Samobalansirajući mobilni robot

Damjanović, Davor

SVEUČILIŠTE JOSIPA JURAJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET

Preddiplomski studij računarstva

SAMOBALANSIRAJUĆI MOBILNI ROBOT

Završni rad

Davor Damjanović

Osijek, 2018.

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA I INFORMACIJSKIH TEHNOLOGIJA OSIJEK

Obrazac Z1P - Obrazac za ocjenu završnog rada na preddiplomskom sveučilišnom studiju

Osijek, 09.09.2018.

Odboru za završne i diplomske ispite

Prijedlog ocjene završnog rada

Ime i prezime studenta:	Davor Damjanović
Studij, smjer:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	4054, 19.09.2017.
OIB studenta:	88048412139
Mentor:	Doc.dr.sc. Ivan Aleksi
Sumentor:	
Sumentor iz tvrtke:	
Naslov završnog rada:	Samobalansirajući mobilni robot
Znanstvena grana rada:	**Automatizacija i robotika (zn. polje elektrotehnika)**
Predložena ocjena završnog rada:	Izvrstan (5)
Kratko obrazloženje ocjene prema Kriterijima za ocjenjivanje završnih i diplomskih radova:	Primjena znanja stečenih na fakultetu: 3 bod/boda Postignuti rezultati u odnosu na složenost zadatka: 3 bod/boda Jasnoća pismenog izražavanja: 3 bod/boda Razina samostalnosti: 3 razina
Datum prijedloga ocjene mentora:	09.09.2018.
Datum potvrde ocjene Odbora:	12.09.2018.

Potpis mentora za predaju konačne verzije rada u Studentsku službu pri završetku studija:	Potpis:
	Datum:

FERIT
FAKULTET ELEKTROTEHNIKE, RAČUNARSTVA
I INFORMACIJSKIH TEHNOLOGIJA **OSIJEK**

IZJAVA O ORIGINALNOSTI RADA

Osijek, 16.09.2018.

Ime i prezime studenta:	Davor Damjanović
Studij:	Preddiplomski sveučilišni studij Računarstvo
Mat. br. studenta, godina upisa:	4054, 19.09.2017.
Ephorus podudaranje [%]:	1

Ovom izjavom izjavljujem da je rad pod nazivom**: Samobalansirajući mobilni robot**

izrađen pod vodstvom mentora Doc.dr.sc. Ivan Aleksi

i sumentora

moj vlastiti rad i prema mom najboljem znanju ne sadrži prethodno objavljene ili neobjavljene pisane materijale drugih osoba, osim onih koji su izričito priznati navođenjem literature i drugih izvora informacija.
Izjavljujem da je intelektualni sadržaj navedenog rada proizvod mog vlastitog rada, osim u onom dijelu za koji mi je bila potrebna pomoć mentora, sumentora i drugih osoba, a što je izričito navedeno u radu.

Potpis studenta:

Sadržaj

1. UVOD

Ovaj završni rad testira koliko se dobro razvojna pločica koja radi na frekvenciji od 16MHz može ponašati poput PID regulatora u samobalansirajućem robotu. Ovaj završni rad obuhvaća realizaciju teorija iz kolegija osnove automatskog upravljanja [1] i arhitektura računala [2]. Najveći izazov s kojim se ovaj rad susreće je stabilizacija i upravljanje nestabilnog procesa koji je potaknut djelovanjem silom težom.

U radu se koristi CROduino Basic2 razvojna pločica koji je kompatibilan s Arduino razvojnim okruženjem.

1.1. Zadatak završnog rada

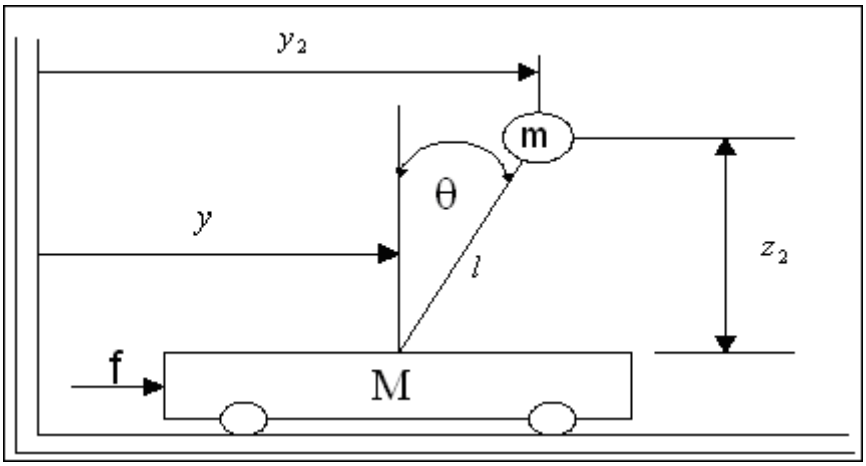
Zadatak završnog rada je napraviti samobalansirajućeg robota kojim upravlja CROduino koji se ponaša kao PID regulator i s kojim se može komunicirati putem bluetooth-a.

2. TEORIJA

Ovaj rad se temelji na teoriji obrnutog (invertiranog) njihala [3] i PID regulatora [4].

2.1. Obrnuto njihalo

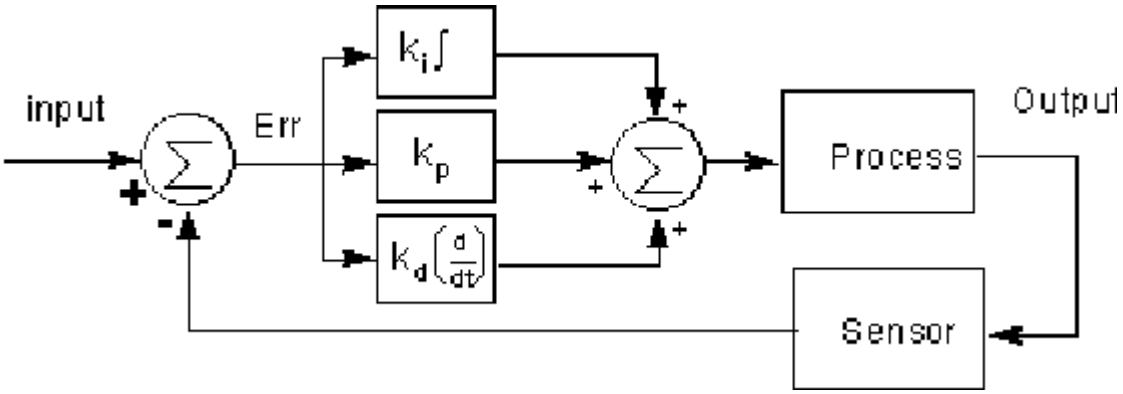
Obrnutim njihalom se smatra bilo koje tijelo kojemu je centar mase iznad centra rotacije.



