

ŽILINSKÁ UNIVERZITA V ŽILINE

ELEKTROTECHNICKÁ FAKULTA

BAKALÁRSKY PROJEKT

DANIEL ADAMKOVIČ

Návrh a konštrukcia dvojkolesového balansujúceho robota

Vedúci práce: Ing. Dušan Nemec, PhD

Evidenčné číslo: Evidenčné číslo práce

Žilina, 2019

ŽILINSKÁ UNIVERZITA V ŽILINE

ELEKTROTECHNICKÁ FAKULTA

BAKALÁRSKY PROJEKT

DANIEL ADAMKOVIČ

Návrh a konštrukcia dvojkolesového balansujúceho robota

Študijný odbor: 5.2.14 Automatizácia

Školiace pracovisko: Žilinská univerzita v Žiline, Elektrotechnická fakulta

Katedra riadiacich a informačných systémov

Vedúci práce: Ing. Dušan Nemeč, PhD

Konzultant: Titul, konzultant práce

Žilina, 2019

Namiesto tejto strany treba vložiť zadanie záverečnej práce!

Do elektronickej verzie práce vložte **oskenované podpísané** zadanie záverečnej práce, napr. vo formáte pdf alebo ako obrázok zväčšený na celú veľkosť papiera.

Abstrakt

Abstrakt obsahuje informáciu o cieľoch práce, jej stručnom obsahu a v závere abstraktu sa charakterizuje splnenie ciela, výsledky a význam celej práce. Abstrakt sa píše súvisle ako jeden odsek a jeho rozsah je spravidla 100 až 500 slov.

Kľúčové slová: robotika, mikropočítač, riadenie, regulácia

Abstract

In this place insert the text of the abstract in English or another foreign language. Sem vložte text abstraktu v angličtine, prípadne v inom cudzom jazyku.

Keywords: robotics, microcomputer, control, regulation

OBSAH

| | |
|--|------|
| Zoznam skratiek | vii |
| Slovník pojmov | viii |
| 1 Úvod | 1 |
| 2 Prehľad existujúcich metód riadenia pre Dvojkolesové balansujúce roboty | 2 |
| 2.1 Lineárne regulátory | 3 |
| 2.1.1 PID | 3 |
| 2.1.2 LQR | 7 |
| 2.2 Zhrnutie a výber regulátora | 11 |
| 3 Analýza dynamiky balansujúceho robota | 12 |
| 3.1 Model kolesa | 12 |
| 3.2 Model šasi | 14 |
| 4 Návrh a realizácia konštrukcie dvojkolesového robota | 17 |
| 4.1 Zoznam použitých komponentov | 17 |
| 4.1.1 Napájanie | 18 |
| 4.1.2 Pohon | 19 |
| 4.1.3 Komunikácia - Bluetooth modul | 22 |
| 4.1.4 Senzory | 23 |
| 4.1.5 Riadiaci počítač - Arduino MEGA | 24 |

| | | |
|-----------------------|---|------------|
| 4.2 | Návrh a výroba šasi robota | 25 |
| 4.3 | Výsledky montáže a zhodnotenie návrhu | 29 |
| 4.4 | Návrh a výroba ovládača robota | 30 |
| 5 | Návrh a implementácia riadiaceho systému robota. | 32 |
| 5.1 | Štruktúra RS robota | 32 |
| 5.2 | Implementácia riadiaceho systému | 34 |
| 5.3 | Ovládanie pohybu robota | 37 |
| 5.4 | Podporné metódy riadenia | 39 |
| 6 | Laboratórne overenie funkčnosti | 41 |
| 6.1 | Základné parametre robota | 41 |
| 6.2 | Parametre robota pri balansovaní na mieste | 42 |
| 7 | Zhrnutie dosiahnutých výsledkov | 44 |
| Prílohová časť | | 46 |
| A | Ziegler-Nicholsova tabuľka | I |
| B | Schéma shieldu | II |
| C | Schéma H-mostíka | III |
| D | Diagram programu | IV |
| E | Výstup z enkodéra | V |
| F | Snímanie prúdu | VI |

ZOZNAM OBRÁZKOV

| | | |
|------|--|----|
| 2.1 | Zapojenie PID | 3 |
| 2.2 | Štruktúra PID regulátora | 4 |
| 2.3 | Stavový priestor - diagram | 7 |
| 2.4 | Stavový priestor - spätná väzba | 9 |
| 3.1 | Model kolesa | 13 |
| 3.2 | Model šasi | 14 |
| 4.1 | Li-ion batéria | 18 |
| 4.2 | Spínaný napäťový regulátor kompatibilný s LM2596 | 19 |
| 4.3 | Platforma s motormi | 20 |
| 4.4 | H-mostík komaptibilný s L298 | 21 |
| 4.5 | Servo motor HJ S3315D | 22 |
| 4.6 | HC05-Bluetooth modul | 23 |
| 4.7 | MPU6050 | 24 |
| 4.8 | Arduino MEGA | 25 |
| 4.9 | Balansujúci robot s osadenými motormi | 26 |
| 4.10 | Pred osadením shieldu | 27 |
| 4.11 | Po osadení shieldu | 27 |
| 4.12 | Robot spredu | 29 |
| 4.13 | Robot zboku | 29 |
| 4.14 | Ovládač balansujúceho robota | 30 |

| | | |
|-----|--|-----|
| 5.1 | Schematické zapojenie PID regulátorov | 32 |
| 5.2 | Znázornenie závislosti medzi striedou a napäťím | 33 |
| 5.3 | Podrobná schéma RS robota | 37 |
| 5.4 | Upravený RS | 40 |
| 6.1 | Náklon robota v čase | 42 |
| 6.2 | Zmena polohy robota počas testu | 43 |
| B.1 | Schéma shieldu pre Arduino | II |
| C.1 | Schematické znázornenie H-mostíka | III |
| D.1 | Diagram programu. | IV |
| E.1 | Výstup z kanálov enkodéra | V |
| F.1 | Shield s vyznačeným rezistorom na snímanie prúdu | VI |

ZOZNAM TABULIEK

| | | |
|-----|---------------------------|----|
| 4.1 | Parametre batérie | 18 |
| A.1 | Zieger-Nicholsova tabuľka | I |

ZOZNAM SKRATIEK

FPS počet snímok za sekundu (angl. frames per second), s. 34

LQR logaritmicko-kvadratický regulátor (angl. logarithmic quadratic regulator), s. 3

PID proporčno derivačný integračný regulátor (angl. proportional integral derivative regulator), s. 3–6

PWM pulzne šírková modulácia (angl. pulse width modulation), s. 21, 35

RPM revolutions per minute (slov. otáčky za minútu), s. 20

RS Označenie riadiaceho systému robota, s. 32–36, 38

SISO Single-Input Single-Output, s. 10

SLOVNÍK POJMOV

Slovník pojmov: Slovník pojmov je nepovinný. Na jeho odstránenie stačí zmazať všetky zadefinované pojmy v súbore modules/abbterms.tex.

Triedenie: Pojmy v slovníku sa automaticky triedia podľa abecedy. Ale pozor: triedenie sa deje prvého argumentu makra `DeclareAcronym` – nie podľa poľa `short`.

Viskozita: Fyzikálna veličina, miera odporu tekutiny deformovať sa pod vplyvom šmykových (tangenciálnych) napäťí. Preja-

vuje sa vnútorným trením.

Zhlukovanie: Trieda metód strojového učenia, ktoré v daných dátach hľadajú zhluky.

Hierarchické zhlukovanie

Metódy zhlukovania, kde rozdelenie do zhlukov má hierarchickú štruktúru.

Fuzzy c-means zhlukovanie

Verzia algoritmu k-means pre fuzzy zhlukovanie.

1 | ÚVOD

Dvojkolesové balansujúce roboty predstavujú v rámci robotických systémov zaujímavú skupinu robotov, ktorá účelovo predstavuje určitý medzikrok medzi klasickými, inherentne stabilnými systémami na kolesách a bipedálnymi robotmi napodobňujúcimi spôsob chôdze ľudí.

Avšak, kým klasické viac-kolesové roboty sa vyznačujú výbornými vlastnosťami ako v oblasti stability tak aj rýchlosťi pohybu, vo všeobecnosti sú väčšie a zložitejšie (a teda aj drahšie) ako dvojkolesové roboty navrhnuté pre rovnaký účel. Výbornou ukážkou je napríklad populárny Segway, pri ktorom výrobca efektívne využil balansovanie na dvoch kolesách, bez obmedzenia užitočnosti produktu.

Naproti tomu návrh, naprogramovanie a skonštruovanie robotických systémov s umelými nohami je v súčasnosti stále problém vyžadujúci nasdanie komplexných, ťažko nalaďiteľných regulátorov. Tieto systémy sa tak väčšinou javia ako príliš drahé, nespoľahlivé a pomalé na nasadenie v praxi.

My sa v tejto práci budeme zaoberať návrhom, analýzou a konštrukciou modulárneho, balansujúceho robota, na ktorom demonštrujeme vyššie popísané vlastnosti.

2 | PREHĽAD EXISTUJÚCICH METÓD RIADE-NIA PRE DVOJKOLESOVÉ BALANSUJÚCE ROBOTY

Balansujúci robot predstavuje z mechanického hľadiska inherentne nestabilnú sústavu, ktorú je pre to, aby bol takýto robot v praxi použiteľný, potrebné stabilizovať pomocou regulátora. V už existujúcich prácach sa stretávame s rôznymi druhmi implementovaných regulátorov a je teda namiesto uviesť aspoň niektoré z najčastejšie používaných. V nasledujúcej časti práce sa teda budeme zaoberať stručným popisom v praxi používaných regulátorov a uvedieme ako sme sa rozhodovali pri výbere regulátora my.

Vo všeobecnosti môžeme ako regulátor označiť každé zariadenie, ktoré v systéme zabezpečuje udržiavanie určitých fyzikálnych veličín na stanovených úrovniach. V priebehu regulácie sa pravidelne zistuje skutočný stav objektu a porovnáva sa s požadovaným. Regulátor následne upravuje stav systému tak, aby bol dosiahnutý požadovaný cieľ.

Jedným zo základných spôsobov rozdelenie regulátorov je na lineárne a nelineárne regulátory. Lineárne regulátory sú určené na riadenie sústav, ktorých prenosové funkcie sa vyznačujú tým, že pre ne platí princíp homogenity a superpozície. Pokiaľ chceme použiť takýto regulátor na riadenie nelineárnej sústavy, t.j. sústavy, pre ktorú neplatí princíp homogenity alebo superpozície, bude takýto regulátor pracovať korektnie len ak sústava zotrvá v okolí bodu, kde je možné nájsť jej lineárnu approximáciu. Tento proces sa nazýva linearizácia.

Pri nelineárnych regulátoroch táto potreba linearizácie odpadá, keďže regulátory tohto typu dokážu pracovať aj s nelineárnymi sústavami. Nevýhodou práce s nelineárnymi sústavami je ale vyššia náročnosť riešenia nelineárnych diferenčných rovníc. Práve kvôli tomuto

problému existuje v praxi tendencia radšej hľadať spôsoby ako čo najpresnejšie reprezentovať nelineárne systémy lineárnymi diferenčnými rovnicami a následne použiť na ich riadenie lineárny regulátor. Je ale nutné ešte podotknúť, že reálne sa pri zohľadnení všetkých vonkajších vplyvov každá sústava javí ako nelineárna.

Štúdiom prác, ktoré už boli napísané na tému riadenia dvojkolesového balansujúceho robota sme zistili, že medzi regulátory, ktoré sú najčastejšie pre túto úlohu používané patria lineárne regulátory:

1. PID (Proportional Integral Derivative Regulator;PID)
2. LQR (Linear Quadratic Regulator;LQR)

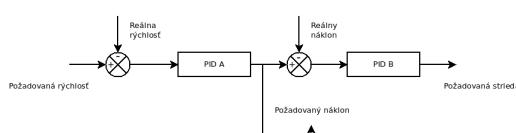
Práve týmito regulátormi sa teda budeme zaoberať v ďalšej podkapitole, no pre úplnosť ešte uvedieme, že vrámci nelineárnych regulátorov sa ako vhodné javia najmä Fuzzy PID a umelé neurónové siete.

2.1 Lineárne regulátory

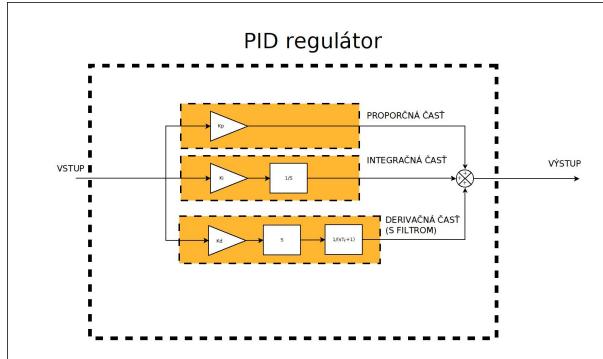
V tejto časti práce zhrnieme základné poznatky o niektorých vybraných typoch lineárnych regulátoroch. Uvedieme výhodné a nevýhodné vlastnosti jednotlivých regulátorov a v závere zhodnotíme, ktorý sa pre naše potreby javí ako najvhodnejší.

2.1.1 PID

PID regulátor je v praxi najčastejšie používaný regulátor, pričom uplatnenie nachádza pri riadení veličín ako sú napríklad: teplota, tlak, prietok, poloha, atď. Medzi jeho výhodné vlastnosti patrí najmä robustnosť, jednoduchosť implementácie a možnosť manuálneho naladenia bez potreby použitia zložitých výpočtových techník. Schematické znázornenie jeho zapojenia v ovláданej sústave je na Obr. 2.1



Obr. 2.1: Zapojenie PID



Obr. 2.2: Štruktúra PID regulátora

Zo schémy je zrejmé, že na vstup privádzame požiadavku na výstup sústavy a od tej následne odčítavame reálne nameranú hodnotu na výstupe (spätná väzba). Tako získavame chybovú veličinu $e(t)$, ktorá vyjadruje nakoľko sa reálna hodnota na výstupe líši od tej požadovanej. Práve s touto chybou ďalej pracuje PID regulátor.

Samotný PID regulátor pozostáva z troch častí, ktoré mu zároveň dávajú jeho názov: proporčnej (P), integračnej (I) a derivačnej (D). Každá s týchto častí iným spôsobom reaguje na vstup do PID a proces ladenia tak pozostáva z nastavenia príslušných parametrov (opísaných nižšie) pre jednotlivé tieto časti. Kombináciou rôznych hodnôt parametrov je možné regulátor nalaďiť tak, aby podľa potrieb používateľa kontroloval výstupnú veličinu.

