Štirinogi robot

Raziskovalna naloga

Avtor: Jaka Gselman

Mentor: Darko Visočnik

Razred: 4.at

Šola: Srednja elektro-računalniška šola Maribor

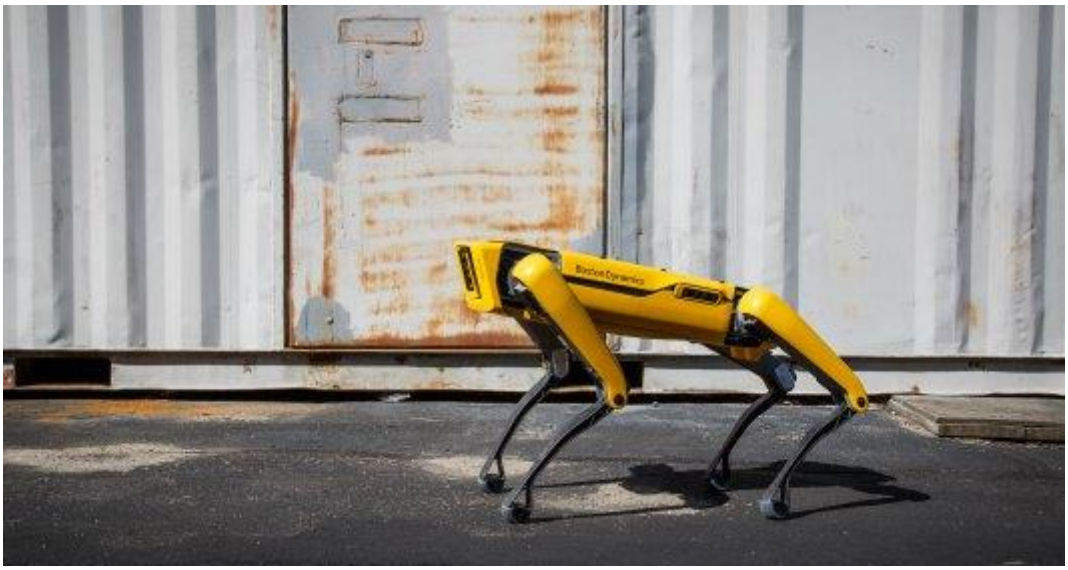
Raziskovalno področje: elektrotehnika, elektronika in robotika

1 Vsebina

2 Kazalo slik

3 Uvod

Živimo v 21. stoletju, kjer pot inovacij temelji na avtomatiki ter robotiki. Napredki v robotiki so vedno večji in kmalu bodo roboti lahko opravljali večino del. Za raziskovalno nalogo sem naredil štirinogega robota. Navdih za robota sem dobil od podjetja Boston Dynamics, ki že leta razvija svojega robota Spot. Spot je zelo gibčen in sposobnost, da premaguje ovire, ki jih roboti s kolesi ne morejo. Prav tako lahko prenaša lažji tovor. Je prilagodljiv in na njega lahko namestimo razne senzorje, odvisne od naših potreb in namena uporabe robota. Ampak Spot ni cenovno dostopen navadnim potrošnikom, zaradi tega sem se odločil, da ga bom naredil sam. Prednost štirinogega robota pa je premagovanje različnih površin in ovir, ki jih navadni roboti ne zmorejo. Dolge noge robotu omogočajo prestopanje ovir, hojo po stopnicah ter večje ravnotežje. Lahko se uporablja za raziskovanje okolja, kjer je večja raven nevarnosti. Roboti lahko opravljajo dela, ki jih ljudje ne morejo ali pa so za ljudi prenevarni.



Slika 1 Robot Spot od Boston Dynamics (vir: Boston Dynamic)

4 Povzetek

Naredil sem štirinogega robota, ki lahko hodi. V vsaki nogi ima tri servomotorje, ki omogočajo natančno premikanje noge v tri smeri. Celoten sistem bo nadzirala razvojna ploščica Teensy 3.6. Za krmiljenje vseh 12 servomotorjev sem potreboval vmesnik, ki na Teensy-u poveča število izhodov, sposobnih za krmiljenje servomotorjev. V telesu robota je nameščen žiroskop in merilnik pospeška, ki spremlja nagib telesa ter informacijo posreduje Teensy-u. Teensy nato preračuna položaj noge, ki je potreben, da je telo robota v ravnovesni legi. Da je robot lahko preračunal položaj nog sem moral sprogramirati kinematični model robota. To je sklop enačb, ki na podlagi spremenljivk, kot so naklon, višina telesa in položaj stopal, izračuna položaj vsakega servomotorja. Robota lahko nadziram na daljavo z radijskim daljincem, ki narejen s pomočjo Arduina ter NRF24L01 radijskega sprejemnika ter oddajnika. Pri delu sem naredil več napak, ki jih je bilo potrebno odpraviti, zato sem naredil več različic robota.

