VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ROBOTICKÉ VOZIDLO S VYUŽITÍM RC KOMPONENTŮ

ROBOTIC VEHICLE USING RC COMPONENTS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. ONDŘEJ DEINGRUBER**
AUTHOR

VEDOUCÍ PRÁCE — **Prof. Dr. Ing. PAVEL ZEMČÍK**
SUPERVISOR

BRNO 2021

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav počítačové grafiky a multimédií (UPGM) Akademický rok 2020/2021

Zadání diplomové práce

Student: **Deingruber Ondřej, Bc.**
Program: Informační technologie
Obor: Inteligentní zařízení
Název: **Robotické vozidlo s využitím RC komponentů**
Robotic Vehicle Using RC Components
Kategorie: Uživatelská rozhraní
Zadání:

1. Prostudujte způsob řízení modelářských serv a možnosti jejich ovládání počítačem. Dále prostudujte existující konstrukce "robotických" zařízení s využitím RC komponentů.
2. Navrhněte jednoduchý model robotického vozidla sestrojený s využitím modelářských RC komponentů a případně dalších dostupných "off the shelf" komponentů, případně 3D tisku tak, aby byl snadno opakovatelně sestrojitelný.
3. Navrhněte konstrukci/implementaci vozidla a způsob jeho ovládání s ohledem na to, aby získalo využitím počítače nové lepší vlastnosti.
4. Popište možnosti konstrukce, ovládání i automatizace vozidla a diskutujte dosažitelné vlastnosti.
5. Navržený model vozidla implementujte a demonstrujte jeho vlastnosti na vhodné úloze.
6. Diskutujte dosažené výsledky a možnosti pokračování práce.

Literatura:

- Dle pokynů vedoucího

Při obhajobě semestrální části projektu je požadováno:

- Body 1 až 4 zadání

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Zemčík Pavel, prof. Dr. Ing.**
Vedoucí ústavu: Černocký Jan, doc. Dr. Ing.
Datum zadání: 1. listopadu 2020
Datum odevzdání: 19. května 2021
Datum schválení: 30. října 2020

Abstrakt

Tato práce se zabývá řízením modelářských RC serv a konstrukcemi robotických vozidel. Cílem je navrhnout model robotického vozidla s využitím RC komponentů, dalších "off the shelf" komponentů a 3D tisku a demonstrovat jeho vlastnosti. V práci byla navržena architektura robota využívající jednodeskový počítač společně s běžně dostupnými komponenty, 3D tisk a nevyžaduje složité sestavování. Pro návrh hřebenového řízení vozidla bylo využito optimalizačního algoritmu. Výsledkem práce je implementace robotického vozidla.

Abstract

This thesis covers the topic of controlling RC servos and constructions of robotic vehicles. The goal is to propose a model of robotic vehicle with RC components and other "off the shelf" components, 3D printing and demonstrate its capabilities. In the thesis, a mobile robot platform was proposed. It uses a single-board computer together with readily available parts and does not require complicated assembly. An optimization algorithm was used for the design of rack and pinion steering. The result of the thesis is the implementation of a robotic vehicle.

Klíčová slova

robotická vozidla, RC modelářská serva, vestavěné systémy, ROS, 3D tisk, Ackermannovo řízení

Keywords

robotic vehicles, RC servos, embedded systems, ROS, 3D printing, Ackermann steering

Citace

DEINGRUBER, Ondřej. *Robotické vozidlo s využitím RC komponentů*. Brno, 2021. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Prof. Dr. Ing. Pavel Zemčík

Robotické vozidlo s využitím RC komponentů

Prohlášení

Prohlašuji, že jsem tuto práci vypracoval samostatně pod vedením prof. Dr. Ing. Pavla Zemčíka. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Ondřej Deingruber
5. srpna 2021

Poděkování

Tímto bych rád poděkoval vedoucímu bakalářské práce prof. Dr. Ing. Pavlu Zemčíkovi za cenné rady, připomínky a vedení práce.

Obsah

Seznam obrázků

Kapitola 1

Úvod

Robotická vozidla v současnosti nachází čím dál tím větší uplatnění v praxi. Mohou, ale mít poměrně vysokou pořizovací cenu nebo jejich vybavení nemusí dostačovat pro potřebné využití. Tato práce se zabývá zhodnocením současného stavu robotických zařízení využívajících běžně dostupné komponenty z rádiově ovládaných modelů. Popisuje vybrané příklady daných zařízení, jejich možnosti a technologie potřebné pro jejich konstrukci. Na základě uvedených informací je vytvořen návrh jednoduše sestavitelného robotického vozidla. Pro zjednodušení stavby modelu je využito nově dostupné technologie 3D tisku, která díky možnosti tvorby složitých geometrií dokáže snížit vstupní bariéru přístupu k robotickým vozidlům.

Mou motivací pro tvorbu této práce bylo vytvořit robotické vozidlo, které bude dostupné pro studenty fakulty. Proto bylo taky využito jednodeskového počítače Raspberry Pi, který většina studentů již vlastní. Také jsem chtěl využít svých znalostí v oblasti návrhu modelů pro 3D tisk a dlouhodobé praxe v leteckém modelářství. Dalšími důvody pro tvorbu této práce byla možnost sestrojit robotické vozidlo dle vlastního návrhu.

Cílem práce je seznámit se s možnostmi řízení RC serv, řízením robotických vozidel, jejich automatizací a prostudovat existující konstrukce robotických zařízení s využitím RC komponentů. Na základě získaných informací a poznatků navrhnout a sestrojit robotické vozidlo využívající pouze 3D tisk a "off the shelf" komponenty. Při výběru komponent je nutno brát ohled na co nejjednodušší sestavení a minimalizování potřebného vybavení. Toto vozidlo navrhnout tak, aby bylo jednoduše a opakovatelně sestrojitelné a získalo s využitím počítače nové, lepší vlastnosti. Popsat možnosti konstrukce, jeho vlastnosti, automatizaci a možnosti ovládání. Navržený model je třeba implementovat a demonstrovat jeho vlastnosti na vhodné úloze.

Práce obsahuje informace potřebné k pochopení návrhu vozidla a jeho řízení. V kapitole 2 je popsáno řízení RC serv pomocí počítačů, vlastnosti robotických vozidel popis modelářských RC serv a příklady současných robotických konstrukcí využívající RC komponenty. Kapitola 3 obsahuje informace o vestavěných systémech využívaných v robotice, software využívaném k jejich ovládání a daších technologiích využívaných při sestavování robotických vozidel. V kapitole 4 se nachází porovnání současně dostupných technologií a jejich vlastností. Na základě zhodnocení současného stavu je vytvořena specifikace pro nové robotické vozidlo a plán práce pro jeho vytvoření. Kapitola 5 obsahuje architekturu navrženého robota, návrh hřebenového řízení, popis jeho mechanické konstrukce, elektrického zapojení a popis řízení jednotlivých komponentů. Dále kapitola obsahuje možnosti ovádání vozidla, ověření funkčnosti a interpretaci výsledků.

Kapitola 2

Shrnutí dosavadního stavu Robotických vozidel s RC komponenty

Tato kapitola obsahuje popis dělení robotických vozidel, popis RC komponentů a vybraných robotických konstrukcí, které je využívají. V této kapitole jsou poznatky s bezprostředním vztahem k práci. Tato kapitola nemá encyklopedický charakter a nutně nemusí obsahovat všechny poznatky v dané problematice.

2.1 Dělení robotických vozidel

Robotické vozidlo je robot, který je schopný se pohybovat ve svém okolí [26]. Obor robotických vozidel je považován za podobor robotiky a informačního inženýrství. Robotická vozidla mohou být autonomní, to znamená, že jsou schopny navigovat se v neznámém prostředí bez potřeby externího zásahu. Robotická vozidla také mohou využívat navigační zařízení, která jim umožňují pohybovat se po předem určené cestě v relativně kontrolovaném prostředí.

