P&O Computerwetenschappen 2012-2013
Semester 1

1 oktober 2012

Deel I: Praktische informatie

1 Context

De opdracht die in dit geïntegreerd practicum wordt uitgewerkt, kadert in het domein van autonome intelligente robots. Meer bepaald zullen we aan de hand van zelf te bouwen en te programmeren Lego Mindstorms robots een aantal taken uitvoeren. Elk team studenten zal trachten een oplossing te vinden voor dezelfde opdracht, maar de oplossingen die elk team zal voorstellen zullen uiteraard verschillend zijn. Dit project omvat verschillende componenten:

1. Het bouwen van een fysieke robot m.b.v. Lego Mindstorms;
2. Het programmeren van deze robot en het programmeren van software op een sturende PC zodat de robot in staat is om het onbekende traject af te leggen;
3. Het rapporteren van verschillende tussenresultaten en de vooruitgang van het project aan de hand van demo's en verslagen, alsook een finale presentatie van het volledige project op het einde van het semester.

Uiteraard is het niet mogelijk om state-of-the-art robots en bijhorende besturing te ontwikkelen in de loop van één semester. We willen jullie echter wel kennis laten maken met een aantal van de onderliggende concepten en bijhorende programmatie- en planningsprocessen.

2 Didactisch team

Docenten: Hendrik Blockeel, Erik Duval en Raf Vandebril.
Assistenten: Rutger Claes; Jeroen De Vlieger; Micol Ferranti; Jelle Peeters; Arun Ramakrishnan; Steven Van Acker; Roel Van Beeumen; Zubair Wadood Bhatti.

3 Structuur van het practicum gedurende het semester

Het practicum bestaat uit één begeleide sessie per week. Aanwezigheid op deze sessies is verplicht. Indien je wegens overmacht of ziekte niet aanwezig kan zijn, meld je dit aan je groepscoördinator en aan prof. Raf Vandebril (Raf.Vandebril@cs.kuleuven.be).

Naast het werk dat verricht wordt in de sessies wordt er ook verwacht dat je buiten de sessies werkt aan de taken die aan jou zijn toevertrouwd. Op basis van de 9 studiepunten worden er iets meer dan 6 uur extra werk per week per student buiten de sessie verwacht.

4 Teamwerking

4.1 Samenstelling team

Elk team is samengesteld uit vijf of zes studenten. Deze samenstelling wordt door de docenten vastgelegd aan het begin van het semester en wordt tijdens de eerste sessie meegedeeld[1].

4.2 Subteams

Voor de uitvoering van het gehele project kan het noodzakelijk zijn dat er subteams gevormd worden, die elk een onderdeel van het project zullen uitwerken. De taakverdeling kan vrij gekozen worden binnen het team in onderling overleg.

4.3 Teamcoördinator

Gedurende de eerste sessie duidt elk team in onderling overleg een teamcoördinator aan. De functies van de coördinator (naast het eigen werk in het team) zijn de volgende:

- De coördinator is de voornaamste contactpersoon tussen het team en het didactisch team (professoren en assistenten).
- De coördinator brengt het didactisch team op de hoogte in geval er zich problemen voordoen binnen het team die een invloed kunnen hebben op de goede teamwerking. Bv. langdurige ziekte van een medestudent, regelmatige afwezigheden, conflicten die een vlotte samenwerking in de weg staan.
- De coördinator is ook verantwoordelijk voor het coördineren van de verschillende subteams en zorgt dat iedereen zich houdt aan de opgelegde planning, zodat het project binnen de opgelegde tijd beëindigd wordt.
- De taken van de teamcoördinator dienen mee in beschouwing genomen te worden bij het verdelen van het feitelijke projectwerk in de subteams.

4.4 Secretaris

Gedurende de eerste sessie duidt elk team in onderling overleg een secretaris aan. De functies van de secretaris (naast het eigen werk in het team) zijn de volgende:

- De secretaris is verantwoordelijk voor het ordelijk bijhouden van alle documenten en verslagen m.b.t. het project, en dit zowel in hardcopy formaat als elektronisch.
- De secretaris zorgt er voor dat er steeds een up-to-date versie is van de planning en het verslag (samen met de groepscoördinator). Zie ook Deel III voor meer informatie betreffende het verslag. De secretaris is ook de eerste verantwoordelijke om de gevraagde verslagen samen te stellen en in te dienen. Dit wil niet zeggen dat de secretaris de inhoud van alle verslagen zelf schrijft, maar wel dat hij zorgt dat elk verslag een coherent geheel vormt.
- De secretaris houdt tevens een overzicht bij van hoeveel tijd (in uren) elk lid van de groep bijgedragen heeft tot het project. Dit overzicht dient om de bijdrage van elk teamlid individueel bij te houden, en tevens om de totale werkhoeveelheid gepresteerd door het volledige team te meten. Opmerking: per teamlid wordt iets meer dan 11 uur werk per week verwacht (incl. de wekelijkse verplichte sessie).
- De taken van de secretaris dienen mee in beschouwing genomen te worden voor het verdelen van het feitelijke projectwerk in de subteams.