Vychádzajúc z blokovej schémy na Obr. 2.2 môžeme podrobne opísť súčasti, z ktorých pozostáva PID.

Proporčný člen pozostáva zo zosilňovača, ktorý má na svojom výstupe K_p násobok chyby $e(t)$. V praxi to teda znamená, že ak je reálny výstup regulovanej sústavy voči požiadavke značne veľký → chyba je veľká a záporná, teda proporčný člen reaguje veľkým záporným výstupom. Tento následne zníži veľkosť výstupu a teda aj odchýlku od požadovanej hodnoty. Matematicky je možné vyjadriť výstup proporčného člena ako:

$$u_p(t) = K_p e(t) \quad (2.1)$$

kde $u_p(t)$ predstavuje výstup a $e(t)$ chybu na vstupe člena v čase t .

Prípadne je ešte možné použiť vyjadrenie v Laplaceovej rovine vo forme prenosovej funkcie:

$$\frac{U_p(s)}{E(s)} = K_p \quad (2.2)$$

kde $UP(s)$ je výstup a $E(s)$ chyba, vyjadrené v Laplaceovej rovine.

Integračný člen reaguje na súčet chýb, naakumulovaných od spustenia regulátora. Tento súčet je následne vynásobený konštantou K_i a prenesený na výstup integračného bloku. Hlavnou výhodou integračnej časti je, že umožňuje PID reagovať aj na veľmi malé konštantné chyby, ktoré by proporčný člen inak nebol schopný korigovať. Aj tá najmenšia chyba voči požiadavke sa totiž procesom integrovania hromadí, až kým je výstup integračného člena dostatočný na jej skorigovanie.

Matematické vyjadrenie integračného člena je:

$$u_i(t) = K_i \int_0^t e(t) dt \quad (2.3)$$

kde $u_i(t)$ predstavuje výstup integračného člena v čase t . Prenosová funkcia je:

$$\frac{U_i(s)}{E(s)} = K_i \frac{1}{s} \quad (2.4)$$

kde $U_i(s)$ je výstup vyjadrený v Laplaceovej rovine.

Derivačný člen v základnom zapojení pracuje so zmenou chyby za čas dt (ten sa v ideálnom prípadne limitne blíži k 0). Výhodnou vlastnosťou tohto člena teda je, že pri náhlych zmenách chyby je schopný pružne reagovať veľkou korekciou na výstupe a pri postupnom, pomalom narastaní chyby adekvátne malou korekciou. Zmenou konštanty Kd vieme korigovať silu tejto reakcie na zmenu chyby.

Matematické vyjadrenie derivačného člena je:

$$U_d(t) = K_d \frac{de(t)}{dt} \quad (2.5)$$

Kde $U_d(t)$ predstavuje výstup integračného člena v čase t .

Prenosová funkcia je:

$$\frac{U_d(s)}{E(s)} = K_d s \quad (2.6)$$

kde $U_d(s)$ je výstup vyjadrený v Laplaceovej rovine.

Existuje taktiež ale aj iná forma derivačného člena, v ktorej sa počíta so zaraďením nízkofrekvenčného filtra určeného konštantou T_f . Použitie tejto formy derivačného člena je obzvlášť výhodné v prípade implementácie derivačného člena na zariadení, ktoré realizuje tento člen v diskrétnej forme (mikropočítač). Pri diskrétnej forme sa totiž ako požiadavka na výstup, tak aj výstup samotný mení skokovo, čo spôsobuje pri absencií filtra silné reakcie derivačného člena, ktoré následne vyvolávajú celkovú nestabilitu regulovaného systému.

Filtračný člen má tvar:

$$F_r(s) = \frac{1}{1 + T_f s} \quad (2.7)$$

pričom filtračnú konštantu T_f spravidla určujeme podľa vzťahu:

$$T_f = \frac{1}{2\pi f_c} \quad (2.8)$$

kde f_c predstavuje najvyššiu frekvenciu signálu, ktorú by ešte filter mal prepustiť. Okrem tohosa ale v praxi používa aj primitívnejšia metóda voľby T_f , kedy sa za T_f jednoducho dosadí celočíselný násobok konštanty K_d .

Konečná forma PID sa teda po spočítaní prenosov všetkých členov teda dá zapísť ako:

$$F_{PID}(s) = K_p + K_i \frac{1}{s} + K_d s \frac{1}{1 + T_f} \quad (2.9)$$

Na samotné naladenie PID regulátora je možné použiť viacero metód, medzi inými napríklad:

- **Ad-hoc metóda** - veľmi primitívou, ale pre jednoduché problémy sústavy dostačujúcou metódou je zvolenie K_p , K_i , K_d parametrov náhodným dosadzovaním a pozorovaním zmien reakcií riadenej sústavy
- **Ziegler-Nicholsová metóda** - pozostáva z vyradenia I, D zložiek a nájdenia takej hodnoty K_p , pri ktorej systém dosiahne stav na hranici stability, t.j. stavu, v ktorom sústava osciluje okolo stabilného stavu. V tomto stave odmeriame periódu oscilácií. Následným odčítaním hodnôt K_p , K_i , K_d z tabuľky A.1, prevzatej zo skripti [SKRIPTA] získame naladený PID.
- **Vytvorením modelu sústavy** - po nájdení matematického modelu regulovanej sústavy je možné vytvoriť sústavu diferenciálnych rovníc, ktoré popísu správanie celej sústavy

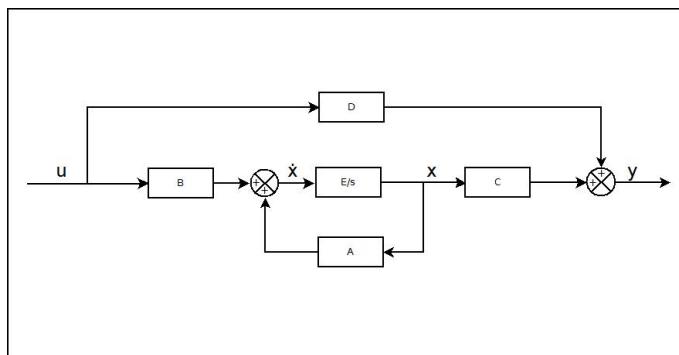
so zaradeným regulátorom. Tento matematický model je možné následne podrobieť analýze pomocou kritérií stability napr. Routh- Schurovým alebo Hurwitzovým a takto nájsť K_p , K_i , K_d , pre ktoré bude sústava stabilná. Nevýhodou tohto postupu je ale zvýšená náročnosť procesu ladenia, ale aj nutnosť dostatočne presne poznáť štruktúru sústavy na vytvorenie jej modelu.

2.1.2 LQR

Pred opisom samotného princípu fungovania LQR, považujeme za potrebné uviest' čitateľa do problematiky reprezentácie systémov v tzv. stavovom priestore, nakoľko regulátor LQR pracuje práve s touto reprezentáciou systému.

Pri reprezentácii časovo invariantnej sústavy, teda sústavy, ktorej vlastnosti sa časom nemenia, v stavovom priestore pracujeme vo všeobecnosti z výrazmi tvaru:

$$\dot{x}(t) = Ax(t) + Bu(t) \quad y(t) = Cx(t) + Dy(t) \quad (2.10)$$



Obr. 2.3: Stavový priestor - diagram

Pri takomto značení:

A = stavová matica systému [n x n]

B = vstupná matica [n x r]

C = výstupná matica [m x n]

D = matica opisujúca priamy prenos na výstup systému [m x r]

$x(t)$ = stavový vektor [n x 1]

$y(t)$ = výstupný vektor [m x 1]

$u(t)$ = vstupný vektor [$r \times 1$]

$\dot{x}(t)$ = prvá derivácia stavového vektora [$n \times 1$] pričom platí, že $m, n, r \subset N$.

Výhody takejto reprezentácie sústav sa prejavia hlavne pri MIMO sústavách (sústavách ktoré majú viacero vstupov a výstupov). S využitím maticovej a vektorovej reprezentácie je možné aj veľmi komplexné systémy vyjadriť len pomocou týchto dvoch rovníc. Maticová reprezentácia je naviac ešte aj veľmi vhodná pre spracovanie s využitím softvérových nástrojov ako napr. Matlab.

Proces samotného vyjadrenia stavovej reprezentácie zo systému diferenciálnych rovníc je mimo rozsah tejto práce, čitateľ sa však s ním môže oboznámiť napr. v [ContSys].

Po nájdení reprezentácie v stavovom priestore a overení, že daný systém je kontrolovanateľný, môžeme z matice A vyjadriť póly stavovej matice sústavy. To je možné napr. výpočtom z rovnice:

$$| pE - A | = 0 \quad (2.11)$$

Kde E je jednotková matica a p vektor pólov matice A .

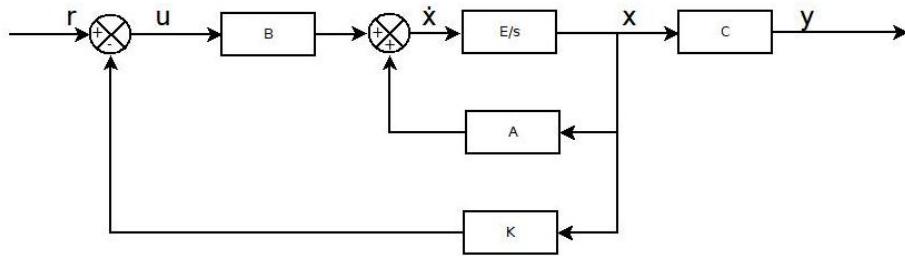
Pripomenieme, že pre stabilné systémy platí, že všetky reálne časti ich pólov sú záporné. Tento poznatok je možné účinne využiť v prípade nestabilných systémov a zaradiť do sústavy vhodne zvolený regulačný vektor K , ktorý nám zavedením spätej väzby do sústavy umožní zmeniť pôvodné póly na ľubovoľné, nami zvolené, stabilné ($Re p_i < 0$) póly.

Reprezentácia takejto sústavy v stavovom priestore bude mať následne tvar:

$$u(t) = -Kx \quad (2.12)$$

$$\dot{x}(t) = Ax(t) + B(-Kx(t)) \quad (2.13)$$

Čitateľ si môže všimnúť, že na Obr. 2.4 už nevystupuje v schéme matica D , ktorú v danom prípade uvažujeme ako nulovú – veličina na vstupe nie je teda privádzaná priamo na výstup. Členom r v schéme na Obr. 2.4 rozumieme referenciu, ktorej zmenou vieme upraviť požiadavku na výstup systému.



Obr. 2.4: Stavový priestor - spätná väzba

Vyššie sme spomínali, že vektor K je možné zvoliť tak aby boli dosiahnuté požadované póly, problémom ale je určiť aké póly sú pre danú sústavu ideálne. Pri zvolení príliš negatívnych pólov bude regulátor natoľko agresívny, že ho nebude možné v praxi realizovať a naopak, pokiaľ nebudú póly dostatočne záporné, bude regulácia trvať pridlho na to aby bola praktická. Práve tu prichádza ako riešenie do úvahy regulátor LQR, ktorý je využitím kvadratickej „cenovej“ funkcie schopný prideliť každej sledovanej veličine na výstupe „váhu“. Táto váha určí nakoľko budú pre regulátor dôležité zmeny jednotlivých veličín. LQR následne zvolí K také, aby regulátor pracoval podľa požiadaviek.

Ako vyplýva z názvu regulátora, pre svoju činnosť využíva princíp minimalizácie cenovej funkcie J , ktorá je pre spojity a konečný časový úsek definovaná ako:

$$J = \frac{1}{2}x^T(T)P_1x(T) + \frac{1}{2} \int_0^T (x^T Q x + u^T R u) dt \quad (2.14)$$

kde $Q \geq 0$; $R > 0$; $P_1 \geq 0$ sú symetrické, kladné matice. Q, R predstavujú váhy, ktoré sú priradené sledovaným veličinám. Riešením rovnice Obr. 2.14 je $P(t)$, nájdené vyriešením tzv. aritmetickej Riccatiho rovnice :

$$-\dot{P} = PA + A^T P - PBR^{-1}B^T P + Q; P(T) = P_1 \quad (2.15)$$

Následkom rovnice Obr. 2.15 je po spätnom chode a vyriešení pre K možné vyjadriť $u(t)$ z rovnice Obr. 2.12 ako:

$$u(t) = -R^{-1}B^T P(t)x \quad (2.16)$$

Kedže riešenie týchto rovníc je do značnej miery komplikované a jednoznačne mimo

rozsah tejto práce uvedieme len, že pri použití nastroja Matlab je možné nájsť K veľmi jednoducho a to použitím funkcie: $K = \text{lqr}(A, B, Q, R)$.

Pred implementáciou LQR je ale ešte potrebné adresovať otázku voľby R , Q matíc. Predpokladajme matematický model balansujúceho robota v stavovom priestore vyjadrený dosadením do rovnice Obr. 2.13 ako:

$$\dot{\mathbf{x}} = (A - BK) \begin{bmatrix} d \\ \dot{d} \\ \theta \\ \dot{\theta} \end{bmatrix} \quad (2.17)$$

kde:

d = poloha základne robota

\dot{d} = rýchlosť základne robota

θ = uhol natočenia šasi robota

$\dot{\theta}$ = uhlová rýchlosť robota

Ďalej uvažujeme Q také, že:

$$Q = \begin{bmatrix} w_1 & 0 & 0 & 0 \\ 0 & w_2 & 0 & 0 \\ 0 & 0 & w_3 & 0 \\ 0 & 0 & 0 & w_4 \end{bmatrix} \quad (2.18)$$

Predpokladajme, že maximálne dovolené odchýlky od požadovanej hodnoty sú:

- $0,005 \text{ m}$ pre polohu základne $\rightarrow w_1 = (0,005)^{-2}$
- $0,0005 \text{ m.s}^{-1}$ pre rýchlosť základne $\rightarrow w_2 = (0,0005)^{-2}$
- $0,02 \text{ rad}$ pre uhlovú odchýlku šasi $\rightarrow w_3 = (0,02)^{-2}$
- $0,0001 \text{ rad.s}^{-1}$ pre uhlovú rýchlosť šasi $\rightarrow w_4 = (0,0001)^{-2}$

Po určení Q by sme podobným spôsobom postupovali aj pre R až kým by sme našli hodnoty, ktoré pre sústavu fungujú uspokojivo.

