Samobalansirajući robot na dva kotača

Radas, Filip

Sveučilište
Sjever

Završni rad br. 009/MEH/2022

Samobalansirajući robot na dva kotača

Student, Filip Radas, 0336036684

Sveučilište
Sjever

Mehatronika

Završni rad br. 009/MEH/2022

Samobalansirajući robot na dva kotača

Student

Filip Radas, 0336036684

Mentor

Miroslav Horvatić, dipl. ing.

Varaždin, Rujan 2022. godine

Sažetak:

Samobalansirajući robot je mobilni robot koji ima dva kotača. Jedan se nalazi s lijeve, a drugi s desne strane pa robot neće biti u ravnoteži ako se ne kontrolira. Završni rad ima cilj izraditi samobalansirajući robot i dizajnirati sustav upravljanja koji može uravnotežiti robot. Komponenta koja se koristi kao ulaz je senzor MPU6050, a izlaz iz senzora u obliku kuta nagiba bit će obrađen pomoću mikrokontrolera. Dobiveni kut će se usporediti sa zadanom vrijednošću kuta koji iznosi 0 stupnjeva. Razlika između zadane vrijednosti i izlaznog kuta sustava kontrolira se pomoću PID regulatora. PID Kp, Kd i Ki vrijednosti regulatora će se odrediti pomoću Ziegler-Nichols-ove metode ruba stabilnosti.

Ključne riječi: PID regulator, MPU6050, Arduino Uno, DC motor, Ziegler-Nichols metoda

Abstract:

A self-balancing robot is a mobile robot that has two wheels. One is on the left and the other on the right, so the robot will not be in balance if it is not controlled. The final work aims to create a self-balancing robot and design a control system that can balance the robot. The component used as the input is the MPU6050 sensor and the output from the sensor in the form of tilt angle will be processed by the microcontroller. The obtained angle will be compared with the default value of the angle which is 0 degrees. The difference between the setpoint and the output angle of the system is controlled by a PID controller. The PID Kp, Kd and Ki values of the controller will be determined using the Ziegler-Nichols stability margin method.

Keywords: PID regulator, MPU6050, Arduino Uno, DC motor, Ziegler-Nichols method

Popis slika:

Popis tablica:

Popis kratica:

Oznaka	Opis	Jedinica
b	Koeficijent otpora gibanju,	Ns/m
F_H	Horizontalna sila između kotača i osovine motora,	Ns/m
F_V	Vertikalna sila između kotača i osovine motora,	N
F_{PH}	Sila trenja,	N
F_{PV}	Vertikalna sila podloge na kotače,	N
g	Zemljina gravitacija,	m/s^2
J_k	Moment inercije kotača oko osi rotacije,	kgm^2
J_n	Moment inercije njihala oko osi rotacije,	kgm^2
K_a	Pojačanje armature,	A/V
K_{ch}	Pojačanje choppera,	V/V
K_e	Pojačanje EMS,	Vs/rad
K_m	Pojačanje momenta,	Nm/A
K_r	Pojačanje redukcije,	rad/rad
K_{rf}	Pojačanje redukcije i gubitaka trenja,	Nm/Nm
L	Udaljenost od z-osi do težišta robota,	m
m_k	Masa kotača,	kg
m_n	Masa njihala,	kg
M_M	Moment motora,	Nm
r	Polumjer kotača,	m
x	Pomak po x-osi,	m
$\dot{x}$	Brzina po x-osi,	m/s
$\ddot{x}$	Ubrzanje po x-osi,	m/s^2
x_n	Pomak težišta njihala po x-osi u odnosu na os rotacije kotača,	m
y	Pomak po y-osi,	m
$\dot{y}$	Brzina po y-osi,	m/s
$\ddot{y}$	Ubrzanje po y-osi,	m/s^2

y_n	Pomak težišta njihala po y-osi u odnosu na os rotacije kotača,	m
Θ_n	Kut nagiba njihala,	rad
$\dot{\Theta}_n$	Kutna brzina njihala,	rad/s
$\ddot{\Theta}_n$	Kutna akceleracija njihala,	rad/s^2
$\ddot{\Theta}_k$	Kutna akceleracija kotača	rad/s^2

SADRŽAJ:

1. UVOD

Samobalansirajući robot je zanimljiv zbog toga što predstavlja prirodno nestabilan sustav, u ovome slučaju invertirano njihalo, kojeg je moguće pretvoriti u stabilan pomoću PID regulacije i dobre fizičke izrade robota. Da bi se riješio problem nestabilnosti potrebna su znanja iz raznih područja kao što su senzorika, programiranja mikrokontrolera i teorije automatizacije sustava. Iz tog razloga je to odličan primjer projekta za studente mehatronike jer objedinjuje sva znanja koja se steku kroz studij.

Za sada glavnu primjenu samobalansirajućih vozila vidimo na primjeru prijevoza ljudi, koju je popularizirala i razvila tvrtka Segway. Također bi se samobalansirajuća vozila mogla koristiti u tvornicama za transport materijala. Prednost samobalansirajućih vozila je ta što zauzimaju manje mjesta i mogu raditi oštre okretaje, a time se mogu kretati u manjem prostoru.

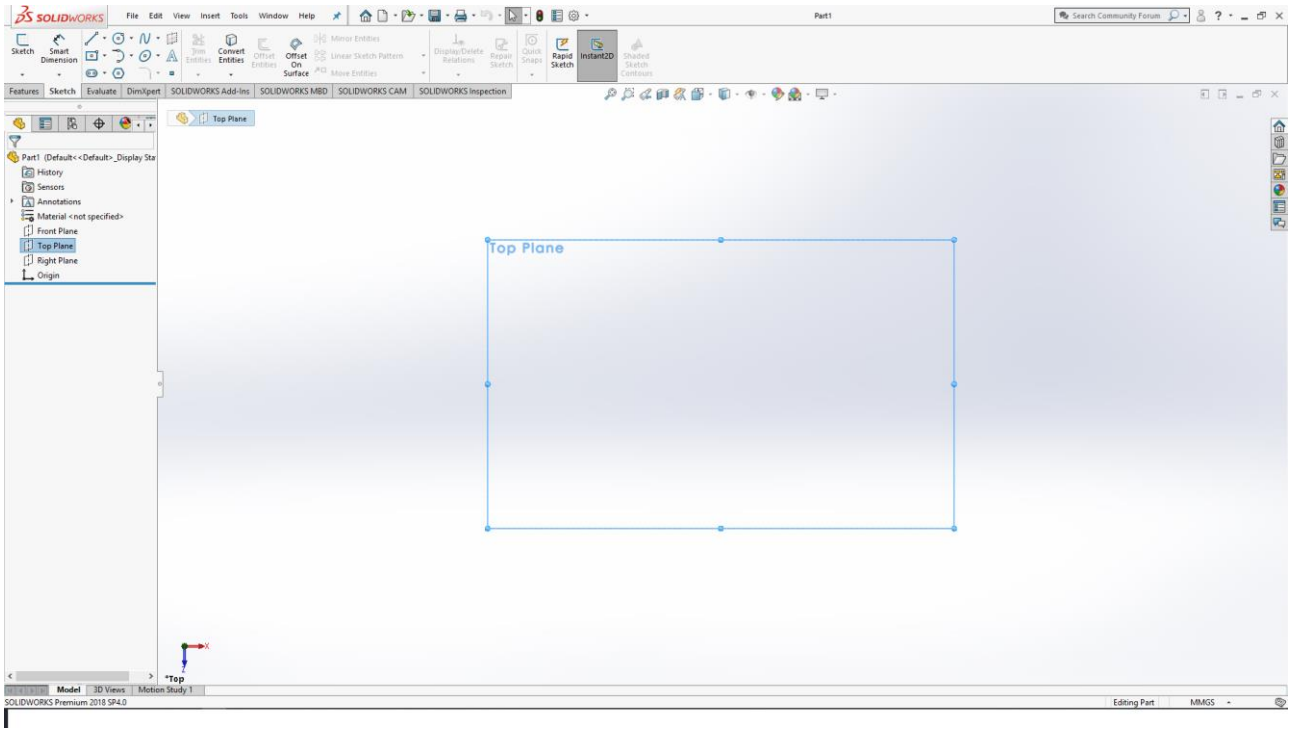
Pretvaranje robota iz nestabilnog sustava u stabilan je jedan od najboljih primjera evolucije tehnologije koja otvara mnoga vrata za nova inovativna otkrića za smanjenje troškova transporta unutar industrije i ubrzanja i povećanja efikasnosti transporta industrije.s

2. MODELIRANJE KUĆIŠTA

Za modeliranje kućišta korišten je SolidWorks. SolidWorks je programski paket koji služi za modeliranje 3D objekata.[3] Korišten je jer je jednostavan za korištenje i može objediniti više pojedinačno izrađenih dijelova u jedan međusobno spojeni sklop te je preglednost cijelog sklopa puno bolja. Pri modeliranju treba obratiti pažnju na dimenzije dijelova da kako bi se svi dijelovi kasnije mogli međusobno povezati, na to da ima dovoljno prostora za sve komponente te da komponente ne budu previše udaljene od težišta robota koje se nalazilo u sredini jer bi inače balansiranje bilo otežano.

