WKaibo

Alexandra Arkut 10583149
Jan Geestman 10375406
Victoria v.d. Mark 10549544
Feli Nicolaes 10542442

27 juni 2014

Abstract

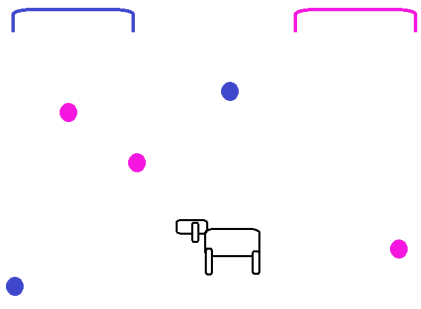
Dit onderzoek was bedoeld om de functionaliteit van de SONY Aibo te testen door met de Urbi programmeertaal een sorteer-algoritme te schrijven. Het doel van het programma was de Aibo balletjes te laten sorteren door gekleurde balletjes te schieten in een doeltje van dezelfde kleur. Na het afronden van het onderzoek is gebleken dat het, onder andere wegens tijdgebrek, niet mogelijk is geweest het programma aan de eisen van het vooraf gestelde doel te laten voldoen. Wel tonen de resultaten aan dat de Aibo in staat is een bal te zoeken, hier naartoe te lopen en deze weg te koppen. Indien meer tijd aanwezig is om dit onderzoek uit te breiden zou het hoogstwaarschijnlijk mogelijk zijn om een simpel werkend sorterings-algoritme te schrijven voor de SONY Aibo.

I. Introductie

De Aibo is een robothond, geproduceerd door SONY. Deze hond is bedoeld als 'gezelschapsrobot' en kan in zijn gebruik veel bewegingen van een echte hond nabootsen. De Aibo kan echter ook gebruikt worden om taken te verrichten. Hierbij kan gebruik gemaakt worden van ingebouwde functionaliteiten. Voorbeelden hiervan zijn het herkennen van een bal en het maken van simpele bewegingen.

De vraag die hieruit volgt, is in hoeverre een Aibo gebruikt kan worden om verschillende ballen op kleur te sorteren; de WKaibo. In deze situatie zijn er balletjes in verschillende kleuren en een aantal doeltjes. Het doel zou zijn deze balletjes te schieten in de doeltjes van dezelfde kleur. De Aibo wordt geprogrammeerd in de programmeetaal Urbi, waarbij de bewegingen van de WKaibo gaandeweg zullen worden uitgebreid.

De verwachtingen zijn dat de WKaibo in ieder geval twee duidelijk verschillend gekleurde ballen van elkaar kan onderscheiden en deze in het juiste doel kan schoppen. De WKaibo zal er waarschijnlijk te lang over doen in een veld met meer dan twee ballen.



Figuur 1: *Een tekening van de opzet*

II. Materiaal en methode

Opzet

In het RoboLab zijn verschillende Aibo's beschikbaar. Voor dit experiment zijn vooral de Aibo's *Moos* en *Lewy* gebruikt, omdat deze Aibo's de minste mankementen vertoonden. Sommige andere Aibo's, zoals *Mina*, hadden hardwareproblemen waardoor ze niet goed functioneerden. Ook weigerden sommige Aibo's zich uit te schakelen. Alle gebruikte Aibo's zijn van het type ERS-7. Naast Aibo's waren er ook verschillende SONY geheugenkaarten beschikbaar, allemaal van 16MB. Ook tussen deze geheugenkaarten bleken verschillen te zijn. Één van de geheugenkaarten voerde telkens hetzelfde programma uit, ook als de inhoud van het kaartje veranderde. Om programma's op het kaartje te zetten, werd er een SONY *MagicGate Pro* kaartlezer gebruikt. Zie voor een compleet overzicht van het gebruikte materiaal sectie V.

