



**ČESKÉ VYSOKÉ
UČENÍ TECHNICKÉ
V PRAZE**

F2

**Fakulta strojní
Ústav přístrojové a řídicí techniky**

Bakalářská práce

Návrh konstrukce a řízení dvoukolového autonomního roboty

František Kráčmar
Informační a automatizační technika

2019

Vedoucí práce: Ing. Jaroslav Bušek



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Kráčmar** Jméno: **František** Osobní číslo: **460008**
Fakulta/ústav: **Fakulta strojní**
Zadávající katedra/ústav: **Ústav přístrojové a řídicí techniky**
Studijní program: **Strojírenství**
Studijní obor: **Informační a automatizační technika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Návrh konstrukce a řízení dvoukolového autonomního robota

Název bakalářské práce anglicky:

Frame and control design of two-wheeled autonomous robot

Pokyny pro vypracování:

Navrhnete prototyp robota k testování algoritmů autonomního řízení ve volném prostoru se záměrem aktivní účasti na soutěži ARLISS. Pro vypracování proveďte:

- 1) proveďte rešerši možných řešení konstrukce robota pro uvedenou soutěž
- 2) proveďte rešerši na téma základního sensorového vybavení pro autonomní pozemní navádění ve volném prostoru
- 3) navrhnete vhodnou konstrukci robota a řídicí elektroniky
- 4) realizujete základní řízení autonomního navádění robota za pomoci nutných senzorů
- 5) proveďte testování navrženého prototypu a zhodnoťte dosažené výsledky

Seznam doporučené literatury:

CARON, Francois, et al. GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects. Information fusion, 2006, 7.2: 221-230.
PETOVELLO, Mark G. Real-time integration of a tactical-grade IMU and GPS for high-accuracy positioning and navigation. Calgary, AB: University of Calgary, Department of Geomatics Engineering, 2003.
MAALOUF, Elie. Nonlinear control of wheeled mobile robots. 2005. PhD Thesis. École de technologie supérieure.

Jméno a pracoviště vedoucí(ho) bakalářské práce:


Ing. Jaroslav Bušek, U12110.3

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

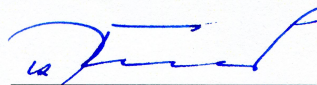
Datum zadání bakalářské práce: **26.04.2019**

Termín odevzdání bakalářské práce: **12.06.2019**

Platnost zadání bakalářské práce: _____


Ing. Jaroslav Bušek
podpis vedoucí(ho) práce


podpis vedoucí(ho) ústavu/katedry

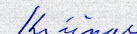

prof. Ing. Michael Valášek, DrSc.
podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

26-04-2019

Datum převzetí zadání


Podpis studenta

Poděkování / Prohlášení

Rád bych poděkoval Ing. Jaroslavu Buškovi, vedoucímu mé bakalářské práce, za odborné vedení, připomínky a čas, který mi věnoval. Dále bych chtěl poděkovat rodině za podporu během celého mého studia.

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne 10. 6. 2019

.....

Abstrakt / Abstract

Cílem práce je realizace prototypu autonomního robota dle pravidel soutěže ARLISS, která má svými požadavky simulovat orbitální mise na planetě Mars. Úvodem je přehled dosavadních přístupů k řešení soutěžního úkolu. Následně práce popisuje vývoj a výrobu vlastního robota obsahující návrh šasi, volbu výkonové a řídicí elektroniky včetně kompletního vývoje desky plošných spojů. Dále pak algoritmus pro autonomní jízdu v otevřeném prostoru. Pozornost je věnována návrhu Kalmanova filtru a jeho využití pro přesnější určení polohy robota. Je představena jeho základní struktura s konkrétní implementací pro tuto úlohu. S tím je spojen experiment pro identifikaci stejnosměrného motoru.

Klíčová slova: robot, diferenciální řízení, IMU, GPS, autonomní navigace, Kalmanův filtr, identifikace DC motoru.

The aim of this work is to realize a prototype of an autonomous robot according to the rules of the ARLISS competition, which is supposed to simulate orbital missions on the planet Mars. The introduction is an overview of existing approaches to solving the competition task. Subsequently, the thesis describes development and production of own robot containing chassis design, choice of power and control electronics including complete development of printed circuit board. Furthermore, the algorithm for autonomous driving in open space. Attention is paid to the Kalman filter design and its use for more precise positioning of the robot. Its basic structure with concrete implementation for this task is presented. This is connected to an experiment to identify a DC motor.

Keywords: robot, differential steering, IMU, GPS, autonomous navigation, Kalman filter, DC motor identification.

Obsah /

1 Úvod	1
2 Soutěž ARLISS	2
2.1 Cesta po zemi	3
2.2 Cesta vzduchem	5
3 Senzorové vybavení pro autonomní pozemní navádění ve volném prostoru	7
3.1 Senzory pro měření absolutní polohy	7
3.1.1 Globální družicový polohový systém	7
3.1.2 Sonar, radar a lidar	9
3.1.3 Magnetometr	12
3.2 Senzory pro měření relativní polohy	12
3.2.1 Akcelerometr	12
3.2.2 Gyroskop	13
3.2.3 Rotační enkodéry	13
4 Návrh rámu roveru	15
5 Návrh řídicí elektroniky	17
6 Řídicí algoritmus pro autonomní jízdu	20
6.1 Určení regulační odchylky na základě rozdílu azimutů	20
6.2 Stanovení aktuálního azimutu pomocí Madgwickova filtru .	21
6.3 Kompletní řídicí algoritmus pro autonomní jízdu	22
7 Určování polohy pomocí Kalmanova filtru	24
8 Experimentální identifikace DC motoru	26
8.1 Návrh experimentu pro stanovení frekvenční charakteristiky motoru	26
8.2 Program pro řízení experimentu	27
8.3 Stanovení přenosové funkce stejnosměrného motoru	30
8.3.1 Odvození diferenciální rovnice stejnosměrného motoru	30
8.3.2 Zpracování experimentálních dat	32
9 Testování navrženého prototypu	34
10 Závěr	36
Literatura	37
A Seznam použitých symbolů a značení	40
B Výrobní výkresová dokumentace navrženého rámu	43
C Výkresová dokumentace navržené desky plošných spojů	49
D Řídicí program mikrokontroléru .	54
E MATLAB skript pro identifikaci přenosové funkce	61

Tabulky / Obrázky

3.1. Porovnání GPS přijímačů NEO-6M a NEO-8P.....9	2.1. Zadání soutěžního úkolu OpenClass2
4.1. Parametry nákladového pro- storu 15	2.2. Rover Tahaku University 2005 ..3
5.1. Proudový odběr jednotlivých komponent..... 19	2.3. Mechanismus UL Lafayette 2016.....3
	2.4. Padák UL Lafayette 2015.....4
	2.5. Zamotaný padák UL Lafa- yette 20154
	2.6. Rover UL Lafayette 20155
	2.7. Letoun TU Tokyo 20185
	2.8. Kvadrotéra FS ČVUT 20156
	2.9. Flying Hopper Georgia Tech.....6
	3.1. Určení 2D polohy pomocí signálu z jednoho satelitu8
	3.2. Určení 2D polohy pomocí signálů ze tří satelitů.....8
	3.3. Určení 3D polohy pomocí signálů ze čtyř satelitů8
	3.4. Princip fungování diferenci- ální GPS9
	3.5. Princip fungování lidarů 10
	3.6. Rozlišení detailů objektů li- darů a radarů..... 10
	3.7. Kapitol Spojených států amerických vizualizovaný jako mračno bodů 11
	3.8. Mapa prostoru vzniklá meto- dou SLAM..... 11
	3.9. Vznik několika ech pro jeden emitovaný signál 11
	3.10. Struktura kapacitního akce- lerometru 13
	3.11. Disk absolutního rotačního enkodéru 14
	4.1. Výsledný CAD návrh rámu roveru 16
	5.1. Blokové schéma řídicí elek- troniky 18
	5.2. Aktuální verze desky ploš- ných spojů 18
	6.1. Zvolený souřadnicový systém .. 20
	6.2. Diferenční řízení..... 21
	6.3. Vývojový diagram řídicího algoritmu 23
	7.1. Schéma jednoho cyklu Kal- manova filtru 24

8.1.	Třívodičové zapojení AS5048 ..	26
8.2.	Časový průběh signálů na sběrnici SPI	27
8.3.	Zapojení měřící aparatury	27
8.4.	LabVIEW program pro řízení experimentu	28
8.5.	Průběh budícího signálu „chirp“	29
8.6.	Nastavené parametry bloku „SPI“	30
8.7.	Náhradní schéma stejnosměrného motoru	30
8.8.	Datový tok na sběrnici SPI	32
8.9.	Frekvenční charakteristika identifikované funkce	33
8.10.	Průběh úhlových rychlostí během experimentu a simulace	33
9.1.	Záznam trajektorie testovací jízdy	35

Kapitola 1

Úvod

Cílem této práce je navržení prototypu robota dle pravidel soutěže ARLISS, jež svými požadavky simuluje orbitální mise na Zemi či Marsu.

V první části bakalářské práce je představena samotná soutěž a její kategorie. Následuje přehled dosud použitých typů konstrukcí robotů pro vybranou kategorii.

Další kapitola slouží jako přehled senzorů pro autonomní navigaci ve volném prostoru. Jsou vysvětleny principy jejich funkce, popsány důležité vlastnosti a částečně jsou zařazeny do historického kontextu.

Praktická část práce se zabývá již samotným návrhem a výrobou robota. Nejprve návrhem hliníkového rámu dle zvoleného typu robota, volbou výkonové a řídicí elektroniky a návrhem řídicího algoritmu pro plně autonomní jízdu.

Dále je nastíněno využití Kalmanova filtru pro řízení robota, k čemuž byla provedena experimentální identifikace použitých stejnosměrných motorů.

Bakalářská práce stejně jako robot byla vytvořena s úmyslem podnítit další pokračování, všechny součásti navrženého prototypu včetně softwaru proto budou volně dostupné online.

Kapitola 2

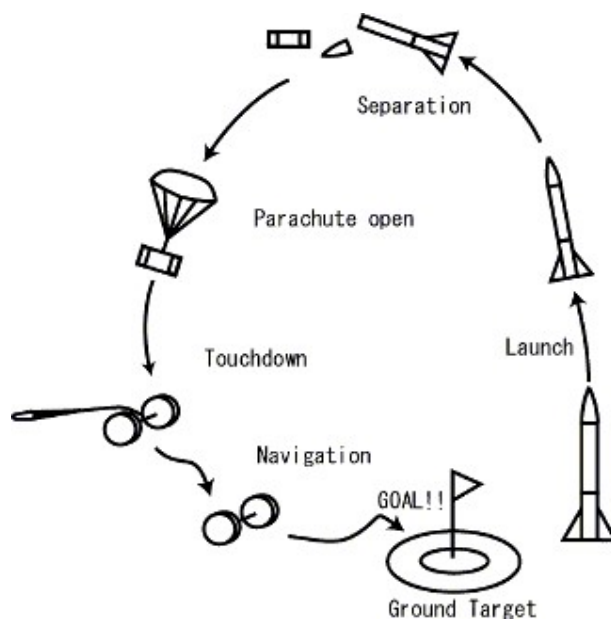
Soutěž ARLISS

Tato práce se zabývá mým vlastním návrhem autonomního robota. Nicméně bude užitečné, pokud na začátek provedu rešerši dosavadních přístupů jiných univerzit. Jednak byla tato rešerše mým prvním krokem před samotným započítím návrhu robota, neboť sám mám se soutěží ARLISS nulové zkušenosti. Po analýze konkurenčních robotů jsem se tak mohl předem vyvarovat nevhodným řešením a naopak inspirovat těmi úspěšnými. Dále tento přehled považuji právě vhodným pro vhléd do samotné soutěže, neboť na něm lze vidět nejen vývoj konstrukcí robotů, ale především nápaditost zahraničních studentů a jejich schopnost využívat nejmodernější technologie.

ARLISS má dvě soutěžní kategorie. První, CanSat, spočívá v tom, aby studenti vyvinuli sondu, která bude schopna při svém letu sbírat experimentální data a následně je vyhodnocovat. Kategorie se jmenuje CanSat, neboť soutěžní satelity se svou velikostí musí vejít do nápojové plechovky.

Druhou kategorií je tzv. OpenClass. Její zadání (viz obr. 2.1) by mělo simulovat orbitální misi na Marsu či Zemi. Soutěžní úkol začíná vynesemím robota v nákladovém prostoru do výšky zhruba 3 km, odkud se musí libovolným způsobem plně autonomně dopravit na zadané souřadnice vzdálené až 6 km od místa startu.

Robot, který je tématem této bakalářské práce, spadá právě do kategorie OpenClass a také z toho důvodu je rešerše konkurenčních konstrukcí provedena pouze pro tuto kategorii.



Obrázek 2.1. Zadání soutěžního úkolu OpenClass [1]

Kapitola 3

Senzorové vybavení pro autonomní pozemní navádění ve volném prostoru

„Robotika je inteligentním spojením mezi vnímáním a činností.“ [11]

Řešíme-li robotickou úlohu, zpravidla je v jejím rámci zapotřebí určovat polohu robota či jeho aktuátoru, neboť jedině tak je schopen fyzicky interagovat se svým okolím. Bez senzorů by tak jen stěží šla použít zpětná vazba, která nám již desítky let slouží k udržování ekvilibria systémů. S přihlédnutím k tématu bakalářské práce se tato kapitola zaměří na nejběžněji používané senzory, které slouží k orientaci pozemního robota v otevřeném prostoru. Senzory jsem rozdělil do dvou skupin, na absolutní, určující svou polohu vzhledem ke svému okolí, a relativní, které polohu odvozují od zvoleného počátečního bodu.

3.1 Senzory pro měření absolutní polohy

Při pohybu robota ve volném terénu je nezbytné zajistit, aby dokázal zjistit svou polohu – přesněji řečeno svou polohu vůči okolí v absolutním souřadném systému. Typicky se údaj o poloze používá – stejně jako v případě soutěže ARLISS – pro navigaci. Absolutní senzory polohy zpravidla slouží pro lokalizaci zařízení jako celku – příkladem může být GPS navigace v automobilu.

3.1.1 Globální družicový polohový systém

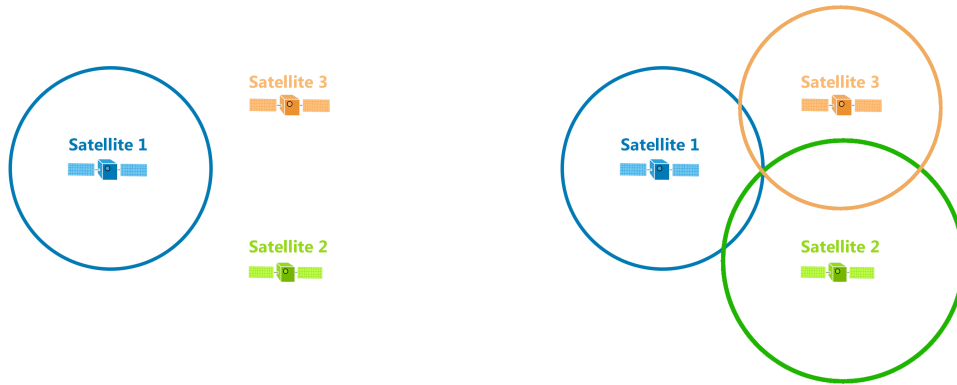
Nejběžnějším způsobem určování absolutní polohy je využití globálního družicového polohového systému (dále jen GNSS z anglického Global Navigation Satellite System), kdy je možné za pomoci signálů z těchto družic v reálném čase určovat pozici v prostoru.

Struktura systémů GNSS se skládá ze tří segmentů: kosmického, řídicího a uživatelského. Kosmický segment je tvořen satelity obíhajícími na oběžných drahách kolem Země. „Řídicí a kontrolní segment monitoruje kosmický segment, zasílá povely družicím, provádí jejich manévry a údržbu atomových hodin.“ [12] Uživatelský segment je pak tvořen samotnými přijímači, které pasivně přijímají signály jednotlivých satelitů.

GNSS přijímač na základě přijatých dat od těchto družic zjišťuje svou vzdálenost od nich. Jelikož se signál šíří do všech směrů stejně, tak při znalosti vzdálenosti od jedné družice víme, že se nacházíme někde na povrchu koule se středem v této družici a poloměrem právě oné vzdálenosti. 2D schéma tohoto případu lze vidět na obr. 3.1.

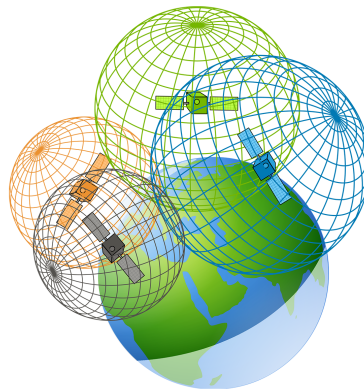
Obdobně pokud (ve 2D případě) přijímač zachytí signály od 3 satelitů, jejichž vzájemná poloha je známá, lze pozici přijímače určit jako průsečík těchto kružnic – viz obr. 3.2. Tomuto principu se říká trilaterace.