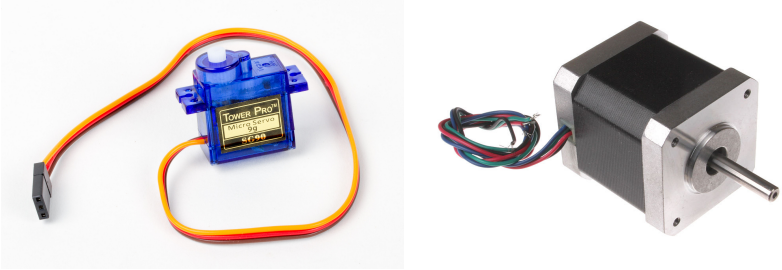
Konstrukce

Robotická vozidla se skládají ze tří základních částí

- aktuátory
- senzory
- řídící systém

Tyto části dohromady tvoří jeden funkční celek.

Robotická vozidla využívají mnoho různých aktuátorů, především motorů, servomotorů, pneumatických systémů nebo např. solenoidů. Ty kombinují s dalšími mechanickými komponenty jako kola, pásy, uchopovací ruce pro pohyb a interakci s prostředím.

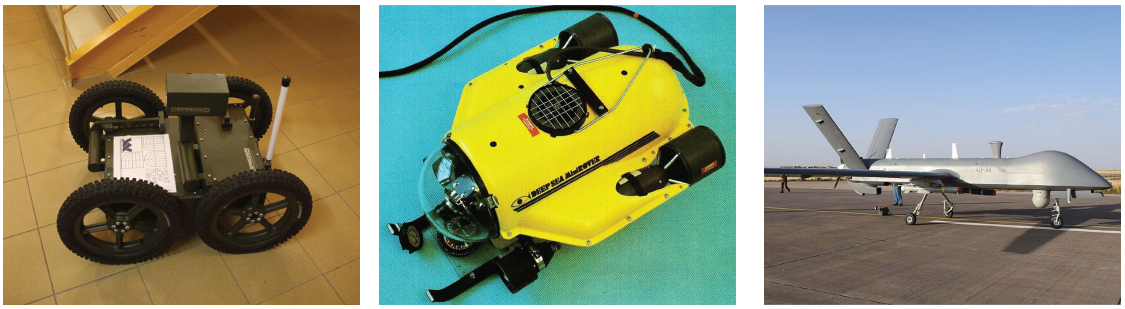


(a) RC servo (b) krokový motor

Obrázek 2.1: Příklady aktuátorů[1]

Roboty lze rozdělit do tří kategorií dle prostředí v kterém se pohybují.

- **pozemní roboti**
- **vzdušní roboti**
- **vodní roboti**



(a) pozemní robot (b) vodní robot (c) vzdušný robot

Obrázek 2.2: Příklady robotů[2]

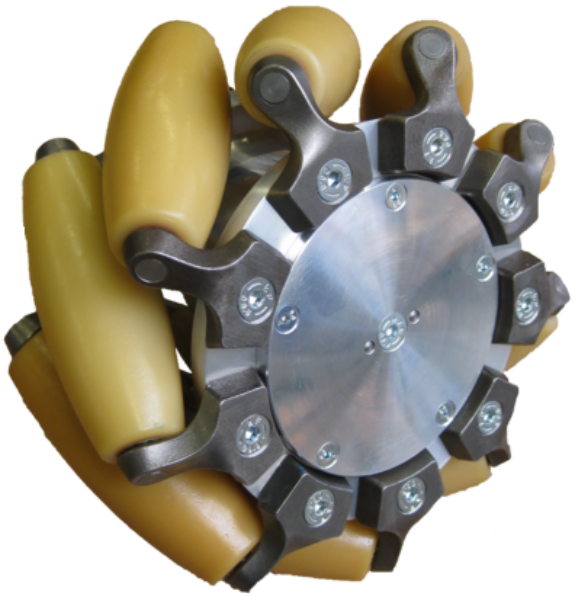
[1] převzato z: https://www.flickr.com/photos/oskay/22731304279, https://commons.wikimedia.org/wiki/File:Nema_17_Stepper_Motor.jpg

[2] převzato z: https://commons.wikimedia.org/wiki/File:Orpheus_AC_robot.jpg, https://en.wikipedia.org/wiki/File:Mini_Rover_ROV.jpg, https://catalog.archives.gov/id/6493217

2.2 Pozemní roboti

Pozemní roboti jsou pravděpodobně největší kategorií, jedná se o roboty pohybující se po zemi [4]. Mají široké využití, především pro autonomní dopravu, nebo pro přístup k místům, která jsou pro člověka nebezpečná, práce v kontaminovaných, stísněných nebo těžce dostupných prostorech. Příkladem může být odstraňování bomb nebo průzkum planet. K tomuto pohybu využívají nohy, kola, pásy i další mechanizmy. Roboti využívající nohy se nazývají chodící roboti, mohou využívat různé množství noh, kdy nejčastěji využívají 2, 4 nebo 6. Jejich konstrukce a pohyb se velmi často inspiruje v přírodě savci nebo hmyzem. Pohyb těchto robotů dělíme do dvou kategorií.

- **statický** nohy, které podpírají robota dokáží udržet jeho stabilitu (minimálně tři)
- **dynamický** při pohybu musí být udržována stabilita



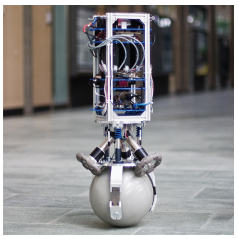
Obrázek 2.3: Všesměrové kolo[3]

Další možností pohybu pozemních robotů jsou kola. Kola mohou být poháněné a nepoháněné a poskytují různý počet stupňů volnosti, kdy nejběžnější jsou dva stupně volnosti, zvláštním případem jsou např. všesměrová kola se 3 stupni volnosti.

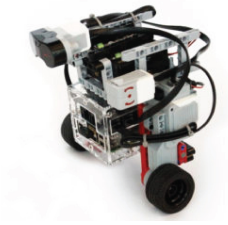
[3]převzato z: https://commons.wikimedia.org/wiki/File:Meacnum-Rad.png

Kola lze využívat v mnoha konfiguracích, nejčastěji používané jsou následující.

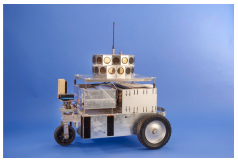
- Jednokoloví roboti
  - robot balancuje na 1 kole, popř. celý robot se nachází v kole, nejedná se o velmi praktické vozidlo
- Dvoukoloví roboti
  - jedno poháněné kolo a jedno říditelné kolo (např. motorka), robot musí balancovat pro udržení rovnováhy
  - obě kola jsou poháněna a umístěny vedle sebe (segway)
- 3 kola
  - 2 nezávisle poháněná kola a jedno všesměrové kolo
  - dvě propojená poháněná kola a jedno řiditelné kolo
  - tři všesměrová poháněná kola
  - tři poháněná nezávisle říditelná kola
- 4 kola
  - 2 páry propojených poháněných kol, jeden pár řiditelný
  - dvě poháněná propojená kola a dvě kola s Ackermannovým řízením
  - dva páry poháněných a řiditelných kol pomocí Ackermannova řízení
  - dvě nezávisle poháněná kola a dvě všesměrová kola
  - čtyři poháněná všesměrové kola
  - čtyři poháněná a nezávisle ovladatelná kola



(a) Jednokolový robot



(b) Dvoukolový robot



(c) Tříkolový robot

(d) Čtyřkolový robot

Obrázek 2.4: Příklady různých pozemních robotů[4]

[4] převzato z: https://commons.wikimedia.org/wiki/File:Ballbot_Rezero_2010.jpg, https://commons.wikimedia.org/wiki/File:Brickpi3_Balance_Bot.jpg, https://americanhistory.si.edu/collections/search/object/nmah_1404648, https://ucsdnews.ucsd.edu/feature/undergraduate-engineers-get-hands-on-experience-with-autonomous-vehicles

Běžně využívané kolové podvozky

Dvěmi nejběžnějšími kategoriemi podvozků jsou robotické vozidla jednokolového typu a podvozky s automobilovým řízením. Zástupci těchto dvou kategorií je diferenciální podvozek a Ackermannovo řízení.

