Izrada programske podrške za Raspberry Pi mobilnu robotsku platformu

Skroban, Matija

Undergraduate thesis / Završni rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:200:393822

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-18**

Repository / Repozitorij:

Faculty of Electrical Engineering, Computer Science and Information Technology Osijek

SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA, OSIJEK

Sveučilišni preddiplomski studij računarstva

Izrada programske podrške za Raspberry Pi mobilnu robotsku platformu

Završni rad

Matija Skroban

Osijek,2019.

SADRŽAJ

1. UVOD 1

2. TEORIJSKA PODLOGA 2

2.1. Raspberry Pi 2

2.2. AlphaBot2-Pi 4

2.3. Raspbian 9

2.4. Python 9

2.5. Robotski operacijski sustav(ROS) 10

2.6. Programska podrška otvorenog koda 10

2.7. Putty 11

3. REALIZACIJA ZADATKA 12

3.1. MovementCommands.py 13

3.2. Infrared_dodging.py 16

3.3. Ultrasonic_dodging.py 18

3.4. Ultrasonic_following.py 20

4.ZAKLJUČAK 24

LITERATURA 25

SAŽETAK 26

ABSTRACT 27

ŽIVOTOPIS 28

1. UVOD

Tema ovog završnog rada je izrada programske podrške za Raspberry Pi mobilnu robotsku platformu. Detaljniji opis zadatka je: istražiti i opisati moguće načine za izradu programske podrške otvorenog koda za mobilnu robotsku platformu temeljenu na Raspberry Pi okruženju. Izraditi pokazne aplikacije koje koriste različite senzore za upravljanje mobilnim robotom. Mobilna robotska platforma za koju će se raditi programska podrška zove se Alphabot2-Pi. Koristit će se dva različita senzora za upravljanje Alphabot2-Pi robotom, to su infracrveni senzor i ultrazvučni senzor. Za izradu pokaznih aplikacija koristit će se programski jezik Python.

2. TEORIJSKA PODLOGA

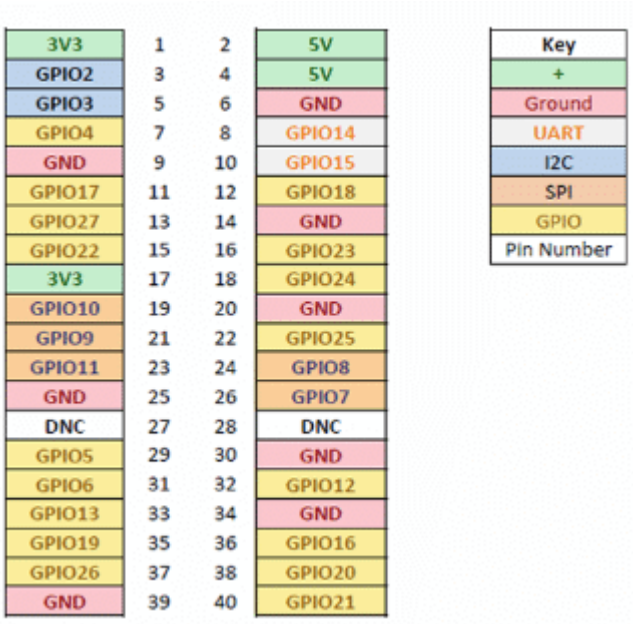
2.1. Raspberry Pi

Raspberry Pi [1] je serija mini računala proizvedena od strane Raspberry Pi zaklade. Razvijena je s ciljem unaprjeđenja podučavanja računalnih znanosti u školama i zemljama u razvoju. Prvi model postao je znatno popularniji nego očekivano, prodavan je izvan ciljnog tržišta za uporabu kao što je robotika.

Trenutno postoje četiri generacije Raspberry Pi računala. Za rad s Alphabot2-Pi koristit će se Raspberry Pi 3 Model B koji dolazi skupa s Alphabot2-Pi kompletom za izgradnju robota. Raspberry Pi 3 Model B ima četverojezgreni 1.2 GHz Broadcom BCM2837 64 bitni procesor, 1 GB RAM-a, 40-iglično GPIO sučelje, 4 USB 2 priključka, HDMI priključak, Mikro SD priključak, Wifi i Bluetooth kartice za bežično spajanje te priključak 100BASE-TX za žično spajanje.

Sl. 2.1 Raspberry Pi 3 model B

Iglice Raspberry Pi-a služe za komunikaciju između Raspberry Pi-a i Alphabot2-Pi-a. Te iglice čine Raspberry Pi pogodnim za robotiku te su one jedan od glavnih razloga zašto je Raspberry Pi tako popularan.

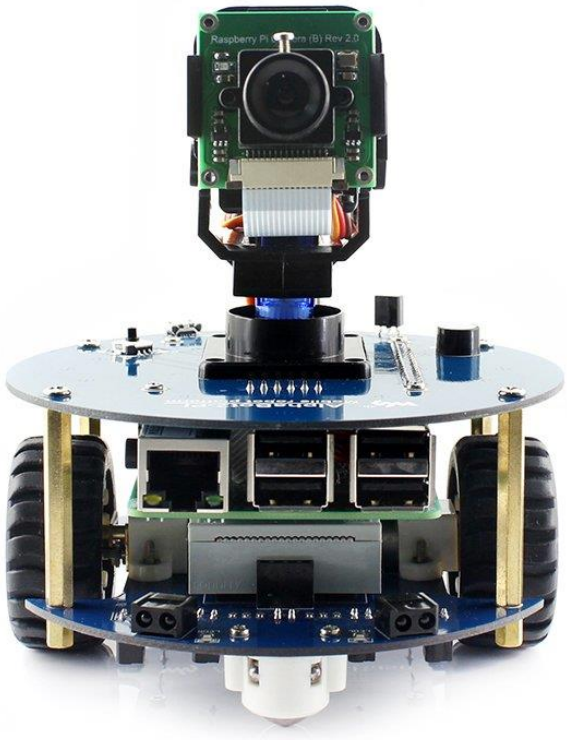


Sl. 2.2. Raspored iglica Raspberry Pi 3 model B [2]

