

**Kauno technologijos universitetas**

Elektros ir elektronikos fakultetas

**Robotizuotų sistemų modeliavimas (T125B151)**

**Sienos sekimo robotas**

Projekto ataskaita

|  |  |
| --- | --- |
|  |  |
| **Viktoras Januška E VS – 9 gr.** Studentas / Studentė |  |
|  |  |
| **lekt. Gintautas Narvydas**  Dėstytojas / Dėstytoja |  |
|  |  |

**Kaunas, 2022**

**Turinys**

[Darbo tikslas 3](#_Toc107206407)

[Uždaviniai 4](#_Toc107206408)

[Naudojama įranga 5](#_Toc107206409)

[Įrangos paruošimas ir dalių montavimas 10](#_Toc107206410)

[Elektrinės schemos jungimas 11](#_Toc107206411)

[Darbui sukonfigūruojamas mikrokompiuteris Raspberry Pi 13](#_Toc107206412)

[Konfigūruojami ultragarsiniai atstumo jutikliai 14](#_Toc107206413)

[Servo pavaros konfigūravimas 15](#_Toc107206414)

[Nuolatinės srovės variklio konfigūravimas 16](#_Toc107206415)

[Nuotolinio valdymo režimas 16](#_Toc107206416)

[Sienos sekimo algoritmas 17](#_Toc107206417)

[Bandymai optimizuoti 20](#_Toc107206418)

[Bendra sistemos topologija ir „ROS“ paketo konfigūravimas 21](#_Toc107206419)

[Koncepcija 23](#_Toc107206420)

[Rezultatai ir išvados 23](#_Toc107206421)

**Paveikslų sąrašas**

[**1 pav.** „Raspberry Pi 3b+“ mikrokompiuteris 6](#_Toc107206336)

[**2 pav.** „L298N“ nuolatinės srovės variklių valdymo modulis 6](#_Toc107206337)

[**3 pav.** „MG995“ servo pavara 6](#_Toc107206338)

[**4 pav.** „HC-SR04“ ultragarsinis atstumo jutiklis 7](#_Toc107206339)

[**5 pav.** „HY-SRF05“ ultragarsinis atstumo jutiklis 7](#_Toc107206340)

[**6 pav.** Žeminantis įtampos keitiklis 7](#_Toc107206341)

[**7 pav.** „Li-ion“ baterijos ir jų laikiklis 7](#_Toc107206342)

[**8 pav.** Išorinė baterija 8](#_Toc107206343)

[**9 pav.** Važiuoklės pradinė būsena be pakeitimų 8](#_Toc107206344)

[**10 pav**. Rottorinis enkoderis „KY-040“ 8](#_Toc107206345)

[**11 pav**. Jungiamieji laidai 9](#_Toc107206346)

[**12 pav**. Montažinė plokštė 9](#_Toc107206347)

[**13 pav**. Priekinių ir galinių jutiklių laikiklis 9](#_Toc107206348)

[**14 pav**. Šoninių jutiklių laikiklis 9](#_Toc107206349)

[**15 pav**. Pagrindas priekiniam jutikliam ir servo pavarai įtvirtinti 10](#_Toc107206350)

[**16 pav**. Pagrindas galiniam jutikliam įtvirtinti 10](#_Toc107206351)

[**17 pav**. Išorinės baterijos laikiklis 10](#_Toc107206352)

[**18 pav**. Servo pavaros petys 10](#_Toc107206353)

[**19 pav**. Krumpliaratis judesio perdavimui iš galinės ašies į enkoderio ašį 11](#_Toc107206354)

[**20 pav**. Rotarinio enkoderio laikiklis 11](#_Toc107206355)

[**21 pav**. Sumontuotos dalys ant roboto važiuoklės priekinės dalies 11](#_Toc107206356)

[**22 pav**. Roboto važiuoklės galas, ant jo sumontuoti du jutikliai ir enkoderis 12](#_Toc107206357)

[**23 pav**. Jutiklių jungimo schema 12](#_Toc107206358)

[**24 pav**. Variklio, servo pavaros ir enkoderio jungimo schema 13](#_Toc107206359)

[**25 pav**. Galutinis roboto prototipas, vaizdas iš šono 14](#_Toc107206360)

[**26 pav**. Galutinis roboto prototipas, vaizdas iš viršaus 14](#_Toc107206361)

[**27 pav**. Ultragarso jutiklio funkcijos atstumo skaičiavimui ir šios informacijos skelbimui 15](#_Toc107206362)

[**28 pav**. Kodo ištrauka iš atskirų jutiklių informacijos surinkimo mazgo 16](#_Toc107206363)

[**29 pav**. Kodo ištrauka, servo pavaros kampo keitimo funkcija 16](#_Toc107206364)

[**30 pav**. Funkcijos: Maksimalių ir minimalių verčių ribojimas, bei keitimo funkcija 16](#_Toc107206365)

[**31 pav**. Kodo ištrauka, variklio valdymo funkcijos priklausomai nuo krypties 17](#_Toc107206366)

[**32 pav**. Joy mazgo paleidimas nuotoliniam valdymui 17](#_Toc107206367)

[**33** **pav**. Joy mazgo žinučių klausymas 18](#_Toc107206368)

[**34 pav**. Kodo ištrauka, priklausomai nuo žinutės, keičiamas servo pavaros pasisukimo kampas 18](#_Toc107206369)

[**35 pav**. Kodo ištrauka, analoginių svirčių žinučių nuskaitymas ir greičio signalo siuntimas ir variklio mazgą 18](#_Toc107206370)

[**36 pav**. Nuotolinio valdymo algoritmas 19](#_Toc107206371)

[**37 pav**. Automatinio valdymo algoritmas 20](#_Toc107206372)

[**38 pav**. Kodo ištrauka, PID reguliatorius 21](#_Toc107206373)

[**39 pav**. Kodo ištrauka, Enkoderio mazgas 21](#_Toc107206374)

[**40 pav**. CmakeLists.txt failas, jame surašyti mazgų failai 22](#_Toc107206375)

[**41 pav**. „params.yaml“ failas, jame surašyti atstumas nuo sienos, PID dedamosios 22](#_Toc107206376)

[**42 pav**. „Auto.launch“ failas, kuriame surašomi norimi paleisti mazgai 23](#_Toc107206377)

[**43 pav**. Roboto ROS sistemos topologija 23](#_Toc107206378)

# 

# Darbo tikslas

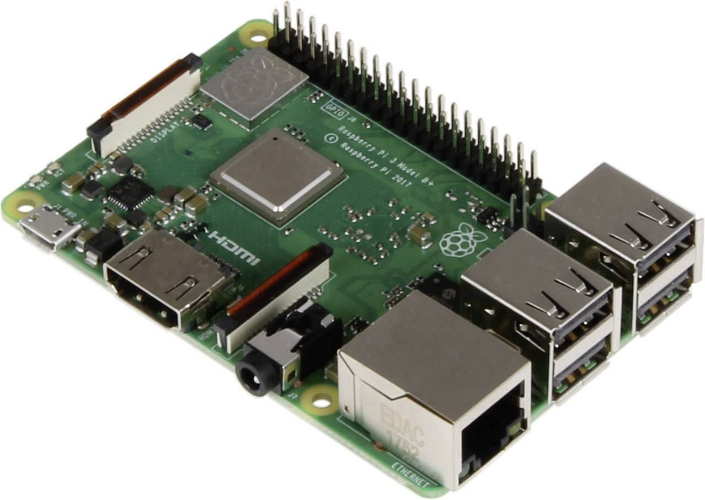
Sukurti robotą, kuris autonomiškai sektų sieną ir išvengtų kliūčių. Robotas sieną turi sekti jam užduotu atstumu ir užtikrinti, šis atstumas turi būti išlaikomas. Taip pat robotą turi būti galima valdyti nuotoliniu būdu. Sistemai naudojama „ROS“, kodas rašomas „Python“ kalba.