2.2 Zhrnutie a výber regulátora

Medzi značné výhody PID regulátora patrí jeho jednoduchá implementácia a nízka náročnosť na výpočtový výkon realizačného hardvéru. Ladenie je možné ako exaktnými, matematickými metódami tak aj metódami založenými na empiricky zozbieraných dátach. Nevýhodou PID je ale, že jeho využitie je obmedzené na SISO systémy, t.j. systémy s jednou vstupnou a jednou výstupnou veličinou – čo napríklad v našom prípade nezarúčí stabilizáciu polohy aj riadenie robota súčasne. Tento nedostatok je ale možné prekonať použitím viacerých vhodne kombinovaných PID regulátorov, čo však zvyšuje náročnosť ich správneho naladenia.

Hlavnou výhodou využitia LQR na reguláciu sústavy je, možnosť riadiť jediným regulátorom všetky nami sledované výstupné veličiny, čím je možné dosiahnuť vcelku komplexné riadenie robota. Pri návrhu regulátora je taktiež možné zvoliť, ktoré sledované veličiny sú pre nás najdôležitejšie. Cenou za takúto presnosť riadenia je ale výrazne vyššia náročnosť výpočtov pri ladení regulátora ako aj nutnosť pracovať s matematickým modelom sústavy. Komplikovanejšie výpočty taktiež spôsobia, že v porovnaní s PID bude dĺžka trvania riadiacej slučky pri použití totožného hárveru potenciálne značne dlhšia čo môže viesť k zhoršeniu vlastností robota.

Po zohľadnení vlastností oboch regulátorov sme sa rozhodli pre použitie PID regulátora. Napriek niektorým jeho nevýhodným vlastnostiam bola jednoduchosť jeho implementácie rozhodujúcim faktorom pri našej voľbe. Viacero prác už demonštrovalo uspokojivé výsledky pri použití PID na riadenie balansujúceho robota a teda vieme, že ide o regulátor dostačujúci na naše účely a na rozdiel od LQR má autor s jeho používaním predošlé skúsenosti.

Kvôli neschopnosti PID riadiť viacero výstupov zároveň budeme pri našej realizácii pracovať s viacerými PID regulátormi zaradenými do kaskády. V takomto zapojení sa zvyčajne používa rozdielna dĺžka riadiacich slučiek - teda rozdielne dt pre jednotlivé PID. Viac sa danou problematikou budeme zaoberať priamo v kapitole venovanej našej implementácii regulátora.

3 | ANALÝZA DYNAMIKY BALANSUJÚCEHO ROBOTA

Dvojkolesový robot predstavuje z mechanického uhla pohľadu sústavu podobnú prevrátenému kyvadlu – teda sústavu, ktorá je inherentne nestabilná a vyžaduje teda implementáciu robustného regulátora. Aby bol tento návrh možný potrebujeme vytvoriť matematický model tohto systému, ktorý nám umožní navrhúť optimálny regulátor. Po zvážení sme usúdili, že bude výhodné takýto model vytvoriť ako pre časť podvozku s kolesami tak aj pre šasi robota, ktoré tvorí prevrátené kyvadlo a tieto následne skombinovať. Pracovať budeme s modelom navrhnutým Rich Chi Ooi [Oochi2003] a mierne upraveným v práci [TWIP].

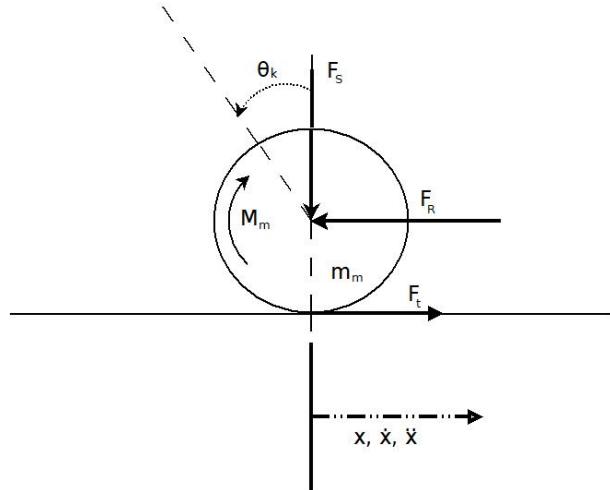
3.1 Model kolesa

Matematický model správania sa kolesa nebudeme odvádzáť pre každé koleso zvlášť, ale budeme predpokladať, že obe kolesá sú identické a teda aj rovnice budú pre obe rovnaké.

Obr. 3.1 znázorňuje koleso robota aj so všetkými silami, ktoré naň pôsobia, pričom:

- | | |
|--|------------------------------|
| • θ = uhol náklonu šasi | • J_k = zotrvačnosť kolesa |
| • m_k = hmotnosť kolesa | • r = polomer kolesa |
| • F_s = sila, ktorou na kolesa pôsobí šasi | • M_m = točivý moment |
| • F_R = reakčná sila medzi kolesom a šasi | • x = poloha v x-ovej osi |
| • F_t = trecia sila medzi zemou a kolesom | |

V rovniciach budeme ďalej predpokladať, že robot sa nepohybuje do strán ale len vpred alebo vzad. Do úvahy budeme musieť ale vziať to, že robot bude pri pohybe ovplyvňovaný ako vonkajšími stimulmi tak aj samotným momentom motora. Na začiatok využitím Newtonovho



Obr. 3.1: Model kolesa

zákona pohybu v X-ovej osi odvodíme:

$$\sum F_x = ma \quad (3.1)$$

$$m_k \ddot{x} = F_t - F_R$$

A potom vyjadríme súčet momentov okolo stredu kolesa:

$$\sum M = J\alpha \quad (3.2)$$

$$J_k \ddot{\theta} = M - F_f r$$

Ak teda vyjadríme točivý moment DC motora ako rozdiel zotrvačnosti motora vynásobeného okamžitým uhlovým zrýchlením a momentu záťaže môžeme pokračovať v odvodzovaní:

$$M_m = J \frac{d\omega}{dt} - M_Z \quad (3.3)$$

$$M = M_m - M_Z = \frac{-K_M K_e}{R} \dot{\theta} + \frac{K_M}{R} V_m$$

kde: K_Z = moment záťaže

K_M = momentová konštantá

K_e = elektrická konštantá motora

R = odpor vinutia motora

U_m = el. napätie na motore

Konštanty K_M , K_e je možné získať zo vzťahov (3.4):

$$K_e = \frac{V_X}{\omega} \quad K_M = \frac{F_X r}{I_X} \quad (3.4)$$

Dosadením do (3.2) z (3.3) tak získame:

$$F_f = \frac{-K_M K_e}{Rr} \dot{\theta}_k + \frac{K_M}{Rr} U_m - \frac{K_k}{r} \ddot{\theta}_k \quad (3.5)$$

Túto rovnicu je možné prepísať pomocou (3.1) a nájsť tak rovnicu pre jedno koleso:

$$\begin{aligned} m_m \ddot{x} &= F_f - F_R = \frac{-K_m K_e}{Rr} \dot{\theta}_k + \frac{K_M}{Rr} U_m - \frac{J_k}{r} \ddot{\theta}_k - F_R \\ &\Downarrow \\ \theta &= \frac{x}{r} \quad (3.6) \\ &\Downarrow \\ m_m \ddot{x} &= F_f - F_R = \frac{-K_M K_e}{Rr^2} \dot{x}_k + \frac{K_M}{Rr} U_m - \frac{J_k}{r^2} \ddot{x}_k - F_R \end{aligned}$$

Vynásobením dvoma (dve kolesá) získavame kompletný model podvozku:

$$\begin{aligned} 2m_m \ddot{x} &= 2F_f - 2F_R = \frac{-2K_M K_e}{Rr^2} \dot{x}_k + \frac{2K_M}{Rr} U_m - \frac{2J_k}{r^2} \ddot{x}_k - 2F_R \\ 2(m_m - \frac{2J_k}{r^2}) \ddot{x} &= \frac{-2K_M K_e}{Rr^2} \dot{x}_k + \frac{2K_M}{Rr} U_m - 2F_R \end{aligned} \quad (3.7)$$

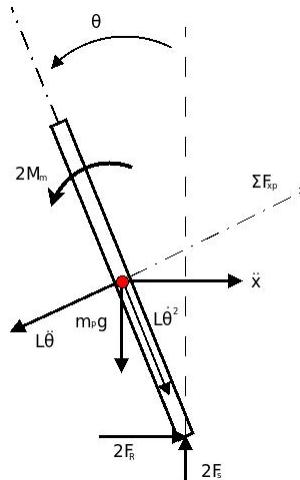
3.2 Model šasi

Na Obr. 3.2 sú znázornené sily pôsobiace na šasi robota pri pohybe, pričom:

L = dĺžka šasi, meraná od stredu kolies

m_p = hmotnosť šasi

θ = uhol náklonu šasi



Obr. 3.2: Model šasi

$$\sum F_{xp} = \text{súčet síl na virtuálnej osi } xp$$

Význam ostatných veličín je zhodný s definíciami v predchádzajúcej časti. Je teda zjavné, že podľa očakávaní sú sily pôsobiace na podvozok premietnuté aj do modelu šasi. Z Newtonovho zákona po úprave dostaneme rovnicu v tvare:

$$2F_R = m_p \ddot{x} + m_p L \ddot{\theta} \cos \theta - m_p L \dot{\theta}^2 \sin \theta \quad (3.8)$$

Táto rovnica predstavuje súčet síl na horizontálnej osi. Pre sily pôsobiace ne virtuálnej osi $F_p x$, kolmo na šasi, platí vzťah (3.9) a súčet momentov síl okolo ťažiska šasi je vyjadrený v (3.10).

$$2F_R \cos \theta + 2F_S \sin \theta - m_p g \sin \theta - m_p L \ddot{\theta} = m_p \ddot{x} \cos \theta \quad (3.9)$$

$$- 2F_R L \cos \theta - 2F_S L \sin \theta - 2M_m = J_p \ddot{\theta} \quad (3.10)$$

Točivý moment motorov je potrebné linearizovať a teda:

$$2M_m = \frac{-2K_M K_e \dot{x}}{Rr} + \frac{2K_M U_m}{R} \quad (3.11)$$

Po dosadení (3.11) do (3.10) dostaneme (3.12). Následne upravíme (3.9) vynásobením $-L$ a prepísaním do tvaru (3.13):

$$- 2F_R L \cos \theta - 2F_S L \sin \theta = \frac{-2K_M K_e \dot{x}}{Rr} + \frac{2K_M U_m}{R} + J_p \ddot{\theta} \quad (3.12)$$

$$- 2F_R L \cos \theta - 2F_S L \sin \theta = -m_p g L \sin \theta - m_p L^2 \ddot{\theta} - m_p L \ddot{x} \cos \theta \quad (3.13)$$

Odčítaním týchto dvoch rovníc dostávame:

$$\frac{-2K_M K_e \dot{x}}{Rr} + \frac{2K_M U_m}{R} + J_p \ddot{\theta} = -m_p g L \sin \theta - m_p L^2 \ddot{\theta} - m_p L \ddot{x} \cos \theta \quad (3.14)$$

Z rovnice (3.15) môžeme odstrániť člen $2F_R$ nahradením z (3.8):

$$2(m_m - \frac{2J_k}{r^2}) \ddot{x} = \frac{-2K_M K_e}{Rr^2} \dot{x}_k + \frac{2K_M}{Rr} U_m - m_p \ddot{x} - m_p L \ddot{\theta} \cos \theta - m_p L \dot{\theta}^2 \sin \theta \quad (3.15)$$

Po vyjadrení (3.14) a (3.15) je potrebné rovnice linearizovať. Pri linearizácii predpokladáme, že uhol $\theta = 0 + \varphi$, kde φ predstavuje malý uhol náklonu robota. Výsledkom linearizácie

a následnej úpravy rovníc je výsledný model balansujúceho robota:

$$\ddot{\varphi} = \frac{m_p L}{J_P L^2} \ddot{x} + \frac{2K_M K_e}{Rr(J_P + m_p L^2)} \dot{x} - \frac{2K_M}{R(J_P + m_p L^2)} U_m + \frac{m_p g L}{(J_P + m_p L^2)} \varphi \quad (3.16)$$

$$\ddot{x} = \frac{2K_M}{Rr(2m_k - \frac{2J_k}{r^2} + m_p)} U_m - \frac{2K_M K_e}{Rr^2(2m_k - \frac{2J_k}{r^2} + m_p)} \dot{x} + \frac{m_p L}{2m_k - \frac{2J_k}{r^2} + m_p} \ddot{\varphi}$$

4 | NÁVRH A REALIZÁCIA KONŠTRUKCIE

DVOJKOLESOVÉHO ROBOTA

Pred začatím procesu návrhu hardvérového riešenia nášho robota sme postupovali najprv vytvorením zoznamu komponentov, potrebných pre realizáciu robota. Pri návrhu sme brali do úvahy požiadavku na modulárnosť robota. Pod pojmom modulárnosť rozumieme skonštrukovanie robota tak, aby jednotlivé jeho časti mohli byť v prípade potreby jednoducho vymené alebo upravené bez potreby väčších zásahov do celkovej konštrukcie.

Robota sme teda rozdelili do viacerých funkčných celkov a pre každý z týchto celkov vytvorili samostatný zoznam komponentov. Vďaka tomuto postupu sme nielen znížili možnosť výberu nevyhovujúcich súčiastok, ale aj zabezpečili, že naše riešenie bude v prípade potreby škálovateľné a jednotlivé celky bude ľahké pozmeniť alebo doplniť o dodatočné časti.

Funkčné celky balansujúceho robota:

- napájanie
- pohon
- komunikácia
- senzory
- riadiaci mikropočítač

4.1 Zoznam použitých komponentov

V nasledujúcej časti práce opíšeme nami vybrané komponenty, ktoré boli využité pri konštrukcii robota. Zameriame sa hlavne na dôvod voľby daného komponentu, porovnáme špecifikácie použitých komponentov s našimi požiadavkami a zhodnotíme, ako dobre sa komponent hodí

pre naše použitie.

4.1.1 Napájanie

Táto podkapitola obsahuje komponenty, ktoré poskytujú vhodné napájanie ostatným časťam robota.

Batéria

Pri výbere batérie sme sa snažili nájsť nabíjateľný model, ktorý by nám poskytol pri nízkej hmotnosti čo najvyššiu kapacitu, relatívne vysoký maximálny výstupný prúd a výstupné napätie pohybujúce sa v rozmedzí 6 V až 12 V, ktoré je dostatočné pre väčšinu bežne dostupných jednosmerných motorov. Rozhodli sme sa pre 12V lítium-iónovú (Li-ion) batériu, využívajúcu tri 3,7 V články typu 18650. Kapacita batérie je 3300 mAh, maximálny okamžitý odoberaný prúd 5 A a maximálny pracovný prúd 3A. K nej priložený nabíjací adaptér je schopný dobíjať je prúdom 1 A pri 12,6 V. Spolu s relatívne nízkou hmotnosťou 150g sa teda batéria z Obr. 4.1 po každej stránke javí ako dobrá voľba.



