# Documentatie Braccio Arm

## Inleiding

Dit project heeft als doel om een robotarm aan te sturen en objecten te laten verplaatsen met behulp van de Microsoft Hololens 1. Deze verplaatsingen gebeuren via commando’s die de gebruiker ingeeft in de Hololens. Voor dit project wordt momenteel nog gebruik gemaakt van een Braccio Arduino robotarm. Dit kan later nog veranderen door gebruik te maken van een robotarm met servo’s en dit aan te passen in de aansturingslibraries.

## Wat?

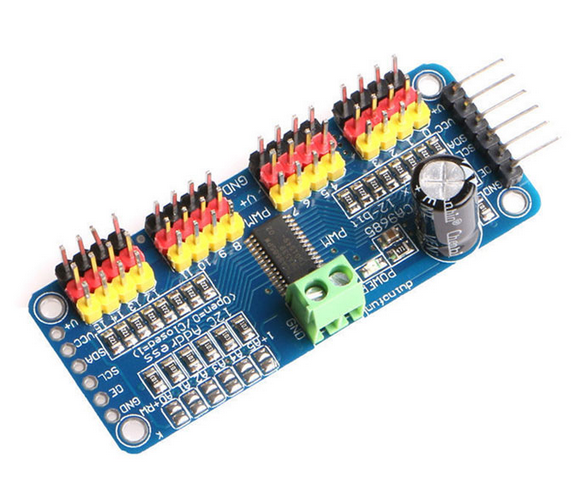
De Braccio robot Arm is een klein setje dat wordt verkocht door Arduino zelf. Dit setje bevat een aantal componenten :

* De plastic componenten waaruit de arm bestaat
* 6 servomotoren
* Een Arduino hat die de aansturing van de servomotoren regelt
* Voedingskabel voor de shield
* USB kabel voor de Arduino

Via dit setje en 2 eenvoudige Arduino libraries kan de robotarm worden ingesteld op basis van een coördinatensysteem. Hierdoor is het redelijk eenvoudig om een eerste basisversie te maken.

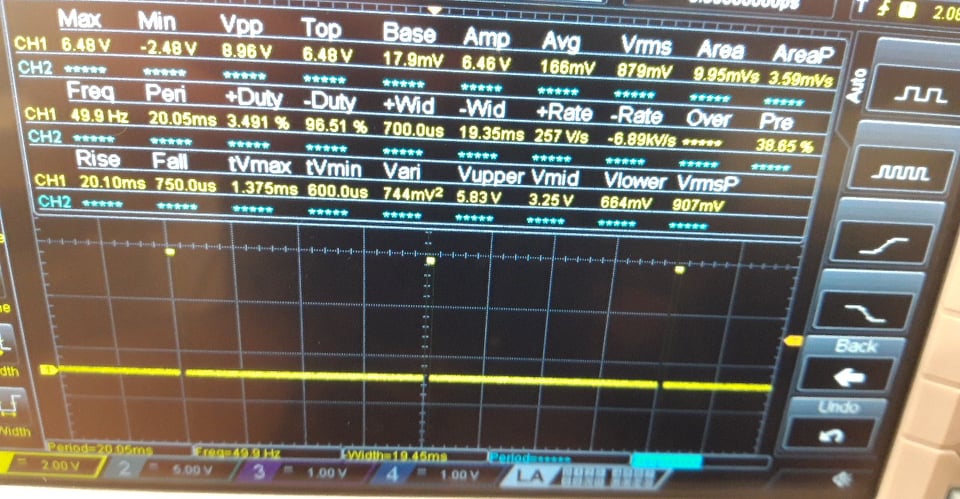
De aangeleverde libraries voor de Braccio zijn:

* Braccio (simpel te vinden via de Arduino library manager)
* InverseK (te vinden op github)