Diferenciální podvozek

Diferenciální podvozek je jeden z nejpopulárnějších podvozků v robotice [4]. Skládá se ze dvou pevně uchycených poháněných kol a jednoho pasivního, volně se otáčejícího, nebo všesměrového kola. Výhody tohoto podvozku jsou:

- jednoduchá mechanická struktura
- jednoduché řešení kinematických rovnic
- nízká výrobní cena
- možnost otáčení na místě
- jednoduchá kalibrace

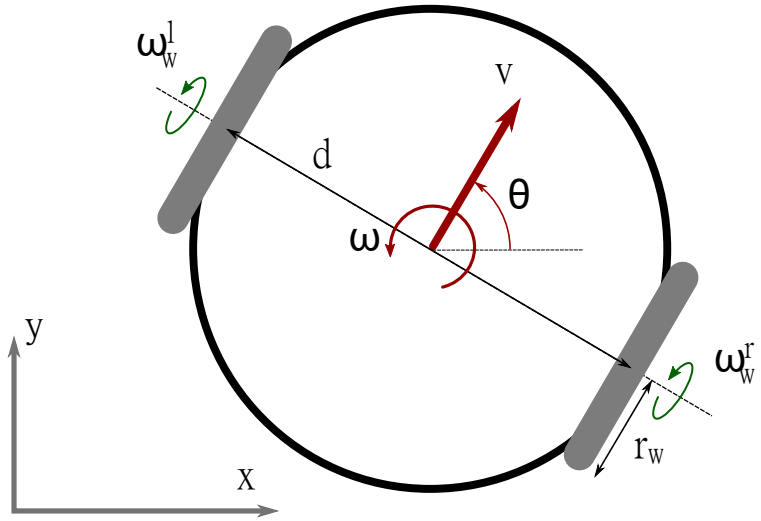
Mezi hlavní nevýhody patří:

- obtížný pohyb po nerovném povrchu
- pohyb pouze ve dvou směrech

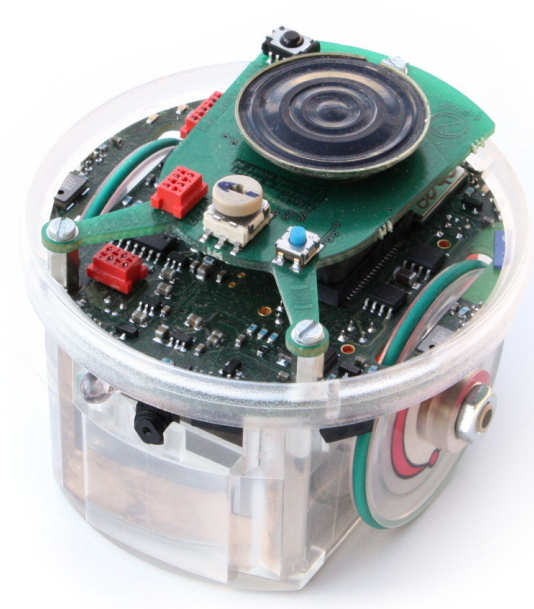
Ovládání směru pohybu a poloměru zatáčení probíhá pomocí dvou poháněných kol. Pokud se kola otáčí stejnou rychlostí a směrem, vozidlo se bude pohybovat rovně. Pokud se obě kola otáčí stejnou rychlostí ale opačným směrem, bude se vozidlo otáčet okolo středu osy poháněných kol. Za předpokladu ideálních kol je poloměr zatáčení popsaný následujícími rovnicemi.

$$\begin{aligned} \theta &= \frac{v_r - v_l}{d} \\ R &= \frac{d(v_r + v_l)}{2(v_r - v_l)} \\ V &= \theta R \end{aligned} \tag{2.1}$$

Kde θ je úhlová rychlost, v_r je rychlost pravého kola, v_l rychlost levého kola, d je vzdálenost mezi poháněnými koly vozidla, R je poloměr zatáčení a V je dopředná rychlost vozidla.



Obrázek 2.5: Geometrie diferenciálního podvozku[5]

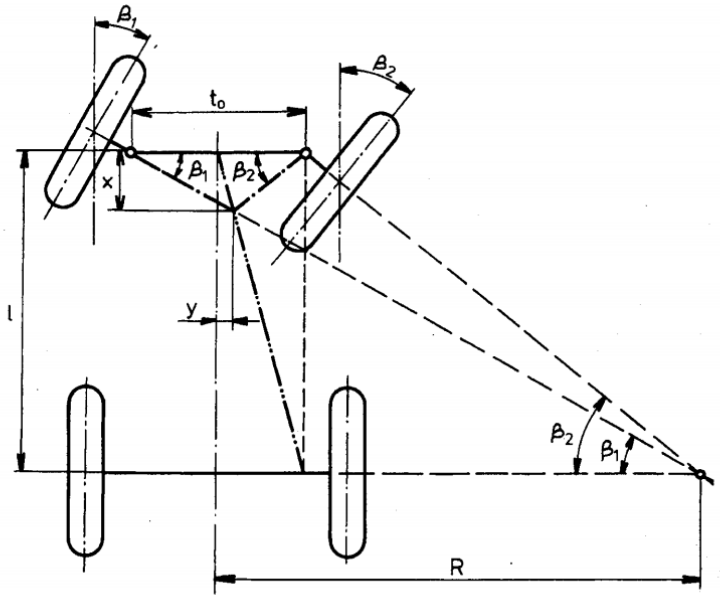


Obrázek 2.6: Příklad diferenciálního podvozku, robot E-puck[6]

[5] převzato z literatury [20]
[6] převzato z: https://commons.wikimedia.org/wiki/File:E-puck-mobile-robot-photo.jpg

Ackermannovo řízení

Ackeramnnovo řízení je běžný typ podvozku využívaný v automobilech [22]. Využívá 4 kola, kde dvě přední jsou řiditelná a zadní kola poháněná. Při zatáčení vozidla je potřeba aby přední kola vozidla zatáčely po kružnici s jiným poloměrem, jinak budou kola vozidla budou podkluzovat. Úhly kol jsou popsány rovnicemi 2.2 a 2.3. Popis proměnných je v obrázku 2.7.