Iako su GPIO iglice Raspberry Pi miniračunala važne, one same po sebi ne rade ništa već im je potrebno narediti što da rade. Kao što im ime govori one mogu djelovati kao ulaz i izlaz te kada programiramo zadajemo im da mogu biti ulaz ili izlaz. Raspberry Pi 3 model B ima 40 iglica. Sve igle nisu jednake te ne obavljaju jednake funkcije, na slici 2.2. su opisane funkcije iglica. Pri programiranju za Raspberry Pi postoje dva načina označavanja iglica BOARD i BCM. BOARD opcijom specificiramo da koristimo brojeve od 1 do 40 na našoj pločici dok opcijom BCM specificiramo da koristimo iglice koje su sposobne za GPIO rad, to su one iglice koje na slici 2.2. imaju GPIO napisan kraj njih.

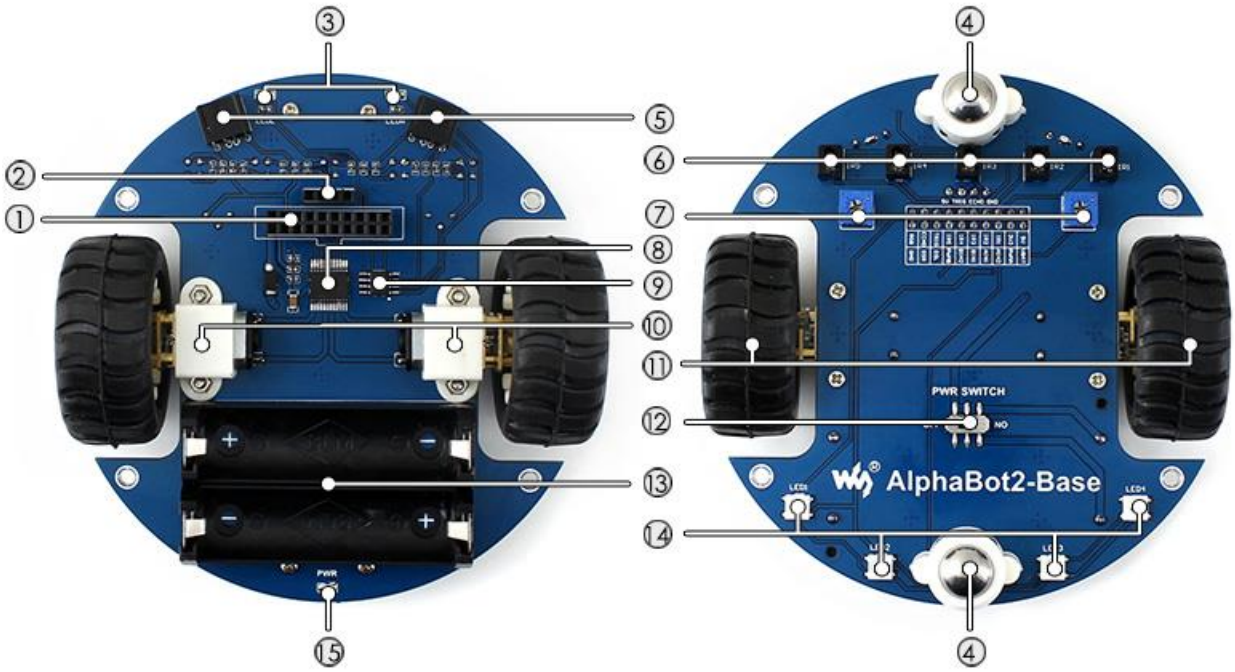
2.2. AlphaBot2-Pi

AlphaBot2-Pi je mali robot u kojemu je ugrađen Raspberry Pi 3 Model B koji upravlja njime. AlphaBot2-Pi je lagan za rukovanje, vrlo brzo ga se može sastaviti i početi raditi njime.



Sl. 2.3. Sastavljeni AlphaBot2-Pi

Postoji više modela AlphaBot2 robota, a to su: AlphaBot2-Pi, AlphaBot2-Ar, AlphaBot2-PiZero. Glavna dva dijela robota su postolje i prilagodna ploča. Sva tri modela imaju jednako postolje, a razlikuju se po prilagodnoj ploči.