Sl. 2.1. Obrnuto (invertirano) njihalo [3]

2.2. PID regulator

PID regulator se koristi za stabilizaciju i upravljanje procesima. PID regulator je regulator koji ima proporcionalno, integralno i derivabilno djelovanje. To je najkompleksniji regulator i ima najopsežnije regulacijsko djelovanje [4].



Sl. 2.2. PID regulator s povratnom vezom [5]

2.2.1 Namještanje regulacijskih parametara

Metoda koje su razvili znanstvenici Ziegler i Nichols za namještanja regulacijskih parametara prema kojoj se metoda i zove Ziegler-Nicholsova metoda. Povećavanjem odnosno smanjivanjem proporcionalnog djelovanja PID regulatora promatra se izlazni signal. Kada taj signal dođe do ruba stabilnosti odnosno ima trajne oscilacije tada računamo Kp, Ti i Td vrijednosti PID regulatora pomoću trenutnog proporcionalnog djelovanja i vremena trajanja oscilacija prema **Tab. 2.1.**

Rule Name	Tuning Parameters
Classic Ziegler-Nichols	Kp = 0.6 Ku Ti = 0.5 Tu Td = 0.125 Tu
Pessen Integral Rule	Kp = 0.7 Ku Ti = 0.4 Tu Td = 0.15 Tu
Some Overshoot	Kp = 0.33 Ku Ti = 0.5 Tu Td = 0.33 Tu
No Overshoot	Kp = 0.2 Ku Ti = 0.5 Tu Td = 0.33 Tu

Tab. 1.1. Tablica za Ziegler-Nicholsova metodu [6]

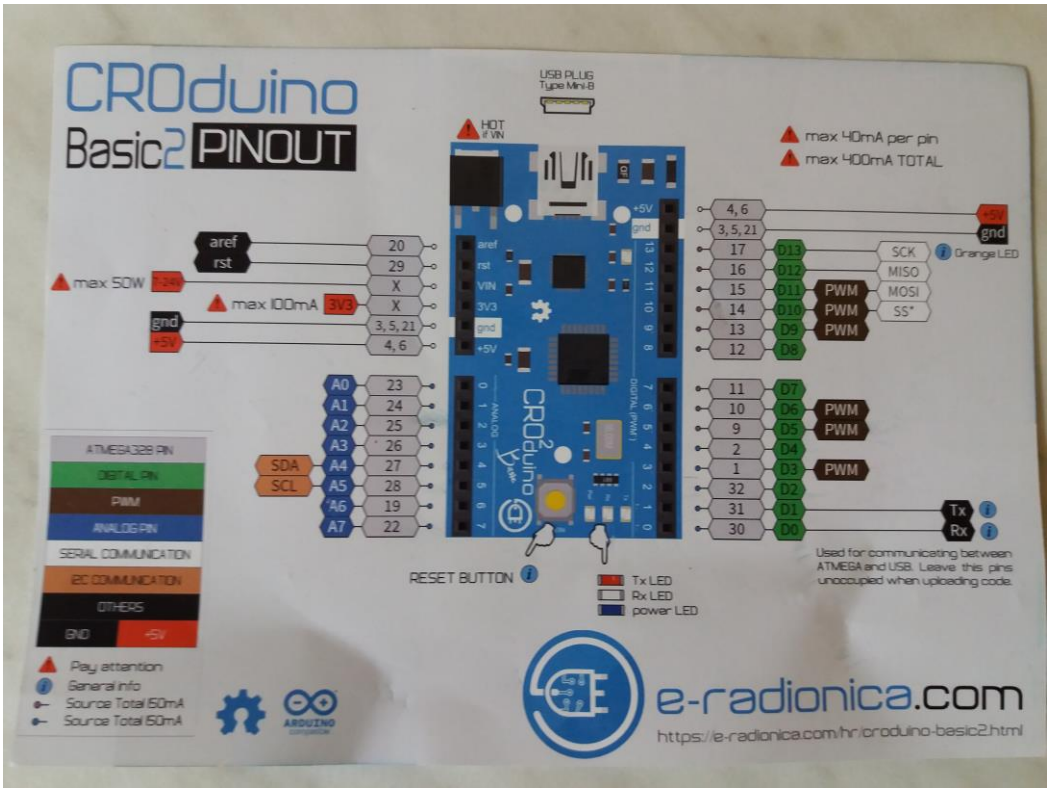
3. FIZIČKA IZVEDBA

3.1. Alati

Za izradu robota korišteni su: bušilica, boreri, odvijači, vilasti ključevi, oprema za lemljenje (lemilica, cin i pasta), skalpel, lijepilo, testera, turpija i nož.

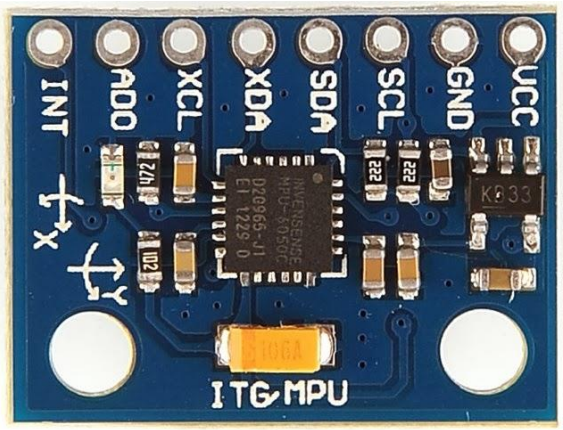
3.2. Komponente, njihova međuovisnost i drugi proizvodi korišteni u izradi

CROduino Basic2 – Arduino kompatibilna pločica s Atmega328P mikrokontrolerom izrađena po uzoru na Arduino Nano. Prima vrijednosti od žiroskopa i akcelerometra, uspoređuje ih sa zadanom vrijednošću. S razlikom između njih se ponaša poput PID regulatora i šalje signale H-mostu. Zadana vrijednost može biti izmijenjena pomoću signala s bluetooth komponente.



Sl. 3.1. CROduino Basic2 izgled i opis pinova

MPU6050 – žiroskop i akcelerometar u jednom. Žiroskop ispravlja pogrešku akcelerometra pri čemu se dobivaju bolji rezultati. Uz to ima i senzor temperature s pomoću kojega dobiva još preciznije rezultate. MPU6050 šalje CROduinu vrijednosti kuteva žiroskopa i vektora akcelerometra. Zbog pogreške prilikom izrade i pogreške prilikom montiranja mora se kalibrirati.