# Uždaviniai

1. Sukonfigūruoti mikrokompiuterį „Raspberry Pi“, kad jame galėtų veikti Robotų Operacinė Sistema ir prie jo būtų galima prisijungti nuotoliniu būdu.
2. Važiuoklės paruošimas. Papildomų 3D dalių projektavimas, kad būtų galima sumontuoti jutiklius ir bateriją.
3. Servo pavaros konfigūravimas, kada būtų galima keisti roboto važiavimo kryptį.
4. Nuolotinės srovės variklio konfigūravimas. Tam panaudojamas nuolatinės srovės variklių valdymo modulis.
5. Jutiklių paruošimas. Robotas turi naudotis jutikliais ir nustatyti atstumus skirtingomis kryptimis iki sienų arba kliūčių.
6. Paruošti nuotolinio valdymo režimą su „Dualshock 4“ pultu.
7. Sienos sekimo algoritmo kūrimas. Robotas turi važiuoti palei sieną ir tuo pačiu išvengti prieš jį esančias kliūtis. Sukonfigūruoti sienos sekimą iš abiejų pusių.
8. Algoritmo ir mechaninių dalių optimizavimas.
9. „ROS“ mazgų paruošimas.

# Naudojama įranga

1. „Raspberry Pi 3b+“ mikrokompiuteris



**1 pav.** „Raspberry Pi 3b+“ mikrokompiuteris

1. „L298N“ nuolatinės srovės variklių valdymo modulis

A picture containing electronics

Description automatically generated

**2 pav.** „L298N“ nuolatinės srovės variklių valdymo modulis

1. „MG995“ servo pavara



**3 pav.** „MG995“ servo pavara

1. 5 x „HC-SR04“ ultragarsiniai atstumo jutikliai



**4 pav.** „HC-SR04“ ultragarsinis atstumo jutiklis

1. 2 x „HY-SRF05“ ultragarsiniai atstumo jutikliai



**5 pav.** „HY-SRF05“ ultragarsinis atstumo jutiklis

1. Žeminantis įtampos keitiklis



**6 pav.** Žeminantis įtampos keitiklis

1. 3 x 3.7V 18650 Ličio jonų baterijos ir jų laikiklis

|  |  |
| --- | --- |
|  |  |

**7 pav.** „Li-ion“ baterijos ir jų laikiklis

7. Išorinė baterija „Swissten“ mikrokompiuterio matinimui



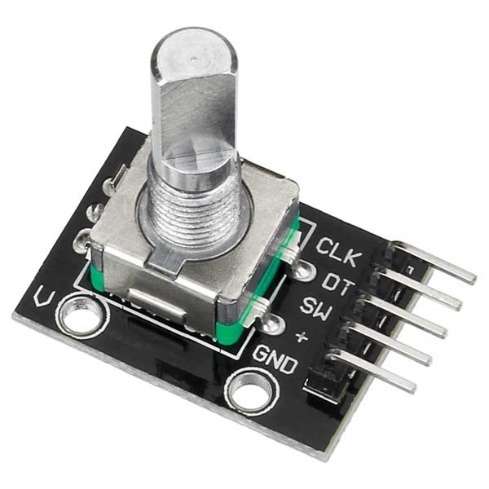
**8 pav.** Išorinė baterija

1. Važiuoklė



**9 pav.** Važiuoklės pradinė būsena be pakeitimų

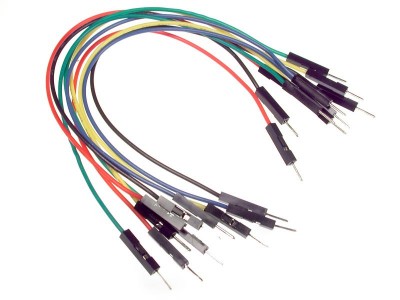
1. Rotarinis enkoderis „KY-040“



**10 pav**. Rottorinis enkoderis „KY-040“

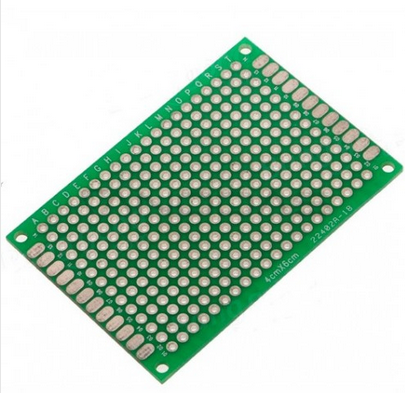
1. Papildomos dalys elektronikai:

* Jungiamieji laidai



**11 pav**. Jungiamieji laidai

* Montažinė plokštė



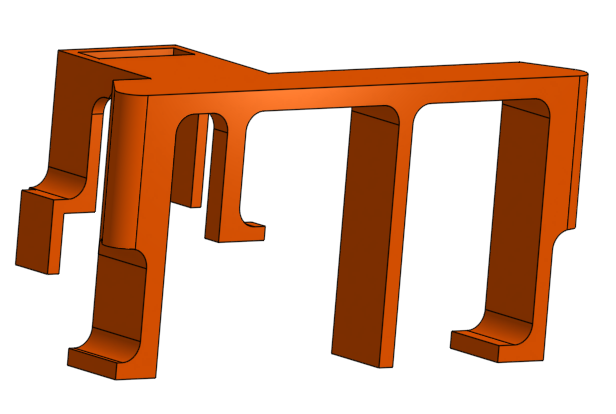
**12 pav**. Montažinė plokštė

1. 3D Spausdintos dalys montavimui:

* Ultrugarsinių atstumo jutiklių laikikliai

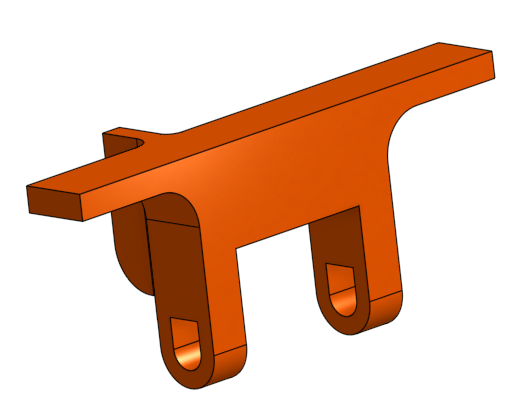
|  |  |
| --- | --- |
| **13 pav**. Priekinių ir galinių jutiklių laikiklis | **14 pav**. Šoninių jutiklių laikiklis |