Obr. 4.1: Li-ion batéria

| Výs. napätie [V] | Max. výs. prúd [A] | Pracovný prúd [A] | Hmotnosť [kg] | Technológia |
|------------------|--------------------|-------------------|---------------|-------------|
| 12 | 5 | 3 | 0.15 | Li-ion |

Tab. 4.1: Parametre batérie

Napäťový regulátor pre logické obvody

Jedným z možných riešení pre napájanie logických odvodov robota, bolo použitie sekundárnej 5 V batérie. Toto riešenie sa ale nejavilo v tomto prípade ako optimálne, keďže druhá batéria by zaberala miesto a vyžadovala si samostatný napájací adaptér. Rozhodli sme sa preto využiť už zabudovanú 12 V batériu napájajúcu motory v kombinácii s paralelne pripojeným napäťovým regulátorom.

Rozhodovali sme sa medzi rozšíreným integrovaným obvodom L7805CV, ktorý predstavuje 5 V lineárny napäťový regulátor a spínaným napäťovým regulátorom LM2596. Vďaka vlastnostiam LM2596 medzi ktoré patrí vysoká účinnosť (až 92%), malé kolísanie výstupného napätia a možnosť riadenia výstupného napätia od 4 V – 35 V sme sa nakoniec rozhodli použiť ako napäťový regulátor Obr. 4.2 práve tento obvod.



Obr. 4.2: Spínaný napäťový regulátor kompatibilný s LM2596

4.1.2 Pohon

Pre samotný pohyb robota je potrebné správne zvoliť vhodný typ motora a obvodu, ktorý bude môcť podľa pokynov mikropočítača dané motory ovládať. V našom prípade sme vyberali ako motory, ktoré budú poháňať kolesá robota, tak aj servomotor umožňujúci robotovi pohybovať sa aj po naklonených plošinách.

Motory

V prípade motorov sme mali na výber z viacerých možností, pričom hlavným kritériom bolo, že sa musí jednať o jednosmerné motory, primeranej veľkosti. Aj tak sme ale mohli voliť medzi bezkomutátorovým, krokovým a motorom s permanentnými magnetmi. Po úvahе

a prieskume bežne dostupných motorov sme sa rozhodli pre klasický motor s permanentnými magnetmi využívajúci komutátor. Tento motor má niekoľko nevýhod ako sú napríklad malá presnosť v porovnaní s krokovým motorom a menší moment spolu s rýchlejším opotrebením v porovnaní s motorom bez komutátora. Napriek týmto nevýhodám sú tieto motory ale vhodné pre naše účely lebo sú lacné, jednoduché na ovládanie a dostupné v mnohých konfiguráciách.

Po zvážení sme sa rozhodli pre 12 V motory typu GM25-370CA, s integrovanou prevedovkou 1:21 a zabudovanými dvojkanálovými enkodérmi, ktoré nám umožnia odometrickým meraním sledovať zmenu pozície kolies napojených na motor. Dokumentácia uvádzia max. rýchlosť nezaťaženého motora ako 280 RPM (otáčok za minútu), pričom túto max. rýchlosť potvrdili aj naše merania. Výhodou týchto motorov je aj, že je možné ich dostať v konfigurácií priamo určenej pre robotické platformy podobné našej spolu s kolesami a kovovým podvozkom.

Doplniť tabuľku



Obr. 4.3: Platforma s motormi

H-mostík

Kedže mikropočítač nie je bez externej elektroniky schopný sám dodávať do motora potrebný výkon je potrebné spolu s ním použiť tzv. H-mostík. Ten predstavuje principiálne iba štyri elektricky ovládané spínače Obr. C.1, ktoré podľa svojej konfigurácie menia smer toku prúdu motorom - teda smer otáčania motora. Naše požiadavky na H-mostík boli: vysoká účinnosť, jednoduché prepojenie s mikropočítačom, galvanicky oddelené vstupy mikropočítača



Obr. 4.4: H-mostík komaptibilný s L298

a schopnosť dodať motorom dostatočný výkon (výrobca uvádza maximálny odber 1,5 A na motor).

Max vstu pdo h mostika

Našou voľbou bol dvojitý H-mostík kompatibilný s integrovaným obvodom L298, ktorý splňa všetky naše požiadavky a je schopný dlhodobo dodávať do motorov až 7 A pri napäti 12 V. Tento obvod je možné prepojiť s mikropočítačom pomocou šiestich vstupov, pričom štyri slúžia na výber konfigurácie spínačov a dva prijímajú PWM signál, ovládajúci pripojenie batérie na vstup motorov. Je teda možné jednoducho meniť striedu jednotlivých motorov a tým aj ich rýchlosť. Pre svoju správnu činnosť vyžaduje tento H-mostík napájanie 5 V, to nám poskytne nami použitý napäťový regulátor. Nevýhodou, ale je relaívne veľká vôľa kolies a nízke rozlíšenie enkodérov.

Servomotor

Jednou z dodatočným požiadaviek na nášho balansujúceho robota bolo, aby sa šasi robota mohlo pohybovať do strán nezávisle od platformy s kolesami. Táto funkcia nám v praxi poskytne vyššiu kontrolu nad robotom v zatáčkach, na naklonených plošinách a do určitej miery zníži pravdepodobnosť pádu v prípade pôsobenia silou na bočnú časť robota.

Za optimálne riešenie sme považovali jednoduché servo HJ S3315D určené prevažne na modelárske účely. Toto servo dodatočne taktiež spojí platformu s kolesami a šasi robota, ktorým bude takto možné v prípade potreby hýbať o presne určené uhly. V takejto konfigurácii

obmedzenie pohybu serva v rozmedzí -90° až 90° nepredstavuje problém.



Obr. 4.5: Servo motor HJ S3315D

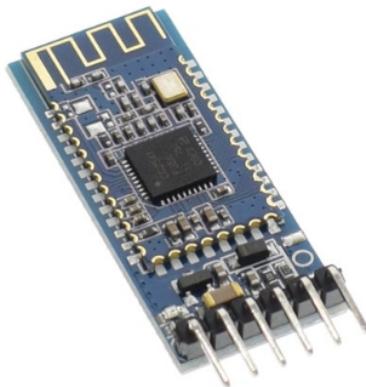
Overit spravne hodnoty HM-05

4.1.3 Komunikácia - Bluetooth modul

Okrem samotného balansovania na mieste musí byť tiež operátor schopný robota na diaľku riadiť tak aby bol tento schopný pohybovať sa v priestore podľa pokynov. Taktiež je nutné aby bol robot v prípade požiadania operátora schopný poskytnúť základné informácie o svojom stave, napr. úroveň nabitia batérie, priemernú rýchlosť pohybu za určitý časový interval, okamžitý uhol náklonu, . . . Tieto informácie musí byť robot schopný poskytnúť s čo najmenším oneskorením, bezdrôtovo a minimálne na vzdialenosť 20 metrov.

Rozhodli sme sa použiť Bluetooth vysielač/prijímač HM-05. Tento modul sa vyznačuje nízkou spotrebou max. $235 \mu\text{A}$, min. $0.4\mu\text{A}$ a je schopný ako prijímať tak aj odosielat dátá pomocou rozhrania štandardného rozhrania UART (Universal Asynchronous Reciever-Transmitter).

Využitím tohto modulu ako v robotovi tak aj v ovládači, ktorým ho budeme ovládať zabezpečíme ako možnosť odosielat jednoduché povely na riadenie robota, tak aj prijímať komplexné dátá o jeho aktuálnom stave.



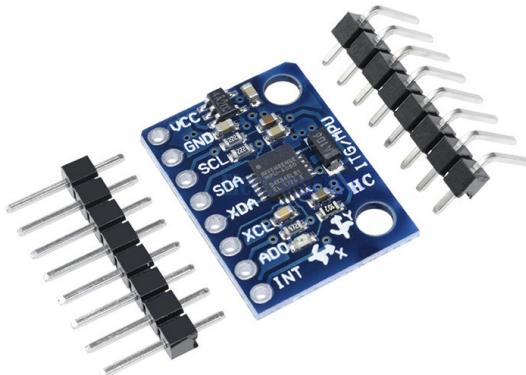
Obr. 4.6: HC05-Bluetooth modul

4.1.4 Senzory

Pod senzormi rozumieme všetky časti robota, ktoré mu umožňujú získavať informácie o jeho okolitom prostredí. Pre naše účely je nevyhnutné presne merať uhol náklonu robota a rýchlosť pohybu motorov.

MPU 6050

Pre úspešné balansovanie robota je potrebná relatívne vysoká presnosť merania uhla náklonu robota. Ak by sme postupovali využitím akcelerometra, prístroja určujúceho uhol náklonu podľa pôsobenia gravitačného zrýchlenia, nebolo by toto meranie dosť presné, keďže pri rýchlych zmenách polohy by nebol akcelerometer schopný poskytnúť presné merania. Výhodou merania z akcelerometra je však veľmi vysoká presnosť merania v ustálenom stave. Naproti tomu gyroskop, prístroj merajúci uhlovú rýchlosť, by po integrovaní nameraných hodnôt bol schopný určiť zmenu uhla za časový okamih dt – teda aj novú polohu v prípade, že poznáme tu predchádzajúcu. Problémom použitia gyroskopu ale je, že aj tá najmenšia chyba pri každom meraní a integrovaní sa započítá do výsledku a po niekoľkých stovkách meraní je už chyba merania uhla značná – tomuto javu sa hovorí gyroskopický drift. Riešenie problému merania uhla v pohybujúceho sa robota predstavuje kombinácia nameraných hodnôt z akcelerometra a gyroskopu, pomocou komplementárneho filtra. Na toto využite sa hodí modul MPU 6050, ktorý v sebe kombinuje trojosový akcelerometer a gyroskop. Modul je schopný komunikovať



Obr. 4.7: MPU6050

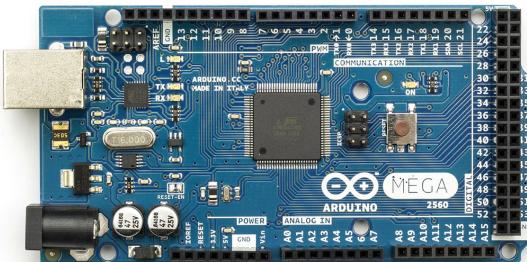
s mikropočítačom pomocou rozhrania I2C maximálnou rýchlosťou 400kHz a obsahuje taktiež integrovaný teplomer a DMP (Digital Motion Processor). Ten je schopný priamo uskutočniť analýzu nameraných dát, čím sa znížia požiadavky na výpočtový čas mikropočítača.

4.1.5 Riadiaci počítač - Arduino MEGA

Vďaka skúsenostiam s programovaním mikropočítačov spoločnosti Microchip-Atmel sme sa pri výbere mikropočítača rozhodovali medzi mikropočítačmi rodiny ATMEGA. Do úvahy spadali hlavne ATMEGA328P a ATMEGA2560, ktorých výpočtové možnosti a zabudované periférie postačovali našim požiadavkám. Jednou zo zvažovaných možností bolo navrhnutie a skonštruovanie obvodu so zabudovaným mikropočítačom, ktorý by priamo zodpovedal požiadavkám našej aplikácie. Toto riešenie sme ale nakoniec po úvahe zavrhlí, keďže by bolo časovo náročné a nespadalo by do koncepcie modularity – takto navrhnutý obvod by v prípade poruchy bol náročný na nahradenie a bolo by veľmi náročné upraviť ho pri zmene požiadaviek na robota.

Rozhodli sme sa teda využiť rozšírenú, otvorenú, vývojovú platformu Arduino, ktorá vo všeobecnosti slúži na vývoj prototypov a testovanie kódu. Našou voľbou bolo Arduino MEGA Obr. 4.8, využívajúce mikropočítač ATMEGA2560. Táto vývojová doska má zabudovaný programovací port USB typu B (ktorý sme zamenili na micro USB), napäťový regulátor na 5V a 3.3V, 16 MHz oscilátor (zdroj hodinového signálu pre mikropočítač) a na rozdiel od lacnejšieho Arduino Uno má až 53 GPIO (General Purpose Input Output) pinov, 16 pinov

podporujúcich ADC (prevod analógového signálu na jeho digitálnu reprezentáciu) a 6 pinov podporujúcich niekoľko režimov prerusení.



Obr. 4.8: Arduino MEGA

Samotný mikropočítač má viacero časovačov, ktoré umožňujú napr. merať časové intervaly a generovať PWM signál, ale taktiež 4 páry RX, TX pinov slúžiacich na komunikáciu pomocou UART protokolu. Ako ďalšie možnosti komunikácie je možné využiť aj I2C a SPI protokoly. Programovanie dosky Arduino MEGA je možné cez USB rozhranie napr. pomocou programu ArduinoIDE, AtmelStudio alebo Eclipse. Našou voľbou bolo využitie všeobecného editora Eclipse, v kombinácii s pluginom umožňujúcim jednoduchú prácu s mnohými produktami Microchip-Atmel.

4.2 Návrh a výroba šasi robota

Pri výrobe šasi robota sme sa rozhodli použiť technológiu 3D tlače. S jej použitím sme boli schopní vytvoriť pevné a ľahké šasi, ktoré presne zodpovedalo našim požiadavkám. Pred samotnou tlačou sme ale museli navrhnúť model v CAD programe, tak, aby bolo nielen jednoduché ho vytlačiť, ale aj aby bolo schopné odolať nárazom a ochrániť tak elektroniku vnútri. Pri návrhu sme použili software Fusion 360, ktorý je pre študentov dostupný zdarma na stránke firmy Autodesk () .

Program Fusion 360 podporuje ako návrh a export modelov do viacerých bežne používaných formátov, tak aj analýzu vlastností modelu, simuláciu jeho správania pri záťaži, nástroje na vizualizáciu, animáciu a mnoho ďalších užitočných funkcií.