Sl. 2.4. Postolje AlphaBot2 robota [3]

Dijelovi postolja:

1.Sučelje ultrazvučnog modula

2.Upravljačko sučelje

3.Pokazatelji za izbjegavanje prepreka

4.Višesmjerni kotači

5.ST188: reflektivni infracrveni fotoelektrični senzor za izbjegavanje prepreka

6.ITR20001/T: reflektivni infracrveni fotoelektrični senzor za praćenje linija

7.Potenciometar za podešavanje dometa za izbjegavanje prepreka

8.TB6612FNG upravljač motora

9.LM393 komparator napona

10.N20 mikro prijenosnik

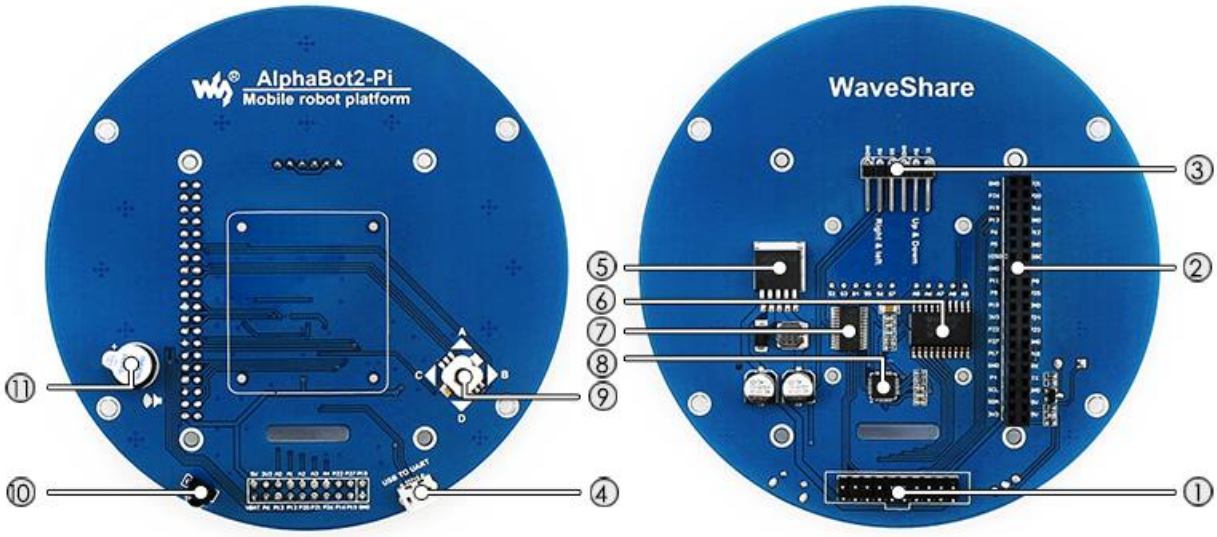
11.Gumeni kotači

12.Prekidač napajanja

13.Držač baterije

14-.WS2812B: RGB LED diode

15.Pokazatelj napajanja



Sl. 2.5. Prilagodna ploča AlphaBot2 robota [3]

Dijelovi prilagodne ploče:

1.AlphaBot2 upravljačko sučelje za povezivanje s podnožjem AlphaBot2-postolja

2.Raspberry Pi sučelje za povezivanje s Raspberry Pi 3 Modelom B

3.Servo sučelje

4.USB u UART pretvarač

5.LM2596: 5V regulator napona

6.TLC1543: 10 bitni pretvarač iz analognog u digitalni signal

7.PCA9685: servo upravljač

8.CP2102: USB u UART pretvarač

9.Komandna palica

10.Infracrveni prijemnik

11.Zvučno-signalizacijski uređaj

Pri izradi programske podrške za Alphabot2-Pi najvažniji elementi koje će kasnije u tekstu biti opisani su infracrveni senzori označeni brojem 5 prikazani na slici 2.4., ultrazvučni senzor ugrađen u sučelje označen brojem 1 na slici 2.4. te TB6612FNG upravljač motora označen brojem 8 na slici 2.4.

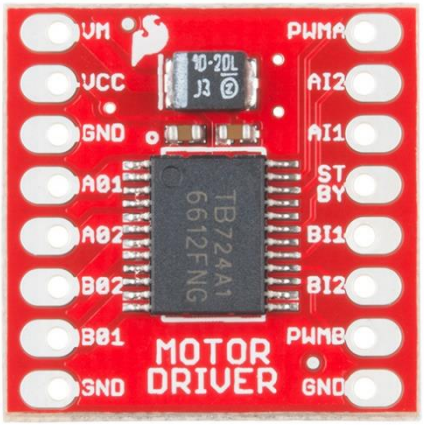
Sl. 2.6. Ultrazvučni modul HC-SRO4

Ultrazvučni modul HC-SRO4 [3] ima 4 nožice: nožica za napon Vcc, nožica za uzemljenje Gnd, nožica predajnika Trig i nožica prijamnika Echo.
Ultrazvučni predajnik šalje ultrazvučni signal te mjeri vrijeme koje je potrebno da se taj signal vrati do prijamnika. Što je Alphabot2-Pi robot bliži nekom objektu to je vrijeme potrebno da se taj signal vrati manje te se može izračunati udaljenost od objekta množenjem tog vremena s brzinom zvuka i dijeljenjem tog rezultata s dva jer je zvuk morao putovati od Alphabot2-Pi robota do objekta i natrag čime je napravio dvostruki put. Maksimalan mogući domet ultrazvučnog modula HC-SRO4 je 550 centimetara, ali zbog raznih faktora u stvarnom svijetu domet koji može ostvariti je 400 centimetara.

Sl. 2.7. Infracrveni senzor ST188

ST188 [4] infracrveni senzor sastoji se od infracrvenog odašiljača i infracrvenog prijamnika. Infracrveni odašiljač konstantno šalje infracrveno svijetlo. Kada se ispred Alphabot2-Pi robota nalazi neki objekt ili prepreka to infracrveno svijetlo se reflektira od tog objekta te infracrveni prijamnik detektira tu reflektiranu svjetlost, u suprotnom, kad nema objekta ispred Alphabot2-Pi

robota infracrveni prijamnik ne detektira ništa. Domet ST188 infracrvenog senzora je između 4 i 13 milimetara te se u Alphabot2-Pi robotu koristi samo za detekciju objekta ispred robota.
Iako se može koristiti i za mjerenje distance, za potrebe Alphabot2-Pi robota je to premali domet, zbog toga ST188 infracrveni senzor u programskom kodu ima dva stanja. 1 kada infracrveni prijamnik detektira reflektiranu infracrvenu svjetlost i 0 kada infracrveni prijamnik ne detektira reflektiranu infracrvenu svjetlost.