[1] Eventueel kunnen, met onderlinge toestemming, studenten van team wisselen. Dit kan echter enkel gebeuren tijdens de eerste sessie van het semester en dient persoonlijk gemeld te worden aan prof. Raf Vandebril.

5 Planning

Er wordt verwacht dat er steeds een algemeen planningsdocument is dat de status en vooruitgang van het project laat zien. Dergelijke planning is voornamelijk een tool om bottlenecks en eventuele problemen, die een negatieve invloed kunnen hebben op het op tijd beëindigen van het project, te detecteren.

In principe is het gebruik van planningstools reeds in vorige P&O's aan bod gekomen: taakstructuur & verantwoordelijkheidsstructuur, Gantt-chart, teamkalender, enz. Maak gebruik van je ervaring uit vorige P&O's om een functionele en nuttige planning uit te werken.

De secretaris houdt het planningsdocument up-to-date, de teamcoördinator zorgt ervoor dat iedereen zich aan de planning houdt en lost coördinatieproblemen op waar nodig.

Het kan tevens nuttig zijn om binnen een enkele sessie een goede werkmethode te hebben:

- Zit samen met de ganse groep aan het begin van de sessie om te overlopen wat iedereen die week plant te doen.
- Op het einde van elke sessie kan je dit herhalen, waarbij de secretaris volgende puntjes kan noteren: Wie heeft deze sessie waaraan gewerkt? Welke resultaten zijn er deze sessie bereikt? Wat wordt er gepland in de loop van de week? Welke zijn de belangrijkste problemen en aandachtspunten?

6 Begeleiding

6.1 Begeleiders

Elk team beschikt in principe over twee vaste begeleiders van het didactisch team. De begeleiders volgen het project op en gaan na of de gemaakte vorderingen het toelaten om het project binnen de tijdspanne van een semester klaar te krijgen. Tijdens elke begeleide sessie zal deze begeleider langskomen om de vooruitgang te bespreken.

De begeleider dient voornamelijk om vragen aan te stellen m.b.t. planning en uitvoering van het project. Hij zal het project niet voor jou maken.

6.2 Hoe problemen bespreken?

- Voor kleine problemen die zich tijdens een sessie zelf voordoen, kan je rechtstreeks één van de aanwezige begeleiders aanspreken.
- Voor problemen die betrekking hebben op de inhoud of planning van het project zelf, spreek je best je eigen vaste begeleider aan. Indien hij niet in de sessie zelf aanwezig is, aarzel dan niet om hem telefonisch of per email te contacteren.
- Belangrijke problemen of problemen van grotere aard meld je best voor de sessie (per email) aan je begeleider, zodat hij ook tijd heeft om eventueel naar een oplossing te zoeken. Je kan niet verwachten dat uitgebreide problemen ogenblikkelijk ter plaatste kunnen opgelost worden.
- Wees je ervan bewust dat problemen oplossen een inherent deel vormt van dit vak, cfr. de titel van het vak.

6.3 Wat kan je niet verwachten?

Je bent in eerste instantie zelf verantwoordelijk voor het beheersen en oordeelkundig gebruik van verschillende software-tools die in het project zullen gebruikt worden. Je mag niet verwachten dat de begeleiders een technische helpdesk vormen die allerlei kleine probleempjes oplossen. We verwachten dat je zelf eerst een poging onderneemt om dit soort problemen weg te werken.

De begeleider zal ook geen rol spelen in het maken van keuzes; elke ontwerpkeuze moet gewikt en gewogen worden door het team zelf. Wel zal de begeleider vaak zeer kritische vragen stellen om te toetsen of de keuze gefundeerd is.

7 Evaluatie

De evaluatie van dit vak zal gebeuren op basis van de volgende criteria:

- Kwaliteit en inhoud van tussentijdse verslagen en demonstraties.
- Kwaliteit en inhoud van de eindpresentatie en eindverslag.
- Functionaliteit en correctheid van de ontwikkelde applicatie.
- Kwaliteit van het ontwerp van de ontwikkelde software.
- Individuele evaluatie van elke student, op basis van prestaties gedurende het semester en peer-assessment. Dit impliceert dat elke student een individuele kwotering krijgt.