5 Cilji

Moj cilj je bil narediti štirinogega robota, ki ima sposobnost hoje. Robot bo velik približno 30 cm, da se lahko premika skozi manjše odprtine. Želim, da bo lahko sam lovil ravnotežje s pomočjo žiroskopa. Robota bom nadziral s pomočjo, radijskega daljinca, ki ga bom naredil sam. Če mi bo projekt uspel bo robot dobra osnova za nadaljnjo razvijanje programa za gibanje ter premagovanje ovir.

6 Zahvala

Zahvaljujem se mentorju za vso pomoč pri delu raziskovalne naloge. Zahvaliti se moram tudi Srednji elektro-računalniški šoli Maribor za vso finančno podporo. Zahvaljujem se tudi vsem, ki so k raziskovalni nalogi kakorkoli pomagali.

7 Vsebinski del

7.1 Teensy 3.6

Teensy 3.6 je razvojna ploščica, ki je združljiva s Arduino programsko opremo. Za razliko od Arduina pa je dosti bolj zmogljiva, saj ima več vhodov in izhodov, hitrejši procesor ter več spomina. Uporablja 32-bitni 180 MHz ARM Cortex-M4 procesor. Ima 40 digitalnih vhodov in izhodov (22 od njih lahko uporablja PWM). Ima tudi 25 analognih vhodov in 2 izhoda. Programira se lahko z računalnikom preko USB priključka. Progam sem pisal v C++ jeziku v Arduino IDE.

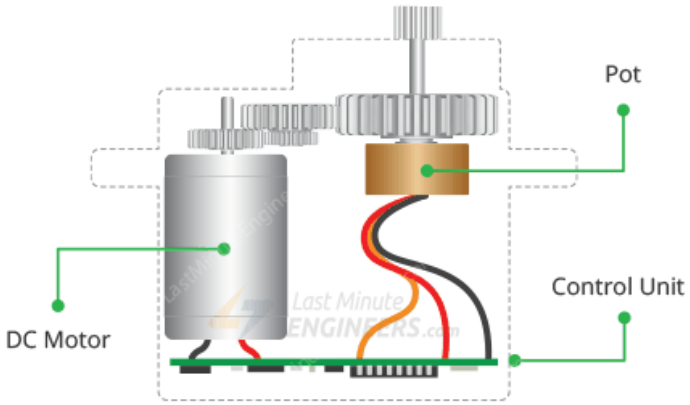
Slika 2 Teency 3.6 Razvojna ploščica (vir: Amazon)

7.2 Servomotor

Odločil sem se, da bom za premikanje nog robota uporabljal servomotorje. Servomotorji so lahko različnih velikosti. Manjši se veliko uporabljajo za hobije, večji pa v industriji. Večinoma je njihova os omejena samo na 180° rotacije. Servomotorji se zelo razlikujejo po obliki in izgledu, odvisno od njihove uporabe. Zelo so kompaktni in lahki, ampak kljub njihovi majhnosti imajo velik navor.

Slika 3 Servomotor (vir: Sumozade Robotics)

Servomotorji imajo tri glavne dele: sklop zobnikov, DC motorček in krmilno vezje. Sklop zobnikov zmanjša število obratov DC motorčka. To je potrebno, da dosežemo večjo natančnost ter navor. Krmilno vezje določi čas, kako dolgo in v katero smer se mora vrteti DC motor, da doseže pravilni položaj osi servomotorja. Os je neposredno priključena na potenciometer, ki krmilnemu vezju sporoča trenutni položaj osi. Za krmiljenje servomotorja uporabljamo tri žice; dve za napajanje in eno za signal. Na ta priključek priključimo PWM (Pulzno širinska modulacija) signal, ki določa želeno pozicijo osi. Krmilno vezje nato požene DC motor in ga ne izklopi, dokler ni trenutna pozicija osi enaka zaželeni.



