Stageverslag robotarm

Herman Heringa 0977832

CMI-TINSTG04-1

Begeleiders:

Thijs de Ruiter

Jacques de Hooge

Bas Klein

Inhoud

[Voorwoord 3](#_Toc63115920)

[Stagebedrijf 4](#_Toc63115921)

[Werkzaamheden 5](#_Toc63115922)

[projectonderdelen 5](#_Toc63115923)

[Startmenu 5](#_Toc63115924)

[Setup interfaces 5](#_Toc63115925)

[Besturingsinterface 6](#_Toc63115926)

[Programmeerinterface 7](#_Toc63115927)

[Robotarm library 8](#_Toc63115928)

[Batch/shell scripts 8](#_Toc63115929)

[Minecraft pi connectivity 9](#_Toc63115930)

[Minecraft java mod 9](#_Toc63115931)

[Pygame interface 9](#_Toc63115932)

[Logboek 10](#_Toc63115933)

[Conclusie 17](#_Toc63115934)

[Aanbevelingen 17](#_Toc63115935)

[Projectdoelen 18](#_Toc63115943)

# Voorwoord

Het afgelopen half jaar heb ik met veel plezier aan mijn stage gewerkt. Het resultaat daarvan is nu hier te zien. Ik heb mijn best gedaan om een mooi product in elkaar te zetten, en om het werk dat ik gedaan heb hier te beschrijven. Dit is een jaar geweest van veel onzekerheid, en dat heeft helaas ook invloed gehad op het stageproces. Desalniettemin ben ik blij met het resultaat. Ik heb veel geleerd de afgelopen periode en daar wil ik graag mijn bedrijfsbegeleider Bas Klein, stagebegeleiders Thijs de Ruiter en Jacques de Hooge, medestagairs Mark van de Heuvel en Nordin el Azzouzi, het Cambridge Innovation Center team en de mensen op Stack Overflow voor bedanken. Zonder jullie hulp zou dit project nooit zo geworden zijn als het nu is.

Rotterdam, 1-2-2021

Herman Heringa

# Stagebedrijf en aanleiding

Ik heb stage gelopen bij Stichting 010010, een bedrijf zonder winstoogmerk dat lespakketten maakt om basisschoolleerlingen les te geven op het gebied van computerwetenschappen en robotica (samengevat als “de 21e eeuw”). Het is een vrij klein bedrijf (5fte) en heeft dus niet echt afdelingen. Mijn opdracht is het maken van leshulpmiddel, bestaande uit een Raspberry Pi en een robotarm, om kinderen mee te leren programmeren. Het idee is dat de robotarm software-concepten tastbaar maakt: programmeren wordt een stuk minder abstract voor kinderen als ze er de echte wereld mee kunnen beïnvloeden. Oorspronkelijk heb ik bij het bedrijf gesolliciteerd voor een andere opdracht, het ontwikkelen van een draadloze aansturing van een robotauto. Die opdracht stond op de vacaturebank van praktijklink en leek me wel leuk. Tijdens het sollicitatiegesprek hebben we besloten dat de huidige opdracht wat uitdagender en leuker zou zijn. Dat was maar goed ook, want later bleek dat de oorspronkelijke opdracht bedoeld was voor 2e-jaars studenten.

# Gestelde eisen

Omdat de opdracht eigenlijk pas vorm kreeg tijdens het sollicitatiegesprek waren er weinig harde eisen. De oorspronkelijke eisen waren:

* Een robotarm van type OWI-535 moet op dusdanige manier aangesloten worden op een raspberry pi dat deze door de pi aangestuurd kan worden.
* De manier van aansluiten moet eenvoudig genoeg zijn om door een kind van 10 jaar oud uitgevoerd te worden.
* Er moet een programmeerinterface geschreven worden, van waaruit de robotarm aangestuurd kan worden.
* De programmeerinterface moet worden geschreven in Python 3.
* De programmeerinterface moet te begrijpen zijn voor een kind van 10 jaar oud.

Lopende het project zijn er meer eisen bijgekomen:

* Code en comments moeten in het Engels geschreven worden, documentatie in het Nederlands.
* Code moet zoveel mogelijk voldoen aan de PEP8 standaard, met uitzondering van maximale regellengte en het gebruik van snake\_case.
* Code moet Object Oriented geschreven zijn.
* Het moet mogelijk zijn ook andere apparaten dan de robotarm op de programmeerinterface aan te sluiten en deze hiermee aan te sturen zonder de code van de interface te hoeven veranderen.
* De robotarm moet gesimuleerd kunnen worden (met behulp van prints) als deze niet aangesloten is.
* De programmeerinterface moet op elk (veelgebruikt) os kunnen draaien.
* Er moet een besturingsinterface geschreven worden, waarmee de robotarm (en andere mogelijke aangesloten apparaten) met het toetsenbord aangestuurd kunnen worden.
* De keybinds van de besturingsinterface moeten in de interface aangepast kunnen worden en worden opgeslagen.