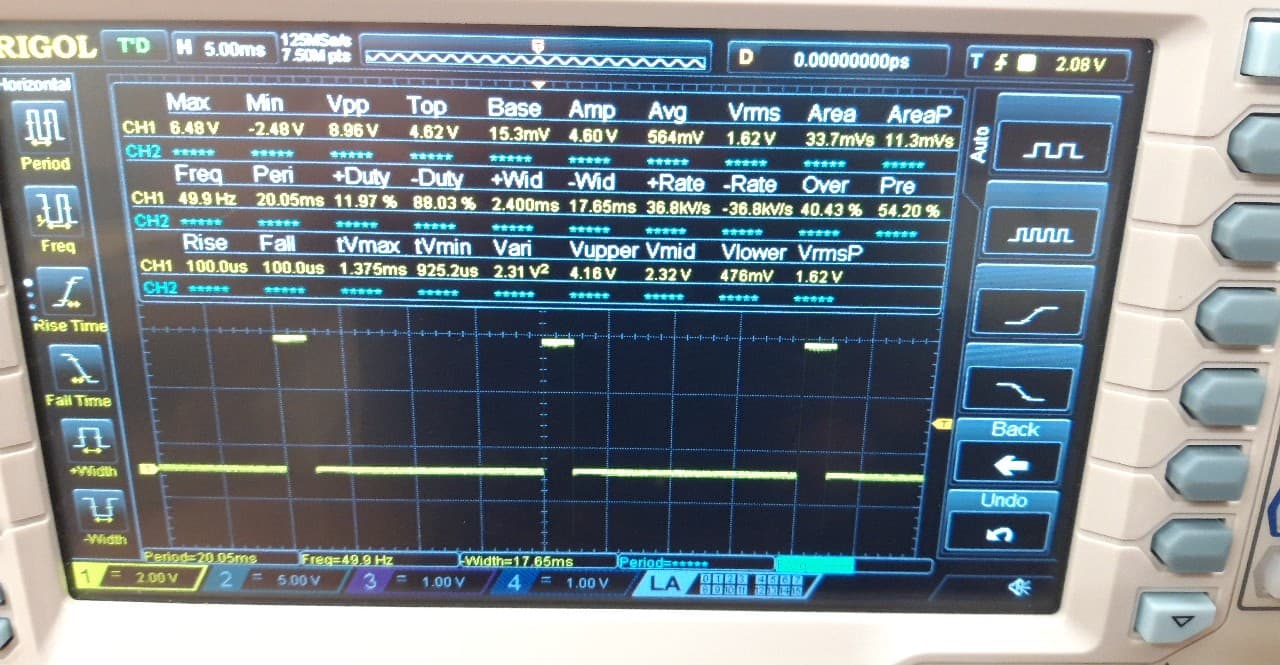
Op een later moment is besloten dat het de bedoeling was om de Arduino uit de aansturing weg te halen en te vervangen door de Raspberry Pi. De aansturing d.m.v. de Raspberry Pi is bereikt door de libraries van de Arduino, die in C zijn geschreven, om te zetten naar Python. De omgezette libraries in combinatie met een PCA9685 servo driver die elke servo aanstuurt i.p.v. de Arduino hat.

## Hoe?

### Aansturing servo’s

In dit deel zal de omgezette library code besproken worden. Deze komt op hetzelfde neer als de originele code voor de Arduino, met als verschil dat deze in een andere taal is geschreven (python i.p.v. C).

De servomotoren van de Braccio reageren slechts op enkele specifieke PWM waarden. Een te grote of te kleine PWM waarde zullen geen reactie opleveren. Er moet dus binnen een bepaalde band worden gebleven met de PWM aansturing.

Op de foto’s is duidelijk het minimale en maximale signaal zichtbaar voor de PWM (verkregen via de Arduino Hat).

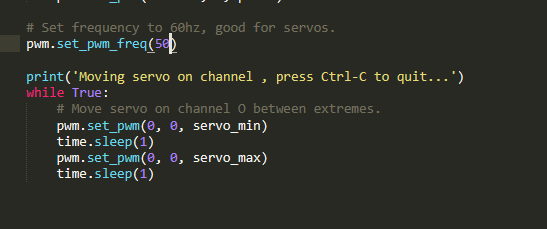
De frequentie voor deze test is ingesteld op 50Hz. Dit levert een periode op van 20ms.

Het maximum van de servomotoren kan bereikt worden door de duty cycle op 12% van de periode positief in te stellen en 88% negatief. Dit resulteert dan in een signaal dat 2,4ms hoog is en 17,6ms laag.

Om het minimum van de servomotoren te bereiken moet de duty cycle voor 3% positief en 97% negatief zijn ingesteld. Dit resulteert dan in een signaal dat 0,6ms hoog en 19,4ms laag is.