* Pagrindas priekiniam jutikliam ir servo pavarai įtvirtinti



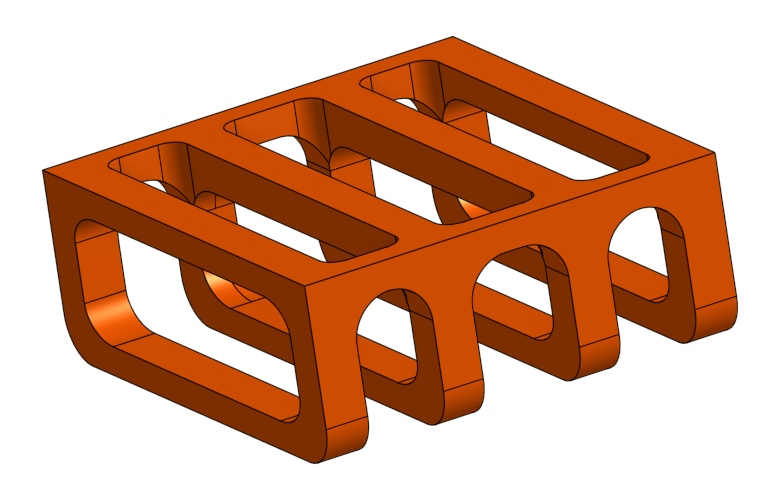
**15 pav**. Pagrindas priekiniam jutikliam ir servo pavarai įtvirtinti

* Pagrindas galiniam jutikliam



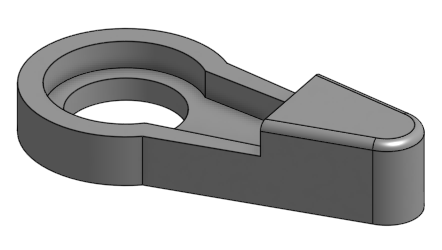
**16 pav**. Pagrindas galiniam jutikliam įtvirtinti

* Išorinės baterijos laikiklis



**17 pav**. Išorinės baterijos laikiklis

* Servo pavaros petys vairavimui



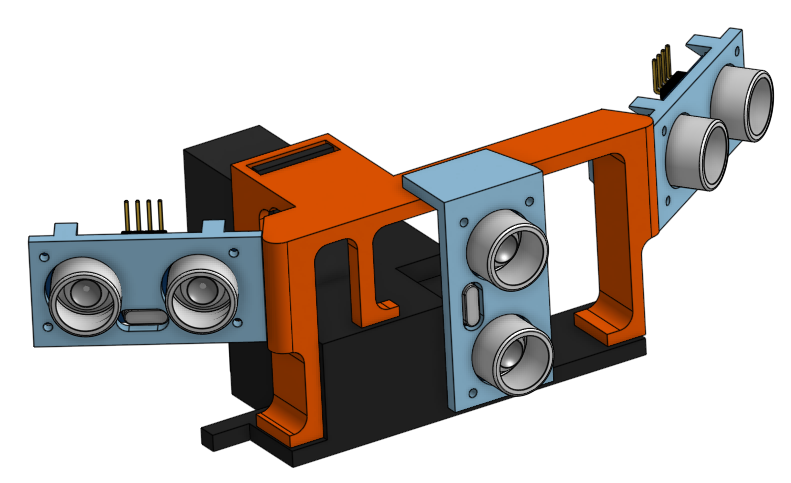
**18 pav**. Servo pavaros petys

* Enkoderio pagalbinės dalys

|  |  |
| --- | --- |
| **19 pav**. Krumpliaratis judesio perdavimui iš galinės ašies į enkoderio ašį | **20 pav**. Rotarinio enkoderio laikiklis |

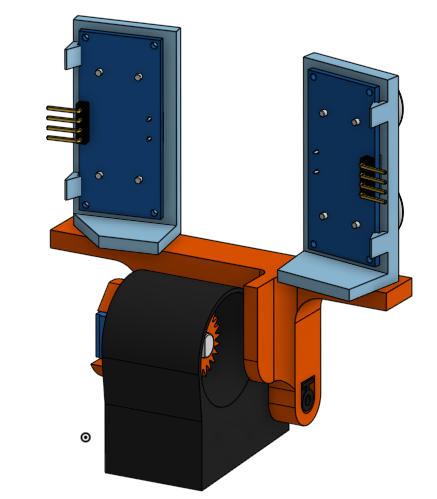
# Įrangos paruošimas ir dalių montavimas

Atspausdinus servo pavaros laikiklį, jis sumontuotas taip, kad sulaikytų servo pavaros pasipriešinimo judesį, tam panaudoti varžtai. Servo pavara įmontuota priekinių ratų krypčiai reguliuoti. Kadangi naudojama „Ackermann“ tipo važiuoklė, stengtasi pavarą įmontuoti kuo arčiau, kad ji galėtų judėti, kuo didesniame diapazone, geresniam valdymo tikslumui.Galiausiai ant servo pavaros laikiklio sumontuoti jutiklių laikikliai.



**21 pav**. Sumontuotos dalys ant roboto važiuoklės priekinės dalies

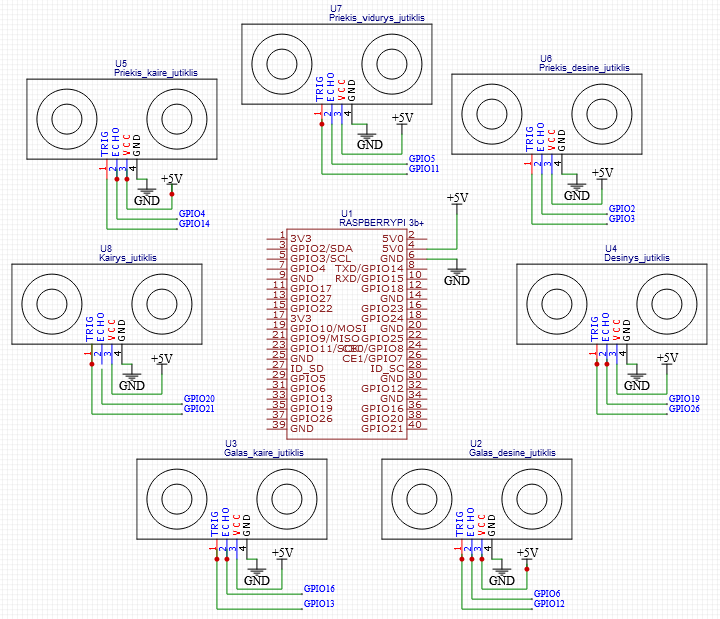
Toliau atspausdintas galinių jutiklių laikiklis, taip pat ir enkoderio priedai. Turima važiuoklė gali būti varoma dviejų nualatinės srovės variklių, tačiau tikslumo dėlei galutinis variklis atjungtas ir išimtas. Jo vietoje įstatytas rotarinis enkoderis, kad robotas galėtų nustatyti savo važiavimo kryptį ir greitį.