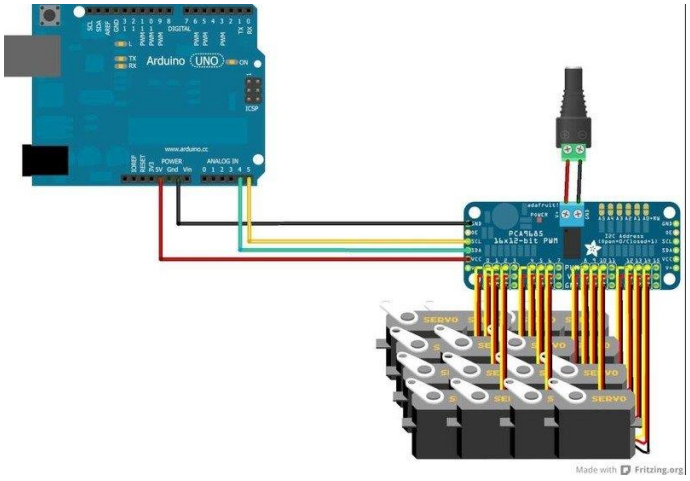
Slika 4 Sestava servomotorja (vir: Last Minute Engineers)

7.3 PWM krmilnik

Da sem zaščitil Teensy pred velikimi tokovi, ki jih potrebujejo servomotorji, sem uporabil razširitveno ploščico. Ploščica ima 16 12-bitnih PWM izhodov. Ploščica komunicira s Teensy-om preko I2C komunikacijskega protokola. Na izhode sem povezal vseh 12 servomotorjev, na vhod pa SDA in SCL priključek na Teency-u. Ploščica potrebuje svoje napajanje 6V, ki ga za enkrat dovaja napajalnik, kasneje pa ga bo nadomestil akumulator.

Slika 5 Adafruit PWM krmilnik (vir: Adafruit Learning System)

Krmilnik lahko podpira vhodno napetost 5V ali 3,3V, vendar moraš to določiti tako, da povežeš skupaj dva kontakta na tiskanem vezju. Ker Teency deluje na 3.3V sem skupaj povezal prva dva kontakta.

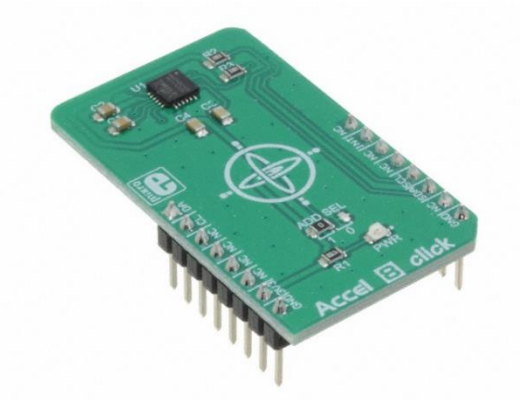


Slika 6 Shema vezja s PWM krmilnikom (vir: Adafruit Learning System)

VCC priključek sem povezal na 3.3V priključek na Teensy-u, prav tako sem skupaj povezal GND priključka na krmilniku in Teensy-u. Teensy ima 3 pare SCL in SDA priključkov. Krmilnik sem priključil na prvi par, poimenovan SCL0 in SDA0, ki se nahaja na priključkih 19 in 18 na Teensy-u.

7.4 Žiroskop in merilnik pospeška

Za robota sem potreboval žiroskop, zato sem uporabil MPU-6050. To je cenovno zelo ugoden žiroskop, ki vsebuje tudi merilnik pospeška. Zelo pogosto se uporablja v preprostih projektih. Žiroskop je čip, ki zaznava spremembo svojega naklona v treh oseh. Pritrdil sem ga na telo robota, da bo lahko meril naklon telesa.



Slika 7 MPU 6050 žiroskop (vir: Digikey)

Za komunikacijo z Teensy-om uporablja I2C komunikacijski protokol. To pomeni, da za prenos informacij uporablja samo dve žici. Žici sta povezani na SDA in SCL priključka na Teensy-u in žiroskopu. Ker ima Teensy več parov SDA in SCL priključkov in, ker sem na prvi par že priključil krmilnik za servomotorje, sem žiroskop priključil na SDA1 in SCL1 priključka na Teensy-u. Prav tako žiroskop potrebuje napajanje, in sicer 3.3V, ki jih dobi iz Teensy.