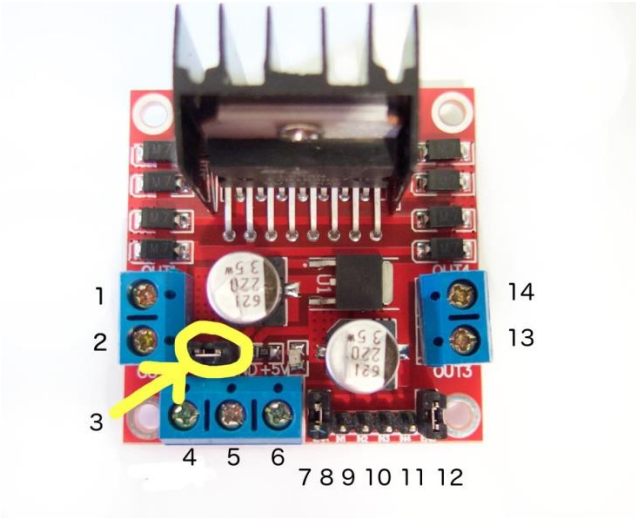


Sl. 3.2. MPU6050 žiroskop i akcelerometar [7]

HC-06 bluetooth komponenta(*eng. modul*) – komponenta za komunikaciju između mobilnog uređaja i robota. Šalje signale CROduinu za upravljanje robotom.

Sl. 3.3. HC-06 bluetooth komponenta [8]

L298N H-most - prima signale od CROduina i prema njima određuje smjer i brzinu vrtnje motora.



Sl. 3.4. L298N H-most [9]

2 DC motora 3-6v s prijenosom i kotačima – pokreću robota prema struju dobivenoj od H-mosta. Pri nagloj promjeni smjera obrnutoj od trenutne izaziva kratki spoj zbog opterećenja. Za smanjenje tog utjecaja koristi se kondenzator.

Sl. 3.5. 3-6V DC motor s prijenosom i kotačima

Lamelice – učvršćivanje motora s kutijom. Raspoređuje silu na veću površinu i time sprječava kidanje kutije

Sl. 3.6. Lamelice koje se postavljaju s vanjske strane kutije

Sl. 3.7. Lamelice koje se postavljaju s nutarnje strane kutije

M3 šarafi i matice – učvršćivanje motora, držača baterija i pretvarača za kutiju

Sl. 3.8. M3 šarafi i matice [10]

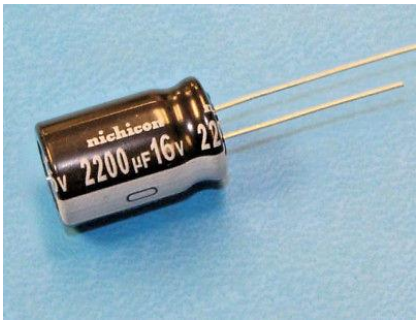
DC-DC pretvarač XL6009 (*eng. Buck-boost converter*) – održavanje napon strujnog kruga konstantnim bez obzira koliki je trenutan napon baterija. Prilagođavanje vrijednosti napona uz pomoć potenciometra.

Sl. 3.9. Pretvarač XL6009 [11]

LED voltmetar – prati trenutno stanje baterija.

Sl. 3.10. LED voltmetar [12]

2200µF 16V elektrolitski kondenzator – stabilizira napon strujnog kruga. Sprječava nagle skokove i padove vrijednosti napona izazvane promjenom smjera vrtnje motora.

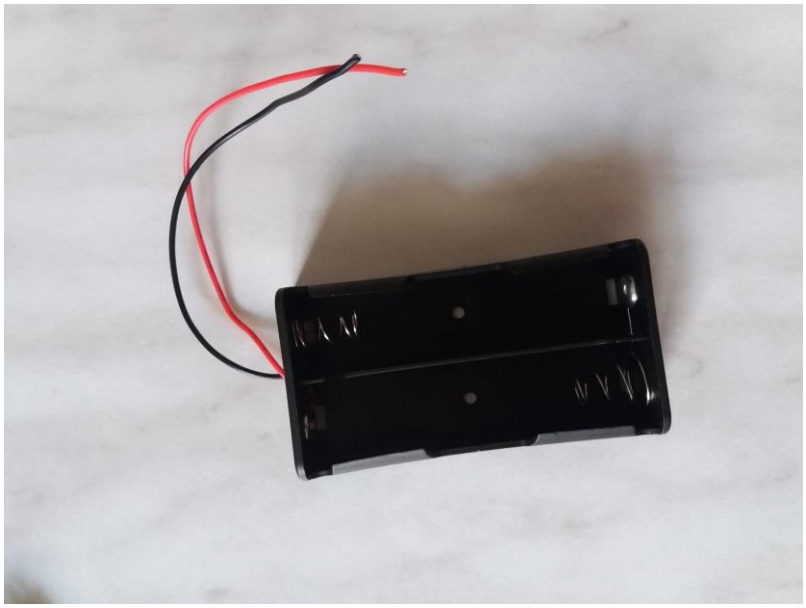


Sl. 3.11. 2200µF 16V kondenzator [13]

2 18650 Li-ion 3.7V baterije – napajanje robota.

Sl. 3.12. 18650 Li-ion 3.7V baterije

Postolje za 2 18650 baterije – drži baterije čvrsto na mjestu te tako smanjuje potencijalne oscilacije koje bi mogle proizaći zbog slobodne baterije.



Sl. 3.13. Postolje za 2 18650 baterije

Prekidač – paljenje i gašenje robota

Sl. 3.14. Prekidač [14]

Spajalice – spajanje žica, pojednostavlja pregled unutrašnjosti, grupira žice bez lemljenja, bolja estetika.

Sl. 3.15. Spajalice

M-M, M-Ž, Ž-Ž žice – spajanje električnih komponenti robota.

Sl. 3.16. M-M, M-Ž, Ž-Ž žice

Kutija – trup robota u kojemu i na kojemu se nalaze sve komponente robota. Kutija je izrađena od plastike, kartona i aluminija.

Sl. 3.17. Kutija

Crni akrilni lak – za bojanje trupa i lamelica. Koristi se zbog estetike.