Ook zijn er een aantal wensen (geen harde eisen, maar ergens tussen should- en could haves) opgesteld:

* Kalibratiecode voor de robotarm: als de arm geijkt kan worden zou deze een stuk preciezer te besturen zijn.
* Support voor meerdere talen: met een systeem om een taal te selecteren zou het makkelijker zijn het programma te vertalen. Dit verlaagt de drempel voor de stichting om te internationaliseren.
* Webversie van de gui: de code zou op een door de stichting gehoste website kunnen draaien, zodat iedereen de code kan gebruiken, ook als ze geen rechten hebben om programma’s op hun computer te installeren.
* Meebewegend 2d- of zelfs 3d-model: In plaats van prints voor de simulatiecode zou een meebewegende virtuele robotarm leuker zijn.
* Mogelijkheid tot aansturing vanuit minecraft: Het veranderen van iets in de echte wereld door in een computerspel ergens op te klikken zou erg leuk zijn voor kinderen en een hoop interesse opwekken.
* Mogelijkheid voor het aansluiten van sensoren: het kunnen aansluiten van input-devices zou een hoop extra functionaliteit aan de programmeerinterface geven.

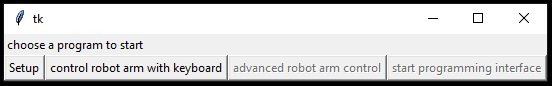
# Werkzaamheden

Dit hoofdstuk bestaat uit twee delen: de onderdelen waar ik tijdens de stage aan gewerkt heb en mijn logboek. Het eerste deel legt uit hoe de verschillende onderdelen van het programma werken. Ook geeft het informatie over eerder gemaakte en geschrapte onderdelen. Het logboek is eigenlijk een chronologische uitleg van mijn werkzaamheden tijdens het project.

## projectonderdelen

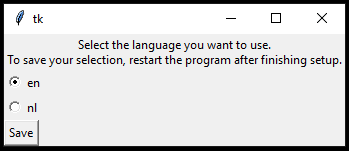
### Startmenu

Het startmenu bevat knoppen voor de verschillende sub-programma’s. Ook update het menu de progress van de gebruiker en unlockt het hiermee nieuwe sub-programma’s.

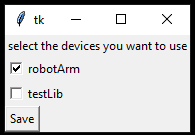


### Setup interfaces

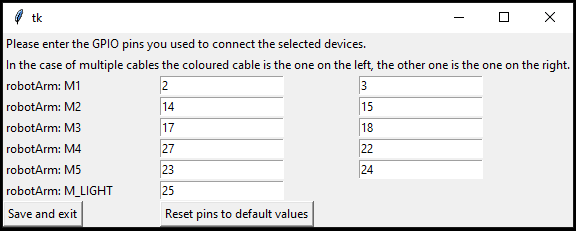
Dit deel van het programma bestaat uit drie interfaces:



1. In de eerste interface kan een taal geselecteerd worden. Het leest de namen van de bestanden in ./localisation/ en maakt voor elk localisatiebestand een optie aan. Deze geselecteerde optie wordt opgeslagen in config.ini.



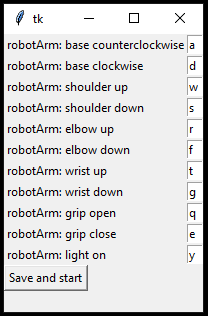
1. In de tweede interface kunnen de aangesloten devices geselecteerd worden uit een lijst met beschikbare libraries. Hiervoor wordt eerst gekeken naar de submappen in ./lib. Per submap wordt vervolgens gekeken of die de vereiste bestanden bevat. Als dat het geval is wordt de library/submap aan de lijst toegevoegd. Vervolgens wordt deze lijst vergeleken met de lijst van libraries in config.ini. Nieuwe libraries worden toegevoegd aan de lijst in de ini en oude worden verwijderd. Daarna wordt er in config.ini gekeken of de library “aan of uit staat” (eerder geselecteerd is) en worden de geselecteerde libraries aangevinkt. Nadat de gebruiker een keus gemaakt heeft wordt die opgeslagen in config.ini. Daarna wordt de lijst met geselecteerde libraries doorgestuurd naar het volgende menu.

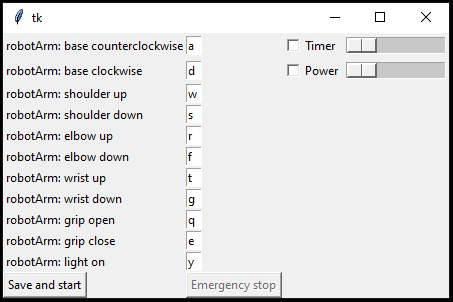


1. In de derde interface kunnen de gebruikte pins per device geselecteerd worden. Voor elke library worden de ingestelde pins opgehaald uit pinout.ini. Deze worden dan in de interface gezet. Nadat de gebruiker de gebruikte pins ingesteld heeft worden deze weer in de goede pinout.ini’s opgeslagen.

### Besturingsinterface

De besturingsinterface heeft twee modi: een normale en een geävanceerde. Hiertussen kan gekozen worden in het startmenu. Het enige verschil is dat de geävanceerde modus ook een tijd en PWM-sterkte in kan stellen. Omdat je met een vooraf ingestelde tijd de motor verder kan draaien dan de bedoeling is heeft de geävanceerde modus ook een nood-stop. Deze is niet per se nodig, de motoren hebben een beveiliging tegen te ver doordraaien, maar omdat de motoren een hard tikkend geluid maken wanneer ze in die beveiliging schieten is het fijn om ze direct uit te kunnen zetten.