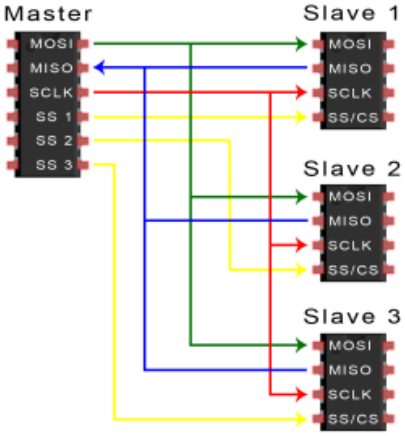
7.5 NRF24L01 sprejemnik in oddajnik

Robota želim nadzirati na daljavo, zato potrebujem način prenašanja informacije brezžično. Po raziskovanju o tej temi, sem našel NRF24L01 modul za brezžično komunikacijo. V enem čipu ima sprejemnik in oddajnik, to pomeni, da bom lahko imel dvosmerno komunikacijo med daljincem in robotom. Modul uporablja 2.4 GHz radijski

signal za prenos informacije in ima doseg več metrov. Modul ima 125 različnih kanalov, kar pomeni, lahko uporabljamo 125 modulov na enem mestu. Vsak kanal ima lahko 6 naslovov. To pomeni, da lahko en modul komunicira s šestimi moduli hkrati.

Slika 8 nrf24L01 oddajnik in sprejemnik (vir:Amazon)

Za komunikacijo s Teensy-om uporablja SPI (Serial Peripheral Interface) komunikacijski protokol. Prednost SPI komunikacije je, da se lahko prenaša informacija brez prekinitev (interupt). Naprave so v master-slave odnosu, kar pomeni, da je ena naprava "glavna" in nadzira suženjsko napravo. SPI uporablja štiri priključke: MOSI- za pošiljanje informacije, MISO- za prejemanje informacij, SCLK- za signal ure in SS/CK- za izbor sužnja, ki mu želimo pošiljati informacije.



Slika 9 SPI master-suženj povezava (vir: Circuit Basics)

Daljinec bo sestavljen iz treh glavnih delov: joystick, Arduino in NRF24L01 modul. Z joystickom lahko nadziram različne funkcije robota, npr. hoja, naklon telesa, dvig telesa. Med temi funkcijami lahko izbiram s pomočjo gumbov na daljincu. Arduino bere vhode gumbov in joystick-a ter, nato pošlje to informacijo Teency-u preko nrf24L01 modula.

8 Potek dela

Delo sem pričel z raziskovanjem po spletu, poiskal sem čim več informacij o podobnih projektih ter o njihovem delovanju. Z raziskovanjem sem ugotovil, da velika večina podobnih robotov uporablja servomotorje. Razlog za to je njihova natančnost in dosti manjša teža v primerjavi s koračnimi motorji. Nato sem iskal načine, kako bom lahko 12 servomotorjev nadziral z enim krmilnikom. Med drugimi rešitvami sem ugotovil, da bi lahko razširitveno ploščico naredil sam, ampak, da bo bolj zanesljiva sem se odločil, da bom uporabil kupljeno. Še vedno pa nisem vedel, s katerim mikrokrmilnikom bom lahko nadziral robota. Najprej sem se odločil za Arduino Mega razvojno ploščico, ki sem jo dobro poznal že iz prejšnjih projektov. Vedel sem, da želim krmilnik, ki ga lahko programiram v C++ programskem jeziku. Na srečo sem našel Teensy 3.6, ki je zmogljivejša kot katera koli Arduino ploščica in za programiranje uporablja isto programsko okolje kot Arduino.

