SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 5545

Samobalansirajući mobilni robot

Sebastian Dovičin

Zagreb, lipanj 2018.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA
ODBOR ZA ZAVRŠNI RAD MODULA

Zagreb, 14. ožujka 2018.

ZAVRŠNI ZADATAK br. 5545

Pristupnik: **Sebastian Dovičin (0036483438)**
Studij: Računarstvo
Modul: Računalno inženjerstvo

Zadatak: **Samobalansirajući mobilni robot**

Opis zadatka:

Samobalasirajući mobilni robot sastoji se od dva motora s kotačima sa zajedničkom osi rotacije. Konstrukcija ovakvog robota je nestabilna i teži prevrtanju. Potencijalno prevrtanje detektira se senzorima, a položaj robota ispravlja se upravljanjem motorima u odgovarajućem smjeru.
U okviru završnog rada razvit će se model samobalansirajućeg robota koristeći Raspberry Pi. Robot mora imati vlastito napajanje i biti mobilan. Kao pogon koristiti dva motora i pripadajuću hardversku upravljačku pločicu. Koristiti akcelerometar, žiroskop i druge senzore po potrebi. Osim održavanja ravnoteže, robot mora imati i sposobnost kretanja prostorom.

Zadatak uručen pristupniku: 16. ožujka 2018.
Rok za predaju rada: 15. lipnja 2018.

Mentor:

Doc. dr. sc. Ana Sović Kržić

Djelovođa:

Prof. dr. sc. Danko Basch

Predsjednik odbora za završni rad modula:

Prof. dr. sc. Mario Kovač

SADRŽAJ

1. Uvod

Kako i samo ime govori, samobalansirajući mobilni robot je robot koji ima sposobnost samostalnog održavanja ravnoteže te kretanja po prostoru. Robot ima dodir s podlogom na dva široko postavljena kotača sa zajedničkom osi rotacije i težište iznad osi rotacije, zbog čega je robot nestabilan i teži prevrtanju po jednom stupnju slobode. Za omogućavanje samostalnog održavanja ravnoteže robota te njegova kretanja prostorom, potrebno je ili dodati treći kotač izvan zajedničke osi rotacije ili složiti mehanizam za samostalno održavanje ravnoteže. U ovom završnom radu obrađuje se varijanta s mehanizmom za samostalno održavanje ravnoteže.

Obzirom da robot teži prevrtanju po jednom stupnju slobode i težište mu je iznad osi rotacije kotača, problem samobalansiranja svodi se na problem invertiranog njihala. Invertirano njihalo jedno je od složenijih i popularnijih inženjerskih problema. Često se koristi kao primjer za realizaciju stabilizacije nestabilnih sustava i može se primijeniti u mnogo praktičnih primjera u mehanici i drugim granama [21] [24].

Primjer samobalansirajućeg robota nije jedinstven. Postoje mnoge slične realizacije, a najčešće uključuju male dimenzije robota, mikrokontroler za upravljanje, jedinica za mjerenje inercije za kontrolu kretanja te istosmjerne motore za pogon robota.

Popularno vozilo "Segway" (Slika 1.1) također je jedna od varijanti samobalansirajućeg robota [7].

Slika 1.1: Segway

Tvrtka "Boston Dynamics" dizajnirala je robot zvan "Handle" (Slika 1.2), što je robot ljudskih dimenzija koji se kreće na dva kotača uz pomoć samobalansiranja i može prenositi teške predmete po grubim terenima [2].

Slika 1.2: Handle robot

2. Dijelovi robota

Da bi robot imao sposobnost samostalnog kretanja i samobalansiranja, potrebne su iduće komponente:

1. Komponente za mobilizaciju robota
2. Jedinica za mjerenje inercije
3. Upravljačka jedinica
4. Izvor napajanja

2.1. Komponente za mobilizaciju robota

Glavna funkcionalnost samobalansirajućeg robota je održavanje ravnoteže u uspravnom položaju, stoga mu trebaju pokretni dijelovi za održavanje navedene ravnoteže. Najčešće korišteni način za to jesu kotači koje pogoni jedan ili više motora.

2.1.1. Motori - vrste, opis

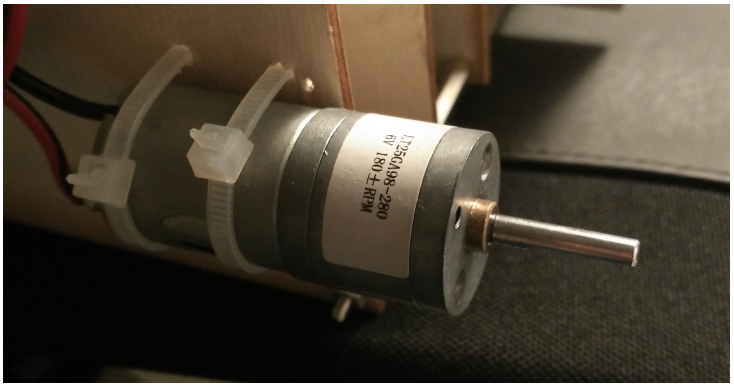
Postoji nekoliko vrsta motora, od kojih su, za robote, najčešće korišteni:

- istosmjerni motori
- koračni motori
- servo motori

Istosmjerni motor

Istosmjerni motor (Slika 2.1) je uređaj koji pretvara istosmjernu struju u mehaničku energiju [25]. Radi na principu interakcije magnetnog i električnog polja. Kada se električni vodič postavi u magnetno polje, vodič ima namjeru pomaknuti se. Navedenom interakcijom magnetnog i električnog polja dolazi do nastanka mehaničkih sila

koje pokreću rotor, na koji je pričvršćena vanjska osovina [3]. Motoru se na osovinu može priključiti i reduktor, odnosno set zupčanika koji smanjuje brzinu vrtnje same osovine u nekom omjeru i time povećava snagu samog motora [19].