Obrázek 2.7: Geometrie Ackermannova řízení[7]

$$cotg(\beta_1) = \frac{R + \frac{t_0}{2}}{l}, cotg(\beta_2) = \frac{R - \frac{t_0}{2}}{l} \tag{2.2}$$

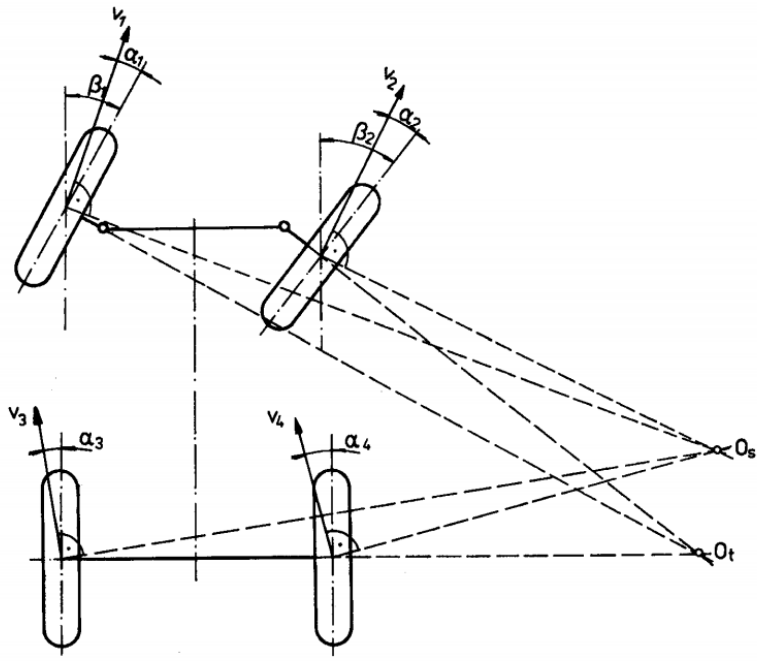
$$cotg(\beta_1) - cotg(\beta_2) = \frac{t_0}{l} \tag{2.3}$$

Pokud úhly kol splňují uvedené rovnice, všechny normály se protnou v jednom bodě a proto bude mít vozidlo jeden bod otáčení a kola proto nebudou podkluzovat.

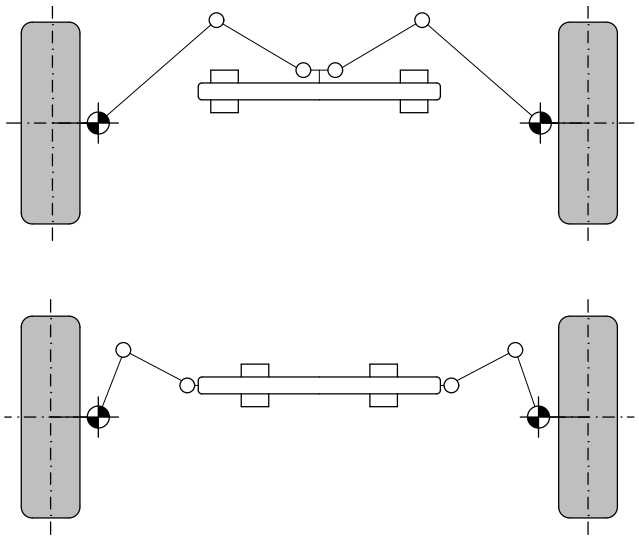
Toto tvrzení platí pouze pokud uvažujeme, že se jedná o ideální kola a zanedbáváme odstředivé síly. Pokud by jsme jej nezanedbávali, normály kol se změní a střed otáčení posune viz obr. 2.8.

Ackermannovo řízení má mnoho implementací. Například řídící lichoběžník, trojúhelník řízení nebo hřebenové řízení viz. 2.9.

[7] převzato z literatury [22]



Obrázek 2.8: Skluz neideálních kol nebo skluz při započtení odstředivých sil[8]



Obrázek 2.9: Ackermannova geometrie s hřebenovým řízením

[8] převzato z literatury [22]

2.3 Vzdušní roboti

Tato kategorie obsahuje velké množství tříd robotů [4], kteří se dokážou často úplně nezávisle pohybovat ve vzduchu a provádět složité úkoly bez jakéhokoliv zásahu operátorů. Díky své konstrukci se vzdušní roboti dokáží dostat do lokací, které jsou často kvůli náročnému terénu pro pozemní roboty nedostupné. Vzdušní roboti se nejčastěji využívají pro mapování, průzkum, kontrolu území nebo přepravu nákladů. Nacházejí uplatnění ve vojenství, kosmonautice, bezpečnostním a dopravním průmyslu. Vzdušné roboty lze rozdělit do dvou obsáhlých kategorií:

- Roboti s pevnými křídly
- Roboti s vrtulemi

(a) robot s pevnými křídly

(b) robot s vrtulemi

Obrázek 2.10: Příklady vzdušných robotů[9]

Roboti s pevnými křídly bývají energeticky nejefektivnější, zatímco roboti s vrtulemi se využívají pro zvýšenou manévrovatelnost a zvýšenou stabilitu při vznášení se na místě. Mezi zástupce okřídlených robotů se řadí například akrobatické letouny, delta křídla nebo kluzáky. Nejběžnějšími zástupci kategorie robotů s vrtulemi jsou konvenční helikoptéry, multikoptéry, koaxiální helikoptéry a roboti se vzduchovými kanály. Toto základní rozdělení je komplikováno relativně velkou skupinou hybridních robotů jako letouny s otočnými vrtulemi nebo tryskami. Dalšími skupinami robotů vymykajících se této klasifikaci jsou roboti inspirovaní přírodou a roboti lehčí než vzduch.

[9] převzato z: https://commons.wikimedia.org/wiki/File:CH-4_UAV.jpeg, https://pixabay.com/photos/uav-drone-aerial-remote-fly-914879/

2.4 Vodní roboti

Vodní roboti jsou roboti pohybující se po vodní hladině nebo pod ní [4]. Jejich existence umožňuje přístup na jinak často těžko dostupné nebo úplně nedostupné lokace.

Obrázek 2.11: Dálkově ovládaný vodní robot[10]

Většina vodních robotických vozidel na trhu je dálkově ovládaných pomocí kabelu. Dálkově ovládaní roboti jsou ovládáni operátorem z mateřského plavidla posíláním kontrolních signálů, energie a přijímají video a jiné informace o stavu vozidla. Roboti bývají běžně vybaveni jednou nebo několika kamerami, světly, sonarem, manipulátorem, robotickou rukou a širokým spektrem vzorkovacích zařízení. Vodní roboti se využívají ve vojenství, průzkumu životního prostředí, vědeckém výzkumu a těžebním průmyslu.

[10] převzato z: https://en.wikipedia.org/wiki/File:Global_Explorer_ROV.jpg

2.5 Současné konstrukce robotických zařízení využívajících RC komponenty

Tato sekce obsahuje příklady robotických konstrukcí využívajících RC serva, jedná se o vybrané příklady demonstrující současný stav dostupných konstrukcí a jejich schopností, nikoli jejich kompletní přehled.

Robot Rapiro

Robot Rapiro je humanoididní robot, jeho konstrukce i software jsou volně dostupné na webu výrobce. Jedná se o typického zástupce výukových robotických sad.

Rapiro je dostupný a jednoduše sestrojitelný robot [10]. Obsahuje 12 modelářských RC serv a základní desku s mikrokontrolérem komptabilním s platformou Arduino. Funkcionalitu robota lze rozšířit pomocí Raspberry Pi.

Obrázek 2.12: Robot Rpiro[11]

Pro ovládání robota je využito 12 RC serv, 2 led diod. Volitelně lze připojit senzor pro měření vzdálenosti a jednodeskový počítač Raspberry Pi s kamerou.

[11] převzato z: https://newatlas.com/rapiro-humanoid-robot-kit-raspberry-pi/28019/#gallery:6

Flexibilní modulární robot GZ-I

Jedná se o nízkonákladového modulárního robota využívajícího RC serva, který je založen na modulárním robotovi Y1.

Modulární roboti jsou obvykle složeni z mnoha poměrně jednoduchých bloků [28], kdy jejich spojování umožňuje přenos mechanických sil, napájení a komunikaci mezi jeho částmi. Výhodou modulárních robotů je možnost jejich využití pro mnoho funkcí, možnost změny jejich účelu a změny konfigurace. Také umožňují rychlé prototypování tak, že je možné rychle zkoušet nové konfigurace.