2.1 SolidWorks

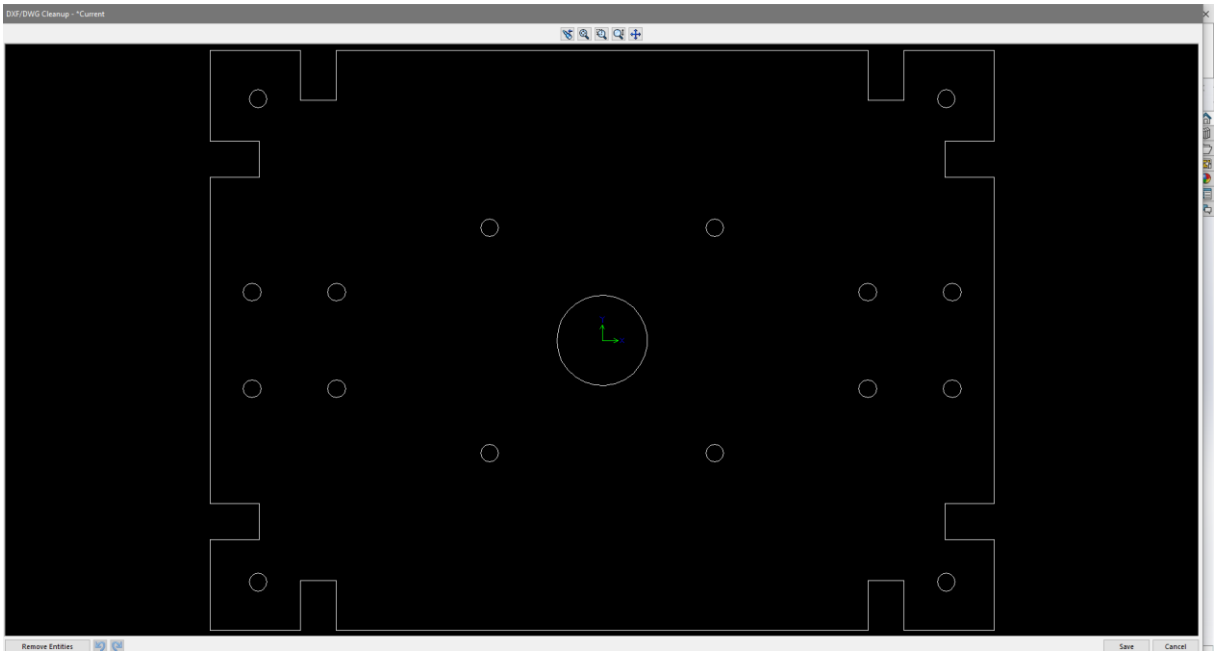
SolidWorks je CAD i CAE računalni program za čvrsto modeliranje. SolidWorks je čvrsti modelar, koristi parametarski pristup kreiranju modela i montaže. Softver je napisan na Parasolid kernelu. SolidWorks datoteke koriste Microsoft *Structured Storage* format datoteka. [3] Ovo znači da postoje različite datoteke ugrađene u okviru svakog SLDDRW (crtež), SLDPRT (dio), SLDASM (montaža) datoteka. SolidWorks je kreirao Dassault Systèmes. [3]



Slika 2.1. SolidWorks sučelje

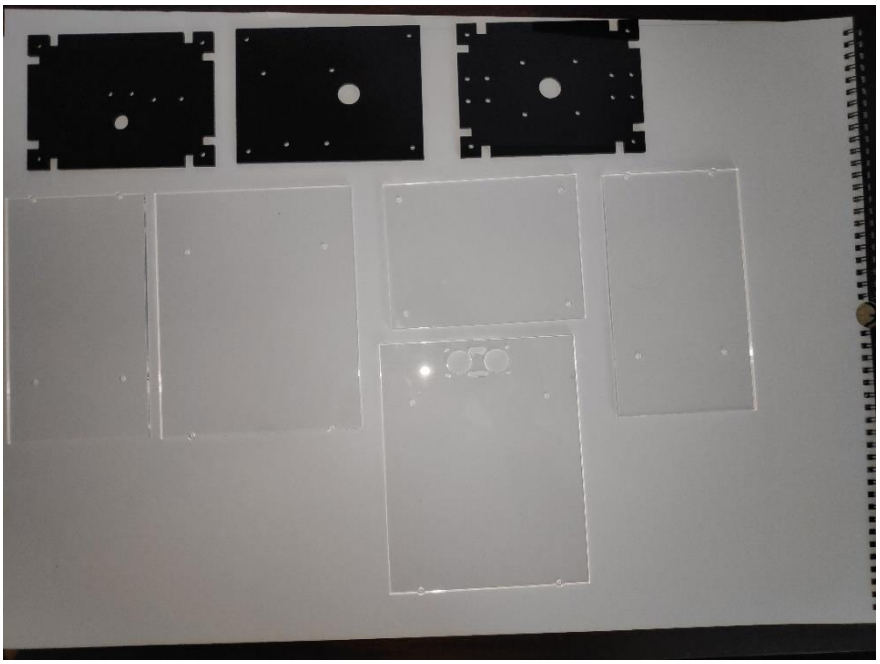
3. IZRADA KUĆIŠTA

Za izradu dijelova kućišta korišten je stroj za lasersko rezanje plastike. Zbog toga su se morali svi dijelovi izrađeni u SolidWorks-u pospremiti u DXF. format datoteke jer je to format kojega stroj može prepoznati i prema njemu točno izrezati dijelove prema mjeri.



Slika 3.1. DXF. prikaz baze robota

Kućište je izrađeno od akrilnih ploča jer su dosta čvrste i elastične pa zbog toga imaju dovoljno jaki otpor vanjskim udarima. Za sklapanje pločica korišteni su vijci M3, matice M3 i odstojnici M3 muško/ženski i odstojnici M3 ženski/ženski.



Slika 3.2. Izrezani dijelovi kućišta od akrilnih ploča

4. ARDUINO

Arduino je open-source platforma za kreiranje elektroničkih prototipova bazirana na sklopovlju i programskom paketu koji je fleksibilan i jednostavan za korištenje.[14] Glavnu jezgru čini mikrokontroler koji može primati i slati signale u okolinu i iz okoline. Većina Arduino ploča sastoji se od Atmel 8-bitnog AVR mikrokontrolera. Ploče su opremljene setovima digitalnih i analognih ulaza / izlaza (I / O) pinova koji se mogu povezati s različitim pločama (štitovima) i drugim sklopovima radi ostvarivanja više različitih funkcija. Arduino se programira pomoću integriranog razvojnog okruženja (IDE) na temelju programskih jezika poput C i C++.

4.1 ARDUINO UNO

Arduino UNO je mikrokontrolerska ploča bazirana na microchipu ATmega328P. Temeljna je komponenta u izradi samobalansirajućeg robota opisanom u ovome radu jer upravlja svim ostalim komponentama. Programira se pomoću integriranog razvojnog okruženja (IDE). Prijenos programa na ploču se vrši preko tip B USB kabla. Arduino UNO je namijenjen za manje projekte jer nema mnogo ulaznih/izlaznih pinova za razliku od Arduino Mega mikrokontrolera. Za izradu robota ima dovoljan broj pinova pa je zbog toga bio odabran. Dodatna prednost mu je da zauzima manje prostora nego Arduino Mega mikrokontroler.

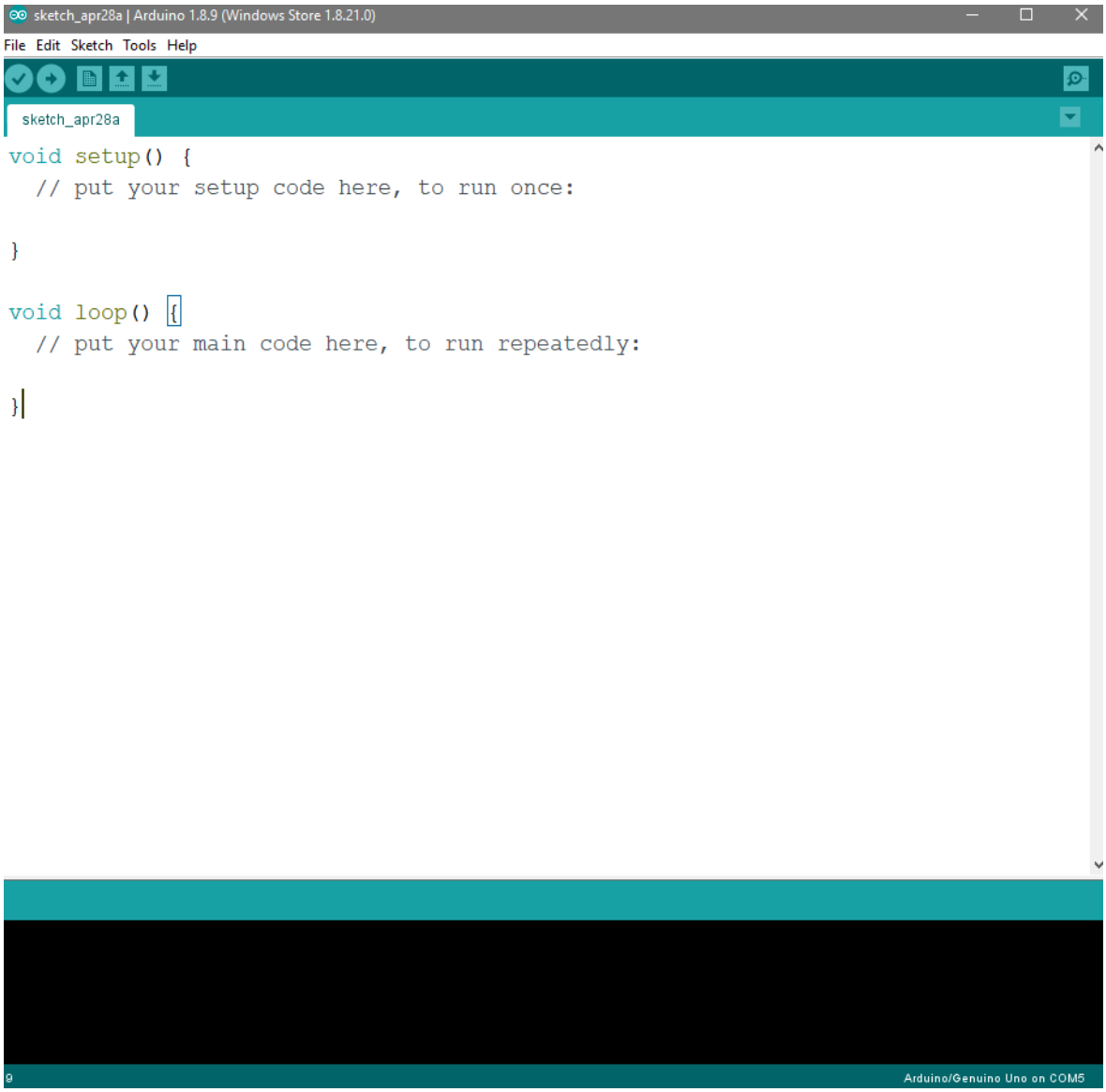
Tehničke karakteristike Arduino UNO mikrokontrolera:

- Radni napon: 5V -12V
- Digitalni I / O pinovi: 14 (od kojih su 6 PWM izlazi)
- Analogni ulazni pinovi: 6
- Flash memorija: 32 KB
- SRAM: 2 KB
- EEPROM: 1 KB

Slika 4.1. Arduino UNO mikroupravljač

4.2 ARDUINO IDE

Arduino IDE je *open source* softver koji se koristi za pisanje i kompajliranje koda. Lako je dostupan operacijskim sustavima kao što su MAC, Windows, Linux i radi na Java platformi koja dolazi s ugrađenim funkcijama i naredbama koje omogućavaju otklanjanje pogrešaka, uređivanju i kompajliranju koda. Podržava C i C ++ jezike. [14]



Slika 4.2. Arduino IDE

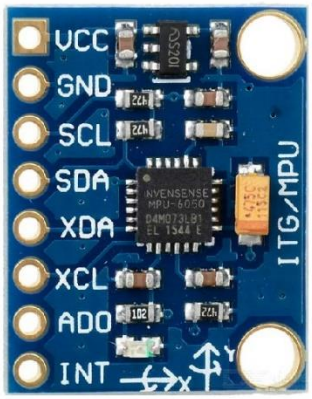
5. MPU-6050 MODUL

MPU modul je 6-osni uređaj za praćenje pokreta. Kombinira 3-osni žiroskop i 3-osni akcelerometar u jednom čipu. Temeljen je na tehnologiji MEMS (mikro elektro mehanički sustavi). MPU-6050 je prije upotrebe potrebno kalibrirati jer je početna brojčana vrijednost osi (offset) različita za svaki MPU6050 modul. [5] Tijekom kalibracije MPU modul mora biti postavljen ispravno tako da nije nakošen jer bi inače utjecalo na kalibraciju modula. Modul koristi I²C protokol za komunikaciju. Za početak komunikacije treba oko dvije sekunde. Modul koristi digitalni *interrupt* pin(2) za rad. MPU je temeljna komponenta u izradi samobalansirajućeg robota jer je potrebna da prepozna ako je došlo do stvaranja nagiba robota. Ovisno o trenutnoj vrijednosti MPU-a, PID regulator određuje koliku korekciju mora učiniti da se izbalansira sustav.

Tehničke karakteristike MPU-6050 modula:

- Radni napon: 1.8 ± 5V
- opseg žiroskopa - 250 500 1000 2000 ° / s
- Raspon akcelerometra - ± 2 ± 4 ± 8 ± 16 g
- Dimenzije modula - 4x4x0.9
- Koristi I²C protokol za komunikaciju – SCL (*serial clock*)

  – SDA (*serial data*)



Slika 5.1. MPU6050 Modul [5]

6. ISTOSMJERNI DVOSTRUKI H-MOST L298N

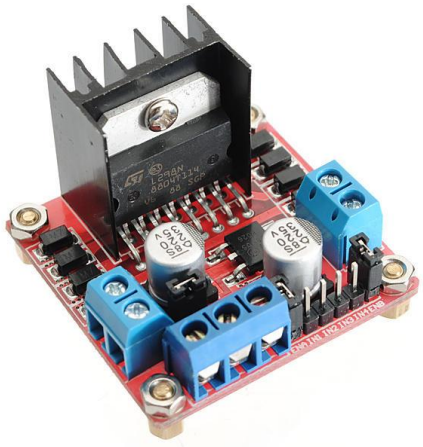
H-most omogućava zamjenu polariteta na svojim izlazima pa tako omogućava kontrolu vrtnje motora u oba smjera na vrlo jednostavan način. [6] Na njega se mogu istovremeno spojiti dva istosmjerna motora. Modul ima posebno napajanje od 7.4V, a koristi GND od Arduino UNO mikrokontrolera. Ima dva EN pina (ENA I ENB) koji omogućuju regulaciju brzine okretaja motora. Te pinove je potrebno spojiti na PWM pinove Arduino ploče. Koristi i četiri IN pina koji služe za određivanje polariteta, to jest određuju smjer vrtnje motora ovisno na koje IN pinove je poslan signal. H-most šalje 7.4V napajanja na dva motora spojena na njega.

L298N ime sljedeće pinove:

- EnA – Služi za kontrolu PWM-a za motor A
- EnB - Služi za kontrolu PWM-a za motor B
- In1 - Služi za kontrolu smjera okretaja motora A.
- In2 - Služi za kontrolu smjera okretaja motora A.
- In3 - Služi za kontrolu smjera okretaja motora B.
- In4 - Služi za kontrolu smjera okretaja motora B.
- +5V –Ulaz u koji ide vanjski izvor napajanja od 5V
- GND – Ground. Potrebno spojiti sve minuse strujnog kruga s ovim GND.
- +12V – Ulaz vanjskog izvora napajanja za motore.

Tehničke karakteristike dvostrukog H-mosta L298M :

- Max. snaga: 25W
- Napon: 5 - 35V(za motor), 5V za logiku
- Struja: 2A
- Dimenzije: 43 x 43 x 23 mm



Slika 6.1. Istosmjerni dvostrukog H-mosta L298M [6]

7. ISTOSMJERNI MOTORI S ENKODEROM I S KOTAČIMA

Pri odabiru motora je bilo vrlo važno odabrati dovoljno jake motore i dovoljne velike kotače da mogu stabilizirati robota kada se počinje naginjati te da motori sadrže enkodere pomoću kojih se može određivati trenutna brzina motora. Elektromotori se spajaju na L298N H-most modul i upravljaju se pomoću njega.

Tehničke karakteristike elektromotora, reduktora i kotača :

- Promjer kotača: cca. 65mm
- Promjer osovine: cca. 4mm
- Napon: DC 6V
- Brzina: 210 okr / min
- Nazivni napon: DC 6V
- Brzina praznog hoda: 210RPM 0.13A
- Maksimalna učinkovitost: 2.0kg.cm/170rpm/2.0W/0.60A
- Maksimalna snaga: 5.2kg.cm/110rpm/3.1W/1.10A
- Okretni moment: 10kg.cm 3.2A
- Omjer redukcije reduktora: 1: 34
- Duljina priključka: cca. 200mm

Slika 7.1. DC Motori s enkoderom i kotačima

8. BATERIJE

Za napajanje motora korištene su dvije 18650 5000mAh 3.7V Li-ion baterije i jedna 9V alkalna baterija koja služi za napajanje Arduino UNO ploče. Li-ion baterije su odabrane jer imaju veliki kapacitet i mogućnost punjena, a alkalna baterija je odabrana jer ju je jednostavno spojiti na Arduino UNO.

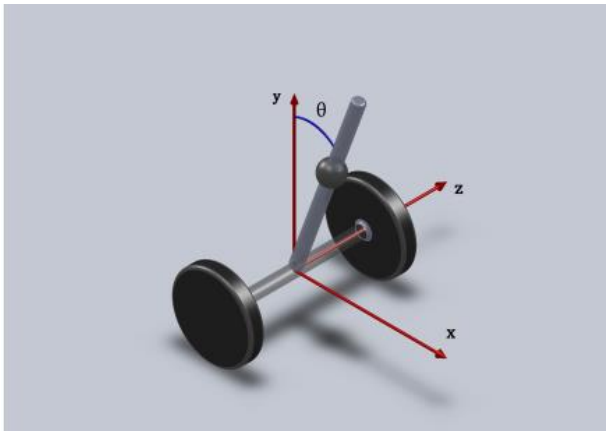
Slika 8.1. 9V alkalna baterija

Slika 8.2. 18650 5000 mAh 3.7V Li-ion baterije

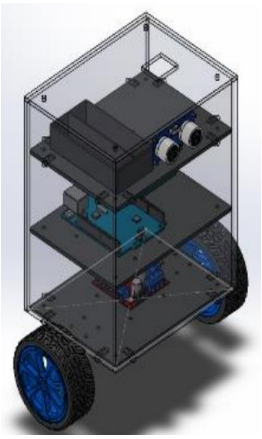
9. MATEMATIČKOG MODELA ROBOTA

9.1 Matematički model robota

Kod izvođenja matematičkog modela potrebno je robot rastaviti na dva dijela; kotače i na njihalo robota. Zatim se za svaki dio postavlja suma sila i momenta oko određene točke. [13]



Slika 9.1. Pojednostavljeni prikaz robota [13]