8 Technische ondersteuning voor Lego Mindstorms & leJOS

8.1 Hardware

De LEGO mindstorms kit bestaat, naast een rijke set aan gebruikelijke LEGO-bouwblokjes, uit de volgende drie hoofdcomponenten:

- **Een programmeerbare unit**, de 'NXT Brick', een mini-computer met volgende specificaties: Amtel (c) 32-bit ARM @ 48MHz 64 KiB RAM + 256 KiB flash 64 x100 pixel LCD matrix display 3 uitgangspoorten voor de motoren, 4 ingangspoorten voor de sensoren. Je kan ofwel een USB-poort gebruiken om programma's op de brick te zetten, ofwel ze wireless uploaden via bluetooth.

- **De actuatoren**
  Servo motoren: drie motoren waarvan de snelheid programmatorisch gecontroleerd kan worden. Op elk gewenst moment kan de positie (hoek) van elke motor opgevraagd worden (proprioception); dit wordt mogelijk gemaakt door een ingebouwde sensor, die het gedraaide aantal graden intern bijhoudt.
  Luidspreker: bevindt zich op de brick en kan een gewenste frequentie gedurende een gewenste tijd afspelen.
  LCD display: bevindt zich op de brick, vooral handig om debug informatie te tonen.
  LED's: drie LED-lampjes die je kan aansluiten op één van de motorpoorten en die je kan omsluiten met een rood, groen of geel omhulseltje.

- **De sensoren**
  Druksensor: dit is een eenvoudige aan/uit schakelaar, nuttig voor bijvoorbeeld een bumper.
  Lichtsensor: sensor die lichtintensiteit (ook IR) detecteert. De lichtsensor kan op zich dus geen kleuren zien, enkel kleuren onderscheiden op basis van hun lichtintensiteit. De sensor heeft twee modi: actief (geeft zelf licht) of passief.
  Geluidsensor: detecteert geluidssterkte, dit kan zowel in dB als in dBA (adjusted decibels, enkel geluid waarneembaar door het menselijk oor) meten. Opnemen van geluid is niet mogelijk.
  Afstandsensor: dit is een ultrasone sensor die afstanden tot 255cm kan meten. De sensor is betrouwbaar van 6 tot 180cm met een nauwkeurigheid van ca. 3cm. De zogenaamde sonar conus opent met een hoek van 30 graden, wat wil zeggen dat op 180cm de conus 90cm breed is.

8.2 Software: leJOS & Eclipse

Eclipse Op de machines in de PC-klassen staat Eclipse reeds voorgeïnstalleerd. Het commando om Eclipse op te starten is `$ eclipse`. Als je dit echter uitvoert in een console, zal je zien dat je met behulp van de -data parameter nog de naam van de workspacedirectory moet meegeven waarin je wil dat Eclipse projectgegevens zal opslaan. Kies hiervoor een zinvolle naam.

Omdat we in Java gaan programmeren, zet je best de Java Perspective van Eclipse aan. Dit doe je door in Window > Open Perspective de Java optie te kiezen.

Als je Eclipse ook thuis of op je kot wil gebruiken, moet je de installatie uiteraard zelf doen. Omdat Eclipse een programma is dat op Java draait, kan dit zowel onder Windows als Linux. Bijvoorbeeld voor Linux kan de installatie gewoon onder je homedirectory gebeuren. Download hiervoor de binary file van http://www.eclipse.org en ga te werk op een gelijkaardige manier. Let er wel op dat je de nodige plugins (bijv. subclipse) ook zelf zal moeten installeren.

SVN plugin voor Eclipse: Subclipse Om van svn gebruik te maken binnen eclipse, wordt de plugin Subclipse voorzien. En aangezien een hulpmiddel als svn onmisbaar is om te collaboreren in groep wordt iedereen geacht hoofdstuk 1, fundamental concepts, en 2,basic usage, van het svnboek te lezen. (http://svnbook.red-bean.com/en/1.4/index.html)

In de computer labo's is deze reeds voorgeïnstalleerd. Indien je thuis, op kot, of op de eigen laptop werkt zal dit natuurlijk zelf moeten installeren. Uitgebreide instructies zijn te vinden op de officiële website: http://subclipse.tigris.org.