Nami vytvorené šasi sa skladá z dvoch častí a to veka a miskovitého tela, v ktorom sú



Obr. 4.9: Balansujúci robot s osadenými motormi

uložené komponenty. Po naštudovaní prác, ktoré už boli na tému návrhu dvojkolesového balansujúceho robota napísané sme zvolili pre celé šasi klinovitý tvar, s úzkou podstavou a širokých vrchom. Tento tvar nám umožní jednoduché napojenie servomotora na spodnú časť robota a osadenie batérie blízko k hornej časti robota. V takejto pozícii batérie dosiahneme relatívne vysoko umiestnené ťažisko, čo následne zjednoduší stabilizovanie robota. Šasi taktiež obsahuje otvory na kabeláž od motorov, dve LED diódy indikujúce stav robota, spínač napájania a prístup k portu micro USB, vďaka ktorému je možné robota programovať bez nutnosti demontáže veka.

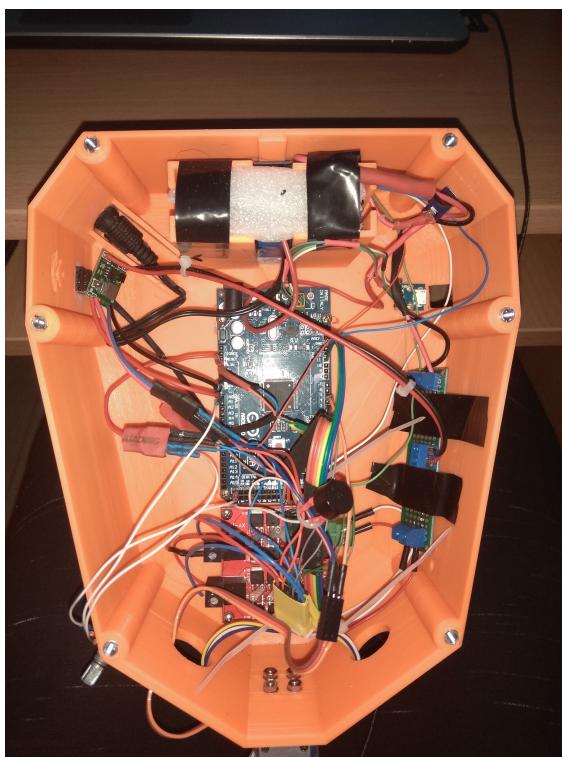
Tlač prebehla po vytvorení .gcode súboru v nástroji Cura na tlačiarni Creality CR-10s. Na tlač bol použitý materiál PLA (Polyactic Acid), ktorý sa vyznačuje nízkou tepelnou rozložnosťou pri chladnutí, nízkou cenou a jednoduchou tlačou.

Samotná tlač trvala približne 50 hodín, kvôli požiadavke na relatívne vysokú presnosť (a teda malú výšku jednotlivých vrstiev) a spotrebovalo sa pri nej približne 0,5 kg filamentu PLA. Po vytlačení, osadení komponentov a nalakovaní sa nami vytvorený model Obr. 4.9

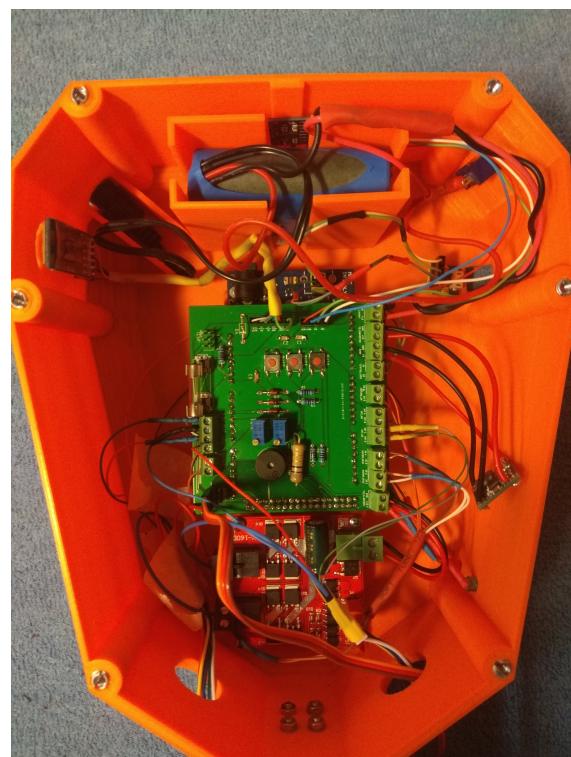
výzorom blížil k jeho počítačovo vygenerovanej podobe Obr. ???. Po kontrole môžeme tiež konštatovať, že vnútorné a vonkajšie rozmery oboch dielov presne zodpovedali nášmu návrhu.

Pri tlači sa vyskytli menšie chyby, ktoré spôsobili artefakty na veku robota v podobe pruhov spôsobených krycou páskou na podstave tlačiarne. Ako problematické sa ukázalo taktiež vytlačenie tela, na ktorého povrchu boli po vytlačení viditeľné vady v podobe nezaplnených miest a príliš zaoblených hrán. Tie mohli byť spôsobené nesprávnym nastavením parametrov v programe Cura, privysokou teplotou vyhrievanej podstavy tlačiarne alebo príliš priľnavou podstavou.

Výsledné výtlaky boli ale použiteľné, kedže vyššie opísané vady boli čisto estetického charakteru. Ako závažnejším problémom sa ukázala realizácia kabeláže. Napriek dostatku miesta bolo po zapojení všetkých komponentov náročné sa v kabeláži vyznať, čo by mohlo spôsobiť problémy pri dodatočnej údržbe. Taktiež, kedže niektoré súčasti zapojenia nebolo možné realizovať na doske Arduino, kedže by si vyžadovali použitie dodatočných komponentov, boli by sme na ich realizáciu nútení použiť dosku plošných spojov, čo by následne ešte zhoršilo prehľadnosť kabeláže.



Obr. 4.10: Pred osadením shieldu



Obr. 4.11: Po osadení shieldu

Ako problém z neprehľadnosťou kabeláže tak aj s ďalšími potrebnými komponentami sme sa rozhodli adresovať návrhom a konštrukciou tzv. shieldu - čiže dosky plošných spojov, ktorá svojim vyhotovením bude predstavovať nadstavbu Arduina. Tento shield sa pripojí na výstupy Arduina a bude obsahovať označené terminály na pripojenie všetkých komponentov.

Návrh tejto dosky plošných spojov sme uskutočnili v bezplatnej verzií nástroja Eagle. Tá umožňuje návrh elektrických schém aj PCB (Printed Circuit Board) a generovanie gerber súborov, ktoré obsahujú informácie pre výrobcu. Ukážka takto vytvorenej dosky je v prílohe , pričom pri jej tvorbe sme vychádzali zo schémy na Obr. B.1.

Zo schémy je zjavné, že nami vytvorený obvod obsahuje okrem vstupno/výstupných terminálov aj dodatočnú ochranu vstupov proti prepätiu. Táto je realizovaná formou zenerových diód s hodnotou prierazného napäcia 5 V, zapojených v závernom smere. Napriek tomu, že mikropočítač ATmega 2560 obsahuje zabudované diódy, ktoré vedia vstupy ochrániť proti krátkodobému prepätiu, externé zenerove diódy prejdú do vodivého stavu skôr a zabránia tak prípadnému poškodeniu mikropočítača prepätím.

Medzi ďalšie ochranné prvky, ktoré sme implementovali patrí, voči kladnému pólu batérie, sériovo zapojená schottkyho dióda, ktorá zabráni poškodeniu komponentov v prípade nesprávneho pripojenia batérie. Podobným spôsobom zapojená tavná poistka ochráni batériu pred možným skratom v obvode. Kedže bluetooth modul, ktorý sme zvolili pre realizáciu komunikácie, pracuje s logickými úrovňami od 0 V do 3,3 V, ale Arduino výstupy pracujú s 5 V pripojili sme RX vstup modulu k mikropočítaču cez napäťový delič.

Doska plošných spojov obsahuje aj dva potenciometre, tri spínače a reproduktor. Potenciometre a spínače budú v prípade potreby slúžiť na jemnú kalibráciu alebo manuálne zadávanie jednoduchých povelov. Reproduktor so zabudovaným oscilačným obvodom slúži ako hlásič, napr. v prípade prebiehajúcej kalibrácie. Kompletná schéma nami navrhnutého shieldu je v prílohe na Obr. B.1. Porovnanie vnútra robota pred a po osadení shieldu je na Obr. 4.10, Obr. 4.11.

4.3 Výsledky montáže a zhodnotenie návrhu

Nami navrhnutý dizajn robota sa v praxi ukázal ako veľmi jednoduchý na skonštruovanie. Po vytlačení šasi robota na 3D tlačiarni stačilo len spojiť skrutkami servo, šasi a podvozok robota a následne zrealizovať kabeláž podľa schémy Obr. B.1. Finálny vzhľad robota po zmontovaní je na Obr. 4.12 Obr. 4.13. Do vnútra šasi robota sme bez problémov nainštalovali všetky potrebné komponenty a ponechali dostatok voľného miesta na prípadné neskoršie úpravy robota.



Obr. 4.12: Robot spredu



Obr. 4.13: Robot z boku

Prípadne upraviť alebo doplniť

Pri motáži sme ale identifikovali niekoľko nedostatkov nášho návrhu. Jedným z nich bolo nevhodné umiestnenie senzora MPU6050 priamo nad batériou, čím sme značne limitovali možnosti prístupu ku senzoru po montáži. Samotný priestor pre osadenie batérie bol pôvodne navrhnutý bez veka, ktoré be zabránilo batérii v pohybe pri činnosti robota a navrhnutý vhodné veko sa ukázalo ako značne obtiažne. Výsledkom bolo, že batéria musela byť ukotvená použitím lepidla, čo nie je najvhodnejšie riešenie kedže značne komplikuje prípadnú možnosť

výmeny batérie za iný model.

4.4 Návrh a výroba ovládača robota

Pri návrhu ovládača robota sme postupovali obdobným spôsobom ako pri návrhu šasi. Rovnako ako pri šasi aj tu bol na návrh ovládača použitý nástroj Fusion 360 a technológia 3D tlače. Takto navrhnutý ovládač obsahuje dvojosový analógový joystick, ktorý riadi pohyb robota a umožňuje používateľovi pohyb v menu nastavení. Informácie získava používateľ zo zabudovaného 1,8 palcového farebného TFT displeja. Orientáciu v menu zjednodušujú dve zabudované tlačidlá.



Obr. 4.14: Ovládač balansujúceho robota

Napájanie ovládača je riešené z klasickej 9V batérie, zredukovanej spínaným zdrojom na 5V a na samotnú komunikáciu slúži modul HC-05 spomenutý v prvej časti tejto kapitoly. Okrem vysielania riadiacich príkazov ovládač v periodických intervaloch prijíma od robota dátá o jeho stave, ktoré následne zobrazuje užívateľovi na displeji.

Ovládač je schopný zobraziť informácie o uhle náklonu robota, aktuálnej rýchlosťi, prie-mernej rýchlosťi, aktuálnych kalibračných hodnotách a stave batérie. Používateľ môže prostredníctvom ovládača dať robotovi pokyn na začatie autokalibrácie, odosielania dát do terminála počítača cez sériový port alebo na vyslanie výstražného zvukového znamenia.

Na rozdiel od predchádzajúceho prípadu, kedy sme navrhli okrem šasi taktiež dosku plošných spojov, v prípade ovládača neexistujú požiadavky na nijakú dodatočnú elektroniku

a tak táto potreba odpadá. Komponenty budú spojené vnútri ovládača priamo s vývojovou doskou Arduino Nano. Malé rozmery tejto dosky zabezpečia, že rozmery ovládača budú podobné bežne predávaným konzolovým ovládačom.

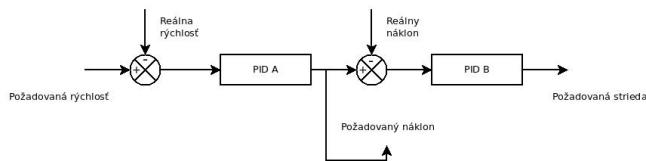
Ako problematická sa ukázala spotreba spínaného regulátora napäťa, ktorá v bola v prípade nezaľaženého regulátora až 8 mA. Takýto trvalý odber by nami použitú batériu rýchlo zničil, pristúpili sme teda k priradeniu spínača, ktorým bude batériu možné priamo odpojiť keď bude ovládač nepoužívaný.

5 | NÁVRH A IMPLEMENTÁCIA RIADIACEHO SYSTÉMU ROBOTA.

Základom nami použitého riadiaceho systému boli dva PID regulátory zapojené do kaskády. Toto zapojenie budeme ďalej označovať ako RS. Vstupom RS je požiadavka na rýchlosť pohybu robota v $rad.s^{-1}$ a výstupom strieda PWM signálu privádzaná na vstup do H-mostíka, vyjadrená hodnotou od -100% až 100% (kde záporná hodnota predstavuje zmenu smeru otáčania motorov). Funkcia jednotlivých PID regulátorov v RS bude vysvetlená v nasledujúcej časti práce. Je nutné podotknúť, že samotné zabezpečenie balansovania robota na mieste je možné realizovať použitím jediného PID regulátora. Nevýhodou takéhoto postupu je ale nemožnosť akejkoľvek regulácie rýchlosťi pohybu robota.

5.1 Štruktúra RS robota

Na Obr. 5.1 sme znázornili zapojenie PID regulátorov v RS.



Obr. 5.1: Schematické zapojenie PID regulátorov

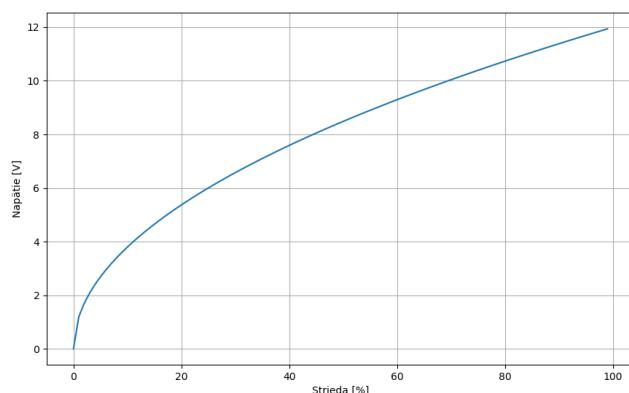
PID A: Vstupom do PID A je požadovaná rýchlosť robota, tá je následne odčítaná od reálnej rýchlosťi nameranej pomocou enkodérov a spracovaná PID regulátorom. Pri riadení robota môžeme špecifikovať požadovanú rýchlosť a na výstupe PID A bude požadovaný uhol náklonu. Týmto spôsobom zabezpečíme ako dosiahnutie požadovanej rýchlosťi, tak aj to,

že robot sa bude pohybovať konštantnou rýchlosťou. Výhodnou vlastnosťou PID A je, že požadovaný uhol náklonu nebude počas celej doby riadenia konštantný pre danú rýchlosť, ale bude dynamicky prispôsobovaný aktuálnym podmienkam. Ak by to tak nebolo robot by pri konštantnom uhle náklonu pokračoval v zrýchľovaní aby sa vyhol pádu až pokiaľ by motory dosiahli maximálnu rýchlosť a neboli schopné dosiahnuť požadované zrýchlenie na udržanie uhla náklonu. Po tomto bode by nasledoval pád.