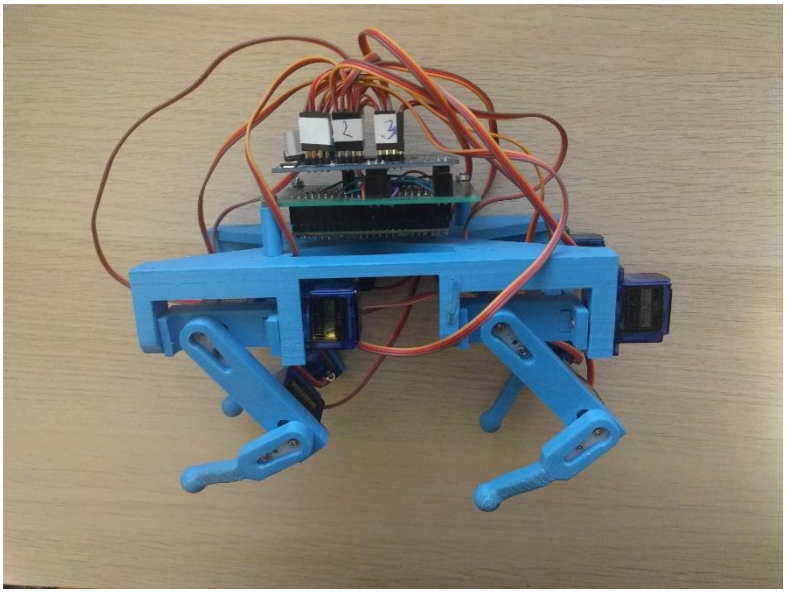
9 3D oblikovanje

Ko sem vedel, katere elemente bom uporabil, sem začel z načrtovanjem oblike robota. Navdih za obliko sem dobil od Boston Dynamic, ki že več let uporabljajo takšno obliko robota, ki ima kolena robota obrnjena nazaj. Za oblikovanje robota sem uporabil program Fusion 360. Tam sem zmodeliral vse dele robota. Naredil sem par sprememb, kar se tiče oblike, ki so mi olajšale delo v naprej. Najpomembnejša je ta, da sem stopalo noge postavil točno pod zgornji sklep robota. To mi je zelo olajšalo računaje pozicije stopala. Prav tako sem poskrbel, da je razdalja med drugim in tretjim sklepom enaka razdalji med tretjim sklepom in stopalom.

Slika 10 3D model robota (vir: Avtor naloge)

10 Verzija 1

Prva in tudi manjša verzija robota uporablja SG90 servomotorje. Za te motorje sem se odločil, saj sem pričakoval, da bodo dovolj močni, da bodo lahko podpirali težo robota, kar v teoriji so, ampak v praksi pa so se pojavile napake. Robot se je lahko dvignil in spustil, ampak samo, če so bile na tleh vse štiri noge, za hojo pa je potrebno, da se lahko obdrži na samo dveh nogah. Čeprav se robot ni zmogel obdržati na nogah, sem, ga lahko uporabil za testiranje kode, in sicer tako, da sem ga podprl. Tako sem lahko opazoval premike nog. Tukaj sem opazil drugo težavo. Zaradi teže noge ter premajhne moči servomotorja se je ob premiku noga zelo tresla.



Slika 11 Prva verzija robota (vir: avtor naloge)

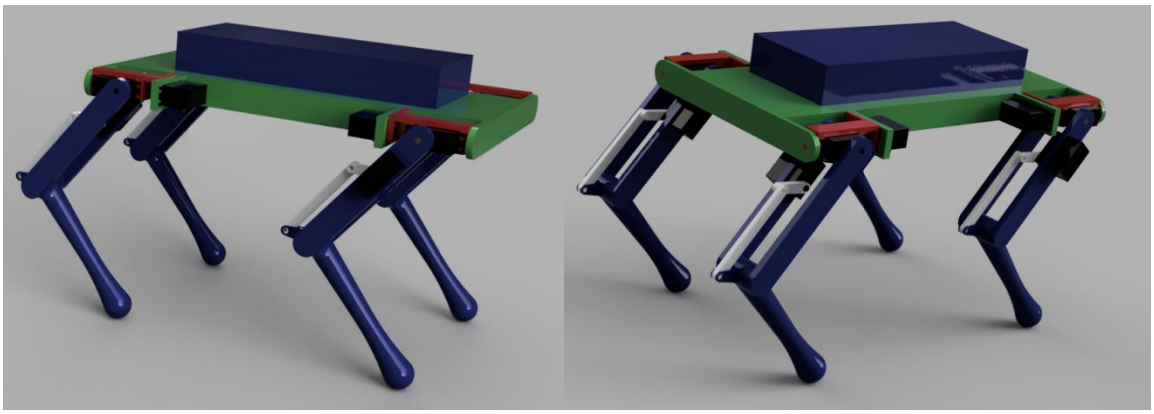
10.1 Ugotovitve

Ugotovil sem, da imajo servomotorji SG90 premalo moči, da bi zanesljivo podpirali težo robota. Ta problem lahko rešim na dva načina, lahko uporabim močnejše in večje servomotorje, ali pa zmanjšam težo celotnega robota. Težo robota, bom zelo težko zmanjšal, saj večino mase predstavljajo Teency, krmilnik, motorčki in ostala elektronika, kar so vsi nujni elementi in jih ne morem odstraniti. Zato sem se odločil, da bom uporabil močnejše servomotorje. Prav tako sem opazil, da je na zadnjih nogah veliko več teže kot na prednjih, kar je povzročilo, da je robot zelo nestabilen in se je prevračal nazaj. To bom popravil tako, da bom spremenil pozicijo zadnjih motorčkov.