Sl. 2.8. TB6612FNG upravljač motora

TB6612FNG [5] upravljač motora kontrolira oba motora Alphabot2-Pi robota, omogućava upravljanje svakim motorom posebno te preko PWM metode kontrolira brzinu okretanje motora.
Za upravljanje smjera kretanja Alphabot2-Pi robota najvažnije su iglice označene s AI1, AI2, BI1 i BI2 prikazane na slici 2.6. AI1 je spojen na iglicu 12 Raspberry Pi-a, AI2 na iglicu 13 , BI1 na iglicu 20 i BI2 na iglicu 21. Lijevi motor je spojen na iglice A01 i A02 a desni motor je spojen na iglice B01 I B02.
Svaki motor je spojen na dvije iglice kako bi se mogao upravljati smjer okretanja motora, a to se radi preko H mosta. H most se sastoji od četiri prekidača s motorom u sredini, preko tih prekidača kontroliramo smjer kretanja struje čime mijenjamo smjer okretanja motora.
Da bi se lijevi motor okretao prema naprijed u programskom kodu postavljamo AI1 na False i AI2 postavljamo na True. U slučaju da želimo da se motor okreće unatrag AI1 postavljamo na True i AI2 na False, desni motor se kontrolira preko iste logike samo je razlika da postavljamo iglice BI1 i BI2.
Za upravljanje brzine okretanja motora koristimo pulsno širinsku modulaciju(PWM). Preko pulsno širinske modulacije pretvaramo digitalni signal Raspberry Pi-a u analogni signal koji

možemo koristiti s našim motorima. To se ostvaruje tako što se struja veoma brzo gasi i pali. Motor brže radi što je struja uključena veći postotak vremena.

2.3. Raspbian

Raspbian [6] je službeni operacijski sustav za Raspberry Pi, baziran je na Debianu koji spada u obitelj Unix operacijskog sustava koji se sastoji samo od besplatnog softvera.
Kreiran je 2012. i još je pod aktivnim razvojem, visoko je optimiziran za Raspberry Pi mikroprocesore s niskim performansama.

2.4. Python

Python [7] je programski jezik opće namjene, jezik visoke razine i jezik otvorenog koda koji će se koristiti za rad s Raspberry Pi-om jer je fleksibilan zato što omogućuje objektno orijentirano, strukturno i aspektno orijentirano programiranje unatoč tome što je sporiji od ostalih sličnih programskih jezika. Sporiji je jer kod izvršava liniju po liniju iako to dovodi do sporog izvođenja koristi se jer kod može izvoditi na bilo kojoj vrsti računala.
Za upravljanje iglica Raspberry Pi-a s pythonom postoje više biblioteka kao što su RPi.GPIO, RPIO.GPIO, wiringPi, pigpio, gpiozero, sve te biblioteke su biblioteke otvorenog koda. Za izradu programske podrške koristit će se RPi.GPIO biblioteku, iako se ne preporučuje za rad u stvarnom vremenu, za upravljanje s Alphabot2-Pi robotom će biti dovoljan. Naredbe RPi.GPIO biblioteke koje ću koristiti su setmode, setup,input, PWM, output, start, stop, ChangeDutyCycle, cleanup i PUD_UP.
Naredbom setmode odabire se označavaju li se iglice BOARD ili BCM metodom, za potrebe ovog rada koristit ću BCM metodu u svim programskim kodovima.
Naredbom setup postavljamo iglice koje želimo kao ulazne ili izlazne.
Naredbom output postavljamo iglicu kao izlaz.
Naredbom input postavljamo iglicu kao ulaz.
Naredba PWM nam omogućuje da radimo s motorima tako što digitalni signal Rasperry Pi-a pretvara u analogni. Tom naredbom postavljamo maksimalnu frekvenciju neke iglice te naredbom start postavljamo koliko posto vremena ta igla odašilje maksimalan napon te naredbom stop zaustavljamo signal.
Naredbom ChangeDutyCycle mijenjamo postotak vremena odašiljanja visokog signala. Preko PWM-a možemo mijenjati brzinu kretanja Alphabot2-Pi robota.

Naredbom cleanup koju zovemo na završetku programa resetiramo sve iglice na originalno stanje prije nego što ih promijenimo svojim kodom.
PUD_DOWN naredbu koristimo da bi uklonili smetnje iz okoliša koje utječu na iglu koja je postavljena kao ulaz, te smetnje bi mogle znatno utjecati na mjerene vrijednosti. PUD_DOWN naredba postavlja odabranu iglu u 0 i jedino ju traženi signal može promijeniti.

2.5. Robotski operacijski sustav(ROS)

Iako mu ime govori da je operacijski sustav on to nije, već je skupina biblioteka i alata koji olakšavaju izradu aplikacija za robote softverskim programerima. Radi na principu otvorenog koda te mu je glavni cilj dijeljenje koda između programera oko svijeta te razvoja robotike preko toga. Kod napisan za ROS[8] može se koristiti i s drugim robotskim okvirima kao što su OpenRAVE, Orocos i Player. Podržava programske jezike Python, C++, Lisp i druge.
Za rad s Python programskim jezikom koristi se rospy biblioteka dok se za rad s C++ koristi roscpp biblioteka.
Robotski operacijski sustav je napravljen za rad s kompleksnim robotskim sustavim koji imaju mnogo senzora, motora i ostalih dijelova. Alphabot2-Pi je jednostavniji robot za kojeg nije potrebno unositi kompleksnost robotskog operacijskog sustava te u ovo radu neće biti korišten.

