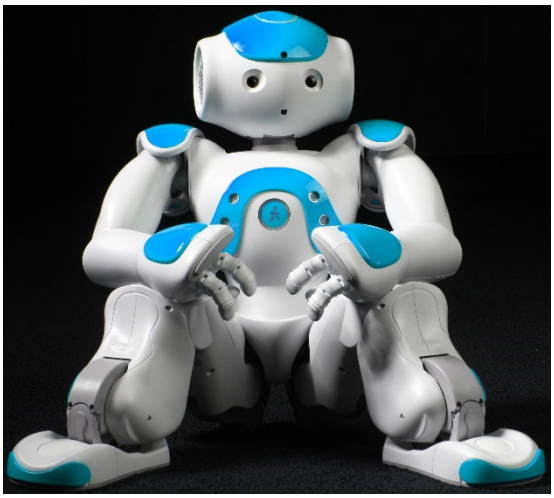
Bachelorproef



howest
De Hogeschool West-Vlaanderen
Lid van de Associatie Universiteit Gent

Studiegebied	**Handelswetenschappen en bedrijfskunde**
Bachelor	**Toegepaste Informatica**
Afstudeerrichting	**-**
Academiejaar	**2014-2015**
Student	**Maxim Vanhockerhout**

Thema

Een vergelijkende studie tussen menselijke bewegingen en robotische bewegingen

Zijn de bewegingen van een robot accuraat ten opzichte van de bewegingen van een mens?

Stageplaats

QBMT/ ZORA robotics
Oostende

Maxim Vanhockerhout
QBMT te Oostende academiejaar 2014-2015

Een vergelijkende studie tussen menselijke bewegingen en robotische bewegingen

Woord vooraf

Ik wil graag de mensen van QBMT bedanken voor de leuke en interessante stage. Ik heb veel bijgeleerd over de werkingen binnenin een softwarebedrijf.

Christophe en Arne wil ik extra bedanken. Christophe omdat hij me veel oefeningen liet maken voor ZORA zodat ik toch voldoende kennis had om dit document op te stellen. Hij kon mij heel gemakkelijk 1 maand laten werken aan de bewegingen en me dan een volledig andere job gegeven hebben, wat hij niet deed. Voor Arne omdat hij altijd klaar stond om me te helpen als ik ergens vast zat.

Samenvatting

Dat robotten een deel gaan uitmaken van de toekomst is niet meer weg te denken, maar wat heeft een robot vandaag de dag als nuttigheid voor de mensheid? Deze vraag en andere vragen worden in dit document uitgelegd. We gaan kijken welke applicaties een robot heeft, meer bepaald in de zorgsector. De zorgsector is groot, dus ook hier gaan we ons focussen op 1 bepaald aspect namelijk de revalidatie van ouderen en kinderen. In de revalidatie zijn bewegingen essentieel, dus gaat dat dan ook het meest aan bod komen.

We gaan kijken welke bewegingen mogelijk zijn en welke niet en daarvoor gebruiken we de bekendste robot in de zorgsector genaamd ZORA, de zorgrobot. Samen met ZORA gaan we kijken hoe bewegingen worden gemaakt en hoe deze dan uitgevoerd zullen worden.

ZORA is een vrouwelijke robot. In dit document verwijzen we daarom naar ZORA met 'zij' en 'haar'.

Robotica, NAO, ZORA, humanoïde, robot, zorgsector, revalidatie, bewegingen, mogelijk, onmogelijk, werkwijze, Aldebaran, QBMT, python, AngularJS

Abstract

Robots will be a part of the future, nobody can deny this, but what are the applications of a robot now? This question and others are answered in this paper. We'll look at the applications that a robot has today, and more specific applications in the health care sector. The health care sector is big, so here we'll zoom in on the rehabilitation of people.

When we talk about rehabilitation then we think of movements and this will be the focus of this paper.

We are going to look which movements are possible and which are not and for that we are using the most famous robot in the health care sector named ZORA, the humanoid robot. Along with ZORA we'll see how movements are made and how it will be implemented.

Robotics, NAO, ZORA, humanoid, robot, health care, rehabilitation, movements, possible, impossible, workflow, Aldebaran, QBMT, Python, AngularJS

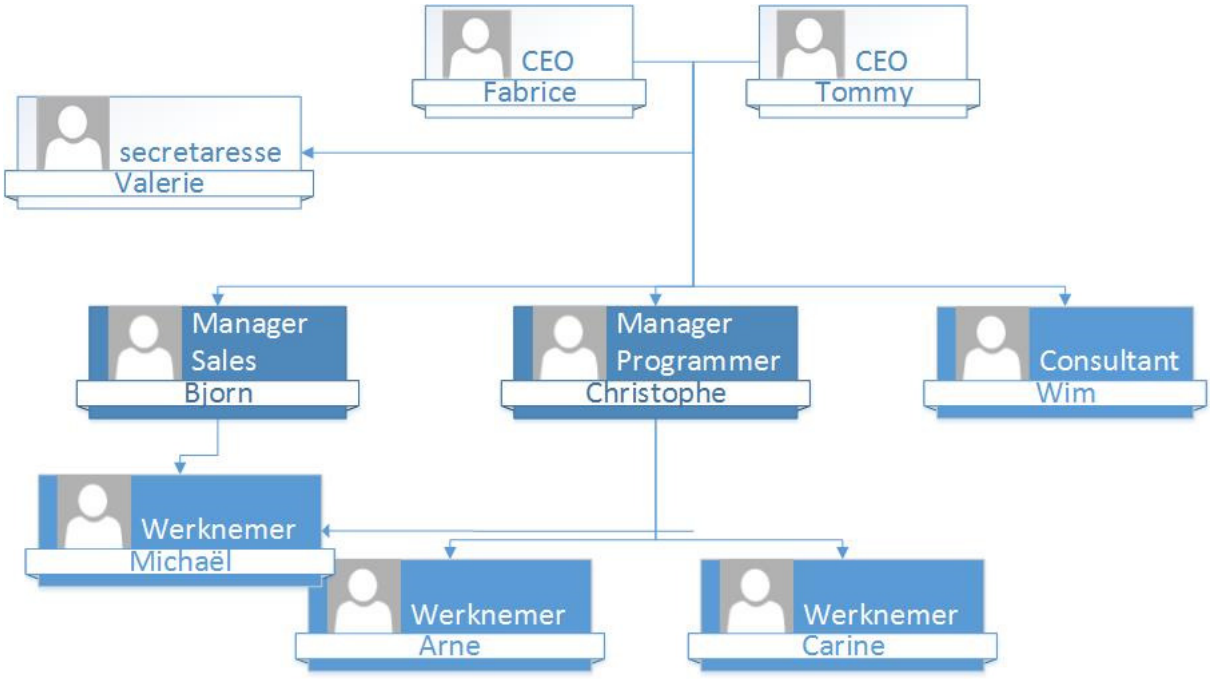
Verklarende woordenlijst

- **Bézierkromme/Béziercurve**: is in de wiskunde een type parametrische kromme. Het wordt bepaald door 2 of meer punten in 1 vlak of ruimte. Het 1ste punt verbindt het laatste punt door te vertrekken in de richting van het 2de punt, de richting wordt steeds aangepast naar een volgende punt en aankomend bij de laatste vanuit de richting van het voorafgaande punt.
- **AngularJS**: een Javascript framework. Kan toegevoegd worden aan een HTML pagina met de <script> tag. Het breidt HTML attributen uit met Directives en bindt data aan HTML met Expressions.
- **Python**: is een hogere programmeertaal ontworpen in het begin van de jaren '90. Het is ook een objectgeoriënteerde programmeertaal.
- **OS**: operating system of besturingssysteem.
- **ZORA-CONTROL**: de web-interface waar je ZORA kan besturen, waar je applicaties kan starten, waar je hem kan laten praten en waar je bewegingen kan uitvoeren.

Inhoudsopgave

Voorstelling van het bedrijf

QBMT (Qatar-Belgium Modern Technology) is een softwarebedrijf dat ZORA de zorgrobot programmeert. Het is opgericht door Tommy Deblieck en Fabrice Goffin en ze hebben 2 vestigingen, 1 in Diepenbeek en 1 in Oostende.