Het werken met de Aibo's bleek lastiger dan verwacht. Het was moeilijk om goedkloppende documentatie te vinden over Urbi, de programmeertaal waarin gewerkt werd. De Aibo's zijn waarschijnlijk geprogrammeerd met Urbi versie 1.5, hoewel ook dit niet helemaal zeker is gebleken [1]. Voor de waarden van verschillende sensoren is de Urbi documentatie voor onder andere de Aibo ERS7 geraadpleegd [4]. De Aibo's zijn relatief oud en er is veel informatie over te vinden, maar de informatie is vaak niet juist of onvolledig. Hierdoor moesten er veel verschillende predicaten geprobeerd worden voor één actie, omdat er telkens meerdere opties stonden en niet duidelijk was welke optie de juiste was.

Het experiment is opgedeeld in verschillende delen, zodat er verschillende tussenstappen zijn. Hierdoor is het makkelijk te zien wat er gedaan moest worden en hoe een probleem moest worden aangepakt.

De verschillende subdoelen zijn:

Een programma op de Aibo laden
: Een bestaand programma op de Aibo laden om uit de vinden hoe de software van de Aibo werkt

Lopen
: De Aibo een paar seconden laten lopen

Een bal herkennen
: Om dit te testen wordt na het zien van de bal een actie uitgevoerd (zoals gaan staan als de bal in het zicht is en zitten als de bal uit het zicht is)

Bal zoeken
: Rondkijken tot hij een bal kan vinden

De bal tracken
: Als de bal in zicht is en bewogen wordt, moet hij zijn hoofd meebewegen zodat hij naar de bal blijft kijken

Naar de bal lopen
: Als de bal in zicht is, moet hij richting de bal lopen

Bal schoppen
: Als hij de bal direct voor zich heeft, moet hij de bal vooruit schoppen

Doel herkennen
: Herken het doel als een object

Bal in een richting schoppen
: De bal kunnen mikken en schoppen in de kijkrichting van de Aibo

Doel zoeken
: Rondkijken tot hij een doel kan vinden

Bal in doel kunnen schoppen

De bal kunnen schoppen in een doel als er een doel in zicht is

Kleur van bal herkennen
Kleur van de bal herkennen als er een bal in zicht is

Kleur van doel herkennen
Kleur van het doel herkennen als er een doel in zicht is

Bal van doel onderscheiden
Herken dat een bal geen doel is en andersom

Herken twee dezelfde kleuren
Herken of een bal en een doel dezelfde kleur zijn

Juiste bal in juiste doel schoppen
Schop een bal in een doel dat dezelfde kleur heeft

Extra's
Doe bijvoorbeeld een dansje als de bal in het juiste doel is geschopt

Proces

De eerste twee dagen is, naast het zoeken naar documentatie over Urbi en de Aibo, vooral aandacht besteed aan het laten lopen van de Aibo naar een bal. Dit betrof de eerste zes subdoelen beschreven in Materiaal en Methode. Om erachter te komen wat de syntax-regels van Urbi zijn, is onder andere gekeken naar code uit papers gevonden op het internet [3]. De aanpak bij het implementeren was het opdelen van het probleem in een aantal delen; allereerst moest de Aibo de bal in zijn zicht krijgen. Hierna moest de Aibo met zijn lichaam in de richting van de bal draaien. Als laatste moest hij richting de bal lopen. Om te zorgen dat de Aibo de bal in zijn zicht kreeg, moest de Aibo in staat zijn om een groot deel van de ruimte te kunnen zien. Dit is getracht te bereiken door de Aibo, wanneer de bal niet in zicht was, zijn hoofd heen en weer te laten bewegen, tot hij een bal had gevonden. Dit zoekproces is weergegeven in figuur 2

Figuur 2: *WKaibo zoekt naar een bal*

Bij het laten draaien van de Aibo naar de bal moest rekening gehouden worden met het feit dat de standaard turn()-functie als parameter het aantal seconden neemt en niet het aantal graden. Hierdoor moest eerst bepaald worden hoeveel graden per seconde gedraaid werd. Met dit gegeven kon aan de hand van de nekpositie berekend worden hoe de Aibo in de richting van de bal kon draaien. Wanneer de Aibo naar de bal gericht stond, kon hij richting de bal lopen. Zie figuur 3 om te zien hoe hij op een bal reageert. Vanwege het feit dat het draaien van de Aibo niet heel secuur was, is ervoor gekozen om hem twee seconden te laten lopen en vervolgens het hele proces hierboven beschreven te herhalen. Eerst was overwogen de Aibo twee stappen te laten lopen in plaats van twee seconden. Een nadeel hiervan was echter de snelheid waarmee de Aibo de bal bereikte. Zie deze link een demonstratie: `http://tinyurl.com/khg9tg7`.