Slika 9.2 .Stvarni model robota

Jednadžbe za kotač [13]:

$$\sum F_x = 0: \qquad m_k\ddot{x} = -\frac{1}{2}F_H + F_{PH} - b\dot{x} \tag{9.1}$$

$$\sum F_y = 0: \qquad m_k\ddot{y} = F_{PV} - \frac{1}{2}F_V - m_k g, \tag{9.2}$$

$$\sum M_T = 0: \qquad J_k\ddot{\Theta}_k = M_M - F_{PH}r. \tag{9.3}$$

F_H - Horizontalna sila između kotača i osovine motora,

F_V -Vertikalna sila između kotača i osovine motora,

F_{PH}- Sila trenja,

M_M - Moment motora,

J_K-Moment inercije kotača oko osi rotacije,

g -Zemljina gravitacija,

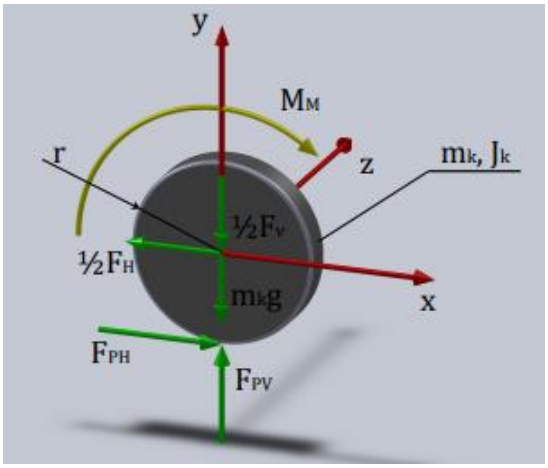
m_k -Masa kotača

$\ddot{\Theta}_k$ - Kutna akceleracija kotača,

$\ddot{y}$- Ubrzanje po y-osi,

$\ddot{x}$- Ubrzanje po x-osi,

$\dot{x}$ - Brzina po x-osi,



9.3. Momenti i sile na kotačima [13]

Preko sljedećih jednadžbi je moguće odrediti kutno ubrzanje kotača $\ddot{\Theta}_k$:

$$x = r\Theta_k, \tag{9.4}$$

$$\dot{x} = r\dot{\Theta}_k, \tag{9.5}$$

$$\ddot{x} = r\ddot{\Theta}_k. \tag{9.6}$$

Iz jednadžbe (9.6) može se vidjeti da je:

$$\ddot{\Theta}_k = \frac{\ddot{x}}{r}, \tag{9.7}$$

Uvrštavanjem jednadžbe (9.7) u (9.3)

$$J_k \frac{\ddot{x}}{r} = M_M - F_{PH} r, \tag{9.8}$$

Iz jednadžbe (9.8) može se izraziti sila: F_{PH}

$$F_{PH} = \frac{M_M}{r} - \frac{J_k}{r^2}\ddot{x} \tag{9.9}$$

Uvrštavanjem jednadžbe (9.9) u (9.1) i (9.3) i izlučivanjem dobiva se:

$$(m_k + \frac{J_k}{r^2})\ddot{x} = \frac{M_M}{r} - \frac{1}{2}F_H - b\dot{x} \tag{9.10}$$

$$J_n\ddot{\Theta}_n = -F_H L\cos\Theta_n + F_V L\sin\Theta_n - 2M_M \tag{9.11}$$

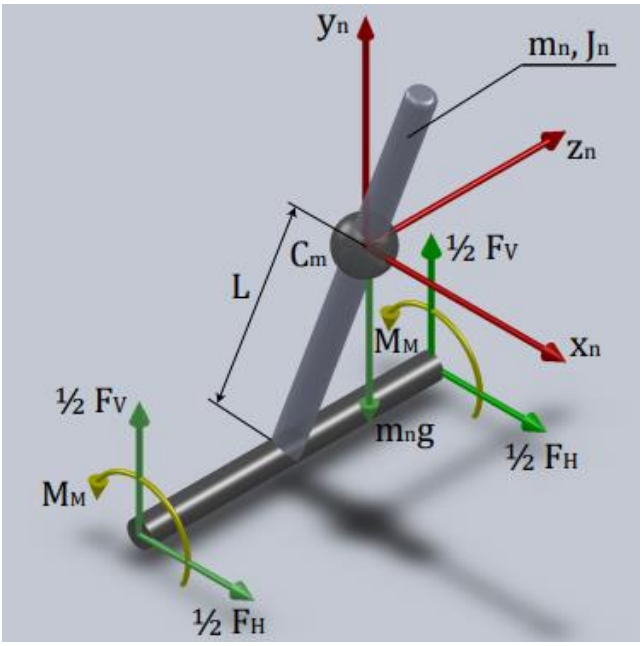
Sada treba pronaći sile F_H i F_V i to se dobiva preko jednadžbi sila njihala:

$$\sum F_x = 0: \qquad m_n \ddot{x}_n = F_H \tag{9.12}$$

$$\sum F_y = 0: \qquad m_n \ddot{y}_n = F_V - m_n g \tag{9.13}$$

Izlučivanjem sile F_V iz (9.13) dobiva se:

$$F_V = m_n \ddot{y}_n + m_n g \tag{9.14}$$



Slika 9.4. Momenti i sile na njihalu [13]

m_n - Masa njihala,

$\ddot{x}_n$ - Pomak težišta njihala po x-osi u odnosu na os rotacije kotača,

$\ddot{y}_n$ - Pomak težišta njihala po y-osi u odnosu na os rotacije kotača

F_H - horizontalna sila između kotača i osovine motora,

Fv - Vertikalna sila između kotača i osovine motora,

g – zemljina gravitacija,

Nakon izvođenja osnovnih jednadžbi sila i momenta kotača i njihala potrebno je pronaći vezu između težišta njihala s kutom nagiba robota i pomaka kotača kako bi dobili potrebne sile F_H i F_V. Iz slike 9.4 dobiva se sljedeći izraz:

$$x_n = x + L\sin\Theta_n, \tag{9.15}$$

Dvostrukim deriviranjem izraza (9.15) dobivamo željenu vezu varijabli:

$$\dot{x}_n = \dot{x} + L\cos\Theta_n\dot{\Theta}_n \tag{9.16}$$

$$\ddot{x}_n = \ddot{x} + L\cos\Theta_n\ddot{\Theta}_n - L\sin\Theta_n\dot{\Theta}_n^2. \tag{9.17}$$

Nakon dobivanja varijabli u smjeru x-osi radi se ista stvar za y-os:

$$y_n = L\cos\Theta_n, \tag{9.18}$$

$$\dot{y}_n = -L\sin\Theta_n\dot{\Theta}_n, \tag{9.19}$$

$$\ddot{y}_n = -L\sin\Theta_n\ddot{\Theta}_n - L\cos\Theta_n.\dot{\Theta}_n^2. \tag{9.20}$$

Izraze za sile F_H I F_V dobivaju se uvrštavanjem (9.17) u (9.12) i (9.20) u (9.14)

$$F_H = m_n(\ddot{x} + L\cos\Theta_n\ddot{\Theta}_n - L\sin\Theta_n\dot{\Theta}_n^2), \tag{9.21}$$

$$F_V = m_n(-L\sin\Theta_n\ddot{\Theta}_n - L\cos\Theta_n\dot{\Theta}_n^2 + g). \tag{9.22}$$

Izraze za sile F_H I F_V se uvrštavaju u (9.10) i (9.11) i sređivanjem se dolazi do sljedećeg:

$$(J_n + L^2m_n)\ddot{\Theta}_n = -Lm_n\ddot{x}\cos\Theta + gLm_n\sin\Theta - 2M_M, \tag{9.23}$$

$$(m_k + \frac{J_k}{R^2} + \frac{1}{2}m_n)\ddot{x} = \frac{M_M}{R} - \frac{1}{2}m_nLcos\Theta\ddot{\Theta} + \frac{1}{2}m_nLsin\Theta\dot{\Theta}^2 - b\dot{x}. \tag{9.24}$$

Gdje su:

m_k - Masa kotača

J_k - Moment inercije kotača oko osi rotacije,

R - polumjer kotača

m_n – masa njihala

M_M - Moment motora,

L -Udaljenost od z-osi do težišta robota

b - Koeficijent otpora gibanju,

$\dot{x}$ - Brzina po x-osi,

$\ddot{x}$ - Ubrzanje po x-osi,

Θ - Kut nagiba njihala,

$\dot{\Theta}$ - Kutna brzina njihala

$\ddot{\Theta}$ - Kutna akceleracija njihala

9.2. Linearni model robota

Linearizacijom se dobije linearni model sustava koji za male vrijednosti kuta oko radne točke daje dobru aproksimaciju ponašanja nelinearnog modela. Za radnu točku odabrana je vrijednost kuta nagiba $\Theta = 0$.

