs@student.hu.nl
Tel: 06-11475710

7. Planning

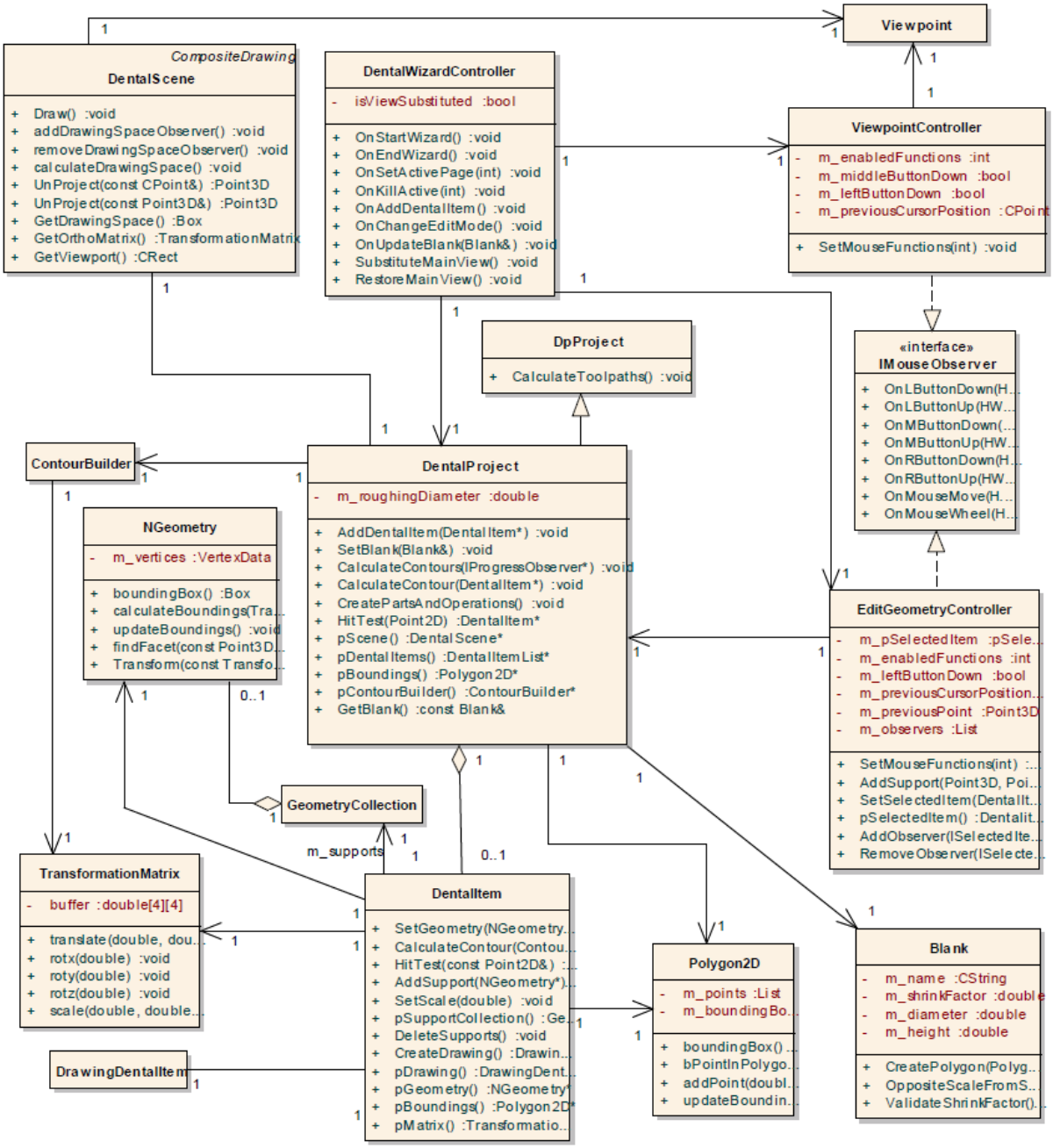

GANTT project
februari 2012
maart 2012
april 2012
mei 2012
juni 2012
Week 5
Week 6
Week 7
Week 8
Week 9
Week 10
Week 11
Week 12
Week 13
Week 14
Week 15
Week 16
Week 17
Week 18
Week 19
Week 20
Week 21
Week 22
Week 23
Week 24
Week 25
Orientatie
PVA
Ontwikkelfase
Wizard
Layout design
Implementatie
Freeformlijn
STL files beheren
Supports
Dental project
Afronding
Presentatie voorbereiden
Afstudeerzitting
Scriptie