Selecteren van de juiste JRE onder Eclipse Het toevoegen van een JRE gebeurt onder Eclipse als volgt: Window > Preferences ... > Java > Installed JRE's > add ... hier kan je ook instellen welke JRE er standaard gebruikt moet worden.

Het toewijzen van een Java Compiler voor een bepaald project doe je op volgende wijze: Project > Preferences > Java Compiler Vink het vakje 'Enable project specific settings' en zet het 'Compliance Level' op de ondersteunde Java versie.

leJOS Om programma's te schrijven die door de NXT brick kunnen worden uitgevoerd, zullen we de JAVA-omgeving leJOS (spreek uit: lechos) gebruiken. Een uitgebreide tutorial van leJOS vind je onder de link

```
http://lejos.sourceforge.net/nxt/nxj/tutorial/
```

en is verplichte lectuur. De volledige interface (API) kan je ook op deze site vinden en kan je best direct bookmarken in uw browser.

Initialisatie: De laatste leJOS-versie staat reeds geïnstalleerd in /localhost/packages/lejos. Deze file bevat de programma's om JAVA/leJOS-bestanden te compileren, linken en te verzenden naar de NXT brick. Om ervoor te zorgen dat al deze bestanden in je pad staan, voeg je volgende lijn toe in je .bashrc bestand (eventueel moet je dit bestand zelf aanmaken):

```
source /localhost/packages/lejos/bash_lejos
```

Het is *niet* de bedoeling dat jullie de brick flashen, i.e. `$ nxjflash`. Hiervoor heb je admin rechten nodig die als student niet beschikbaar zijn in de computer labo's. Indien nodig kan je dit aan een assistent vragen[2].

Compile, link & run via command line: Als je nu een nieuwe console opent, zou je het commando nxj moeten kunnen uitvoeren. Om leJOS te testen kan je een voorbeeldprogramma compileren, linken, uploaden en runnen (bvb. Tune). Sluit de brick aan op de computer via USB en zet hem aan. Voer daarna deze commando's [3] uit:

```
mkdir Pen0
cd Pen0
cp -R /localhost/packages/lejos/lejos_nxj/projects/samples .
cd samples/Tune
nxjc Tune.java
nxj -u Tune
```

[2]In de eerste sessie zal een assistent langskomen om de brick te flashen als dit niet reeds gebeurd is.

[3]zorg dat je weet wat deze commando's doen, bekijk bvb de tutorial en hun help info

Zorg dat de brick aanstaat anders zal je geen bestanden kunnen uploaden. Je kan je geüploade programma nu uitvoeren. Selecteer daarvoor op de brick Files > Tune.nxj en daarna Execute Program.

Compile, link & run m.b.v. ant buildfiles: Er is ook een eenvoudigere manier om alles te compileren en naar de brick te sturen door gebruik te maken van voorgedefinieerde ant-scripts. In elke directory van elk voorbeeld vind je een build.xml bestand die door ant gebruikt kan worden om acties uit te voeren.

Uitvoeren doe je in een console. bvb:

```
cd ~/PenO
cd samples/HelloWorld
ant
```

Je kan ook specifiek ant taken uitvoren door deze op de cmdline te vermelden. Als je geen taak opgeeft wordt de default taak uitgevoerd.

Ook via eclipse kan je deze ant scripts uitvoeren. In de package explorer dubbelklik je op het build.xml bestand. Dan komen rechts de taken te staan die gedefinieerd zijn. Via de rechtermuisknop kan je deze taken uitvoeren.

Het is *zeer* sterk aangeraden om een compileer en upload proces vast te leggen in bvb een makefile gebruik makende van de nxjc en nxj commando's of een ant-script (laat je inspireren door de voorbeeld ant-scripts).

8.3 JAVA Threads

Een goede tutorial is te vinden op: http://www.freejavaguide.com/java-threads-tutorial.pdf

8.4 Bluetooth (verbinding robot-PC)

Lees grondig volgend document.
http://lejos.sourceforge.net/nxt/nxj/tutorial/Communications/Communications.htm

Deel II: Opgave

1 Robot

De robot wordt gebouwd met Lego-blokjes, gebruik makende van Lego Mindstorms. Informatie i.v.m. Lego Mindstorms is te vinden op http://mindstorms.lego.com/. Een Google-search kan je tevens extra informatie opleveren i.v.m. Lego Mindstorms projecten die reeds aan andere universiteiten werden verwezenlijkt.