Figuur 3: *WKaibo heeft een bal gevonden en draait er naartoe*

Figuur 4: *WKaibo met plastic lepel*

Op de derde dag stonden subdoelen *bal schoppen* tot en met *doel zoeken* op het programma. Voor het vooruit krijgen van de bal zijn verschillende opties overwogen. Het meest voor de hand liggende leek om de Aibo de bal met zijn voorpoten te laten schoppen en zelfs de achterpoten werden overwogen.

De problemen die echter op zouden kunnen treden waren dat de Aibo niet zou weten met welke poot hij de bal moest schoppen en dat hij op de bal zou kunnen staan waardoor deze onvindbaar zou worden. Met de korte tijd die aan deze opdracht besteed kon worden waren dit te grote problemen om aan te pakken.

Hierdoor is besloten om WKaibo de bal te laten koppen. Om de kracht van de kop te vergroten is er voor gekozen om een lepel aan de kop van de Aibo te bevestigen (zie figuur 4 en 5). Door dit lichtgewicht instrument te bevestigen, werd de bal verder weggeslagen. Hierdoor werden de kopballen effectiever en deed de Aibo er minder lang over om het uiteindelijke doel te bereiken.

In eerste instantie was de kopbal op dergelijke wijze geïmplementeerd dat de Aibo vanuit een staande positie meteen overging in een liggende positie. Uit een aantal tests bleek echter dat de positie van de Aibo ten opzichte van de bal niet consistent was, aangezien de gekozen beweging van staan naar liggen niet vloeiend genoeg verliep. Vanwege deze kwestie is ervoor gekozen de Aibo eerst van een staande positie naar een zittende positie te laten overgaan en vervolgens kon op een geleidelijke manier de liggende positie bereikt worden.

Een alternatief probleem was dat de Aibo soms begon te koppen terwijl hij nog aan het koppen was. De reden hiervoor was dat de bal tijdelijk uit het zicht van de Aibo verdween en daarna weer terug kwam. Op dat moment initialiseerde de kopbalfunctie zich opnieuw, terwijl de oude functie nog niet getermineerd was. Om dit te verhelpen, is overwogen om, geïnspireerd door een verslag van een ander jaar, een boolean te gebruiken, die pas van waarde zou veranderen wanneer de héle kopbal voltooid was [2]. Hoewel dit functioneerde naar behoren, is dit niet in het uiteindelijke programma gebruikt, aangezien ervoor is gekozen dit probleem op te lossen door gebruik te maken van de *touch-*

sensoren die aanwezig zijn op de Aibo. Zie voor demonstraties van beide kopballen `http://tinyurl.com/khg9tg7` en `http://tinyurl.com/kw4zxkk`.

Figuur 5: *WKaibo kopt een bal weg*

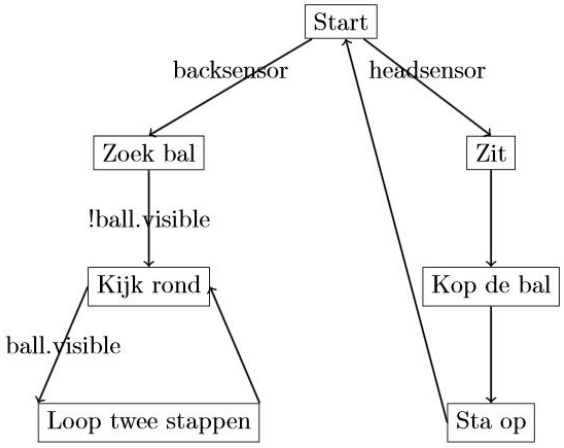
De laatste dag is besteed aan het *finetunen* van het programma. Het implementeren van kleurherkenning bleek ingewikkelder dan gedacht en vooral teveel werk in de tijd die voor dit project gegeven was. Daarnaast bleek het door de vele hardwareproblemen van de Aibo en zijn accu's onmogelijk om het hele programma in één keer uit te voeren, zelfs voor het testen. Hierdoor is besloten om het programma in losse delen te implementeren, om op die manier alles te kunnen laten zien waar de Aibo toe in staat is.