Bijlage 2
Diagrammen en Code

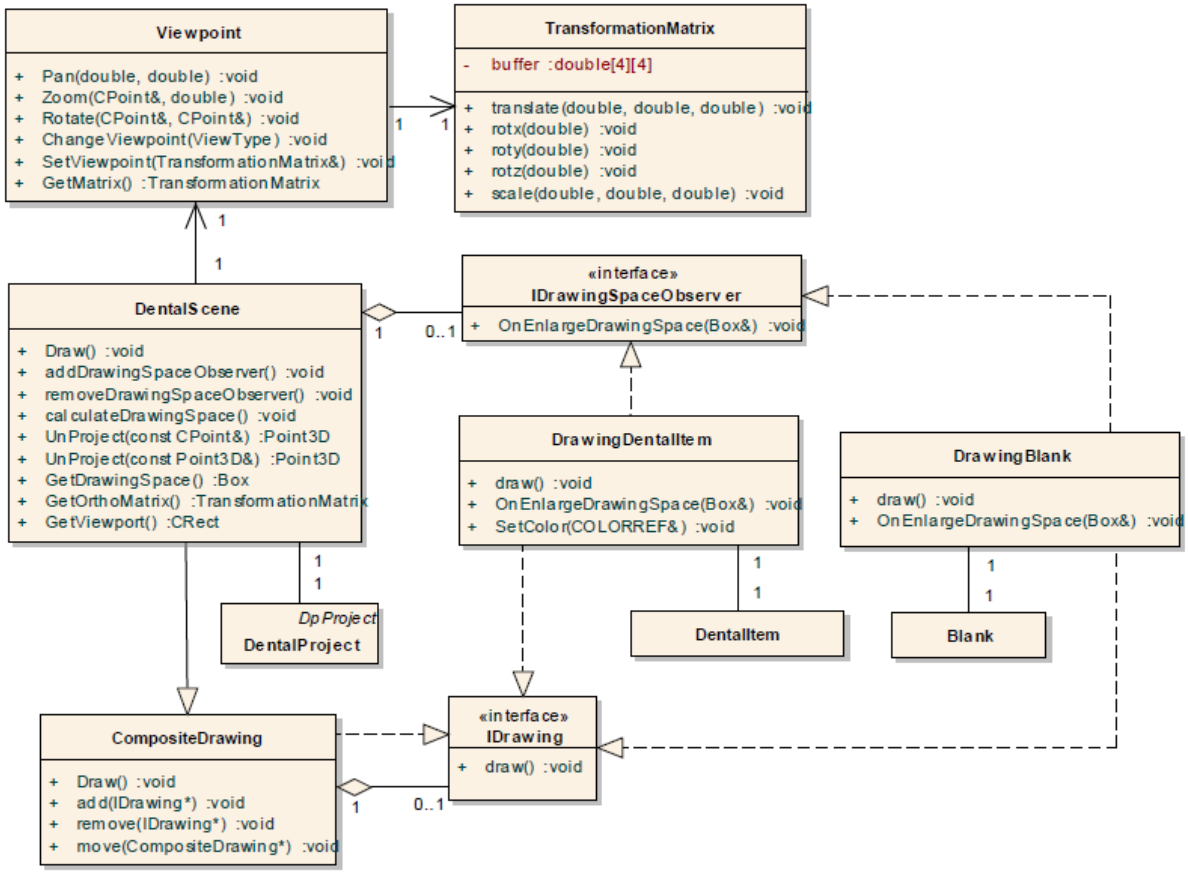
Inhoudsopgave

1. Class models

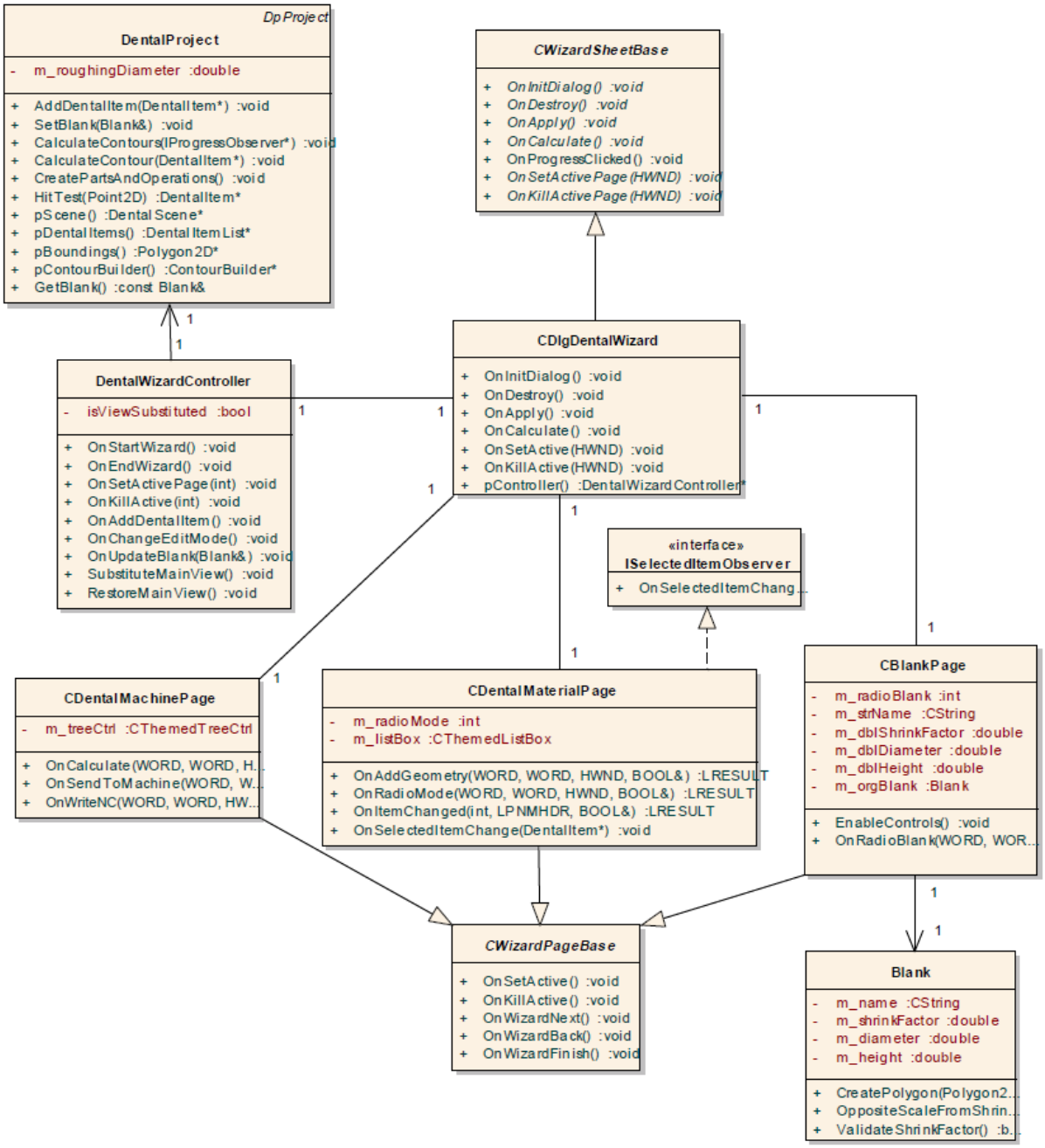
1.1 DpDental model



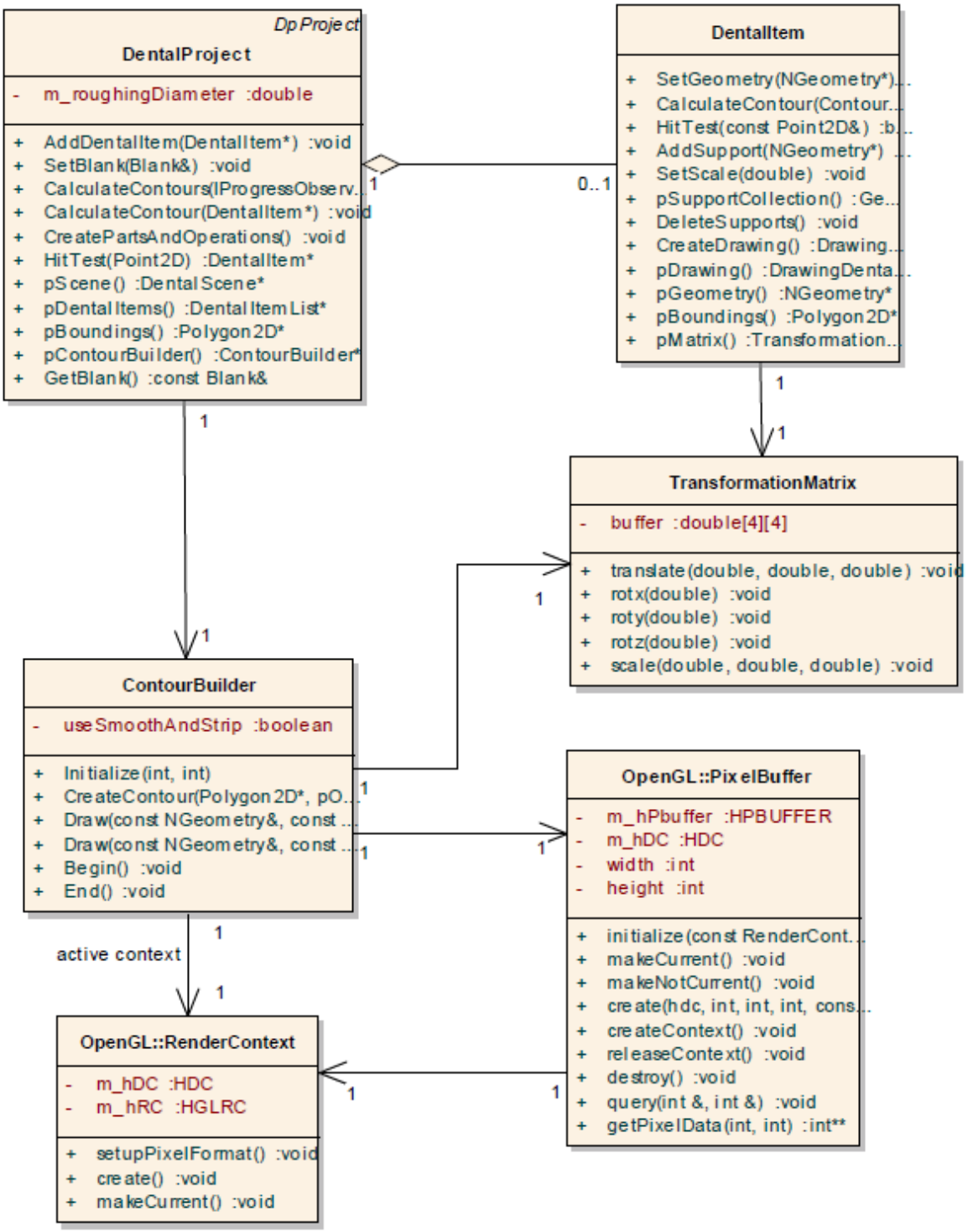
1.2 View model



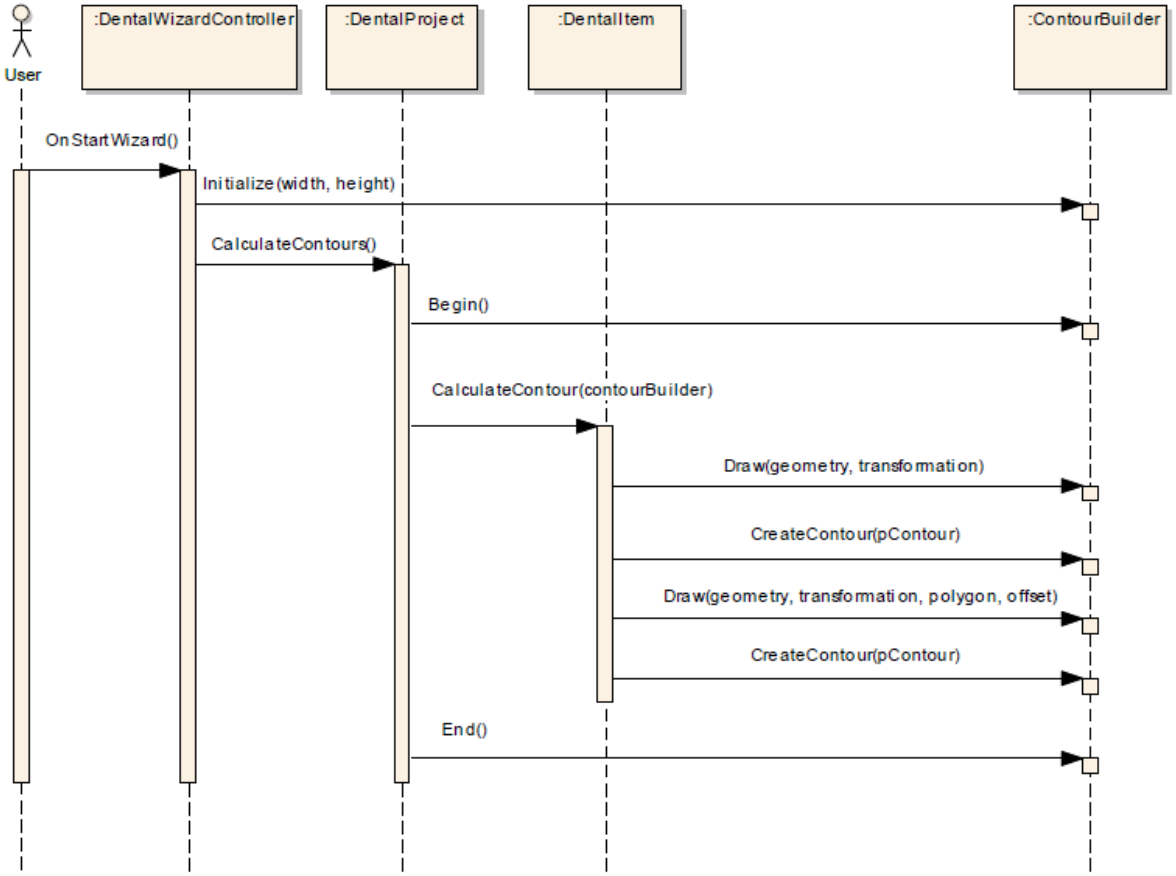
1.3 Wizard model



1.4 Contour model



1.4.1 Contour interaction model



2. Subversion log

Revision	Actions	Author	Date	Message
2595		**sluijs**	**vrijdag 22 juni 2012 15:43:35**	**- Supports are now removed after changing the orientation**
2594		sluijs	vrijdag 22 juni 2012 15:18:32	- It`s now possible to change the scaling and update the contour for a single dentalitem - Updating of dentalitems is now
2593		sluijs	dinsdag 19 juni 2012 12:05:25	- Renamed some methods - Contour drawing now always on top
2592		sluijs	maandag 18 juni 2012 14:36:31	- Coupled functionality to DentalItem listbox on the material page - Dentalitems can now be selected
2591		sluijs	donderdag 14 juni 2012 17:52:07	- Some refactoring