**22 pav**. Roboto važiuoklės galas, ant jo sumontuoti du jutikliai ir enkoderis

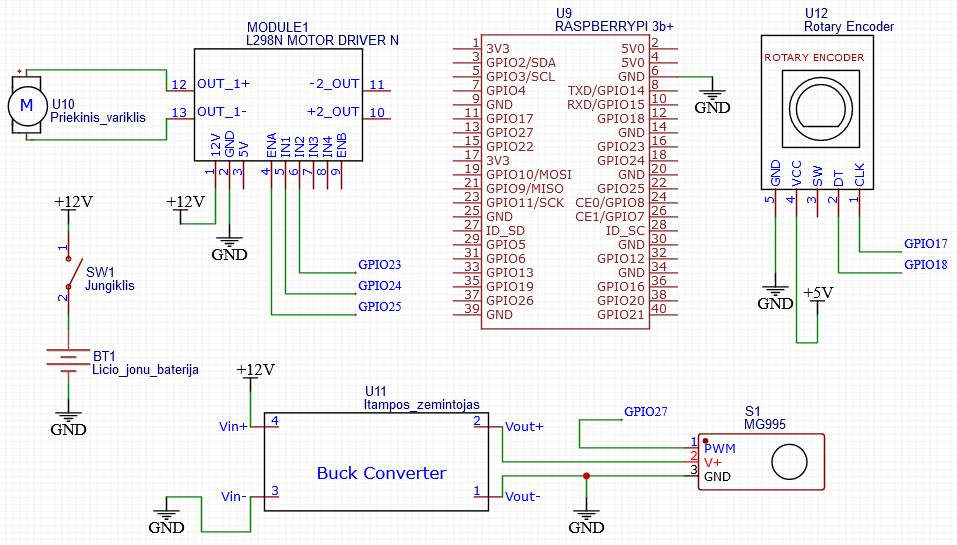
# Elektrinės schemos jungimas

Roboto mikrokompiuteris yra užmaitinamas 5V išorinės baterijos. Ultragarsiniai jutikliai taip pat užmaitinami iš šios baterijos. Kiekvienam ultragarsiniam jutikliui reikia vienos diskretinės išvesties (nusiųsti ultragarso bangą) ir vienos diskretinės įvesties jungties (gauti ultragarso bangą). Kadangi naudojami 7 jutikliai, vien jiems sunaudojama 14 įvesčių/išvesčių.



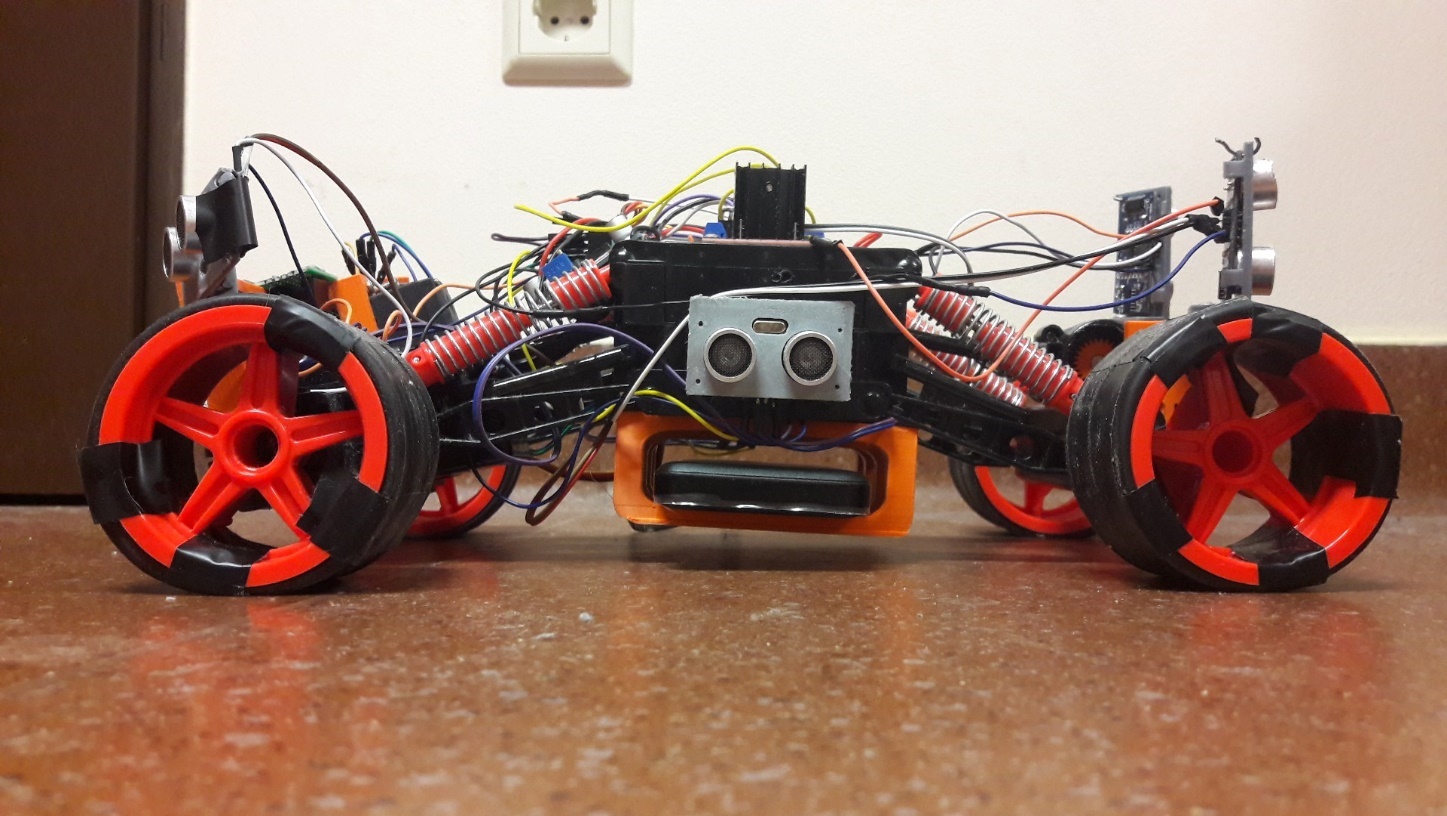
**23 pav**. Jutiklių jungimo schema

Roboto priekinio variklio greičiui ir krypčiai reguliuoti naudojamas “L298N” modulis, kuris yra užmaitinamas iš nuosekliai sujungtų 3 Ličio jonų baterijų. Šio modulio išėjime prijungiame variklio kontaktus. Iš mikrokompiuterio į modulį paduodame 3 signalus: Tiesioginiai krypčiai, atbulinei krypčiai, greičiui. Taip pat, kad galėtume atskirai įjungti ir išjungti šią bateriją, įdėjome jungiklį. Baterijas prijungiame prie įtampos žemintuvo, ant kurio nustatėme 6V įtampą, iš jo tiekiame maitinimą servo pavarai “MG995”. Servo pavaros pečio kampui keisti iš mikrokompiuterio siunčiame impulso pločio moduliacijos signalą į pavaros signalo įėjimo jungtį. Paskutinis komponentas, kurį įdėjome buvo rotarinis enkoderis, jį užmaitiname iš 5V išorinės baterijos išėjimo. Į dvi mikrokompiuterio jungtis paduodame enkoderio impulsus. Priklausomai nuo impulsų sekos, nustatoma galinės ašies sukimosi kryptis.