Hieruit weten we dus dat we binnen deze waarden van de periode moeten blijven, dus binnen de 0,6 en 2,4ms, om de servo te laten werken.

De PCA9685 servo driver kan zonder problemen tussen deze waarden werken.



De PCA9685 passen we aan naar een werkfrequentie van 50Hz i.p.v. de standaard van 60Hz. Alsook is gekend dat de PCA9685 een minimum heeft van 0 en een maximum van 4096. Uit deze waarden kunnen we dan berekenen hoeveel er doorgegeven moet worden om tussen de 0,6 en 2,4ms te blijven.

0 = 0ms

4096 = 20ms

409,6 = 2ms

409,6/2 = 204,8 = 1ms

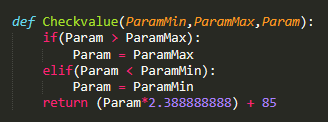
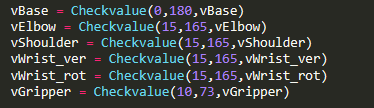
20,48 = 0,1ms

20,48\*6 = 122,88 = 0,6ms

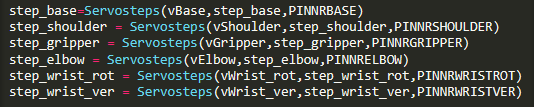
409,6 + 4\*20,48 = 491,52 = 2,4ms

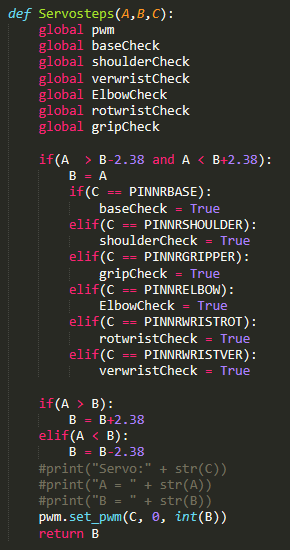
Uit de berekeningen blijkt dat de maximale waarde voor de PCA 491,52 is en de minimale waarde 122,88 is. Zolang er dus tussen deze waarden gestuurd wordt zal de PWM compatibel zijn voor de servomotoren. Op deze manier kunnen de servomotoren aangestuurd worden met de Raspberry Pi door de geconverteerde library.

Op de afbeelding hierboven valt te zien hoe de servo aansturing precies gedaan wordt door de code. Alle variabelen die worden doorgegeven in de methode bestaan uit een hoek tussen 0 en 180 graden.



Eerst wordt nagekeken of de waarden die worden doorgestuurd naar de servomotoren wel degelijk tussen de 0 en 180 graden zijn. Nadien worden de waarden omgezet naar waarden die de PCA9685 aankan.







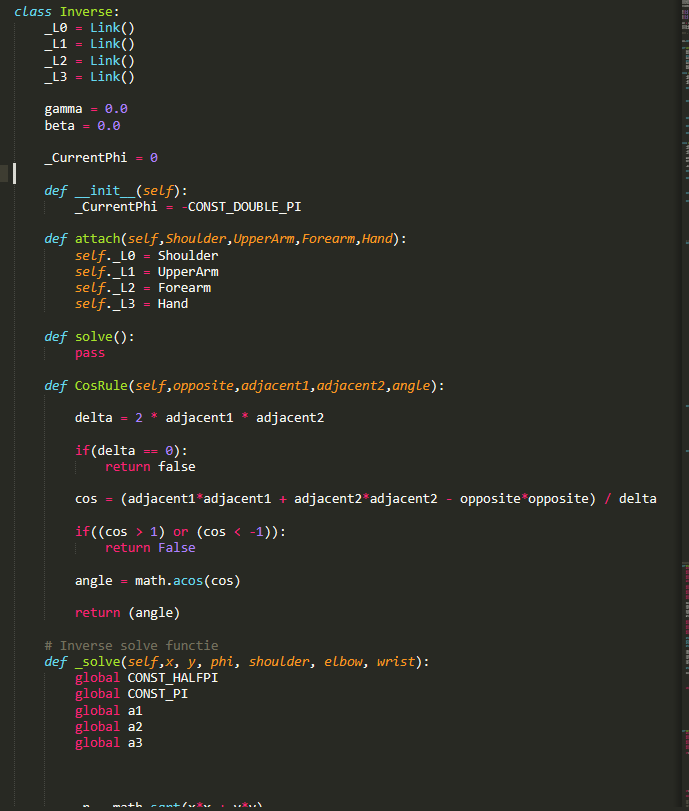
Hierna worden stap voor stap de waarden van de servomotoren aangepast. De PCA9685 kijkt eerst na of de servomotor op dit moment een hogere of lagere waarde heeft dan de doorgegeven waarde. Afhankelijk van het resultaat van deze controle zal er bepaalt worden of er opgeteld of afgetrokken moet worden. Mocht de originele waarde en de doorgegeven waarde overeen komen, dan wordt er een boolean op true gezet. Hierdoor weet de PCA9685 dat de servomotor al correct staat en dus geen aanpassing nodig heeft. Wanneer er geen veranderingen aan de servomotoren nodig zijn kunnen er opnieuw waarden binnengehaald worden en begint de code opnieuw.

