OSM-S “RobotWorld” 2017-2018 sem 2

Bram Knippenberg Joost Kraaijeveld Jorg Visch

1 maart 2018

De opdracht bestaat uit twee delen. Het eerste, individuele, gedeelte wordt gevormd door het schrijven van een opstel over de architectuur en kwaliteit van code. Het tweede, gezamenlijke, gedeelte bestaat uit het schrijven van een programma. De basis van beide opdrachten wordt gevormd door de code van het programma RobotWorld.

Het eerste gedeelte van de opdracht is het schrijven van een beschouwing of rapport over de architectuur en kwaliteit van de gegeven code van RobotWorld. Het idee is dat zowel de architectuur als kwaliteit kritisch beschouwd worden en er een opbouwend en gebalanceerd oordeel en advies gegeven wordt dat bruikbaar is bij het aanpassen van het programma.

Het tweede gedeelte van de opdracht is het aanpassen en uitbreiden van RobotWorld. In RobotWorld moeten twee robots in overleg met elkaar zich een weg zoeken door een wereld. In de wereld staan diverse muren. Bij het begin van het programma hebben beide robots hun eigen variant van de wereld. Ze wisselen de gegevens over de wereld uit zodat ze beiden uiteindelijk een volledig beeld hebben van de wereld. De routes die beiden kiezen moet zodanig zijn dat ze elkaar gegarandeerd tegenkomen. Het is niet toegestaan dat de robots botsen of door muren heen rijden. Tijdens het rijden moet in beide applicaties de voortgang van beide robots in de wereld te zien zijn.

1 Het opstel (individueel)

Schrijf ieder een eigen essay van tussen de 1000 en 1400 woorden[1] over de architectuur en kwaliteit van code van “RobotWorld”. Het gaat hierbij om de code zoals in de `ROBOTWORLD_ROOT/src`-directory staat. De C++ standard library, Boost en wxWidgets mogen worden opgevat als externe packages en hoeven niet besproken te worden.

In het essay moeten onderstaande onderwerpen aan bod komen. Schrijf een echt essay, dus maak er geen “vraag-antwoord” uitwerking van, maar een doorlopend verhaal (eventueel verdeeld in paragrafen met eventuele tussenkopjes).

- Architectuur
  - Teken een package diagram en beschrijf elk package kort.
  - Vul de packages aan met de classes (en alles wat daarbij hoort). Natuurlijk alleen de classes van het project zelf, niet van de gebruikte externe librairies.
  - RobotWorld is volgens het MVC pattern opgebouwd (dat is een software architectuur pattern (Wikipedia, 2018)).

[1] Dit is ongeveer 2-3 pagina A4, als je “Helvetica” of “Times New Roman” met letergrootte 12 en regelafstand 1 gebruikt. De eventuele plaatjes kunnen het aantal bladzijden (wellicht aanzienlijk) uiteraard vergroten.

    - Zoek uit wat het MVC model inhoudt en beschrijf het kort.
    - In de Model en View packages zijn de models en views van MVC uitgewerkt. Beschrijf hoe de controller van het MVC-pattern is uitgewerkt in RobotWorld.
  - Welke software design patterns (Wikipedia, 2017) worden gebruikt? Beschrijf er minstens drie. Geef voor elk design pattern aan welke classes een rol spelen, welke rol dat is en hoe dat dan gerealiseerd is.
- Kwaliteit van code
  - Kies uit de JSF-styleguide (Lockheed Martin Corporation., 2005) minstens 5 "AV Rules" en leg uit waarom jij vindt dat deze horen bij de belangijkste regels uit de JSF-styleguide? Met *belangrijk* wordt niet bedoeld "het makkelijkst te beantwoorden", maar wat zijn zinnige patterns die de codekwaliteit ten goede komen (CamelCase is belangrijk, maar niet de belangrijkste bijv.).
  - Geef per, door jou hiervoor gekozen, regel aan of die wordt gebruikt in de code en in hoeverre die wel of niet goed is toegepast.