Pokud se přeneseme zpět do 3D, je zapotřebí ještě čtvrtý satelit pro určení 3. rozměru. Viz obr. 3.3.



Obrázek 3.1. Určení 2D polohy pomocí signálu z jednoho satelitu [13]

Obrázek 3.2. Určení 2D polohy pomocí signálů ze tří satelitů [13]



Obrázek 3.3. Určení 3D polohy pomocí signálů ze čtyř satelitů [13]

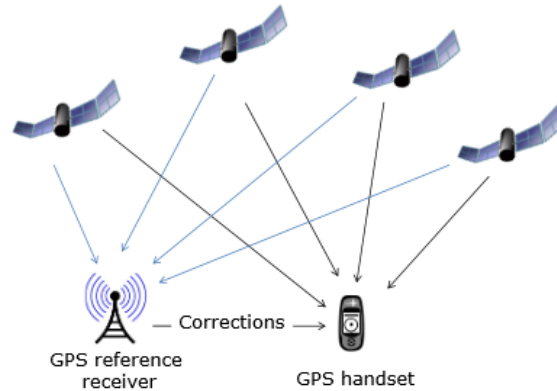
Tím nejznámějším GNSS systémem je bezesporu americký GPS (Global Positioning System), jehož satelity Zemi obíhají již od roku 1978. Přesnost určení polohy antény přijímače GPS je v současnosti až 30 cm. Tomu tak ale nebylo vždy, ještě do 1. 5. 2010 americká vláda do signálů z družic pro komerční účely přidávala umělou odchylku, která dosahovala desítek metrů.

V současnosti k GPS přibývají mezinárodní konkurenti. Jmenovitě ruský GLONASS, čínský Beidou (ten však v současnosti pokrývá pouze asijsko-pacifický region) a evropský navigační systém Galileo, který spravuje Evropská kosmická agentura. Každý navigační systém pak má na oběžné dráze typicky kolem 30 satelitů.

Co se týče GNSS přijímačů, ty se dělí především podle toho, od satelitů jakých navigačních systémů jsou schopny přijímat signály. Mezi další parametry pak patří: rychlost prvního určení polohy po zapnutí, citlivost antény, obnovovací frekvence a přesnosti určování pozice a rychlosti. Dále je lze dělit podle toho, jaké dodatečné funkce nabízejí.

Mezi tyto dodatečné funkce patří další zpřesnění polohy použitím asistované-GNSS (Assisted-GNSS). Kdy přijímač jednak přijímá signály ze satelitů, zároveň však komunikuje s lokální mobilní sítí, která mu zasílá data z lokálního serveru a zároveň slouží jako pomocná síť vysíláče signálu ze kterého lze určit polohu.

Na obdobném principu funguje také diferenciální GNSS. Její princip spočívá v tom, že mobilní přijímač svou polohu dále koriguje o korekci získanou z referenčního přijímače – ten musí být fixně umístěný a samozřejmě být schopný svou polohu určovat přesněji než přijímač mobilní. Schéma lze vidět na obr. 3.4



Obrázek 3.4. Princip fungování diferenciatní GPS [14]

Na závěr uvádím porovnání dvou volně dostupných senzorů od firmy U-blox. Model NEO-6M je starý model, který se běžně prodává na čínských eshopech (proto tak nízká cena). Právě i proto je hojně používán pro hobby využití. Oproti tomu NEO-8P je označován jako „High Precision GNSS Module“, především pro možnost fungování v režimu diferenciatní GNSS a tím pozici určovat s vyšší frekvencí a řádově přesněji než NEO-6M.

parametr	NEO-6M	NEO-8P
Studený start	36 s	26 s
Horký start	1 s	1 s
Citlivost navigace	-162 dBm	-160 dBm
Obnovovací frekvence	5 Hz	10 Hz
Přesnost polohy	2,5 m	až 0,025 m
Další funkce	A-GPS	A-GPS, diferenciatní GNSS
Cena (k 28.5.2019)	\$ 4	\$ 149

Tabulka 3.1. Porovnání GPS přijímačů NEO-6M a NEO-8P

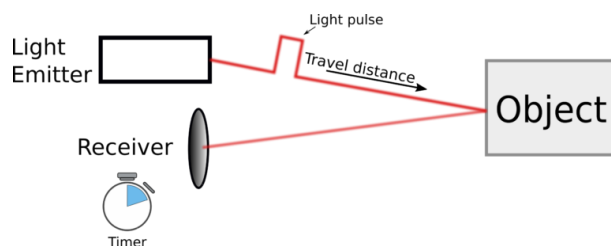
„Využívání možností GNSS je i silným impulsem pro ekonomický a průmyslový rozvoj každé země. Jen trh s těmito produkty a službami roste ročně o 25 %. Očekává se, že v roce 2020 budou v provozu asi 3 miliardy přijímačů družicové navigace. Družicová navigace se stále více stává součástí každodenního života občanů, nejen v jejich automobilech a mobilních telefonech, ale také v rámci energetických rozvodných sítí nebo v bankovních systémech a mnoha dalších službách.“ [15]

3.1.2 Sonar, radar a lidar

Absolutní polohu v prostoru lze také určit senzory, které jsou založeny na principu doby letu. I zde je pozice od pevných bodů odvozena z jejich vzdáleností, nicméně v tomto případě je senzor aktivní a sám vysílá měřený signál.

Pracovní postup senzoru vyobrazený na obr. 3.5 je následující:

- vysílačem je vyslán signálový impuls
- signál narazí na překážku
- signál se od překážky odrazí
- signál je zaznamenán přijímačem

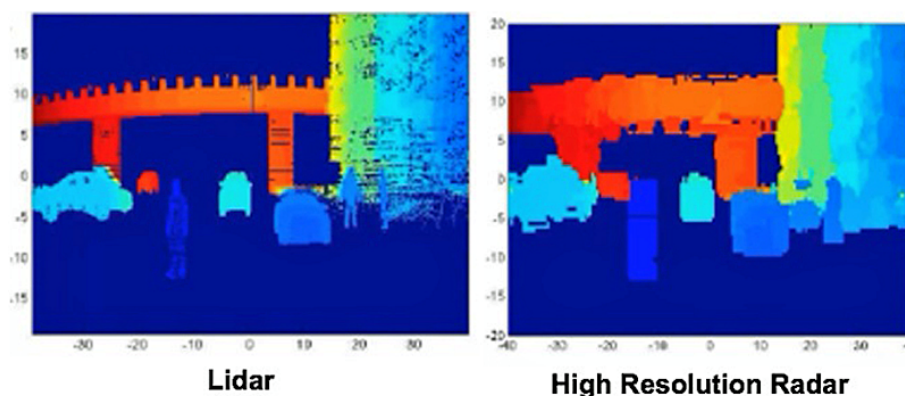


Obrázek 3.5. Princip fungování lidarů [16]

Emitovaným signálem může být zvukový signál – v tom případě hovoříme o sonaru (z anglického Sound Navigation And Ranging), který vysílá zvukové vlny. Jeho principu využil již Leonardo da Vinci, když roku 1490 poslouchal vzdálené lodě pomocí dlouhé trubky [17]. Technologie sonaru jako takového byla rozvinuta během první světové války Brity a Američany pro lokalizaci nepřátelských ponorek. V současnosti se sonaru nejvíce užívá právě ve vodním prostředí – pomocí něj lze například mapovat dno či hledat mořské vraky nebo lokalizovat ryby. Ve zvířecí říši sonar k echolokaci využívají kosatky, delfíni a netopýři [18]. Alternativně emitovaný signál pochází z elektromagnetického spektra – ten využívají radar a lidar.

Radar (z anglického Radio Detection and Ranging) vysílá radiové vlny o délce několika mm až desítek metrů, díky tomu lze detekovat objekty i na desítky kilometrů. Jestliže vývoji sonaru patřila první světová válka, vývoji radaru pak patřila druhá světová válka, neboť právě pomocí radaru Británie hlídala vzduchový prostor nad Lamanšský průlivem. V současnosti se radaru využívá v letecké a lodní dopravě, dále v meteorologii, pozorování vesmírných těles ale také v automatizačním průmyslu. Největší radioteleskopy pro pozorování vesmírných těles dosahují velikosti až 500 metrů. Dále se v těchto letech začal radar prosazovat i v automotive průmyslu, jelikož lze dobře využít například pro funkci adaptivního tempomatu.

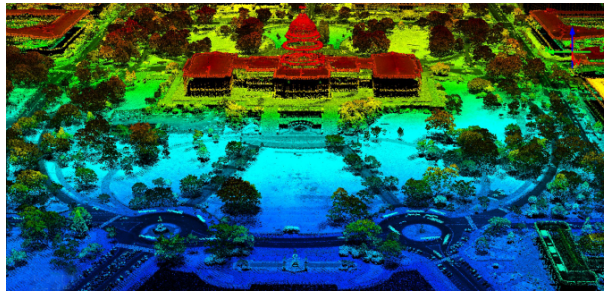
S vývojem laserů se pak začalo uvažovat o jejich využití právě v lokalizaci za využití doby letu a tím započal vývoj lidarů (z anglického Light Detection And Ranging). Lidar emituje pulsy laserové paprsku (typicky infračerveného), díky čemuž je jeho výsledné rozlišení lepší než v případě radaru, jak lze vidět na obr. 3.6. S využitou kratší vlnovou délkou se však pojí také kratší měřitelný radius v porovnání s radarem. Velkou výhodou senzorů emitujících elektromagnetické záření v porovnání se sonarem je jejich rychlost, neboť se světlo šíří rychleji než zvuk.



Obrázek 3.6. Rozlišení detailů objektů v podání lidarů a radarů [19]

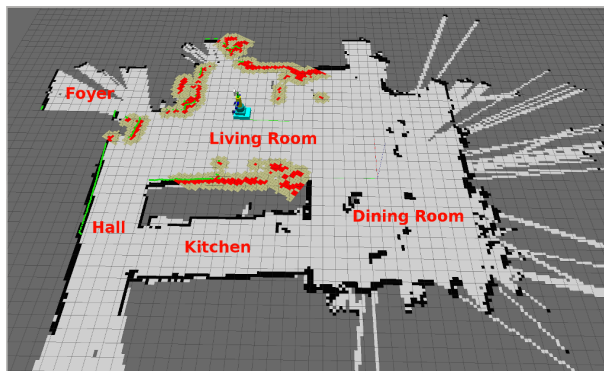
Jelikož se laserový impuls šíří po přímce, lze zjišťovat vzdálenost konkrétního bodu. Pokud je lidar vybaven rozmítací mechanikou (již ne nutně pohyblivou), lze tak vytvořit „bodové mračno“ okolí senzoru v celém jeho rozsahu horizontálně a částečně i vertikálně.

Toho se v současnosti hojně využívá v asistenčních systémech automobilů, ale také například při tvorbě map terénních povrchů. Na obr. 3.7 lze vidět budovu Kapitolu Spojených států amerických a jeho okolí vyobrazeného pomocí mračka bodů.



Obrázek 3.7. Kapitol Spojených států amerických vizualizovaný jako mračko bodů [20]

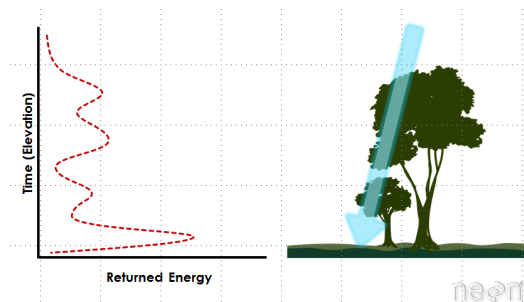
Specifickou aplikací mapování okolí pomocí mračka bodů využívanou v robotice je tzv. SLAM (z anglického Simultaneous Localisation and Mapping), kdy lze jednak tedy určit přesnou polohu robota v prostoru ale průběžně jsou získaná data ukládána a je z nich tvořena virtuální kopie okolí. Ukázka vzniklé mapy předem neznámého prostoru za využití The Robot Operating System (ROS) lze vidět na obr. 3.8



Obrázek 3.8. Mapa prostoru vzniklá metodou SLAM [21]

Pravděpodobně největší nevýhodou lidarů je jeho cena. Jelikož se jedná o složitější zařízení než radar – a to jak mechanicky, tak elektronicky –, tak například 3D lidar od firmy Waymo zajišťující možnost autonomní jízdy na úrovni 4 až 5 stojí přibližně \$ 5000. [22]

Společnou vlastností senzorů založených na principu doby letu je případ, kdy jeden emitovaný puls doletí na snímač několikrát neboť se na více překážkách vytvoří jeho echa, jak lze vidět na obr. 3.9. Tím v datech vzniká šum, který lze částečně potlačit modulací emitovaného signálu, častěji se však odstraňuje v rámci post-processingu i za využití umělé inteligence.



Obrázek 3.9. Vznik několika ech pro jeden emitovaný signál [23]

■ 3.1.3 Magnetometr

Magnetometr měří magnetické pole pomocí Hallova jevu. Pokud na vodivou desku, kterou prochází proud, působí externí magnetické pole, dochází k deformaci drah elektronů a tím vzniká měřitelná změna napětí závislá na intenzitě tohoto pole a jeho orientaci. Při používání magnetometrů (digitálních kompasů) je zapotřebí dát pozor na možná magnetická rušení a to jednak v okolí, ale v robotice především od použitých elektromotorů. Další typickou vlastností magnetometrů je jejich nutná kalibrace, která potlačí vlivy hard/soft iron efektů. Této kalibraci se věnuje např. Ozyagcilar [24].

■ 3.2 Senzory pro měření relativní polohy

Zatímco senzory pro určení absolutní polohy vyjadřují pozici přijímače v globálním souřadném systému, senzory relativní polohy pozici odvozují na základě pohybu vlastního přijímače vůči své předchozí poloze. Pokud se dostatečně přesně určí počáteční poloha, lze takové senzory úspěšně používat bez externích referenčních bodů. Nevýhodou však je postupný drift vůči skutečné poloze způsobený diskretní integrací, který bez dalších senzorů lze pouze minimalizovat, nikoliv však úplně odstranit.

■ 3.2.1 Akcelerometr

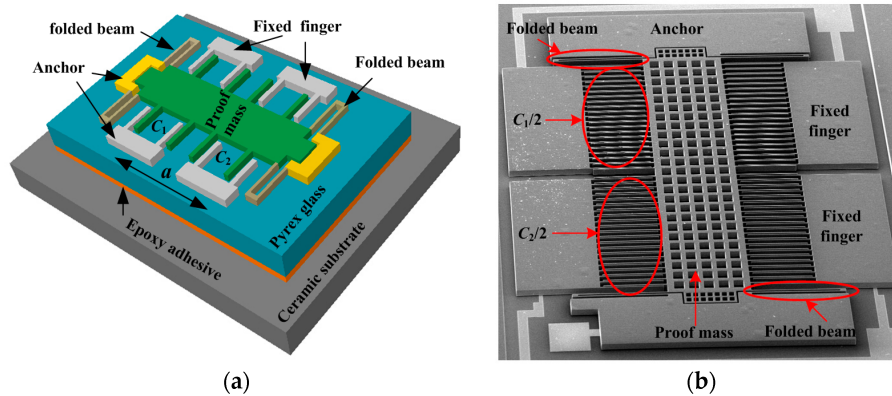
„Snímače zrychlení jsou založené na měření účinku setrvačné síly na objekt, podle Newtonova 2. zákona

$$F = -m \cdot a \quad (3.1)$$

Účinky zrychlení je možné měřit mnoha způsoby, např. měřit přímo sílu nebo napětí tenzometry, případně měřit posunutí snímači polohy. Různé principy jsou vhodné pro různé rozsahy frekvencí,...” [25]

Využívanými principy jsou piezoelektrický jev, kde krystal generuje napětí přímo úměrné své deformaci, optický jev používající optická vlákna, elektrostatický a kapacitní.

Kapacitního principu se dnes využívá v MEMS senzorech (z anglického Micro Electro Mechanical Systems). MEMS je spojení integrovaných obvodů vyrobených klasickými metodami typu CMOS a mikromechanických struktur vyrobených mikroobráběním (typicky selektivním leptáním nebo fotolitografií) a jejich integrací v jednom mikročipu. Velkou výhodou MEMS senzorů je jejich kompaktnost (nyní se klasicky dodávají v SMD pouzdrech) a díky velkoobjemové výrobě také nízká cena (pod 1 \$ pro akcelerometry používané v mobilních telefonech). Kapacitní princip spočívá v měření rozdílné kapacity dvou kondenzátorů. Tento rozdíl vznikne posuvem seismické hmoty vůči kondenzátorům pevně přichyceným k rámu. Diagram takového senzoru lze vidět na obr. 3.10 vlevo, vpravo pak samotnou MEMS strukturu.