Slika 2.1: Primjer istosmjernog motora

Koračni motor

Koračni motor je istosmjerni motor koji se okreće u koracima. Svaki korak vrši se uključivanjem ili isključivanjem određenog elektromagneta ili zavojnice, tzv. faze, koji pomakne rotor za određeni dio kruga. Naizmjeničnim uključivanjem ili isključivanjem elektromagneta dolazi do okretanja rotora, odnosno vanjske osovine. Koračni motor odlikuje veliki okretni moment pri niskim okretajima [16].

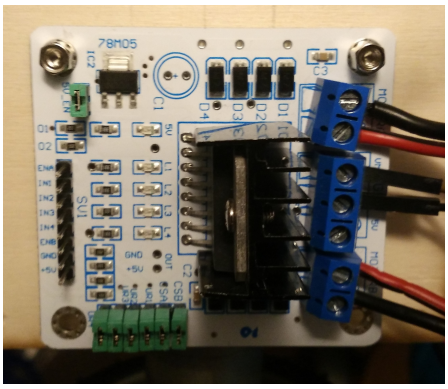
Servo motor

Servo motor je istosmjerni ili izmjenični motor sa senzorom za položaj, pomoću kojeg određuje u kojem se položaju motor trenutno nalazi i u kojem bi trebao biti. Motoru se na ulaz postavi impuls koji označava odredišni položaj, a motor sam odradi traženi pomak u traženom smjeru. Servo motori obično imaju slobodu kretanja u području od najviše 180 stupnjeva [8], zbog čega nisu pogodni za korištenje u svrhu kretanja robota u prostoru.

2.1.2. Upravljanje motorima

Zbog velikog opsega različitosti motora te ograničenja izlaza upravljačkih jedinica, pri čemu se misli na ograničenja u naponu i struji, motori se vrlo rijetko spajaju na upravljačku jedinicu direktno. Umjesto toga, koriste se međusklopovi, tzv. *driveri*, koji na ulazu primaju signale upravljačke jedinice i pretvaraju ih u željeni izlaz prema

motorima. Ovisno o željenoj brzini upravljanja i točnosti, driveri mogu komunicirati s upravljačkom jedinicom omogućavanjem jedinici direktne kontrole motora ili pomoću komunikacijskih protokola od kojih je najčešći I2C, pri čemu se o prijenosu signala prema motorima brine sam driver. Time driver postaje tzv. *kontrolerom* [23]. Primjer kontrolera prikazan je na slici 2.2. U nastavku je opisano upravljanje za svaku vrstu motora zasebno.



Slika 2.2: Motor kontroler sa L298N motor driverom

Upravljanje istosmjernim motorima

Za upravljanje istosmjernim motorima potrebna su dva ulaza za smjer te varijabilni napon ulaza za brzinu vrtnje motora. Driveri su uglavnom napravljeni kao pojačala ulaznih signala koji se direktno prenose na ulaze motora. Obzirom da upravljačke jedinice često nemaju analogne izlaze nego samo digitalne, brzina vrtnje motora nerijetko se regulira pulsno-širinskom modulacijom [23] (engl. *Pulse-Width Modulation*), koja predstavlja određenu analognu vrijednost ovisnu o odnosu duljine trajanja visokog i niskog signala [15].

Upravljanje koračnim motorima

Koračni motori imaju složeniju izvedbu, a time i složenije upravljanje koje djelomično ovisi o izvedbi samog motora. Upravljanje je najčešće podijeljeno po fazama, gdje se pojedina faza uključi aktiviranjem odgovarajućeg ulaza. Da bi se motoru osovina okretala, potrebno je paliti i gasiti faze određenim redoslijedom, što odgovara izvođenju više koraka okreta [16].

Upravljanje servo motorima

Servo motor dovodi se u željeni položaj tako što mu se pošalje električni impuls sličan pulsno-širinskoj modulaciji. Svaki položaj obilježen je specifičnim odnosom između trajanja visokog i niskog signala [8].

2.1.3. Praćenje položaja motora

Nerijetko je potrebno znati u kojem se položaju motor nalazi, odnosno koliko se okrenuo. Svaka vrsta motora može se pratiti na više načina, od kojih neki načini vrijede za više vrsta motora.

Praćenje položaja istosmjernih motora

Brzina vrtnje istosmjernog motora ovisi o naponu koji mu se zada te o opterećenosti motora, stoga je teško precizno odrediti u kojem se položaju motor nalazi. Jedan od načina praćenja položaja je pomoću tzv. enkodera. Enkoder je komponenta koja očitava pokret ili položaj motora i pretvara ga u analogni ili digitalni signal. Sastoji se od pokretnog i nepokretnog dijela. Pokretni dio pričvrsti se za osovinu na prednjem ili na stražnjem dijelu motora (neki modeli imaju predviđeno mjesto na stražnjem dijelu motora za cijeli enkoder). Vrtnjom osovine okidaju se senzori na nepokretnom dijelu enkodera koji je pričvršćen za kućište motora. Senzori navedene signale pretvaraju u analogni ili digitalni signal pogodan za slanje upravljačkoj jedinici na obradu [17]. Enkoder detektira okretanje osovine motora neovisno o uzroku. Pod uzrokom misli se na željeni pomak, odnosno pokretanje motora pogonskim signalima, ili na neželjeni pomak djelovanjem vanjskih sila, odnosno prisilnim okretanjem osovine motora.

Praćenje položaja koračnih motora

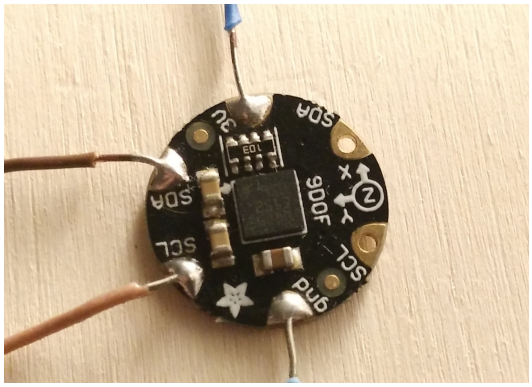
Poznavanjem karakteristika koračnog motora, praćenje položaja izvedivo je bez dodatnih opipljivih dijelova. Svaki koračni motor ima nepromjenjiv broj koraka za puni okret i svaki korak ima određenu jednaku veličinu [16], stoga je za praćenje položaja motora dovoljno pratiti količinu izvedenih koraka. Djelovanje vanjskih sila na okretanje osovine, ovisno o jačini vanjske sile, obično nema nikakvog utjecaja zbog velike snage motora odnosno elektromagneta u motoru [16], stoga se nepredviđeni pomaci u položaju često mogu zanemariti kao mogućnost. Koračni motor može se pratiti i na jednaki način kao i istosmjerni motori.