PID B: Výstupom PID B je ako už bolo skôr spomenuté strieda signálu vstupujúceho do H-mostíka. V prípade nami použitého mostíka platí priama úmera medzi striedou vstupného signálu a napäťím do motorov. Nedá sa ale povedať, že by závislosť medzi striedou a výstupným napäťím bola lineárna. V skutočnosti pre efektívne napätie na výstupe mostíka platí vzťah (5.1), v ktorom δ je z intervalu $< 0; 1 >$ a predstavuje striedu vstupného signálu. Závislosť výstupného napäťia a striedy je zobrazená na Obr. 5.2. V našom prípade sa ale nejedná o závažný nedostatok, ktorý by výrazným spôsobom ovplyvňoval výsledky regulácie.

$$U_{EF} = U_{MAX} \sqrt{\delta} \quad (5.1)$$

Ladenie RS prebiehalo po častiach. Ako prvý sme naladili PID B, pričom využitá bola Zieger-Nicholsova metóda, nasledovaná jemným, manuálnym doladením parametrov. Pri ladení bol na vstup privedený nulový vstupný signál, reprezentujúci požiadavku na nulový uhol náklonu. Výsledkom tohto ladenia bol robot schopný balansovať vo vzpriamenej polohe



Obr. 5.2: Znázormenie závislosti medzi striedou a napäťím

a taktiež odolať aj silnejším pokusom o jeho prevrhnutie. Ako nedostatkom sa ale javilo to, že robot nebol nijakým spôsobom penalizovaný za pohyb a tak trvalo aj niekoľko sekúnd kým sa po postrčení opäť dostal do ustáleného stavu (stav charakterizovaný minimálnou osciláciou okolo osi natočenia a minimálnou rýchlosťou pohybu).

Po naladení PID B, bol ladený PID A, pričom sme na vstup RS opäť priviedli nulovú hodnotu - teda požiadavku na nulovú rýchlosť. Kedže pri prevádzke bude práve tento stav s nulovou požiadavkou najčastejší pri ladení PID A sa kládol silný dôraz na správanie systému pri návrate do ustáleného stavu. PID A bol naladený obdobným spôsobom ako PID B. Výsledkom bol RS schopný zabezpečiť minimálnu uhlovú výchylku robota pri minimálnej rýchlosti pohybu.

Kedže pri realizácii PID bola použitá forma popísaná v 2.9 bolo ešte potrebné určiť konštantu T_f podľa vzťahu 2.8. Pri zisťovaní oscilačnej frekvencie robota pri pohybe sme na presné zmeranie krátkych períód oscilácií použili kamerový záznam pohybu robota. Pri známej frekvencií snímkovania, ktorá bola v našom prípade 60 FPS (frames per second), je spočítaním snímok a ich vynásobením períodou snímkovania možné presne odmerať aj krátke časové okamihy. Zavedenie konštanty T_f zlepšilo vlastnosti RS robota.

Takto naladené PID regulátory pracujú v zapojení do kaskády, pričom PID B pracuje s períodou 10ms a PID A s períodou 50ms. Tento čas je dostatočný na to aby bol robot schopný rýchlo reagovať na podnety a dostatočne dlhý na to aby sme mohli získať a spracovať dátu zo senzorov. Dôvodom relatívne dlhej períody PIDu B je princíp fungovania enkodéra, z ktorého by sme pri výrazne kratšej període meraní neboli schopní získať presné dátu.

5.2 Implementácia riadiaceho systému

Implementácia RS robota bola realizovaná v jazyku C++, pričom sme využili objektovo orientovaný spôsob programovania. V praxi to znamená, že jednotlivé funkčné celky robota sú reprezentované v dátových štruktúrach nazývaných objekty. Každý objekt predstavuje inštanciu triedy, v ktorej sú definované jeho vlastnosti a metódy - teda jeho štruktúra. prostredníctvom týchto vlastností a metód objekt vykonáva v rámci funkčného celku určité

funkcie. Výhodou tohto postupu je, že takto napísaný kód je znova použiteľný aj v iných projektoch a v prípade potreby jednoducho rozšíriteľný o dodatočnú funkcionality. Zapuzdrenie jednotlivých funkčných celkov taktiež pomáha udržiavať kód prehľadný.

Príkladom triedy v nami použitom kóde je napríklad trieda *PID*. V tejto triede sú definované metódy, ktoré by mal byť každý objekt triedy PID schopný realizovať. V našom prípade sa jedná o metódy pre nastavenie a úpravu parametrov P, I, D a metódu pre vypočítanie výstupu PIDu vzhľadom na určitý vstup. Vytvorenie tejto triedy nám umožnilo reprezentovať PID A a PID B ako inštancie tejto triedy.

Pri triede PID je treba poznamenať, že priame použitie 2.9 na vypočítanie výstupu by nebolo možné, keďže v nami predstavenej forme sa jedná o spojitý regulátor. Mikroprocesor je ale digitálne zariadenie, ktoré nie je schopné priamo realizovať spojité funkciu, ani prijímať spojité dátá (ktoré by väčšina nami použitých senzorov ani nebola schopná poskytnúť). Je preto potrebné vyjadriť 2.9 aj vo spojitej forme. Výstup PID pre vzorku číslo i je teda:

$$y_i = Pe_i + I \sum_{k=0}^i e_k \Delta t + D \Delta e_i \quad (5.2)$$

pričom e_i je rozdiel požadovaného uhla a skutočného uhla, pre vzorku i , Δt je čas medzi jednotlivými vzorkami a Δe_i je možné vyjadriť ako:

$$\Delta e_i = \frac{T_f \Delta e_{i-1} + (e_i - E_{i-1})}{\Delta t + T_f} \quad (5.3)$$

Lst. 1 predstavuje ukážku jednej nami implementovanej metódy triedy PID. Jej výstup predstavuje výstup daného PID regulátora.

Lst. 1: Príklad jednej z metód triedy PID

```

1 float PID::giveOutput(float input, float target, float dt, float constrainI){
2     float output = 0;
3     float Perror = target-input;
4     error_integral += Perror;
5     if(constrainI){
6         error_integral = constrain(error_integral,-constrainI,constrainI);
7     };
8     error_derivative = (Tf*error_derivative + (Perror - old_error))/(dt + Tf); //implements
9     filtering constant Tf
10    output = P*(Perror)+I*(error_integral)*dt+D*error_derivative;
11    old_error = Perror;
12    return output;
13 }
```

Celý nami vytvorený kód pracuje po úvodnej inicializácii premenných a vytvorení objektov na princípe kontrolnej slučky, t.j. skupiny príkazov, ktoré sa periodicky opakujú. V našom prípade sme túto slučku navrhli tak aby prebehla raz za každých 10 ms. V tejto riadiacej slučke načítavame nové údaje zo senzorov, spracovávame ich prostredníctvom RS a na vstup H-mostíka privádzame výsledný signál PWM. Celý proces je znázornený v diagrame Obr. D.1.

V Obr. D.1 nie sú znázornené spracovania prerušení, v ktorých sa spracúvajú údaje z enkodérov, použitých na monitorovanie aktuálnej rýchlosťi robota. Výstup zo samotných enkodérov prichádza do mikropočítača po štyroch dátových linkách (dve pre každý enkodér), vo forme impulzov vzájomne oneskorených o čas t_e . Výstup z dvoch kanálov enkodéra je zobrazený na Obr. E.1. Na jedinú otáčku kolesa pripadá N takto vzájomne posunutých impulzov (pre jeden kanál). Spočítaním týchto impulzov za čas t_S vieme určiť uhlovú rýchlosť kolesa ω a z nej následne odvodiť celkovú rýchlosť pohybu robota. Smer pohybu určíme podľa poradia detekcie impulzov. Vzorec pre výpočet rýchlosťi pohybu robota je teda možné vyjadriť ako:

$$\begin{aligned}\Delta\phi_w &= \frac{2\pi}{N}k \quad [rad] \\ \omega &= \frac{\Delta\phi_w}{t_S} \quad [rad.s^{-1}] \\ v &= \omega r \quad [m.s^{-1}]\end{aligned}$$

kde $\Delta\phi_w$ predstavuje zmenu uhla natočenia kolesa k predstavuje počet detegovaných impulzov z jednej linky za čas t_S a r polomer kolesa robota.

Následne je ešte potrebné zredukovať šum prítomný v takto nameraných dátach, v našom prípade pomocou komplementárneho filtra. Komplementárny filter bol taktiež použitý aj pri filtrovaní a fúzií dát nameraných z akcelerometra a gyroskopu.

Podrobnejšia schéma zobrazujúca celkovú činnosť RS robota je na Obr. 5.3. Znázorňuje ako RS robota, tak aj spôsob spracovania dát na jeho vstupoch.

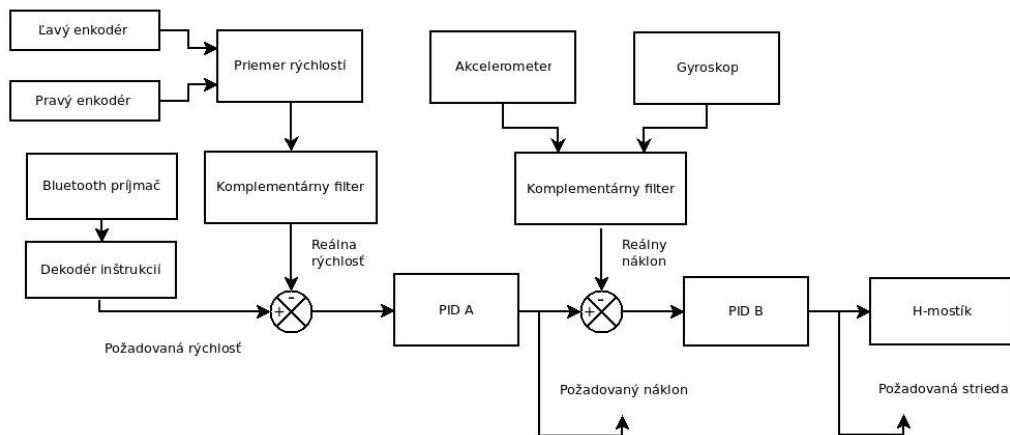
Zaujímavosťou je, že pri fúzií dát z gyroskopu a akcelerometra sa ukázalo ako výhodné v komplementárnom filtri s postupom času potláčať vplyv merania z akcelerometra až po

dosiahnutie určitej konečnej hodnoty. Dôvodom je, že pri spustení potrebuje robot chvíľu na to aby presne zistil svoj uhol natočenia, tento môže presne získať len z akcelerometra. Po zorientovaní a správnom nastavení polohy je výhodné potlačiť vplyv akcelerometra čím sa zamedzí vplyvu náhodných otriasov a silných postrčení na presnosť merania uhla (akcelerometer je na tieto omnoho citlivejší ako gyroskop).

5.3 Ovládanie pohybu robota

Použitím komunikačného rozhrania využívajúceho technológiu Bluetooth sme zabezpečili jednoduchú a dostatočne rýchlu komunikáciu medzi robotom a ovládačom, prostredníctvom ktorého je operátor schopný jednoducho zadávať robotovi príkazy. Následne ale bolo potrebné zabezpečiť aby sa tieto príkazy spoľahlivo premietali do riadenia pohybov robota. Taktiež bolo potrebné zabezpečiť aby robot nebol závislý od príkazov operátora a dokázal balansovať na mieste aj bez prítomnosti riadiaceho signálu.

Riadiaci signál prichádzajúci z ovládača prenáša riadiace príkazy vo forme jednotlivých bajtov, pričom prvý bajt slúži ako jedinečný identifikátor príkazu. Takýmto spôsobom sme zabezpečili jednoduché odlišenie až 256 jedinečných príkazov, ktoré robot môže od ovládača prijať. V prípade príkazu prenášajúceho pohybové povely bude tento prvý jedinečný identifikátor nasledovaný dvoma bajtami dát, ktoré obsahujú preškálované hodnoty z 10-bitového prevodníka ADC v ovládači, napojeného na X-ovú a Y-ovú os analógového joypadu.



Obr. 5.3: Podrobná schéma RS robota

Dosiahnutie plynulého riadenia rýchlosťi po prijatí riadiaceho bajtu, bolo jednoduché. Hodnota prijatá z X-ovej osi joypadu sa preškáluje z rozsahu 0-100 na $-v_d$ až v_d . Túto hodnotu po preškálovaní ozn. v_s a použitá je ako vstup do PID A. Po praktickom testovaní rôznych hodnôt v_d sme vo finálnej verzií robota použili hodnotu $v_d = 7$. V praxi hodnota v_d predstavuje max. rýchlosť pohybu robota v $rad.s^{-1}$.

Praktické testy ďalej ukázali, že pred vložením vypočítanej rýchlosťi v_s na vstup PID A je vhodné použiť komplementárny filter, ktorý spomalí rýchlosť nárastu v_s . Týmto postupom sme docieliли plynulejšie zrýchlenie a spomalenie robota, čo v konečnom dôsledku prispelo k celkovému zlepšeniu stability robota.

Ako problematické sa v praxi ale ukázalo riadenie smeru robota. Pri prvých pokusoch sme prijatú hodnotu z Y-osi joypadu preškálovali a filtrovali spôsobom analogickým ako pri riadení rýchlosťi a takto získanú hodnotu o_s sme používali pri riadení motorov. V prípade ľavotočivej zatáčky tak bola k hodnote požadovanej striedy pravého motora pripočítaná hodnota o_s a od striedy ľavého odpočítaná (opačne pre pravotočivú zatáčku). Tento systém spoľahlivo pracoval pri rozbehnutom robotovi, ale ukázal sa ako nedostatočný pri pokusoch otáčať nehybného robota.

Problémom pri otáčaní nehybného robota bol fakt, že v skutočnosti nie je možné dosiahnuť balansovanie robota bez rýchlych zmien striedy vstupujúcej do H-mostíka. Tieto zmeny striedy sú natoľko rýchle, že v praxi sa prejavia len jemným chvením robota, ale pri snahe upravovať striedu o hodnotu o_s dôjde len k trhaným pohybom robota na mieste. Tento nedostatok sa nám nepodarilo odstrániť ani zmenou parametrov komplementárneho filtra ani zmenami o_s .