### Berekening van de coordinaten

Nu dat de servomotoren aangestuurd kunnen worden kan er begonnen worden aan het berekenen van de coördinaten. Om dit te doen zetten we de Arduino library (geschreven in C) om naar Python voor de Raspberry Pi. Hierna zijn er nog kleine tweaks nodig om de library op de Raspberry Pi beter te laten werken.

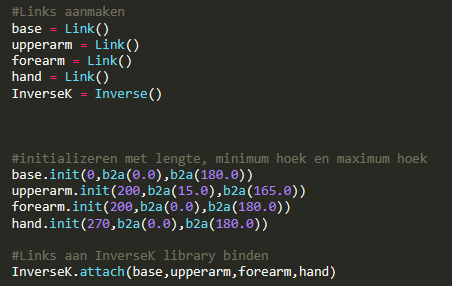
Deze library maakt gebruik van de volgende 2 klassen:

* De **Link class**: Deze klasse representeert de verbinding tussen de servomotoren. Er wordt o.a. bijgehouden wat de hoek momenteel is en hoe lang de verbinding tussen de motoren is. Dit alles is belangrijke info voor de berekeningen.



* De **Inverse class**: Deze klasse verzamelt alle links en bevat ook de wiskundige logica. Alle berekeningen worden uitgevoerd in deze klasse. In het voorbeeld in de afbeelding is slechts een klein deel hiervan te zien.

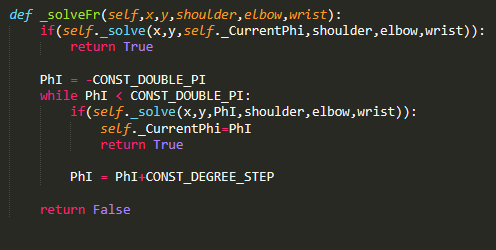
We instantiëren enkele variabelen van deze klassen in het begin van de library. Omdat de robotarm uit 4 verschillende delen bestaat moeten er 4 variabelen van de Link class worden aangemaakt. Deze 4 variabelen worden hierna gekoppeld aan de InverseK library.