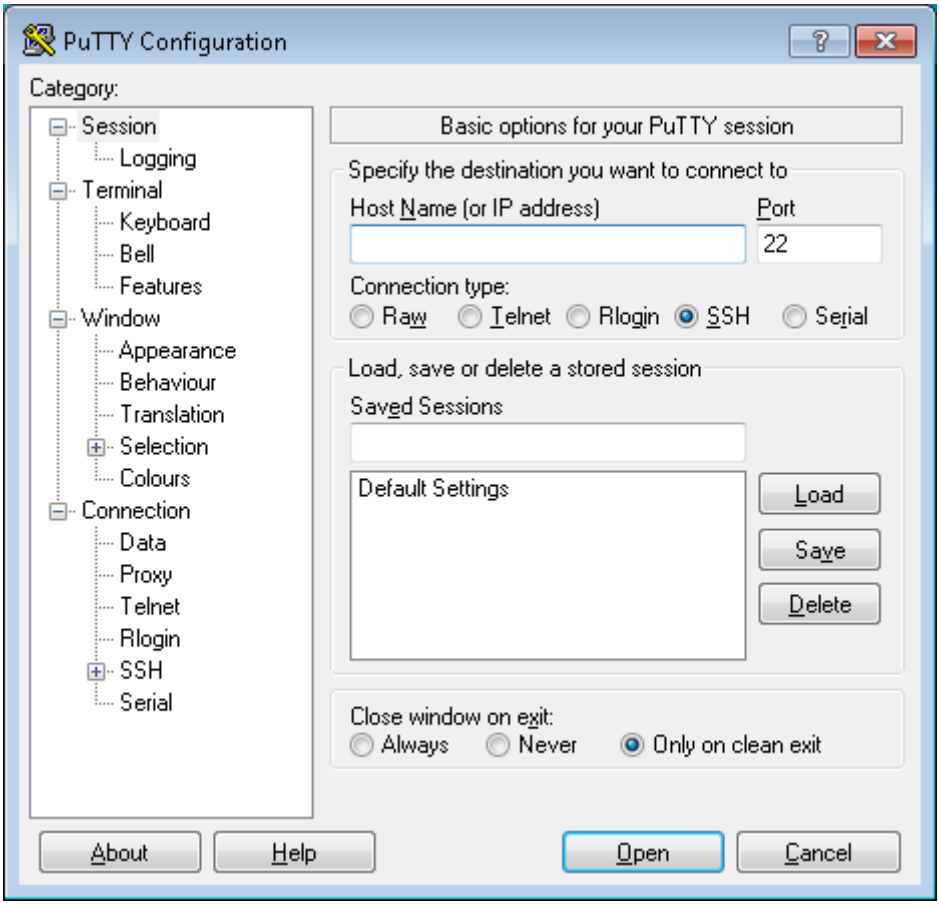
2.6. Programska podrška otvorenog koda

Programska podrška otvorenog [9] koda je vrsta programske podrške u kojoj je izvorni kod dostupan svim korisnicima te ga oni mogu koristiti i izmjenjivati. Cilj programske podrške otvorenog koda je širenje znanja te poboljšavanje koda međusobnim radom mnogih programera cijelog svijeta. Programska podrška otvorenog koda je vrlo značajna te da ne postoji, većina programske podrške korištene u ovom radu ne bi postojala. Python, Putty, Rasbian i ROS su bazirani na programskoj podršci otvorenog koda.

.

2.7. Putty

Putty[10] je emulator terminala koji će u ovom radu biti korišten za bežično spajanje računala s Raspberry Pi uređajem.



Sl. 2.9. Putty sučelje

3. REALIZACIJA ZADATKA

Za početak rada s Raspberry Pi-om potrebno je instalirati operacijski sustav na naš Raspberry Pi. Operacijski sustav koji će biti korišten za ovaj rad je službeni operacijski sustav Raspberry Pi-a, Raspbian.

Za instalaciju Raspbiana prvo što je potrebno je formatiranje microSD kartice koja se kasnije koristi u Raspberry Pi-u. Nakon formatiranja microSD kartice na nju prebacujemo operacijski sustav Raspbian koji je preuzet sa službenih stranica Raspberry Pi zaklade.

Nakon uspješnog prebacivanja svih potrebnih podataka microSD karticu spojimo s Raspberry Pi-om. Sve što je sada potrebno je instalacija operacijskog sustava na Raspberry Pi i Raspberry Pi je spreman za rad. Sljedeći korak je spajanje na Raspberry Pi preko računala. Za to je potrebno u Raspberry Pi-u omogućiti SSH, nakon što je SSH uključen potrebno je saznati IP adresu Raspberry Pi-a. IP adresa se može saznati tako što se u terminalu upiše ifconfig i očita adresa koja je prikazana. Poznavanjem IP adrese Raspberry Pi-a možemo se povezati na njega drugim računalom preko programa Putty-a. U Putty se jednostavno upiše IP adresa Raspberry Pi-a što uspostavlja vezu između računala i Raspberry Pi-a. Nakon toga se treba upisati korisničko ime i lozinka Raspberry Pi-a, nakon što smo to obavili možemo početi s radom.