Prema tome vrijede jednadžbe:

$$sin\Theta = \Theta \tag{9.25}$$

$$cos\Theta = 1 \tag{9.26}$$

$$\dot{\Theta}^2 = 0, \tag{9.27}$$

Uvrštavanjem tih jednadžbi u (9.23) i (9.24) dobiva se:

$$\ddot{x} = \frac{1}{m_k + \frac{J_k}{R^2} + \frac{m_n}{2}}(\frac{M_M}{R} - b\dot{x} - \frac{m_n}{2}L\ddot{\Theta}), \tag{9.28}$$

$$\ddot{\Theta} = \frac{-2M_n - LM_n\ddot{x} + gLM_n\Theta}{J_n + L^2m_n}. \tag{9.29}$$

Zatim se (9.28) i (9.29) uvrštavaju jedna u drugu i sređivanjem se dobiva konačni izraz:

$$\ddot{x} = a_{22}\dot{x} - a_{23}\Theta + b_2u, \tag{9.30}$$

$$\ddot{\Theta} = a_{42}\dot{x} - a_{43}\Theta + b_4u, \tag{9.31}$$

gdje su a_{22}, a_{23}, a_{42}, a_{43} koeficijenti matrice sustava, a b2, b4 koeficijenti matrice ulaza sustava [17] :

$$a_{22} = \frac{2\{J_n(K_aK_eK_rK_mK_{rf} + bR^2) + Lm_n[bLR^2 + K_aK_eK_rK_mK_{rf}(L+R)]\}}{2J_k(J_n + L^2m_n) + [2L^2m_km_n + J_n(2m_k + m_n)]R^2}, \tag{9.32}$$

$$a_{23} = -\frac{gL^2m_n^2R^2}{J_k(2J_n + 2L^2m_n) + [2L^2m_km_n + J_n(2m_k + m_n)]R^2}, \tag{9.33}$$

$$a_{42} = \frac{2\{2J_kK_aK_eK_rK_mK_{rf} + R[bLm_nR^2 + K_aK_eK_rK_mK_{rf}(Lm_n + 2m_kR + m_nR)]\}}{R\{2J_k(J_n + L^2m_n) + [2L^2m_km_n + J_n(2m_k + m_n)]R^2\}} \tag{9.34}$$

$$a_{43} = \frac{gLm_n}{J_n + L^2m_n} + \frac{gL^3m_n^3}{2(J_n + L^2m_n)^2(m_k + \frac{m_n}{2} - \frac{L^2m_n^2}{2(J_n+L^2m_n)} + \frac{J_k}{R^2})}, \tag{9.35}$$

$$b_2 = \frac{2K_aK_{ch}K_mK_{rf}R[J_n + Lm_n(L+R)]}{2J_k(J_n + L^2m_n) + [2L^2m_km_n + J_n(2m_k + m_n)]R^2}, \tag{9.36}$$

$$b_4 = -\frac{2K_aK_{ch}K_mK_{rf}\{2J_k + R[Lm_n + (2m_k + m_n)R]\}}{2J_k(J_n + L^2m_n) + [2L^2m_km_n + J_n(2m_k + m_n)]R^2} \tag{9.37}$$

Zatim se raspisuje diferencijalna jednadžba u obliku prostora stanja:

$$\dot{x} = Ax + Bu \tag{9.38}$$

$$y = Cx + Du \tag{9.39}$$

x - vektor stanja

ẋ - vektor derivacije varijabli stanja

u - vektor ulaza

y - vektor izlaza

A - matrica koeficijenata sustava

B - matrica ulaza sustava

C - matrica izlaza sustava

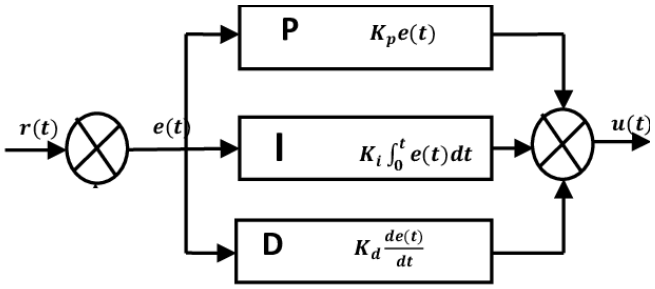
D - matrica direktnog preslikavanja ulaza na izlaz

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dot{\Theta} \\ \ddot{\Theta} \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & a_{22} & a_{23} & 0 \\ 0 & 0 & 0 & 1 \\ 0 & a_{42} & a_{43} & 0 \end{bmatrix}}_{A} \begin{bmatrix} x \\ \dot{x} \\ \Theta \\ \dot{\Theta} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ b_2 \\ 0 \\ b_4 \end{bmatrix}}_{B} u, \tag{9.40}$$

$$\begin{bmatrix} x \\ \Theta \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{C} \begin{bmatrix} x \\ \dot{x} \\ \Theta \\ \dot{\Theta} \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}_{D} u. \tag{9.41}$$

10. PODEŠAVANJE PID REGULATORA

Proporcionalno-integralno-derivacijski regulator (PID regulator) je regulator koji se često koristi u industrijskim sustavima upravljanja i raznim drugim aplikacijama koje zahtijevaju kontinuirano modulirano upravljanje. PID regulator kontinuirano izračunava vrijednost pogreške kao razliku između željene zadane vrijednosti i izmjerene procesne vrijednost i primjenjuje korekciju temeljenu na proporcionalnoj, integralnoj i derivacijskoj vrijednosti.



Slika 10.1. Shema PID regulatora [15]

Glavna prijenosna funkcija PID regulatora ima oblik:

$$u(t) = K_p\, e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} = K_p(e(t) + \frac{1}{Ti} \int_0^t e(t)dt + T_d \frac{de(t)}{dt}) \qquad (10.1)$$

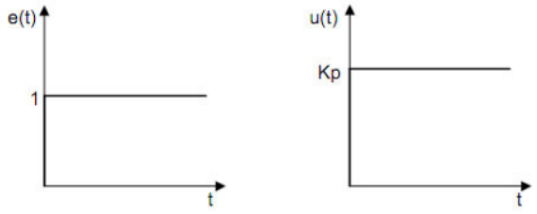
Izostavljanjem pojedinih članova PID regulatora dobivaju se jednostavnije strukture ostalih regulatora.

P regulator je najjednostavnija vrsta regulatora opisana jednadžbom:

$$u(t) = K_p * e(t) \qquad (10.2)$$

Kp predstavlja faktor proporcionalnog djelovanja ili pojačanja regulatora, a e(t) je signal greške. Svaki proporcionalni regulator karakterizira proporcionalni raspon koji je definiran kao potreban postotak promjene ulazne veličine da bi se izlazna veličina promijenila za 100%. Proporcionalna veličina također se može definirati kao recipročna vrijednost pojačanja Kp (%). Povećanjem pojačanja Kp smanjuje se konstantno odstupanje regulirane veličine od zadane vrijednosti. Istodobno se povećava brzina odziva.

Slika prikazuje djelovanje P regulatora u(t) ako se na njegov ulaz dovede signal greške e(t) u obliku skokovite funkciji.



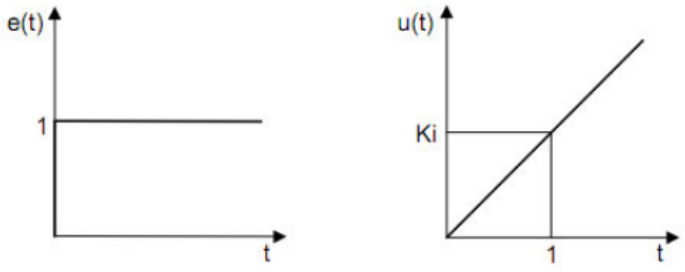
Slika 10.2. Djelovanje P regulatora [10]

I (Integralno djelovanje) regulatora je opisano jednadžbom: koja proporcionalno povezuje pogrešku e(t) s brzinom promjene kontroliranje varijable u(t). Recipročna vrijednost pojačanja Ki je konstanta Ti i predstavlja integralno vrijeme djelovanja (vrijeme integracije).

$$u(t) = K_i \int_0^t e(t)dt \tag{10.3}$$

$$K_i = \frac{1}{T_i} \tag{10.4}$$

Uvođenjem integralnog regulatora povećava se inertnost sustava, odnosno sustav sporije reagira na vanjske utjecaje, ali zato u većini slučajeva trajno otklanja grešku rada sustava u stacionarnom stanju. Negativna značajka ovog tipa regulatora je destabilizirajući učinak u sustavu zbog njegovog kašnjenja.



Slika 10.3. Djelovanje I regulatora [10]

Slika prikazuje rad I regulatora, ako se na njegov ulaz primijeni signal greške e(t) u obliku jedinične skokovite funkcije.

D (diferencijalno djelovanje) regulator:

$$u(t) = K_d \frac{de(t)}{dt} \tag{10.5}$$

Samostalno postojanje diferencijalnog regulatora nema previše smisla, jer je u uspostavljenom režimu rada signal greške konstantan, a derivacija tog signala jednaka je nuli. Zbog svojstva da je varijabla u(t) proporcionalna brzini promjene (prvi izvod) pogreške u vremenu, vidi se da bi D regulator reagirao samo na brze promjene, a spore i dugotrajne promjene ne bi imale značajan utjecaj na reguliranu veličinu pa se i zbog toga razloga rijetko koristi samo D regulator sam po sebi. Kombinacijom s P i/ili I regulatorom ovaj regulator dobiva na značaju, posebno u prijelaznom režimu rada sustava. Njegovo postojanje omogućuje bolje praćenje dinamike sustava, jer prati veličinu promjene pogreške, a ne samo njenu apsolutnu vrijednost. Uvođenjem diferencijalnog regulatora povećava se stabilnost i brzina odziva sustava.

10.1 Ziegler-Nichols metoda ruba stabilnosti

Popularna metoda za podešavanje P, PI i PID regulatora je Ziegler-Nicholsova metoda. Ova metoda počinje postavljanje integralnog i diferencijalnog pojačanja na nulu i zatim povećanjem proporcionalnog člana sve dok sustav ne postane nestabilan. Tako dugo povećavamo vrijednost proporcionalnog pojačanja dok se ne pojave trajne oscilacije sustava. Vrijednost K_P u točki nestabilnosti gdje sustav ima trajne oscilacije naziva se K_{Rkr}.