De fysieke vorm van de robot (wielaandrijving, plaats sensoren, e.d.) is volledig vrij te kiezen. Het design van je robot zal uiteraard een invloed hebben op de nauwkeurigheid waarmee de robot kan bestuurd worden. De software voor de robot wordt ontwikkeld a.d.h.v. leJOS, een JAVA-implementatie voor Lego Mindstorms.

Enkele interessante links:

- http://mindstorms.lego.com/
- http://lejos.sourceforge.net/
- http://www.aaai.org/AITopics/html/autveh.html
- http://www.robocup.org/

2 Probleemstelling

> *Ontwikkel een robot die in staat is om een doolhof te verkennen en hierdoor te navigeren.*

In deze sectie zal het einddoel van deze opdracht besproken worden, een beschrijving van de tussentijdse demo's staat in Deel III.

Meer specifiek:

- De robots rijden in een op voorhand onbekend doolhof opgebouwd uit panelen.
- De robots verkennen het doolhof en brengen dit in kaart.
- De robots lezen de barcodes die zich in het doolhof bevinden en voeren de taak gelinkt aan deze barcode uit (bv. geluid produceren, 360 graden draaien,...)
- Een virtuele robot (simulator) kan de rol van de echte fysieke robot overnemen.

De opdracht is tot een goed einde gebracht, als het doolhof succesvol in kaart is gebracht (zowel door de virtuele als fysieke robot) en alle taken gekoppeld aan de barcodes uitgevoerd werden.

3 Opbouw van het doolhof

Het doolhof is opgebouwd uit verschillende sectoren, die in verschillende configuraties geplaatst kunnen worden. Elke sector meet 40x40 cm (een paneel van 80x80 cm bestaat dus uit 4 sectoren), de rand van elke sector wordt gevormd door ofwel een rechtopstaande plint, ongeveer 20cm hoog, ofwel door een witte lijn. In dit semester zal geen gebruik gemaakt worden van de hellingen, de wip en de versmalling. Er kan vanuit worden gegaan dat het volledige doolhof zich op hetzelfde niveau bevindt. Een aantal van de panelen is gemarkeerd met een unieke barcode. Deze barcodes maken een exacte plaatsbepaling mogelijk en zullen gelinkt worden aan acties, uit te voeren door de robot. De barcodes liggen in het midden van een sector en worden alleen op 'rechtdoor' sectoren geplaatst.

4 Grafische Interface (Robot)

Elke robot heeft een GUI, hier volgt een beschrijving van de informatie die *tenminste* moet worden weergegeven in de GUI.

- Informatie op basis waarvan de robot aangestuurd wordt.
- Weergave van kennis van het doolhof op basis van de eigen waarnemingen.
- Weergave van eigen positie in het doolhof.
- Weergave van interne toestand en besluitvormingsproces.
- Debug info (tekst log).

Het is duidelijk dat hier redelijk wat vrijheid is om deze interface te maken. Je moet steeds voor ogen houden dat we op één of andere manier willen zien wat de robot op dat moment aan het doen is op software-niveau, zodat dit kan gelinkt worden aan wat de robot feitelijk aan het uitvoeren is op het speelveld.

5 Simulator (virtuele robot)

Elk team dient een simulator te bouwen, dit is bedoeld als hulpmiddel bij de ontwikkeling van de robot. Ook wordt de simulator tijdens de demo gebruikt als demonstratie platform. De simulator moet de volgende eigenschappen hebben:

- De fysieke eigenschappen van de doolhof moeten gesimuleerd worden: muren, lijnen en barcodes.
- De output van de volgende sensoren dient gesimuleerd te worden:
  - afstandsensor,
  - lichtsensor,
  - druksensor.
- De robots hebben een positie op een **continu** vlak (x,y).
- De robots hebben een oriëntatie in graden.
- Simulatie van de afmetingen van de robot zijn optioneel, muurbotsing detectie wordt gesimuleerd.
- De simulator moet een grafische weergave van doolhof en robot geven.

6 Barcodes

Barcodes zijn 16cm breed en bestaan uit 8 stroken van elk 2cm breedte.