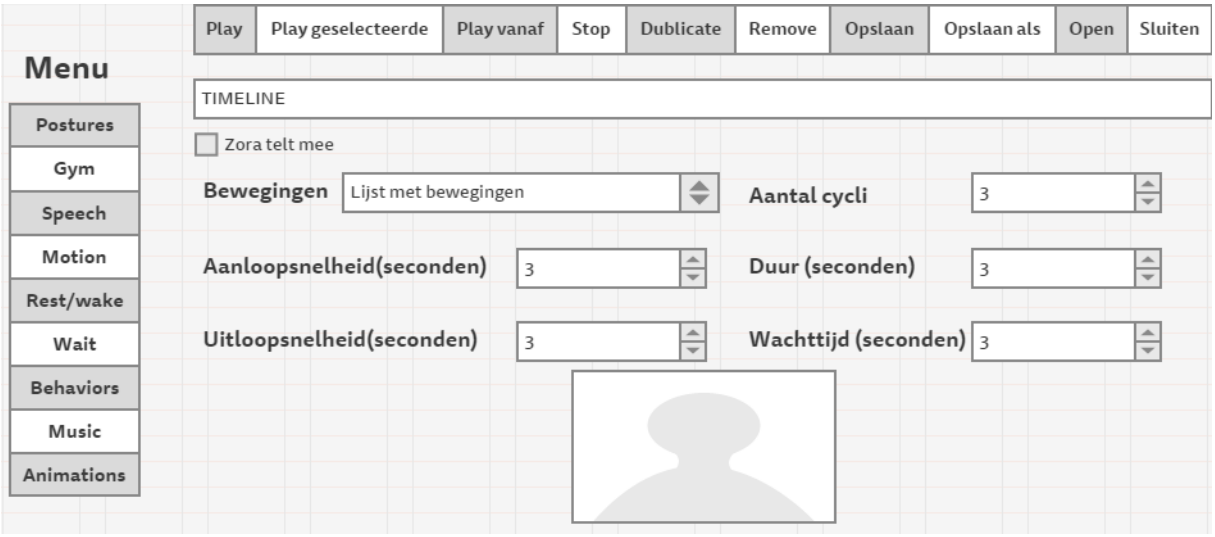
Onderzoeksvraag

Zijn de bewegingen van een robot accuraat ten opzicht van de bewegingen van een mens?

Er zijn verschillende soorten robotten, maar er is maar 1 soort robot die het dichtst bij de mens staat en dat is een humanoïde robot. Voor dit onderzoek gebruiken we een humanoïde robot, die leveren ook de beste resultaten.

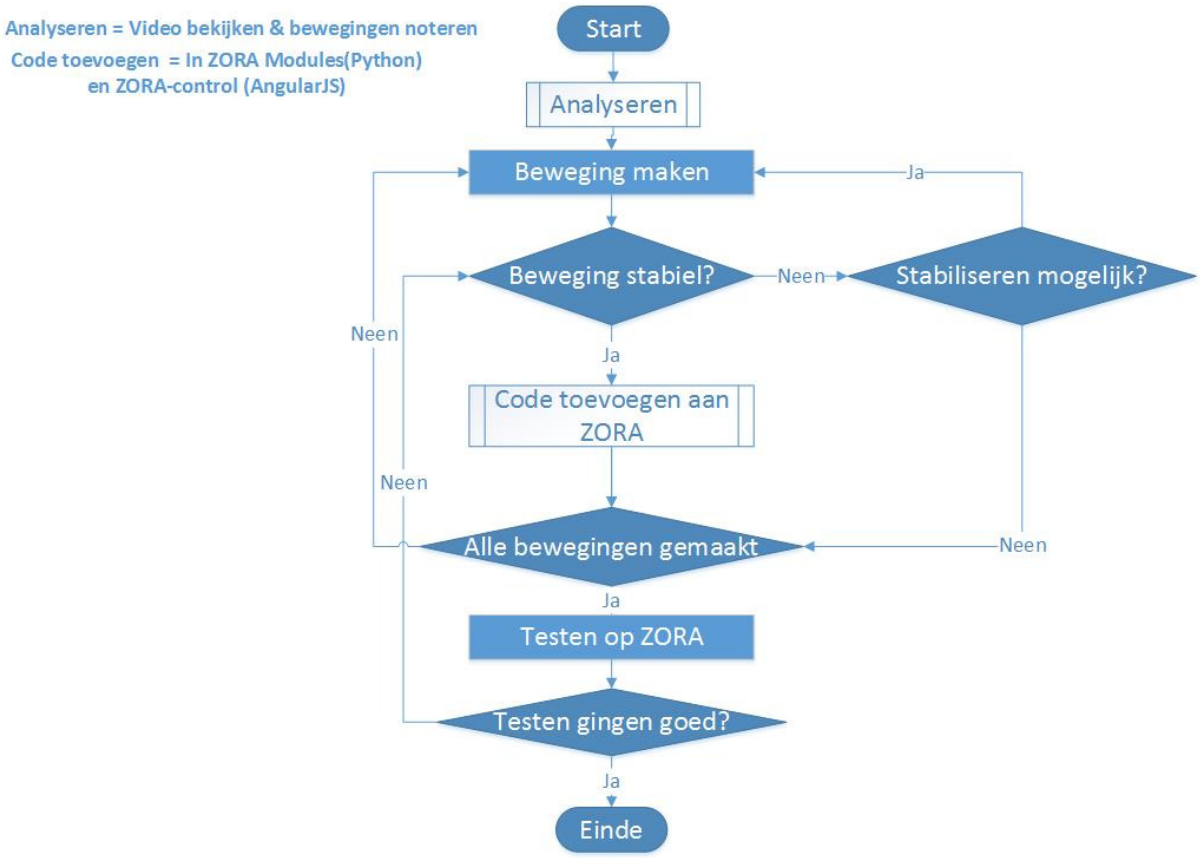
Analyse

Een mogelijk nieuw design voor de composer:



Wegens te weinig tijd jammer genoeg niet geïmplementeerd

Hoe je te werk gaat met bewegingen:



Oplossing

In mijn stage ben ik hoofdzakelijk bezig geweest met het creëren van revalidatie oefeningen/bewegingen. Ik moest verschillende bewegingen analyseren en namaken voor ZORA. Hier lag de moeilijkheid voornamelijk bij ZORA. Ik moest zorgen dat ZORA in evenwicht bleef en toch nog een goede oefening/beweging kon maken. Waar er ook moest opgelet worden, is dat ZORA niet ging haperen tijdens de beweging. De vingers bijvoorbeeld zijn heel kwetsbaar en kunnen ook zeer gemakkelijk ergens vast zitten.

Na lang zoeken en veel proberen heb ik nog een redelijk aantal bewegingen kunnen maken. In totaal zijn er nu 87 bewegingen en al deze bewegingen worden zeer veel gebruikt door de kinesisten in de bejaardentehuizen en ziekenhuizen. Er zijn ook bewegingen die speciaal gemaakt zijn voor een instelling, zoals de rolstoelbeweging. Hier gaat ZORA voortonen aan de mensen hoe je in een rolstoel moet gaan zitten en er terug uit moet gaan. Dit allemaal op een heel realistisch manier.

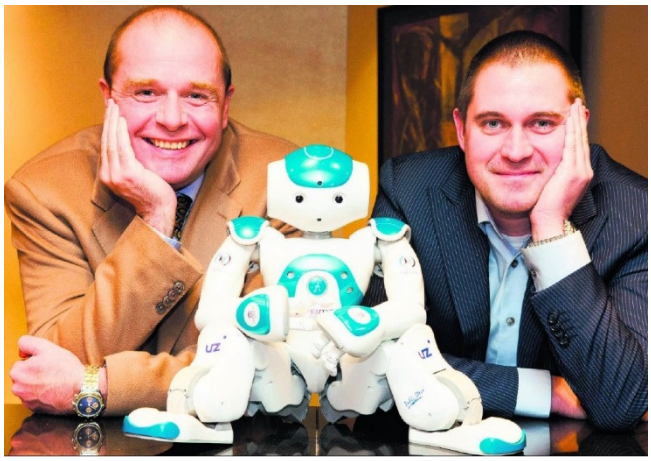
Een revalidatiedansje maken, was een doelstellingen en dat is ook gebeurd. Het dansje Macarena is een goed voorbeeld om op een leuke en speelse manier te leren dansen, maar ook om te revalideren.

De mensen zijn blij dat er zoveel nieuwe bewegingen zijn, maar waar ze heel blij mee zijn is dat de nieuwe bewegingen goed en zeer realistisch zijn.

Bespreking van het project

1. De geboorte van ZORA

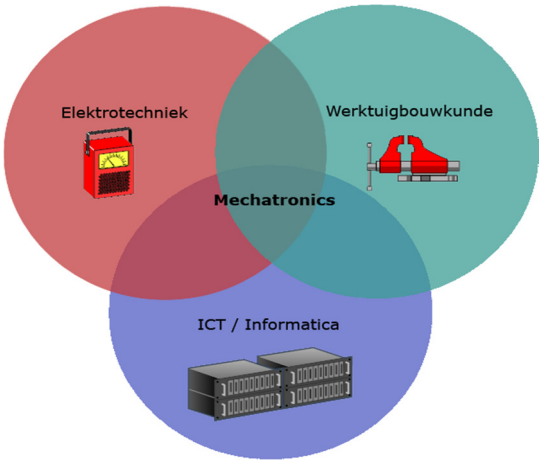
In a galaxy far far away, zijn er 2 vrienden die grote fans zijn van Star Wars. Hun namen zijn Tommy Deblieck en Fabrice Goffin. Hun favoriete personage, hoe kon je het raden, C3PO. Zij zagen een toekomst in robotica en dus gingen ze opzoek naar hun eigen C3PO en die vonden ze. In 2012 kwamen ze in contact met een Frans bedrijf dat robots produceert, dat bedrijf heet Aldebaran. Aldebaran produceert 3 verschillende robots, 1 daarvan is NAO. Het duurde niet lang of ze hadden al een werkend prototype waarmee ze naar hun gekende partner, de Christelijke mutualiteit, trokken. Daar hadden ze direct door dat dit een geweldig idee was voor de zorgsector. Dus gaven de 2 Star Wars fans een passende naam aan hun eigen C3PO en werden ze de papa's van de inmiddels bekendste robot van Vlaanderen, ZORA. ZORA is al inmiddels 63 keren verkocht in een tijdsperiode van 3 jaar. Net zoals bij mensen stopt ook ZORA niet met evolueren, maar in dit geval zijn het ideeën die ZORA laat groeien. Wie weet gaat ZORA binnen een paar jaar er volledig anders uitzien.