Nakon određivanja točke oscilacije $K_{R\,kr}$ mora se izmjeriti iznos perioda trajnih oscilacija. Taj iznos predstavlja kritični iznos periode oscilacije T_{kr}. Na temelju dobivenih vrijednosti $K_{R\,kr}$ i T_{kr} , upisuju se u formule iz tablice i izračunavaju se iznosi proporcionalnog pojačanja (Kp) i vremenskih konstanti (T_I ,T_D) PID regulatora.

Tip regulatora	Vrijednosti parametara regulatora		
	K_R	T_I	T_D
P	$0{,}5\,K_{R\,kr}$	-	-
PI	$0{,}45\,K_{R\,kr}$	$0{,}85\,T_{kr}$	-
PID	$0{,}6\,K_{R\,kr}$	$0{,}5\,T_{kr}$	$0{,}12\,T_{kr}$

Tablica 10.1. Ziegler-Nichols-ova tablica [12]

U tablici se koristi oznaka K_R za vrijednost proporcionalnog pojačanja PID regulatora. Oznaka K_R se u prijašnjem tekstu odnosi na vrijednost Kp i dalje se u završnome radu koristi oznaka Kp za vrijednost proporcionalnog pojačanja.

10.2 Primjena Zigler-Nichols metode ruba stabilnosti na sustav

Da bi se saznala kritična vrijednost $K_{R\,kr}$ morala se vrijednost Kp povećavati unutar Arduino koda tako dugo dok sustav nije počeo oscilirati. Nakon više pokušaja sustav je počeo oscilirati kada mu je bila zadana vrijednost Kp od 226.66 , a oscilacije su trajale 1624 ms. Preko tih vrijednosti ralunaju se potrebni Kp, Kd i Ki parametrei.

Prema Ziegler-Nichols tablici provode se sljedeći proračuni:

Izračun proporcionalnog parametra Kp:

$$Kp = 0.6 * K_{R\,kr} = 0.6 * 226.66 = 136 \tag{10.5}$$

Izračun integralnog parametra K_i :

$$K_i = \frac{1}{T_I} = \frac{1}{0.5 * 1624\ ms} = 0.0012315\ ms \tag{10.6}$$

Izračun derivacijskog parametra K_D:

$$K_D = 0.12 * T_{t\,kr} = 0.12 * 1624\ ms = 194.88\ ms \tag{10.7}$$

Uzimaju se nove vrijednost parametra i upisuju su u Arduino programski kod. Nakon testiranja se zaključuje da te vrijednosti donose najbolju stabilnost sustava.

10.3 Postavljanje PID vrijednosti u Arduino programski kod

Programski kod je bio preuzet [18] i samo su se dodavali potrebni paketi (libary) za rad koda, mijenjali su se pinovi unutar Arduino programskog koda koje mikrokontroler Arduino UNO koristi, i mijenjale su se vrijednosti varijabli za PID regulator i MPU-6050 vrijednost nagiba da odgovaraju 0° nagiba na početku rada samoga robota.

Prije samog upisivanja dobivenih vrijednosti u kod moramo spojiti sve potrebne pakete Arduino koda potrebne za rad glavnog koda. Glavni korišteni paketi su: paket za PID regulaciju, paket za Kalmanov filtar za PID regulator, paket za čitanje signala s MPU6050 žiroskopa i akcelerometra i paket za upravljanje motorima.

```
#include <PID_v1.h>

#include <LMotorController.h>

#include "I2Cdev.h"

#include "MPU6050_6Axis_MotionApps20.h"

#include "KalmanFilter-master"
```

Nakon izračunavanja Kp, Ki, Kd vrijednosti upisuju se nove vrijednosti u programski kod. Vrijednosti se upisuju u double tip varijable, to jest varijable imaju veličinu od 8-byta (64-bita) i postavlja se početna točka u 0 (setpoint = 0).

```
//PID postavke

double originalSetpoint = 0; // referentna točka robota

double setpoint = originalSetpoint;

double movingAngleOffset = 0.1;

double input, output;

//Regulacija PID parametri

double Kp = 136;

double Kd = 194.88;

double Ki = 0.0012315;

PID pid(&input, &output, &setpoint, Kp, Ki, Kd, DIRECT);
```

11. KALMANOV FILTAR

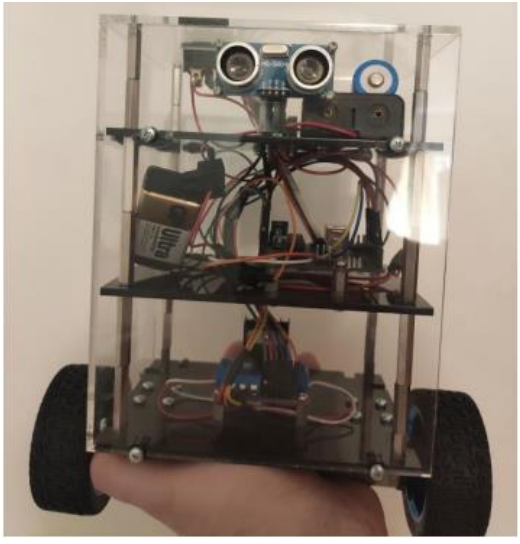
Za određivanje trenutnog nagiba i regulaciju robota koristi se MPU6050 akcelerometar i žiroskop. Njihov princip radi na temelju jednostavnih fizikalnih teorijama kao što su: Coriolisov efekt u slučaju žiroskopa i teorija inercije mase u slučaju akcelerometra. Žiroskop služi za određivanje kutne brzine robota, koji pokazuje visoku točnost i zanemarivu vrijednost šuma. Integriranjem te vrijednosti pomoću Arduino UN-a se može dobiti apsolutni nagib. Za dobivanje kuta vidi se da je potreban samo žiroskop, ali zbog greške integracije dugoročno se dobiva posve kriva vrijednost kuta. Zato se mora sustavu dodati akcelerometar kojim se mjeri iznos ubrzanja u pojedinim osima. Njime se određuje smjer vektora ubrzanja gravitacijskog polja zemlje to jest određuje pod kojim kutom je središte zemlje u odnosu na robot. Na akcelerometar, za razliku od žiroskopa, jako utječu vibracije koje stvaraju smetnje, pa su kratkoročno nepouzdani. Kalmanov filtar se koristi da bi se izbjegao drift žiroskopa i uklonio utjecaj smetnji u akcelerometrima.[10]

Kalmanov filtar je rekurzivni procjenitelj stanja koji omogućuje da se na efikasan način procijene stanja procesa, tako da se minimizira srednja kvadratna pogreška. Filtar podržava procjenu prošlih, sadašnjih te budućih stanja sustava i to radi čak i kada je točno ponašanje modeliranog sustava nepoznato. Svrha Kalmanovog filtra je filtriranje uz pomoć poznatog modela sustava i uz prisutan izražen šum u mjerenju senzora, točno određivanje željenih varijabli stanja promatranog procesa.[9]

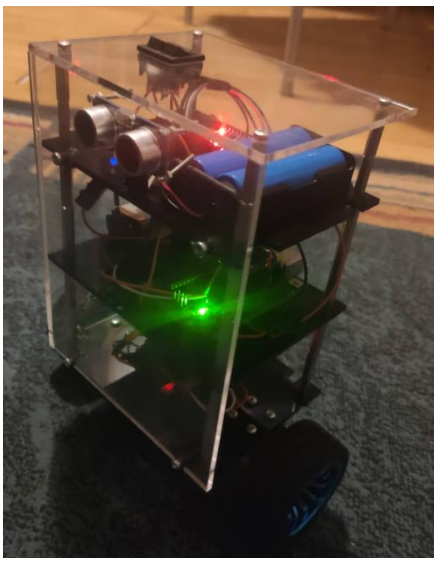
Kalmanov filtar je korišten kao paketu (libary) u Arduino programskom kodu. Paket koristi gotov dimenzioniran filtar za PID regulaciju. [16]

12. OPIS RADA SAMOBALANSIRAJUĆEG ROBOTA

Rad samobalansirajućeg robota je kompliciran jer mnogo faktora utječu na njegov rad. Rad robota započinje pritiskom na ON/ON sklopku koja predstavlja zapravo dvije ON/OFF sklopke. Jedna sklopka upravlja napajanjem Arduino UNO mikroupravljača, a druga sklopka napajanjem H-most L298N motor drivera. Rad robota započinje tek nakon dvije sekunde jer je to vrijeme potrebno da MPU6050 počinje učitavati vrijednosti nagiba robota i tek onda kada su vrijednosti stabilizirane započinje rad motori. Ovisno o kutu nagiba robota Arduino UNO regulira brzinu okretaja motora. Što je veći kut nagiba to je potrebna veća brzina motora da robot izvrši korekciju, a ako je manji kut onda je potrebno vrlo mala korekcija brzine robota. Robot koristi PID regulaciju čije se vrijednosti reguliraju unutar IDE koda. PID regulacija osigurava puno bolji i stabilniji sustav. Prije početka reguliranja sustava trebalo je postaviti *setpoint* (referentnu) točku prema kojoj se regulacijski sustav upravlja. *Setpoint* vrijednost je jednaka vrijednosti MPU6050. Regulacija se vrši preko promjene Kp, Kd i Ki parametra. Prvo je bila potrebna regulacije Kp (proporcionalni rast) vrijednosti. Promjenom toga parametra robot počinje balansirati, ali balansiranje ima vrlo visoke oscilacije što je nepovoljno. Premala Kp vrijednost i robot nema dovoljno korekcije, a previše Kp vrijednosti robot ima previše korekcije. Nakon dostizanje prihvatljivog balansiranja robota promjenom Kp vrijednosti s minimalno oscilacija postavlja se Kd (rast od deriavcije) parametra. Kd vrijednost služi za smanjivanje oscilacija. Krivo postavljeni Kd će povećati broj oscilacija zato je precizna regulacije tog parametra potrebna. Nakon što se postignu željene vrijednosti Kp i Kd nastavlja se podešavanjem Ki vrijednosti. Ki (integralni rast) je vrijednost koja omogućuje brže reagiranje robota na nagibe i nije nužno potrebna za rad robota. Ako je vrijednost premala robot će sporije reagirati, a ako je prevelika robot će postat nestabilan. Potrebna je precizna regulacija toga parametra. Tu preciznost se može dobiti i korištenjem Zigler-Nichols-ove metode za postizanje tih parametara.