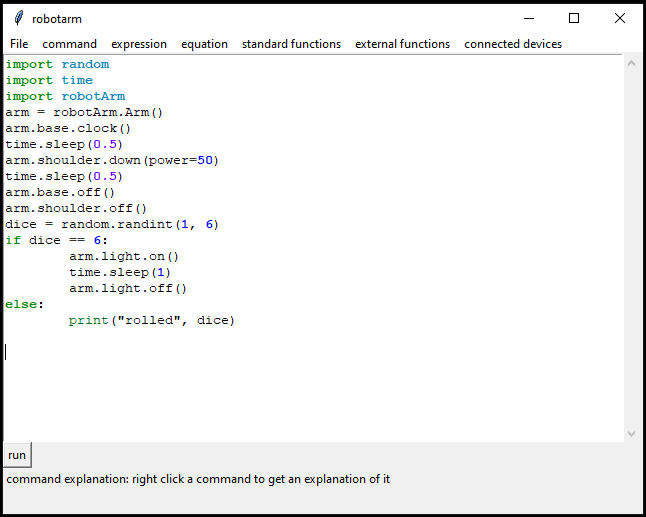
De besturingsinterface begint met kijken in config.ini welke devices/libraries zijn aangesloten. Vervolgens haalt het programma voor iedere library de bijbehorende keybinds op uit lib/[library]/controls.ini en zet ze in de interface. Nadat de gebruiker deze naar zijn voorkeur heeft aangepast en opgeslagen kan hij de apparaten besturen met de ingestelde toetsen. Desgewenst kan hij de besturing weer aanpassen en opslaan. Ook is het mogelijk om meerdere onderdelen tegelijk aan te sturen.

### Programmeerinterface

De programmeerinterface bestaat uit een tekstvak en een invoegbalk. Via de invoegbalk kunnen stukjes code in het tekstvak gezet worden. Het idee hierachter is dat er meerdere dingen tegelijk worden gedaan: indien nodig wordt er een library geïmporteerd en/of een object van een klasse gemaakt en de daadwerkelijke opdracht wordt geschreven. Ook kan de balk als een cheat-sheet gebruikt worden: een gebruiker kan zo direct alle (nuttige) functies in een library zien, zonder daarvoor in documentatie of op internetfora naar te zoeken.

De code in het tekstvak kan opgeslagen worden als .py bestand of direct uitgevoerd worden in de interface. Ook kunnen .py- (of tekst-) bestanden in het tekstvak geöpend worden.

Elke keer dat er een niewe regel wordt begonnen, en bestand geöpend of een code snippet wordt ingevoerd wordt er syntax-highlighting code gedraaid (met dank aan de pygments library).



Deze interface is meer een proof of concept dan een volledig bruikbaar product. Er missen nog een hoop onderdelen die men zou verwachten in een programmeeromgeving, zoals een undo-functie, error-detectie, breakpoints, suggesties etc.

### Robotarm library

Vrij simpele library voor het aansturen van de robotarm. Bestaat uit een klasse Arm en een klasse Part, met subclasses voor de individuele gewrichten van de robotarm. De Arm class maakt objecten aan voor de individuele objecten. En test of de code op een pi gedraaid wordt. Als dat het geval is worden de pins ingelezen uit pinout.ini en ingesteld. Via Arm kunnen de methods van de individuele objecten aangeroepen worden. Als de code op een pi draait worden de relevante pins aan en uit gezet door de methods (met PMW, mocht dat gespecificeerd zijn). Als dit niet het geval is worden de opdrachten naar de console geprint (als simulatie).

### Batch/shell scripts

Run.bat en run.sh zijn scripts om het programma te starten op respectievelijk Windows en Unix/MacOS. Beide scripts doen hetzelfde: als het programma start wordt de ingestelde taal uit config.ini gehaald. De scripts kopiëren (en overschrijven) de bijbehorende “taal-libary” uit ./localisation/ naar ./bin/localisationdata.py. Dit doen ze ook voor alle libraries (dus bijvoorbeeld ./lib/robotArm/localisation/en.py naar ./lib/robotArm/robotArmLocalisationdata.py). Als er geen library beschikbaar is voor de ingestelde taal wordt de engelse library gebruikt. Ook maken ze per library pinout.ini en controls.ini aan als die nog niet bestaan (kopie van de default bestanden). Daarna wordt met pip gekeken of de pygments library al geïnstalleerd is (voor de syntax highlighting) Als dit niet het geval is wordt deze (met pip) geïnstalleerd. Hierna wordt het programma gestart.