2588		sluijs	dinsdag 12 juni 2012 16:37:38	- Changes from trunk
2585		sluijs	dinsdag 12 juni 2012 16:01:09	- Updated the function which creates parts and operations from a dental project
2583		sluijs	dinsdag 12 juni 2012 10:58:01	- Coupled more functionality to wizard
2582		sluijs	maandag 11 juni 2012 12:24:59	- Updated layout of material page and machine page
2581		sluijs	maandag 11 juni 2012 10:03:53	- Removed some compiler warnings
2580		sluijs	vrijdag 8 juni 2012 17:31:26	- Updated the layout of the dental page - Added icons for the dental pages - Added dental machine page
2578		sluijs	donderdag 7 juni 2012 17:36:47	- DentalItem now has support collection - Supports with a certain orientation and length can now be added - Mousefuncti
2576		sluijs	donderdag 31 mei 2012 13:12:42	- Moved contour code to a class ContourBuilder - An contour offset in mm can now be specified
2575		sluijs	woensdag 23 mei 2012 12:00:33	- Merge fail!
2574		sluijs	woensdag 23 mei 2012 11:42:53	- Merged changes from trunk
2566		sluijs	vrijdag 4 mei 2012 16:57:49	- Added events to dental wizardcontroller which get called when a new page gets activated
2565		sluijs	vrijdag 4 mei 2012 14:53:57	- Added new page to wizard - Improved camera functionality, it is now possible to rotate
2561		sluijs	donderdag 3 mei 2012 9:21:54	- It's now also possible to load a DpDental projectfile
2560		sluijs	woensdag 2 mei 2012 15:34:52	- Added code to save and load a Dental Project
2559		sluijs	woensdag 2 mei 2012 9:30:13	- Functionality to position an item - Added code to unproject windowcoordinates to objectcoordinates - Added hittesting -
2536		jaspers	woensdag 25 april 2012 13:50:57	- gridcontourer lus-fix, stript nu ook punten op 1 lijn uit schuine lijnen
2535		sluijs	woensdag 25 april 2012 12:11:59	- Added code to change the viewpoint
2534		sluijs	maandag 23 april 2012 13:15:01	- Added a function which generates a contour
2533		sluijs	maandag 23 april 2012 12:00:08	- More missing files
2532		sluijs	maandag 23 april 2012 11:56:56	- Missing files
2531		sluijs	maandag 23 april 2012 11:43:26	- Added new Dental model classes - Added Dental related drawing code
2528		sluijs	dinsdag 17 april 2012 17:24:17	- Added a few controls to the blank page - Some key events in the modeless wizard did not work (related to issue #177)
2527		sluijs	vrijdag 13 april 2012 15:52:54	- Added a page to Dental wizard (CBlankPage)
2526		sluijs	vrijdag 13 april 2012 13:35:13	- Added a new wizard, Dental wizard - Created a new menu entry to show the Dental wizard
2525		sluijs	vrijdag 13 april 2012 10:12:05	- Created a framework to be able to edit geometry in the mainview
2521		sluijs	dinsdag 10 april 2012 17:05:45	- Merged changes from trunk
2510		sluijs	vrijdag 30 maart 2012 14:45:23	- Changes to be able to start the new Wizard modeless
2507		build	donderdag 29 maart 2012 10:5...	Branch for DpDental project