Praćenje položaja servo motora

Obzirom da se konačni položaj servo motora određuje slanjem specifičnog oblika signala [8], položaj servo motora jednostavno je pratiti pamćenjem prethodno poslanog signala. Slično kao i koračni motor, servo motor će nastojati zadržati zadani položaj, no moguće je pomaknuti ga djelovanjem vanjske sile i upravljačka jedinica o tome neće biti obaviještena. Da bi se servo motor vratio u željeni položaj, dovoljno je ponovno mu poslati signal za pomak u odredišni položaj [8].

2.2. Jedinica za mjerenje inercije

Jedinica za mjerenje inercije (engl. *Inertial Measurement Unit*, slika 2.3) je uređaj za mjerenje položaja, orijentacije i inercije nekog tijela. Redovito se koristi kao komponenta u sustavima kojima je potrebno snalaženje u prostoru, poput zrakoplova, satelita ili drugih bespilotnih vozila. Jedinica za mjerenje inercije tipično sadrži akcelerometar i žiroskop, a nerijetko i magnetometar [12].



Slika 2.3: Komponenta koja sadrži jedinicu za mjerenje inercije LSM9DS0

Akcelerometar je uređaj koji mjeri akceleraciju tijela. Najčešće mjeri sve tri prostorne osi zbog veće upotrebljivosti [18].

Žiroskop je uređaj koji služi za održavanje referentnog smjera ili pružanje stabilnosti uređaja. Sadrži disk ili kotač koji se vrti velikom brzinom oko svoje osi zbog čega se odupire promjeni položaja, što omogućava detekciju promjene položaja tijela [22].

Magnetometar je uređaj za mjerenje magnetskih sila, najčešće raznih magnetskih oscilacija ili Zemljinih magnetnih polova [20].

Jedinica za mjerenje inercije u samobalansirajućem robotu služi za detekciju nagiba robota. Obzirom da robot teži prevrtanju po jednoj osi, potrebno je očitati otklon

robota od stabilnog položaja ravnoteže i navedenu informaciju iskoristiti za pomicanje robota u svrhu ponovne uspostave ravnoteže. Da bi očitana vrijednost bila iskoristiva, potrebno je spojiti informacije akcelerometra i žiroskopa. Više o tome u 3.4.2. Magnetometar nema ulogu u ovom radu.

2.3. Upravljačka jedinica

Interakciju između pojedinih komponenti uređaja mora kontrolirati neka upravljačka jedinica. Najčešći odabir su mikrokontroler ili razne varijacije računala poput Raspberry Pi-ja.

2.3.1. Arduino

Arduino je elektronička platforma otvorenog koda koja služi upravljanju izlazima na temelju dobivenih ulaza. Glavni dio Arduino pločice je programibilni mikrokontroler koji upravlja ulazima i izlazima na isprogramirani način. Postoji veliki broj dodataka i modula za Arduino platformu što ga čini pogodnim za razne projekte. Ima toleranciju na ulazni napon napajanja viši od traženog zbog ugrađenog regulatora koji spušta napon na traženu vrijednost. Ulazi i izlazi rade na naponu od 5 volti [1].

2.3.2. Raspberry Pi

Raspberry Pi (Slika 2.4) je računalo veličine kreditne kartice. Odlikuju ga male dimenzije, niska potrošnja električne energije i niska cijena. Funkcionalnost mu je usporediva sa običnim stolnim računalom slabijih, ali solidnih performansi. Također ima nekoliko desetaka digitalnih ulazno-izlaznih pin-ova, zbog čega može služiti i kao IoT uređaj ili elektronička platforma poput Arduino platforme. Raspberry Pi operativan je na naponu od 5 volti te je preporučljivo osigurati izvor napajanja koji može pružiti barem 2 ampera struje. Raspberry Pi može služiti kao izvor napajanja za komponente male potrošnje i napona od 5 ili 3.3 volta, poput jedinice za mjerenje inercije i enkodera za motore [5].

2.4. Izvor napajanja

Da bi robot bio operativan, potreban mu je izvor napajanja. Obzirom da mora biti mobilan, preporučljivo je osigurati mu napajanje koje ne ovisi o električnoj mreži.

Slika 2.4: Raspberry Pi

Također, poželjno je imati odvojeno napajanje za upravljačku jedinicu i komponente za mobilizaciju robota.

Za komponente za mobilizaciju robota mogu poslužiti litij-ionske baterije. Odlikuju ih veliki kapacitet, visok vijek trajanja i mali memorijski efekt.

Za upravljačku jedinicu prigodan je izvor mobilni USB punjač (engl. power bank), koji daje izvor od 5 volti.

2.5. Analogno-digitalni pretvarač

Analogno-digitalni pretvarač (engl. Analog-to-Digital Converter) je sustav za pretvaranje analognog signala u digitalni oblik [9]. Koristi se za pretvaranje analognih signala dobivenih od enkodera u digitalni, upravljačkoj jedinici čitljiv oblik.

3. Priprema

3.1. Odabir komponenti

Zbog dobavljivosti, malih dimenzija i pristojne snage, za naš robot odabrani su istosmjerni motori. Kao upravljačka jedinica odabrano je računalo Raspberry Pi model 3B+ zbog posjedovanja čipa za bežični pristup mreži (tzv. *Wi-Fi*) [5] te većih mogućnosti nadogradnje. Od jedinice za mjerenje inercije odabran je čip LSM9DS0 zbog malih dimenzija i visoke preciznosti. Za upravljanje motorima odabran je kontroler koji koristi L298N kao motor driver.