Programski kod se može pisati preko terminala pomoću urednika teksta kao što je nano, a može se napisati i na računalu i kasnije prebaciti na Raspberry Pi pomoću naredbe scp. U ovom radu kod je pisan na računalu i prebacivan s naredbom scp na Raspberry Pi.

Biblioteka RPi.GPIO dolazi ukomponirana s Raspbian operacijskim sustavom te ju nije potrebno posebno preuzimati, da bi bili sigurni da koristimo najnoviju verziju biblioteke u terminal upišemo: sudo apt-get update i u novoj liniji sudo apt-get install python-rpi.gpio python3-rpi.gpio.

3.1. MovementCommands.py

```
import RPI.GPIO as gpio

import time

class MovementCommands(object)

def __init__(self):

        gpio.setmode(gpio.BCM)

        gpio.setup(12,gpio.OUT)

        gpio.setup(13,gpio.OUT)

        gpio.setup(20,gpio.OUT)

        gpio.setup(21,gpio.OUT)

        gpio.setup(6,gpio.OUT)

        gpio.setup(26,gpio.OUT)

        self.PWMfirst=gpio.PWM(6,1000)

        self.PWMsecond=gpio.PWM(26,1000)

        self.PWMfirst.start(50)

        self.PWMsecond.start(50)

        self.stop()


def forward(self,value):

        speed=value

        self.PWMfirst.ChangeDutyCycle(speed)

        self.PWMsecond.ChangeDutyCycle(speed)

        gpio.output(12,False)

        gpio.output(13,True)
```

```
        gpio.output(20,False)
        gpio.output(21,True)

def backwards(self,value):
        speed=value
        self.PWMfirst.ChangeDutyCycle(speed)
        self.PWMsecond.ChangeDutyCycle(speed)
        gpio.output(12,True)
        gpio.output(13,False)
        gpio.output(20,True)
        gpio.output(21,False)
def stop(self):
        self.PWMfirst.ChangeDutyCycle(0)
        self.PWMsecond.ChangeDutyCycle(0)
        gpio.output(12,False)
        gpio.output(13,False)
        gpio.output(20,False)
        gpio.output(21,False)

def left(self):
        self.PWMfirst.ChangeDutyCycle(40)
        self.PWMsecond.ChangeDutyCycle(40)
        gpio.output(12,True)
        gpio.output(13,False)
```

```
        gpio.output(20,False)
        gpio.output(21,True)

def right(self):
        self.PWMfirst.ChangeDutyCycle(40)
        self.PWMsecond.ChangeDutyCycle(40)
        gpio.output(12,False)
        gpio.output(13,True)
        gpio.output(20,True)
        gpio.output(21,False)
```

Programski kod 3.1. MovementCommands

U programskom kodu 3.1. kreirana je klasa MovementCommands čije će se naredbe koristiti u drugim programskim kodovima. Metoda __init__ je konstruktor koji se prvi poziva i kodu postavlja određene iglice kao izlaze te postavlja rad motora na 1000 Hz, a radni ciklus motora postavlja na 50%.

Naredba forward postavlja brzinu rada motora na vrijednost koju mu kasnije pošaljemo u kodu te postavlja da oba motora idu naprijed.

Naredba backwards postavlja brzinu motora na vrijednost koju mu kasnije pošaljemo te postavlja da oba motora idu natrag.

Naredba stop postavlja brzinu motora na 0% te gasi motore.

Naredba left postavlja brzinu motora na 40% te stavlja da lijevi motor ide prema natrag dok desni motor ide prema naprijed što rotira Alphabot2-Pi robota u lijevo.

Naredba right postavlja brzinu motora na 40% te stavlja da lijevi motor ide prema naprijed dok desni motor ide unatrag što rotira Alhabot2-Pi robota u desno.

3.2. Infrared_dodging.py

```
import RPI.GPIO as gpio
import time
from MovementCommands import MovementCommands
DesniSenzor=16
LijeviSenzor=19
MC=MovementCommands()
gpio.setmode(gpio.BCM)
gpio.setup(DesniSenzor,gpio.IN,gpio.PUD_DOWN)
gpio.setup(LijeviSenzor,gpio.IN,gpio.PUD_DOWN)
try:
    while True:
        if(gpio.input(LijeviSenzor)==1):
            MC.right()
            MC.stop()
        elif(gpio.input(DesniSenzor)==1):
            MC.left()
            MC.stop()
        else:
            MC.forward(50)

except KeyboardInterrupt:
    gpio.cleanup()
```

Programski kod 3.2. Infrared dodging

Da bi koristili prethodno definirane metode u drugom kodu koristimo from MovementCommands import MovementCommands. Desni infracrveni senzor spojen je na iglicu 16, a lijevi na iglicu 19. Oba senzora postavimo kao ulaz te ih postavljamo kao PUD_DOWN. Koristim beskonačnu petlju koja se ponavlja sve dok se program ne prekine preko tipkovnice. Ako lijevi infracrveni senzor uoči prepreku poziva se naredba right koja okreće robota u desno te stop da ga zaustavi.

Ako desni infracrveni senzor uoči prepreku poziva se naredba left te se robot okrene u lijevo i robot se zaustavlja. To se ponavlja sve dok senzori ispred sebe uočavaju prepreke. Ako senzori ne uočavaju prepreke Alphabot2-Pi se samo kreće ravno sve dok ne uoči prepreku ili dok ga se ne zaustavi preko tipkovnice.