Obrázek 2.13: Robot GZ-I[12]

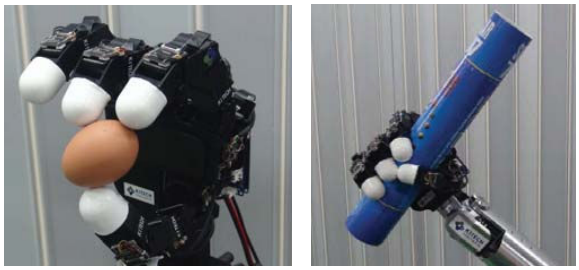
Tento robot se skládá z modulů o velikosti přibližně 80 mm × 50 mm × 50 mm. Každý modul se skládá z 6 mechanických částí RC serva, a elektronického ovladače s dostatečným počtem vstupů a výstupů. Každý modul má 4 strany, které lze využít pro připojování dalších modulů. Toto bylo demonstrováno na pomocí reálných prototypů viz obrázek 2.13. Také byly navrženy další způsoby zapojení jako například čtyřnohý robot a humanoidní robot.

[12] převzato z literatury [28]

Anatomorfická robotická ruka

Jedná se o robotickou ruku s 4 prsty využívající pro svůj pohyb modelářské RC serva.

Tato robotická ruka využívá RC serva jako levnou náhradu za aktuátory využívané v robotické ruce KITECH hand [2]. Při návrhu ruky bylo využito možnosti zpětně pohánět servomotor a byl prezentován způsob řízení kroutivého momentu RC serv.



Obrázek 2.14: Anatomorfická robotická ruka[13]

Robotická ruka má 4 prsty, kde je každý řízen 4 RC servy. Robotická ruka tak má 16 stupňů volnosti. Její velikost je přibližně 1,5 násobek velikosti lidské ruky. Možnost zpětně pohánět servomotor se ukázala jako výhoda, jelikož zlepšuje možnosti úchopu objektu. Aktuátory v robotické ruce byly upraveny tak, aby využívaly komponenty RC serv. Pro kontrolu kroutivého monentu RC serv byl navržen modul omezující maximální proud odebíraný RC servy. Díky kontrole kroutivého momentu bylo docíleno kontroly síly úchopu bez využití senzorů.

[13] převzato z literatury [2]

Robotické rameno Tinkerkit Braccio Robot

Toto robotické rameno je nabízeno společností Arduino AG. Jedná se o malé robotické rameno ovládané pomocí RC serv.

Tinkerkit Braccio je plně funkční robotické rameno ovládané pomocí mikrokontroléru Arduino [19]. Umožňuje mnoho konfigurací včetně přichycení kamery nebo solárního panelu.

Obrázek 2.15: Robotické rameno Tinkerkit Braccio Robot[14]

Pro pohyb robota je využito 6 RC serv poskytujících robotickému ramenu 6 stupňů volnosti. Maximální dosah ramena je 80 cm a maximální zatížení robota ve vzdálenosti 32 cm je 150 g. Toto robotické rameno bylo využito například ke Studii proveditelnosti Robotického sbírání chilli papriček [12].

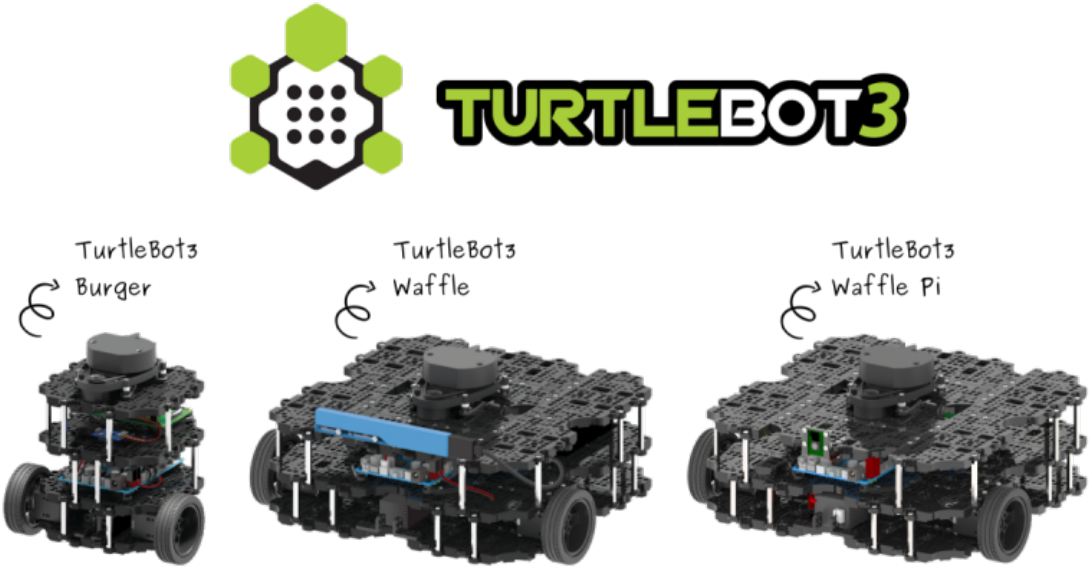
[14] převzato z: https://store.arduino.cc/tinkerkit-braccio-robot

2.6 Současné konstrukce robotických vozidel

Tato sekce obsahuje příklady konstrukcí robotických vozidel, jedná se o vozidla s bezprostředním vztahem k obsahu práce.

TurtleBot3

Jedná se o standardní platformu pro metaoperační systém ROS. Jde o derivát robota Turtle robot řízeného výukovým programovacím jazykem Logo v roce 1967 [21]. TurtleBot3 je již 3. verze tohoto robotického kitu. Jedná se o kolaborační projekt mezi Open Robotics, ROBOTIS a dalšími partnery. Open Robotics řídí vývoj software a ROBOTIS výrobu a distribuci. Tento projekt má volně dostupné zdrojové kódy i hardware.



Obrázek 2.16: Platforma TurtleBot[15]

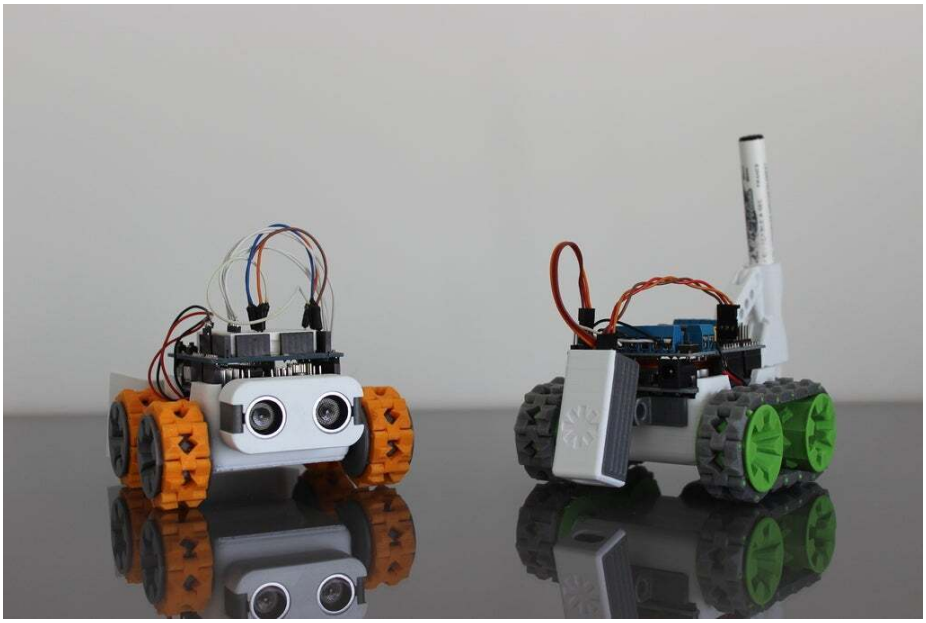
Je to malý, dostupný robot, pro použití ve výuce, vývoji hobby a prototypování. Tato platforma se snaží nabídnout dostupnou platformu bez omezení kvality, funkcionality a rozšířitelnosti. Je nabízen ve 3 základních konfiguracích – TurtleBot3 Burger, TurtleBot3 Waffle a TurtleBot3 Waffle Pi viz obr. 2.16.