Een strook is wit of zwart en duidt respectievelijk een 1 of een 0 aan. De eerste en laatste strook van de barcode zijn altijd zwart. De overige 6 stroken stellen een 6-bit getal voor. Om te vermijden dat barcodes verkeerdelijk geïnterpreteerd worden, worden alle spiegelbeelden van barcodes als ongeldige barcodes beschouwd. Van de 64 mogelijke waarden die de barcode kan aannemen, zijn dus enkel de volgende 28 barcodes geldig: 000001 (1); 000010 (2); 000011 (3); 000100 (4); 000101 (5); 000110 (6); 000111 (7); 001001 (9); 001010 (10); 001011 (11); 001101 (13); 001110 (14); 001111 (15); 010001 (17); 010011 (19); 010101 (21); 010110 (22); 010111 (23); 011001 (25); 011011 (27); 011101 (29); 011111 (31); 100011 (35); 100101 (37); 100111 (39); 101011 (43); 101111 (47); 110111 (55);

Barcodes worden meest-significante-bit eerst gelezen. Dwz dat als je robot 4 zwarte stroken leest, gevolgd door 3 witte stroken en dan terug een zwarte, dat je dan 00001110 uitgelezen hebt, of barcode 000111. Deze barcode stelt het getal 7 voor.

De barcodes zullen enkel gepositioneerd worden op rechte baanstukken en steeds exact in het midden van de tegel. Ook zullen ze de volledige breedte van de sector overbruggen.

Deel III: Fasering, demo's en verslag

1 Fasering gedurende het semester

Om de uitwerking van het project te faseren, wordt er verwacht dat je enkele tussentijdse demonstraties geeft. Deze demonstraties dienen zowel als deadlines om naar toe te werken, maar ook om je vooruitgang te vergelijken met de andere teams. De week vóór elke demonstratie dien je ook een tussentijdsverslag in. Dit verslag informeert het didactisch team over de achtergrond, de verantwoording van de gemaakte keuzes, de wetenschappelijke onderbouw, de details over softwaredelen, enz.

De verschillende tussentijdse demo's vinden plaats op vaste tijdstippen, op maandagen, startend om 15 uur. Het verslag moet de week voordien op woensdag vóór 24 uur ingediend worden, via Toledo. Elk team komt individueel zijn project voorstellen, met een beknopte uitleg over de nakende demonstratie. Na het demonstreren van de gevraagde functionaliteit zullen er een aantal vragen gesteld worden door het didactisch team. Op de tussentijdse demo's en verslagen zal feedback worden gegeven.

Op de woensdag na de laatste demonstratie wordt een eindverslag ingediend. De week nadien geeft elke groep ook een presentatie over het ganse project en het verloop.

2 Overzicht tussentijdse demonstraties

In totaal vinden er drie demonstraties plaats. Hieronder volgen meer details over de verschillende demonstraties, waaronder de voornaamste doelstellingen, data en gerelateerde secties in het eindverslag. Het eindverslag wordt besproken in de volgende sectie.

Het is zeer belangrijk om te realiseren dat het falen op een demonstratie inhoudt dat je niet aan de volgende taak kan werken aangezien die op de voorgaande gebaseerd is! Indien een team faalt op de tussentijdse demonstraties, zal het didactisch team samenzitten en bespreken wat er vervolgens dient te gebeuren.

Anderzijds kan het ook geen kwaad om latere demo's in het achterhoofd te houden en op toekomstig werk te anticiperen. Bvb. voor demo 1 heeft de robot geen sensoren nodig, maar wel voor demo 2. Het kan dus nuttig zijn om de robot al van sensoren te voorzien van in het begin. Let wel op je prioriteiten: zorg dat je niet in de problemen komt met de huidige doelstellingen.

2.1 Demo 1: Communicatie en besturing

Datum 22 oktober **(verslag inleveren woensdag 17 oktober)**

Verloop Deze demo speelt zich af in een lege ruimte zonder doolhof.

Tijdens de demo word je gevraagd om via bluetooth te verbinden met je robot. Daarna laat je zien dat de robot met de pijltjestoetsen bestuurd kan worden. Vervolgens vraagt het didactisch team je om een bepaalde veelhoek te rijden met een bepaalde zijde-lengte. Je geeft deze parameters in in je GUI, waarop de robot de veelhoek afrijdt en nadien uiteraard terug op zijn oorspronkelijke positie terechtkomt.

Vervolgens connecteer je met je simulator en herhaal je de demo daarin.

Je GUI moet in een debugvenster laten zien welke berichten je uitwisselt tussen GUI en robot/simulator, en eventuele debug informatie tonen.

Voorzie dat iemand van je team tijdens de demo kan uitleggen wat er gebeurt, zodat het didactisch team daar niet naar moet vragen. Verwacht je ook aan enkele vragen over de werking van software/robot en de keuzes die jullie gemaakt hebben.

Doelstellingen De voornaamste doelstellingen van deze demo zijn: laten zien dat je via bluetooth met de robot kan communiceren, dat je GUI debug-informatie kan weergeven en dat je de robot nauwkeurig via pijltjestoetsen kan besturen en een veelhoek kan laten rijden met parameters ingesteld vanaf de GUI. Bovendien moet je simulator eveneens in staat zijn om dezelfde demo te doen.

