Virtuele realiteit

Project: Zeeslag

Davy Friedrich Mieke Haesen

18 maart 2003

Inhoudsopgave

Hoofdstuk 1

Opdracht

Het doel van dit project is om, in een groep van twee personen, een spel te ontwerpen in VRML (Virtual Reality Modelling Language). De keuze van het spel ligt vrij. Het moet enkel voldoen aan volgende vereisten :

- Er moet minstens 1 deelobject zijn in de scène dat verplaatsbaar is via de cursor of door middel van een script verplaatst wordt.
- Het project moet vlot draaien op een 'normale computer'.
- Er moet minstens 1 animatie inzitten via scripting.

We hebben hierbij geopteerd voor het maken van een 3D versie van het spel Zeeslag. De belangrijkste vereisten van het spel zijn als volgt ingevuld:

- *Minstens 1 deelobject in de scène moet verplaatsbaar zijn via de cursor of doormiddel van een script.*
  De bootjes en de helicopter kunnen bij het begin van het spel naar de juiste plaats van het bord worden gesleept met behulp van de muis. Verder verlaten de bootjes en de helicopter de haven via scripting.

- *Het project moet vlot draaien op een 'normale computer'.*
  Er is getracht de berekeningen en de rendertijd in het programma te beperken. Het middelste bord, waarop de sleutelpennen worden geplaatst is vereenvoudigd. Voor ieder vakje was er een geanimeerd blokje voorzien waarin zo'n sleutelpen geplaatst kon worden om aan te geven of het vorige schot al dan niet raak was. Omdat dit grote aantal blokjes echter teveel berekeningen met zich meebracht, zijn de blokjes achterwege gelaten, en is het middelste bord vervangen door een texture, met een rooster dat de vakjes aangeeft.
  Zowel de helicopter als de duikboot zijn voorzien van animaties, namelijk het draaien van respectievelijk de helicopterschroef, en de periscoop. Deze animaties worden echter enkel geactiveerd als het viewpoint dichtbij is. Dit is eveneens een maatregel om het programma niet nodeloos te vertragen.

- *Er moet minstens 1 animatie inzitten via scripting.*
  Het afvuren van bommen en raketten, het neerkomen van deze wapens in het water of op een doelwit, dat zich respectievelijk vertaalt in een „plons" en een „explosie" en het zinken van een boot of helicopter, zijn allemaal voorbeelden van animaties die verwezenlijkt werden met behulp van scripting.

Hoofdstuk 2

Gebruikte VRML Tools

2.1 Bestuderen van VRML

Voor het implementeren van het programma hebben we beroep gedaan op een aantal tools. In het begin waren natuurlijk vooral de tutorials een onmisbaar hulpmiddel:

- Floppy's web3D guide : een goede stap-voor-stap introductie tot VRML (`http://web3d.vapourtech.com/`)
- Web 3D Consortium : de internationale specificatie van het VRML formaat (`http://www.vrml.org/`)

2.2 Werken met VRML

Om de geïmplementeerde wereld weer te geven, gebruikten we in eerste instantie de Cosmo Player (`http://www.cai.com/cosmo/`). Hierbij was het echter minder interessant dat een kleine fout in de VRML-file, een crash van de browser teweegbracht. Om deze reden zijn we overgestapt naar de Cortona VRML Client 4.0 (`http://www.parallelgraphics.com/products/cortona/`). Deze plugin is hierop voorzien door een nuttige foutboodschap te geven, met soms zelfs de plaats waar de fout zich bevindt.
Als editor voor de code hebben we VrmlPad 2.0 gebruikt. (`http://www.parallelgraphics.com/products/vrmlpad/`). Grote voordelen bij deze tool zijn onder andere de syntax highlighting, auto-aanvulling, real-time foutdetectie, en vooral ook de debug-mogelijkheden.
Beide programma's zijn ontwikkeld door Parallelgraphics.

Tenslotte hebben we ook nog beroep gedaan op een hulpmiddel om samengestelde rotaties rond meerdere assen te berekenen, rechtstreeks toepasbaar via volgende url : `http://www.ddnet.es/personales/paulo/vrml/rotat.phtml`.

Hoofdstuk 3

Het spel Zeeslag

Voor degenen die nog niet vertrouwd zijn met Zeeslag, volgt hier een korte uitleg van het spel. Eveneens wordt een handleiding aangeboden voor onze versie van Zeeslag.