3. Code

3.1 Matrices

3.1.1. Berekenen van een SRT-matrix

```
/// <summary>calculates a matrix from all values in this transformset</summary>
TransformationMatrix TransformSet::calculateMatrix()
{
    TransformationMatrix m;
    m.translate(m_xTranslation, m_zTranslation, m_zTranslation);
    m.rotz(m_zRotation * DEG2RAD);
    m.roty(m_yRotation * DEG2RAD);
    m.rotx(m_xRotation * DEG2RAD);
    m.scale(m_xScale, m_yScale, m_zScale);
    return m;
}
```

3.1.2. Ontleden van een matrix

```
/// <summary>
/// decomposes the matrix to transform parameters (Matrix decomposition)
/// extracts the scalar, translation and rotation components from a SRT matrix
/// </summary>
void TransformSet::decomposeMatrix(const TransformationMatrix& matrix)
{
    // scaling
    m_xScale = matrix.getColumnVector(0).mag();
    m_yScale = matrix.getColumnVector(1).mag();
    m_zScale = matrix.getColumnVector(2).mag();
    // translation
    Point3D translation = matrix.getColumnVector(3);
    m_xTranslation = translation.x;
    m_yTranslation = translation.y;
    m_zTranslation = translation.z;
    // rotation
    TransformationMatrix orientation = matrix.getRotationMatrix();
    m_xRotation = orientation.xAngle() * RAD2DEG;
    m_yRotation = orientation.yAngle() * RAD2DEG;
    m_zRotation = orientation.zAngle() * RAD2DEG;
}
```