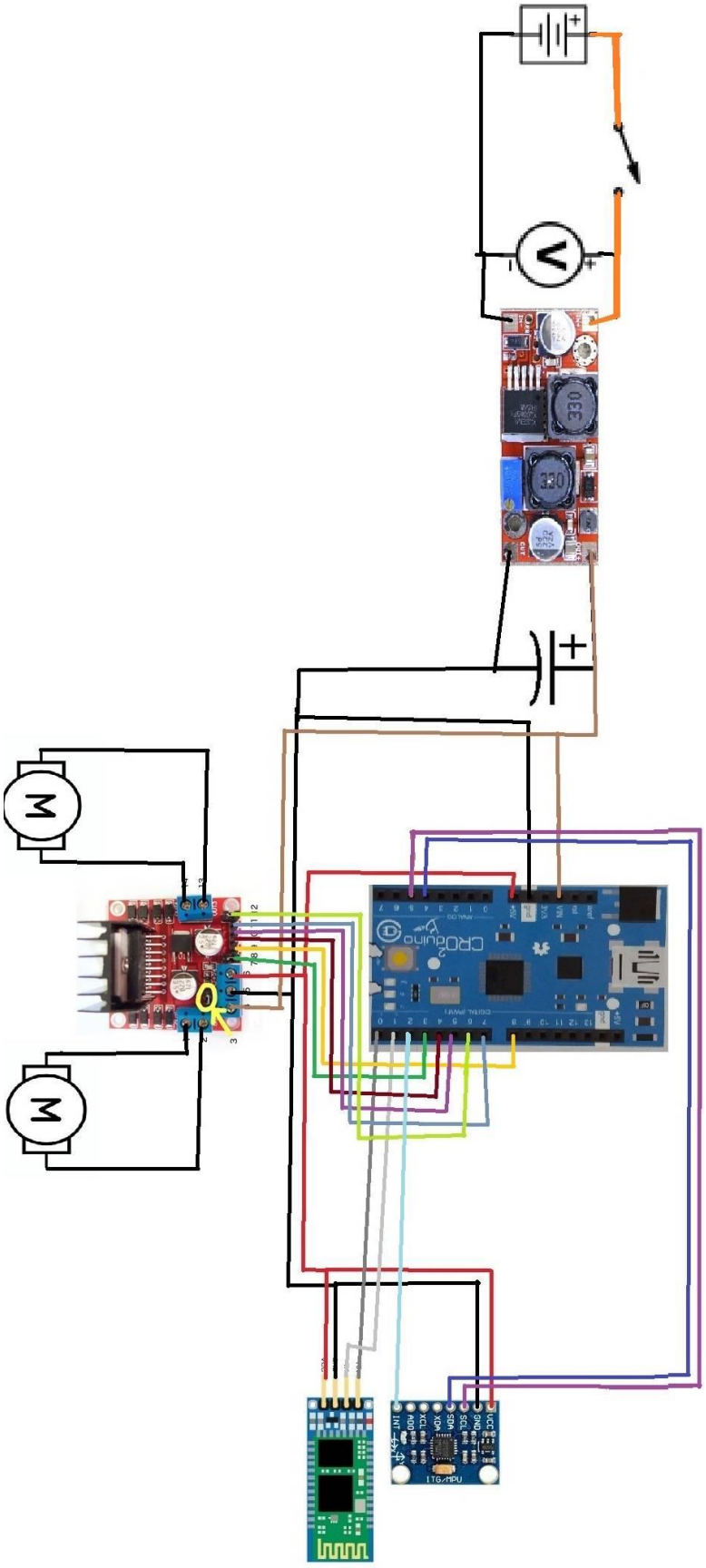
Sl. 3.18. Crni akrilni lak

Papir – za izolaciju komponenata od aluminijskih stijenki kutije.

Sl. 3.19. Papir

3.3. Shema spajanja

Pozitivan pol držača baterije spojen je na prekidač. Nakon prekidača paralelno s držačem baterija spojen je voltmetar i pozitivan i negativan pol su spojeni na ulaz DC-DC pretvarača. DC-DC pretvarač održava stalni napon od 7V. Na izlazu DC-DC pretvarača paralelno je spojen elektrolitički kondenzator. Negativan izlaz DC-DC pretvarača spojen je na GND pinove CROduina, MPU6050 žiroskopa i akcelerometra, HC-06 bluetooth modula i L298N H-mosta. Pozitivan izlaz DC-DC pretvarača spojen je na Vin pin CROduina i +12 pin L298N H-mosta. +5V pin CROduina spojena je na +5 pin L298N H-mosta i VCC pinove MPU6050 žiroskopa i akcelerometra i HC-06 bluetooth modula. SCL pin MPU6050 žiroskopa i akcelerometra spojen je na A5 pin CROduina. SDA pin MPU6050 žiroskopa i akcelerometra spojen je na A4 pin CROduina. INT pin MPU6050 žiroskopa i akcelerometra spojen je na interrupt pin (D2 pin) CROduina. TX pin HC-06 bluetooth modula spojen je na RX pin (D0 pin) CROduina. RX pin HC-06 bluetooth modula spojen je na TX pin (D1 pin) CROduina. Pinovi L298N H-mosta 7, 8, 9, 10, 11, 12 redom su spojeni na D3, D8, D4, D5, D7, D6 pinove CROduina. Desni motor je spojen na 1 i 2 pinove L298N H-mosta. Lijevi motor je spojen na 13 i 14 pinove L298N H-mosta.



Sl. 3.20. Shema spajanja robota

3.4. Izgled

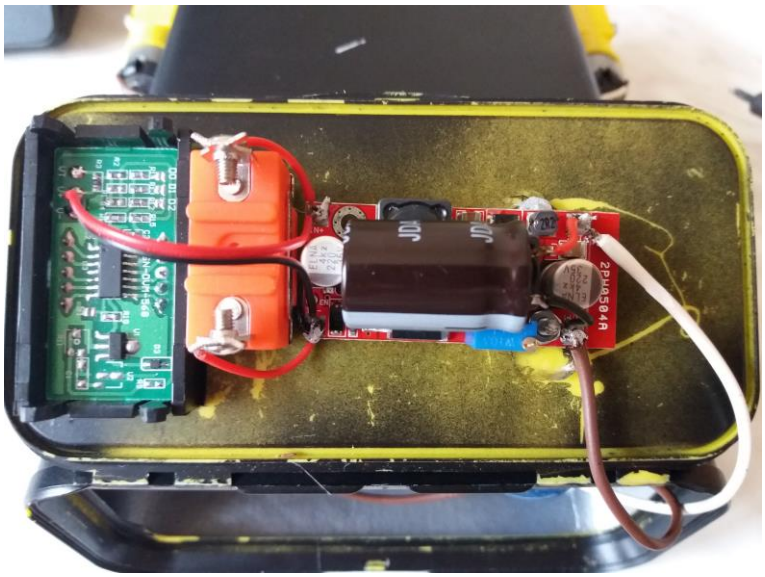
Lamelice, koje su stavljene između kutije i motora, i sama kutija su obojeni u crno akrilnom bojom.

Sl. 3.21. Montirani motor sa lamelicom na kutiju

Na vanjskoj strani poklopca kutije se nalazi voltmetar, prekidač i držač baterija s baterijama. Držač baterija je pričvršćen za kutiju uz pomoć M3 šarafa i matica. Za prekidač i voltmetar su napravljene rupe u poklopcu te su pričvršćene uz pomoć vlastitih dijelova koje sadrže.