Obrázek 3.10. Struktura kapacitního akcelerometru (a) diagram akcelerometru (b) MEMS struktura [26]

Při použití akcelerometrů v praxi je potřeba se vypořádat s jejich notným zašuměním. K tomu se typicky využívají low pass filtry, případně jsou data z akcelerometru zpracována jedním z několika sofistikovanějších algoritmů (např. Kalmanův filtr).

3.2.2 Gyroskop

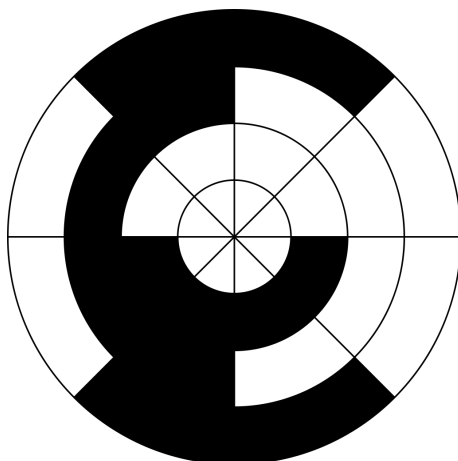
Obdobně jako akcelerometr, také gyroskop využívá několika principů. Rotující gyroskopy zachovávají moment setrvačnosti rotující hmoty, optické gyroskopy využívají Sagnacův jev, kdy vlivem rotace dochází k fázovému posuvu dvou laserových svazků jejichž interferenci lze následně změřit. Toto řešení je vhodné především pro kosmická zařízení, neboť mají minimum pohyblivých částí a jsou tak méně náročná na průběžný servis. MEMS gyroskopy založené opět na kapacitním principu využívají Coriolisova efektu. Necht se hmota pohybuje ve směru jedné dané ose s danou hybností, způsobí-li na ni vnější úhlová rychlost kolem osy kolmé k pohybu hmoty, tato rotace způsobí vychýlení hmoty směrem kolmým k těmto osám.

Nepříjemnou vlastností gyroskopů je fakt, že poloha jimi určená vzniká integrací dat ze senzoru, postupem času tak dochází k jejímu driftu od skutečné polohy. S tímto je potřeba se v aplikaci vypořádat. Částečně si lze pomoci zvýšením snímkovací frekvence, lepším řešením však je průběžná korekce dalšími senzory.

3.2.3 Rotační enkodéry

Odometrie je jednou z nejpoužívanějších metod pro určení okamžité polohy robota [27]. Ta přes matematický model svazuje průběžné hodnoty otáček kol s pohybem v prostoru. K měření otáček tak lze vhodně využít inkrementální rotační enkodéry, což je optický digitální snímač skládající se z disku s ryskami. Pomocí dvojice LED a fotodiody jsou pak snímány přechody přes tyto rysky, čímž lze určovat aktuální natočení disku. Pro dodatečně zjišťování směru rotace je nutné použít dva páry LED + fotodiody, které jsou vůči sobě posunuté o 1/4 periody rysek. Rozlišení rotačních enkodérů dosahuje až tisíců pulzů na otáčku.

Rotační enkodéry je možné vyrobit také v absolutním provedení, čehož je dosaženo více drahami na rotačním disku, které jednoznačně kódují polohu. Každá z drah vyžaduje vlastní pár LED + fotodiody, jejich velikostí je pak dán počet možných drah. Pro rozložení barevných polí na drahách se používá binární kódování případně Greyův kód, který s rozlišením 8 poloh/otáčka lze vidět na obr. 3.11.



Obrázek 3.11. Disk absolutního rotačního enkodéru s 3 bitovým Grayovým kódem [28]

Pokud jsou využity enkodéry, je nutné při volbě řídicího HW počítat s tím, že obsluha jednoho relativního enkodéru vyžaduje 2 piny s funkcí interrupt. SW obsluha interruptů pak může nepříjemně ovlivňovat kritické části programu.

Kapitola 4

Návrh rámu roveru

Po rešení dosavadních konstrukcí robotů pro soutěž ARLISS jsem se rozhodl vlastního robota udělat jako rover, volím tedy cestu k cíli po zemi. Hlavním důvodem této volby bylo právě dostatečné množství informací k konstrukcím roverů, které již úkol zvládli. Tyto informace pro mě byly velmi cenné, neboť sám s touto soutěží mám nulové zkušenosti. Dalším důvodem, proč se vydat cestou po zemi, byla značně snížená náročnost řídicího algoritmu oproti cestě vzduchem.

Jak již bylo zmíněno v kapitole 2, robot je na začátku úkolu vyneseno nosnou raketou. Aby všichni soutěžící měli stejné podmínky, tak je předem daná velikost nákladového prostoru, která je pro všechny stejná. Hlavním omezením pro samotný rám tak jsou parametry toho válcového prostoru. Jejich hodnoty jsou uvedeny v tabulce 4.1.

Parametr	Přípustná hodnota
Průměr	146 mm
Výška	254 mm
Hmotnost	1,8 kg

Tabulka 4.1. Parametry nákladového prostoru

Pro prvotní verzi jsem chtěl využít celého průměru nákladového prostoru, navrhl jsem proto dvě velká kola zabírající téměř celý průměr. Kola jsou poměrně tenká a každé má se zemí pouze jeden bod kontaktu – o-kroužek, který je na kole natažen. Toto řešení pro prototyp postačuje, neboť bylo plánováno testovací jízdy provádět pouze na hladkých površích. Výhodou takového řešení navíc je snadná tvorba matematického modelu popisujícího pohyb roveru.

Poté přišlo na řadu rozmístění komponent – jmenovitě akumulátoru a elektroniky. Tím, jak jsou použité stejnosměrné motory umístěny, zabírají válcový prostor kolem osy prostoru. Jako jediné možné řešení se ukázalo rozmístění zbytku komponent kolem tohoto pomyslného válce. Usazení komponent jsem vyřešil podobně jako jiné týmy, kdy se rám skládá z jedné hlavní desky. Já jsem však použil desky dvě, jednu umístěnou horizontálně, na které bude přichycen akumulátor, a další vertikálně, na níž bude přišroubovaná deska plošných spojů. V jednom z prvních návrhů byly motory přichyceny pomocí plechů, které se ke mnou použitým motorům dodávají. Avšak aby byl rám roveru pevnější, ke dvěma hlavním deskám jsem přidal dvě bočnice do nichž se motory přišroubovují. Přichycení kol k hřídelím motorů je vyřešeno montážními náboji, které se k těmto motorům prodávají jak volitelné příslušenství. Náboje v sobě mají otvory se závity, jichž jsem využil pro přišroubování samotných kol.

Aby byl robot stabilní, navrhl jsem ostruhu na čepu s ostruhovým kolečkem. Výsledný návrh lze vidět na obr. 4.1

Kapitola 5

Návrh řídicí elektroniky

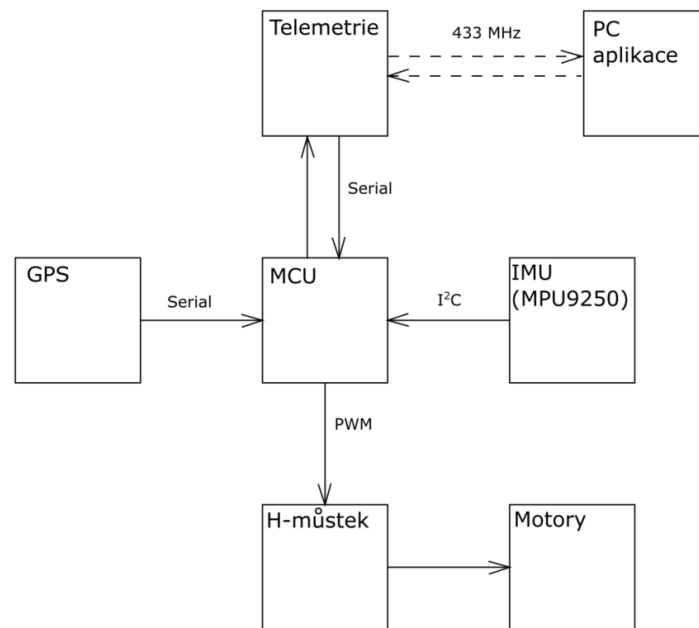
Následující text je převzat z mého příspěvku na Studentské tvůrčí činnosti 2019 [29], který vznikl jako dílčí práce na mé bakalářské práci.

Dalším důležitým bodem návrhu a realizace prototypu bylo připravit řídicí elektroniku pro plně autonomní řízení. Její blokové schéma lze vidět na obr. 5.1. Pro ovládání celého robota byl zvolen mikrokontrolér Teensy ve verzi 3.2 postavený na procesoru ARM, který svým výkonem násobně převyšuje velmi oblíbené Arduino. Zároveň si však Teensy z Arduina přebírá jeho nejsilnější zbraň, a sice možnost tvorby řídicího softwaru v Arduino IDE, kde je možné využívat již vytvořené knihovny jinými uživateli pro ovládání periférií.

Mezi tyto periferie patří prvně H-můstek TB6612FNG zvolený především pro stálý pracovní proud až 1,2 A na každý ze dvou kanálů, což je pro použité motory dostatečné. H-můstek spolu s DC motory Pololu 1104 zajišťuje kontrolovaný pohyb robota. DC motory byly vybrány především pro skvělý poměr poskytovaného momentu a maximálních otáček na zastavěný prostor. Mezi kontakty a kostru každého motoru byly dle doporučení výrobce [30] napájeny odrušovací kondenzátory.

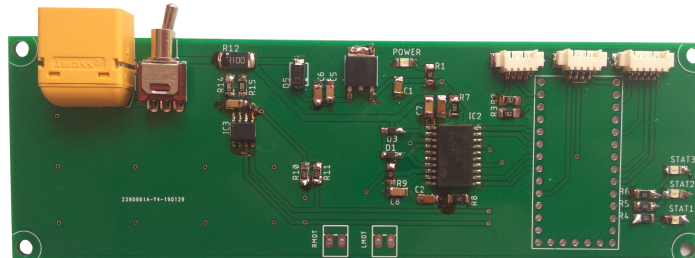
Ze senzorů byl pro dobře zpracovanou dokumentaci a již vytvořenou SW knihovnou vybrán senzor pro měření polohy MPU9250, který obsahuje akcelerometr s rozsahem až ± 16 g, gyroskop s rozsahem až ± 2000 °/s a magnetometr s měřicím rozsahem ± 4800 μ T. Ten je doplněný o GPS modul NEO-6M, jenž je díky své maximální přesnosti pozice 2,5 m plně dostačující pro splnění zadání.

Pro možnost komunikace s robotem ze vzdáleného stanoviště byl přidán Telemetry modul pracující na frekvenci 433 MHz, a tedy zajišťující velkou komunikační vzdálenost (až 1,6 km). Ten však bude muset být pro samotnou soutěž v USA vyměněn z legislativních důvodů. Na samotnou telemetrii se však nelze spolehnout ve všech případech a ani není úmyslem přes ni přenášet naprosto všechna aktuální data z robota, proto se do finální verze plánuje zakomponovat SD karta, která bude sloužit jako černá skříňka. Telemetrie bude využita především pro možnost nastavování parametrů před samotným startem bez nutnosti přehrávat celý firmware, lokalizaci roveru při jeho ztrátě, případně pro jiné úlohy řízení mimo soutěž ARLISS.



Obrázek 5.1. Blokové schéma řídicí elektroniky

Pro kompaktnost potřebné elektroniky byla navržena deska plošných spojů s využitím softwaru Autodesk Eagle. Deska prošla několika iteracemi – zprvu kvůli problémům se stabilitou napájení, další změny zohledňovaly vývojové požadavky – aktuální verze (na obr. 5.2) je již tvořena primárně SMD součástkami pro maximální úsporu místa na desce. Byl doplněn senzor proudu pro průběžné měření stavu akumulátoru a v budoucnu také energetickou optimalizaci celého robota. Kompletní výkresovou dokumentaci lze nalézt v příloze C.



Obrázek 5.2. Aktuální verze desky plošných spojů

Robot je po celou dobu plnění úkolu napájen z akumulátoru, u kterého je stěžejní návrh jeho kapacity. Ta musí být dostatečná pro zvládnutí vzdálenosti k cíli, zároveň by však akumulátor měl být co nejmenší, protože svou velikostí se významně projevuje v návrhu rámu robota a navíc se jedná o nejtěžší komponentu. Z tohoto důvodu byla zvolena technologie LiPo, právě pro svou energetickou hustotu.

Při stanovení požadované kapacity se vycházelo z katalogových údajů použitých motorů, konkrétně z údaje o maximálních otáčkách, kdy Pololu 1104 dosahuje 200 rpm. Rovnicí (5.1) jsme schopni z tohoto údaje a průměru kola d vypočítat maximální teoretickou rychlost v_{max} .

$$v_{max} = f_{max} \pi d = 200 \cdot \pi \cdot 0,104 = 1,09 \text{ m/s} \quad (5.1)$$

Kapitola 6

Řídicí algoritmus pro autonomní jízdu

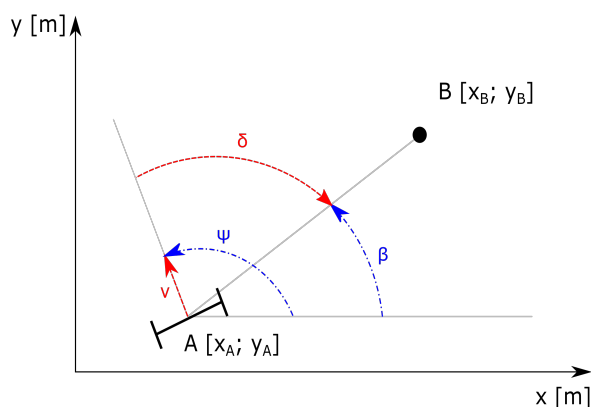
Po návržení kostry robota a výběru kompletní elektroniky přišlo na řadu robota „oživit“ a za využití osazených senzorů navrhnout a implementovat řídicí algoritmus pro plně autonomní jízdu.

6.1 Určení regulační odchylky na základě rozdílu azimutů

Při návrhu algoritmu jsem vycházel ze dvou faktů. Za prvé: soutěžním zadáním je dopravit robota na zadané souřadnice, je tedy nezbytné průběžně určovat jeho zeměpisnou polohu. Druhým faktem je pak nutnost zjištění aktuálního směru jízdy (azimutu) robota, na základě něhož bude probíhat úprava trajektorie. Řízení lze teoreticky obstarat na základě těchto dvou veličin, neboť poušť, kde se soutěž koná, je planina a neuvažují tedy s potřebou aktivně zjišťovat překážky.

Jak aktuální polohu, tak azimut robota by bylo možné zjišťovat pouze z GPS dat, nicméně s ohledem na přesnost zvoleného senzoru by takovýto řídicí program měl dlouhý reakční čas. Proto jsem se rozhodl prostorovou orientaci robota určovat z dat z inerciální jednotky.

Celý řídicí algoritmus vychází ze mnou zvoleného souřadného systému, který lze vidět na obr. 6.1. „Jedná se o souřadný systém typu „Lokální sever-východ-nahoru“, jenž je variací na NED z [32], kdy je v blízkém okolí robota uvažována tangenciální rovina s povrchem země a její osy x , y korespondují se zeměpisnými směry východ, respektive sever. Jednotkami os jsou metry vzniklé linearizací vzdáleností ve stupních. Bod A označuje aktuální polohu robota určenou souřadnicemi x_A a y_A . Analogicky jsou pak značeny cílové souřadnice v bodě B x_B a y_B .“ [29]



Obrázek 6.1. Zvolený souřadnicový systém

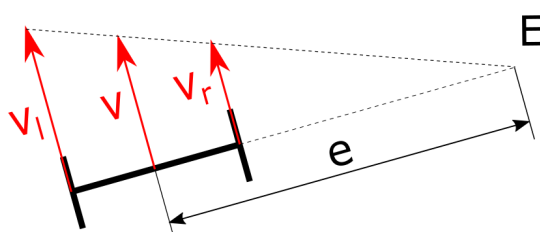
Úhlem ψ je označován azimut robot. Jedná se o jeden ze tří polohových úhlů – konkrétně o tzv. yaw. Úhel β označuje požadovaný azimut k cílovým souřadnicím od aktuální polohy robota a je vyjádřen jednoduchým goniometrickým vztahem

$$\beta(t) = \arctan\left(\frac{y_B - y_A(t)}{x_B - x_A(t)}\right). \quad (6.1)$$

Úhel δ označuje úhlovou odchylku, jenž je vstupující veličinou do navrženého regulátoru. Regulační odchylka je dána rovnicí

$$\delta(t) = \psi(t) - \beta(t) \quad (6.2)$$

„Tento úhel je následně využit pro diferenční řízení, které je znázorněno na obr. 6.2. Rychlosti jednotlivých kol jsou dány rovnicemi (6.3), kde v je rychlost robota předem určená konstantou a K_p je konstanta proporcionálního zesílení regulátoru. Z předpisu těchto rovnic je vidět, že se jedná o P regulátor, který na řízení tohoto systému postačuje, neboť tato soustava je integrační.