11 Verzija 2

V drugi verziji sem se odločil, da bom uporabil močnejše in večje servomotorje MG996R. Ti so dvakrat večji kot prejšni SG90, ampak so deset krat močnejši. Ker so motorčki večji, sem povečal tudi celotno velikost robota, osnovna oblika pa je ostala ista. Prva verzija je imela več teže na zadnjih noga, zato sem v drugi verziji zadnje servomotorje prestavil v notranjost robota. Prav tako sem spremenil lokacijo motorčka v kolenu, ki je bil prej neposredno na kolenu. Prestavil sem ga čim višje po nogi, tako bo moral zgornji servomotor opraviti manj dela za isti gib. Koleno sedaj premikam posredno preko vzvoda.

Slika 12 Servomotor MG996R (vir: Hobbyking)



Slika 13 3D model 2 verzije robota (vir: Avtor naloge)

12 Programiranje

Vso programiranje sem opravil v programskem okolju Arduino IDE, v C++ programskem jeziku. Programiranje sem opravil po delih, najprej sem preskusil vsak sistem posebej, nato pa sem vse skupaj združil v eno samostojno kodo.

12.1 Kalibracija servomotorjev

Najprej sem moral kalibrirat servomotorje, saj iste pulzne vrednosti na različnih servomotorjih podajo različen kot na servomotorju. Te razlike se kažejo tudi na istih modelih servomotorjev istih znamk, zato je to nujen proces, pri kakršnem koli projektu s servomotorji. Pri kalibraciji določaš, katera pulzna vrednost je potrebna za maksimalen kot (180°) in katera pulzna vrednost za minimalen kot (0°). To opravimo s postopkom poizkušanja, najprej na servomotor nastavimo neko pulzno vrednost in jo po malo večamo, dokler servomotor ne pride do maksimalnega kota. Postopek nato ponovimo tako, da vrednost manjšamo, dokler servomotor ne doseže minimalnega kota. To sem ponovil za vse servomotorje, tako sem za vsakega dobil maksimalno in minimalno pulzno vrednost.

```
/* SERVO   0     180
 * 0      101    558
 * 1      101    558
 * 2      101    558
 * 3      100    550
 * 4      565    115
 * 5      565    107
 * 6      118    570
 * 7      112    567
 * 8      95     553
 * 9      116    570
 * 10     95     553
 * 11     125    575
```

Slika 14 Pulzne vrednosti po kalibraciji (vir: Avtor naloge)

V krmilnik za servomotorje moramo poslati pulzno vrednost, in da ne rabim računati pulzne vrednosti za vsaki kot sem uporabil funkcijo map(), ter v njo vstavil vrednosti, ki sem jih pridobil iz kalibracije. Nekatere maksimalne in minimalne vrednosti sem zamenjal, saj so servomotorji v robotu obrnjeni za 180°.

```
pulse0 = map(angle0a,0,180, 115, 565);
pulse1 = map(angle1a,0,180, 101, 558);
pulse2 = map(angle2a,0,180, 101, 558);
pulse3 = map(angle3a,0,180, 100, 550);
pulse4 = map(angle4a,0,180, 101, 558);
pulse5 = map(angle5a,0,180, 107, 565);
pulse6 = map(angle6a,0,180, 570, 118);
pulse7 = map(angle7a,0,180, 567, 112);
pulse8 = map(angle8a,0,180, 553, 95);
pulse9 = map(angle9a,0,180, 570, 116);
pulse10 = map(angle10a,0,180, 95, 553);
pulse11 = map(angle11a,0,180, 125, 575);
```

Slika 15 Uporaba map() funkcije s kalibriranimi vrednostmi (vir: Avtor naloge)