3.1 Doel van het spel

Iedere speler beschikt over een even groot watervlak, onderverdeeld in vakjes (15 bij 15), en een identieke vloot, bestaande uit :

- 2 Slagschepen (5 vakjes)
- 2 Kruisers (4 vakjes)
- 1 Schip (3 vakjes)
- 2 Duikboten (2 vakjes)
- 1 helicopter (1 vakje)

Het komt erop neer dat men de bootjes zo weet te schikken op het bord, dat de tegenspeler er zo lang mogelijk over doet om de ligging van de bootjes en de helicopter te achterhalen. Eenmaal alle bootjes geplaatst zijn, kan het spel beginnen. Om beurt mag een speler naar een vakje van de tegenstander schieten, en krijgt men als antwoord of er al dan niet een doelwit geraakt werd. Als er bovendien een boot of helicopter helemaal geraakt is, wordt ook dat meegedeeld. Zo speelt men verder tot een hele vloot „vernietigd" is. De speler die zijn tegenstander op die manier weet uit te schakelen, wint het spel.

3.2 Handleiding

In onze versie van Zeeslag word het traditionele gezelschapsspel in een 3D-wereld voorgesteld. De gebruiker kan dit spel spelen tegen de computer. Aan de hand van volgende stap-voor-stap handleiding, zou men zonder problemen zeeslag moeten kunnen spelen. Maar ook de tips onderaan het scorebord zouden de gebruiker een handje moeten helpen gedurende het spel. Op het scorebord wordt eveneens bijgehouden hoeveel bootjes elke vloot reeds verloren heeft.

1. Het spel wordt gestart door de file „zeeslag.wrl" uit te voeren.

2. Bij de aanvang van het spel dient men de bootjes op het bord te plaatsen. Dit doet men door volgende stappen voor alle bootjes en de helicopter te herhalen.

   (a) Klik op een bootje of op de helicopter, zodat deze de haven verlaat.

   (b) Nu kan men de muis over het wateroppervlak bewegen om het bootje te slepen. Wil men het bootje draaien, dan kan men dat doen door op het compas te klikken. Indien er plaats is om het bootje op zijn huidige positie te draaien, zal dit ook gebeuren, anders blijft zijn richting ongewijzigd. De helicopter kan niet geroteerd worden, omdat deze juist gericht moet blijven om bij een helicopter-aanval rond het bord te vliegen.

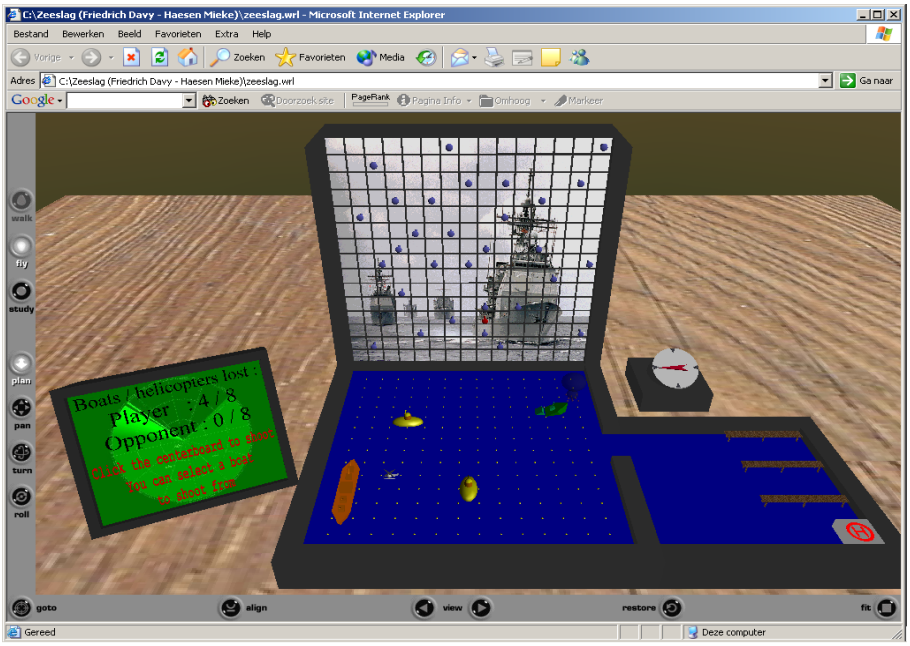
   (c) Heeft men de gewenste plaats voor het bootje gevonden, dan kan men deze vastleggen door op het bootje te klikken.

3. Als alle bootjes en de helicopter op het watervlak geplaatst zijn, verschijnt er een grote knop: „Done". Men kan de bootjes nog verplaatsen, maar eenmaal op de knop geklikt, worden de plaatsen van de bootjes definitief ogeslagen. Hierna kan men pas verdergaan met het spel.