Het *finetunen* van deze laatste dag bestond uit het uitbreiden van de kopbal. Nadat de Aibo naar zijn bal is gelopen rest hem namelijk nog de precieze plek van de bal te vinden om gericht te kunnen koppen. Ervan uitgaande dat de hond dus eerst naar de bal is toegelopen hoeft hij alleen maar om zijn eigen as te draaien, met zijn hoofd naar beneden gericht, om te kunnen beslissen op welke plek hij moet bukken. Na uitvoerig testen werd echter duidelijk dat de turn()-functie van de Aibo onder geen omstandigheden precies genoeg was om dit probleem om te lossen.

Hierna is besloten om alle programma's op een bepaalde manier aan elkaar te verbinden om zo de situatie met goedwerkende robots te simuleren. Dit is getracht door alle losse functies in één programma te zetten en deze aan te laten roepen door de verschillende sensoren op de rug van de hond. Een mogelijk probleem dat op kon treden was dat *Moos* hier wellicht niet bruikbaar voor was; deze had namelijk last van de rechter achterpoot, waardoor het eerste programma niet soepel door deze Aibo uitgevoerd zou kunnen worden. Dit bleek in de praktijk echter amper problemen te veroorzaken, waardoor er toch voor gekozen is *Moos* als demonstratierobot te gebruiken.

III. Resultaten

De resultaten van dit onderzoek tonen aan dat de Aibo in staat is om een bal te herkennen, hiernaartoe te lopen en te koppen. De hardware, maar vooral de accu van de Aibo's beschikbaar in het RoboLab, laten echter niet meer toe dit programma in één keer te laten draaien. Hierdoor is besloten het programma in stukken op te delen en daarna gebruik te maken van de *touch-sensoren* van de Aibo om de verschillende functies aan te roepen. In onderstaand figuur is de werking van het programma weergegeven 6.



Figuur 6: *Een diagram dat de programmastructuur simpel uitlegt*

Van de subdoelen genoemd in sectie II zijn er zeven bereikt. Uiteindelijk bleek het subdoel "De bal tracken"niet noodzakelijk te zijn voor het uitvoeren van het aan het begin van het onderzoek gestelde doel. De subdoelen die betrekking hadden op het herkennen van kleuren zijn niet voltooid. Hoewel het wel degelijk mogelijk is om de Aibo kleuren van elkaar te laten onderscheiden [2], was er wegens de grote hoeveelheid subdoelen niet genoeg tijd om het programma aan alle subdoelen te laten voldoen.

IV. Discussie

Door problemen met de hardware van de Aibo's was het in sommige gevallen lastig te bepalen of delen van het geschreven programma niet naar behoren functioneerden door een fout in de code of door een klein defect in de Aibo, batterij of geheugenkaart.

Ook leek er een probleem te zijn met de manier waarop de Aibo de programma's van de geheugenkaart leest. Goed functionerende bestanden stopten met werken als de bestandsnaam werd aangepast. Het kan zijn dat deze fout komt door gecorrumpeerde *ini* files op de Aibo of geheugenkaart.

Door toedoen van de technische mankementen en het tijdstekort is het niet mogelijk geweest om de WKaibo daadwerkelijk ballen van elkaar te laten onderscheiden. In theorie zou het mogelijk moeten zijn om het doel van het project te laten slagen, maar dit is niet verifieerbaar gebleken.

Daarnaast zou het project - ook met werkende hardware - wellicht te ingewikkeld zijn geweest voor de gegeven tijd. Ook zou het überhaupt lastig zijn geweest om de Aibo ballen van elkaar te laten onderscheiden door hun kleur. De camera zou wellicht niet goed genoeg zijn gebleken en het registreren van kleuren zou problemen opgeleverd kunnen hebben, omdat kleuren onderhevig zijn aan externe factoren.