Verslag In je verslag moet je wetenschappelijk onderbouwen hoe je je robot gecalibreerd hebt om nauwkeurig te rijden en het protocol beschrijven dat je gebruikt tussen robot/simulator en GUI.

2.2 Demo 2: Sensoren en de menselijke bestuurder

Datum 12 november **(verslag inleveren woensdag 7 november)**

Verloop Deze demo speelt zich af in een ruimte met een doolhof met barcodes. Al wat jullie robot geacht werd te kunnen in de vorige demo, moet nog steeds mogelijk zijn.

Jullie robot wordt op het doolhof geplaatst en er wordt gevraagd om via de GUI aan de robot de opdracht te geven om autonoom een witte lijn te vinden en zich daarop loodrecht te oriënteren.

Daarna neemt iemand van je team plaats aan de GUI om de robot te besturen, zonder de robot te zien (Bvb. met de rug naar de robot toe). Het didactisch team maakt enkele aanpassingen aan het doolhof. De bestuuder dient nu op basis van enkel de uitgelezen sensorwaarden de robot te besturen en het doolhof te verkennen. Met pen en papier tekent de bestuurder de vorm van het doolhof zoals hij of zij denkt dat het eruit ziet. Uiteraard mag de bestuurder gebruik maken van alle functionaliteit waarover jullie GUI en robot beschikt.

Vervolgens connecteer je met je simulator en herhaal je de demo daarin.

Voorzie dat iemand van je team tijdens de demo kan uitleggen wat er gebeurt, zodat het didactisch team daar niet naar moet vragen. Verwacht je ook aan enkele vragen over de werking van software/robot en de keuzes die jullie gemaakt hebben.

Doelstellingen De voornaamste doelstellingen van deze demo zijn: Je bewust maken over welke informatie de robot en GUI zullen beschikken tijdens de verkenning van het doolhof. Je moet laten zien dat je de sensoren kan uitlezen, weergeven in je GUI en de werking voldoende begrijpt om ze correct te interpreteren en simuleren in je simulator. Bovendien moet je robot in staat zijn om op commando zichzelf recht te zetten op een witte lijn. Uiteraard dient dit alles ook in de simulator te werken.

Verslag In je verslag moet je wetenschappelijk onderbouwen hoe je de sensoren getest en gecalibreerd hebt, en hoe je er mee omgaat in je simulator. Bovendien moet je het algoritme uitleggen waarmee de robot zich orienteert op een witte lijn. Uiteraard moet je ook beschrijven welke aanpassingen je gemaakt hebt aan de simulator, de GUI, het protocol enz.

2.3 Demo 3: de autonome doolhofverkennende robot

Datum 10 december **(verslag inleveren woensdag 5 december)**

Verloop Deze demo speelt zich af in een een ruimte met een doolhof met barcodes. Al wat jullie robot geacht werd te kunnen in de vorige demo, moet nog steeds mogelijk zijn.

Jullie robot wordt voor een barcode gezet en er wordt gevraagd om de barcode uit te lezen.

Daarna legt het didactisch team voor een aantal barcodes vast welke actie ermee verbonden is. Deze acties kunnen zijn:

- draai een rondje naar links
- draai een rondje naar rechts

- speel een muziekje naar keuze
- wacht 5 seconden
- rij vanaf nu aan lage snelheid
- rij vanaf nu aan hoge snelheid
- onthou de positie van deze barcode als “finish”

Jullie robot wordt op het doolhof geplaatst en er wordt gevraagd om het doolhof te verkennen. De GUI tekent het doolhof zoals de robot denkt dat het eruitziet. Wanneer de robot een barcode leest die verbonden is aan een bepaalde actie, voert de robot die actie uit. Nadat het hele doolhof verkend is, bepaalt de robot het kortste pad naar de “finish” en rijdt er aan hoge snelheid naartoe.

Voorzie dat iemand van je team tijdens de demo kan uitleggen wat er gebeurt, zodat het didactisch team daar niet naar moet vragen. Verwacht je ook aan enkele vragen over de werking van software/robot en de keuzes die jullie gemaakt hebben.

Doelstellingen In tegenstelling tot de vorige demo, rijdt de robot in deze finale demo autonoom. Je moet laten zien dat de robot intelligente beslissingen kan maken en een kortste pad kan berekenen. Kortom: je moet de hele opgave geïmplementeerd hebben.