### Minecraft pi connectivity

Oorspronkelijk was het plan om ook code te schrijven om de robotarm aan te sturen vanuit minecraft pi. Hier heb ik demo-code voor geschreven. Helaas bleek minecraft pi erg gelimiteerd. De pi-versie van minecraft is gemaakt om code te leren schrijven, niet om dingen in het spel zelf te doen. Dit maakt het wel geschikt voor een mogelijke toekomstige koppeling aan de programmeerinterface, maar niet als losstaand project. De demo-code is dan ook uit het project geschrapt (maar nog wel te zien op <https://github.com/HermanHeringa/robotarm/blob/8a357a91d33b44046aab295bc0ea7e8e564ccbe7/bin/McRobotarmTest.py>)

### Minecraft java mod

Vanwege de ervaringen met Minecraft Pi is besloten om te kijken of het mogelijk is de java-versie van minecraft te gebruiken, zodat kinderen de robotarm met *redstone* (de in-game versie van programmeren) aan te sturen. Na onderzoek bleek de pi in staat om minecraftversies tot 1.12.2 te draaien. Versies hierna vereisen een 64-bit Linux distro (zie <https://help.minecraft.net/hc/en-us/articles/360035131371-Minecraft-Java-Edition-system-requirements->). De 64-bits versie van Raspbian is op het moment van schrijven nog steeds in beta en een stuk minder stabiel dan de 32-bit versie. Hierom is gekozen het bij versie 1.12.2 te houden. Met behulp van de Forge Modloader (<https://minecraftforge.net>) zou in Java een mod geschreven kunnen worden die kon communiceren met de python-code van de robotarm.

Dit bracht echter een aantal problemen met zich mee: Allereers is het een heel proces om de java versie van Minecraft op een pi te installeren. Dit vereiste een aantal workarounds:

Deze versie van minecraft (en Forge) werkt alleen op Java 7, een verouderde versie. Omdat ik de code op mijn laptop wilde schrijven betekende dit dat ik de verwijzingen naar Java in mijn PATH-variabele moest aanpassen naar die oudere versie, wat er weer voor zorgde dat andere Java programma’s niet meer werkten totdat ik de verwijzingen weer terug zette naar Java 15.

Op de pi moest deze versie gestart worden via Optifine (<https://optifine.net/>), wat weer ruzie gaf met Forge. Nadat ik dit eindelijk allemaal aan de praat had ging mijn laptop kapot (zie logboek 22/23 september). Mijn werklaptop was op Windows te traag om Eclipse te kunnen draaien, dus daar heb ik op aanraden van mijn bedrijfsbegeleider Linux op gezet. Op Linux begon het hele drama met path-variabelen en java versies weer opnieuw, maar nu op een OS waar ik verder geen tot weinig ervaring mee had. Na een volle dag zonder vooruitgang besloten dit project uit te stellen. Daarna niet meer aan begonnen.

### Pygame interface

Oorspronkelijk zou de besturing van de robotarm (en de programmeerinterface) via pygame gedaan worden. Het was erg makkelijk om pygame voor keybinds te gebruiken, dus een demo was al snel geschreven (dag 3 van de stage). Het bleek helaas een stuk ingewikkelder om deze library te gebruiken voor het maken van interfaces (zoals de naam suggereert is de library daar niet voor gemaakt). Eigen font ontwikkeld om tekst to kunnen laten zien. Na me een aantal dagen in te lezen in de documentatie besloten een andere library te gebruiken. Deze gevonden in de vorm van tkinter. Bijkomend voordeel van tkinter is dat de library standaard meegeleverd wordt met python, en dus niet handmatig geïnstalleerd hoeft te worden.

## Logboek

2 september:

opderzoeksplan robotarm en pi opgesteld, github en trello aangemaakt. User stories geschreven, planning gemaakt.

3 september:

robotarm en pi aangesloten, pi ingesteld, vnc viewer geinstalleerd, testcode geschreven om robotarm aan te sturen via usb-module. Geprobeerd arm direct aan te sturen via windows (en usb-module), dit is niet gelukt. Mosfets besteld om aansluiting via GPIO-pins te kunnen doen.

4 september:

library voor aansturing vanaf pi (via usb-module) geschreven. Pygame code geschreven om toetsenbord-input af te vangen. Code geschreven voor aansturing vanaf minecraft-pi.

7 september:

mosfets zijn in het weekend binnen gekomen. framework geschreven voor aansturen robotarm via de GPIO-pins van de pi (met behulp van mosfets). Aansturen over mosfets werkte niet. Lijst van extra benodigde apparatuur opgesteld.

8 september:

besloten h-bruggen te gebruiken in plaats van mosfets, omdat dit goedkoper en makkelijker bleek te zijn. H-bruggen besteld. Nieuw aansluitingsdiagram gemaakt, kleine bugfixes toegevoegd aan code.

9 september:

code geschreven om een arduino aan te sturen met de pi, in de hoop dat die direct genoeg stroom kon leveren over zijn pins. De aansturing werkte niet, maar de verbinding tussen de pi en arduino wel.

10 september:

GPIO aansturings-framework netter herschreven, alles klaar voor aansluiten via h-bruggen.

11 september:

h-bruggen zijn binnen gekomen. Aangesloten via diagram dat 8 september gemaakt is. Dit werkte niet. Motoren direct aangesloten ipv via het aanstuurbord van de controller, motoren bewogen direct. Na een kleine bugfix van de GPIO library zijn ze aan te sturen, maar ze stoppen nog niet.

14 september:

gefocust op documentatie, begonnen aan reflectieverslag en stageverslag. Begonnen aan code voor interface in pygame. Bitmap-lettertype gemaakt voor interface.

15 september:

extra functies geschreven voor textverwerken: kleur, regenboog, vetgedrukt en onderstreept. Support geschreven voor andere lettertypes.

16 september:

Code geschreven om motoren nu ook direct uit te zetten als een toets losgelaten wordt. Echte versie van minecraft (1.12.2) draaiende gekregen op de pi.