Ako uspokojivé riešenie sa nakoniec ukázalo riadenie zatáčania za jazdy pomocou o_s a riadenie otáčania na mieste priamym zavedením predefinovanej striedy na oba motory, podľa požadovaného smeru otáčania a pri dodržaní maximálnej dovolenej rýchlosťi robota, pri ktorej je ešte možné otočku vykonať. Pri tomto spôsobe otáčania ostáva strieda motorov po určitý, krátkej čas stabilná a dostatočne veľká na to, aby spôsobila otočenie robota na mieste bez výrazného narušenia jeho stability.

5.4 Podporné metódy riadenia

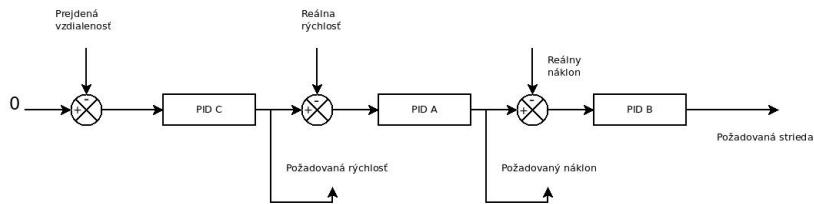
Pri testovaní RS sme zvažovali aj niektoré ďalšie, podporné, metódy riadenie správania sa robota. Tieto sme nepovažovali za natoľko kritické aby ich bolo nutné uplatniť ako hlavnú časť RS, ale keďže ich implementácia nebola náročná rozhodli sme sa ich zahrnúť do celkového dizajnu robota.

Jedným z menej závažných problémov pri nami navrhnutom robotovi bola jeho neschopnosť zotrvať na jednom mieste pri balansovaní s nulovou požiadavkou na rýchlosť. Aj keď tento problém sme do značnej miery eliminovali regulovaním požadovanej rýchlosťi, robot aj tak nebol schopný nijakým spôsobom určovať svoju pozíciu v priestore. Následkom toho bolo, že po postrčení sa robot niekedy nevrátil presne do miesta, v ktorom pôvodne balansoval. Tento problém sa prejavoval aj pri dlhodobom balansovaní bez externých stimulov. Kvôli náhodným osciláciám pri balansovaní a nerovnakou charakteristikou motorov (ktorú sa nepodarilo celkom odstrániť ani zavedením kalibračných konštánt) sa robot pomaly pohyboval priestorom v okolí bodu kde začal balansovať.

Prvou z týchto podporných metód bolo teda zaradenie dodatočného, tretieho PID C regulátora pred hlavnú RS Obr. 5.4 .Tento regulátor má na svojom vstupe celkovú prejdenú vzdialenosť robota za celú dobu jeho prevádzky, získanú z enkodérov. Jeho výstupom je požadovaná rýchlosť, pri ktorej dodržaní sa robot dostane späť na miesto, v ktorom začal balansovať.

Tento tretí PID C v praxi fungoval výborne, ale ani s jeho použitím sa však nepodarilo úplne odstrániť problém s nerovnomerným pohybom motorov. Tento spôsob riadenia je taktiež nepoužiteľný pri ovládanom pohybe, pri ktorom by zaradenie PID C spôsobovalo nestabilitu systému. Aj z tohto dôvodu sme sa rozhodli, že PID C bude pri ovládaní robota permanentne odpojený a zaradiť do RS sa bude dať len manuálne pokynom z ovládača a to len v režime balansovania na mieste.

Ďalšou metódou, ktorá mala pôvodne byť súčasťou hlavnej časti RS robota, bolo meranie elektrického prúdu vstupujúceho do H-mostíka. Myšlienkovu bolo navrhnutú a skonštruovať



Obr. 5.4: Upravený RS

elektrický obvod, ktorý by s použitím tohto merania a dát z mikropočítača bol schopný generovať požadovaný prúd pre motory.

Výhodou tohto postupu by bolo presnejšie riadenie motorov. Pri nízkych rýchlosťach je totiž točivý moment motorov úmerný prúdu, ktorý nimi tečie - nie napätiu samotnému. Pri balansovaní na mieste kde sú potrebné rýchle a presné reakcie motorov, by sme teda teoreticky mohli priamym riadením momentu motorov dosiahnuť menšie oscilácie pri pohybe.

Meranie prúdu bolo zabezpečené rezistorom s malým odporem Obr. F.1 zaradeným do série s H-mostíkom. Meraním úbytku napäťa na tomto odporu sme pri známom odpore R schopní vypočítať prúd tečúci odporom do H-mostíka. Toto zapojenie bolo realizované na nami navrhnutom shielde a v praxi sme takto boli naozaj schopní zistiť prúd do motorov.

6 | LABORATÓRNE OVERENIE FUNKČNOSTI

V tejto kapitole predstavíme vlastnosti nami skonštruovaného robota pri laboratórnych testoch. Zameriame sa prevažne na správanie sa robota v režimoch statického balansovania a jeho schopnosť presúvať sa po šikmých plochách.

Pri režime statického balansovania na mieste sa zameriame na priemernú odchýlku od nulového náklonu, amplitúdu oscilácií a celkovú prejdenú vzdialenosť (so zaradením PIDu C aj bez neho).

Všetky grafy požité v tejto kapitole boli vytvorené s použitím nami napísaného python scriptu a dát v nich zobrazené boli získané prostredníctvom sériovej komunikácie medzi počítačom a robotom.

6.1 Základné parametre robota

Pre nami vytvoreného robota sme namerali nasledujúce parametre. Tie by mali byť dostatočné pre vytvorenie jednoduchého matematického modelu tohto robota.

- | | |
|--------------------------------------|---------------------------------|
| • $R = 7,2 \Omega$ | • $r = 3,4 \times 10^{-2} m$ |
| • $K_e = 3,34 \times 10^{-1}$ | • $m_k = 4,7 \times 10^{-2} kg$ |
| • $K_m = 9,3 \times 10^{-2}$ | • $m_p = 1,07 kg$ |
| • $J_k = 1,598 \times 10^{-3} kgm^2$ | • $L = 0,145 m$ |
| • $J_s = 4,025 \times 10^{-2} kgm^2$ | |

Pri meraní zotrvačnosti kolesa sme použili vzorec 6.1:

$$J_k = m_k r^2 \quad (6.1)$$

a pre zotrvačnosť celého robota:

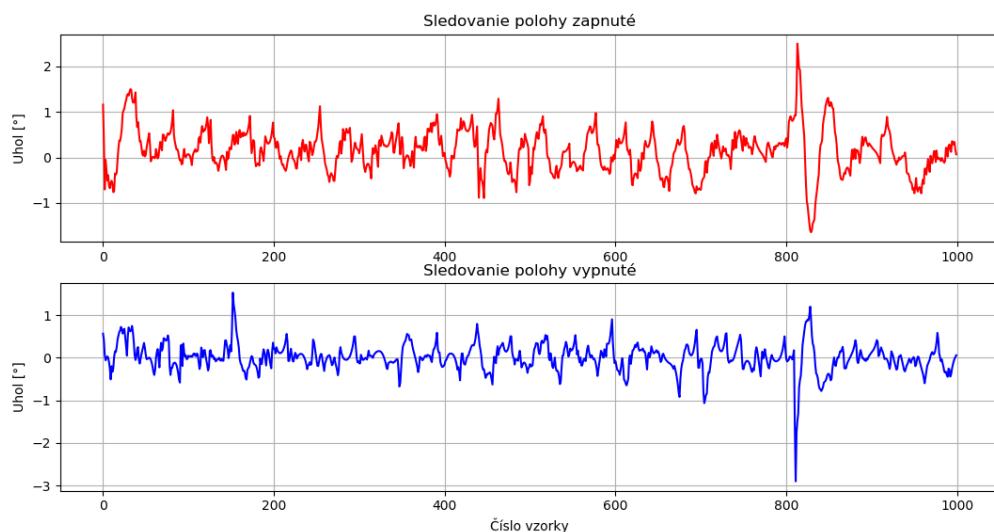
$$J_s = \frac{m_p g L T^2}{4\pi^2} \quad (6.2)$$

kde T je perióda oscilácie robota okolo osi kolies po vychýlení o malý uhol.

6.2 Parametre robota pri balansovaní na mieste

Pri balansovaní na mieste môže užívateľ zvoliť z dvoch režimov prevádzky. V prvom režime robot balansuje bez toho aby sa snažil udržať počiatočnú pozíciu, v druhom sleduje celkovú prejdenú vzdialenosť a snaží sa ju udržať na minime. Druhý stav je výhodný najmä v prípade ak sa robot nachádza na naklonenej plošine, kde mu sledovanie prejdenej vzdialenosť umožňuje pridávať rýchlosť natoľko aby sa udržal v nehybnosti.

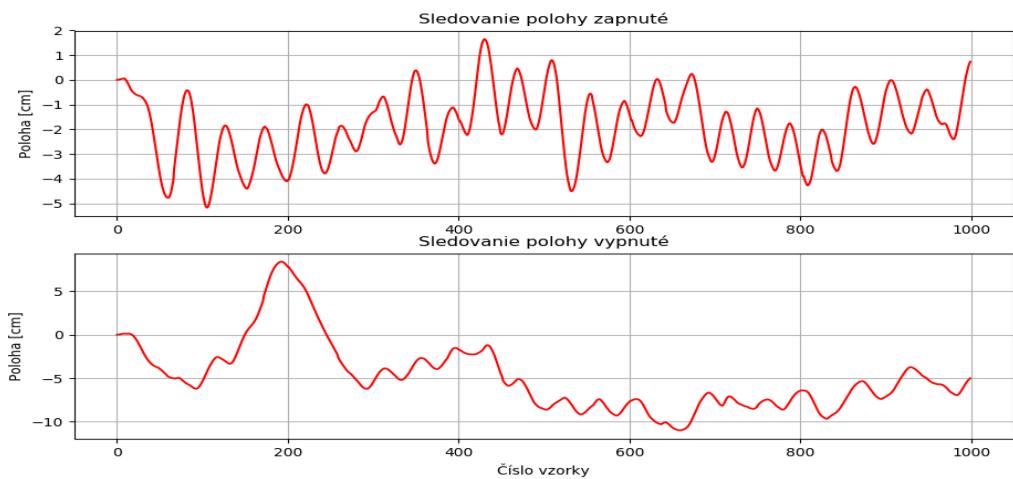
Testovanie správania robota prebiehalo po pripojení k počítaču dlhým, flexibilným káblom, ktorý nám umožnil komunikáciou cez sériovú linku zhromaždiť údaje o stave robota v jednotlivých časových okamihoch. Pomocou týchto dát boli následne vytvorené grafy zobrazené na Obr. 6.1. Na osi Y sa nachádzajú odchýlky od pravého uhla pre jednotlivé sledované vzorky. Čas medzi jednotlivými vzorkami bol $t_{vz} = 50ms$, čiže celkový časový interval, počas ktorého sme sledovali správanie robota trval 50 sekúnd.



Obr. 6.1: Náklon robota v čase

Z porovnania grafov na Obr. 6.1 je zrejmé, že pri vypnutom sledovaní polohy bol robot

schopný presnejšie regulovať uhlovú odchýlku od pravého uhla. V tomto režime odchýlka málokedy prekročila hodnotu 1° , čo sa nedá povedať o režime, v ktorom bolo zapnuté sledovanie polohy. Nevýhodou ale bolo, že aj za relatívne krátky čas merania došlo k posunu robota o viac než 5 cm oproti počiatočnej polohe, čo v prípade merania v druhom režime neboli problém. Pri sledovaní polohy vzdialenosť robota od počiatočnej polohy nikdy výrazne neprekročila 5 cm pričom mal vždy robot tendenciu vracať sa do počiatočnej polohy a oscilovať okolo nej Obr. 6.2.



Obr. 6.2: Zmena polohy robota počas testu

Okrem balansovania na rovine sme testovali aj správanie robota pri balansovaní na náklonenej rovine, ktorej sklon bol 15° . Pri tomto teste bol trvalo zapnutý režim so sledovaním polohy keďže, ako už bolo spomenuté, robot neboli bez neho schopný ani zotrať na plošine.

Z tohto testu podľa očakávania vyplynulo, že robot vykazoval pri takomto balansovaní zhoršené vlastnosti oproti pohybu na rovine. Oscilácie okolo počiatočného bodu boli väčšie, robot bol však nadálej schopný zotrať v okolí počiatočnej pozície. Toto ale platilo len pre pohyb v smere náklonu plošiny. Keďže servo pohon neboli v čase písania tejto práce ešte zakomponovaný do riadenia robota, neboli schopní nakloniť šasi tak, aby ostal robot stabilný aj pri pohybe priečnom voči uhlu náklonu plošiny. Tento nedostatok sa prejavoval nerovnomerným pohybom robota, hlavne pri náhlych smeroch jazdy.

7 | ZHRSNUTIE DOSIAHNUTÝCH VÝSLEDKOV

V tejto práci sme sa venovali popisu činnosti práce pri návrhu a skonštrúovaní dvojolesového balansujúceho robota. Nami vyrobený robot pri meraniach dosiahol uspokojivé výsledky ako pri samostatnom balasovaní na mieste tak aj pri riadenom pohybe na rovine a naklonenej plošine. Za účelom riadenia robota bol taktiež skonštruovaný ovládač, umožňujúci komunikáciu operátora s robotom.

V rámci práce vznikol taktiež matematický model robota, ktorý neboli priamo použity v práci, ale v prípade potreby môže byť použitý pri návrhu pokročilejších regulátorov ako napríklad LQR.

ČESTNÉ VYHLÁSENIE

Vyhlasujem, že som zadanú prácu vypracoval samostatne, pod odborným vedením vedúceho práce, ktorým bol Ing. Dušan Nemeč, PhD a používal som len literatúru uvedenú v práci.

Súhlasím so zverejnením práce a jej výsledkov.