Verslag In je verslag moet verklaard worden hoe de barcodelezer werd ontworpen en hoe je het kortste pad naar de finish bepaalt. Wederom moet je ook beschrijven welke aanpassingen je gemaakt hebt aan de simulator, de GUI, het protocol enz.

3 Verslag

Enkele algemene richtlijnen: probeer het de lezers zou eenvoudig mogelijk te maken: breng een klare boodschap en schrijf bondig. ”Leid” de lezer door je tekst: gebruik inleidende zinnen en verwijs nadien. Definieer begrippen en laat de lezer niet met vragen zitten. Hou figuren en tabellen leesbaar, benoem assen en gebruik eenheden. Voorzie figuren van een verklarend onderschrift en refereer ernaar in je tekst. Vergeet ook de appendices niet van de nodige korte uitleg te voorzien.

Voor elke tussentijdse demonstratie wordt een verslag verwacht. Deze verslagen dienen jullie in te dienen in LaTeX, vertrekkende van een template die op TOLEDO zal worden geplaatst. Voor het verslag van demo 1 mogen jullie **maximaal 10 bladzijden** indienen, **voor demos 2 en 3 wordt dit respectievelijk maximaal 15 en 20 bladzijden**. Op de verslagen van demo 1 en 2 krijgen jullie feedback vanwege het didactisch team.

Op het einde van het semester wordt een eindverslag verwacht. Dit verslag omvat grofweg twee grote delen. Een eerste wetenschappelijk luik dat op een *wetenschappelijk, onderzoeksgebaseerde en duidelijk gefundeerde wijze* de gemaakte keuzes in het project toelicht. Bronvermelding is hierin ook essentieel.

> *Wanneer tekst en/of code gebruikt wordt die overgenomen is van, of sterk geïnspireerd op, ander werk, moet dit duidelijk aangegeven worden in het rapport. Het nalaten van dergelijke vermelding kan als plagiaat beschouwd worden, en tot ernstige sancties leiden, overeenkomstig het examenreglement.*

Het tweede luik bevat een meer ervaringsgerichte bespreking van het verloop van het project, dit deel maakt op zich geen deel uit van de wetenschappelijke bespreking van het project. Dit tweede deel bevat een beschrijving van het proces, de werkverdeling, een kritische analyse, enz.

De **deadline** voor het eindverslag is de woensdag na de finale demonstratie, **woensdag 12 december**. Let op, het eindverslag bevat in zijn totaliteit, alles inclusief, **maximum 30 bladzijden** en hiervoor vertrek je wederom van de gegeven LaTeX-template.

Het is niet de bedoeling dat dit verslag pas op het einde gemaakt wordt, maar wel dat je het eindverslag opbouwt aan de hand van de tussentijdse verslagen. Het tussentijdsverslag dat je voor elke demo indient is niets anders dan het eindverslag waarin bepaalde secties zijn geschreven, maar het is een verslag dat op zichzelf een mooi afgerond geheel vormt. Deze verslagen (buiten het eerste) bevatten ook een appendix waarin de resultaten van de vorige demonstraties, conclusies en bijhorende aanpassingen in dit verslag t.o.v. het vorige kort aangeduid worden.

De structuur van het verslag is terug te vinden in de LaTeX-template.

4 Finale presentatie

Op **17 december, ganse dag**[4] geeft elke groep een presentatie van het project: een voordracht van ongeveer 20 minuten, gevolgd door 20 minuten vragen en discussie. Het ganse team moet op deze presentatie aanwezig zijn.

De presentatie bevat minstens volgende onderdelen:

- Beschrijving van de robot;
- Gebruikte algoritmes en oplossingen;
- Kort overzicht van behaalde resultaten;
- Werken in groep: ervaringen, moeilijkheden;
- Eigen conclusies: sterktes en zwaktes van verwezenlijkte project.

De informatie op de slides mag beknopt zijn, voor details kan je verwijzen naar het verslag. De bedoeling is om zoveel mogelijk dubbel werk tussen presentatie en verslag te vermijden.

5 Peer-assessment

Op het einde van het semester zal ook via peer-assessment gevraagd worden naar ieders evaluatie m.b.t. de andere groepsleden. Deze peer-assessment maakt geen deel uit van het eindverslag, maar zal afzonderlijk via TOLEDO opgevraagd worden.

Daarenboven zal er na demo 1 ook een preliminaire peer-assessment plaatsvinden.

[4] Indien er problemen zijn om voormiddag de presentatie te geven, verwittig dan onmiddellijk de coördinator van het vak.