2. Robotica

2.1. De Basis van Robotica is Mechatronica

Elektrotechniek, werktuigbouwkunde, meet-,besturings- en regeltechnieken zijn verschillende ingenieursdisciplines die samen de mechatronica vormen.



Mechatronica is een aanpak bij het integraal en optimaal ontwerpen van een mechanisch systeem en het bijbehorende regelsysteem. Dit regelsysteem kan elektronisch zijn, maar vaak bestaat het uit een embedded computer.

Deze ingenieursdiscipline heeft als grote voordeel dat de systemen die eruit komen superieure eigenschappen hebben en ook goedkoper en meer flexibel zijn. Dit doen ze door tegelijkertijd de mechanische constructie samen met het regelsysteem te ontwerpen.

Er zijn veel producten die een mechatronisch systeem zijn zonder dat mensen dit weten, zoals een Cd-speler, Dvd-speler, het ABS-systeem in de moderne auto's en klimaatregelingen. Ook robotten zijn mechatronische systemen. Hier wordt er dan hoofdzakelijk gewerkt met de theoretische implicaties en de praktische toepassingen van robotten.

2.2. Robotica als wetenschap

Robotica is een wetenschap die zich voornamelijk richt op systeemintegratie. Een goed ontworpen robot zal verschillende waarnemingen, met behulp van sensoren verzamelen, analyseren en er gepast op reageren. Deze reacties zijn natuurlijk voorgeprogrammeerd in de robot en alles daarbuiten negeert de robot gewoon. We spreken hier nog niet van kunstmatige intelligentie, dat is nog een stapje verder en staat ook volledig los van de robotica.

2.2.1. Onderzoeksdomeinen

Tijdens het ontwerpen van robotten zijn er verschillende onderzoeksdomeinen ontstaan, elk met hun eigen problemen. Al deze problemen moeten eerst overwonnen worden bij de constructie van een robot.

De verschillende domeinen zijn:

- Sensing → Zoals de term al zegt, gaat het hier hoofdzakelijk over het ontwerpen en de werking van de sensoren.
- Robotsturing → Hier wordt er onderzoek gedaan naar hoe de robot van 1 bepaalde toestand naar een andere gaat. Dit gebeurt door iedere keer de huidige toestand te vergelijken met de gewenste toestand. Er zullen dan commando's gestuurd worden naar de motoren om zo dichter bij de gewenste toestand te komen.
- Robotplanning → Gebeurt net zoals bij de robotsturing, maar dan op hoger niveau. Een planner gaat een taak in meerdere subtaken verdelen en deze dan doorsturen naar een controller (robotsturing).
- Kinematica → Dit domein beschrijft de wiskundige modellen die er gaan voor zorgen dat de robot bijvoorbeeld zijn armen kan bewegen naar een specifieke positie. Dit om de robot de mogelijkheid te geven om zijn opgegeven taken te vervullen.
- Robotarchitectuur → Hier wordt er gekeken naar het design van de robot. Ze willen hier hoofdzakelijk kijken welk design beter is voor de robot om het efficiënter en beter te maken voor specifieke taken.

2.3. Toepassingen van robotica

Er zijn vele toepassingen voor robots in onze moderne wereld, de meeste hiervan maken het menselijke leven veiliger of gemakkelijker.

In de industrie worden robotten ingezet voor massaproductie, waardoor het product sneller gemaakt kan worden. Voordeel voor de mens is dat het product ook goedkoper is. Natuurlijk is er ook een nadeel, het neemt werk af van de mensen.

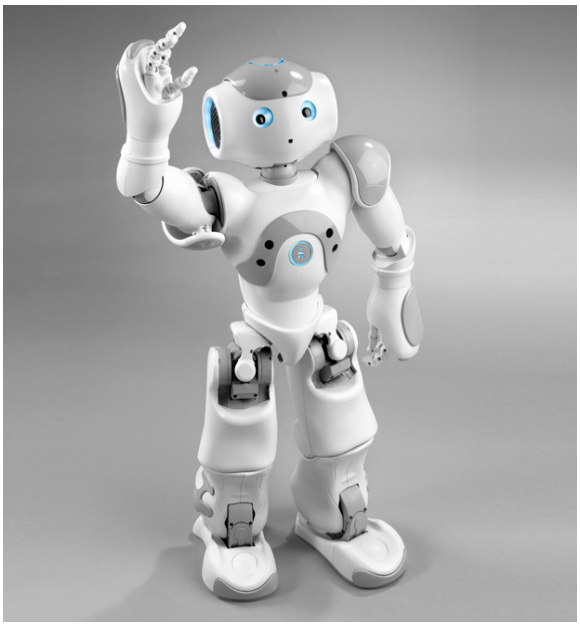
Niet enkel in de industrie worden robots gebruikt, ook in de geneeskunde. Robotten kunnen veel preciezer werken dan de dokters. De dokters moeten zeker niet bang zijn voor hun job, want in tegenstelling tot de industrie worden de robotten niet bestuurd door computers maar door de dokters zelf. De grote voordelen zijn hoofdzakelijk voor de patiënt zelf, want door het precisiewerk duurt de revalidatie minder lang.

Niet alleen op aarde zijn robots nuttig, ook in de ruimte. Robots worden hier gebruikt om waarnemingen te verzamelen en terug te sturen naar de aarde. Het grootste voorbeeld van deze toepassing is de Mars rover op Mars.

Robotten worden voornamelijk ingezet om de mensen te vervangen in gebieden waar het voor de mens gevaarlijk is, zoals de ruimte, het diepzee of het slagveld. Dit zijn niet de enige toepassingen waar robotten worden ingezet, kijk maar naar ZORA. ZORA wordt het meest gebruikt voor revalidatie van oudere mensen, dus robotten worden ook ingezet in de zorgsector.

3. NAO

3.1. Wat is NAO



NAO is één van de robotten die geproduceerd wordt door het Franse bedrijf Aldebaran. Deze robot is een autonome programmeerbare humanoïde robot die hoofdzakelijk door universiteiten en onderzoeksgroepen gebruikt wordt om onderzoek te doen naar robotica. Er worden niet enkel onderzoeken gedaan naar robotica, maar ook naar de praktische toepassingen die een robot zou kunnen hebben in het onderwijs.

Het 58 cm groot robotje was geboren in 2004, maar zijn debuut kwam pas in 2007. NAO 's debuut was in de RoboCup Standard Platform League, een internationale robot voetbalcompetitie, waar NAO het robothondje Aibo verving.

Er zijn verschillende software versies uitgebracht sedert 2008, maar de meest populaire versie was wel de NAO Academics-versie. Deze werd gemaakt voor de universiteiten. Nu is er al een nieuwe versie, genaamd NAO Evolution. Deze versie is al beschikbaar sinds 2014. Gelukkig kunnen de kopers van de NAO robot zelf hun

robot updaten met de laatste versies, want NAO is zeer populair bij de mensen. Sedert 2015 zijn er al 5000 NAO robotten in gebruik over 50+ landen wereldwijd.

3.2. NAO 's design

Wat maakt NAO een humanoïde robot? De definitie van een humanoïde robot is het volgende: "Een robot die een lichaamsvorm heeft die lijkt op die van een mens.". NAO heeft een lichaam dat lijkt op dat van een kind, maar een lichaam is niet voldoende. NAO heeft ook 25 motoren waardoor het verschillende bewegingen kan doen die mensen kunnen doen. NAO bezit 8 krachtsensoren en samen met het inert systeem kan het robotje in evenwicht blijven. Met de 2 camera's, 2 luidsprekers en de 4 microfoons kan NAO heel gemakkelijk met de mens communiceren.