3.3. Ultrasonic_dodging.py

```
import RPI.GPIO as gpio
import time
from MovementCommands import MovementCommands
Trigger=22
Echo=27
MC=MovementCommands()
gpio.setmode(gpio.BCM)
gpio.setup(Trigger,gpio.OUT)
gpio.setup(Echo,gpio.IN,PUD_DOWN)
gpio.output(Trigger,False)
def Distance():
        gpio.outpup(Trigger,True)
        time.sleep(0.0001)
        gpio.output(Trigger,False)
        while gpio.input(Echo)==0:
                NoEcho=time.time()
        while gpio.input(Echo)==1:
                YesEcho=time.time()
                TravelTime=YesEcho-NoEcho
        distance=TravelTime*34300/2
        return distance
try:
                while True:
```

```
        D=Distance()
        if D<15:
                MC.left()
        else:
                MC.forward(50)
                time.sleep(0.01)

except KeyboardInterrupt:
gpio.cleanup();
```

Programski kod 3.3. Ultrasonic dodging

Kako je i u ovom programskom kodu potrebno micati robota, koristit će se prethodno definirane metode. Kako se ultrazvučni senzor sastoji od predajnika i prijamnika koristimo dvije iglice, jednu kao odašiljač i jednu kao prijamnik. Odašiljač je spojen na iglicu 22 te ga u programskom kodu nazivamo „Trigger“, a prijamnik je spojen na iglicu 27 te ga zovemo „Echo“. Odašiljač postavimo kao izlaz, a prijamnik kao ulaz te koristimo PUD_DOWN kako bi ga postavili na stalnu nulu kada ne prima signal koji odašiljač šalje. Odašiljač prvo postavljamo da ne šalje signal sve dok ga ne pozovemo u programu da odašilja. Definiramo metodu Distance koja postavlja odašiljač da šalje signal i nakon 0.1 milisekunde gasi taj signal. U while petljama provjerava se je li se poslani signal vratio, ako se poslani signal nije vratio ulazimo u prvu while petlju te pomoću „time“ naredbe zabilježavamo to vrijeme kao NoEcho, dok se poslani signal vrati ulazimo u drugu while petlju te naredbom „time“ zabilježavamo vrijeme vraćenog signala kao YesEcho. Nakon toga se ta dva vremena oduzmu te se izračuna distanca koju je kao umnožak vremena koje je bilo potrebno da zvuk putuje od Alphabot2-Pi robota do određene prepreke i natrag i brzine zvuka u centimetrima po sekundi pri normalnim uvjetima i podijeli s dva. U programu se izvodi beskonačna petlja koja ispituje udaljenost između robota i prepreke te ako je udaljenost manja od 15 centimetara robot se okreće lijevo, a ako je udaljenost veća od 15 centimetara robot putuje prema naprijed. Kao i u prošlom programu, program se zaustavlja preko tipkovnice te se nakon zaustavljanja resetiraju sve iglice na originalno stanje.

3.4. Ultrasonic_following.py

```
import RPI.GPIO as gpio
import time
from MovementCommands import MovementCommands

Trigger=22
Echo=27
MC=MovementCommands()

gpio.setmode(gpio.BCM)
gpio.setup(Trigger,gpio.OUT)
gpio.setup(Echo,gpio.IN,PUD_DOWN)
gpio.output(Trigger,False)
def Distance():
        gpio.outpup(Trigger,True)
        time.sleep(0.0001)
        gpio.output(Trigger,False)
        while gpio.input(Echo)==0:
                NoEcho=time.time()
        while gpio.input(Echo)==1:
                YesEcho=time.time()
        TravelTime=YesEcho-NoEcho
        distance=TravelTime*34300/2
        return distance
```

```
try:
        while True:
        D=Distance()
        if 90 <= D <= 99:
                MC.stop()
        elif 80 <= D <= 89:
                MC.backward(10)
        elif 70 <= D <= 79:
                MC.backward(20)
        elif 60 <= D <= 69:
                MC.backward(30)
        elif 50 <= D <= 59:
                MC.backward(40)
        elif 40 <= D <= 49:
                MC.backward(50)
        elif 30 <= D <= 39:
                MC.backward(70)
        elif 20 <= D <= 29:
                MC.backward(90)
        elif 0 <= D <= 19:
                MC.backward(100)

        elif 100 <= D <= 109:
```

```
            MC.forward(10)
        elif 110 <= D <= 119:
            MC.forward(20)
        elif 120 <= D <= 129:
            MC.forward(30)
        elif 130 <= D <= 139:
            MC.forward(40)
        elif 140 <= D <= 149:
            MC.forward(50)
        elif 150 <= D <= 159:
            MC.forward(70)
        elif 160 <= D <= 169:
            MC.forward(90)
        elif 170 <= D <= 200:
            MC.forward(100)
        else:
            MC.stop()
            time.sleep(0.01)

except KeyboardInterrupt:
gpio.cleanup();
```

Programski kod 3.4. Ultrasonic following

Programski kod 3.4. je sličan programskom kodu 3.3. te mu je jedina razlika u if else petlji. Programski kod 3.4. slijedi neki objekt koji je ispred njega i bježi od njega ako mu on prilazi. Kako je već prije spomenuto da je maksimalni domet ultrazvučnog modula HC-SRO4 četiri metra, u kod je postavljeno da radi na dva metra u slučaju da zbog vanjskih smetnji ultrazvučni modul ima smanjeno polje rada.