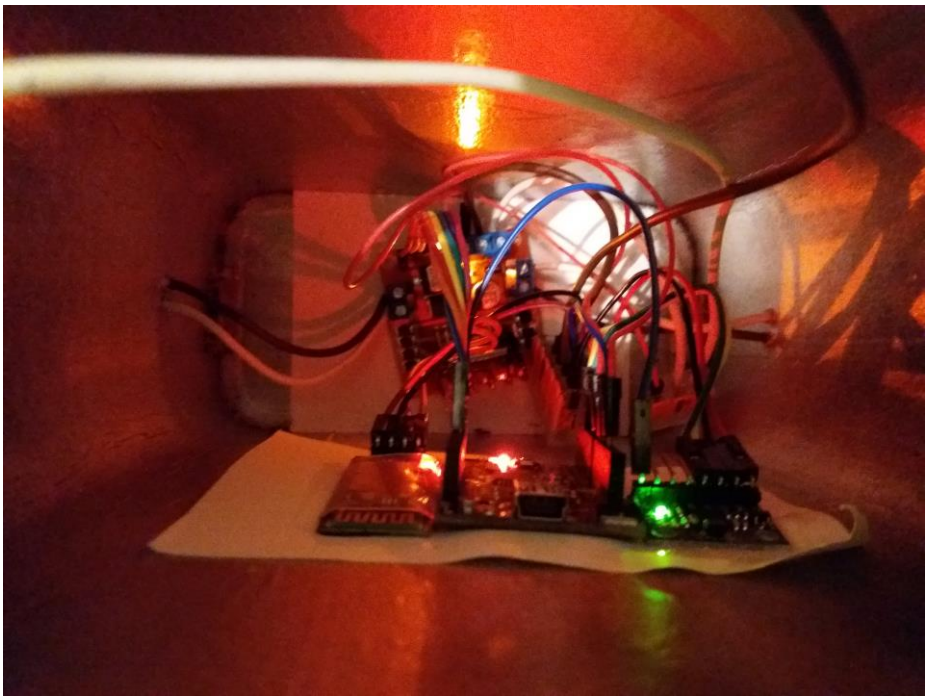
Sl. 3.22. Vanjska strana poklopca kutije

Na unutarnjoj strani poklopca se nalazi DC-DC pretvarač ,učvršćen na kutiju s pomoću M3 šarafa i matica, i kondenzatorom učvršćen ljepilom za DC-DC pretvarač.



Sl. 3.23. Unutarnja strana poklopca kutije

Unutar kutije se nalazi ostatak strujnog kruga. Papir je korišten kao izolacija od aluminijske unutrašnjosti kutije. Papir i sve komponente su zalijepljene ljepilom. Komponente su spojene spajalicama i žicama.



Sl. 3.24. Unutrašnjost kutije

Izgled cijelog robota

Sl. 3.25. Izgled cijelog robota odozgor

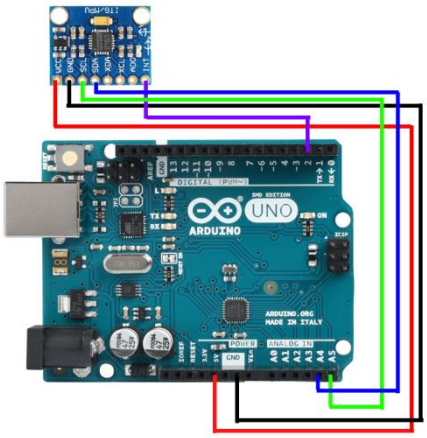
Sl. 3.26. Izgled cijelog robota sprijeda

4. ARDUINO KOD

Za izradi rada korištena su 2 programa Arduino IDE [15] i Processing [16]. Processing je korišten samo za provjeru kalibracije žiroskopa. Arduino IDE je razvojno okruženje s pomoću kojega možemo pisati Arduino kod, prenositi ga na Arduino kompatibilne uređaje i komunicirati s tim uređajima. Arduino kod je jako sličan C++ programskom jeziku.

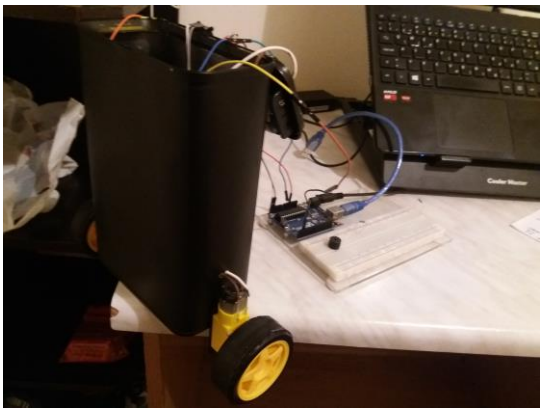
4.1 Kalibracija žiroskopa i akcelerometra

Kalibracija žiroskopa i akcelerometra je potrebna jer izrada pločica uvijek ima pogrešku koja se događa prilikom stavljana komponenata i lemljenja istih na nju te također prilikom montiranja žiroskopa na željeno mjesto. Bez kalibracije žiroskop bi tijekom vremena lagano mijenjao vrijednost. Što više vremena prođe te iste vrijednosti postaju sve nepreciznije. Kalibracijom žiroskopa se potpuno ne uklanja problem, ali se znatno smanjuje promjena vrijednosti zbog čega se dobivene vrijednosti duže vremena mogu upotrebljavati. Za kalibraciju žiroskopa se koristi IMU_Zero program iz MPU6050 biblioteka za Arduino IDE [17].



Sl. 4.1. Shema spajanja MPU6050 s Arduino pločicom [18]

Robot je postavljen uspravno dok se MPU6050 kalibrira. Nakon kalibracije na Arduino IDE serial monitoru se dobiju rezultati i pogreške(*eng. offsets*). Za ispravljanje pogreške se promatra zadnje očitavanje. Pogreške se oduzimaju od očitane vrijednosti unutar glavnog koda te tako smanjuju pogrešku tijekom vremena i nuliraju trenutnu poziciju.



Sl. 4.2. Robot postavljen na rub stola preko trupa s ciljem kalibracije

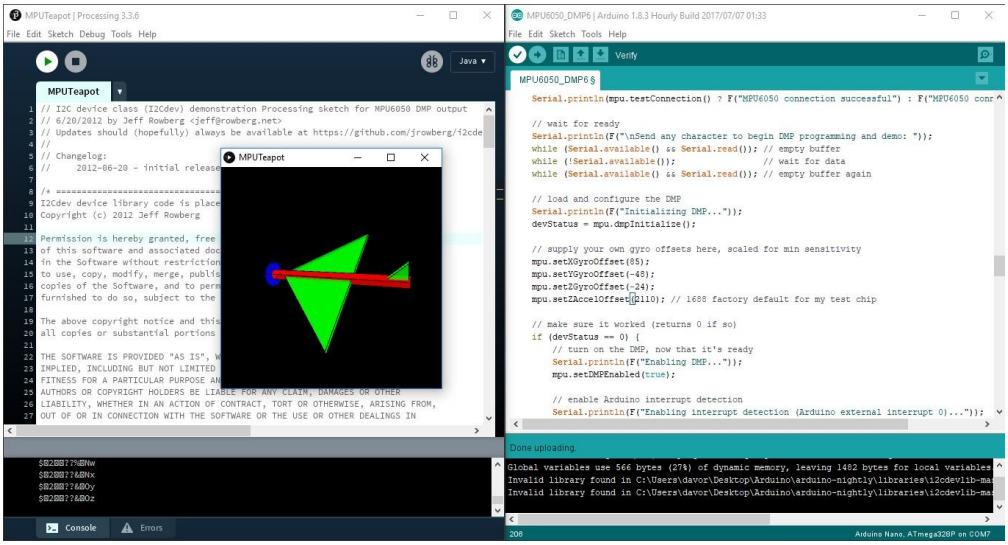
```
...
...

FINISHED!

Sensor readings with offsets:   8       8       16387   -1      1       -1
Your offsets:   -3840   -2278   2110    85      -48     -24
```