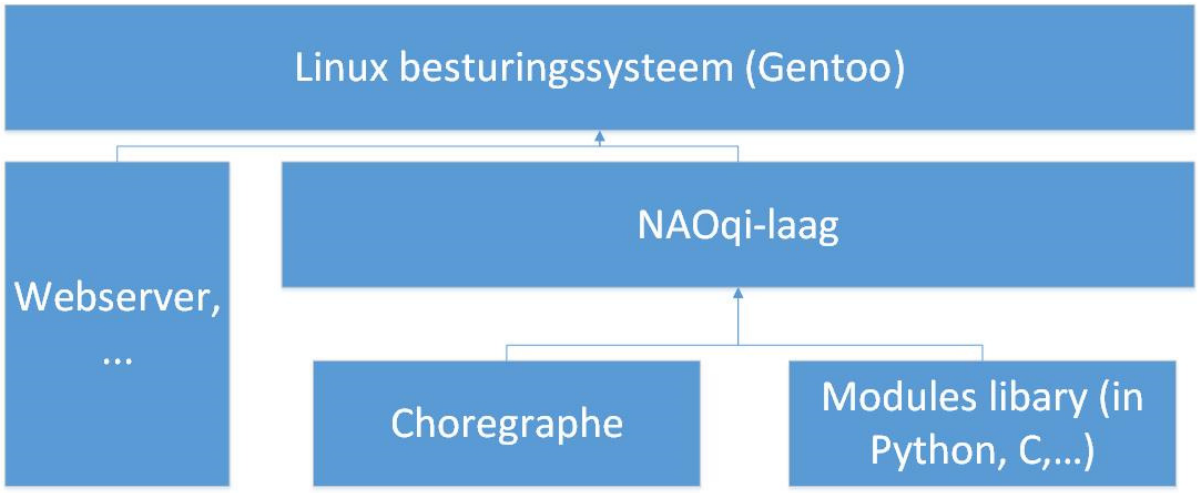
3.3. Onder de motorkap

Aan de basis van NAO ligt een gewoon Linux besturingssysteem, namelijk Gentoo. Hier gebeuren alle operaties die betrekking hebben tot de computer zelf, zoals tijdinstellingen, services activeren en meer. Direct onder het besturingssysteem bevindt zich de NAOqi-laag, die de motoren gaat aansturen. Daaronder liggen er ook nog 2 lagen, namelijk Choregraphe en Modules Libary.

> Choregraphe is een programma, geschreven door Aldebaran, dat het voor de gebruiker gemakkelijk maakt om applicaties voor de robot te maken.
> De Modules Libary, geschreven in Python, C, … , zijn verschillende klassen met daarin verschillende functies die het mogelijk maken om de commando's te versturen.

Beide lagen zenden commando's door naar de NAOqi-laag die op zijn beurt het vertaalt naar machinetaal en dan de motoren aanstuurt. Het enige verschil tussen de 2 lagen is dat de Choregraphe weinig tot geen programmeervaardigheden vereist, maar de Modules Libary vereisen dit wel.

NAO heeft ook een webserver lopen op de robot om zo de robot te besturen en applicaties te laten runnen via de web-interface.


Linux besturingssysteem (Gentoo)
Webserver, ...
NAOqi-laag
Choregraphe
Modules libary (in Python, C,...)

3.4. NAO versus ZORA

Er bestaat enige verwarring als het aankomt op NAO-ZORA. Dit is zeer begrijpelijk, want ze staan ook zeer dichtbij elkaar. Zo dicht zelfs dat ze praktisch hetzelfde zijn.

Zo kan je de NAO-ZORA relatie het best vergelijken:

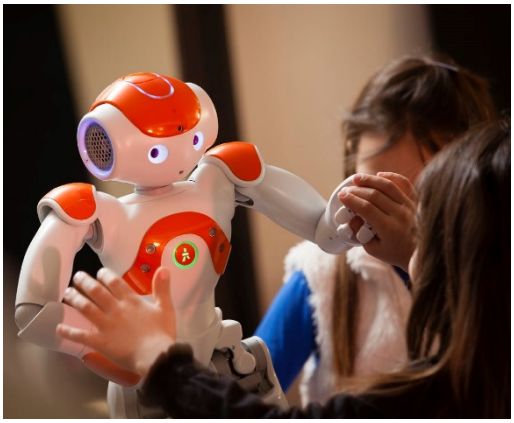
NAO is gewoon een lege robot. Er staat niets op, behalve de nodige elementen (OS, software). ZORA is, in tegenstelling tot NAO, geen robot maar een softwarepakket dat geïnstalleerd wordt op NAO.
Laten we stellen dat NAO een HP-computer is. Op deze computer staan er al verschillende zaken vooraf geïnstalleerd. Stel nu dat ZORA het Adobe-softwarepakket is. Op de HP-computer installeer je dan het Adobe-softwarepakket, omdat het een nuttige uitbreiding is.

Simpel gezegd ZORA is NAO, maar NAO is niet ZORA.

4. ZORA de zorgrobot

4.1. Wie/wat is ZORA

ZORA is een humanoïde robot, de eerste in zijn soort die wordt ingezet in de zorgsector. Het is een 58 cm grote robot met de naam **Z**org **O**uderen **R**evalidatie en **A**nimatie of kort **ZORA**. Zoals de naam al zegt, werkt zij hoofdzakelijk met ouderen om hen te helpen revalideren, maar zij werkt ook met kinderen. Net zoals bij de ouderen helpt ZORA de kinderen revalideren wanneer ze een ongeluk gehad hebben.



ZORA wordt al in verschillende wooncentra ingezet om de ouderen te helpen. De zorgverleners zijn heel blij met ZORA, zij is een aanvullend zorgmateriaal. Vroeger deden de zorgverleners de oefening voor, maar wanneer er één persoon extra aandacht nodig had dan hadden de zorgverleners een probleem. Ze moesten stoppen met de oefening en die persoon dan helpen, maar dan stopten de andere personen ook met de oefening. ZORA is de oplossing hiervoor. Nu gaat ZORA de

oefening voortonen en de zorgverleners kunnen dan de personen die het nodig hebben, bijsturen en de rest van de groep doet gewoon verder met ZORA.

Niet alleen in de wooncentra wordt ZORA ingezet als motivator, maar ook in scholen waar de leerlingen een leerachterstand of een mentale achterstand hebben. Hier heeft ZORA een groot voordeel omdat zij geen emoties toont. De kinderen verstaan haar en weten wat ze moeten doen. ZORA is klaar en duidelijk. Net zoals in de wooncentra gaat ZORA de oefening tonen en kan de leerkracht de kinderen bijsturen waar nodig. Samen met ZORA leren de kinderen alles op een speelse en leuke manier.

Iedereen is gewoon weg van ZORA en zeer blij dat ZORA hen helpt.

4.2. Verwarring bij ZORA

Er bestaat veel verwarring bij de mensen over het geslacht van ZORA. ZORA is nu wel een humanoïde robot die zeer dicht bij de mens staat, maar het blijft een robot en robotten hebben geen geslacht.

Natuurlijk is het een menselijke gewoonte om aan voorwerpen een naam te geven en naargelang de naam ook een geslacht. Kijk maar naar de kinderen die hun knuffelbeer een naam geven.

Om de mensen het gemakkelijker te maken hebben ze bij QBMT aan ZORA een geslacht gegeven. Als de naam nog niets prijsgeeft, dan zal het haar stem zijn die het zal weggeven. ZORA is een meisje.

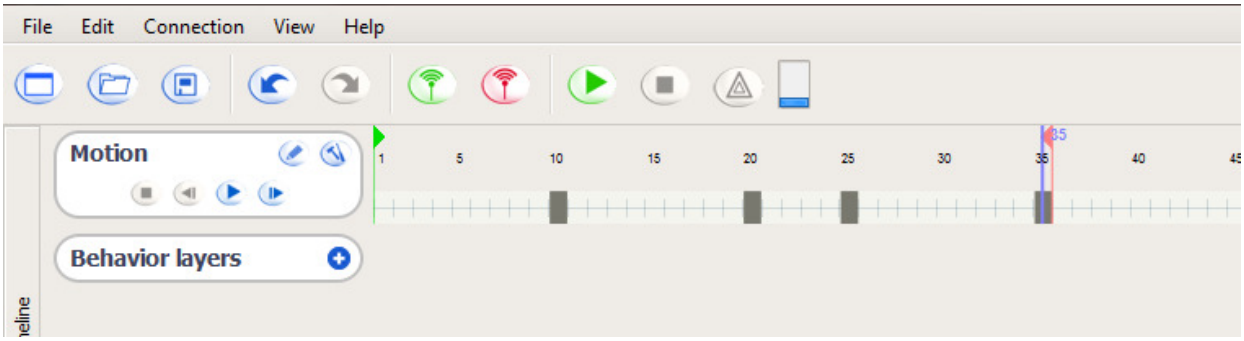
4.3. Bewegen met ZORA

De 'R' in ZORA staat voor **revalidatie**, dus moet ZORA oefeningen kennen en kunnen. Bewegingen maken is gemakkelijk, maar de bewegingen correct maken zodat ZORA alles kan doen zonder om te vallen of zonder zichzelf kapot te maken, is iets anders.

Er zijn veel bewegingen mogelijk en ZORA kan er al veel, namelijk 87 verschillende bewegingen. Elk van die bewegingen bracht een eigen moeilijkheid met zich mee. Zeker de bewegingen waar heel het lichaam aan bod komt.

4.3.1. Bewegingen maken

Bewegingen creëren gebeurt in het programma Choregraphe. Daar is er een tijdlijn waar je mooi de verschillende delen van een beweging kunt op plaatsen. Dan kun je de verschillende delen afspelen om zo een vloeiende beweging te creëren.