3.1.2 Rotatie matrices

```
/// <summary>a rotation matrix is any matrix that acts as a rotation in Euclidean space</summary>
/// <remarks>the matrix is an orthogonal matrix whose determinant is equal to 1 and its transpose equals its inverse</remarks>
/// @todo find better name because this method creates an orthogonal matrix basically
TransformationMatrix TransformationMatrix::getRotationMatrix() const
{
    TransformationMatrix rotation;

    rotation.setColumnVector( 0, getColumnVector(0).normalize() );
    rotation.setColumnVector( 1, getColumnVector(1).normalize() );
    rotation.setColumnVector( 2, getColumnVector(2).normalize() );

    ASSERT( rotation.isOrthogonal() );

    return rotation;
}

/// <summary>an orthogonal matrix is a matrix whose columns and rows are orthogonal unit vectors</summary>
/// <returns>true if orthogonal</returns>
bool TransformationMatrix::isOrthogonal() const
{
    bool isOrthogonal = false;

    // a matrix A is orthogonal if its transpose is equal to its inverse
    TransformationMatrix a(*this);  // copy
    a.transpose(); // transpose

    if ( a == inverse() ) // compare transpose with its inverse
        isOrthogonal = true;

    return isOrthogonal;
}

/// <summary>function to test if the matrix is a pure rotation matrix, a matrix which represents a proper rotation</summary>
/// <remarks>when the determinant equals -1 the matrix consists of a reflection and or a rotation also called an inproper rotation</remarks>
bool TransformationMatrix::isRotationMatrix() const
{
    return ( isOrthogonal() && doubleEquals( det(), 1.0 ) ) ? true : false;
}
/// <summary>gets the rotation angle around x in radians also called the yaw</summary>
double TransformationMatrix::xAngle() const
{
    return atan2(m_buffer[1][2],m_buffer[2][2]);
}

/// <summary>gets the rotation angle around y in radians also called the pitch</summary>
double TransformationMatrix::yAngle() const
{
    return -asin(m_buffer[0][2]);
}

/// <summary>gets the rotation angle around z in radians als called the pitch</summary>
double TransformationMatrix::zAngle() const
{
    return atan2(m_buffer[0][1],m_buffer[0][0]);
}

/// <summary>calculates a rotation matrix from a normal</summary>
TransformationMatrix TransformationMatrix::CalculateRotationMatrixFromNormal(const Point3D& normal)
{
    const Point3D worldUpVector( 0.0, 0.0, 1.0 );

    double xRot;
    double zRot;

    worldUpVector.getAimAngles( normal, xRot, zRot );

    TransformationMatrix orientation;
    orientation.rotz( zRot );
    orientation.rotx( xRot );

    return orientation;
}
```

3.2 Toevoegen van supports

```
void EditGeometryController::AddSupport(Point3D clickedPoint, Point3D objectCoordinate, const Facet& facet, Point3D normal)
{
    // obtain pointer to polygon for easy access
    Polygon2D* pContour = m_pSelectedItem->pBoundings();
    double supportLength = 0;

    // create a line which goes from the clicked point to a point which lies outside the contour,
    // which is used to find the intersection with the contour
    Point2D p1(objectCoordinate.x, objectCoordinate.y);
    Point2D p2(normal.x, normal.y);
    doubleEquals(p2.mag(), 1.0);
    p2 *= pContour->boundingBox().diagonalSize() / 2.0f;
    p2 += p1;

    // find intersections
    std::vector<Point2D> intersections;
    intersections.reserve(5);   // make the container big enough to hold at lease 5 elements (allthough most of the time 1 will be enough)
    pContour->findIntersections(p1, p2, intersections);
    // determine which intersection is nearest
    Point2D intersection;
    double nearestDistance = DBL_MAX;
    for (std::vector<Point2D>::iterator it = intersections.begin(); it != intersections.end(); ++it)
    {
        Point2D t = *it - p1;
        double distance = t.mag();
        if (distance < nearestDistance)
        {
            nearestDistance = distance;
            intersection = *it;
        }
    }

    // thus if an intersection was found we can calculate the support length, which will be the length from the clicked point to the contourline
    // taking into account its orientation
    if (intersections.size() > 0)
    {
        // calculate point on z-axis
        Point2D intersectionT = intersection - p1;
        Point2D n(normal.x, normal.y);
        double theta = atan2(normal.z, n.mag());
        double z = tan(theta) * intersectionT.mag();
        // create a 3D point and obtain its length (magnitude)
        Point3D a(intersectionT.x, intersectionT.y, z);
        supportLength = a.mag();
    }
    else
    {
        TRACE(_T("could not find intersection\n"));
        return;
    }

    // create and add support
    GeometryBuilder builder;
    SupportBuilder::LoadShape(SP_SHAPE_CONE, &builder);
    // convert to geometry object
    NGeometry* pSupport = builder.ToGeometry();
    TransformationMatrix orientation = TransformationMatrix::CalculateRotationMatrixFromNormal(facet.normal);

    SupportBuilder supportBuilder(nullptr);

    // calculate position
    clickedPoint += orientation.transform( Point3D(0.0, 0.0, 0.5 * supportLength) );

    // calculate transform matrix
    TransformationMatrix m ( supportBuilder.calculateMatrix( clickedPoint, 1, 1, supportLength, orientation, false ) );

    // set support properties
    pSupport->transform(m);
    pSupport->setShape(SP_SHAPE_CONE);
    pSupport->setMatrix(m);
    pSupport->setName( _T("dental_support") );
    pSupport->updateBoundings();
    pSupport->updateVertexBuffer();

    // add to support collection
    m_pSelectedItem->AddSupport(pSupport);
}
```