De herziene hypothese is dat de software van de Aibo's hen in theorie in staat zou stellen om te kunnen voetballen. Wellicht is ook een simpel sorterings-algoritme mogelijk. In de pratkijk is echter gebleken dat de gebruikte hardware dit amper toestaat. Toekomstige studenten zouden daarom gewaarschuwd moeten worden voor deze mankementen.

Referenties

[1] Jean-Christophe Baillie, Mathieu Nottale, Benoit Pothier, Nicolas Despres (2007), "Urbi tutorial for Urbi 1.5". `http://www.gostai.com/doc/en/urbi-tutorial-1.5/index.html`

[2] M. Offerhaus, B. Postma, T. A. Unger, "Action Through Color Recognition for Sony's AIBO", 2013 `http://staff.science.uva.nl/`

~toto/zsb/old_papers/papers_2012-2013/Offerhaus.M.,Postma.B.,Unger.T.A.-Action.Through.Color.Recognition.for.Sony%e2%80%99s.AIBO.pdf

[3] F. Papadopoulos, "Socially Interactive Robots as Mediators in Human-Human Remote Communication", 2012

[4] Author Unknown (2006-2007), "URBI Doc for Aibo ERS2xx ERS7 and URBI 1.0, Devices documentation". http://www.gostai.com/doc/en/aibo.pdf

V. Materiaal

Bij dit onderzoek is gebruik gemaakt van de volgende materialen:

- SONY Aibo ERS-7 (Moos, Mina, Lewy, Carlos)
- SONY MagicGate kaartlezer
- Urbi programmeertaal (waarschijnlijk 1.5)
- Geheugenkaarten 16 MB
- Plastic lepel
- Gaas-tape
- Telefoon met video camera
- Laptop (Windows & Mac-OS)

VI. Broncode

```
##############################################################################
# File: headKop.u   #
# Description: This file contains code that enables the    #
# Aibo to look for a ball and make headers. This file is loaded in the    #
# urbi.ini file.   #
# Names: Feli Nicolaes, Jan Geestman, Victoria van der Mark, Alexandra Arkut  #
# Student Numbers: 10542442, 10375406, 10549544, 10583149   #
##############################################################################


robot.StandUp();
load("swalk.u");

# Aibo makes a header
at(headSensor.val > 0)
```

```
{
robot.sit();
robot.initial();
headPan.val = 0 time:0.5s &
headTilt.val = -16 time:0.5s ;
neck.val = -79 time:0.5s ;
neck.val = -20 time:0.05s &
headTilt.val = 50 time:0.025s ;
robot.StandUp();
headPan.val = 0 time:0.5s &
headPitch.val = -16 time:0.5s &
neck.val = -70 time:0.5s ;
};


# Aibo searches for a ball
at(backSensorR.val > 0)
{
looking: robot.looking() ;
};


function robot.looking()
{
# Turn head from left to right
balltracking: whenever (!ball.visible)
{
headPan.val = headPan.val sin:6s ampli:50 &
neck.val = -20 time:0.25s & headTilt.val = -10 smooth:0.25s ;
},

# Stop looking for ball; turn/walk towards ball
ballfound: at (ball.visible)
{
freeze balltracking;
robot.turnTo();
headPan.val = 0 time:0.25s;
robot.walkTo();
unfreeze balltracking;
};

# When distance from head IR sensor to ball is smaller than 10
```

```
at (distanceNear.val < 10)
{
stop looking;
};

};

# Calculates how long Aibo has to turn using value of headPan
# 1 second equals 36 degrees of turning
function robot.turnTo()
{
var a;
a = headPan.val / 36;
a = -a;
robot.sturn(a);
};


# Walk for two seconds
function robot.walkTo()
{
robot.walk(2s);
};
```

VII. Latex Template

Voor de lay-out van het verslag is gebruik gemaakt van het volgende template:

Journal Article
LaTeX Template
Version 1.3 (9/9/13)
This template has been downloaded from: http://www.LaTeXTemplates.com
Original author: Frits Wenneker (http://www.howtotex.com)
License: CC BY-NC-SA 3.0 (http://creativecommons.org/licenses/by-nc-sa/3.0/)

Verder is dit template aangepast en zijn hierin een aantal packages toegevoegd.