2 Het programma (tweetal[2])

Pas het programma RobotWorld zo aan dat er met twee robotjes, ieder op een aparte computer rijdend, door een gemeenschappelijke wereld gereden kan worden. Hierbij gelden de volgende eisen:

- Het moet mogelijk zijn om met één druk op een knop de (halve) wereld te vullen.
- Het moet mogelijk zijn om met één druk op een knop de twee werelden samen te voegen.
- Het moet mogelijk zijn om met één druk op een knop de robot een route te laten plannen en rijden.
- Tijdens het rijden moeten beide werelden gesynchroniseerd blijven. Dat wil zeggen dat op beide schermen de voortgang van beide robots in de wereld te zien moet zijn.
- De robotjes mogen niet door muren heen rijden of tegen elkaar aanbotsen.
- De werelden en routes moeten zo gemaakt worden dat de robotjes elkaar gegarandeerd tegenkomen.
- Het is niet toegestaan om twee programma's te schrijven. Eventuele configuratie moet gebeuren via argumenten meegegeven aan het programma.

Een aantal losse opmerkingen om jullie op weg te helpen:

- De code is voorzien van DoxyGen-commentaar en een DoxyGen-configuratiebestand.
- In de class MainApplication zitten diverse methoden om iets met argumenten te doen.
- De class WayPoint kan gebruikt worden om via-via routes te maken.
- Klik eens met je rechtermuisknop op het linker scherm om het popupmenu te zien.

[2] Het is de bedoeling dat je met iemand anders samenwerkt dan degene waar je mee samengewerkt hebt bij de algoritme-opdracht

- In de class Shape2DUtils zitten een aantal methoden die je kunnen helpen bij het vinden van muren en robotjes.
- Als je muren probeert te verplaatsen met de muis en je klikt ongelukkig vlak naast de vierkantjes of de lijn dan kan de applicatie crashen.
- Hoewel er opvallend veel kan met de applicatie is het geen gepolijste applicatie: er zitten hier en daar wat losse eindjes en wellicht werkt niet altijd alles zoals geadverteerd.
- Als je een bug tegenkomt dan moet je hem melden aan de docent en dan gaat die hem eruit halen. Dat geldt natuurlijk niet voor die bugs die je zelf hebt geïntroduceerd.
- Mocht je je meetkunde vergeten zijn: in beginsel kan je alles oplossen met de class Shape2DUtils of wxWidgets classes. Mocht je exotischere dingen willen dan kan je om hulp vragen bij de docent.

3 Inleveren

De uitwerking van RobotWorld (code en bijbehorend document) moet eind van deze periode (zie iSAS voor de exacte deadline) in iSAS worden ingeleverd. In overleg met de docent wordt een moment afgesproken waarop je een demonstratie geeft van de werkende applicatie.

Tijdens de demonstratie laat je zien dat je programma werkt. In de bijlage vind je een aantal voorbeeldsituaties waar je applicatie goed mee om dient te gaan.

4 Software

4.1 Boost installatie

RobotWorld maakt gebruik van de Boost libraries. Zorg er voor dat een recente versie van boost is geïnstalleerd. Zie de OSM-Software Studentenhandleiding voor de installatieprocedure.

Let op: RobotWereld maakt gebruik van een *makefile* configuratie voor het compileren van RobotWorld. Deze kan niet omgaan met boost-dll bestanden waar de compiler- en boostversienummers in de naam zijn opgenomen. Compileer zonodig de bestanden (opnieuw) door `--layout=tagged` als extra optie aan de b2 (Boost.Build systeem) commandoregel toe te voegen.

4.2 wxWidgets installatie

RobotWorld maakt gebruik van de wxWidgets libraries. Zorg er voor dat een recente versie van wxWidgets is geïnstalleerd. Zie de OSM-Software Studentenhandleiding voor de installatieprocedure.