Om een actie (een deel van een beweging) te maken moet je ZORA in *"animation mode"* zetten. In deze mode kan je haar motoren zonder problemen in- en uitschakelen, zo kun je één van haar ledematen afzonderlijk laten bewegen.

Het is dan naargelang de instelling dat ZORA de verschillende acties gaat opnemen en in de tijdlijn plaatsen.

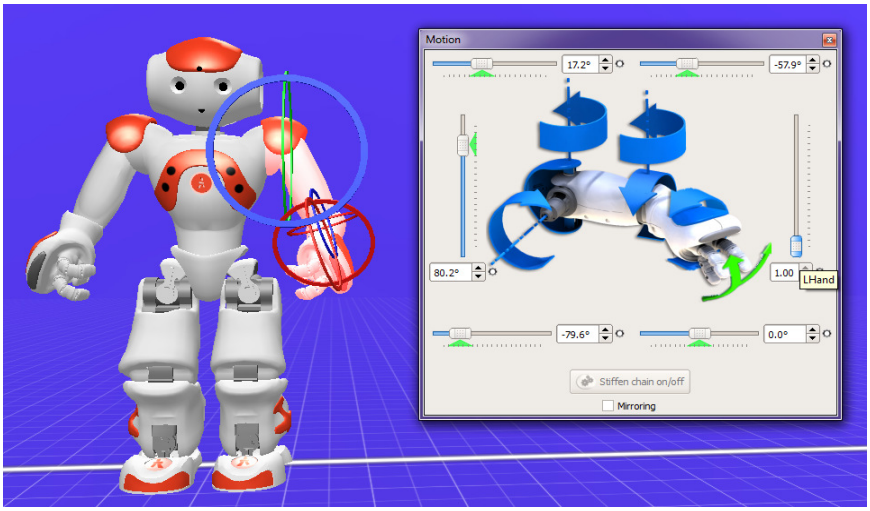
Dit zijn de verschillende opties:

- Na een bepaalde tijd onthouden wat de positie is van de motoren.
- Onthoud de positie van de motoren wanneer er op het hoofd wordt gedrukt.
- Onthoud de positie van de motoren wanneer er op de sensoren wordt gedrukt.
- Handmatig zeggen welke motoren er onthouden moeten worden en dan via een sneltoets die motoren in de tijdlijn plaatsen.

Er zijn verschillende motoren die kunnen onthouden worden. Deze motoren zijn verdeeld in 5 categorieën of gewrichten, een overzicht:

Hoofd	Linkerarm	Rechterarm	Linkerbeen	Rechterbeen
Yaw	ShoulderPitch	SchoulderPitch	HipYawPitch	HipYawPitch
Pitch	ShoulderRoll	ShoulderRoll	HipRoll	HipRoll
	ElbowYaw	ElbowYaw	HipPitch	HipPitch
	ElbowRoll	ElbowRoll	KneePitch	KneePitch
	WristYaw	WristYaw	AnklePitch	AnklePitch
	Hand	Hand	AnkleRoll	AnkleRoll

- De Pitch zorgt voor de hoogte van het gewricht.
- De Roll gaat meer kijken naar de afstand van het gewricht tot het lichaam.
- De Yaw zorgt ervoor dat het gewricht kan gedraaid worden.

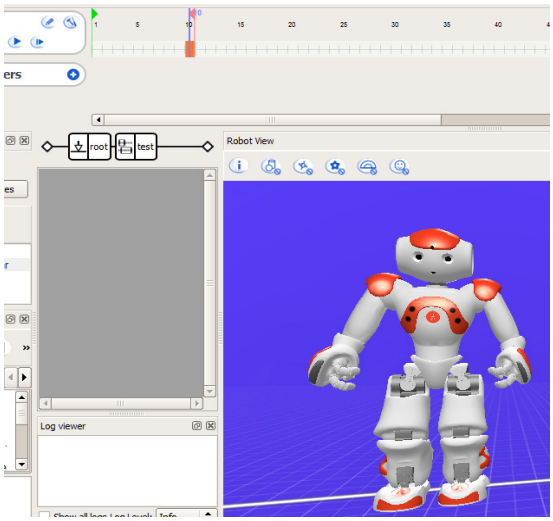


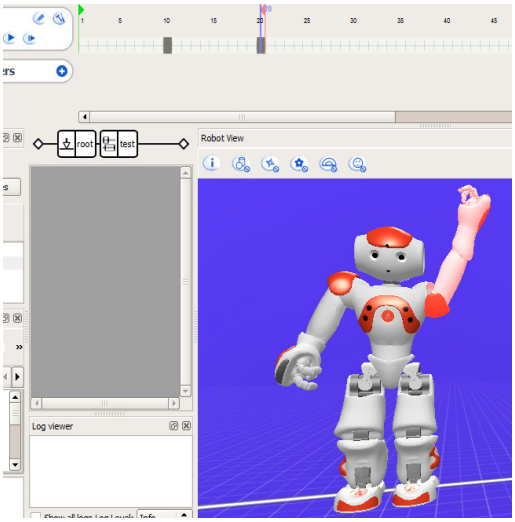
Er is ook iets genaamd *"Robot view"*. Hier kan je een digitale robot de bewegingen zien voortonen zonder dat je het op een robot moet testen. Dit kan zeer handig zijn, want de beweging kan misschien verkeerd lopen en zo de robot stuk maken.
In deze view kun je ook nog last minute kleine of grote aanpassingen maken zonder heel de tijdlijn opnieuw te moeten maken.

4.3.1.1. Case study: Linkerarm omhoog doen

De linkerarm omhoog laten gaan is één van de gemakkelijkste bewegingen die er bestaan.

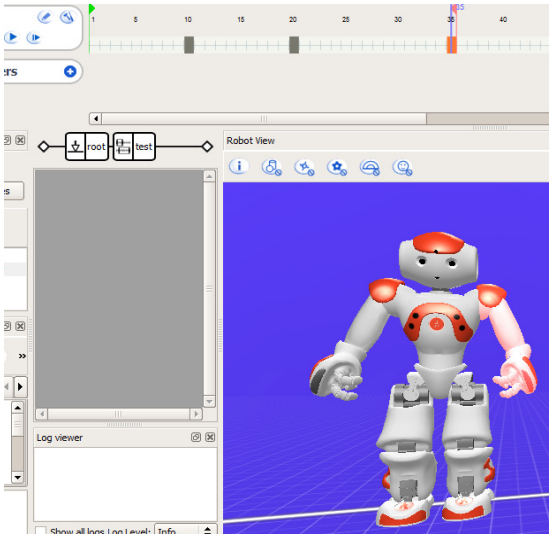
Voor je de beweging start, moet je in de tijdlijn een beginpositie instellen en dit doe je door alle motoren te onthouden (stap 1). Dan moet je de robot in animatiemode plaatsen en kun je heel gemakkelijk de linkerarm bewegen. Dan doe je de linkerarm omhoog en onthoud je de positie (stap 2).





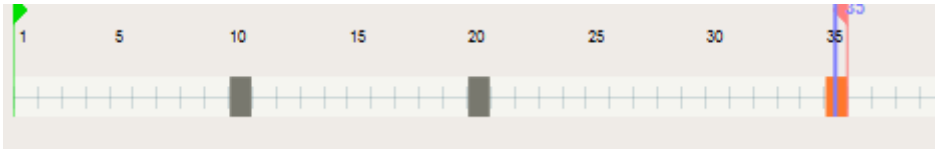
Stap 1 *Stap 2*

Natuurlijk moet de arm terug naar beneden. De gemakkelijkste manier om dit te doen is gewoon het eerste tijdlijnblok (zie stap 1) te kopiëren en die achter het 2de tijdlijnblokje (stap 3) te zetten.



Stap 3

Nu hebben we een simpele bewegingsoefening gecreëerd in Choregraphe. Ook andere oefeningen worden hier gemaakt. Natuurlijk hoe complexer, hoe meer gewrichten je nodig zult hebben en hoe complexer je tijdlijn er gaat uitzien. Nu zijn er maar 3 blokjes, maar bij kickboksen bijvoorbeeld gaat dat al gauw oplopen tot 30 blokjes.



Tijdlijn: Linker arm omhoog en omlaag doen

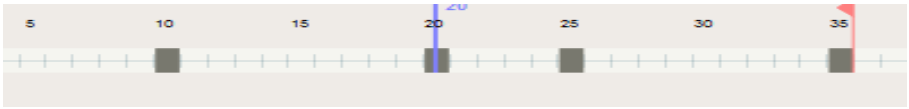
Tijdlijn: Kickboksen

4.3.2. Revalidatie oefening

In de case study is er een beweging gemaakt, maar is het nu een goede revalidatiebeweging? Stel dat een kinesist wil dat een patiënt zijn linkerarm 10 seconden omhoog houdt, dan moet er een nieuwe oefening gemaakt worden. Niet moeilijk, je verplaatst gewoon het blokje in de tijdlijn. Maar als je dat voor iedere

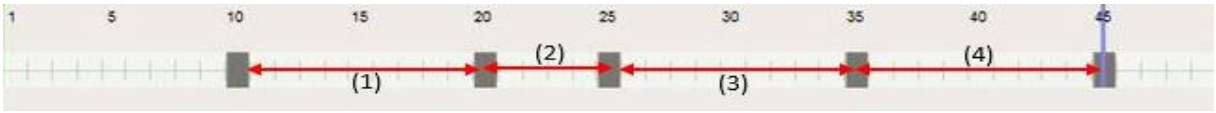
patiënt moet doen, is het niet echt haalbaar, dus maak je 1 algemene beweging waar de kinesist de parameters zelf kan aanpassen.