3.2. Postavljanje Raspberry Pi-ja

Da bi Raspberry Pi poslužio kao upravljačka jedinica, potrebno ga je pravilno postaviti. Kao operativni sustav odabran je Raspbian, Linux distribucija bazirana na Debianu namijenjena Raspberry Pi računalima [6]. Obzirom da se zahtijeva mobilnost samog robota, potrebno je omogućiti bežični pristup i upravljanje Raspberry Pi-jem. Najjednostavniji pristupni način je pomoću bežične Wi-Fi mreže i SSH mrežnog protokola. Dodatno olakšanje u obliku pristupa grafičkom sučelju Raspberry Pi-ja moguće je pomoću programa za udaljeni pristup radnoj površini poput TightVNC-a.

3.3. Programski jezik

Nužna funkcija Raspberry Pi-ja po pitanju ovog rada je interakcija sa drugim komponentama robota preko ulazno-izlaznih pin-ova. Za upravljanje pin-ovima na Raspberry Pi-ju, moguće je koristiti mnogo različitih programskih jezika od kojih su najčešće korišteni C, C++, Java i Python. Za svaki od njih postoji gotovi set klasa i metoda, odnosno biblioteka koje su održavane od strane poveće zajednice na Internetu ili od strane službenih izvora za neki programski jezik. Zbog svoje brzine i niske razine, pri

čemu se misli na "blizinu hardveru računala", odnosno udaljenosti programskog jezika od same arhitekture računala po pitanju izvođenja instrukcija [13], najbolji izbor za implementaciju samobalansirajućeg robota je programski jezik C.

Jedna od najboljih biblioteka za upravljanje ulazno-izlaznim pin-ovima Raspberry Pi-ja programskim jezikom C je WiringPi. Odlikuju ga visoka brzina izvođenja, odlična dokumentiranost i široka mogućnost primjene. Navedena biblioteka koristit će se pri implementaciji ovog rada.

3.4. Biblioteke za interakciju s komponentama robota

Za svaku komponentu preporučljivo je koristiti službene biblioteke za interakciju. Međutim, biblioteke su uglavnom namijenjene radu s Arduinom, ili s Raspberry Pi-jem ali koristeći Python programski jezik što ne odgovara zbog odabira C-a kao programskog jezika za korištenje u ovom radu. Biblioteke komponenti koje koriste WiringPi ili nije moguće pronaći na Internetu ili su presložene za upotrebu, stoga su u okviru ovog rada implementirane vlastite funkcije za interakciju sa komponentama.

3.4.1. Interakcija s driverom za motore

Jedna od učestalo korištenih varijanti kontrolera za motore koji omogućava upravljanje dvama istosmjernim motorima ima šest ulaznih pin-ova, odnosno za svaki motor po tri pin-a: jedan za omogućavanje ili onemogućavanje motora i po jedan za svaki smjer vrtnje.

Svaki pin može se kontrolirati pulsno-širinskom modulacijom. WiringPi ima metode koje se brinu o pravilnom i učinkovitom korištenju modulacije bez da se korisnik mora brinuti o implementaciji iste.

Primjer postavljanja drivera za motore

```
void setupMotorDriver(){
        int errno = 0;
        //Inicijalno postavljanje WiringPi biblioteke
        wiringPiSetup();

        //Postavljanje pin-ova u izlazni mod
        pinMode (ENA, OUTPUT);
```

```
        pinMode (IN1, OUTPUT);
        pinMode (IN2, OUTPUT);
        pinMode (IN3, OUTPUT);
        pinMode (IN4, OUTPUT);
        pinMode (ENB, OUTPUT);

        //Omogucavanje motora
        digitalWrite(ENA, HIGH);
        digitalWrite(ENB, HIGH);

        //Postavljanje pulsno-sirinske modulacije za
           svaki pin, te provjera je li doslo do pogreske
            u kojem slucaju dolazi do prestanka rada
           programa
        errno = softPwmCreate(IN1, LOW, MAX_SPEED);
        if (errno != 0){
                exit(1);
        }
        errno = softPwmCreate(IN2, LOW, MAX_SPEED);
        if (errno != 0){
                exit(1);
        }
        errno = softPwmCreate(IN3, LOW, MAX_SPEED);
        if (errno != 0){
                exit(1);
        }
        errno = softPwmCreate(IN4, LOW, MAX_SPEED);
        if (errno != 0){
                exit(1);
        }
}
```

Primjer metode za kontrolu jednog od motora

```
void setLeftMotorSpeed(uint8_t direction, uint32_t value)
   {
        if (direction == FORWARD){
```

```
            softPwmWrite(IN1, value);
            softPwmWrite(IN2, LOW);
        }
        if (direction == BACKWARD){
            softPwmWrite(IN1, LOW);
            softPwmWrite(IN2, value);
        }
}
```

3.4.2. Interakcija sa jedinicom za mjerenje inercije

Odabrana inačica jedinice za mjerenje inercije komunicira sa upravljačkom jedinicom pomoću I2C komunikacijskog protokola. I2C je komunikacijski protokol koji omogućuje interakciju upravljačke jedinice sa više uređaja istovremeno koristeći samo dva pin-a [4]. Jedinica za mjerenje inercije dostupna je na dvije I2C adrese, jednu namijenjenu akcelerometru i magnetometru a drugu žiroskopu [11].

Primjer postavljanja jedinice za mjerenje inercije

```
void setupIMUI2C(){
        //Uspostavljanje veze izmedu IMU i upravljacke
           jedinice
        accMagFd = wiringPiI2CSetup(ACC_MAG_I2C_ADDR);
        gyrFd = wiringPiI2CSetup(GYR_I2C_ADDR);

        //Provjera je li veza uspjesno uspostavljena
        if ((accMagFd < 0) || (gyrFd < 0)){
            printf("Error while setting up IMU I2C
               connection.\n");
            exit(1);
        }
}
```

Prije početka čitanja podataka, potrebno je postaviti parametre ponašanja, poput frekvencije čitanja podataka, načina rada i mjere preciznosti podataka. Parametri se postavljaju zapisivanjem određene strukture zapisa u određeni registar.