Sl. 4.3. Rezultati kalibracije u Arduino IDE serial monitoru

Za provjeru uspješnosti kalibracije na Arduino pločicu se postavi kod MPU6050_DMP6 s unesenim pogreškama dobivenima kalibracijom i sklone se 2 kose crte ispred „#define OUTPUT_TEAPOT“ a postave na „#define OUTPUT_READABLE_EULER“ te se otvori program processing s MPUTeapot. Problem s „READABLE_EULER“ je taj što vrijednosti vrlo brzo krene gubiti točnost pa se za provjeru najčešće koristi Teapot ili „READABLE_YAWPITCHROLL“ gdje se za trenutne vrijednosti dobiju kutevi u stupnjevima. [17]



Sl. 4.4. MPU6050_DMP6 s unesenim pogreškama i MPUTeapot s 3D prikazom žiroskopa

4.2 Glavni program

U glavnom programu CROduino prima trenutni kut od žiroskopa i akcelerometra i podatak od bluetooth modula. Nakon toga prvo se izvršava naredba primljena bluetooth modulom nakon toga drugo se izvršava računanje koliko trenutna vrijednost kuta odskače od zadane te tada implementirani PID regulator računa smjer i brzinu u kojima se motori trebaju kretati da kompenziraju pogrešku kuta.

4.2.1 Biblioteke

U izradi Arduino koda korišteno je 5 biblioteka.

```
#include <PID_v1.h>
#include <LMotorController.h>
#include "I2Cdev.h"
#include "MPU6050_6Axis_MotionApps20.h"

#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
#include "Wire.h"
#endif

#define MIN_ABS_SPEED 30
```

Sl. 4.5. Biblioteke korištene u Arduino kodu

Biblioteka PIDv1.h [19] omogućava vrlo jednostavnu implementaciju PID regulatora.
Biblioteka LMotorController.h [20] omogućava jednostavno upravljanje H-mostom.
Biblioteka I2Cdev.h [21] i Wire [22] omogućavaju komunikaciju sa žiroskopom i akcelerometrom.
Biblioteka MPU6050_6Axis_motionApps20.h omogućava lakši pristup podacima sa žiroskopa i akcelerometra.

4.2.2 Globalne funkcije i varijable

MPU6050

Za žiroskop i akcelerometar potrebne su varijable vezane za trenutni status MPU6050, veličinu paketa, trenutan broj podataka u nizu i status interrupta, varijable za vektore i kuteve i funkciju koja mijenja vrijednost varijable kada se dogodi interrupt. Žiroskop i akcelerometar predstavlja klasu mpu tipa MPU6050 dobivenu iz biblioteke MPU6050_6Axis_motionApps20.h preko koje se lakše pristupa podacima dobivenima od žiroskopa i akcelerometra.

```
MPU6050 mpu;

uint8_t mpuIntStatus;
uint8_t devStatus;
uint16_t packetSize;
uint16_t fifoCount;
uint8_t fifoBuffer[64];

//vektori i položaji
Quaternion q;
VectorFloat gravity;
float kutovi[3]; // samo kutovi[1] potreban
double MPUkut; //kut dobiven žiroskopom i pretvoren u stupnjeve

//funkcija koja reagira na interrupt
volatile bool mpuInterrupt = false;
void dmpDataReady(){
  mpuInterrupt = true;
}
```

Sl. 4.6. Izgled globalnih varijabli, funkcija i klasa za MPU6050

PID regulator

Za PID regulator potrebne su varijable položaja oko kojeg sustav titra, vrijednosti Kp, Ki i Kd regulatora, ulazna vrijednost dobivena od žiroskopa i akcelerometra i vrijednost koju daje PID regulator nakon računanja. Sve navedene varijable se stavljaju u klasu pid tipa PID dobivenu iz biblioteke PIDv1.h.

```
//PID
double nulaSetpoint = 227.37; //nula,početni položaj, mjesto oko kojeg sustav titra
double setpoint = nulaSetpoint;
double s_i_b_motora;// -255 do 255 za motore određuje smjer i brzinu

//vrijednosti PID regulatora
double Kp = 60;  //80
double Ki = 600; //500
double Kd =2 ;//2.25

PID pid(&MPUkut, &s_i_b_motora, &setpoint, Kp, Ki, Kd, DIRECT);
```

Sl. 4.7. Izgled globalnih varijabli i klasa za PID regulator

H-most

Za H-most potrebne su dvije varijable za brzina motora, 2 PMW (*eng. Pulse-width modulation*) pina za brzinu motora, 4 pina za smjer kretanja motora i klasa H_most tipa LMotorController iz biblioteke LMotorController.h u koju ide sve navedene varijable te koja pojednostavlja upravljanje H-mostom.

```
//H-most
double BrzinaA = 0.9;
double BrzinaB = 0.9;
int MA = 3;
int MA1 = 4;
int MA2 = 8;
int MB1 = 5;
int MB2 = 7;
int MB = 6;
LMotorController H_most(MB, MB1, MB2, MA, MA1, MA2, BrzinaB, BrzinaA);
```

Sl. 4.8. Izgled globalnih varijabli i klasa za H- most

Komunikacija

Za komunikaciju su potrebne varijable stanja koje određuju buduću radnju robota.

```
//za komunikaciju
int inByte;
int stari_inByte='S';
```

Sl. 4.9. Izgled globalnih varijabli za komunikaciju

4.2.3 Setup kod

Na početku setup-a pokreće se I2C komunikacija na 400kHZ. Pokreće se MPU6050 i provjerava ima li pogrešaka u pokretanju. Unose se pogreške žiroskopa i akcelerometra dobivene kalibracijom. Ako nije bilo pogrešaka prilikom pokretanja MPU6050 uključuje se reagiranje na interrupt i provjerava se stanje MPU6050.

```
void setup(){
  // priključivanje I2C bus-u
  #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
  Wire.begin(); // pokretanje I2C komunikacije
  TWBR = 24;    // 400kHz I2C sat
  #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE
  Fastwire::setup(400, true);
  #endif

  mpu.initialize();
  devStatus = mpu.dmpInitialize();

  // pogreške (unos pogrešaka dobivenih kalibracijom)
  mpu.setXGyroOffset(220);
  mpu.setYGyroOffset(76);
  mpu.setZGyroOffset(-85);
  mpu.setZAccelOffset(1788);

  if (devStatus == 0){// ako nema errora pokreni MPU
    mpu.setDMPEnabled(true);

    attachInterrupt(0, dmpDataReady, RISING);// uključi reagiranje na interrupt
    mpuIntStatus = mpu.getIntStatus();
    packetSize = mpu.dmpGetFIFOPacketSize();
```