Slika 11.1.Samobalansirajući robot



Slika 11.2.Samobalansirajući robot u radu

13. ZAKLJUČAK

Izradom robotskih samobalansirajućih kolica prikazuje se mogućnost pretvaranja nestabilnog sustava u stabilan pomoću PID regulacije.

Odabrani istosmjerni motori nemaju dovoljno veliku brzinu da naprave korekciju za veće stupnjeve nagiba i zbog toga je potrebno lagano utjecati na gibanje robota. Zamjenom motora se može postići veća preciznost balansiranja, a rješavanje problema moguće je i postavljanjem većih kotači za bolje balansiranje.

Iz matematički model samobalansirajućeg robota se može vidjeti koliko faktora treba unaprijed proračunati da bi se mogla napraviti linearizacija sustava koja je potrebna za proračunavanje i odabir regulatora koji bi se koristio za stabiliziranje samobalansirajućeg robota. U radu je prikazana linearizacija za opći oblik sustava samobalanisirajućeg robota.

Vrijednosti PID regulatora su bile određene preko Zigler-Nichols-ove metode ruba stabilnosti jer je tom metodom bilo moguće najlakše doći to Kp (faktor proporcionalnog djelovanja), Ki (integralnog djelovanje) i Kd (diferencijalno djelovanje), a da se robot balansira. Ta metoda se pokazala dovoljno dobra da sustav balansira, ali nije bio potpuno stabilan već je imao male oscilacije. Problem je bio u tome, kao što je prije bilo navedeno, da motori, zbog premale brzine i premalog okretnog momenta, nedovoljno brzo reagiraju na promjene nagiba samobalanisirajućeg robota.

Iako je samobalanisirajući robot uspješno izrađen i uspješno se balansira potrebno je još mnogo rada na njemu da mu se smanje oscilacije to jest poveća stabilnost. To se može uspjeti s boljim razumijevanjem postupka linearizacije robota kako bi se mogao odabrati i bolji regulator i primjerenije komponente samog robota.

14. LITERATURA

[1] Arduino**,** https://www.arduino.cc/ (03.05.2022.)

[2] Arduino IDE, https://www.arduino.cc/en/Main/Software (3.05.2022.)

[3] SolidWorks, https://www.solidworks.com/ (03.05.2022.)

[4] Arduino UNO, https://components101.com/microcontrollers/arduino-uno (3.05.2022.)

[5] MPU6050, www.alldatasheet.com/Mpu-6050 (3.05.2022.)

[6] L298n, www.alldatasheet.com/L298n (3.05.2022.)

[7] Faragher, Ramsey. 2012. Understanding the Basis of the Kalman Filter Via a Simple and Intuitive Derivation.pdf (12.07.2022)

[8] https://www.sciencedirect.com/topics/computer-science/ziegler-nichols-method (25.08.2022)

[9]] Lauszus, Kristian. 2012. Kalman Filter https://github.com/TKJElectronics /KalmanFilter (06.8.2022)

[10] Jose' Luis Corona Miranda. Application of Kalman filtering and PID control for direct inverted pendulum control. Faculty of California State University, Chico, 2009. (25.08.2022)

[11] Merlin: AUP - PID regulator.pdf (25.08.2022)

[12] Merlin: AUP - Sinteza sustava upravljanja u vremenskom podrucju.pdf(25.08.2022)

[13] R. C. Hibbeler, Engineering Mechanics Dynamics, 12th ed., Upper Saddle River, New Jersey: Pearson Prentice Hall, 2010 https://scholarworks.uark.edu/cgi/viewcontent.cgi?article=1092&context=meeguht

(15.07.2022)

[14] Arduino Uno: https://www.arduino.cc/

[15] https://www.researchgate.net/figure/The-Model-of-a-PID-Controller-Configuration-From-Figure-3-the-mathematical-expression_fig3_344378374

(25.08.2022)

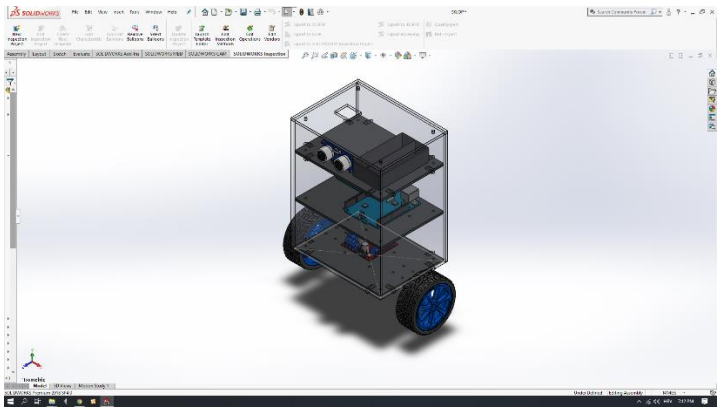
[16] https://www.arduino.cc/reference/en/libraries/kalman-filter-library/ (15.05.2022.)

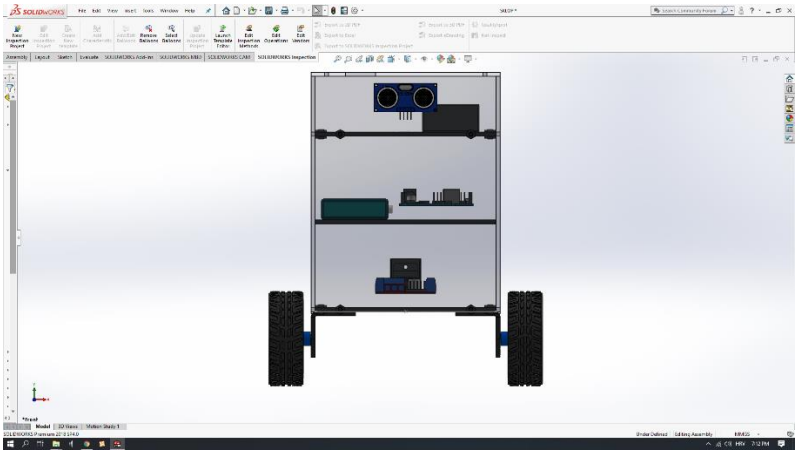
[17] D.J. Block, K.J. Astrom, and M.W. Spong. The Reaction Wheel Pendulum. Morgan and Claypool Publishers, University of Illinois at Urbana-Champaign, 2007 – pdf (25.08.2022)

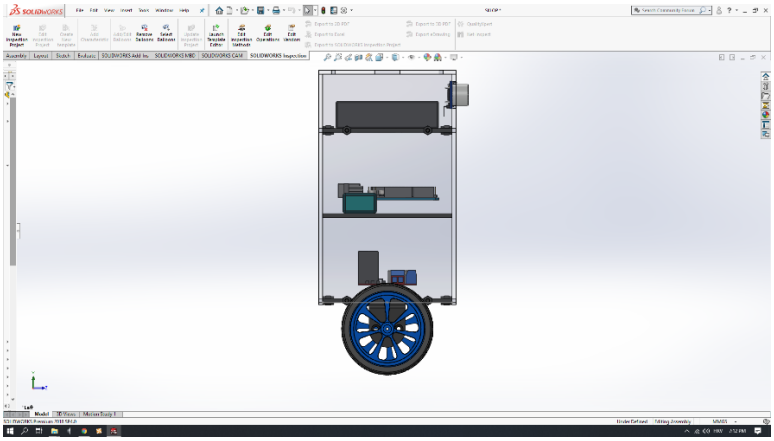
[18] https://github.com/topics/self-balancing-robot (06.05.2022.)