Dátum odovzdania práce, Žilina

podpis

Žilinská univerzita v Žiline

Elektrotechnická fakulta

Evidenčné číslo práce

**NÁVRH A KONŠTRUKCIA DVOJKOLESOVÉHO
BALANSUJÚCEHO ROBOTA**

PRÍLOHOVÁ ČASŤ

2017

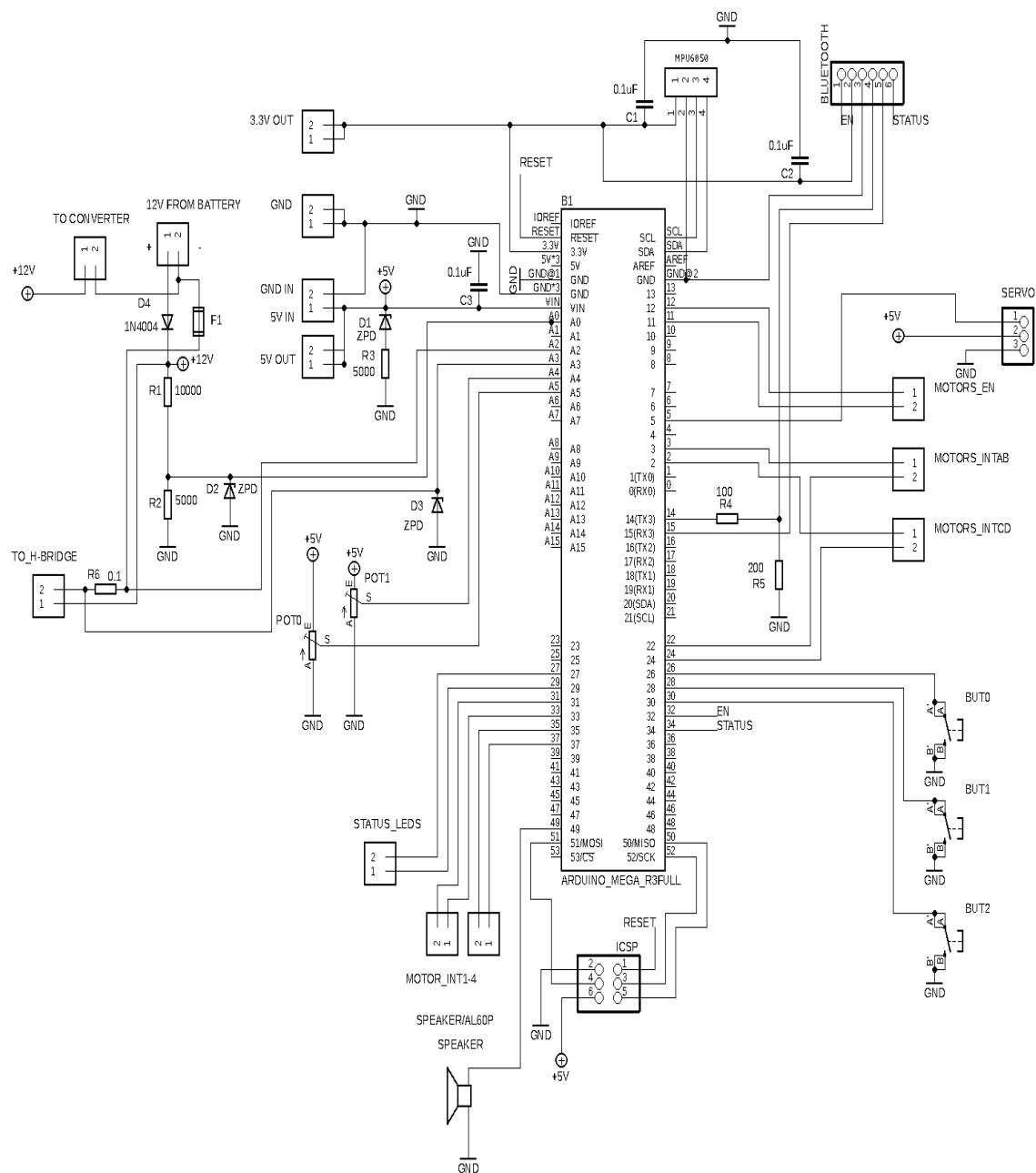
Daniel Adamkovič

PRÍLOHA A | ZIEGLER-NICHOLSOVA TA- BULKA

| | K_p | K_i | K_d |
|-----|----------|-----------------------|---------------|
| P | $0.5K_p$ | - | - |
| PI | $0.5K_p$ | $\frac{0.54K_p}{T_k}$ | - |
| PD | $0.5K_p$ | - | $0.02K_pT_k$ |
| PID | $0.5K_p$ | $\frac{1.2K_p}{T_k}$ | $0.075K_pT_k$ |

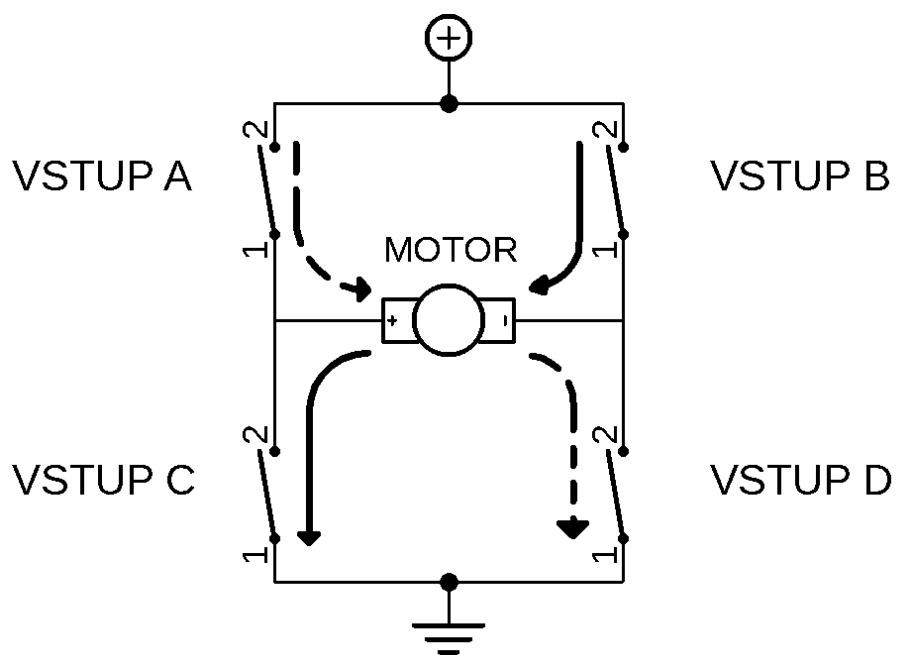
Tab. A.1: Zieger-Nicholsova tabuľka

PRÍLOHA B | SCHÉMA SHIELDU



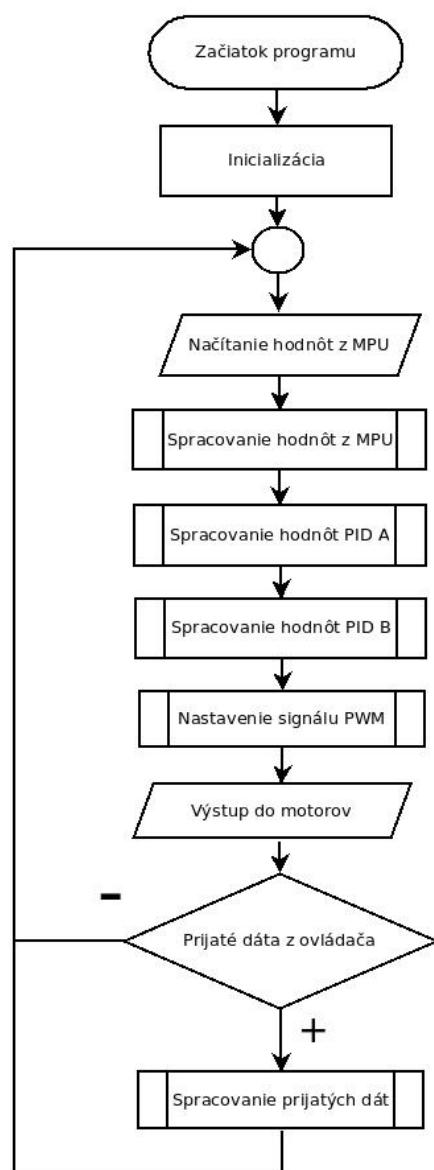
Obr. B.1: Schéma shieldu pre Arduino

PRÍLOHA C | SCHÉMA H-MOSTÍKA



Obr. C.1: Schematické znázornenie H-mostíka

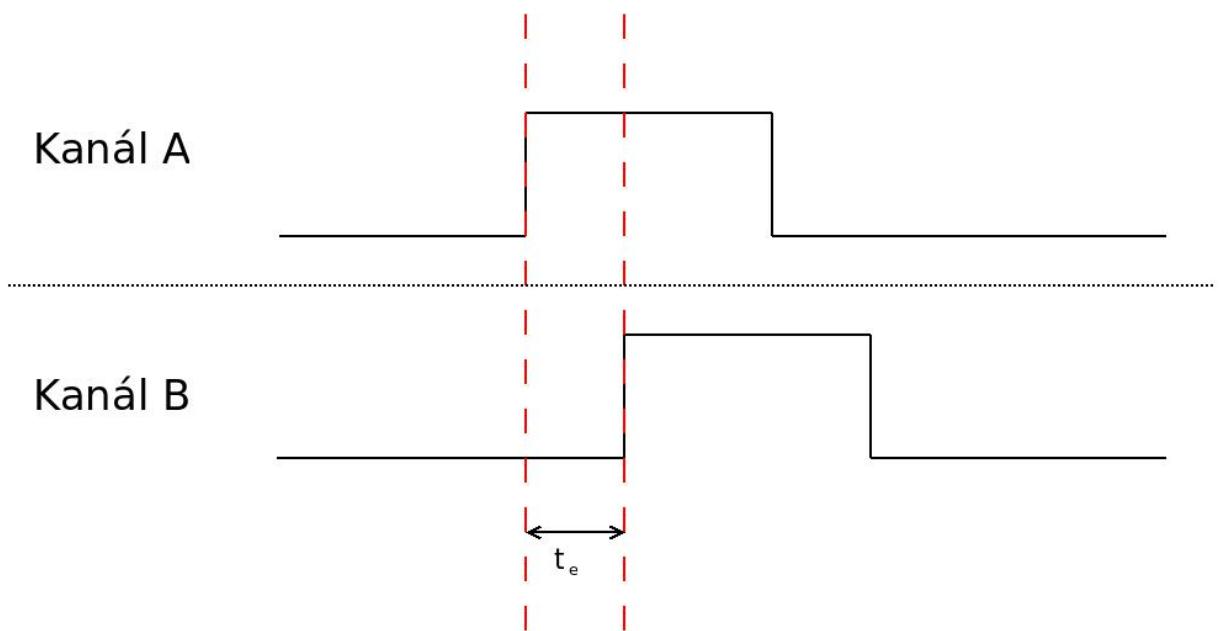
PRÍLOHA D | DIAGRAM PROGRAMU



Obr. D.1: Diagram programu.

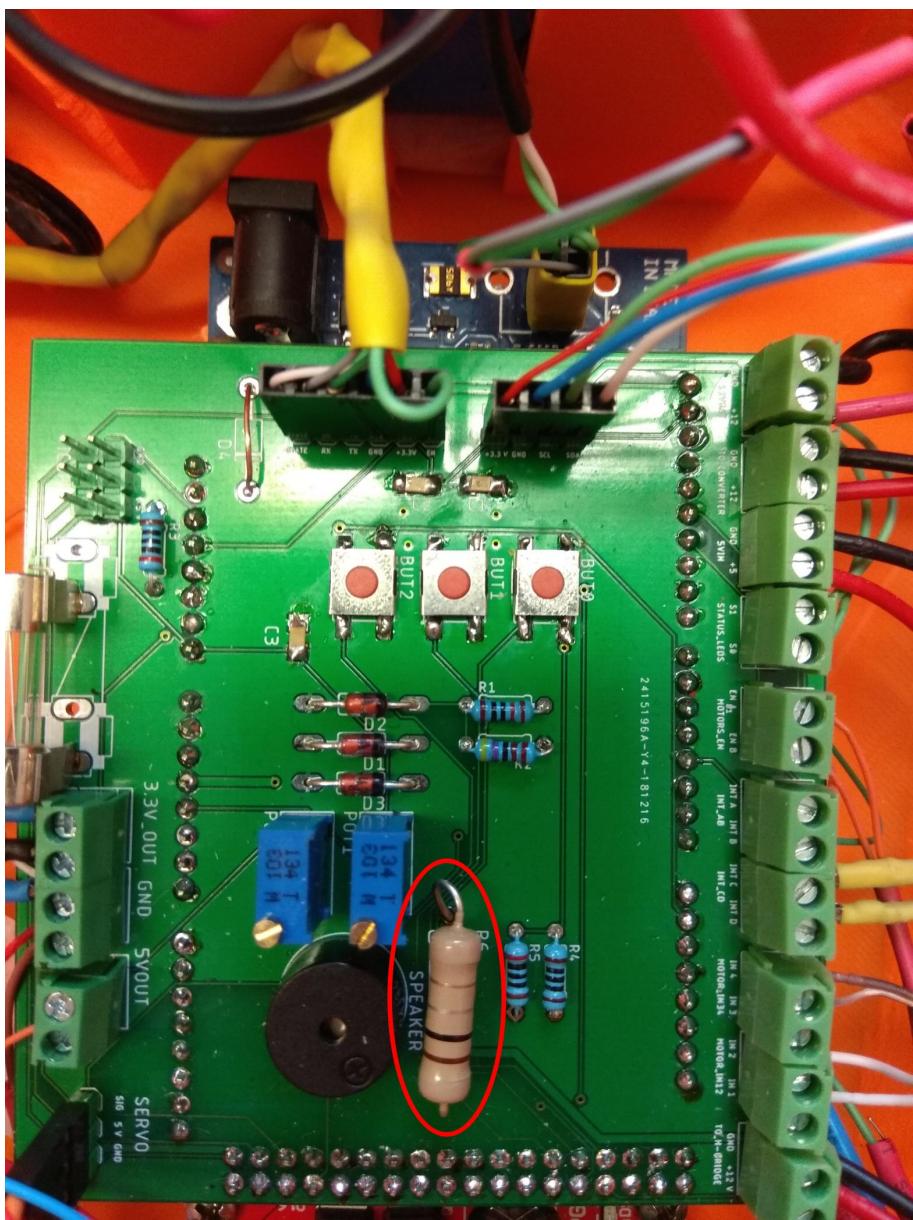
^oPozn.: V diagrame nie sú kvôli prehľadnosti znázornené obsluhy prerušení, v ktorých sa aktualizujú údaje z enkodérov.

PRÍLOHA E | VÝSTUP Z ENKODÉRA



Obr. E.1: Výstup z kanálov enkodéra

PRÍLOHA F | SNÍMANIE PRÚDU



Obr. F.1: Shield s vyznačeným rezistorom na snímanie prúdu