Sl. 4.10. Prvi dio setup koda

Postavljaju se uzorkovanje i ograničenja PID regulatora te se pokreće serijska komunikacija koja će se odvijati putem bluetooth-a.

```
    //postavke PID regulatora
    pid.SetMode(AUTOMATIC);
    pid.SetSampleTime(10);
    pid.SetOutputLimits(-255, 255); //max i min vrijednost koju PID može dati na izazu
  }

  Serial.begin(9600); //pokrenuta komunikacija (za bluetooth)
}
```

Sl. 4.11. Drugi dio setup koda

4.2.4. Loop kod

U dijelu koda koji se ponavlja događaju se dvije glavne radnje. Prvo se provjerava serijska veza, odnosno ima li naredbi koje su došle putem bluetooth-a. Ako ih nema vrši se stara radnja jer se može dogoditi da program nekoliko puta provrti kod koji se ponavlja bez da primi novu naredbu.

```
void loop(){

  if (Serial.available() > 0) {
    inByte = Serial.read();
    stari_inByte=inByte;}
  else {inByte=stari_inByte;}
   switch (inByte){
          case 'F':
                    setpoint = nulaSetpoint + 2;
                    break;
          case 'B':
                    setpoint = nulaSetpoint - 2;
                    break;
          case 'R':
                    setpoint = nulaSetpoint;
                    H_most.turnRight(150,0);
                    break;
          case 'L':
                    setpoint = nulaSetpoint;
                    H_most.turnLeft(150,0);
                    break;

          default:                              //'S'
                    setpoint = nulaSetpoint;

     }
```

Sl.4.12. Dio loop koda odgovornog za serijsku komunikaciju

Nakon provjere serijske komunikacije program čeka da se dogodi interrupt i da se napuni FIFO niz do veličine paketa u kojem se nalaze očitavanja MPU6050, a u međuvremenu PID regulator računa smjer i brzinu vrtnje motora te ih prosljeđuje dijelu programa odgovornom za H-most. Kada se napuni FIFO niz i dogodi interrupt program provjerava se status mpu6050 uzima se broj podataka u FIFO nizu te se varijabla za interrupt vraća na nulu.

```
while (!mpuInterrupt && fifoCount < packetSize){ //dok se ne dobije MPU paket podataka PID regulator računa i
  pid.Compute();                                 //šalje H-most brzinu i smjer kretanja koje je PID regulator izračunao
  H_most.move(s_i_b_motora, MIN_ABS_SPEED);
}

mpuInterrupt = false;
mpuIntStatus = mpu.getIntStatus();
fifoCount = mpu.getFIFOCount();
```

Sl.4.13. Dio loop koda za rad PID regulatora i H-mosta

Nakon rada PID regulatora i H-mosta provjerava se mpu6050 interrupt status i broj podataka u FIFO nizu. Ako postoji interrupt s kodom 0x10 ili se FIFO niz napuni do 1024 FIFO niz se prazni. Inače program ako je interrupt s kodom 0x02 uzima se trenutan broj podataka u FIFO nizu i čitaju se podaci iz njega. Na kraju se kut pretvara u stupnjeve radi lakšeg računanja. [23]

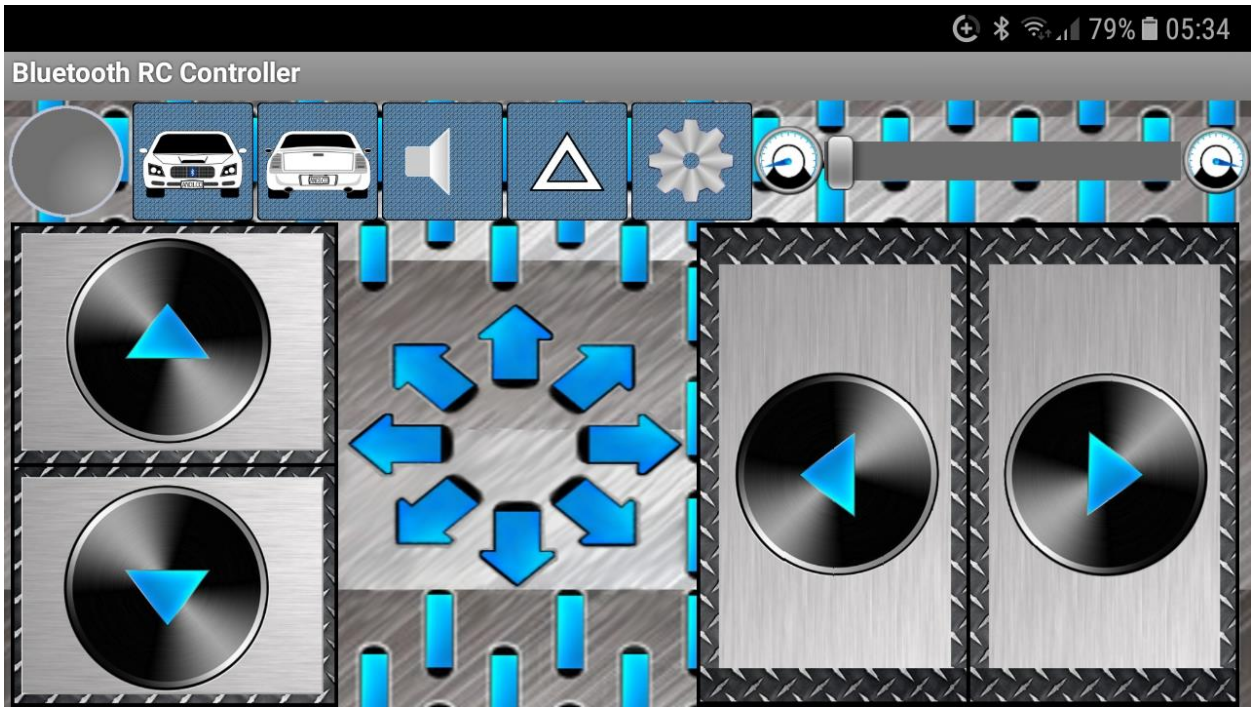
```
  if ((mpuIntStatus & 0x10) || fifoCount == 1024){ // ako je FIFO ni pun isprazni ga
    mpu.resetFIFO();
  }
  else if (mpuIntStatus & 0x02){ //provjera za interrupt
    while (fifoCount < packetSize) fifoCount = mpu.getFIFOCount();
    mpu.getFIFOBytes(fifoBuffer, packetSize);
    fifoCount -= packetSize;

    //čita podatke dobivene od MPU
    mpu.dmpGetQuaternion(&q, fifoBuffer);
    mpu.dmpGetGravity(&gravity, &q);
    mpu.dmpGetYawPitchRoll(kutovi, &q, &gravity);
    MPUkut = kutovi[1] * 180/M_PI + 180;
  }
}
```

Sl.4.14. Dio loop koda za provjeru interrupta i računanje kuta

5. MOBILNA APLIKACIJA

Za upravljanje robotom putem bluetooth-a koristi se aplikacija Arduino Bluetooth RC Car [24]. Aplikacija radi na 9600 bauda i svaki puta pošalje jedan znak. U slučaju ovog robota potrebni znakovi su: 'F', 'B', 'R', 'L' i 'S'. F na naprijed, B za nazad, R za desno, L za lijevo i S da se ne miče s mjesta.