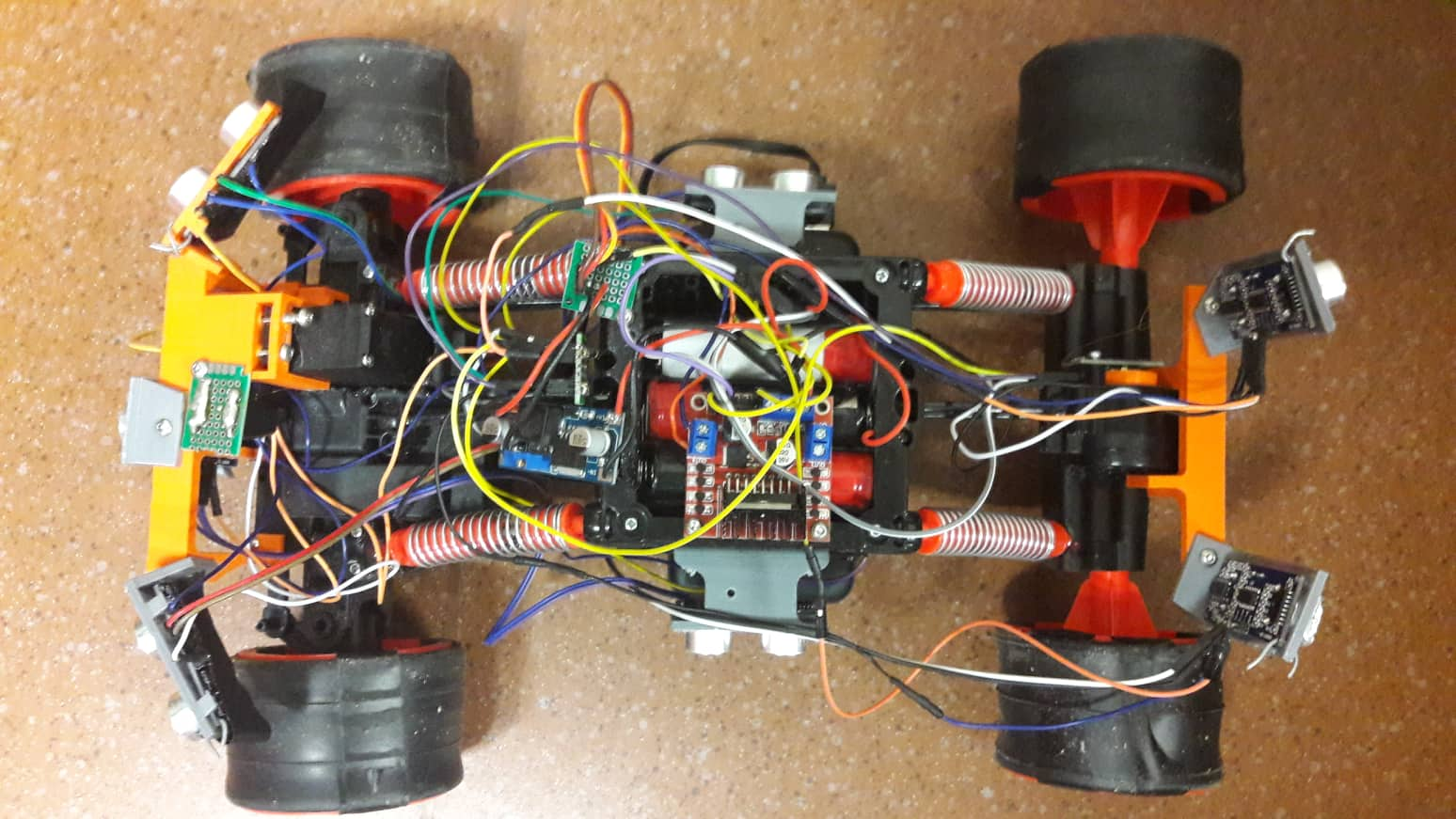


**24 pav**. Variklio, servo pavaros ir enkoderio jungimo schema

Galiausiai visi komponentai sujungti pagal schemas. Tam panaudoti jungiamieji laidai, montažinės plokštės, į kurias bendrai sujungtas 5V maitinimas. 24 paveiksle matome galutinį roboto prototipą iš šono. 25 paveiksle matome galutinį roboto prototipą iš viršaus. Išorinės baterijos laikiklis prisuktas po važiuokle, o pats mikrokompiuteris virš jos. Padaryta skylė korpuse, kad būtų galima tiekti maitinimą į mikrokompiuterį per USB. Ličio jonų baterijos ir variklio valdymo modulis prikabinti viršutinėje korpuse dalyje.



**25 pav**. Galutinis roboto prototipas, vaizdas iš šono



**26 pav**. Galutinis roboto prototipas, vaizdas iš viršaus

# Darbui sukonfigūruojamas mikrokompiuteris Raspberry Pi

Šį projektą buvo norima įgyvendinti naudojant robotų operacinę sistemą (Angl. Robot Operating System). Ji pasirinkta dėl to, nes yra daug atviro kodo pavyzdžių, patogu dirbti nuotoliniu būdu. Taip pat nereikia visko rašyti viename valdiklyje, kiekvienai skirtingai funkcija gali būti sukurti atskiri mazgai.

Pirmiausia į kompiuterį parsisiųstas [„Ubuntu 16.04”](https://learn.ubiquityrobotics.com/kinetic_pi_image_downloads) operacinės sistemos atvaizdas su įrašyta robotų operacine sistema (ROS kinetic). Šis atvaizdas įrašomas į SD kortelę ir ją įdedame į roboto mikrokompiuterio SD kortelių skaitytuvą.

Mikrokompiuteriui suteikę maitinimą prijungime robotą prie iš savo kompiuterio sukurto mobilaus ryšio taško. Dabar galime naudotis internetu ir naudodami SSH protokolą gauti prieigą prie roboto.

Patogesniam darbui vykdyti į mikrokompiuterį įrašyta programa „Github“. Naudodami ją galime lengvai ir greitai vystyti programinę įrangą nuosavame kompiuteryje ir testuoti ją internetu parsisiuntus į robotą.

Darbo pradžioje sukuriame „catkin\_ws“ direktoriją ir joje sukuriame savo roboto projektui skirtą paketą.

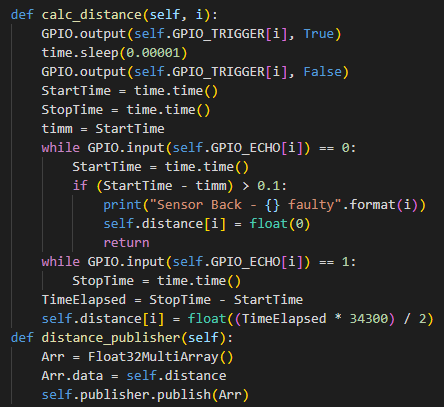
# Konfigūruojami ultragarsiniai atstumo jutikliai

Kiekvienas ultragarso jutiklis aptinka objektą 15 laipsnių kampu, jeigu objektas yra pakankamai statmenas jutiklio centrinei ašiai .

Jutiklis veikia taip: suteikus 10ms įtampos pulsą į paleidimo (TRIG) įvestį jutiklis išduoda daugiau negu 18 kHz dažnio bangą. Banga atsitrenkus į objektą grįžta atgal, jutiklis priklausomai nuo kiek ilgai truko gauti atsaką į aido (ECHO) išvestį suteikia proporcingai ilgą impulsą.

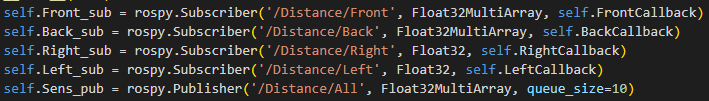
Kadangi naudojami 7 jutikliai, reikia, kad jie vienas su kitu neinterferuotų. Nuspręsta, kad šoniniai (kairys ir dešinys) jutikliai gali veikti nepriklausomai nuo kitų, nes jie abu nusisukę į skirtingas puses). Priekyje sumontuoti 3 jutikliai, juos atskirai leidžiame vienas po kito, kad nesusidarytų interferencija. Gale turime 2 jutiklius, juos leidžiame irgi vienas po kito.