15. PRILOZI

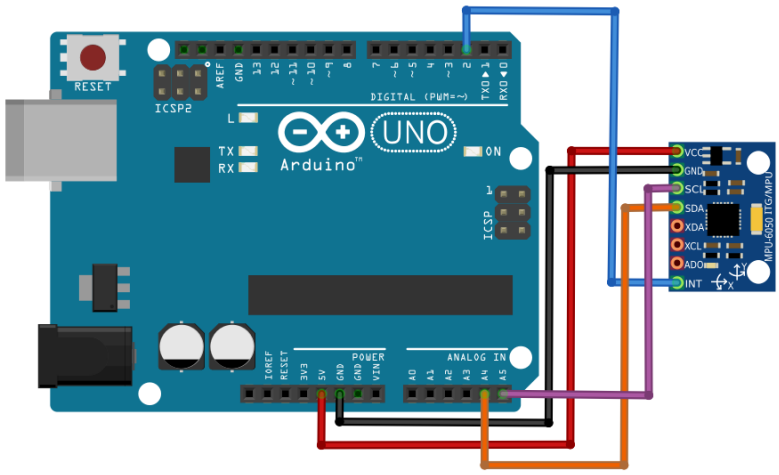
15.1 GOTOV MODEL ROBOTA U SOLIDWORKSU



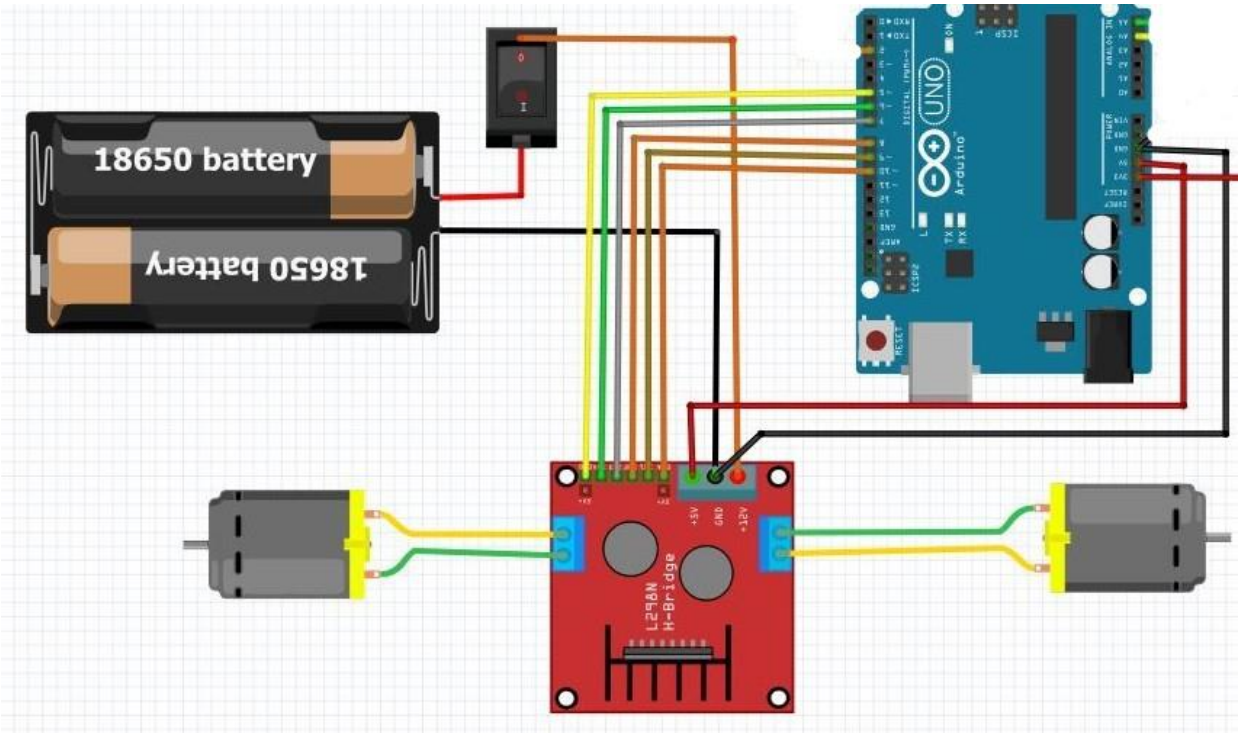




15.2 SHEMA SPAJANJA MPU6050 S ARDUINO UNO



15.3 SHEMA SPAJANJA L298n MODULA S ARDUINO UNO, MOTORIMA I NAPAJANJEM



15.4 Kod izrađen u programu „Arduino IDE“

```
#include <PID_v1.h>
#include <LMotorController.h>
#include "KalmanFilter-master"
#include "I2Cdev.h"
#include "MPU6050_6Axis_MotionApps20.h"
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
 #include "Wire.h"
#endif
#define MIN_ABS_SPEED 30
MPU6050 mpu;
// MPU kontrola/status varijabli
bool dmpReady = false;
uint8_t mpuIntStatus;
uint8_t devStatus;
uint16_t packetSize;
uint16_t fifoCount;
uint8_t fifoBuffer[64];
// varijable orijentacije/pozicije
Quaternion q;
VectorFloat gravity;
float ypr[3];
//PID postavke
double originalSetpoint = 0; // referentna točka robota
double setpoint = originalSetpoint;
double movingAngleOffset = 0.1;
double input, output;
//Regulacija PID parametri
double Kp = 136;
double Kd = 194.88;
double Ki = 0.0012315;
PID pid(&input, &output, &setpoint, Kp, Ki, Kd, DIRECT);
```

```
//Upravljanje brzinom motora i njihovo sinkroniziranje

double motorSpeedFactorLeft = 0.8;

double motorSpeedFactorRight = 0.7;

//Kontrola Motora

int ENA = 5; //siva

int IN1 = 9; //crvena

int IN2 = 8; //ljubičasta

int IN3 = 7; //crna

int IN4 = 6; //plava

int ENB = 10; //narančasta

LMotorController motorController(ENA, IN1, IN2, ENB, IN3, IN4,
motorSpeedFactorLeft, motorSpeedFactorRight);

volatile bool mpuInterrupt = false; // Indikator kada se MPU interrupt PIN
ukinuo u stanju HIGH

void dmpDataReady()

{

 mpuInterrupt = true;

}

void setup()

{

 // Spajanje na I2C bus (I2Cdev library ne učini to automatski)

 #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE

 Wire.begin();

 TWBR = 24; // 400kHz I2C clock

 #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE

 Fastwire::setup(400, true);

 #endif

 mpu.initialize();

 devStatus = mpu.dmpInitialize();

 // Postavljane vrijednosti  offseta žiroskopa nakon kalibracije

 mpu.setXGyroOffset(220);

 mpu.setYGyroOffset(76);

 mpu.setZGyroOffset(-85);

 mpu.setZAccelOffset(1788);
```

```
if (devStatus == 0)
 {
 mpu.setDMPEnabled(true);
 // dopusti Arduinu da koristi interrupt funkciju
 attachInterrupt(0, dmpDataReady, RISING);
 mpuIntStatus = mpu.getIntStatus();
 /
 dmpReady = true;
 packetSize = mpu.dmpGetFIFOPacketSize();
 //PID postavke
 pid.SetMode(AUTOMATIC);
 pid.SetSampleTime(10);
 pid.SetOutputLimits(-255, 255);
 }
 else
 {
 Serial.print(F("DMP Initialization failed (code "));
 Serial.print(devStatus);
 Serial.println(F(")"));
 }
}
void loop()
{
 if (!dmpReady) return;
 while (!mpuInterrupt && fifoCount < packetSize)
 {
 // PID kalkulacija za upravljanje motorima
 pid.Compute();
 motorController.move(output, MIN_ABS_SPEED);
 }
 // Resetiranje interrupt funkcije
 mpuInterrupt = false;
 mpuIntStatus = mpu.getIntStatus();
 fifoCount = mpu.getFIFOCount();
 if ((mpuIntStatus & 0x10) || fifoCount == 1024)
 {
 mpu.resetFIFO();
 Serial.println(F("FIFO overflow!"));
```

```
{
 mpu.resetFIFO();
 Serial.println(F("FIFO overflow!"));
 }
 else if (mpuIntStatus & 0x02)
 {
 while (fifoCount < packetSize) fifoCount = mpu.getFIFOCount();
 mpu.getFIFOBytes(fifoBuffer, packetSize);
 fifoCount -= packetSize;
 mpu.dmpGetQuaternion(&q, fifoBuffer);
 mpu.dmpGetGravity(&gravity, &q);
 mpu.dmpGetYawPitchRoll(ypr, &q, &gravity);
 input = ypr[1] * 180/M_PI + 180;
 }
}
```

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL: Odjel za mehatroniku

STUDIJ: preddiplomski stručni studij Mehatronika

PRISTUPNIK: Filip Radas | MATIČNI BROJ: 0336036684

DATUM: 29.08.2022. | KOLEGIJ: Automatsko upravljanje

NASLOV RADA: Samobalansirajući robot na dva kotača

NASLOV RADA NA ENGL. JEZIKU: A self-balancing robot on two wheels

MENTOR: Miroslav Horvatić, dipl. ing. | ZVANJE: viši predavač

ČLANOVI POVJERENSTVA:
1. mr. sc. Ivan Šumiga, dipl. ing., viši predavač
2. Josip Srpak, dipl. ing., viši predavač
3. Miroslav Horvatić, dipl. ing., viši predavač
4. dr. sc. Dunja Srpak, docent, rezervni član
5.

Zadatak završnog rada

BROJ: 009/MEH/2022

OPIS:

Potrebno je osmisliti i izraditi samobalansirajući robot na dva kotača. Obzirom da navedeni robot predstavlja nestabilan mehanički sustav potrebno je odrediti i realizirati strukturu i parametre regulatora koji će osigurati njegovo stabilno ponašanje.

U radu je potrebno:
- osmisliti samobalansirajući robot te odabrati, izraditi i smisleno povezati njegovo sklopovlje u funkcionalnu cjelinu
- objasniti matematički model samobalansirajućeg robota
- osmisliti, realizirati i podesiti regulator koji će stabilizirati robot
- ispitati rad samobalansirajućeg robota upravljanog realiziranim regulatorom.

REPUBLIKA HRVATSKA
SVEUČILIŠTE SJEVER

ZADATAK URUČEN: 30.08.2022. | POTPIS MENTORA: Horvatić

Sveučilište
Sjever

SVEUČILIŠTE
SJEVER

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, Filip Radoš *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Samobalansirajući robot na dva kotača *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)

Radoš
(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, Filip Radoš *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog (*obrisati nepotrebno*) rada pod naslovom Samobalansirajući robot na dva kotača *(upisati naslov)* čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)

Radoš
(vlastoručni potpis)