Sl. 5.1 Sučelje Arduino Bluetooth RC Car aplikacije

6. ZAKLJUČAK

U ovom završnom radu izrađen je samobalansirajući mobilni robot. Prvo je osmišljena shema i princip po kojemu je robot trebao raditi. Nakon toga izrađen je fizički dio robota. Potom je napisan kod s kojim se provjerilo rad i smjer vrtnje motora. Sljedeći korak bila je kalibracija žiroskopa i akcelerometra. Nakon kalibracije žiroskopa i akcelerometra napisan je kod za samobalansiranje robota. Nakon što je kod napisan prvo se morala odrediti točka u kojoj se robot drži uspravno od tla te neposredno tijekom toga su se naređivale vrijednosti PID regulatora. Tijekom naređivanja vrijednosti PID regulatora Ziegler-Nicholsova metoda se nije pokazala uspješnom zato što su kotači imali praznog hoda i od nagli trzaja su proklizavali. Parametri PID regulatora su potom odabrani „po osjećaju“ što je dalo puno bolje rezultate od Ziegler-Nicholsova metode. Zadnje što se se napisalo je implementacija bluetooth komunikacije što je bio vrlo lagan zadatak.

LITERATURA

[1] Osnove automatskog upravljanja: https://www.ferit.unios.hr/studiji/sveucilisni-preddiplomski-studij/PER501/1

[2] Arhitektura računala: https://www.ferit.unios.hr/studiji/sveucilisni-preddiplomski-studij/PRK503/2

[3] Obrnuto njihalo: http://laris.fesb.hr/digitalno_vodjenje/primjer_4.htm

[4] PID regulator: http://www.unidu.hr/datoteke/majelic/ABP-18.pdf

[5] PID regulator: https://www.researchgate.net/figure/A-PID-regulator_fig1_255591999

[6] Tablica za Ziegler-Nicholsova metodu: http://www.mstarlabs.com/control/znrule.html

[7] MPU6050 žiroskop i akcelerometar: https://nexiot.com/product/triple-axis-accelerometer-gyro-breakout-mpu-6050/

[8] HC-06 bluetooth komponenta: https://www.dx.com/p/hc-06-serial-port-passthrough-wireless-slave-transceiver-bluetooth-module-for-arduino-382686#.W46nSuj7TIU

[9] L298N H-most: https://tronixlabs.com.au/robotics/motor-controllers/l298n-dual-motor-controller-module-2a-australia/

[10] M3 šarafi i matice: http://www.hobbytronics.co.uk/nut-bolt-m3-12

[11] Pretvarač XL6009: https://www.ebay.com/itm/Boost-Buck-DC-adjustable-step-up-down-Converter-XL6009-Module-Voltage-NEW/191673952440?epid=592613290&hash=item2ca0a868b8:g:0CAAAOSwLVZVs4ch

[12] LED voltmetar: https://www.ebay.com/itm/Mini-DC-5-30V-Voltmeter-LED-Panel-3-Digital-Display-Voltage-Meter-2-wire-Hot-XG/282579189492?hash=item41cb083ef4%3Am%3AmmIpiqL9cRHZUN5X_pNFKhA&var=581751537209

[13] 2200µF 16V kondenzator: https://picclick.com/6pcs-Nichicon-VZ-2200uF-16v-105c-Radial-Electrolytic-122139455139.html

[14] Prekidač: https://www.conrad.hr/Prevjesni-prekida%E8-250-V%2FAC-3-A-1-x-isklop%2Fuklop-SCI-MTE101A1-zaporni-1-kom..htm?websale8=conrad-hr&pi=1304031

[15] Arduino IDE: https://www.arduino.cc/

[16]Processing: https://processing.org/

[17] MPU6050 biblioteka: https://github.com/jrowberg/i2cdevlib/tree/master/Arduino/MPU6050

[18] Shema spajanja MPU6050 s Arduino pločicom: https://arduino.stackexchange.com/questions/44937/how-important-is-the-int-pin-of-the-mpu6050-gy-521

[19] PIDv1.h biblioteka: https://github.com/br3ttb/Arduino-PID-Library/blob/master/PID_v1.h

[20] LMotorController.h biblioteka: https://github.com/lukagabric/Franko/blob/master/libraries/LMotorController/LMotorController.h

[21] I2Cdev.h biblioteka: https://github.com/jrowberg/i2cdevlib

[22] Wire biblioteka: https://www.arduino.cc/en/Reference/Wire

[23] MPU6050 registri: https://www.invensense.com/wp-content/uploads/2015/02/MPU-6000-Register-Map1.pdf

[24] Arduino Bluetooth RC Car: https://play.google.com/store/apps/details?id=braulio.calle.bluetoothRCcontroller&hl=hr

SAŽETAK

Naslov: Samobalansirajući mobilni robot

U ovom završnom radu izrađen je samobalansirajući mobilni robot. Izrađena je na temelju mikroupravljačkog sustava CROduino.Preko I2C komunikacije i bluetooth-a CROduino prima podatke obrađuje ih i šalje H-mostu koji pokreče motore kako mu je naređeno. CROduino se ponaša poput PID regulatora te tako bez velikih oscilacija održava poziciju.

Ključne riječi: Arduino, CROduino, Samobalansirajuće, Robot, MPU6050, Žiroskop, Akcelerometar, L298N, H-most, Bluetooth, Motori, PID regulator

ABSTRACT

Title: Self-balancing mobile robot

In this final work self-balancing mobile robot was created. The design is based on CROduino microcontroller system. Through I2C communication and bluetooth, CROduino collects data, processes them and sends them to H-bridge which moves motors accordingly. CROduino behaves like a PID regulator and without large oscillations, maintains its position.

Keywords: Arduino, CROduino, Self-balancing, Robot, MPU6050, Gyro, Accelerometer, L298N, H-bridge, Bluetooth, Motors, PID controller

ŽIVOTOPIS

Davor Damjanović rođen je dana 17.09.1996. godine U Vinkovcima. Od rođenja živi u selu Strizivojna koje se nalazi u okolici grada Đakova. Osnovnu školu pohađa u OŠ „Ivana Brlić-Mažuranić“ u Strizivojni te ju završava s odličnim uspjehom. Obrazovanje nastavlja upisom u gimnaziju „Matija Mesić“ u Slavonskom Brodu te pohađa prirodoslovno matematički smjer. 2015. godine upisuje se na Elektrotehnički fakultet Osijek na preddiplomski studij elektrotehnike. 2016. godine prebacuje se na 2. godinu preddiplomskog studija računarstva.