17, 18 september:

thuis gebleven ivm migraine. Niet veel meer dan een paar regels documentatie geschreven.

21 september:

comments bij alle code gezet, code aangepast om zich aan de PEP-8 standaard te houden (op snake\_case na. De code gebruikt nog steeds camelCase omdat dit gewoon objectief beter is.) mappenstructuur van github aangepast, vooral de map voor documentatie ziet er nu netter uit.

22 september:

Bezig geweest met het opzetten van een java-project om te kunnen communiceren met de echte versie van minecraft ipv de pi versie. Cola over mijn laptop heen gekregen. Laptop open gemaakt, van binnen droog/schoon gemaakt. Laptop bewust niet meer opgestart tot de volgende dag.

23 september:

laptop start niet meer op. Werklaptop geregeld, hier alles op geinstalleerd. Werklaptop is net te traag om eclipse goed te kunnen draaien. Op aanraden van opdrachtgever linux op de laptop gezet. Zonder goed na te denken gekozen voor arch. Arch bootable gekregen.

24 september:

Verder gegaan met instellen van arch. Internet werkend gekregen via usb-tethering vanaf telefoon. Git, python etc geinstalleerd. Begonnen met het installeren en instellen van een display manager. Thuis met een multimeter gekeken waar mijn eigen laptop kapot was: grote stukken van de (on-board) videokaart. Voor reparatie zou een volledig nieuw moederbord (met vastgesoldeerde cpu en videokaart) nodig zijn, voor een laptop waarvan de garantie verlopen is. Besloten om af te zien van reparatie en naar een nieuwe laptop te zoeken.

25 september:

Thuis gewerkt; Display adapter, desktop etc. werkend gekregen. Nog steeds geen funcitionele drivers voor mijn wifi-adapter gevonden.

28 september:

besloten over te stappen op linux mint. Installatie ging een stuk makkelijker. Aan het eind van de dag zelfs functionele wifi-drivers gevonden. Pycharm geinstalleerd en gekoppeld aan github.

29 september:

passwordmanager, msTeams geinstalleerd. Eclipse geinstalleerd en geprobeerd weer een functioneel java-project te maken. Na drama met java versies, path-variabelen en ander onheil dit even vooruit geschoven. Logboek aangemaakt en bijgewerkt tot vandaag.

30 september:

probleem met pycharm opgelost. Aansturing van de arm vanaf de pi multithreaded gemaakt, zodat meerdere motoren tegelijk kunnen bewegen. Dit zou complexere manieren van bewegen mogelijk moeten maken. Ingelezen in Tkinter, zou een goede optie zijn om een programmeerinterface te maken.

1 oktober:

verder ingelezen in Tkinter. Simpele interface gemaakt waarin motoren en beweegrichtingen in geselecteerd kunnen worden en deze data vervolgens doorgestuurd wordt.

2 oktober:

verder gewerkt aan tkinterface. Automatisch scriptje geprobeerd te schrijven voor het invullen van rijen en kolommen, dit lukte niet helemaal. Ingelezen in ast (op aanraden van Jaques).

5 oktober:

theorie-examen auto gedaan, niet gehaald. Niet aan project gewerkt.

6 oktober:

scriptje voor rijen en kolommen gefixt. Geprobeerd te communiceren met ast (ophalen van informatie uit een dictionary), dit lukte niet. Geprobeerd tooltips met uitleg toe te voegen aan mijn dropdown-menus, communicatie hiertussen kreeg ik niet aan de praat. Stagebezoek gehad. Mogelijke opties na gui besproken (web-interface, bewegend (opengl) 3d-model).

7 oktober:

pi-4b binnen gekregen, geprobeerd deze aan te sluiten op de pi-top (als mogelijke programeerhardware voor de kinderen). Hierbij liepen we tegen een paar problemen aan: de 4b gebruikt usb type-c voor voeding en micro-HDMI voor beeld in plaats van micro usb (type-b) en normale hdmi. Ook is onze versie van de pi-top niet meer te leveren. De nieuwe versie van de pi-top support de pi-4b niet. Kabels besteld zodat we toch iig onze pi-top werkend zouden kunnen krijgen. Gekeken naar alternatieven (zelf te 3d-printen laptop-frame met goedkoop scherm en toetsenbord?).

8 oktober:

thuis gewerkt. Requirements en gemaakte keuzes beschreven.

9 oktober:

Requirements besproken met Projectbegeleider en aangepast. Code voor het veranderen van de taal binnen het programma geschreven.

12 oktober:

Code geschreven voor het wisselen tussen een geavanceerde en normale modus. In de geavanceerde modus kunnen bijvoorbeeld bitwise operations gebruikt worden. Deze zullen waarschijnlijk te moeilijk zijn (en te weinig echt nut hebben) om gebruikt te worden, maar kunnen nu voor de volledigheid wel gebruikt worden.

13 oktober:

Pi4b in de pi-top gestopt en aangesloten met de kabels die besteld waren. Werkte goed na het opnieuw flashen van de sd-kaart. Minecraft geinstalleerd om te testen, draaide verrassend goed, de pi haalde een vrij constante 50 fps, zelfs zonder overklokken. Ter vergelijking, de pi 3b haalde met moeite 50 frames per minuut.