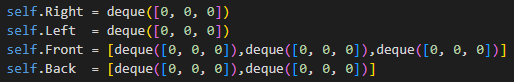
Gaunasi, kad galima sukurti 4 atskirus ROS mazgus, 4 skirtingom pusėm. Priekinis siunčia žinutę, kuri susidaro iš 3 jutiklių komponentų, galinis iš 2, šoniniai siunčia po vieną žinutės komponentą. Taip pat sukurtas atskiras mazgas, kuris prenumeruoja šių jutiklių temas, naudodamas slenkamo vidurkio filtrą, šiek tiek sumažina triukšmus ir skelbia išfiltruotą visų jutiklių informaciją. 26 pav. funkcija apskaičiuojanti atstumą atitinkamam jutikliui ir funkcija skelbianti šių jutiklių informaciją. Taip pat, jeigu signalas atgal negrįžta, ciklas kartojamas ir parašoma žinutė, jog kažkas negerai su jutikliu.

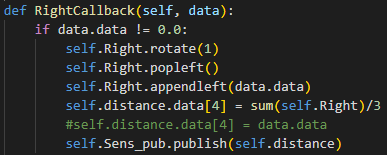


**27 pav**. Ultragarso jutiklio funkcijos atstumo skaičiavimui ir šios informacijos skelbimui

Jutiklių informacijos surinkimo mazgas kiekvieną kartą gavęs žinutę iš atskiro jutiklio apskaičiuoja vidurkį ir skelbia visų jutiklių atstumus:



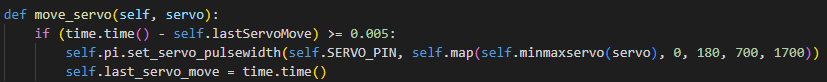




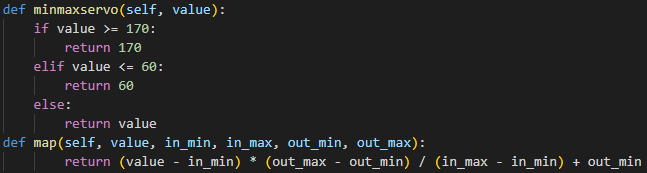
**28 pav**. Kodo ištrauka iš atskirų jutiklių informacijos surinkimo mazgo

# Servo pavaros konfigūravimas

Servo pavara turi gauti 50 Hz impulso pločio moduliacijos signalą. Šio signalo plotis gali kisti skirtingai. Dažniausiai keičiant signalo plotį nuo 1000 iki 2000 mikrosekundžių pavara pasisuka nuo 180 laipsnių. Tačiau mūsų naudojama pavara veikė 700 iki 1700 diapazone. Kad keisti pavaros pasisukimo kampą, sukurta funkcija map(), kuri yra Arduino bibliotekoje. Naudojant šią funkciją atitinkamai uždavus reikiamą pasisukimo kampą išskaičiuojamas impulso plotis. Taip pat panaudojame funkciją mimaxservo(), kuri neleidžia pavarai nukrypti daugiau nuo leidžiamų reikšmių.



**29 pav**. Kodo ištrauka, servo pavaros kampo keitimo funkcija

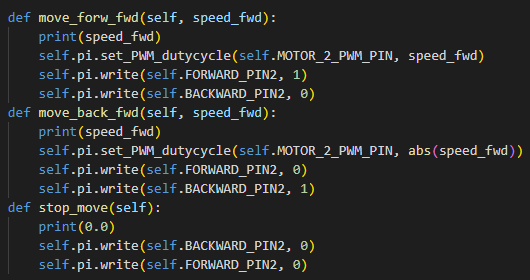


**30 pav**. Funkcijos: Maksimalių ir minimalių verčių ribojimas, bei keitimo funkcija

# Nuolatinės srovės variklio konfigūravimas

Nuolatinės srovės variklis turi 2 krypties signalo įvedimo jungtis. Viena tiesioginei krypčiai, kita atbulinei krypčiai. Jeigu į tiesioginės krypties įvestį paduodame aukšto potencialo signalą (būlinės logikos vienetą), o į atbulinės krypties įvestį žemo potencialo signalą (būlinės logikos nulį), variklio sukimosi kryptis bus tiesioginė. Sukeitus signalus vietomis variklis judės atgal. Variklio greitis keičiamas, keičiant jam paduodamą įtampą. „L298N“ Modulis keičia įtampą priklausomai nuo paduodamo impulso pločio. Vidutinė impulso pločio moduliuojamo signalo įtampa, paduodamo iš mikrokompiuterio, bus atitinkamai proporcinga vidutinei iš baterijos moduliuojamai paduodamai įtampai. Iš mikrokompiuterio impulso plotį galime keisti nuo 0 vertės (vidutinė įtampa 0) iki 255 vertės (vidutinė įtampa 5V).

Turimoje sistemoje sukurtas “ROS” mazgas, kuris gauna greičio žinutes iš valdymo mazgo ir iš mikrokokompiuterio siunčia atitinkamą krypties ir greičio signalą nuolatinės srovės variklių valdymo moduliui. Šiame mazge taip pat įdėtas greičio valdymas, kuris gali būti įjungtas, jeigu sistemoje veikia rotarinis enkoderis.



**31 pav**. Kodo ištrauka, variklio valdymo funkcijos priklausomai nuo krypties

Mūsų atveju variklis juda, padavus impulso plotį nuo 90 iki 255. Kitais atvejais neužtenka variklio sukimo momento, kad pajudinti robotą.

# Nuotolinio valdymo režimas

Nuotoliniam valdymui naudojame Bluetooth komunikaciją ir prie mikrokompiuterio prijungiame „PS4” pultą. „ROS“ turi standartizuotą paketą įvairiems pultams naudoti pavadinimu „Joy“. Kad jį panaudoti, į savo .launch failą įrašome 31 pav. Pateiktą eilutę:



**32 pav**. Joy mazgo paleidimas nuotoliniam valdymui

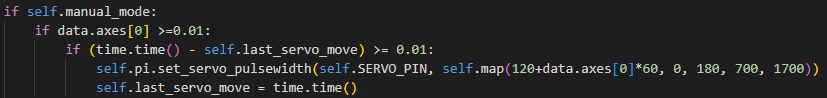
Paleidę šį mazgą galime klausyti iš šio mazgo gaunamas žinutes. Jis siunčia skirtingų mygtukų paspaudimus ir analoginių svirčių reikšmes.

Sukurtas teleoperacinio režimo valdymo mazgas, kuris klauso „/Joy“ žinučių ir atitinkamai su jomis atlieka veiksmus:

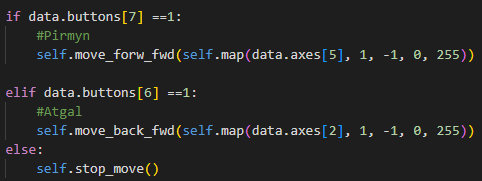
* Pasirenkamas „X“ mygtukas autonomnio važiavimo režimui pradėti.
* Mygtukas „O“ sustabdo šį režimą
* mygtukas „Trikampis“ paleidžia rankinio valdymo režimą.
* Analoginė svirtis R2 siunčia signalą greičio valdikliui judėti į priekį priklausomai nuo svirties paspaudimo dydžio. Veikia tik pasirinkus nuotolinio valdymo režimą
* Analoginė svirtis L2 daro tą patį, ką ir R2, tik siunčiumas signalas važiuot atgal.
* Analoginė svirtis L3 pasisukus į šonus valdo servo pavaros kampą.