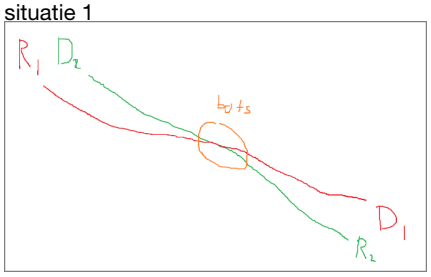
4.3 RobotWorld installatie

De source-code die als basis voor de RobotWorld opdracht wordt gebruikt is aangeleverd in een zip-bestand. Compilatie gaat in dit geval met behulp van een Autotools/makefile-systeem. Maak hiervoor een c++-project aan met als projecttype een *makefile project* in plaats van een *executable*. Neem de in de zip aangeleverde basiscode op in dit project. Compilatie-instructies zijn te vinden in het README bestand in de RobotWorld sources.

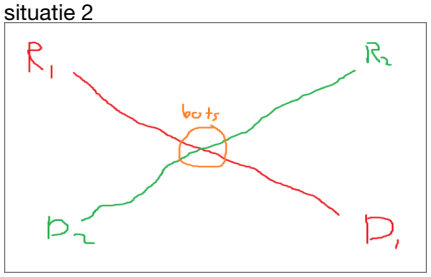
5 Bijlage

5.1 RobotWorld voorbeeldsituaties

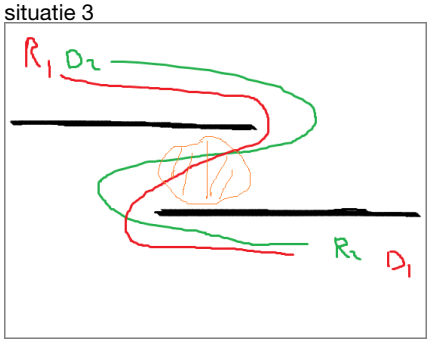
situatie 1



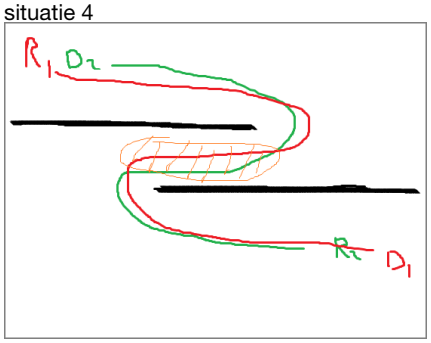
situatie 2



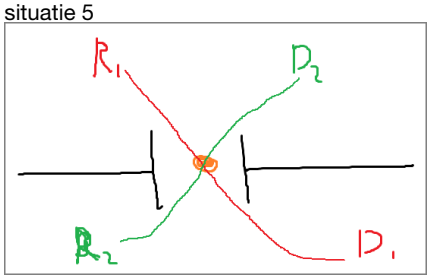
situatie 3



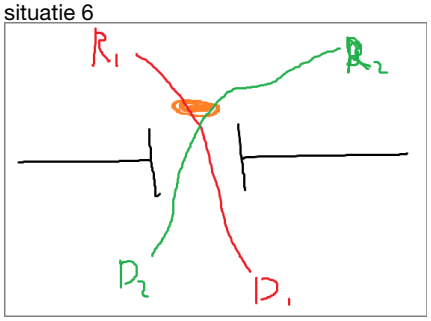
situatie 4



situatie 5



situatie 6



Referenties

Lockheed Martin Corporation. (2005). *Joint Strike Fighter - Air Vehicle - C++ Coding Standards*. Geraadpleegd van www.stroustrup.com/JSF-AV-rules.pdf

Wikipedia. (2017, november 22). Design Patterns - Wikipedia. Geraadpleegd van https://en.wikipedia.org/wiki/Design_Patterns#Patterns_by_Type

Wikipedia. (2018, februari 13). Architectural pattern - Wikipedia. Geraadpleegd van https://en.wikipedia.org/wiki/Architectural_pattern