14-20 oktober:

Bugs gefixt die veroorzaakt werden door de taalveranderingscode. Rechts klikken op een optie geeft nu uitleg over die optie. Hier uitleg voor geschreven. Equation opties (==, <, <= etc.) gemaakt en geïmplementeerd. Deel van een document geschreven om kinderen mee te helpen programmeren.

21-23 oktober:

Code voor de menubalk volledig herschreven. Deze gebruikt nu events in plaats van algemene functies omdat tkinter daar ruzie mee maakte. De functies veranderde de naam van het dropdown menu in de geselecteerde optie, wat voor een puinhoop zorgde in de interface. Tekstvak toegevoegd waar de geselecteerde opties in terecht komen. Programma gedemonstreerd aan opdrachtgever, feedback genoteerd. Op verzoek van opdrachtgever hulptekst ook laten verschijnen als een optie geselecteerd wordt (met linker muisknop). Simpel besturingsprogramma gemaakt om de robotarm met het toetsenbord mee te besturen. Startmenu gemaakt om tussen de twee opties te kiezen.

26 oktober:

Besturingsprogramma gekoppeld aan code voor robotarm, werkt nu. Batch bestand gemaakt om code ook vanuit windows te kunnen runnen. Niet getest, omdat ik nog geen nieuwe laptop heb. Nieuwe laptop besteld. Keybinds in het besturingsprogramma zelf geregeld, pygame code is hier nu niet meer voor nodig.

27 oktober:

Nieuwe laptop binnengekregen, deze gebruiksklaar gemaakt. Persoonlijke dingen van werklaptop afgehaald.

28 oktober:

Besturingsprogramma werkend gekregen. Batch bestand getest, werkte meteen. Kleine aanpassing gemaakt zodat er geen text naar de console gestuurd wordt. Comments toegevoegd, instructies in readme gezet.

29 oktober:

Pwm functionaliteit toegevoegd aan robotarm code, niet getest. Pwm en timer functief toegevoegd aan besturingsinterface. Fout in lambda functies om arm aan te sturen vanaf interface gefixt.

30 oktober:

Ziek

2 november:

Programmeerinterface verbeterd: kennis die is opgedaan bij het fixen van de lambda functies in de besturingsinterface toegepast om ook hier functies te gebruiken in plaats van events. Fullscreen optie toegevoegd aan programmeerinterface. Cursor wordt nu op de goede plek gezet in het tekstvak van de programmeerinterface. File-menu in programmeerinterface is nu functioneel: Code in het tekstvak kan opgeslagen worden als .py bestand, en .py bestanden kunnen het tekstvak ingeladen worden.

3 november:

Geprobeerd de code te draaien met robotarm erbij, dit werkte niet. Dummy code geschreven om robotarm mee te testen. Deze werkte ook niet. Alles opnieuw aangesloten, pitop en pi4b gesloopt door per ongeluk kortsluiting te veroorzaken. Uiteindelijk bleken de problemen te zijn: de kabels zaten op de verkeerde pins van de pi, een van de h-bruggen was kapot (afgebroken pin) en een deel van het breadboard was kapot, waardoor de helft geen stroom kreeg. Na dit allemaal te fixen en een pi3b aan te sluiten werkte de dummy code eindelijk. Hiermee kon ook de fout uit de besturingscode gehaald worden: de pwm-code bleek niet te werken.

4 november:

Nieuwe pwm-code geschreven en getest, deze werkte wel. Laatste foutjes uit de code gehaald. Toevallig kwam vandaag een basisschoolleerling langs bij het bedrijf die het programma wel wilde uitproberen. Besturing bleek niet geheel intuitief te zijn en de hulptekst stond op een plek waar deze niet opgemerkt werd. Feedback meegenomen in verbeterpunten.

5 november-10 november:

Aan 50%-presentatie gewerkt. Presentatie gegeven op 10 november, na de presentatie brainstorm gehouden over verdere ontwikkelingen aan het programma. Nog een brainstorm ingepland voor de vrijdag (13e), om de op te leveren documentatie te bespreken. Oververmoeid naar huis gegaan.

11, 12 november:

Rust genomen, niet aan project gewerkt.

13 november:

Brainstorm gehouden over documentatie, besloten een aantal ambitieuze geplande features te schrappen wegens tijdgebrek, meer te focussen op polish en documentatie. Code geschreven om te detecteren of het programma gedraaid wordt op een pi of pc, om niet elke keer code uit te moeten commenten. Overal meer comments toegevoegd.

16-20 november:

Comments aangepast na feedback. Oude ongebruikte code en testcode weggegooid. Alle code compatible gemaakt met vertaalcode. Vrijwel alle variabelen private gemaakt. Support voor leestekens als keybinds toegevoegd. Advanced mode aan besturingsinterface toegevoegd, hier pwm en timer functies in gestopt. Startmenu aangepast om progress te tracken. Advanced mode kan pas opgestart worden als de normale modus een keer gedraaid is. Begonnen aan systeem om libraries voor verschillende devices te kunnen selecteren in het programma en deze dan allemaal tegelijk te kunnen besturen via de besturingsinterface. Hier een selectiemenu voor gemaakt. Menu aangemaakt om per library de aangesloten pins van het apparaat in te vullen.

23-25 november:

Ziek

26 november:

Pinout menu gekoppeld aan robotarm code, werkt nu volledig.