1. **pav**. Joy mazgo žinučių klausymas



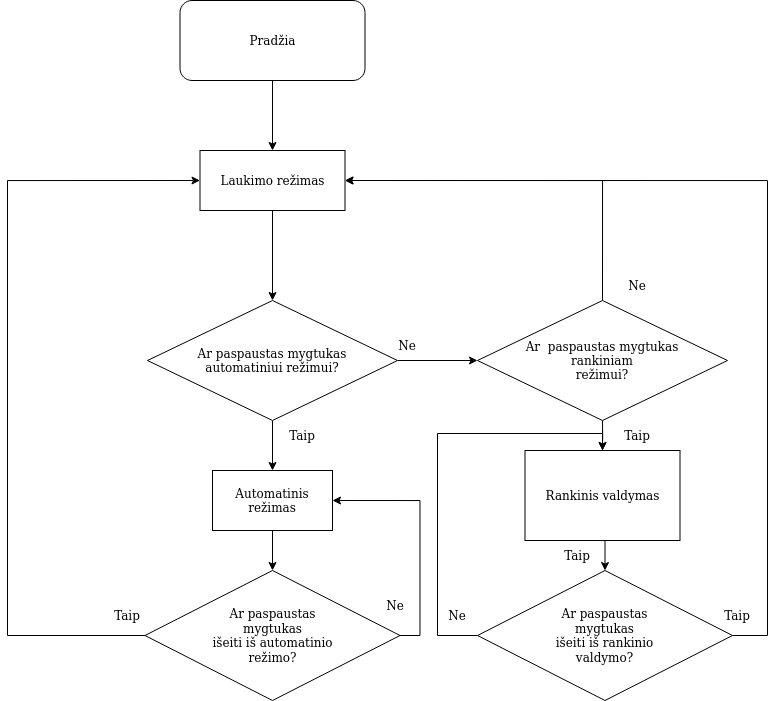
**34 pav**. Kodo ištrauka, priklausomai nuo žinutės, keičiamas servo pavaros pasisukimo kampas



**35 pav**. Kodo ištrauka, analoginių svirčių žinučių nuskaitymas ir greičio signalo siuntimas ir variklio mazgą

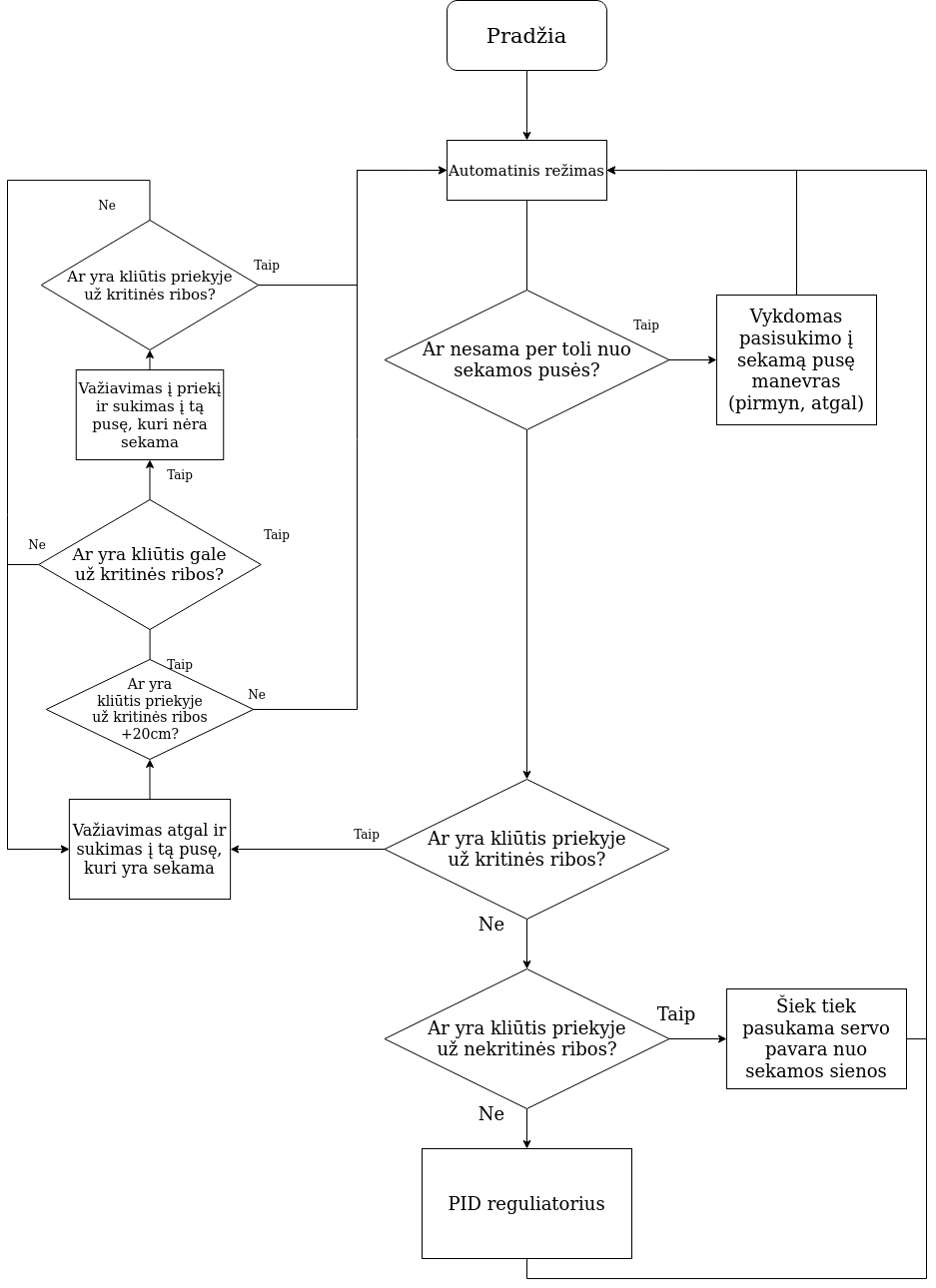
# Sienos sekimo algoritmas

Parašytas pagrindinis valdymo mazgas, kuriame galima pasirinkti rankinį arba automatinį režimą. Paspaudus atitinkamus mygtukus galima pereiti į laukimo režimą, rankinį režimą arba automatinį režimą.



**36 pav**. Nuotolinio valdymo algoritmas

Automatinio režimo metu robotas stengiasi sekti pasirinktą sieną, panaudojamas PID reguliatorius sekimo efektyvumui. Jeigu robotas pastebi kliūtį priekyje, pereina į kliūties priekyje būseną ir važiuoja atgal, kol nuo kliūties nuvažiuoja 20cm. Jeigu bevažiuojant atgal pastebi gale esančią kliūtį, sustoja ir važiuoja į priekį, kol nuo galinės kliūties nuvažiuoja 20cm. Kai nuo galinės kliūties nuvažiuoja 20 cm, pereina į normalų režimą. Taip pat jeigu dešinėje pusėje siena yra už atitinkamos ribos, robotas stengiasi pasitukti link šios sienos priekiu, nes tikėtina, kad dešinėje pusėje yra siena.