TurtleBot3 nabízí technologie jako SLAM (anglicky simultaneous localization and mapping, současná lokalizace a mapování), navigaci a mapování. Tato funkcionalita jej dělá vhodným robotem pro práci v domácnosti. Lze na něj připevnit manipulátory jako OpenMANIPULATOR. Tím lze docílit ještě širšího využití. Může být dálkově ovládáno pomocí počítače, joysticku nebo telefonu s operačním systémem Android.

[15] převzato z: https://www.turtlebot.com/assets/images/turtlebot3_with_logo.png

modulární robotické vozidlo SMARS

SMARS (anglicky Screwless/Screwed Modular Assemblable Robotic System, modulární složitelný model bez šroubů / se šrouby) [18] je jednoduché robotické vozidlo navržené pro výukové účely. Je navrženo tak aby bylo plně 3D tisknutelné a využívalo 'off the shelf" komponenty. Jedná se o komunitní projekt, který se sdružuje okolo platformy Thingiverse a sociálních sítí. Existuje mnoho konfigurací vozidla, které lze využít. Nejčastější jsou kolové vozidla, pásové vozidla a vozidlo s nohami.



Obrázek 2.17: Robotické vozidlo SMARS[16]

Nejběžnější kolová nebo pásová varianta vozidla využívá pro svůj pohyb dvou motorů, jedná se tedy o diferenciální podvozek. Pro ovládání vozidla je využit mikrokontrolér Arduino Uno s rozšířením Arduino Motor Shield obsahující čip L298 pro ovládání motorů. Pro interakci s okolím lze využít mnoho modulů, například sonar, senozor pro sledování čáry, senzor vlhkosti půdy nebo senzor barvy. Po bezdrátové ovládání vozidla lze využít Bluetooth modul.

[16] převzato z: https://www.thingiverse.com/thing:2662828

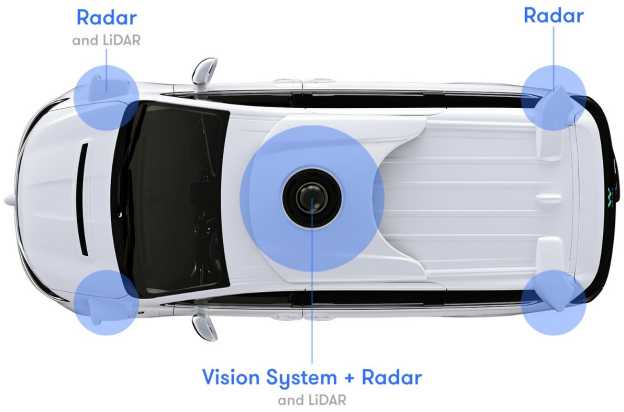
Waymo Pacifica minivan

Jako příklad robotického vozidla využívající velké množství senzorů uvádím robotický automobil společnosti Waymo LLC.

Waymo LLC [24] je americká společnost vyvíjející technologie pro autonomní řízení. Je dceřinou společností Alphabet Inc. Tato společnost provozuje na omezeném území v USA komerční autonomní taxislužbu operující bez záložního řidiče.

Jedno z vozidel využívané společností Waymo je upravený automobil Pacifica minivan [23] vyráběný automobilkou Chrysler. Tento automobil byl poprvé představen na konferenci Automobili-D v Detroitu v únoru 2017.

Obrázek 2.18: Vozidlo Waymo Pacifica minivan[17]



Obrázek 2.19: Senzory vozidla Waymo Pacifica minivan[18]

Toto vozidlo využívá hardware navržený a vyrobený společností Waymo. Pro navigaci v okolí využívá tři různé lidary včetně lidaru pro detekci na krátkou vzdálenost a lidar s douhým dosahem schopným zaměřit objekty na dlouhou vzdálenost. Dalšími senzory je systém vidění kombinující data z více senzorů a několika radarů.

[17] převzato z: https://blog.waymo.com/2019/08/introducing-waymos-suite-of-custom.html
[18] převzato z: https://blog.waymo.com/2019/08/introducing-waymos-suite-of-custom.html

Kapitola 3

Počítačové a ostatní technologie pro robotická vozidla

Pro stavbu robotických vozidel je potřeba mnoho technologií, tato kapitola popisuje vestavěné systémy v robotice, software využívaný pro řízení robotů a další technologie jako 3D tisk. V této kapitole jsou poznatky s bezprostředním vztahem k práci. Tato kapitola nemá encyklopedický charakter a nutně nemusí obsahovat všechny poznatky v dané problematice.

3.1 Vestavěné systémy v robotice

V literatuře [27] se nachází velmi pěkná definice vestavěných systémů.

Vestavěný systém (zabudovaný systém, embedded system) je jednoúčelový počítač, ve kterém je řídicí systém zcela zabudován do zařízení, které ovládá. Na rozdíl od univerzálních počítačů, jako jsou osobní počítače, jsou zabudované systémy většinou specializované, určené pro předem definované činnosti. Vzhledem k tomu, že operační systém tohoto počítače je určen pro konkrétní účel, mohou ho tvůrci systém při návrhu zjednodušit a optimalizovat hlavní aplikaci a tak snížit cenu výrobku. Vestavěné systémy jsou často vyráběny sériově ve velkém množství, takže úspora bývá znásobena velkým počtem vyrobených kusů. Další výraznou výhodou je rychlost a jednoduchost použití.

Jedna z oblastí kde se využívají vestavěné systémy je robotika. Tyto vestavěné systémy dělíme na tři kategorie [8].

- **založené na mikrokontrolérech**: mikrokontrolér je samostatný čip skládající se z CPU, paměťových modulů, možnosti rozšíření pomocí periférií skrz I/O porty a komunikačního rozhraní
- **založené na mikroprocesorech**: tyto systémy se skládají především z CPU. Další součásti systému jsou realizovány samostatnými moduly, jedná se například o komunikační rozhraní, konektivitu k periferním zařízením nebo časovače.
- **založené na systému na čipu** (anglicky system on chip, zkratkou **SoC**). Jedná se o kombinaci mikrokontroléru či mikroprocesoru a dalších periférií v jednom čipu.

Moderní řídící moduly pro robotická vozidla, také nazývané jednodeskové počítače se skládají z CPU a dalších komponent umístěných společně na jedné desce plošných spojů tvořící tak kompaktní počítač.

Literatura [8] uvádí porovnání těchto vestavěných systémů viz tabulka 3.1.

	Mikrokontrolér (MCU)	**Mikroprocesor (MPU)**	**SoC**
OS	Ne	Ano	Může být založen na MCU nebo MPU. Pokud MPU tak se jedná o kompaktní málo náročný systém
Šířka slova	4, 8, 16, 32-bit	16, 32, 64-bit	16, 32, 64-bit
Takt	$\leq$ MHz	Ghz	MHz až GHz
Operační paměť	Často KB, výjmečně MB	512 MB až několik GB	MB až GB
Úložiště	KB až MB (FLASH, EEPROM)	MB až TB (FLASH, SSD, HDD)	MB až TB (FLASH, SSD, HDD)
Cena	Nízká	Vysoká	Vysoká
Příklad	mikrokontroléry Atmel 8051, PIC, série mikrokontrolérů ATMEGA	x86, Raspberry Pi, BeagleBone black	Cypress PSoc, Qualcomm Snapdragon

Tabulka 3.1: Porovnání vestavěných systémů[1]

Hlavními úkoly vestavěných systémů v robotice je komunikovat se **vstupními periférie-riemi** (např. senzory nebo uživatelský vstup), **výstupními periférie-riemi** (např. aktuátory, nebo obrazovka), provádět **řídící algoritmy** nebo počítat **výpočetní modely** a komunikovat s **jinými periférie-riemi** (např. komunikační rozhraní, sběrnice, síťové rozhraní atd.).