Obrázek 6.2. Diferenční řízení

$$\begin{aligned} v_l(t) &= v(t) + K_p \cdot \delta(t) \\ v_r(t) &= v(t) - K_p \cdot \delta(t) \end{aligned} \quad (6.3)$$

Rozdílem v rychlostech jednotlivých kol je dán okamžitý střed otáčení E ve vzdálenosti e od středu robota, což umožňuje matematické vyjádření pohybu robota. Poloměr zatačení e může nabývat hodnot od 0 m, kdy se robot otáčí na místě kolem středu pomyslné spojnice kol, do $\pm\infty$, což znamená přímočarý pohyb vpřed/zpět. Takto zavedený systém řízení je vhodný pro pozdější vývoj, kdy lze mimo jiné ze zvoleného poloměru zatačení zpětně dopočítávat požadované rychlosti kol.” [29]

6.2 Stanovení aktuálního azimutu pomocí Madgwickova filtru

Jak již bylo zmíněno, aktuální azimut by bylo možné určit čistě pomocí GPS senzoru, avšak já jsem se rozhodl pro orientaci robota v prostoru použít data z inerciální jednotky.

MPU9250 je poměrně oblíbený senzor, lze k němu tedy snadno najít kvalitně napsanou knihovnu [33], jenž prací se samotným senzorem notně zjednodušuje. Při instalaci v Arduino IDE tak stačilo pouze nahrát .ZIP soubor této knihovny. Co se týče implementace do algoritmu, tak nejprve bylo samozřejmě nutné knihovnu importovat, inicializovat interrupt pin a pomocí příkazu `Wire.begin()` povolit I²C komunikaci. Posléze již stačí volat jednotlivé funkce obsažené v knihovně. Pomocí této knihovny jsem tak v hlavním programu: resetoval IMU, nastavoval měřící rozsahy jednotlivých senzorů, vyčítal kalibrační hodnoty určené z výroby, vyčítal rozlišení jednotlivých senzorů a konečně vyčítal aktuální hodnoty senzorů.

MPU9250 sice má uložené kalibrační hodnoty z výroby, pro jistotu jsem se rozhodl provést vlastní kalibraci. V případě akcelerometru a gyroskopu se jednalo pouze o odečtení offsetů, které senzory ukazovaly když robot stál na místě. U magnetometru bylo nutné provést transformaci dat z obecného pohybu senzoru na povrch normalizované koule se středem v počátku.

Takto zkalibrovaná data už mohla být použita jako vstup pro určování prostorových úhlů robota. Za tímto účelem jsem zvolil Madgwickův filtr především pro svou výpočetní jednoduchost, ale také proto, že je potřeba ladit pouze dva parametry. I zde jsem použil již hotovou knihovnu [34], aby byl výsledný kód co nejpřehlednější a také tím odpadlo riziko možnosti mé chyby kdybych se snažil kód celého filtru napsat sám. V hlavním programu pak funkce z této knihovny využívám k: nastavení parametrů filtru, nahrání aktuálních hodnot ze senzorů do filtru a volání aktuálních prostorových orientačních úhlů.

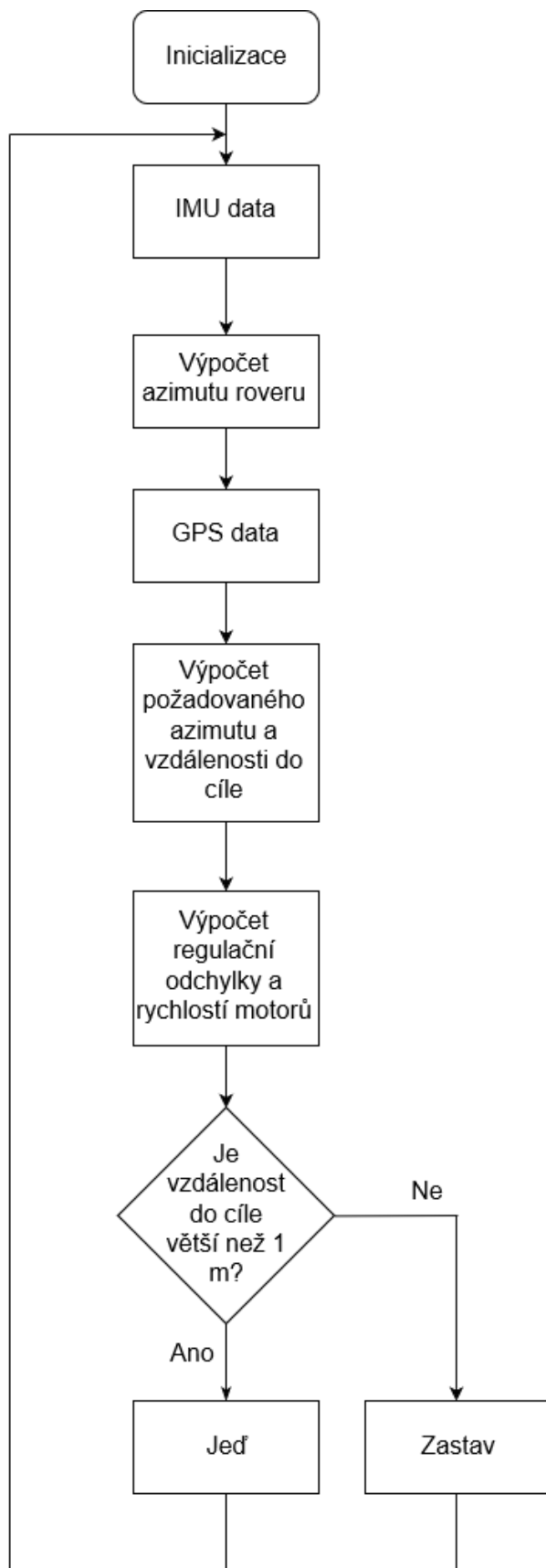
Jak bylo zmíněno v kapitole 6.1, pro samotnou navigaci robota postačuje pouze údaj o jeho azimutu, tedy o prostorový úhel yaw. Při prvním testování se však ukázalo, že úhel yaw, který vrací filtr z dat ze senzoru, nekoresponduje se světovými stranami. Bylo tedy nutné provést jeho přepočítání. K tomu jsem využil interní funkce Arduina `map()`.

6.3 Kompletní řídicí algoritmus pro autonomní jízdu

Z těchto dílčích úkonů popsaných v kapitolách 6.1 a 6.2 již bylo možné napsat kompletní řídicí algoritmus. Jeho vývojový diagram lze vidět na obr. 6.3.

Mikrokontrolér nejdříve získá aktuální data z inerciální jednotky a pomocí Madgwickova filtru určí aktuální azimut roveru. V následujícím kroku zjistí pomocí GPS senzoru zeměpisnou polohu. Z aktuální polohy se dále vypočte požadovaný azimut dle rovnice (6.1) a také vzdálenost do cílového bodu. Poté algoritmus pomocí regulační odchylky dle rovnice (6.2) vypočte požadované rychlosti motorů (6.3). Pokud je vypočtená vzdálenost od cílových souřadnic větší než stanovená hodnota (při experimentech to byl 1 metr), tak mikrokontrolér na motory posílá požadované rychlosti, jestliže ne, tak robot plynule zastaví.

Kompletní přepis řídicího programu lze nalézt v příloze D.



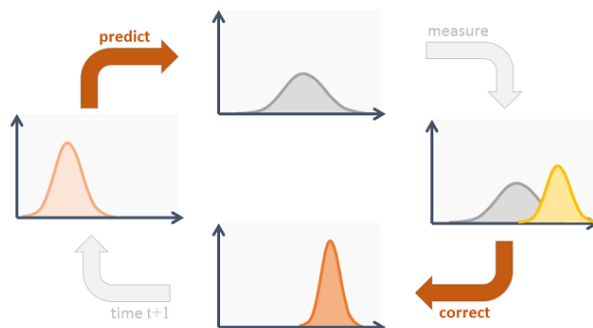
Obrázek 6.3. Vývojový diagram řídicího algoritmu

Kapitola 7

Určování polohy pomocí Kalmanova filtru

Maximální přesnost lokalizace pomocí GPS senzoru NEO-6M v otevřeném prostoru je 2,5 m, což by pro splnění zadání kategorie OpenClass plně stačilo. Nicméně s přihlédnutím k tomu, že se v budoucnu plánuje využít pokročilejší řídicí algoritmy, bylo vhodné tuto přesnost řádově zlepšit. Hardwarové varianty řešení s diferenciální GPS či lidarem byly zavrhnuty, neboť by jejich implementace byla velice náročná – především prvotní kalibrace. Dalším důvodem zavržení byl záměr vyhnout se dalším podpůrným technologiím. Jestliže soutěž má simulovat orbitální misi Marsu, je více než vhodné šetřit každý gram váhy. Rozhodl jsem se tedy pro řešení softwarové, konkrétně o využití Kalmanova filtru, jenž umožňuje kombinovat vícero zdrojů informací o pohybu robota. Ať se již jedná o matematický model nebo dříve uvedené senzory. Kalmanův filtr je ideálním řešením pro úlohy s poměrně omezeným operačním výkonem, což potvrzuje i fakt, že jednou z jeho prvních reálných aplikací byl lunární modul projektu Apollo.

Jedná se o diskrétní filtr pracující v cyklech. V každém cyklu pak proběhnou 3 kroky – pro názornost jsou vyobrazeny na obr. 7.1 – predikce výstupní veličiny pomocí teoretického modelu, měření přímé či nepřímé a nakonec korekce těchto dvou množin. Množin proto, že filtr pracuje statisticky, přičemž zohledňuje normální rozdělení veličin.



Obrázek 7.1. Schéma jednoho cyklu Kalmanova filtru [35]

Tento filtr se dá použít na velmi široké spektrum úloh. V zásadě stačí, aby bylo možné sestavit základní rovnice, se kterými filtr pracuje – čemuž se bude věnovat tato část textu.

Jak bylo uvedeno dříve, Kalmanův filtr pracuje několika kroci. Prvnímu kroku tzv. „predikci“ odpovídají následující rovnice, kde \hat{x}_k zastupuje vektor stavových proměnných po predikci v kroku k , F_k matice „vnitřních vlivů“ stavových proměnných, \hat{x}_{k-1} vektor stavových proměnných po aktualizaci v kroku $k - 1$, B_k matice vlivů externích proměnných a \vec{u}_k vektor externích proměnných. P_k je kovarianční matice procesu po predikci v kroku k , P_{k-1} kovarianční matice procesu po aktualizaci v kroku $k - 1$ a Q_k je matice „šumu procesu“

$$\begin{aligned}\hat{x}_k &= F_k \hat{x}_{k-1} + B_k \vec{u}_k \\ P_k &= F_k P_{k-1} F_k^T + Q_k.\end{aligned}\tag{7.1}$$

Kapitola 8

Experimentální identifikace DC motoru

Jedním ze vstupních matematických modelů Kalmanova filtru byl zvolen pohyb robota podle aktuální úhlové rychlosti motorů. Vzhledem k tomu, že mnou použité motory nemají zabudované enkodéry pro přímé odměřování úhlu hřídele, rozhodl jsem se tuto rychlost odvozovat z aktuální střidy PWM signálu, kterým jsou jednotlivé motory řízeny. Bylo tedy nutné provést experiment pro stanovení přenosové funkce popisující dynamiku tohoto motoru, z níž byla vytvořena jeho frekvenční charakteristika.

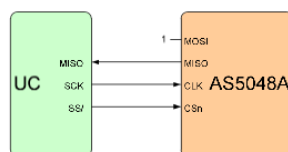
8.1 Návrh experimentu pro stanovení frekvenční charakteristiky motoru

Samotný experiment lze rozdělit na dvě části – část ovládací a část měřicí. Ovládací část má zajistit vstupní signál požadovaných parametrů pro řízení rychlosti motorů, měřicí část poté musí být schopna s dostatečnou vzorkovací frekvencí získávat aktuální hodnoty otáček motorů a ty následně uložit pro další zpracování. Po uvážení dostupných zařízení byla zvolena tato kombinace: pro spínání motorů sloužil h-můstek TB6612FNG, generování vstupního signálu obstarávalo myRIO od firmy National Instruments, jenž také přes sběrnici SPI získávalo údaje o natočení kola z již ověřeného senzoru AS5048A.

Pro co nejkompletnější frekvenční charakteristiku bylo nutné vybrat takový řídicí člen, který by dokázal generovat vstupní signál v rozmezí 0,01 až 50 Hz a zároveň obsluhovat celou měřicí smyčku na frekvenci dosahující až 1 kHz a to bez náhodných zpoždění pro zachování konstantního časového rozestupu mezi jednotlivými vzorky měření.

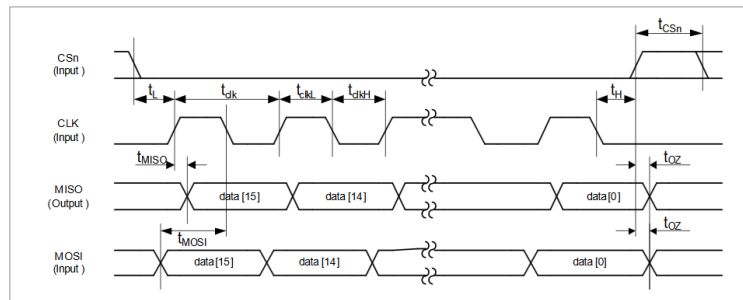
V úvahu připadaly 3 varianty: využít běžně dostupný mikrokontrolér typu Arduino/Teensy, spojit možnosti programu Simulink s měřicími kartami Humusoft, nebo zařízení myRIO od firmy National Instruments. Přičemž jsem zvolil třetí variantu, myRIO totiž plně splňuje mé prvotní nároky a nebylo zapotřebí řešit dostupnost licencí jako v případě Simulinku.

Pro snímání aktuálního natočení kola jsem použil senzor AS5048. Jedná se o magnetický rotační enkodér mezi jehož výhody patří 14-bitový převodník, standardní SPI či I²C rozhraní a tolerantnost ke změnám vzduchové mezery mezi samotným senzorem a magnetem během měření. Senzor byl k myRIO připojen přes sběrnici SPI ve třívodičovém zapojení viz obr. 8.1, protože pracoval v režimu „Single Slave, Read only“.



Obrázek 8.1. Třívodičové zapojení AS5048 [37]

myRIO bohužel nemá samostatný pin s funkcí „slave select“. Jento signál (CSn na obr. 8.2) jsem tedy vyřešil programově na mnou zvoleném pinu digitálního výstupu. Samotná implementace je zmíněna v kapitole 8.2.

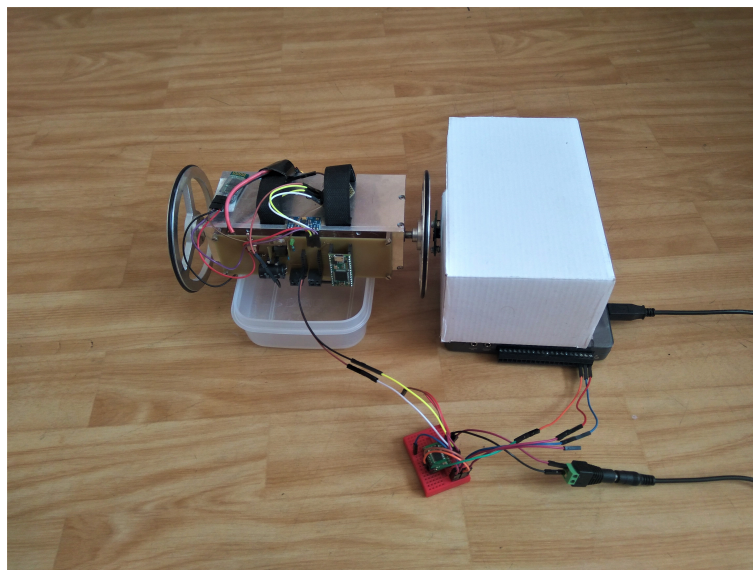


Obrázek 8.2. Časový průběh signálů na sběrnici SPI [37]

Po dobu experimentu zůstalo k hřídeli motoru připevněné hliníkové kolo, aby byl zachován stejný moment setrvačnosti působící na motor jako při skutečném pohybu robota.

Magnetické pole snímané senzorem bylo tvořeno magnetem připevněným na střed vnější plochy kola. Na základě datasheetu senzoru jsem použil neodymový magnet.