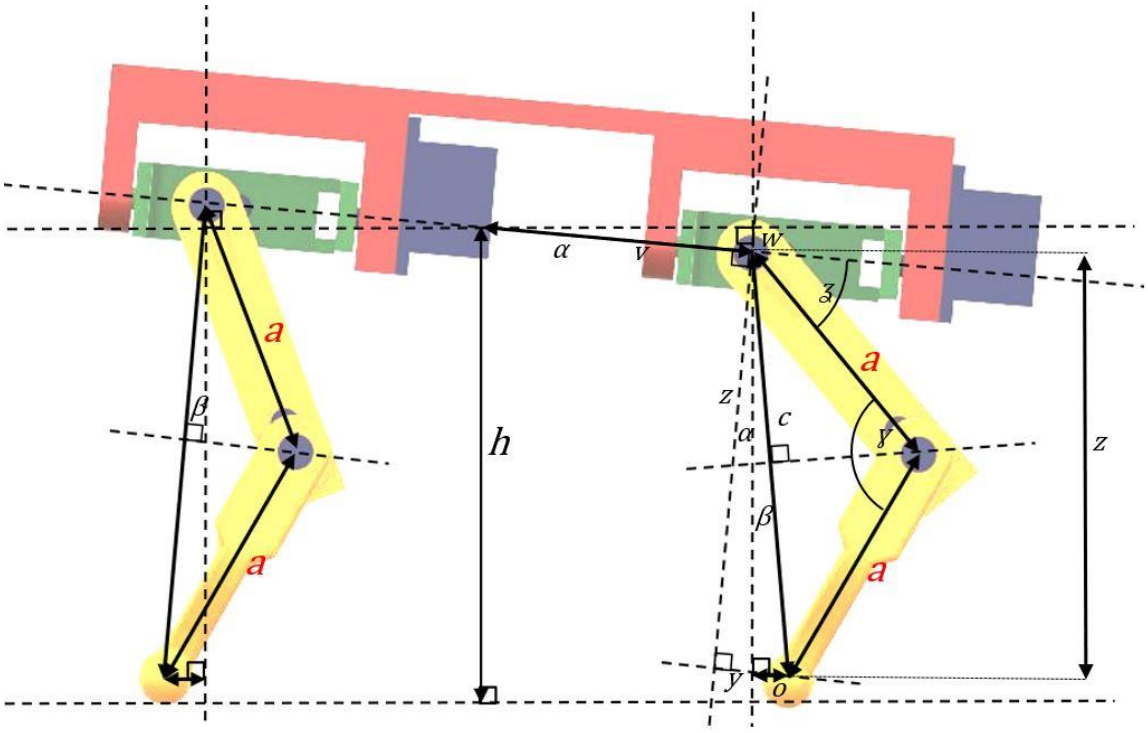
Na koncu moram pulzno vrednost poslati dalje v krmilnik servomotorjev. To naredim s ukazom pmw.setPWM(), kjer je prva spremenljivka številka servomotorja in zadnja pulzna vrednost.

```
pwm.setPWM(0, 0,pulse0);
pwm.setPWM(1, 0,pulse1);
pwm.setPWM(2, 0,pulse2);
pwm.setPWM(3, 0,pulse3);
pwm.setPWM(4, 0,pulse4);
pwm.setPWM(5, 0,pulse5);
pwm.setPWM(6, 0,pulse6);
pwm.setPWM(7, 0,pulse7);
pwm.setPWM(8, 0,pulse8);
pwm.setPWM(9, 0,pulse9);
pwm.setPWM(10, 0,pulse10);
pwm.setPWM(11, 0,pulse11);
```

Slika 16 Pošiljanje pulzne vrednosti v krmilnik (vir: Avtor naloge)

12.2 Kinematični model

Za računaje kotov servomotorjev sem naredil kinematični model. Napisal sem program, kjer lahko vstavim željen položaj telesa, višino telesa od tal in položaj stopal, ter mi program izračuna vrednosti kotov v vsakem sklepu. Te vrednosti, nato pretvorim v pulzne vrednost ter jih pošljem na krmilnik.