Robotická vozidla často využívají jak mikrokontroléry, tak mikroprocesory. Tato kombinace umožňuje využít výhody obou systémů, kdy mikrokontroléry vynikají především pro specifickou obsluhu vyžadující přesné časování, jako obsluha periférií. To je důsledek běhu bez operačního systému plánujícího úkoly. Zatímco mikroprocesory zvládají běh více procesů současně a také jsou vhodnější pro běh výpočetně náročných výpočtů jako zpracování obrazu nebo trénovaní datových modelů.

Dále literatura [8] uvádí populární řídící moduly:

- Arduino Mega
- STM32
- ESP8266
- Tinkerboard S
- BeagleBone Black
- Raspberry Pi
- Jetson TX2
- Jetson Nano

[1] převzato z literatury [8]

Řízení robotických komponentů

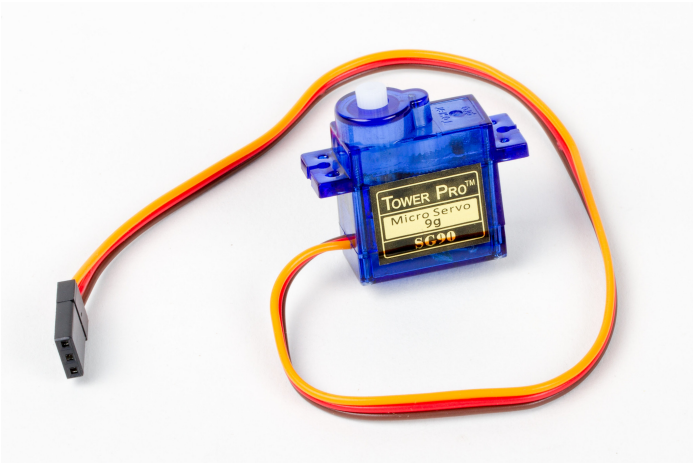
Pro ovládání robotů je použito jednoduchých modulů a spínačů, integrovaných obvodů případně je elektronika ovládána přes sběrnice např. USB, I2C, nebo UART [6]. Některé elektronické komponenty je potřeba ovládat pomocí logických signálů. Pro tyto signály je kritické časování. Pro dosažení přesného časování se využívá hardware časovačů, výstupních periférií, mikrokontrolérů, nebo dedikovaných obvodů zajišťující kritické časování nezávisle na řídícím systému. Pro dosažení časování blížícího se těmto komponentům je využíváno počítače s RTOS (anglicky Real Time Operating System, operační systém reálného času).

Robotická vozidla pak mohou být dálkově ovládaná nebo úplně autonomní, kdy se samostatně snaží splnit předem zadaný cíl.

Instrukce a zpětná odezva vozidla může být přenášena jak drátově, tak bezdrátově. Drátový přenos se často využívá v prostředích, kde by byla bezdrátová komunikace náročná jako prostředí s velkým rušením, nebo útlumem signálu. Bezdrátovou komunikaci je možno rozdělit na analogovou a digitální. S vývojem techniky se čím dál tím více využívá digitálního přenosu informací díky vyšší odolnosti proti rušení, vyšší přenosové rychlosti a vyšší bezpečnosti.

3.2 Modelářská RC serva

RC serva navržené pro dálkově ovládané modely vozidel a letadel jsou aktuátory běžně využívané při tvorbě robotických modelů [3]. Také slouží jako aktuátory pro mnoho komerčně dostupných robotických kitů. RC serva také mohou sloužit jako levná alternativa pro studentské projekty a laboratorní pokusy.



Obrázek 3.1: Servo SG90[2]

Modelářská RC serva jsou využívány k ovládání dálkově ovládaných modelů vozidel a letadel. Typicky nabízí rotační aktuaci v rozmezí 0 až 180° bez zpětné vazby. Tyto serva jsou často využívány jako aktuátory v ručně vyrobených robotech i komerčně dostupných robotických kitech. Tyto serva lze ovládat pomocí pulzně modulovaného signálu. Serva lze jednoduše konvertovat z přibližně 180° rozsahu na kontinuální rotační pohyb. RC serva jsou nabízeny v mnoha velikostech, které nabízejí dostatečnou rychlost a kroutivý moment pro

[2]převzato z: https://www.flickr.com/photos/oskay/22731304279

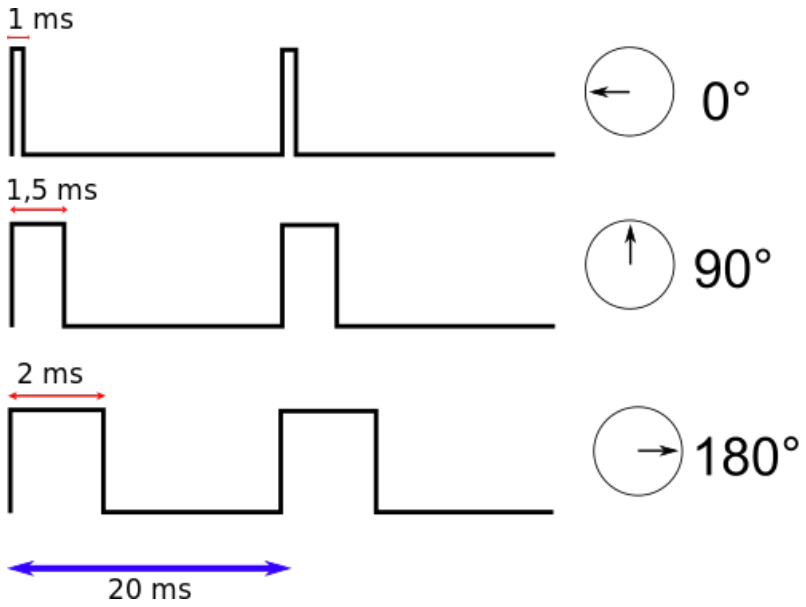
velké množství aplikací v robotice. Další výhodou modelářských RC serv je jejich cenová dostupnost.

Konstrukce

Modelářská RC serva se typicky skládají ze čtyř hlavních částí [6] integrovaných do jednoho kompaktního modulu.

- motor
- převodovka
- aktuátor
- řídící elektronika

Jsou využívány především kompaktní stejnosměrné kartáčové motory, které využívají převodovku pro zvýšení kroutivého momentu. Převodovky jsou konstruovány z plastových materiálů jako např. nylon nebo také z kovů a slitin v servech s vyšším kroutivým momentem. Převodovka je napojena na otočný aktuátor a potenciometr poskytující zpětnou vazbu řídící elektronice. Typickým zástupcem je např. mikro servo SG90 viz obr. 3.1.



Obrázek 3.2: Typický řídící signál modelářského RC serva

Modelářská RC serva jsou řízeny pomocí pulzně šířkově modulovaného signálu. Signál má typicky frekvenci 50 Hz. Pulz délky 1 ms značí minimální výchylku a pulz délky 2 ms maximální výchylku viz obr. 3.2. Běžně serva využívají jednoduché analogové obvody založené na RC článku a časovači. To ale způsobuje při malých změnách výchylky vysokou odezvu a malý kroutivý moment. Proto se pro aplikace vyžadující rychlé změny výchylky s velkým kroutivým momentem využívají serva s digitálním zpracováním signálu, které jsou interně řízeny šířkově pulzně modulovanými signály o frekvenci 300 až 350 Hz, přitom poskytují stejné ovládací rozhraní jako analogové serva.