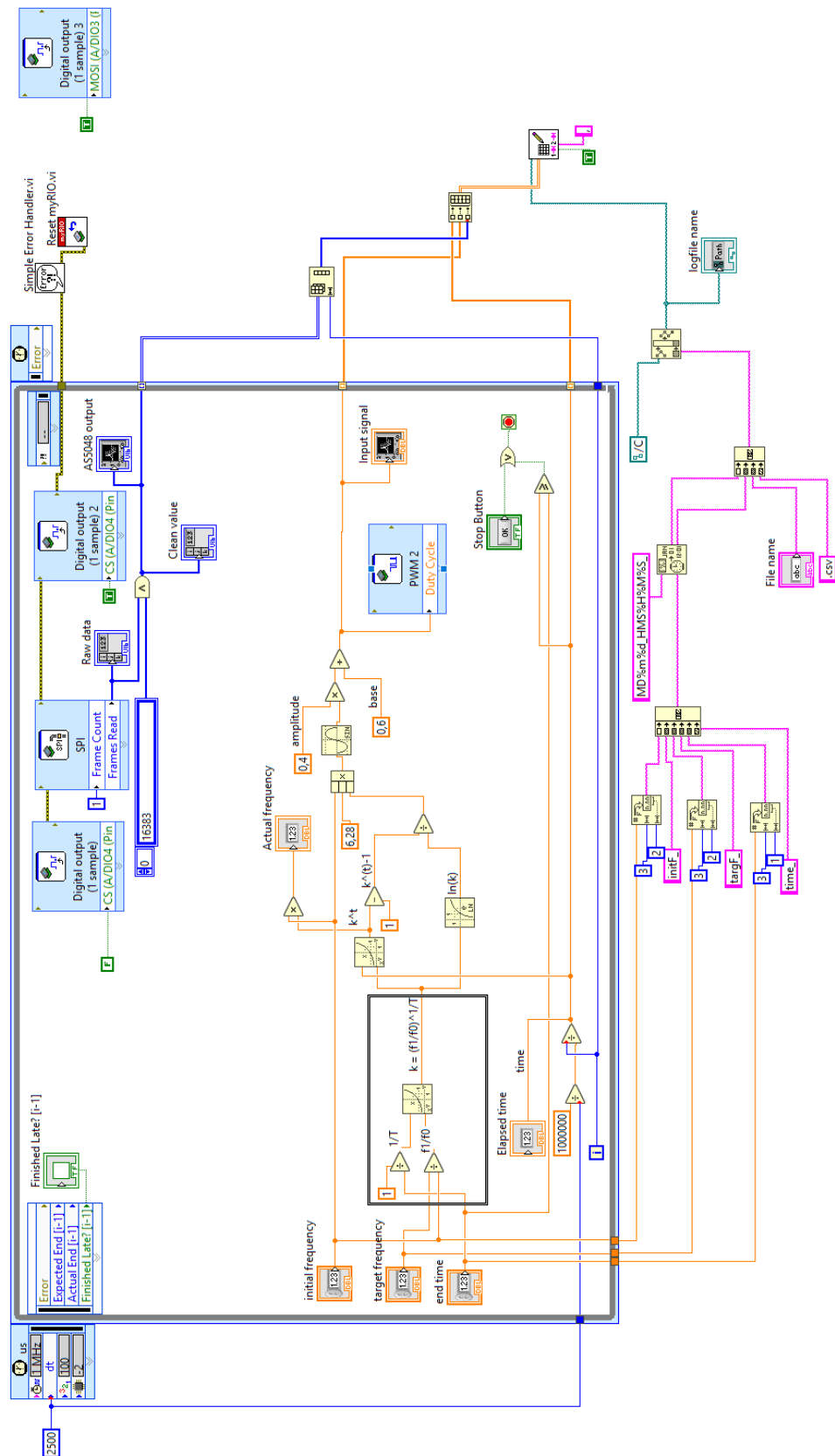
Kompletně sestavenou měřicí aparaturu lze vidět na obr. 8.3.



Obrázek 8.3. Zapojení měřicí aparatury

8.2 Program pro řízení experimentu

Po návrh kompletní aparatury jsem se přesunul k tvorbě řídicího programu. Využití myRIO má v tomto další výhodu, neboť se programuje graficky v programovacím jazyce LabVIEW a i s mými nulovými předchozími znalostmi této celé platformy jsem dokázal finální program vytvořit v řádu dní. Výsledný programu pro řízení experimentu lze vidět na obr. 8.4.



Obrázek 8.4. LabVIEW program pro řízení experimentu

Samotný program pracuje jednak v řídicí/měřicí smyčce, ale je také složen z jednorázových událostí (na obr. 8.4 jsou vyvedeny mimo hlavní smyčku). Mezi tyto jednorázové události patří nastavení logické hodnoty „1“ na pin A/DI03 simulující signál MOSI z obr. 8.1, a po ukončení hlavní smyčky dochází k uložení vstupního signálu motoru a výstupního signálu ze senzoru AS5048. V samotné hlavní smyčce se vykonávají dvě činnosti: generování aktuální hodnoty vstupního signálu (na obr. 8.4 oranžovou barvou) a obsluha sběrnice SPI pro odečtení aktuální hodnoty natočení hřídele motoru (modrou barvou). Smyčka bez problému zvládá pracovat na frekvenci 4 kHz, což zajišťuje dostatek dat pro následnou identifikaci motoru.

Programovací jazyk LabVIEW má již před-připravené bloky pro jednotlivé funkce, což pomáhá k jednoduché tvorbě programů, jelikož uživatel musí „pouze“ nastavit požadované parametry. Jedním z mnoha využitých bloků byl blok „PWM“, u kterého se nastavují 2 parametry – frekvence a střída. Frekvenci PWM jsem pevně nastavil na hodnotu 500 Hz, neboť na této frekvenci pracuje použitý mikrokontrolér. Střída signálu je nastavována dynamicky do požadovaného tvaru vstupního signálu.

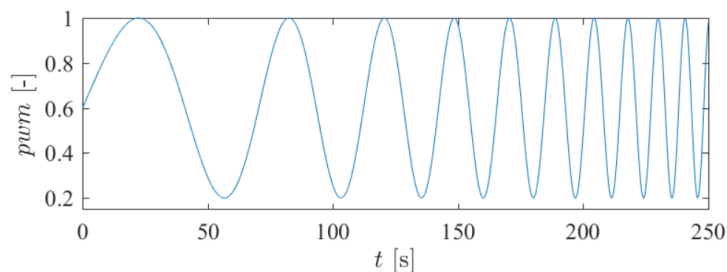
Vstupním signálem pro tvorbu frekvenční charakteristiky je zpravidla funkce sinus. Pokud by však každé jedno měření mělo mít konstantní (avšak rozdílnou) frekvenci budícího signálu, bylo by pro proměření rozsahu 0,01 až 50 Hz zapotřebí velké množství jednotlivých měření. Rozhodl jsem se touto cestou nevydat a za cenu určité nepřesnosti výsledné přenosové funkce vůči naměřeným datům byl jako vstupní signál zvolen „exponenciální chirp“. Předpis této funkce je

$$g(t) = \sin \left[2\pi f_0 \left(\frac{h^{(t)} - 1}{\ln(h)} \right) \right]. \quad (8.1)$$

Kde h je „míra exponenciálního růstu“ vyjádřená dle (8.2). f_0 označuje počáteční frekvenci a f_1 koncovou frekvenci dosaženou v čase T .

$$h = \left(\frac{f_1}{f_0} \right)^{\frac{1}{T}} \quad (8.2)$$

Amplitudu a střední hodnotu signálu je možné měnit do požadovaných hodnot. Pro tento experiment jsem zvolil střední hodnotu 0,6 a amplitudu 0,4. Čímž střída vstupního PWM signálu pro řízení rychlosti motoru nabývala hodnot v rozmezí 0,2 až 1, neboť se předpokládá, že právě v tomto rozmezí budou motory pracovat. Navíc pokud by střída klesala pod úroveň 0,2, docházelo by během experimentu k zastavování motoru, což by pouze zhoršilo přesnost identifikované přenosové funkce vůči realitě. Časový průběh budícího signálu použitého pro experiment je zobrazen na obr. 8.5.



Obrázek 8.5. Průběh budícího signálu „chirp“

Pro obsluhu AS5048 jsem využil bloky „Digital output“ a „SPI“. Pomocí bloků „Digital output“ simulují signály „MOSI“ a „CSn“ z obr. 8.2 na vstupy senzoru, blok „SPI“

pak slouží k samotnému čtení dat. U tohoto bloku bylo nutné nastavit jak samotné HW zapojení tak několik parametrů podle dokumentace k senzoru – viz obr. 8.6.

Node name:

Channel:

Connections: MISO: Pin 23, MOSI: Pin 25, CLK: Pin 21

Mode: Write
 Read
 Write/Read

Frequency: MHz
 Detailed information about valid frequencies for this Express VI.
[Learn more](#)

Frame length: bits

Advanced options

Clock phase:

Clock polarity:

Data direction:

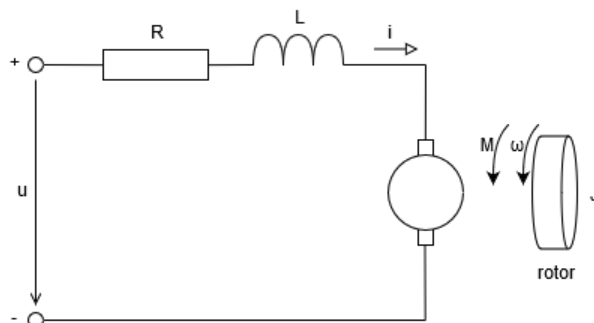
Obrázek 8.6. Nastavené parametry bloku „SPI“

8.3 Stanovení přenosové funkce stejnosměrného motoru

Při tvorbě LabVIEW programu byl kladen důraz na jeho jednoduchost, aby bylo možné dosáhnout co nejvyšší vzorkovací frekvence. Proto je jeho funkce omezena pouze na generování vstupního signálu, vyčítání surové hodnoty A/D převodníku enkodéru a ukládání těchto hodnot do .csv souboru. Veškerou následující práci nad daty jsem provedl až po skončení experimentu, kdy nebyla potřeba provádět tyto operace v reálném čase.

8.3.1 Odvození diferenciální rovnice stejnosměrného motoru

Aby bylo možné identifikovat DC motoru, je nutné nejprve sestavit diferenciální rovnici, jejíž parametry budou určovány.



Obrázek 8.7. Náhradní schéma stejnosměrného motoru

Nahradíme-li stejnosměrný motor se sériovým buzením náhradním schématem dle obr. 8.7, lze jeho diferenciální rovnici odvodit z 2. Kirchhoffova zákona. Ten říká, že součet napětí ve smyčce je roven nule. Lze tedy psát:

$$u(t) - u_R(t) - u_L(t) - u_i(t) = 0 \quad (8.3)$$

u_R – napětí na odporu – je dáno rovnicí

$$u_R(t) = Ri(t) \quad (8.4)$$

kde R je odpor motoru a i proud jím procházejícím.

Napětí indukované na cínce u_L lze vyjádřit jako

$$u_L(t) = L \frac{di(t)}{dt} \quad (8.5)$$

kde L je indukčnost motoru.

Indukované napětí na motoru pak lze vyjádřit rovnicí

$$u_i(t) = C_{ss}\phi\omega(t). \quad (8.6)$$

Je tedy přímo závislé je úhlové rychlosti ω , konstantě stejnosměrného stroje C_{ss} a magnetickému toku ϕ .

Jestliže rovnice (8.3) popisuje elektrické vlastnosti DC motoru. Mechanické vlastnosti lze popsat pomocí rovnováhy momentů

$$M(t) - M_z(t) = M_\alpha(t) + M_\omega(t) \quad (8.7)$$

kde M je magnetický moment motoru daný rovnicí

$$M(t) = C_{ss}\phi i(t). \quad (8.8)$$

Tento moment působí na hmotu, což je charakterizováno rovnicemi

$$\begin{aligned} M_\alpha(t) &= J \frac{d\omega(t)}{dt} \\ M_\omega(t) &= b\omega(t) \end{aligned} \quad (8.9)$$

kde J je moment setrvačnosti rotoru a b je třecí koeficient.

M_z je moment zátěže motoru. Uvažovaný jako nulový.

Po dosazení rovnic (8.4), (8.5) a (8.6) do (8.3) a rovnic (8.8) a (8.9) do (8.7) dostaneme následující rovnice (pro zjednodušení je zavedena substituce $C_{ss}\phi = j$)

$$\begin{aligned} L \frac{di(t)}{dt} + Ri(t) &= U(t) - j\omega(t) \\ J \frac{d\omega(t)}{dt} + b\omega(t) &= ji(t) \end{aligned} \quad (8.10)$$

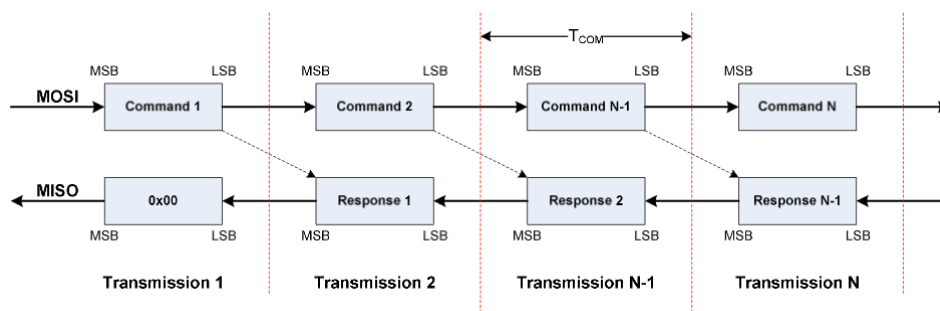
Následnou Laplaceovou transformací a dosazením jedné rovnice do druhé přes vyjádřený proud dostaneme přenosovou funkci úhlové rychlosti na vstupním napětí $G_{U\Omega}(s)$

$$G_{U\Omega}(s) = \frac{\Omega(s)}{U(s)} = \frac{y}{(Js + b)(Ls + R) + j^2} \quad (8.11)$$

8.3.2 Zpracování experimentálních dat

Aby zpracování všech dat po samotném měření bylo co nejrychlejší, bylo automatizováno vytvořeným skriptem v programu MATLAB. Tento skript byl odladěn, čímž se také předchází chybám vzniklým nepozorností při jakémkoliv dalším zpracování.

Samotný skript nejprve načte požadovaný .csv soubor a vytvoří vektory z hodnot během experimentu – jmenovitě se jedná o vektor „time“, tedy v jakém časovém okamžiku proběhlo odečtení hodnot, „pwm“ a „divs“, vektory vstupního signálu PWM respektive hodnoty z enkodéru. V dalším kroku dochází k synchronizaci těchto vektorů, neboť z výňatku z technické dokumentace k senzoru AS5048 na obr. 8.8 je patrné, že při komunikaci přes sběrnici SPI je odpověď na dotaz aktuální hodnoty A/D převodníku o jeden krok opožděna.



Obrázek 8.8. Datový tok na sběrnici SPI [37]

Výsledkem tak jsou tři synchronizované vektory – čas, vstup a výstup. Nicméně hodnoty výstupu nyní nabývají pouze hodnot 0 až 16383, tedy z rozsahu využitého 14bitového převodníku, ignorují tím fakt, že kolo vykonává pohyb ve větším rozsahu než jedna celá otáčka. Tento fakt je ošetřen pomocí příkazu *unwrap* při tvorbě vektoru úhlového natočení kola

```
phi = unwrap(k*divs)
```

kdy je navíc rovnou interpretována hodnota enkodéru do reálného světa a sice úhel natočení kola vypočten pomocí koeficientu k . Ten je dán vztahem

$$k = \frac{2\pi}{16384} \doteq 3,8 \cdot 10^{-4}. \quad (8.12)$$

Následně je vytvořen vektor úhlové rychlosti příkazem

```
w = [0;diff(phi)/dt]
```

kde příkaz $\text{diff}(\text{phi})/\text{dt}$ prakticky provádí následující výpočet

$$\omega(k) = \frac{\zeta_k - \zeta_{k-1}}{\Delta t} [\text{rad/s}], \quad (8.13)$$

ζ_k a ζ_{k-1} jsou úhly natočení kola v aktuálním respektive předchozím kroku, Δt je vzorkovací perioda.