4. Vervolgens zal de computer zijn bootjes plaatsen. Dit zal hij aangeven door middel van een denkballonnetje met de tekst : „Positioning boats".

5. Nu kan het spel echt beginnen. De speler krijgt de eerste schietbeurt, en kan nu beginnen raden waar zijn tegenspeler zijn bootjes heeft geplaatst. Dit kan men doen door een vakje op het middelste bord aan te klikken. Als er raak is geschoten, verschijnt er een rode sleutelpen in het vakje, anders een blauwe. Indien een doelwit helemaal vernietigd is, zal de speler hiervan op de hoogte gesteld worden door de computer. Merk op dat men kan kiezen vanop welk bootje een raket of bom wordt afgevuurd, door deze te selecteren alvorens men een vakje aanklikt. Als men hier de helicopter aanklikt, vliegt deze rond het bord om een bom te droppen. Indien één van de bootjes van de speler geraakt wordt, zal er een nummer boven komen zweven dat aangeeft hoeveel keer de boot reeds geraakt werd.

6. Het spel is beëindigd van zodra één partij alle bootjes en de helicopter verloren heeft. Als de vloot van de speler gezonken is, verliest hij. Als men de vloot van de tegenstander helemaal heeft kunnen vernietigen, wint men.



Figuur 3.1: Screenshot: Kruiser van de speler zinkt

Hoofdstuk 4

Problemen

Uiteraard verloopt het maken van een programma niet altijd vlekkeloos. Hieronder volgt een opsomming van de problemen die we zijn tegengekomen, en de oplossingen ervoor.

4.1 Het plaatsen van de bootjes en de helicopter

- **probleem:** Het slepen en plaatsen van een bootje via scripting was mogelijk voor één bepaald bootje. Maar hoe kan je het script gebruiken voor willekeurige bootjes? Hetzelfde probleem komt voor bij het uitvaren van de bootjes.

  oplossing: Aan elk bootje wordt een ID nummer toegekend. Als er op het bootje geklikt wordt, zal deze zijn eigen ID nummer uitzenden. Aan de hand van dit nummer wordt in het script de juiste eventOut gestuurd naar het juiste bootje.

- **probleem:** Bij het het plaatsen van de bootjes worden de bootjes over het watervlak bewogen, maar hierbij kan de rand van het bord overschreden worden. Eveneens konden boten op overlappende posities geplaatst worden.

oplossing: Collision detection met de randen van het bord werd toegevoegd, meerbepaald met de functie "testCollision". Deze functie werd nadien uitgebreid met collision detection tussen de boten onderling. Hierbij hadden we echter het probleem dat bepaalde bootposities wel geweigerd werden (de boot bewoog niet naar die positie), maar als dan op het water geklikt werd (om de boot vast te leggen), de onjuiste positie toch opgeslagen werd. Dit kwam doordat we de posities eerst aanpasten, en er pas daarna getest werd op collision. Dit probleem werd dan opgelost door een mogelijke positie eerst in een buffer te steken, en op basis hiervan te testen op collision.

- **probleem:** Als er op een bootje wordt geklikt terwijl het de haven uitvaart, blijft het op die positie liggen.

  oplossing: We onderscheiden voor elk bootje 4 toestanden : in de haven aan de steiger liggen, varen naar de uitgang van de haven, klaarliggen aan de uitgang van de haven, en op het watervlak liggen. Op basis van deze toestanden kunnen we in sommige gevallen het aanklikken negeren.

4.2 Het spel

- **probleem:** Soms bleek een bootje ineens te stoppen met zinken, nadat de pc met zijn helicopter had aangevallen.

  oplossing: Het veldje „raak", dat aangeeft welke boot geraakt werd, werd terug op -1 gezet als de helicopter van de tegenstander terug op zijn plaats aankwam. Op dat moment was het bootje echter nog niet volledig gezonken. Het zinkalgoritme probeerde vanaf dat moment dan echter bootje nummer „-1" te laten zinken, wat natuurlijk een ongeldig ID nummer is. Vermits de zinkanimatie enkel getoond wordt voor bootjes van de speler, en vermits de speler pas kan terugschieten nadat de zinktijd volledig is afgelopen, kon dit probleem worden opgelost door het aanpassen van het „raak" veld uit te stellen tot het moment dat er opnieuw geschoten wordt.

- **probleem:** We hebben geprobeerd de sleutelpennen in het centrale bord dynamisch aan te maken, m.a.w pas op het moment dat ze nodig zijn. We probeerden echter een eigen object aan te maken, een Extern-Proto. Dit is echter niet mogelijk in VRML.

  oplossing: De code voor de sleutelpen, werd dan maar in de file met het script toegevoegd. We maken daar dan gebruik van het browser object, meerbepaald diens functie „createVrmlFromString" om een nieuw object aan te maken.