Primjer postavljanja parametara akcelerometra

```
errno = wiringPiI2CWriteReg8(accMagFd,
   LSM9DS0_CTRL_REG1_XM, 0b10000111); //  400Hz data rate
   , continuous update, XYZ axis enabled
if (errno < 0){
        printf ("Error enableing IMU.\n");
        exit(1);
}
errno = wiringPiI2CWriteReg8(accMagFd,
   LSM9DS0_CTRL_REG2_XM, 0b00100000); // +/- 16G full
   scale
if (errno < 0){
        printf ("Error enableing IMU.\n");
        exit(1);
}
```

Parametri žiroskopa postavljaju se na isti način.

Nakon postavljanja parametara, podaci sa komponenti mogu se čitati. Čitanje vrijednosti vrši se čitanjem sadržaja određenih registara.

Primjer čitanja neobrađenih vrijednosti sa akcelerometra

```
xyzRaw readRawAcc(){
        //struktura u koju se spremaju neobradene
           vrijednosti akcelerometra
        xyzRaw accData = { 0, 0, 0 };
        int xLow, xHigh,
                yLow, yHigh,
                zLow, zHigh;

        //Citanje pojedinih osi akcelerometra. Citaju se
           16-bitni podaci
        xLow  = wiringPiI2CReadReg8(accMagFd,
           LSM9DS0_OUT_X_L_A);
        xHigh = wiringPiI2CReadReg8(accMagFd,
           LSM9DS0_OUT_X_H_A);
```

```
        yLow  = wiringPiI2CReadReg8(accMagFd,
            LSM9DS0_OUT_Y_L_A);
        yHigh = wiringPiI2CReadReg8(accMagFd,
            LSM9DS0_OUT_Y_H_A);

        zLow  = wiringPiI2CReadReg8(accMagFd,
            LSM9DS0_OUT_Z_L_A);
        zHigh = wiringPiI2CReadReg8(accMagFd,
            LSM9DS0_OUT_Z_H_A);

        //Spajanje dvije 8-bitne vrijednosti u jednu 16-
            bitnu
        accData.xData = (int16_t)(xLow | (xHigh << 8));
        accData.yData = (int16_t)(yLow | (yHigh << 8));
        accData.zData = (int16_t)(zLow | (zHigh << 8));

        return accData;
}
```

Čitanje vrijednosti žiroskopa vrši se na isti način.

Pretvaranje vrijednosti u smisleni oblik

Nakon čitanja podataka, potrebno ih je pretvoriti u čovjeku smisleni oblik.

Vrijednosti žiroskopa potrebno je pomnožiti sa faktorom osjetljivosti kutne brzine. Primjerice, ako je konfiguracijom odabrana osjetljivost žiroskopa od +-2000 stupnjeva po sekundi, potrebno je očitanu vrijednost pomnožiti sa 70 milistupnjeva po sekundi po broju da bi dobivena vrijednost bila prikazana u stupnjevima po sekundi. Tablica mogućih vrijednosti za navedenu komponentu nalazi se u [11]. Često žiroskopi u stabilnom položaju ne daju vrijednosti od 0 stupnjeva po sekundi, stoga je poželjno izmjeriti prosjek navedenog odstupanja i oduzeti ga od očitanih vrijednosti.

Vrijednosti akcelerometra također je poželjno pretvoriti u stupnjeve. Navedeno se može učiniti trigonometrijom, odnosno računanjem arkus tangensa dviju osi i pretvaranjem dobivenog kuta iz radijana u stupnjeve. Dovoljno je izračunati dva od tri moguća arkus tangensa zato što vrijednost akceleracije rotacije robota oko svoje osi ("u krug") nije potrebna.

Kombinacija vrijednosti akcelerometra i žiroskopa

Korištenje samo jednog od navedenih modula ne daje precizne rezultate. Akcelerometar je dobar za dugoročna mjerenja, ali zbog svoje osjetljivosti na vibracije i nagle trzaje nije dobar za detekciju kratkih promjena. Žiroskop daje vrlo precizne rezultate pri kratkotrajnim i brzim promjenama, ali je pri dužim intervalima sklon popriličnim odstupanjima od točnih vrijednosti. Iz navedenih razloga koristi se filtar koji će u kraćim intervalima prednost dati žiroskopu, a u duljim periodima akcelerometru. Najčešće se koriste Kalmanov i Komplementarni filtar [14].

Kalmanov filtar daje vrlo precizne rezultate, ali je računski poprilično zahtijevan, dok je Komplementarni filtar veoma jednostavan ali nije precizan koliko i Kalmanov. Za potrebe ovog rada Komplementarni filtar je bolja opcija zbog manje računske složenosti, zbog čega će konačni program koristiti manje resursa, a time i manje vremena za računanje vrijednosti otklona robota od ravnotežnog položaja.

Formula za Komplementarni filtar [14]:

$$kut = 0.98 * (kut + ziroskop * vrem.interval) + 0.02 * akcelerometar$$

Konačno, dobiveni filtrirani kut može se koristiti za detekciju i ispravak otklona robota iz ravnotežnog položaja.

3.4.3. Interakcija sa analogno-digitalnim pretvaračem

Ovisno o izvedbi analogno-digitalnog pretvarača, postoji više načina interakcije s istim.