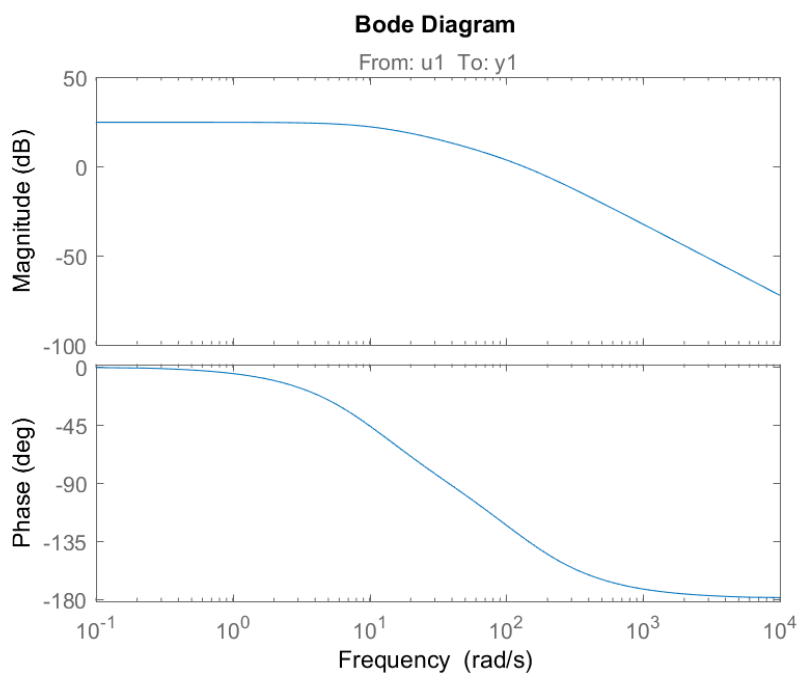
Metodou nejmenších čtverců byla určena výsledná přenosová funkce následujícím příkazem:

```
sys_est = tfest(sys_est,np,nz)
```

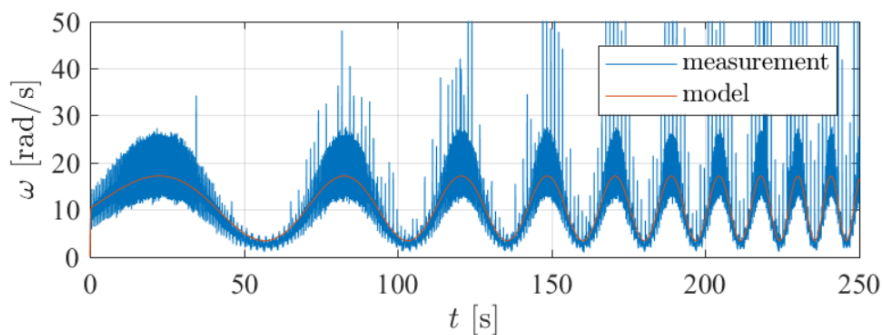
kde np a nz je počet pólů, respektive počet nul z rovnice (8.11).

Přenosová funkce s identifikovanými koeficienty má předpis (8.14) a její frekvenční charakteristika je na obr. 8.9.

$$G_{U\Omega}(s) = \frac{2380}{s^2 + 133s + 1387} \quad (8.14)$$



Obrázek 8.9. Frekvenční charakteristika identifikované funkce



Obrázek 8.10. Průběh úhlových rychlostí během experimentu a simulace

Na obr. 8.10 je porovnání naměřené úhlové rychlosti během experimentu (modře) a simulovaná odezva identifikované přenosové funkce na stejný vstupní signál (oranžově). Lze vidět, že i přes značný šum v naměřených datech se tyto křivky dobře překrývají. Kompletní přepis MATLAB skriptu lze nalézt v příloze E.

Kapitola 9

Testování navrženého prototypu

Na závěr bylo provedeno experimentální vyzkoušení funkčnosti navrženého roveru. Z pohledu rámu jsem se zaměřil především na to, jak vydrží dlouhodobější jízdu na nerovném terénu a občasné nárazy do překážek. Avšak tím hlavním, co jsem při testování hodnotil, byla funkčnost a přesnost algoritmu pro plně autonomní jízdu.

Kvůli prodávám ve výrobě a dopravě desky plošných spojů jsem pro testování upravil prvotní verzi DPS. Konkrétně byly dopájeny vodiče pro připojení senzorů. Připojení GPS si vyžadovalo uvolnění nešťastně zabrané sériové linky, provedl jsem tedy vyvedení jednoho řídicího signálu H-můstku na jiný pin mikrokontroléru než je na původním návrhu DPS – s čímž byla spojena nutná úprava řídicího programu.

Vzhledem k tomu, že algoritmus vychází z GPS dat, bylo nutné pro testování vybrat otevřený prostor ideálně bez budov, které by stínily signálům ze satelitům. Dalším požadavkem byla délka takového prostoru, aby se ukázala stabilita navigace během delší jízdy. Také bylo nutné, aby prostor neobsahoval překážky, které by robota omezovaly v jízdě.

Jako první místo k testování byl vybrán chodník v Parku I. Gándhiové (N 50.103302°, E 14.3883233°). Ukázalo se všem, že při ozkoušeném nastavení Madgwickova filtru na hladkém betonu úhel yaw na takto hrubém asfaltu příliš osciloval. To bylo způsobeno násobně většími hodnotami zrychlení v ose z než bylo dosahováno na betonu. Částečným řešením tak bylo snížit měřitelný rozsah akcelerometru z 10 na 2 g, avšak i poté se robot často odchyloval od přímé cesty k cíli. Bylo tedy nutné vybrat prostor, který je nejen dlouhý, ale také široký. K dalším testům jsem tedy zvolil veřejné parkoviště v Hořicích (N 50.3656097°, E 15.6446628°), které se ukázalo jako vyhovující.

Testování probíhalo tím způsobem, že jsem provedl zjištění hodnot úhlu yaw pro jednotlivé světové strany, na základě kterých probíhá mapování v algoritmu. Následně zbývalo souřadnice předem vytipovaného bodu zadat do řídicího programu a ten nahrát do mikrokontroléru.

Do řídicího programu byla přidána část, která přes Bluetooth modul odesílala aktuální data. Tato data sloužila jednak pro ladění samotného programu před finálním testováním, ale především byla během testování logována a následně vyhodnocována.

Na obr. 9.1 lze vidět porovnání přímé trajektorie spojující bod začátku a cíle (oranžovou barvou) a skutečnou trajektorii pocházející z dat GPS senzoru (modrou barvou). Skoková změna v trajektorii kolem N 50.3655° byla způsobena nejspíše poblíž stojícím autem, neboť během jízdy nebyla pozorována žádná skoková odchýlení od požadovaného azimutu.

Kapitola 10

Závěr

Cílem této bakalářské práce byl návrh prototypu autonomního robota tak, aby splňoval pravidla soutěže ARLISS.

Na základě rešerše dosavadních konstrukcí rámu robotů jsem navrhl rám pro typ rover. Po CAD návrhu vznikla výrobní dokumentace (přiložena v přílohách), následně byl rám odeslán do výroby a sestaven.

Dále proběhl návrh řídicí a výkonové elektroniky, pro kterou byla navržena, vyrobena a osazena deska plošných spojů. Výkresová dokumentace je opět přiložena v přílohách.

Pro možnost účasti v soutěži byl pomocí osazených senzorů vytvořen řídicí algoritmus umožňující plně autonomní jízdu ve volném prostoru. Přepis tohoto program je také přiložen v přílohách.

Následně byla představena implementace Kalmanova filtru pro tuto úlohu a popsán matematický model popisující pohyb roveru na základě rychlosti kol. Pro stanovování těchto rychlosti pak byl proveden experiment, pomocí něhož jsem identifikoval odvozenou přenosovou funkci závislosti úhlové rychlosti na střídě řídicího signálu použitých stejnosměrných motorů.

V rámci testování navržený prototyp úspěšně urazil potřebnou vzdálenost a zastavil se na požadovaných souřadnicích. Lze ho tedy hodnotit jako úspěšný.

V rámci dalšího vývoje je nutné se zaměřit na upravení rámu robota tak, aby byl schopný pohybu v drsnějším terénu. Zajímavou roli by v tomto mohlo hrát využití 3D tisky (např. na výroby měkkých kol). Dále by bylo vhodné navázat na rozpracovanou implementaci Kalmanova filtru, který by umožňoval přesnější určování polohy roveru a tím by bylo možné ušetřit energii zmařenou prodlužováním jízdní trasy.

Literatura

- [1] THE SPACE ROBOTICS LAB, TOHOKU UNIVERSITY. Arliss sequence. In: Tohoku University The Space Robotics Lab [online]. Tohoku: The Space Robotics Lab, Tohoku University, 2003 [cit. 2019-05-08]. Dostupné z: <http://www.astro.mech.tohoku.ac.jp/~ishigami/ARLISS2003/fig/sequence2.gif>
- [2] KANAI, Akiko. NOKO NOKO. In: University Space Engineering Consortium [online]. Japonsko: University Space Engineering Consortium, 2005 [cit. 2019-05-08]. Dostupné z: <http://unisec.jp/history/arliss2005/img/nokonoko.jpg>
- [3] C.R.A.W.LAB AT UNIV OF LOUISIANA AT LAFAYETTE. ARLISS 2016 - Day 1 - 13. In: Flickr [online]. San Francisco: flickr, 2016 [cit. 2019-05-08]. Dostupné z: <https://www.flickr.com/photos/crawlab/29581873982/in/album-72157669559929794>
- [4] C.R.A.W.LAB AT UNIV OF LOUISIANA AT LAFAYETTE. ARLISS 2015 - Day 2 - 40. In: Flickr [online]. San Francisco: flickr, 2015 [cit. 2019-05-08]. Dostupné z: <https://www.flickr.com/photos/crawlab/21318906360/in/album-72157649810217779>
- [5] C.R.A.W.LAB AT UNIV OF LOUISIANA AT LAFAYETTE. ARLISS 2016 - Day 2 - 29. In: Flickr [online]. San Francisco: flickr, 2016 [cit. 2019-05-08]. Dostupné z: <https://www.flickr.com/photos/crawlab/29127015794/in/album-72157669559929794/>
- [6] KIM, YooSeok. Wheel Transformer; A Wheel Leg Hybrid Robot with Passive Transformable Wheels. In: YouTube [online]. San Bruno: YouTube, 2014 [cit. 2019-05-08]. Dostupné z: <https://www.youtube.com/watch?v=syUfxj40Yi8>
- [7] C.R.A.W.LAB AT UNIV OF LOUISIANA AT LAFAYETTE. ARLISS 2015 - Day 1 - 10. In: Flickr [online]. San Francisco: flickr, 2015 [cit. 2019-05-08]. Dostupné z: <https://www.flickr.com/photos/crawlab/21296147100/in/album-72157649810217779/>
- [8] KOBAYASHI, Kosuke. Successful landing. In: Faculty of Engineering, The University of Tokyo [online]. Tokyo: Faculty of Engineering, The University of Tokyo, 2018 [cit. 2019-05-08]. Dostupné z: http://www.t.u-tokyo.ac.jp/shared/topics/images/setnws_201809201756516885649080_958597.jpg
- [9] Autonomní samorozkládací koptéra pro účel soutěže ARLISS/RescueBot. In: HLA-VÁČ, Vladimír. Nové metody a postupy v oblasti přístrojové techniky, automatického řízení a informatiky 2017. Praha: České vysoké učení technické v Praze, 2017, s. 35-39. ISBN 978-80-01-06300-2.
- [10] AVGOUSTOPOULOS, Constantine. ME-4699: FlyingHopperresearch. Atlanta, 2013. Závěrečná zpráva. Georgia Tech.
- [11] GREGORY, R. L. a Pauline MARSTRAND. Creative intelligences. Norwood, N.J.: Ablex Pub., c1987. ISBN 0893914401.



- [online]. 12(6), 869-880 [cit. 2019-06-09]. DOI: 10.1109/70.544770. ISSN 1042296X. Dostupné z: <http://ieeexplore.ieee.org/document/544770/>
- [28] Encoder Disc (3-Bit). In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2019-05-08]. Dostupné z: [https://en.wikipedia.org/wiki/File:Encoder_Disc_\(3-Bit\).svg](https://en.wikipedia.org/wiki/File:Encoder_Disc_(3-Bit).svg)
- [29] Návrh konstrukce a řízení dvoukolového autonomního robota. In: MORAVEC, Jiří. Konference studentské tvůrčí činnosti STČ 2019. Praha: ČVUT v Praze, Fakulta strojní, 2019. ISBN 978-80-01-06564-8.
- [30] Dealing with Motor Noise. Pololu Robotics Electronics [online]. Las Vegas: Pololu Corporation, c2011-2019 [cit. 2019-05-08]. Dostupné z: <https://www.pololu.com/docs/0J15/9>
- [31] H., Larry. Getting the Best Out of Your LiPo Batteries. HeliDirect [online]. Middletown: HeliDirect, 2018 [cit. 2019-05-08]. Dostupné z: <https://www.helidirect.com/blog/getting-the-best-out-of-your-lipo-batteries.html>
- [32] GUOWEI, Cai. Unmanned Rotorcraft Systems. London: Springer, 2011. ISBN 9780857296344.
- [33] WINER, Kris. Arduino sketches for MPU9250 9DoF with AHRS sensor fusion. GitHub [online]. San Francisco: GitHub, c2019 [cit. 2019-05-08]. Dostupné z: <https://github.com/kriswiner/MPU9250>
- [34] X-IO TECHNOLOGIES. Open source IMU and AHRS algorithms. X-io Technologies [online]. UK: x-io Technologies, c2012-2016 [cit. 2019-05-08]. Dostupné z: <http://x-io.co.uk/open-source-imu-and-ahrs-algorithms/>
- [35] JURIC, Darko. Cycle. In: Code project [online]. CodeProject, 2015 [cit. 2019-05-08]. Dostupné z: <https://www.codeproject.com/KB/recipes/865935/cycle.png>
- [36] HOANG, T. T., P. M. DUONG, N. T. T. VAN, D. A. VIET a T. Q. VINH. Development of a multi-sensor perceptual system for mobile robot and EKF-based localization. In: 2012 International Conference on Systems and Informatics (ICSAI2012) [online]. IEEE, 2012, 2012, s. 519-523 [cit. 2019-05-08]. DOI: 10.1109/ICSAL.2012.6223050. ISBN 978-1-4673-0199-2. Dostupné z: <http://ieeexplore.ieee.org/document/6223050/>
- [37] AUSTRIAMICROSYSTEMS AG. AS5048A/AS5048B Datasheet: Revision 1.3. Premstätten, c1997-2012. Dostupné také z: <https://media.digikey.com/pdf/Data%20Sheets/Austriamicrosystems%20PDFs/AS5048A,B.pdf>