Dit doen we door een extra blokje toe te voegen aan de tijdlijn. Het blokje is identiek hetzelfde als het blokje van Stap 2 in de case study. Zo kun je later de bewegingparameters aanpassen.



Er zijn 4 verschillende bewegingparameters die de kinesist kan aanpassen (gebaseerd op de case study)

- Aanloopsnelheid (1)→ De snelheid waaraan de linkerarm omhoog moet gaan.
- Duur (2) → De tijd die de linkerarm omhoog moet blijven.
- Uitloop snelheid (3) → De snelheid waaraan de linkerarm omlaag moet gaan.
- Wachttijd (4) → De tijd tussen deze oefening en de volgende.



De parameters aanpassen wordt later besproken. (Zie 4.4.Bewegingen uitvoeren in de Composer)

4.3.3. Bewegingen die mogelijk zijn bij ZORA

Zoals eerder vermeld is het niet moeilijk om bewegingen te maken, maar ervoor zorgen dat ZORA in evenwicht blijft, is de uitdaging.

In het begin kon ZORA exact 4 bewegingen maken, waarvan 2 goed werkten maar de parameters konden niet aangepast worden. Deze 4 bewegingen zijn nu aangepast zodat je wel de parameters kunt aanpassen. Door deze aanpassing gaat ZORA niet vallen tijdens de uitvoering. Het aantal bewegingen is uitgebreid, nu zijn er 87 in totaal.

Een overzicht:

Bewegingen met de armen, vingers en polsen	**Bewegingen met de benen, knieën en enkels**
Armen omhoog (links, rechts en beide)*	Enkels horizontaal bewegen (links, rechts en beide)*
Armen zijwaarts omhoog (links, rechts en beide)	Enkels omhoog (links, rechts en beide)*

Armen zijwaarts half omhoog (links, rechts en beide)	Draaien met de enkels (links, rechts en beide)
Armen omhoog-midden, midden-omlaag, omhoog-midden-omlaag*****	Knieën omhoog (links, rechts en beide)*****
Draaien met de armen (linker, rechter en beide)	Benen omhoog (links, rechts en beide)*****
Vingers bewegen (links, rechts en beide)*****	Op 1 been staan (links en rechts)
Armen plooien (links, rechts en beide)*****	Warrior-Lunge (links en rechts)*****
Zwaaien met de armen (links, rechts en beide)*****	Knieën omhoog en benen uitstrekken
Polsen bewegen (links, rechts en beide)*****	Kickbox kicks
Boksen (links, rechts en beide)*****	**Bewegingen met de hoofd**
Hallo zwaaien (links en rechts)	Hoofd naar voren en naar achteren (1 beweging)*****
Kickbox slagen	Hoofd naar links en naar rechts (1 beweging)*****
Highfive	Draaien met het hoofd
Hand geven	Hoofd naar links en naar rechts (2 bewegingen)*****
Bewegingen met de armen, vingers en polsen (vervolg)	**Bewegingen met het hoofd (vervolg)**
Vuistje geven	Hoofd naar voren en naar achteren (2 bewegingen)*****
Fietsbewegingen maken met de armen	**Bewegingen hoofd-arm combinaties**
Molentje (vooruit en achteruit)	Hoofd naar voren met handen op het hoofd*****
Schouder bewegen (links en rechts)*****	Hand op hoofd (links, rechts en beide)*
Heel het lichaam	
Squat*****	Squat zonder armen*****
Hurken*****	Bridging*****
Tenen aanraken*****	Tenen aanraken vanop de rug*****
Draaien (links en rechts)	Wandelen (links, rechts, voren en achteren)
Pompen*****	

***** = waar de parameters gewijzigd kunnen worden

Bewegingen waar de parameters kunnen aangepast worden, zijn revalidatiebewegingen. De bewegingen waar je de parameters niet kan aanpassen, zijn gewone bewegingen om te tonen aan de mensen.

De bewegingen die beschreven werden, worden zeer actief gebruikt door de verschillende bejaardentehuizen en ziekenhuizen. De mensen zijn verbaasd als ZORA gaat beginnen pompen of kickboksen, maar hun favoriet is toch wel op één been staan. Deze beweging is ook het meest veranderd ten opzichte van de allereerste versie. In de eerste versie viel ZORA 80% van de keren, maar nu is ze zo stabiel. De mensen houden van ZORA en zijn volledig weg van haar bewegingen.

4.3.4. Onmogelijke bewegingen

Dit is zeer relatief gezien, ZORA kan elke beweging doen. Natuurlijk als de beweging ervoor zorgt dat ZORA valt, dan is het een slechte beweging en kan het niet gebruikt worden. Het is geen mooi zicht als ZORA iedere keer omvalt als ze een beweging uitvoert.

Elke beweging die gemaakt wordt, wordt eerst gezien als een onmogelijke beweging. Het is de job van de programmeur om te zorgen dat het een mogelijke en goede beweging wordt. Dit betekent iedere mogelijke manier bedenken om ZORA in een stabiele houding te houden. Het kan dan wel zijn dat er meerdere tijdlijnblokjes nodig zijn, dan is dit geen revalidatie oefening meer. Het is dan geen nutteloze beweging, maar een handige beweging om ZORA te demonstreren aan mensen. Zulke bewegingen zijn meestal wel spectaculair maar hebben meerdere tijdlijnblokjes nodig. Bij de spectaculaire bewegingen is het moeilijker voor de programmeur om ZORA in evenwicht te houden. Hoe spectaculairder, hoe moeilijker.

Zoals in 4.3.3.Bewegingen die mogelijk zijn bij ZORA beschreven zijn er heel wat bewegingen die ZORA kan doen zoals een mens. Er zijn daar geen verschillen te zien ten opzichte van de mens. Nu zijn er bewegingen die een mens kan, maar ZORA niet.

4.3.4.1. Bewegingen die mensen kunnen, maar ZORA niet

Een voorbeeld: *Benen achterwaarts strekken terwijl de handen een stoel vasthouden.*

Een mens kan deze beweging heel gemakkelijk maken. Wij gaan ons evenwicht verplaatsen van het ene naar het andere been en we gaan ook steunen op de stoel zelf. Wij hebben spieren in onze armen en kunnen er kracht opzetten, maar ZORA

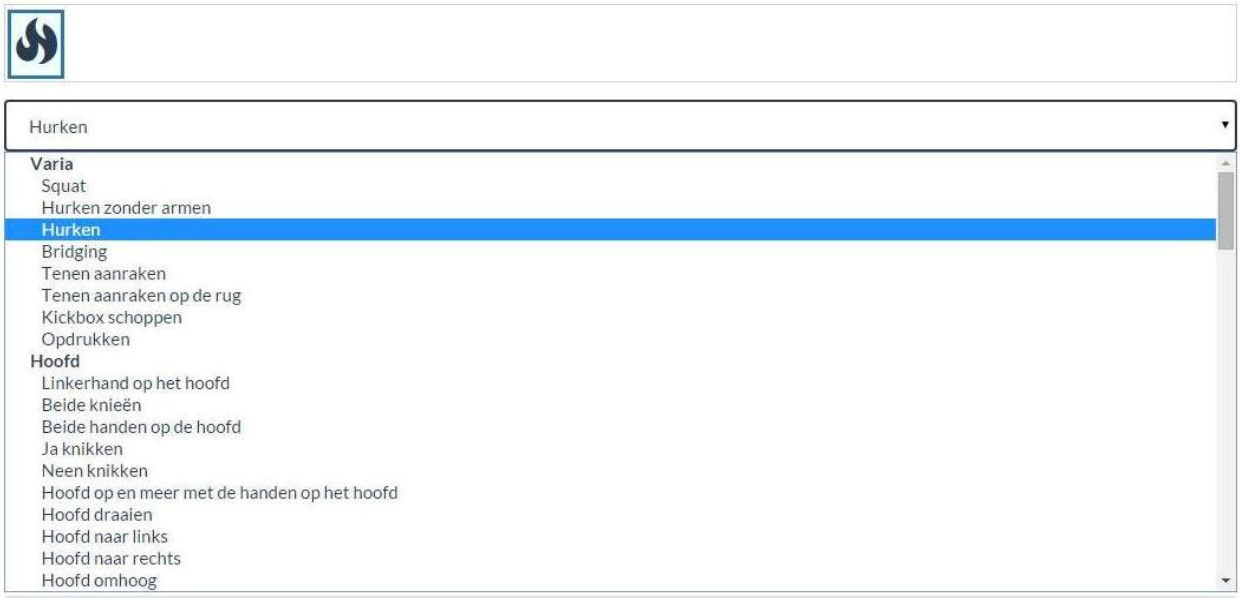
heeft enkel motoren. Dus wanneer je het ene been gaat verzetten en de steun wegneemt, gaat het inert systeem van ZORA denken dat ze uit evenwicht is. Hierdoor gaat ZORA haar motoren afzetten en gaat ze vallen.