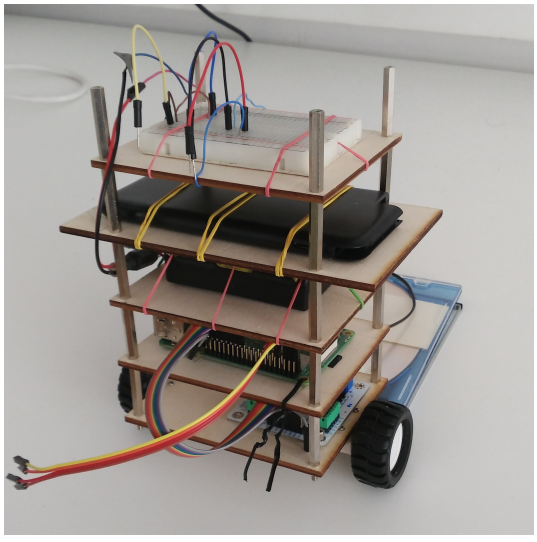
Način komunikacije modela koji komunicira s upravljačkom jedinicom pomoću I2C komunikacijskog protokola veoma je sličan primjeru interakcije sa jedinicom za mjerenje inercije opisanom u 3.4.2. Potrebno je uspostaviti vezu između analogno-digitalnog pretvarača i upravljačke jedinice, postaviti parametre konfiguracije pisanjem u registar, i potom slijedi čitanje podataka iz registra s podacima.

4. Struktura robota

4.1. Fizička struktura

Za učinkovitu implementaciju robota, poželjno je rasporediti elemente na način da olakšaju upravljanje samim robotom, kao i nadogradnju bez narušavanja osnovne funkcionalnosti robota. Postoji više načina za realizaciju navedenog, od čega su razinska i kompaktna među najefikasnijima.

Razinska struktura (Slika 4.1) je među najjednostavnijim strukturama u smislu složenosti implementacije. Svaka razina može biti podijeljena po ulogama i funkcijama te smisleno postavljena u odnosu na cijeli robot. Prednost ove strukture je mala složenost nadogradnje, dovoljno je umetnuti novu razinu na odgovarajuće mjesto. S druge strane, robot građen ovom strukturom često je veći nego isti robot kompaktne strukture, što stvara problem ako je potrebno dizajnirati što manji robot podložan promjenama i nadogradnjama.



Slika 4.1: Primjer razinske strukture

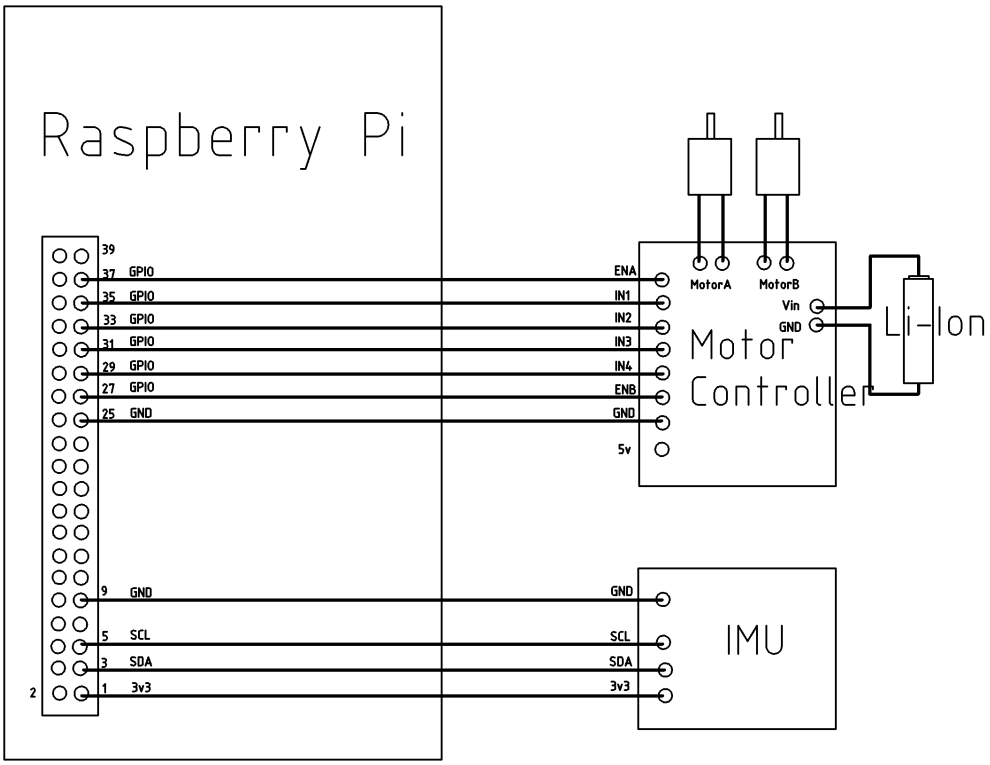
Kompaktna je struktura malo složenija za implementaciju od razinske zbog veće važnosti dimenzija pojedinih komponenti kao i njihove mase. Navedena struktura nez-

godna je po pitanju nadogradnje zbog unaprijed posloženog rasporeda komponenti, zbog čega dodavanjem jedne dodatne komponente može doći do potpunog redizajna cijele unutarnje strukture robota.

Iako se međusobno razlikuju, poželjna zajednička osobina svim strukturama jest visoko težište zbog lakšeg upravljanja robotom, stoga bi sve strukture komponente najveće mase trebale imati najudaljenije od osi rotacije kotača robota.

4.2. Veza između komponenti

Da bi komponente mogle komunicirati, moraju biti povezane. Na slici 4.2 nalazi se primjer veza između komponenti.



Slika 4.2: Primjer sheme spajanja komponenti

GPIO označava digitalne ulazno-izlazne pin-ove Raspberry Pi-ja. IN1 i IN2 su direktna veza na motor A, a IN3 i IN4 na motor B. ENA i ENB označavaju pinove za omogućavanje/onemogućavanje upravljanja motorima.

5. Balansiranje robota u mjestu

Da bi robot bio stabilan u mjestu, potrebno je iskoristiti podatke jedinice za mjerenje inercije i prenijeti ih u odgovarajuću vrijednost brzine i smjera motora. Stoga, robot mora neprekidno:

1. **Čitati otklon ravnoteže**

   Potrebna informacija je otklon od ravnotežnog položaja robota. Često robot nije potpuno točno fizički balansiran, odnosno, stabilno stanje ravnoteže robota nije pri nagibu od 0 stupnjeva, zbog čega je prvi korak, prije pokretanja samobalansiranja, pronalazak otklona ravnoteže. Najjednostavniji način je inicijalno ručno pridržavanje robota u stabilnom položaju ravnoteže i bilježenje vrijednosti. Precizno očitana vrijednost jedna je od ključnih komponenti dobre izvedbe balansiranja robota.