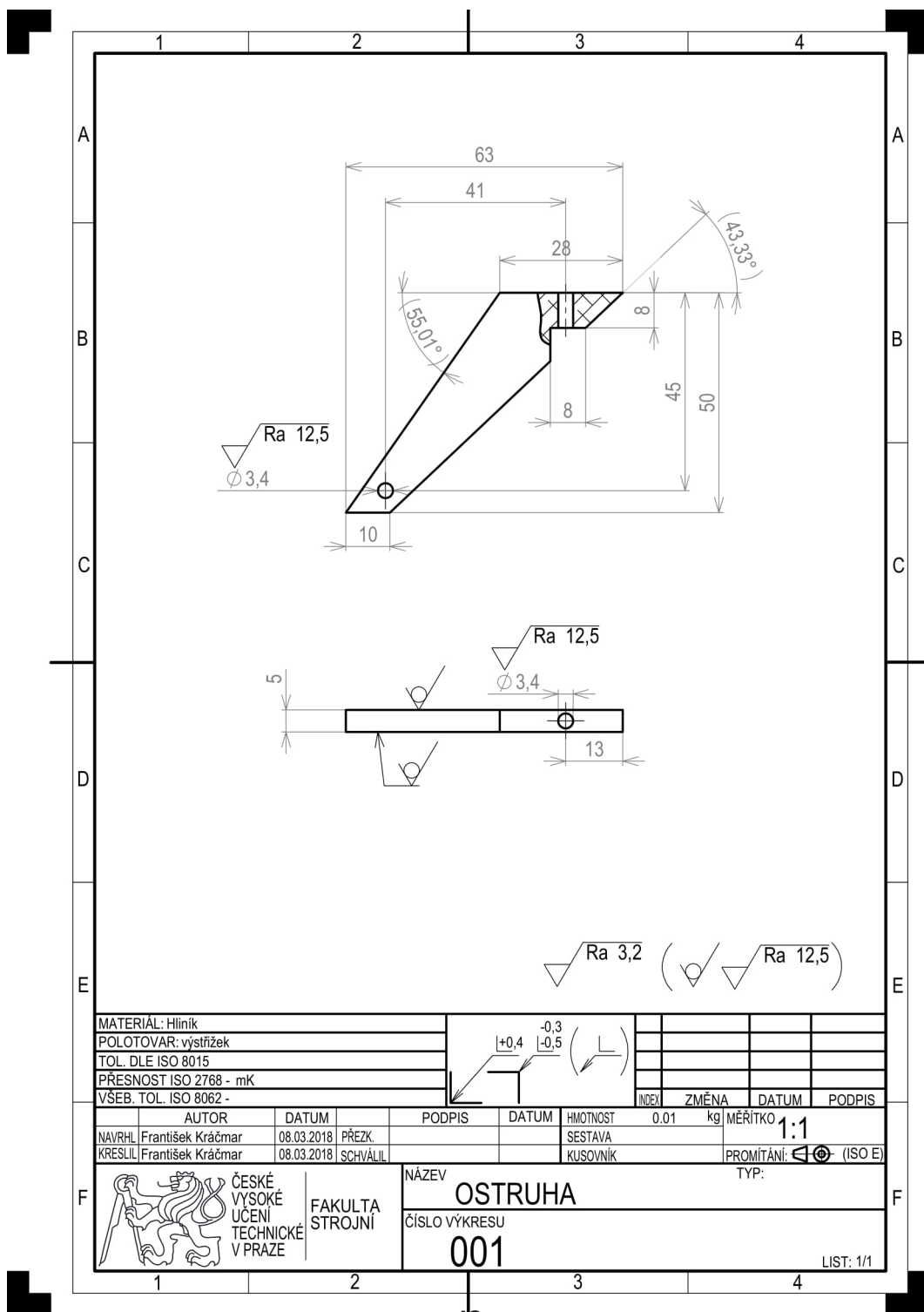
Příloha A

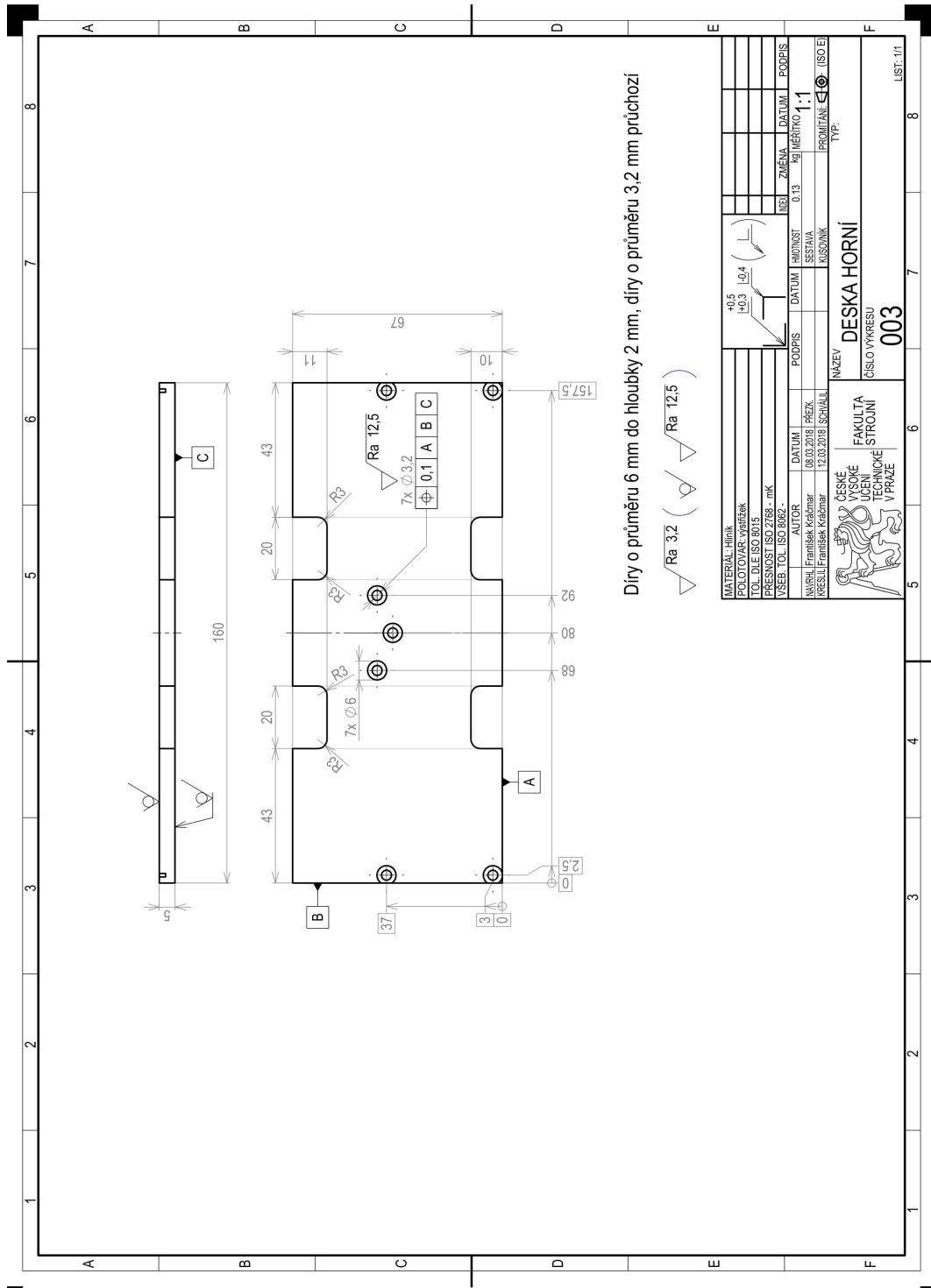
Seznam použitých symbolů a značení

α	Úhlové zrychlení
β	Požadovaný azimut
δ	Regulační úhlová odchylka
$\Delta\psi$	Změna v azimutu robota za čas Δt
Δs	Vzdálenost, kterou robot urazí za čas Δt
Δs_l	Vzdálenost, kterou levé kolo urazí za čas Δt
Δt	Vzorkovací perioda
ω	Úhlová rychlost
Ω	Laplaceův obraz úhlové rychlosti
ω_l	Úhlová rychlost levého kola
ω_r	Úhlová rychlost pravého kola
ϕ	Magnetický tok
π	Matematická konstanta (3,14159)
ψ	Aktuální azimut roveru
ψ_k	Azimut v kroku k
ψ_{k+1}	Azimut v kroku $k + 1$
ζ_k	Úhel natočení kola v kroku k
ζ_{k-1}	Úhel natočení kola v kroku $k - 1$
a	Zrychlení
b	Třecí koeficient
B_k	Matice vlivů externích proměnných
c	Vzdálenost středů kol
C_{ss}	Konstanta stejnosměrného stroje
CAD	Počítačem podporované projektování (z anglického Computer Aided Design)
d	Průměr kola
e	Poloměr zatačení
E	Střed otáčení
DPS	Deska plošných spojů
f_{max}	Maximální otáčky kol
F	Síla
f_0	Počáteční frekvence funkce chirp
f_1	Konečná frekvence funkce chirp
F_k	Matice vnitřních vlivů stavových proměnných
g	Generovaná funkce typu chirp
GNSS	Globální družicový polohový systém (z anglického Global Navigation Satellite System)
GPS	Globální polohový systém (z anglického Global Positioning System)
h	Míra exponenciálního růstu
H_k	Transformační matice vektoru naměřených hodnot na vektor stavových proměnných

Příloha B

Výrobní výkresová dokumentace navrženého rámu

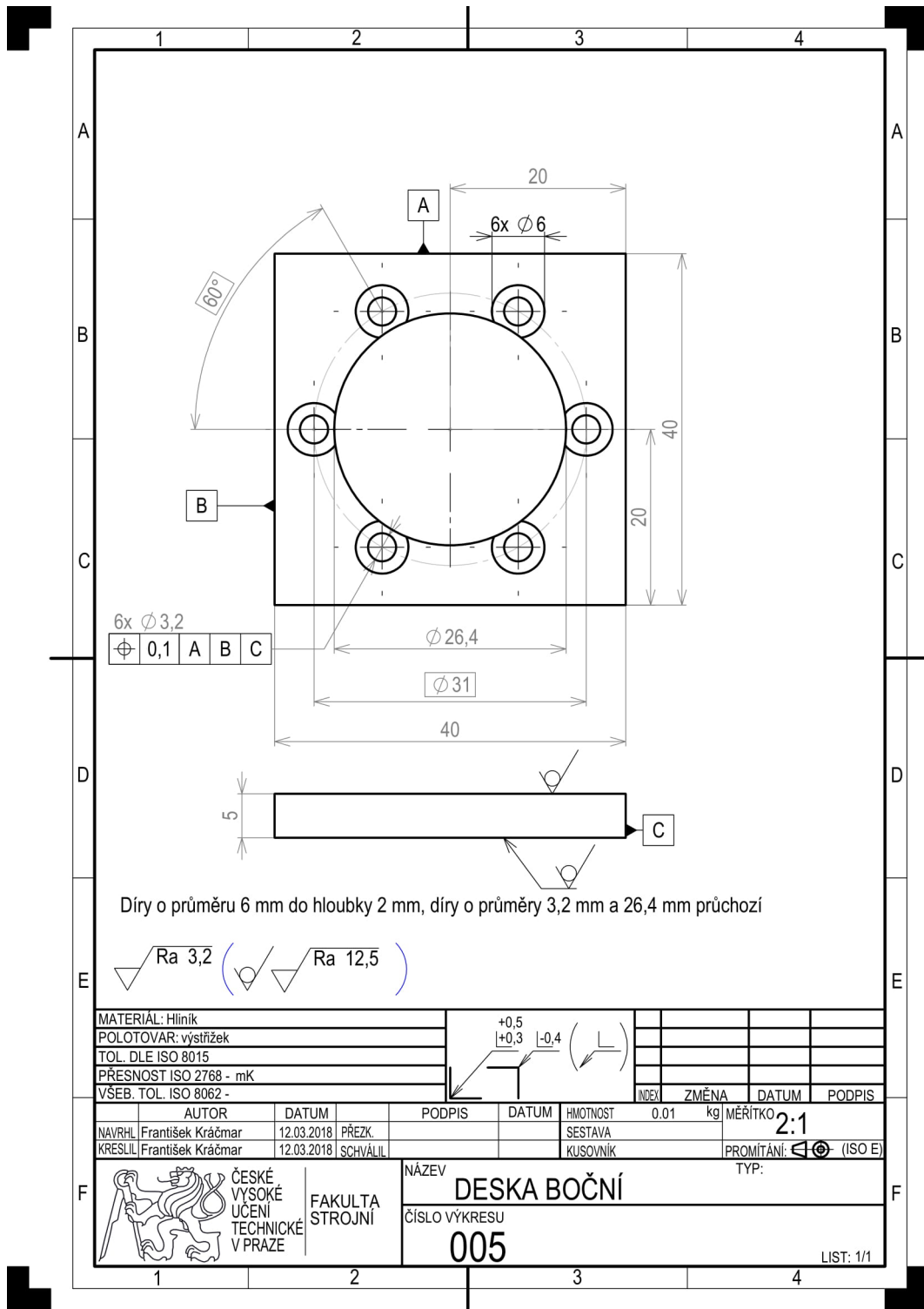


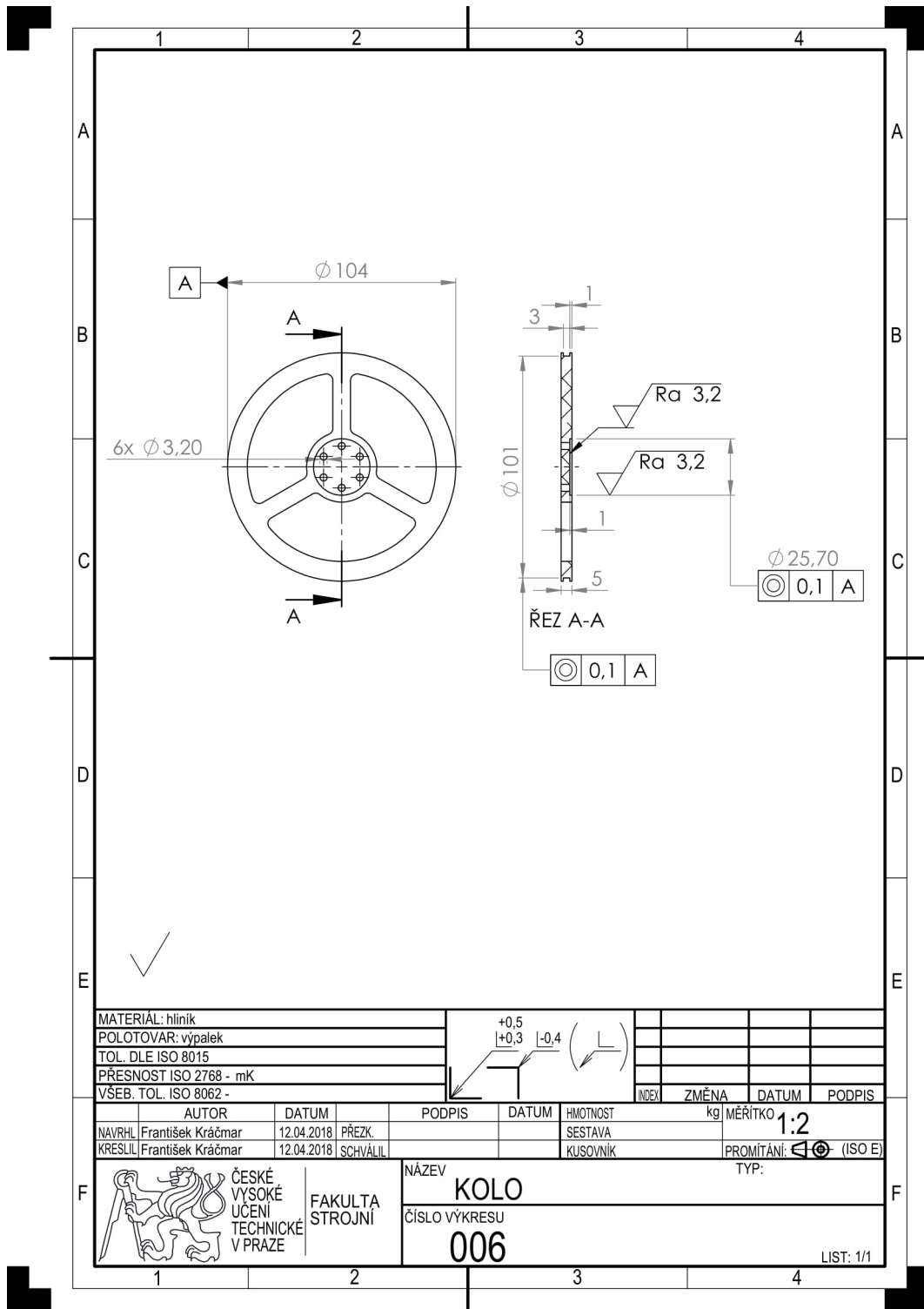


Diry o průměru 6 mm do hloubky 2 mm, díry o průměru 3,2 mm průchozí

∇ Ra 3,2 (∇ Ra 12,5)