27 november:

Dynamische libraries gekoppeld aan besturingsinterface. Alle geselecteerde libraries/apparaten kunnen nu tegelijk aangestuurd worden. Besturingsinterface verder aangepast zodat duidelijker is wanneer onderdelen aangestuurd kunnen worden. Oude stukken code die niet meer gebruikt worden weggehaald.

30 november:

Code aangepast om te werken met frames en toplevels in plaats van meerdere instanties van tkinter. Het programma kan nu meerdere keren tegelijk uitgevoerd worden. Pinout menu aangepast, eerst werd alle data ook in de ini van de eerste library opgeslagen, nu niet meer. Vertaaldata van de libraries losgekoppeld van de algemene vertaaldata. Batch en shell files aangepast om de data van alle libraries goed in te stellen. Standaard geschreven voor het maken van nieuwe libraries, zie lib/readme.txt. Gezorgd dat keybinds in de besturingsinterface niet meer werken als deze in de “bewerkingsmodus” staat.

1-4 december:

Libraries worden nu volledig dynamisch gebruikt door de besturingscode, geen harde links meer voor bijvoorbeeld imports. Error uit pin menu gehaald die optrad als het vakje geleegd werd. Oude imports verwijderd. Spaties uit ini files gehaald die python hier om de een of andere duistere reden automatisch inzette. Testlibrary toegevoegd die een tweede aangesloten apparaat simuleert.

7, 8 december:

RobotArm en tesTLib code aangepast en onnodige herhaling van code verwijderd (elk onderdeel had zijn eigen class, terwijl meerdere onderdelen functioneel identiek waren). Programmeerinterface werkend gekregen met dynamische libraries. Rare indexError in library-selectie menu afgevangen. Pins gelimiteerd naar 0-40 (aangezien de pi maar 40 pins heeft). Resetknop toegevoegd aan pinout menu toegevoegd die de pins terugzet naar de defaults.

9, 10 december:

Niet aan project gewerkt.

11 december:

Pwm tekstsvak omgezet naar een schuifbalk. Medestagair geholpen met zijn project.

14 december:

Coronatest gedaan in Zeeland (ik was op bezoek bij mijn ouders toen ik het bericht kreeg dat ik me moest laten testen), niet aan project kunnen werken. Test was gelukkig negatief.

15 december:

Andere soort H-brug binnen gekregen. Deze is in principe ongeschikt voor het project omdat er headers op gesoldeerd moeten worden (blijkbaar kunnen kinderen van 10 niet solderen). Voor mijn prototype kon ik de H-brug wel goed gebruiken om de H-brug die op 3 november kapot bleek te zijn te vervangen. Had geen headers ligen, dus heb er jumper wires op gesoldeerd. Code geprobeerd op pi te draaien. Shell bestand gefixt, detectiecode gefixt. Oude stukken code verwijderd. Pi via de nieuwe h-brug op een deel van de robotarm aangesloten, die kon bestuurd worden met de code.

16 december:

Robotarm volledig aangesloten op de pi (2 oude h-bruggen, 1 nieuwe). Oude h-bruggen werden erg heet en begonnen erg slecht te werken. Opgelost door ze aan te sluiten op een 3v3 pin van de pi, ipv 5v. Ookal zei de documentatie dat de h-bruggen 5v nodig hadden. Dit werkte goed, de h-bruggen raken nu niet meer oververhit. Kleine verandering gemaakt aan hulptekst, in plaats van “pins” staat er nu “GPIO pins”.

17 december:

Brainstorm over op te leveren documentatie gehad, planning gemaakt voor januari. Laptop van medestagair gefixt. Aan documentatie gewerkt.

18 december:

Aan documentatie gewerkt. Gezorgd dat library detectie code alleen kijkt naar mappen.

21 december – 1 januari:

Kerstvakantie

4-8 januari:

Bugfixes gedaan. Meeting gehad over op te leveren documentatie. Aan documentatie gewerkt. Op verzoek van timer in besturingscode ook een slider gemaakt.

11-15 januari:

Code aangepast zodat meerdere onderdelen tegelijk aangestuurd kunnen worden. Timer functie uit robotarm code gehaald (werkte oorspronkelijk met delays), zodat de tkinter loop kan blijven draaien (en dus meerdere instructies kan sturen). Test gedaan op robotarm, paar kleine bugfixes gedaan om die weer werkend te krijgen (vooral updaten van shell-bestand dat het programma start). Meeting gehad over laatste aanpassingen. Advanced programming mode uit menu gehaald, omdat die weinig toevoegde. Mappen met te gebruiken (zelf geschreven) libraries worden nu al in het startmenu aan path toegevoeds, zodat mensen die de programmeerinterface gebruiken dit niet meer hoeven te doen. Gitignore aangepast: gebruikersafhankelijke config bestanden worden nu niet meer geupload. Syntax highlighting toegevoegd aan programmeerinterface met behulp van de pygments library. Batch- en shell-bestanden aangepast om de library te installeren als hij niet gevonden wordt. Aan documentatie gewerkt.

18-22 januari:

Syntax highlighting thema veranderd (nieuwe library heeft meer contrast). Comments up to date gebracht. Documentatie.

25 januari:

Interface voor selectie van taal gemaakt. Presentatie afgemaakt.