- **probleem:** Een „explosie" of „plons", het plaatsen van een sleutelpen, en het zinken van een boot, moet bij een helicopteraanval vroeger gebeuren dan anders, namelijk op het moment dat de helicopter zijn bom dropt.

  oplossing: Er werd daarom een nieuwe functie afgesplitst, die precies voorziet in deze functionaliteit. Er werd daarnaast een extra timer voorzien, die half de cycleTime heeft van een volledige aanval. Als deze „halve" timer afloopt, wordt de bewuste functie aangeroepen. Bij een gewone aanval gebeurt dit pas nadat de „volledige" timer is afgelopen.

- **probleem:** De bootjes kunnen tijdens het spel nog altijd geroteerd worden.

  oplossing: Er werd een test toegevoegd of er wel geroteerd mag worden, afhankelijk van de fase waarin het spel zich bevindt.

4.3 Viewpoints

- **probleem:** Viewpoint binding aan het begin en het einde van het spel gebeurt niet, of te vroeg.

  oplossing: Er werden timers toegevoegd om de viewpoint bindings te starten.

- **probleem:** Viewpoint binding gebeurt in een flits. Onze interpolatie bij viewpoint binding, veranderde de coördinaten van een viewpoint, waardoor deze binding slechts éénmaal mogelijk is.

  oplossing: Het ingebouwde veldje „jump" in een viewpoint zorgt er automatisch voor dat er wordt geïnterpoleerd tussen de verschillende viewpoints.

4.4 Algemeen

- **probleem:** Omdat de helicopter bij elke aanval een andere route moet volgen, worden zijn interpolatiewaarden berekend en naar een interpolator gestuurd. Onze helicopter weigerde echter halsstarrig zijn route te volgen, ondanks het feit dat de coördinaten volledig in orde waren.

  oplossing: De fout bleek uiteindelijk te zitten in een klein hoekje : in onze interpolator hadden we door een typfout de keys [0 0.8 0.16 ...] staan. De volgorde is hier uiteraard verkeerd, waardoor de interpolator vreemde verschijnselen veroorzaakte.

Hoofdstuk 5

Werkschema

Gedurende de tien weken, hebben we de onderdelen volgens dit schema afgewerkt.

- **Week 1:** 6 januari 2003 - 12 januari 2003
  (Bekendmaking van de opdracht.)
  De groep samenstellen en een onderwerp kiezen.

- **Week 2:** 13 januari 2003 - 19 januari 2003
  Tutorials lezen.
  Een paar eerste objecten maken: een primitief zeeslag bord, en een bootje.

- **Week 3:** 20 januari 2003 - 26 januari 2003
  Tutorials lezen.
  Belangrijkste objecten afgewerkt: bord, bom, raket, helicopter en verschillende bootjes.

- **Week 4:** 27 januari 2003 - 2 februari 2003
  Tutorials lezen.
  Onze eerste scripts proberen : een draaiende helicopterschroef en bootjes op het bord slepen.

- **Week 5:** 3 februari 2003 - 9 februari 2003
  Tutorials lezen.
  Nieuwe objecten maken: sleutelpen en haven.
  Scripts : animatie op de periscoop van de duikboot, bootjes de haven uit laten varen, collision tussen bootjes en de randen van het bord.

- **Week 6:** 10 februari 2003 - 16 februari 2003
  Scripts : toevoegen van een „denkende" pc speler.

- **Week 7:** 17 februari 2003 - 23 februari 2003
  Scripts : „denker" afgewerkt, klikken op middelste bord vertalen naar vakjes, bij afvuren van wapens de route ervan bepalen, pc laten terugschieten, bootjes laten zinken, bootjes niet meer tekenen als ze gezonken zijn, helicopter laten rondvliegen bij schieten.

- **Week 8:** 24 februari 2003 - 2 maart 2003
  Nieuwe objecten toevoegen: rookpluim, fontein, compas en radar.
  Scripts: compas als knop voor rotatie, rookpluim of fontein laten zien als een bom of raket neerkomt, radar als scorebord laten fungeren, begin- en eindsequentie van het spel toevoegen met behulp van viewpoint binding en geluiden.

- **Week 9:** 10 maart 2003 - 16 maart 2003
  Code samenvoegen.
  Geluiden toevoegen.
  Het spel is klaar.
  Verslag schrijven.

- **Week 10:** 17 maart 2003 - 20 maart 2003
  Spel testen en kleine bugs eruit halen.
  Verslag afwerken.