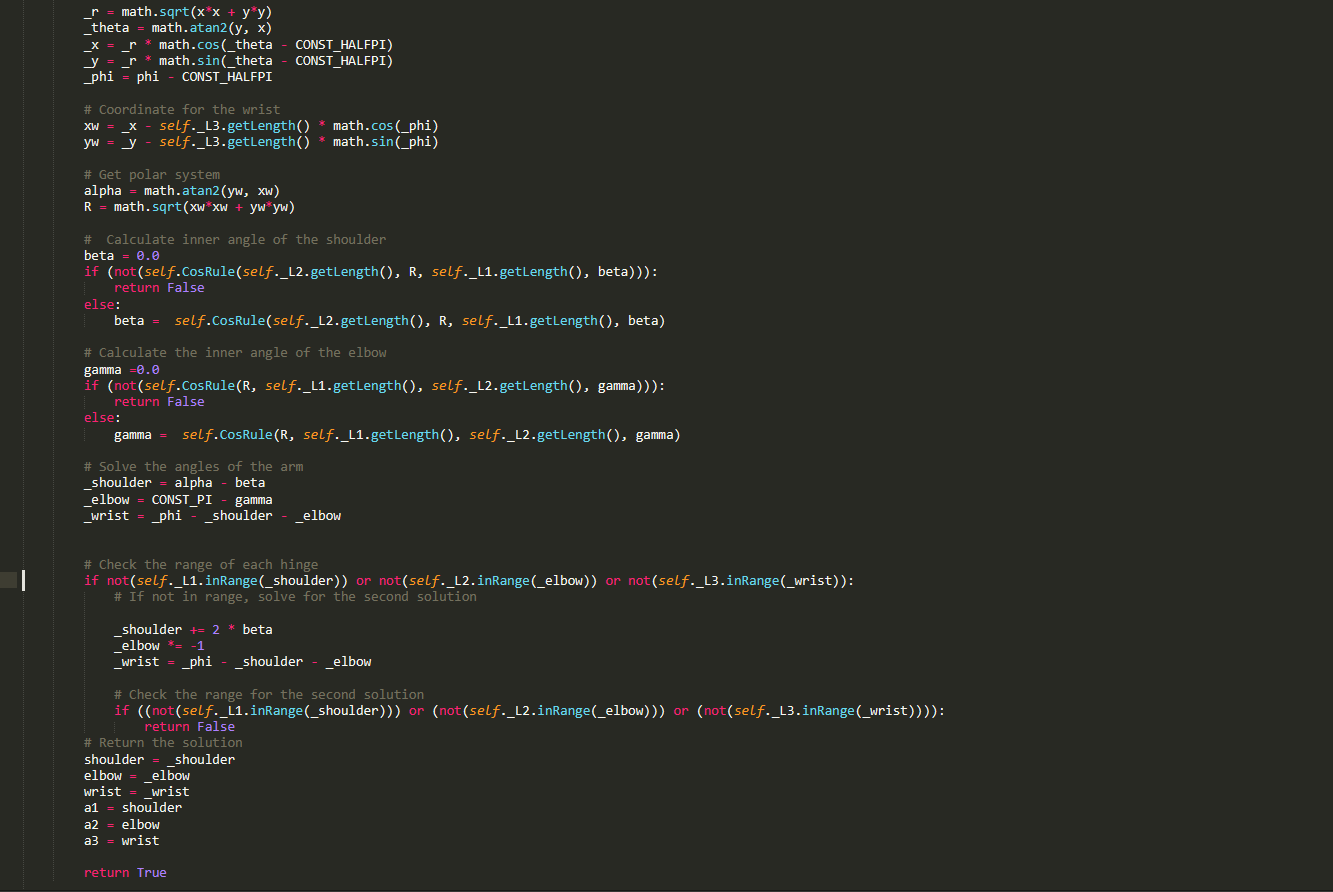


Na het initialiseren wordt er gekeken naar de haalbaarheid van de coördinaten d.m.v. de volgende lijn code:

Deze code verwijst naar het begin van de InverseK library en zal een boolean teruggeven die afhankelijk is van het al dan niet mogelijk zijn van de ingegeven waarden. Indien de boolean true is (en de waarden dus mogelijk zijn) zullen de waarden van A0 t.e.m. A3 omgezet naar de ingegeven waarden.

In de library zal eerst nagekeken worden naar welke positie de base van de robotarm moet worden gedraaid om in het gewenste kwadrant te eindigen. Nadien wordt pas verder gedaan met de controle van de coördinaten.



Via wiskundige berekeningen (zie onderstaande foto) wordt er gecontroleerd of de gewilde coördinaten te bereiken zijn via de gegevens die zijn binnengekomen. In de code zal er d.m.v. de cosinusregel bepaald worden of 2 hoeken samen genoeg afstand kunnen overbruggen om de coördinaten de bereiken. Dit zal herhaald blijven worden tot de berekeningen buiten de maximale servomotor waarden terechtkomen of er een mogelijke hoekcombinatie is gevonden. Indien er een vaste hoek wordt meegegeven zal dit maar 1 keer worden uitgevoerd en weergeven of de positie bereikbaar is via de meegegeven hoek.

## Waarom?

De Arduino uit de setup halen zorgt ervoor dat er 1 verbinding minder is waar er uiteindelijk rekening mee moet worden gehouden. Hierdoor zal de opstelling makkelijker worden, zal de kost lager uitvallen en werkt alles efficiënter. Deze voordelen zorgen ervoor dat het uiteindelijke project beter en efficiënter zal werken.