4.4. Bewegingen uitvoeren in de Composer

In Choregraphe worden de verschillende bewegingen gemaakt. Het uitvoeren van de bewegingen in Choregraphe kan ook, maar dan enkel om het te testen. Dan kan je geen enkele parameter wijzigen, dus moet er een andere oplossing zijn. Die oplossing noemt de "*Composer*" en is onderdeel van ZORA-control. De Composer bestaat uit 2 delen, de front-end AngularJS en de back-end Python Modules.

4.4.1. Front-end AngularJS

In de front-end van de Composer kun je de beweging/oefening kiezen die ZORA moet voortonen.



Hier kan je de parameters wijzigen van de geselecteerde oefening, als het mogelijk is natuurlijk.

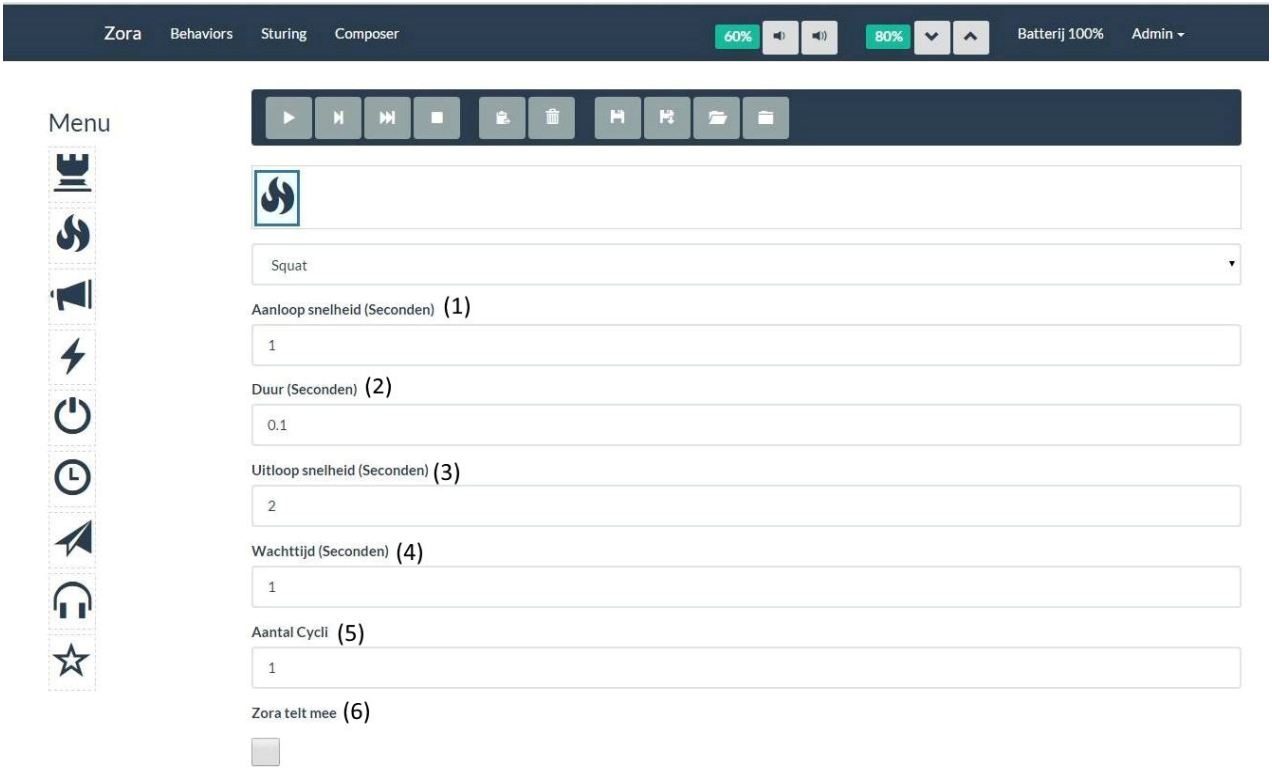
Zoals eerder vermeld, zijn er 4 parameters voor de oefening/beweging zelf :

- Aanloop snelheid (1)→ De snelheid waaraan de beweging moet starten.
- Duur (2) → De tijd tussen het begin van de oefening en het einde van de oefening.
- Uitloop snelheid (3) → De snelheid waaraan de beweging moet eindigen.
- Wachttijd (4) → De tijd tussen de oefening/beweging en de volgende.

In de Composer zijn er nog 2 parameters die de mensen kunnen veranderen:

- Aantal cycli (5) → Hoeveel keer moet ZORA deze oefening/beweging doen?
- Mee tellen (6) → Moet ZORA meetellen met de oefening/beweging?

Deze parameters (5,6) gaan meer zeggen over hoe ZORA met de oefening/beweging moet omgaan.



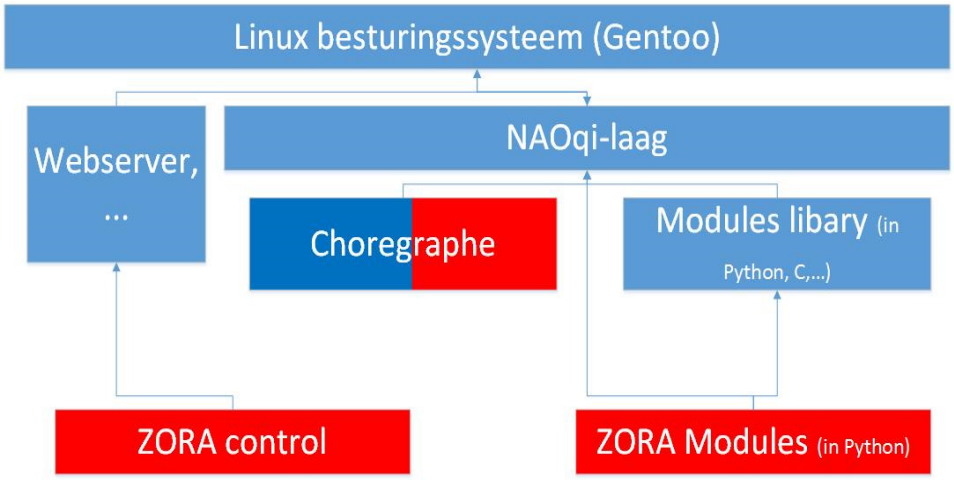
Wanneer er op Play gedrukt wordt, worden alle gegevens verstuurd naar de back-end. Deze back-end is een eigen geschreven Python Module genaamd ZOGym.

```
$qi.call(function(ZOGym,ALMotion){
    var parameters = {'exercise':self.exercise,'speedIn':self.speedIn,'duty':self.duty,'speedOut':self.speedOut,
                      'period':self.period,'cycles':self.cycles,'countCycles':self.countCycles};
    parameters = angular.toJson(parameters);
    ZOGym.execute(parameters).then(function(){ deferred.resolve();});

}
```

Versimpelde versie van de code

4.4.2. Back-end Python Module

Er zijn 2 soorten Modules Libary, de reeds toegankelijke van Aldebaran en die van QBMT/ZORA zelf.



De modules van Aldebaran zijn krachtig maar beperkt. Als je meerdere modules wil combineren gaat dit niet, want de ene module kent de functies niet van de andere. Dus moest er een oplossing komen en dit kwam er door zelf modules te schrijven, die verschillende Aldebaran modules kunnen combineren. Een extra voordeel dat er bij komt, is dat je extra functionaliteit kunt toevoegen aan een functie. Je kunt checken op parameters, eerst iets anders laten uitvoeren of eerst iets uit de database halen.

De zelf geschreven modules noemen ZORA Modules en die communiceren zowel met de NAOqi-laag als met de Modules Libary van Aldebaran.

4.4.2.1. ZOGym Module voor bewegingen

Een voorbeeld van een zelf geschreven module is de ZOGym Module. Deze module gaat ervoor zorgen dat de juiste oefening/beweging gaat uitgevoerd worden. We lopen door een lijst en kijken welke oefening/beweging er bij past. Als de oefening/beweging gevonden is, wordt de execute functie opgeroepen.

```
class ZOGymModule(ALModule):
    def __init__(self, name):
        ALModule.__init__(self, name)
        self.postureProxy = ALProxy("ALRobotPosture")
        self.motionProxy = ALProxy("ALMotion")

        self.exercises = {"squat":Diffrent.Diffrent("squat"),
            "hamstring":Diffrent.Diffrent("hamstring"),
            "bridging":Diffrent.Diffrent("bridging")}


    def execute(self,jsonParameters):
        parameters = json.loads(jsonParameters)
        ex = self.exercises[parameters["exercise"]]
        ex.execute(float(parameters["speedIn"]), float(parameters["duty"]),
            float(parameters["speedOut"]), float(parameters["period"]),
            int(parameters["cycles"]),bool(parameters["countCycles"]))
```

Elke oefening/beweging zit in een specifieke klasse die alles weet over de oefening/beweging. In 4.3.2.Revalidatie oefening werd er gesproken over beweging parameters aanpassen, wel hier gebeurt het. Door de beweging in Choregraphe om te zetten in een bézierkromme-functie is het heel gemakkelijk om de parameters aan te passen.