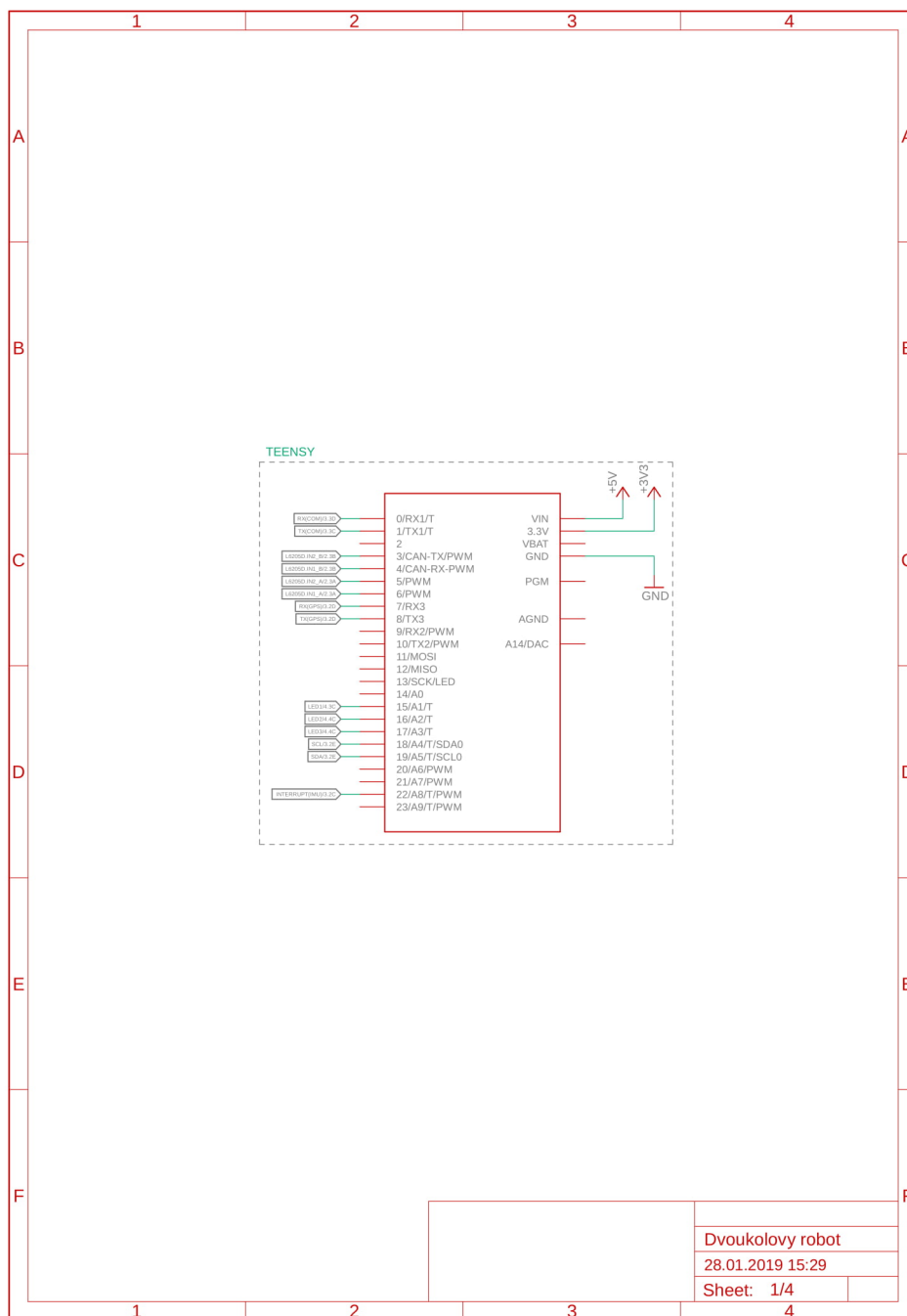
MATERIAL: Hliník	POLOTVAR: výstřelek	TOL. DLE ISO 8015	PRESNOST ISO 2768 - mK	VSEB. TOL. ISO 8662 -	AUTOR: MVR/PL, František Kráčmar	DATUM: 08.03.2018	PŘÍZK: 72.03.2018	SPRÁVIL: ŠPAVÁL	PODPIS: [Signature]	HMĚNOST: 0,13	HMĚNOST: 0,13	DATUM: 1.1.	DATUM: 1.1.	POPPIS: [Signature]
<p>ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ STROJNÍ V PRAZE</p> <p>FAKULTA UČENÍ TECHNICKÉ STROJNÍ</p> <p>DESKA HORNÍ</p> <p>ČÍSLO VÝKRESU 003</p> <p>PROJITÁNÍ: [Signature]</p> <p>TYP: [Signature]</p>														
LIST: 1/1														

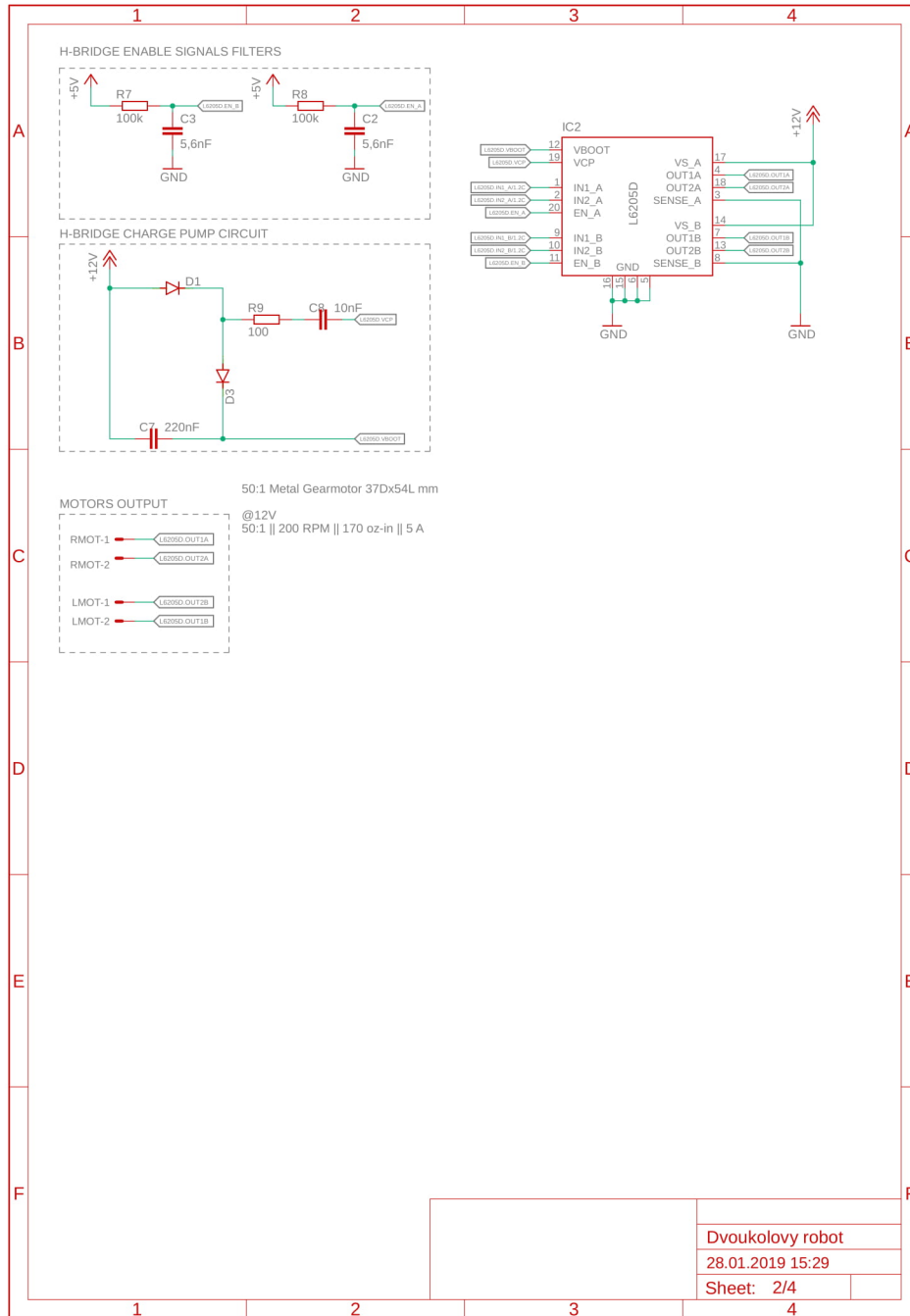


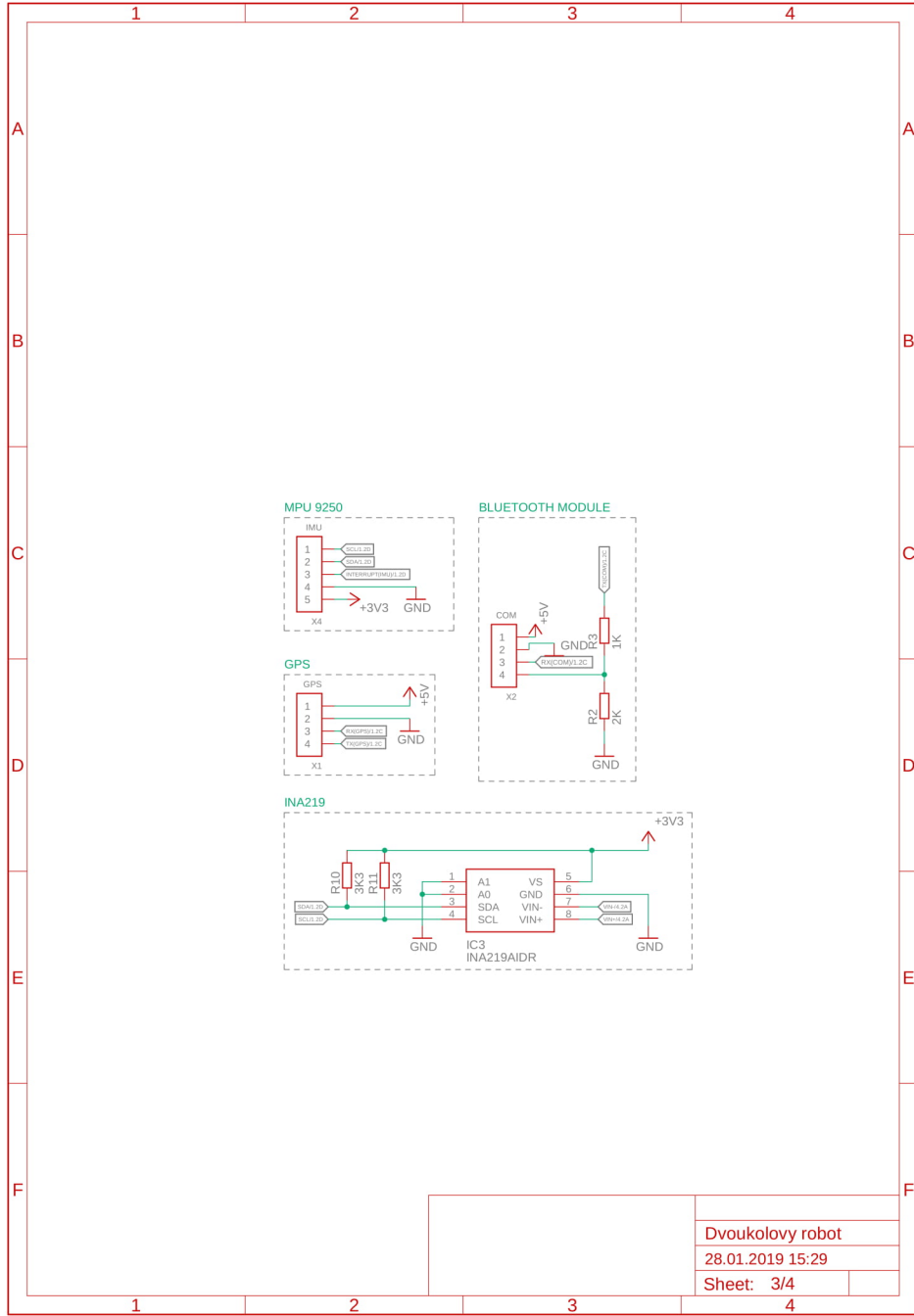


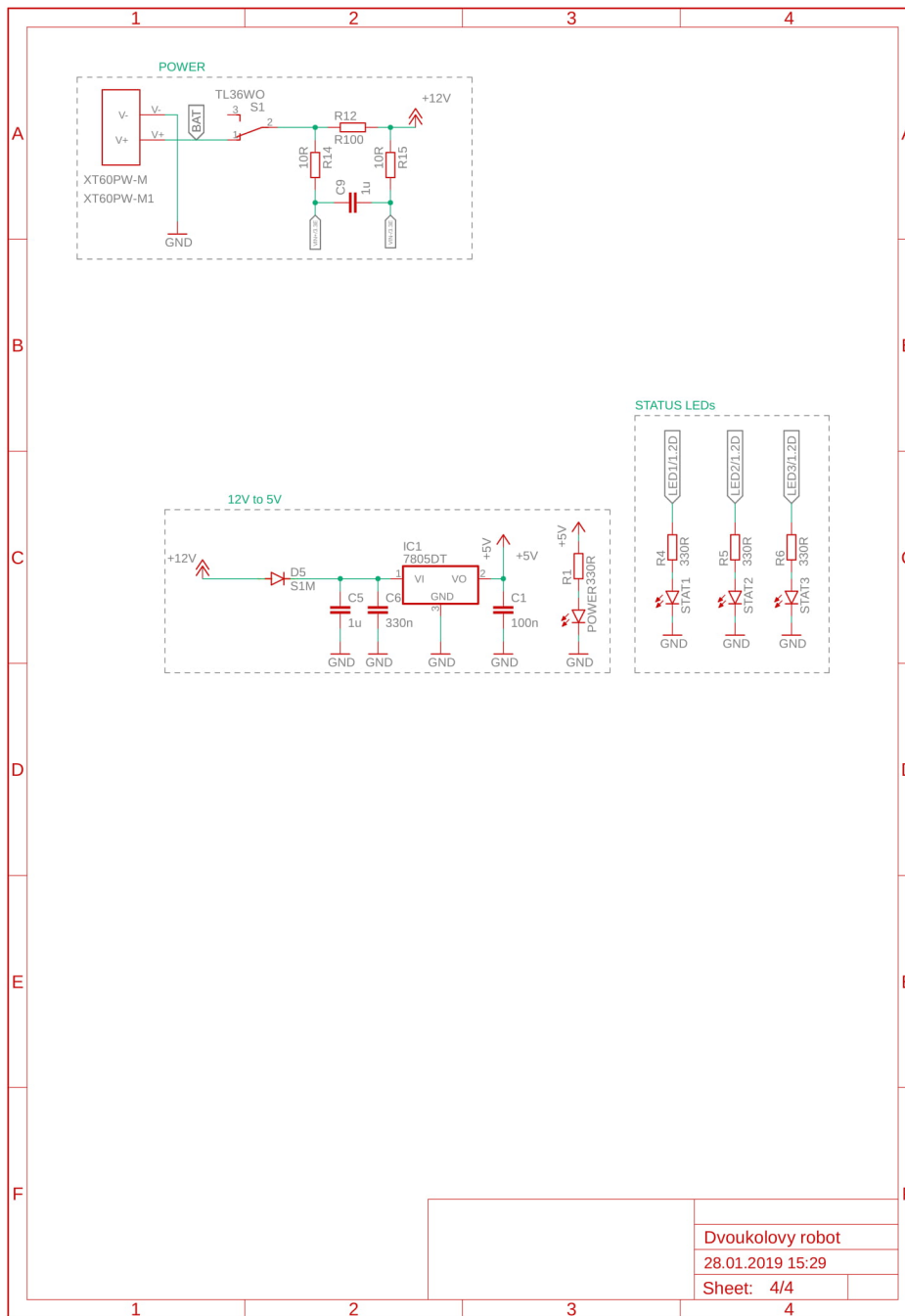
Příloha C

Výkresová dokumentace navržené desky plošných spojů









Příloha D

Řídicí program mikrokontroléru

```
#include "MPU9250.h"
#include "MadgwickAHRS.h"
#include "SparkFun_TB6612.h"
#include <Wire.h>
#include <TinyGPS.h>

// H-bridge PINs
#define PWMB 3
#define BIN2 5
#define BIN1 4
#define STBY 6
#define AIN1 7
#define AIN2 8
#define PWMA 22

// Motors offset (correct rotation)
#define MOTOR_A_OFFSET 1
#define MOTOR_B_OFFSET 1

// Initialize motors
Motor motor1 = Motor(AIN1, AIN2, PWMA, MOTOR_A_OFFSET, STBY);
Motor motor2 = Motor(BIN1, BIN2, PWMB, MOTOR_B_OFFSET, STBY);

// DATA ACQUISITION AND AHRS ALGORITHM
int intPin = 15;
float ax, ay, az, gx, gy, gz, mx, my, mz;
// MPU9250 raw data
float accResolution, gyroResolution, magResolution;
// Resolutions
float pitch, roll, yaw;
// Magdwick outputs
int yaw_mapped;
// Rover's azimuth corresponding with coordinate system
float magCalibration[3] = {0, 0, 0};
// Not used, but necessary
int16_t accelCount[3], gyroCount[3], magCount[3];
// MPU9250 raw data vectors

// GPS
float latitudeRover, longitudeRover, GPSAltitude, GPSSpeed;
// Rover's coordinates. latitude and speed
int GPSSatellites, GPShdop;
// Additional GPS data (satellites, horizontal dilution of precision)
unsigned long dataAge;
```


Příloha E

MATLAB skript pro identifikaci přenosové funkce

```
%% Initialization
clc
close all
clearvars
%% Set parameters
dt = 0.0025;
k = 2*pi/16384;
filename = 'complete_spectrum0,2-1.csv';
%% Import data from text file
delimiter = ',';
formatSpec = '%f%f%f%[\n\r]';
fileID = fopen(filename,'r');
dataArray = textscan(fileID,formatSpec,...
    'Delimiter',delimiter,...
    'TextType','string',...
    'EmptyValue',NaN,...
    'ReturnOnError',false);
fclose(fileID);
time = dataArray{: , 1};
pwm = dataArray{: , 2};
divs = dataArray{: , 3};
clearvars filename delimiter formatSpec fileID dataArray ans;
%% Synchronize vectors
time(end) = [];
pwm(end) = [];
divs(1) = [];
%% Postprocessing
phi = unwrap(k*divs); % angle
w = [0;diff(phi)/dt]; % angular velocity
%% System identification
u = pwm; % input
y = w; % output
t = linspace(0,time(end),length(y)); % time
np = 2;
nz = 0;
sys_est = iddata(y,u,dt);
sys_est = tfest(sys_est,np,nz);
y_est = lsim(sys_est,u,t);
%% Graphical output
% Time plot
figure
subplot(3,1,1)
plot(time,pwm)
```