26 januari:

Presentatie gegeven, aan stageverslag gewerkt

27-31 januari:

Aan stageverslag/reflectieverslag etc. gewerkt

1 februari:

Deadline verslag

# Conclusie

Al met al was dit een project waar veel dingen samenkwamen. Niet alleen hardware en software, maar vooral ook de communicatie tussen verschillende stukken software, tussen verschillende sessies en communicatie met in de toekomst nog te schrijven software.

## Aanbevelingen

Dit project is nog lang niet gebruiksklaar, maar dat was ook niet het doel. Er zijn veel elementen die nog toegevoegd moeten of kunnen worden:

### Mooiere interface

De huidige interface ziet er uit alsof ie rechtstreeks uit Windows 95 geplukt is. Functioneel maakt dat niet uit, maar voor kinderen is het niet bepaald aantrekkelijk.

### Tutorials/lesmateriaal

Dit project is alleen maar een hulpmiddel, het hele lespakket eromheen moet nog geschreven worden. Om te leren programmeren heb je tutorials en oefeningen nodig. Ook is er voor het aansluiten waarschijnlijk meer uitleg nodig dan alleen een Fritzing.

### (Pi-)Platform

Computers op basisscholen zijn vaak een probleem. Ze zijn traag, er mag niets op geïnstalleerd worden etc. Om dit project tot zijn recht te laten komen is het handig om de pi als volledige computer te gebruiken. Dit kan met een beeldscherm, muis en toetsenbord, maar een pi-laptop zou ideaal zijn.

### Libraries voor andere devices

Denk bijvoorbeeld aan een bestuurbaar autootje, of iets simpels als een seven-segment display of een led-array.

### Support voor input devices

Momenteel kan het programma alleen output devices aansturen. Met een uitbreiding van de besturingsinterface (en extra regels voor device libraries) zouden in theorie ook input-devices (sensoren) modulair kunnen worden aangesloten. Dit zou een hoop functionaliteit toevoegen, maar ook een hoop complexiteit.

### Webversie

Een webversie van het programma zou de noodzaak voor een platform deels wegnemen, omdat het programma dan op elk denkbaar apparaat gedraaid kan worden zonder iets te installeren. Deze versie zou dan een 2D (of 3D) animatie van een robotarm kunnen aansturen. Misschien een mooi tweedejaarsproject voor Informatica?

### Verbeteringen programmeerinterface

Hoewel de programmeerinterface werkt is ie nog erg Spartaans. Er zijn een hoop fijne functionaliteiten die nog ontbreken om er iets van te maken dat je met droge ogen een IDE kan noemen. Voorbeelden zijn werkende undo- en redo shortcuts, debugging tools (error detectie, breakpoints etc.), code completion, highlighten van alle plekken waar een dezelfde variabele/functie/class/etc. genoemd wordt enzovoorts.

# Projectdoelen

|  |  |
| --- | --- |
| Projectdoel 1: De student kan zelfstandig in een bedrijf een TI gerelateerde opdracht organiseren en uitvoeren. |  |
| Werkzaamheden | Pagina |
| * Dit hele project | - |
| Projectdoel 2: De student kan binnen een team van professionals zelfstandig en pro-actief communiceren en samenwerken. (formatief) |  |
| Werkzaamheden | Pagina |
| * Meetings met bedrijfsbegeleider | 12, 12, 14, 16 |
| Projectdoel 3: De student kan reflecteren op zichzelf als professional in het werkveld en hieruit passende leerdoelen formuleren. (formatief) |  |
| Werkzaamheden | Pagina |
| * Reflectieverslag | Bijlage ./docs/Reflectieverslag.docx |
| * Eigen leerdoelen | Bijlage ./docs/95% presentatie.pptx slide 33 |
| Projectdoel 4: De student kan bepalen welke ontwerpen relevant zijn voor de eigen opdracht en deze ontwerpen passend opstellen. |  |
| Werkzaamheden | Pagina |
| * Ontwerpen | Aansluitingen en class diagram in ./img/ |
| Projectdoel 5: De student kan de opdrachtgever op passende wijze adviseren over de resultaten en conclusies van de verrichte analyse. |  |
| Werkzaamheden | Pagina |
| * Aanbevelingen | 17 |
| Projectdoel 6: De student kan aantonen dat de gerealiseerde oplossing voldoet aan de door de opdrachtgevers gestelde eisen |  |
| Werkzaamheden | Pagina |
| * MoSCoW lijst (dit is een vrij simpele lijst met requirements, het voldoen hieraan is mijns inziens zelfevident) | Bijlage ./docs/requirements.docx |
| * Meetings met bedrijfsbegeleider | Niet woordelijk beschreven |
| Projectdoel 7: De student kan de scope en eisen van de opdracht definiëren en onderbouwde keuzes aantonen |  |
| Werkzaamheden | Pagina |
| * MoSCoW lijst | Bijlage ./docs/requirements.docx |
| * Gemaakte keuzes document | Bijlage ./docs/gemaakte keuzes.docx |
| Projectdoel 8: De student kent de bedrijfsprocessen rondom projectbeheer en kan deze indien mogelijk toepassen |  |
| Werkzaamheden | Pagina |
| * Git voor versiebeheer | <https://github.com/HermanHeringa/robotarm> |