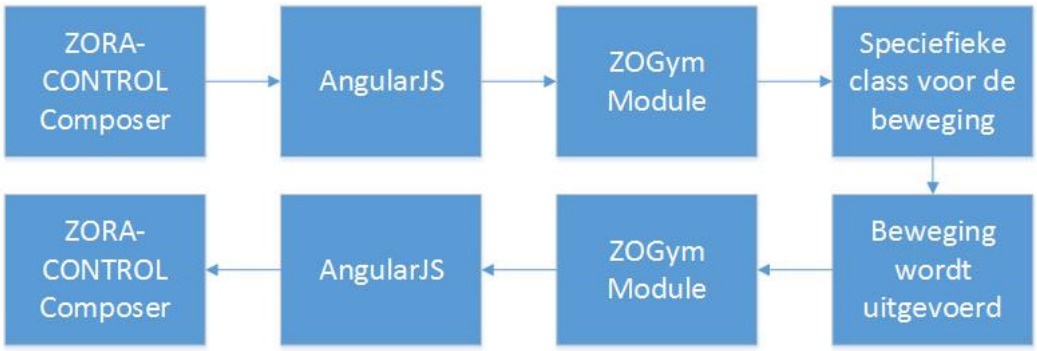
Om de parameters door te voeren moet je gewoon de tijdsmomenten, die gegenereerd worden door de bézierkromme-functie veranderen. Wanneer dit gebeurd is, worden de parameters gerespecteerd en gaat de oefening/beweging gebeuren hoe de kinesist het wil.

```
class Diffrent():
    def execute(self, speedIn=1, duty=0.5, speedOut=2, period=0.5, cycles=1, countCycles=True):
        for x in range(0, cycles):
            if(countCycles):
                self.tts.post.say(str(x+1))
            self.squat(speedIn, speedOut, duty)
            time.sleep(period)
        pass
    def squat(self, speedIn, speedOut, duty):
        timeArray = self.fffffffTimeArray(speedIn, speedOut, duty)

        names = list()
        times = list()
        keys = list()

        names.append("HeadPitch")
        times.append(timeArray)
        keys.append([[-0.147306, [3, -0.0133333, 0], [3, 0.72, 0]], [-0.33292, [3, -0.72, 0], [3

        names.append("HeadYaw")
        times.append(timeArray)
        keys.append([[-0.0153821, [3, -0.0133333, 0], [3, 0.72, 0]], [-0.0153821, [3, -0.72, 0]

        names.append("LAnklePitch")
        times.append(timeArray)
        keys.append([[0.0966001, [3, -0.0133333, 0], [3, 0.72, 0]],[-0.489388, [3, -0.72, 2.0373

        names.append("LAnkleRoll")
        times.append(timeArray)
        keys.append([[-0.133416, [3, -0.0133333, 0], [3, 0.72, 0]], [-0.030638, [3, -0.72, 0], [
```

Vereenvoudigde code (rood=de 4 verschillende bewegingen parameters; groen = de bézierkromme-functie

4.4.3. De flow van een beweging

Hoe de flow, de uitvoering van een beweging eruit ziet, kan je hier zien:



Kritische reflectie

Mijn stage is zeer goed verlopen. Alles werkte wat ze me vroegen qua programmeren, maar de manier waarop het werkte, was soms niet goed. Er zijn altijd momenten waarop een programmeur zijn code herbekijkt en denkt "*Wat heb ik hier in godsnaam gedaan*". Ik ben zeer blij dat ik deze situatie niet zoveel had, maar wanneer ik het wel had, deed ik alles om de code beter te maken. Dubbele codes verwijderen, een methode dynamischer maken, zo weinig mogelijk if-else gebruiken, maar in sommige gevallen moest je je erbij neer leggen dat het niet anders kon. De code kon niet aangepast worden.

Als ik deze stage opnieuw zou mogen doen, dan is mijn eerst taak de composer een beetje opkuisen. Dan bedoel ik niet qua code, deze was goed, dan heb ik het over het design en het gebruik ervan. Nu moet je op 3 verschillende plaatsen praktisch dezelfde code toevoegen, dit is natuurlijk NOT DONE. Een mogelijke oplossing is het gebruik van een database. Dit kan een zeer goede oplossing zijn, maar is het de beste oplossing? Ik ben wel fan van het gebruik van een database. Zelf kan ik er heel gemakkelijk mee werken en is het ook dynamischer. Met een database gaan alle parameters op één plaats zitten en moet je het maar op één plaats aanpassen, wat heel gemakkelijk is.

Ik moet natuurlijk ook meer groeien in het programmeren en dan hoofdzakelijk meer gebruik maken van design patterns. In ZORA-control wordt er gebruik gemaakt van de factory pattern. Mijn eerste reactie was niet echt positief, omdat ik design patterns ingewikkeld vond. Die mening veranderde zeer snel, ik zag de kracht van design patterns in waardoor ik het niet meer ingewikkeld vond. In school leerden we verschillende design patterns zeer snel. Dit om er zoveel mogelijk te zien, maar het waren er te veel. Ik vind persoonlijk dat je best een paar, de meest gebruikte, er uitneemt en dat samen gaat implementeren. Er zijn mensen die dat niet leuk vinden omdat het dan dactylo is, maar zo let je wel op en ben je mee met wat er gebeurt. Voor mij mag dat wel meer gebeuren in de programmeerlessen. Nu ik zelf het voorbeeld zag van wat design pattern kan teweegbrengen, vind ik dat design pattern veel vroeger in de lessen moet aan bod komen, bijvoorbeeld in semester 2. Ik zou ook durven zeggen dat er al veel sneller moet geleerd worden hoe codes moeten geschreven worden. Sommige lectoren doen dit al, anderen niet waardoor het verwarrend wordt als student.

Persoonlijk vind ik dat ze meer aandacht aan "het testen" moeten geven in school en er ook niet zo snel overgaan. In de praktijk moet er veel getest worden om een goed eindresultaat te behalen.

Conclusies

Onderzoeksvraag: *Zijn de bewegingen van een robot accuraat ten opzichte van de bewegingen van een mens?*

Zoals eerder vermeld wordt er in het onderzoek enkel gewerkt met een humanoïde robot. Bij andere soorten robotten kan het zijn dat er andere resultaten gaan uitkomen.

Met 80+ verschillende bewegingen die getest zijn kan je snel zien wat een factor gaat spelen bij de accuraatheid van de bewegingen. Bij die factoren moet je dan ook veel aandacht besteden als je bewegingen maakt. Dan zal je wat fantasie moeten hebben om die problemen te overkomen. Zo kun je een beweging hebben, zoals molentje met de armen, die je niet perfect kan uitvoeren omdat de armen te klein zijn, maar dat wil niet zeggen dat de beweging niet accuraat is.

De factoren of problemen die je moet overkomen zijn namelijk:

- evenwicht van de robot;
- de soort beweging;
- de lengte van de ledematen;
- het aantal motoren.

Het laatste is een zeer belangrijke factor en ook de factor waar je niets kan aandoen. Als de motor niet bestaat, kan je de beweging ook niet maken.

Voorbeeld: De pols op en neer doen. Dit gaat niet lukken, want ZORA heeft enkel een motor die de pols kan draaien. Je kan de volledige arm omhoog doen en terug naar beneden, maar een mens gaat dit niet doen.

Algemeen kunnen we stellen dat bij een humanoïde robot de bewegingen zeer accuraat zijn aan de bewegingen die mensen zouden maken. Je kunt ook concluderen dat hoe meer motoren er zijn, hoe accurater de bewegingen zijn.

Bronnen- & literatuurlijst

Bronnen:

- http://nl.wikipedia.org/wiki/Robotica
- http://nl.wikipedia.org/wiki/Mechatronica
- http://en.wikipedia.org/wiki/Nao_%28robot%29
- http://doc.aldebaran.com/2-1/home_nao.html
- http://zorarobotics.be/
- http://doc.aldebaran.com/2-1/software/choregraphe/index.html
- Interne documenten
    - De grote ZORA boek
- De ZORA presenteert applicatie
- Ervaring van Michaël als verkoper
- Uitleg over de interne werking van ZORA door Wim Devos
- Foto's
    - Van ZORA: komen van QBMT zelf
    - De rest komt van google afbeeldingen

Literatuur:

- https://angularjs.org/
- http://doc.aldebaran.com/2-1/dev/js/index.html?highlight=qimessaging
- https://docs.angularjs.org/api/ng/service/$q
- https://docs.python.org/2/