Slika 17 Shema kinematičnega modela (vir: Avtor naloge)

$\mathrm{w} = \sin\alpha \times \mathrm{v}$ $o = v - cos\,\alpha \times v$ $z = h - w$ $tan\,\beta = \frac{o}{z}$ $y = tan\,\alpha + \beta \times z$

$$c = \sqrt{y^2 + z^2} \qquad 2sin\,\gamma = \frac{\frac{c}{2}}{a} \qquad \zeta = \frac{\gamma}{2} - \alpha - \beta$$

Tukaj imam prikazan samo del kinematičnega modela, ki je potreben za nagib telesa robota naprej in nazaj tako, da je središče telesa vedno na isti višini h. Iz tega modela želim izvedeti vrednosti ɣ in ʒ za naklon α. Edina vrednost, ki jo ročno spreminjam je kot α. Program nato uporabi zgornje enačbe, da izračuna kota servomotorjev ɣ in ʒ.

12.3 Programiranje NRF24L01 radijskega oddajnika

Najprej za delovanje potrebujemo par knjižnic, ki nam olajšajo delo. Posebej pomembni sta SPI.h knjižnica, ki je potrebna za SPI komunikacijski protokol, ter RF24.h, ki se uporablja s tem modulom. Najprej moramo modul incializirati, kako pa ga inicializramo je odvisno od njegove vloge.

Če je modul oddajnik mu moramo ukazati, da pripravi oddajnik ter, da neha sprejemati signale.

```
radio.openWritingPipe(address);
radio.stopListening();
```

Če pa je sprejemnik, pa mu ukažemo, da se pripravi za sprejem ter začne sprejemati signale.

```
radio.openReadingPipe(0, address);
radio.startListening();
```

Potem sem z ukazom radio.write(), poslal vrednosti iz daljinca v Teensy.

13 Hoja

Odločil sem se, da bo robot hodil tako, da bo hkrati premikal sprednjo levo in zadnjo desno, v naslednjem koraku pa s sprednjo desno in zadnjo levo ter to ponovi. V koraku nogo dvigne in jo prestavi približno 5 cm naprej. Nato jo ponovno spusti na tla, ter pomakne svojo težo naprej. Ker je sedaj teža na sprednjih dveh nogah, lahko dvigne nogi, ki sta zadaj, ter jih pomakne naprej. Postopek nato ponovi. Ker med hojo dosti časa preživi na samo dveh nogah je pomembno, da je teža v telesu robota enakomerno razporejena. Trenutno nimam sprogramirane nikakršne aktivne stabilizacije robota med hojo.

14 Zaključek

Za celotno nalogo sem porabil približno 250 ur, to vključuje čas izdelave delov. V projekt je bilo vloženo ogromno truda in dela. Z raziskovalno nalogo sem pridobil ogromno novega znanja, sploh na področju robotike in kinematike. Veliko novega znanja sem pridobil tudi na področju programiranja, ki je bilo zelo pomemben del naloge, in je zahtevalo največ mojega časa. S svojim delom sem zadovoljen, saj sem presegel svoja pričakovanja. Čeprav sem naredil veliko napak, sem te sprejel in jih uporabil kot osnovo za nadgradnjo. Z delom pa še nisem končal, robota bom še naprej razvijal ter ga nadgrajeval. Upam, da bom s svojo raziskovalno nalogo navdihnil druge, da bodo izdelali svojega štirinogega robota.

15 Družbena odgovornost

Robot je odlična osnova, za nadaljnjo razvijanje sistema za hojo, različnih senzorjev in dodatkov, kot so na primer robotska roka ali pa kamera. Zaradi svoje majhnosti in sposobnosti premagovanja različnih terenov je robot zelo primeren za iskanje ponesrečencev med ruševinami. Robot je tudi primerno učilo za izobraževanje robotike, kjer bi lahko učenci pisali svoje programe za premikanje ter s tem večali svoje znanje programiranja.

16 Viri in literatura

(Vsi viri so bili nazadnje uporabljeni 5.2.2019)

- https://www.pjrc.com/store/teensy36.html
- https://www.bostondynamics.com/spot
- https://en.wikipedia.org/wiki/Kinematics
- https://howtomechatronics.com/tutorials/arduino/arduino-wireless-communication-nrf24l01-tutorial/
- https://www.sparkfun.com/datasheets/Components/nRF24L01_prelim_prod_spec_1_2.pdf
- https://en.wikipedia.org/wiki/Servomotor
- https://lastminuteengineers.com/servo-motor-arduino-tutorial/
- https://www.mikroe.com/accel-8-click
- http://www.circuitbasics.com/basics-of-the-spi-communication-protocol/
- http://www.circuitbasics.com/basics-of-the-i2c-communication-protocol/
- https://en.wikipedia.org/wiki/I%C2%B2C