Ako ultrazvučni modul mjeri da je udaljenost do objekta od 90 do 99 centimetara onda se Alphabot2-Pi robot zaustavlja koristeći naredbu stop, a ako se objekt počne približavati Alphabot2-Pi robotu on se miče unatrag. Alphabot2-Pi robot se s obzirom na udaljenost od objekta miče različitom brzinom, ako je Alphabot2-Pi robot udaljen između 80 i 89 centimetara od objekta onda se on pomoću naredbe backwards(10) kreće unatrag koristeći 10 posto svoje sveukupne brzine, što je objekt bliži Alphabot2-Pi robotu on se sve brže kreće unatrag te pri udaljenosti od 0 do 19 centimetara on se kreće unatrag maksimalnom brzinom, ako se udaljenost između Alphabot2-Pi robota i objekta počne povećavati Alphabot2-Pi robot se počinje sporije udaljavati od objekta.

Ako se objekt počinje udaljavati od Alphabot2-Pi robota preko 99 centimetara, Alphabot2-Pi robot počinje slijediti određeni objekt. Ako je objekt udaljen od 100 do 109 centimetara od Alphabot2-Pi robota onda se on pomoću naredbe forwards(10) kreće naprijed koristeći 10 posto svoje sveukupne brzine, što je objekt dalje Alphabot2-Pi robotu on se sve brže kreće naprijed te pri udaljenosti od 170 do 200 centimetara on se kreće naprijed maksimalnom brzinom.

Ako se objekt udalji od Alphabot2-Pi robota na udaljenost veću od 200 centimetara Alphabot2-Pi robot se koristeći naredbu stop zaustavlja.

4.ZAKLJUČAK

U ovom završnom radu prikazana je izrada programske podrške za Alphabot2-Pi robota.
Prvo su objašnjeni Raspberry Pi i njegova važnost za robotiku te zašto je tako važan, nakon toga je objašnjen Alphabot2-Pi robot i koje njegove senzore koristim u radu. Poslije toga su objašnjeni operacijski sustavi koje koristimo u Raspberry Pi-u i programski jezik koji koristimo. Prikazan je drugi način izrade programske podrške koji se mogao koristiti, objašnjen je izraz otvoreni kod te se objasnio program koji koristim za spajanje na Raspberry Pi.
U glavnom dijelu zadatka objašnjeno je kako osposobiti Alphabot2-Pi robota za rad te je dan i objašnjen programski kod pokaznih aplikacija napravljenih za ovaj rad.
Programski kod napisan u ovom radu može se koristiti samo za upravljanje Alphabot2-Pi robotom, ali s malo preuređivanja mogao bi se koristiti i u drugim robotima, kako Alphabot2-Pi ima još senzore za praćenje linija mogla bi se napraviti i programska podrška za to, uz to se još mogu dodati nove funkcije i za korištene senzore.

LITERATURA

[1] Raspberry Pi,
https://en.wikipedia.org/wiki/Raspberry_Pi [22.6.2019]
[2] Raspberry Pi,
https://pimylifeup.com/raspberry-pi-gpio/ [22.6.2019]
[3] Alphabot2-Pi,
https://www.mouser.com/pdfdocs/Alphabot2-user-manual-en.pdf [22.6.2019]
[4] ST188 infrared sensor,
http://www.npnec.com/en_pdf/ST188-EN.pdf [24.6.2019]
[5] TB6612FNG motor driver,
https://www.instructables.com/id/Using-the-Sparkfun-Motor-Driver-1A-Dual-TB6612FNG-/
[24.6.2019]
[6] Raspbian,
https://www.raspberrypi.org/downloads/raspbian/ [22.6.2019]
[7] Python,
https://docs.python.org/3/ [5.7.2019]
[8] ROS,
http://wiki.ros.org/ [6.7.2019]
[9] Open source,
https://opensource.org/ [5.7.2019]
[10] Putty,
https://www.putty.org/ [7.7.2019]

SAŽETAK

Cilj ovog završnog rada bio je istražiti i opisati moguće načine za izradu programske podrške otvorenog koda za Alphabot2-Pi robota te izrada pokaznih aplikacija koje koriste senzore. Koristila su se dva infracrvena senzora Alphabot2-Pi robota koji su korišteni za izbjegavanje objekata ispred Alphabot2-Pi robota te je korišten ultrazvučni senzor Alphabot2-Pi robota za slijeđenje i bježanje od objekta ispred Alphabot2-Pi robota i isti ultrazvučni senzor Alphabot2-Pi robota se koristio za mjerenje udaljenosti i izbjegavanja objekta koji mu je preblizu.
Za izradu programske podrške korišten je programski jezik Python i objašnjen je način rada svakog programskog koda.
Ključne riječi: Raspberry Pi, Alphabot2-Pi, Python, otvoreni kod, robotski operacijski sustav

ABSTRACT

The goal of this final paper was to study and to describe possible ways to build open source software for the Alpabot2-Pi robot and to create software that uses his sensors.
Two infrared sensors from the Alphabot2-Pi robot are used to evade objects that are in front of the Alpabot2-Pi robot and an ultrasonic sensor from the Alpabot2-Pi robot is used to follow and escape an object that is in front of the Alpabot2-Pi robot, the same ultrasonic sensor from the Alpabot2-Pi robot is used for distance measuring to avoid an object that is too close to the Alpabot2-Pi robot . The programming laguage that was used to create the software is Python, all the created code was explained.

Keywords: Raspberry Pi, Alphabot2-Pi, Python, open source, robotic operating system

ŽIVOTOPIS

Matija Skroban rođen je 28.09.1995. u Bjelovaru. Pohađao je osnovnu školu Ivana Viteza Trnskog u Novoj Rači. Srednjoškolsko obrazovanje nastavlja u Tehničkoj školi Bjelovar te nakon završetka 2014.godine upisuje Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek kojeg trenutno pohađa.