2. **Odrediti potrebni pomak robota PID kontrolerom**

   Nakon očitane vrijednosti otklona stabilne ravnoteže, odnosno iznosa pogreške između ravnotežnog i trenutnog položaja, potrebno je odrediti koliko je potrebno pomaknuti robot u kojem smjeru da bi ravnoteža bila uspostavljena. Jedan od načina za navedeno je upotreba PID kontrolera.

PID kontroler [10]

PID (skraćeno od *Proportional-Integral-Derivative*) kontroler je univerzalno prihvaćeni algoritam za upravljanje postizanjem i održavanje željene vrijednosti. Ima široku primjenu u industriji zbog svoje jednostavnosti i učinkovitosti. Osnovni način rada kontrolera je zatvorena petlja čitanja ulazne vrijednosti i računanja izlazne vrijednosti na način da se ulazna vrijednost što više približi željenoj vrijednosti. Za učinkovito računanje koristi se zbroj tri odziva: proporcionalnog, integralnog i derivacijskog.

Proporcionalni odziv je jednostavno množenje pogreške, odnosno razlike između željene i trenutne vrijednosti, određenim faktorom da bi se konačna vrijednost što više

približila željenoj vrijednosti.

Integralni odziv zbraja pogreške kroz interval vremena, zbog čega se akumulirana vrijednost konstantno povećava ili smanjuje osim ako je pogreška jednaka nuli. Odziv pridonosi preciznijoj reakciji svođenja pogreške na nulu.

Derivacijski odziv omogućava brzu reakciju na nagle promjene u sustavu. Uzima u obzir stopu promjene ulaza, na temelju koje može brzo promijeniti izlaznu vrijednost da bi ulazna odgovarala željenoj. Problem derivacijskog odziva je osjetljivost na šum, što može utjecati na stabilnost sustava. U nekim implementacijama derivacijski odziv nije potreban.

Svaki odziv sadrži konstantu, odnosno faktor, koji varira od sustava do sustava, stoga ih je potrebno odrediti za svaki sustav posebno. Postoji više načina određivanja navedenih faktora, a najjednostavniji je ručnim određivanjem, odnosno metodom pokušaja i pogrešaka.

Ručno određivanje faktora PID kontrolera [10]

U ovom načinu određivanja, svi se faktori prvotno postave na nulu. Potom se faktor proporcionalnog odziva (Kp) povećava sve dok izlaz ne počne oscilirati dovoljnom brzinom. Nakon toga, povećava se faktor integralnog odziva (Ki) da bi se smanjile oscilacije stabilnog stanja. Za kraj, povećava se faktor derivacijskog odziva (Kd) da bi se smanjilo vrijeme stabilizacije sustava.

Primjena PID kontrolera na samobalansiranje

U slučaju samobalansirajućeg robota, ulazna vrijednost je otklon robota, a izlazna vrijednost su brzina i smjer vrtnje motora. Proporcionalni odziv služi za osciliranje robota oko ravnotežnog položaja, integralni odziv za stabilizaciju oscilacija i ispravak akumuliranih pogrešaka, a derivacijski odziv za bržu stabilizaciju te mogućnost pronalaska ravnoteže robota unatoč vanjskom podražaju poput kratkotrajnog poguravanja robota. Ispravne vrijednosti faktora učinit će robot mirnim i stabilnim, te će na vanjski podražaj reagirati dovoljno brzo i točno.

Primjer koda koji vrši samobalansiranje

```
xyzFloat currentAngle;
PID pid1 = {
```

```
        .timeInterval = INTERVAL_TIME_LENGTH,
        .Kp = 7,
        .Ki = 45,
        .Kd = 0.35,
        .setPoint = GOAL,
        .downLimit = -100,
        .upLimit = 100,
        .integral = 0

};

while(1){
        markStartTime();
        updateIMUAngle(&currentAngle);
        input = currentAngle.yData;

        output = calculatePIDOutput(&pid1, input);

        if (output < 0){
                direction = FORWARD;
                output = -output;
        }
        else {
                direction = BACKWARD;
        }

        moveRobot(direction, (int)output);

        markEndTime();
        waitUntil(INTERVAL_TIME_LENGTH);
}
```

6. Kretanje robota prostorom

Kretanje robota prostorom u smjeru mogućeg prevrtanja robota najlakše se realizira mijenjanjem željenog otklona ravnotežnog položaja uz uključen mehanizam balansiranja u mjestu (opisan u poglavlju 5). Time mehanizam samobalansiranja pokušava stabilizirati robot pri nagibu koji nije ravnotežni, što proizvodi kontinuirani pomak robota prostorom u svrhu uspostavljanja "lažne" ravnoteže. Povratak u ravnotežni položaj jednostavno se izvodi postavljanjem željenog otklona u otklon ravnoteže.

Jedna od jednostavnijih mogućih izvedbi rotacije robota oko svoje osi je pokretanje jednog od motora, pri čemu mehanizam samobalansiranja ostaje uključen. Navedeno ima za posljedicu zakretanje robota oko svoje osi pri čemu zadržava ravnotežu.

7. Zaključak

Koncept samobalansirajućeg robota nije pretjerano složen, no zahtijeva pomno planiranje po pitanju odabira dijelova te mnogo vremena po pitanju kalibracije parametara robota i pisanja programskog koda. Pažljivim odabirom konstrukcije i dijelova te programskih biblioteka i rješenja problema moguće je stvoriti precizni samobalansirajući robot visoke otvorenosti prema nadogradnjama.

Zbog problema sa nabavom motora odgovarajuće brzine i snage te pripadnih enkodera, u sklopu ovog rada nije bilo moguće izvesti kontrolirano kretanje robota prostorom.