**37 pav**. Automatinio valdymo algoritmas

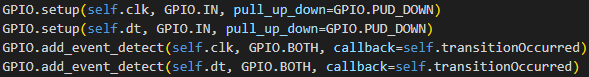
PID reguliatorius sistemoje naudojamas pagerinti valdymo tikslumą, jam užduodamas atitinkamas atstumas, kurio turi laikytis. Pagal dešinio arba kairio jutiklio išmatuotą atstumą iki sienos, suskaičiuojama paklaida ir pagal ją skaičiuojamas PID reguliatoriaus dedamosios.



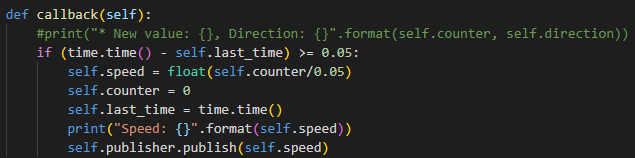
**38 pav**. Kodo ištrauka, PID reguliatorius

# Bandymai optimizuoti

Bandyta sistemą optimizuoti panaudojant rotarinį enkoderį. Sukurtas mazgas greičio skaičiavimui ir reguliavimui ir mazgas greičio reguliavimui.





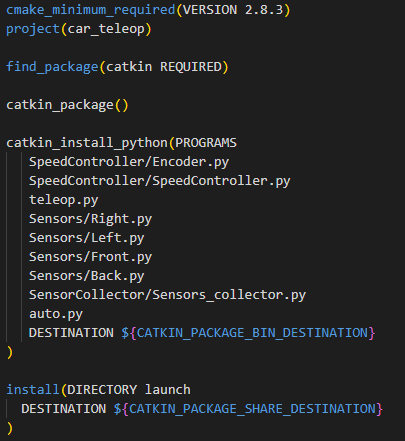


**39 pav**. Kodo ištrauka, Enkoderio mazgas

Tačiau pastebėta, jog jis ne visada per pavarų sistemą gerai prasisuka ir dažnai užstringa. Todėl galutiniame variante nuspręsta jo nenaudoti.

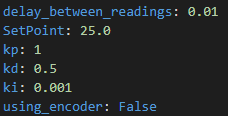
# Bendra sistemos topologija ir „ROS“ paketo konfigūravimas

Norint, kad robotas veiktų, reikia paleisti visus reikalingus mazgus, visų pirma į „CmakeLists.txt“ failą surašome reikalingų mazgų failų vietas, tai matom 38 pav.



**40 pav**. CmakeLists.txt failas, jame surašyti mazgų failai

Į failą „params.yaml“ surašomi reikalingi sistemos parametrai, kuriuos užkrauna parametrų serveris prieš paleidžiant robotą. Kodo eilutėje užkraunami šie parametrai.



**41 pav**. „params.yaml“ failas, jame surašyti atstumas nuo sienos, PID dedamosios

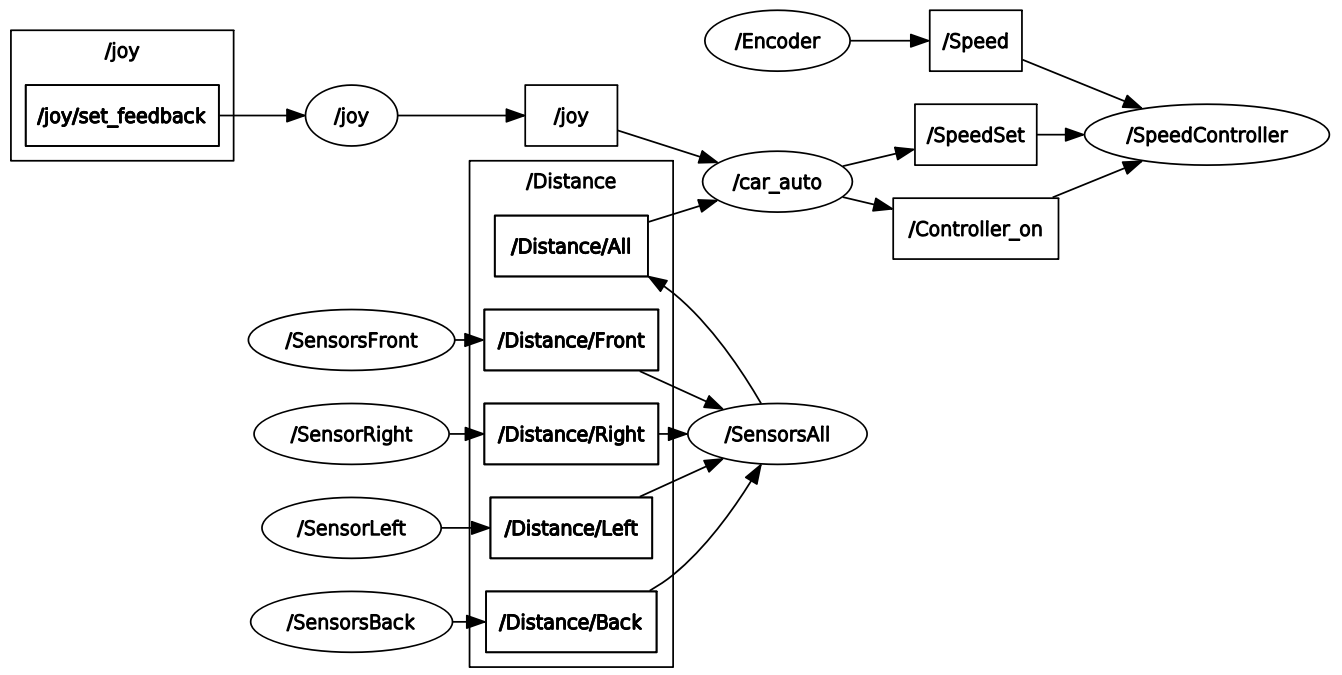
Galiausiai norint paleisti visus mazgus, suršytas kiekieno mazgo paleidimas atskirame faile „Auto.launch“ ir rašoma:

“$roslaunch car\_auto Auto.launch”



**42 pav**. „Auto.launch“ failas, kuriame surašomi norimi paleisti mazgai

Galutinę sistemos topologiją matome paveiksle:



**43 pav**. Roboto ROS sistemos topologija

# Koncepcija

Sistema sukurta taip, kad robotas išvengtų kliūčių priešais jį, nenuvažiuotų už kampo, taip pat, kad naudodamas PID reguliatorių sektų sieną. Naudodami Bluetooth ir „DS4” pultelį galime valdyti robotą, arba paleisti jį autonominiame režime, tai leidžia lengvai ir patogiai naudotis robotu.

# Rezultatai ir išvados

Atspausdintos reikalingos roboto dalys, sukurtas sekimo algoritmas, taip pat stengtasi optimizuoti robotą. Tačiau algoritmas dar turi kėblumų, dėl ko robotas dažnai atsitrenkia į sieną arba nuvažiuoja į šalį. Manoma jog būtų paprasčiau buvę naudoti mažiau jutiklių, tačiau turėdami daugiau sumažiname šansą robotui atsitrenkti į kliūtį.