Kao nadogradnju robota moguće je ugraditi kameru u svrhu robotovog samostalnog praćenja mete kroz prostor. Još jedna mogućnost za nadogradnju robota je ugradnja pomoćnih nožica koje bi spriječile nepovratno prevrtanje robota i omogućilo potencijalni robotov vlastiti povratak iz prevrnutog u samobalansirajući položaj.

LITERATURA

[1] What is Arduino? https://www.arduino.cc/en/guide/introduction. Pristupano: 2018-05-31.

[2] About Handle. https://www.bostondynamics.com/handle. Pristupano: 2018-06-08.

[3] DC motor or Direct Current Motor. https://www.electrical4u.com/dc-motor-or-direct-current-motor/. Pristupano: 2018-05-31.

[4] I2C. https://learn.sparkfun.com/tutorials/i2c. Pristupano: 2018-06-11.

[5] FAQs. https://www.raspberrypi.org/help/faqs/, . Pristupano: 2018-05-31.

[6] Raspbian. https://www.raspbian.org/, . Pristupano: 2018-06-11.

[7] Priča o Segway-u. http://hr-hr.segway.com/the-story-of-segway. Pristupano: 2018-06-08.

[8] How Do Servo Motors Work? https://www.jameco.com/jameco/workshop/howitworks/how-servo-motors-work.html. Pristupano: 2018-05-31.

[9] analog-to-digital conversion (ADC). https://whatis.techtarget.com/definition/analog-to-digital-conversion-ADC, 2005. Pristupano: 2018-06-11.

[10] PID Theory Explained. http://www.ni.com/white-paper/3782/en/, 2011. Pristupano: 2018-06-11.

[11] LSM9DS0. https://cdn-shop.adafruit.com/datasheets/LSM9DS0.pdf, 2013. Pristupano: 2018-06-01.

[12] What is an IMU? https://www.spartonnavex.com/imu/, 2015. Pristupano: 2018-06-01.

[13] Low-level language. https://www.computerhope.com/jargon/l/lowlangu.htm, 2017. Pristupano: 2018-06-11.

[14] Bonnie Baker. Apply Sensor Fusion to Accelerometers and Gyroscopes. https://www.digikey.com/en/articles/techzone/2018/jan/apply-sensor-fusion-to-accelerometers-and-gyroscopes, 2018. Pristupano: 2018-06-11.

[15] Jordan Dee. Pulse Width Modulation. https://learn.sparkfun.com/tutorials/pulse-width-modulation. Pristupano: 2018-06-11.

[16] Bill Earl. What is a Stepper Motor? https://learn.adafruit.com/all-about-stepper-motors/what-is-a-stepper-motor, 2014. Pristupano: 2018-05-31.

[17] Elisabeth Eitel. Basics of Rotary Encoders: Overview and New Technologies. http://www.machinedesign.com/sensors/basics-rotary-encoders-overview-and-new-technologies-0, 2004. Pristupano: 2018-06-01.

[18] Ryan Goodrich. Accelerometers: What They Are and How They Work. https://www.livescience.com/40102-accelerometers.html, 2013. Pristupano: 2018-06-11.

[19] Arpit Jain. Insight - How Geared DC Motor works. https://www.engineersgarage.com/insight/how-geared-dc-motor-works, . Pristupano: 2018-05-31.

[20] Preeti Jain. Magnetometers. https://www.engineersgarage.com/articles/magnetometer, . Pristupano: 2018-06-11.

[21] Ioannis Kafetzis i Lazaros Moysis. Inverted Pendulum: A system with innumerable applications. https://www.researchgate.net/profile/Lazaros_Moysis/publication/313975571_Inverted_Pendulum_A_system_with_innumerable_applications/links/58b073e345851503be97e171/

Inverted-Pendulum-A-system-with-innumerable-applications.pdf, 03 2017. Pristupano: 2018-06-08.

[22] Christopher McFadden. What are Gyroscopes and Why are They Important? https://interestingengineering.com/what-are-gyroscopes-and-why-are-they-important, 2017. Pristupano: 2018-06-01.

[23] Sam. Motor Drivers vs. Motor Controllers. https://core-electronics.com.au/tutorials/motor-drivers-vs-motor-controllers.html, 2017. Pristupano: 2018-06-11.

[24] K. Sultan. Inverted pendulum, analysis, design and implementation. http://www.engr.usask.ca/classes/EE/480/Inverted%20Pendulum.pdf, 2003. Pristupano: 2018-06-08.

[25] Guo Xiaoli. MCU 101: How does a DC Motor work? https://community.nxp.com/docs/DOC-1067, 2012. Pristupano: 2018-05-31.

Samobalansirajući mobilni robot

Sažetak

Samobalasirajući mobilni robot sastoji se od dva motora s kotačima sa zajedničkom osi rotacije. Konstrukcija ovakvog robota je nestabilna i teži prevrtanju. Potencijalno prevrtanje detektira se jedinicom za mjerenje inercije, a položaj robota ispravlja se korištenjem PID kontrolera, kojem je ulaz otklon od ravnotežnog položaja, a izlaz vrijednost koja služi za upravljanje motorima u odgovarajućem smjeru odgovarajućom brzinom. Upravljanju motorima kao komponenta posreduje motor kontroler. Za upravljanje robotom koristi se Raspberry Pi, a programski jezik koda je C. Robot ima vlastito napajanje i mobilan je. Napajanje je odvojeno za motore i za Raspberry Pi.

Ključne riječi: samobalansiranje, robot, akcelerometar, žiroskop

Self-Balancing Mobile Robot

Abstract

Self-balancing mobile robot is made of two motors with wheels, which have a common rotation axis. A robot constructed in that way is physically unstable and strives to overturn, which is detected with an inertial measurement unit. Robot's vertical position is corrected by PID controller, whose input is the deviation from balanced position, and output is a value used for turning the motors in proper direction with proper turning speed. A motor controller is used as a mediator for controlling the motors. Raspberry Pi is used as a control unit, and the robot's main program is written in C. Robot has its own power supply and is mobile. Motors and Raspberry Pi each have its own power supply.

Keywords: self-balancing, robot, accelerometer, gyroscope