Ovládání modelářských RC serv počítačem

Pro ovládání RC modelářských serv je potřeba generovat šířkově pulzně modulovaný signál [6] (dále PWM, anglicky pulse width modulation). Takový signál je možné generovat pomocí specializovaných periferií, přerušení nebo DMA řadiče (anglicky Direct Memory Access, tj. přímý přístup do paměti). Často ale bývá nejjednodušší využít dostupných modulů schopných ovládat velké množství serv. Takovéto moduly lze připojit například pomocí sběrnice I2C.

Dále se tato sekce zabývá možnosti ovládání modelářských RC serv na platformě Raspberry Pi.

Pro řízení serv je potřeba generovat poměrně přesný PWM signál. Při generování signálu pomocí software nelze zaručit přesné časování. Přesný PWM signál lze proto generovat pouze pomocí hardware komponent. Raspberry Pi obsahuje periférie schopné generovat PWM signál – PWM a PCM, obě jsou používány pro tvorbu zvuku [13]. Pro tvorbu PWM signálu lze tak využít jeden výstupní pin a ponechat si možnost zvukového výstupu, nebo využít oba generátory a nevyužívat zvukový výstup. Pro více PWM výstupů generovaných pomocí hardware periferií lze použít DMA řadič, toho využívá knihovna pigpio, kde ale také nelze při využití hardwarově generovaného PWM využívat zvukový výstup, protože jedna periferie je využívána pro časování DMA přenosů a druhá pro generování PWM signálu. Poslední možností je využít externí modul pro generování PWM signálu. Na trhu existuje několik různých modulů pro kontrolu RC serv pomocí USB nebo např. I2C portů např. [1], tyto externí moduly nabízí generování několika nezávislých PWM kanálů.

3.3 Software pro tvorbu robotických vozidel

Pro robotická vozidla existuje několik operačních systémů, knihoven a sad nástrojů, které ulehčují implementaci řešení, obsahují ovladače pro senzory a implementují různé plánovací, navigační a mapující algoritmy. Mezi nejvyužívanější patří:

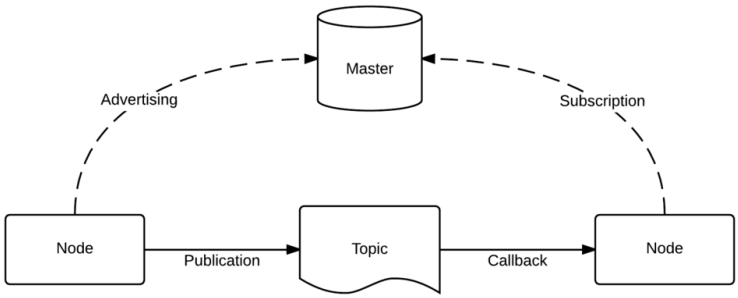
- ROS
- The Player Project

Dále se v této kapitole nachází jejich popis.

ROS

ROS [16] (anglicky The Robot Operating System, operační systém pro roboty) je kolekce nástrojů, knihoven a konvencí, které mají za úkol zjednodušit tvorbu složitého a robustního chování robotů pro velké množství robotických platforem.

Jedná se o open-source meta-operační systém [17]. Poskytuje služby, které lze očekávat od operačního systému včetně abstrakce od hardware, nízkoúrovňovou kontrolu, implementuje běžně používaných funkcí jako předávání zpráv mezi procesy a zprávu balíčků. Také obsahuje nástroje a knihovny pro získávání, sestavování, tvorbu a běh kódu napříč mnoha platformami. Při běhu je využíváno grafové struktury, která může běžet na několika zařízeních. Je implementováno několik stylů komunikací obsahujících synchronní komunikaci pomocí služeb, asynchronní komunikaci pomocí témat a ukládání dat na parametrické servery.



Obrázek 3.3: Témata v ROS[3]

ROS není framework běžící v reálném čase, i když je možné ROS integrovat s kódem běžícím v reálném čase. Cílem ROS je podporovat znovuvyužití kódu při výzkumu a vývoji v robotice. To je umožněno díky frameworku distribuovaných procesů (uzly), který umožňuje nezávislý vývoj samostatných modulů a zajištění jejich současného běhu. Tyto procesy mohou být spojeny do balíčků, které mohou být jednoduše distribuovány a sdíleny. ROS je nezávislý na programovacím jazyce a umožňuje vývoj v jazycích Python, C++, Lisp a má experimentální moduly v jazycích Java a Lua.

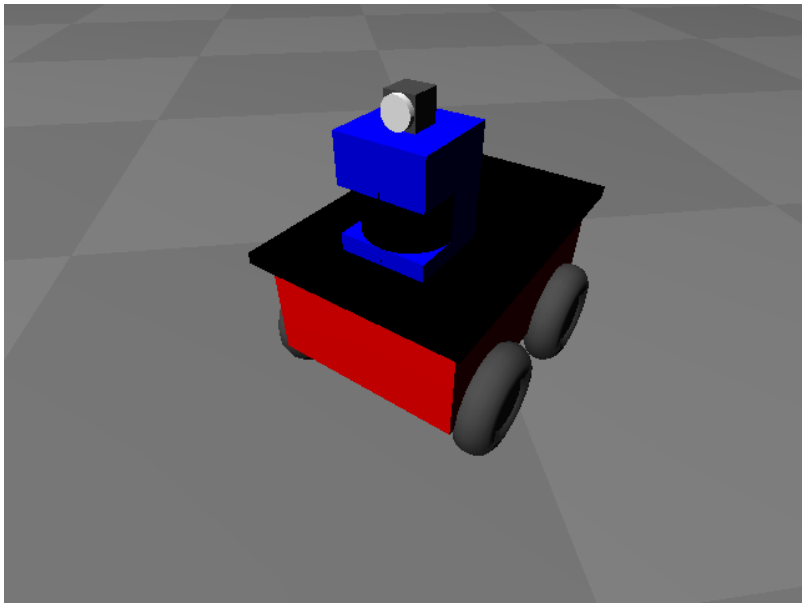
[3] převzato z: https://commons.wikimedia.org/wiki/File:ROS-master-node-topic.png

The Player Project

The Player Poject [5], software vydávaný pod GNU licencí, je jedeno z nejpopulárnějších open-source robotických rozhraní využívaných k výzkumu a na vysokých školách.

Jedná se o server umožňující kontrolu nad robotickým vozidlem, funguje jako abstrakční vrstva, která umožňuje jednoduché ovládání hardware. Server běžící na robotickém vozidle poskytuje jednoduché rozhraní pro přístup k senzorům a aktuátorům přes IP síť. K robotickému zařízení se připojuje pomocí TCP socketu, přes který je možné číst data ze senzorů, ovládat aktuátory a za běhu konfigurovat zařízení.

Player podporuje velké množství robotického hardware. Je také možné implementovat podporu pro nový hardware.



Obrázek 3.4: Simulační prostředí The Player Project[4]

Player je navržen jako nezávislý na programovacím jazyce [14]. Klientský program může běžet na libovolném zařízení s připojením k robotovi. Může být vytvořen v libovolném programovacím jazyce podporujícím TCP sockety. V současnosti existují nástroje pro klientské aplikace v programovacích jazycích C++, Tcl, Java a Python.

Nedílnou součástí The Player Project je Stage [9]. Stage simuluje populaci robotických vozidel, senzorů a objekty v prostředí. Umožňuje tak rychlý vývoj software, který řídí roboty a umožňuje provádět robotické experimenty bez přístupu k fyzickému hardware nebo prostředí.

[4] převzato z: https://commons.wikimedia.org/wiki/File:Pioneer_3-AT_in_Gazebo.png

3.4 Automatizace robotických vozidel

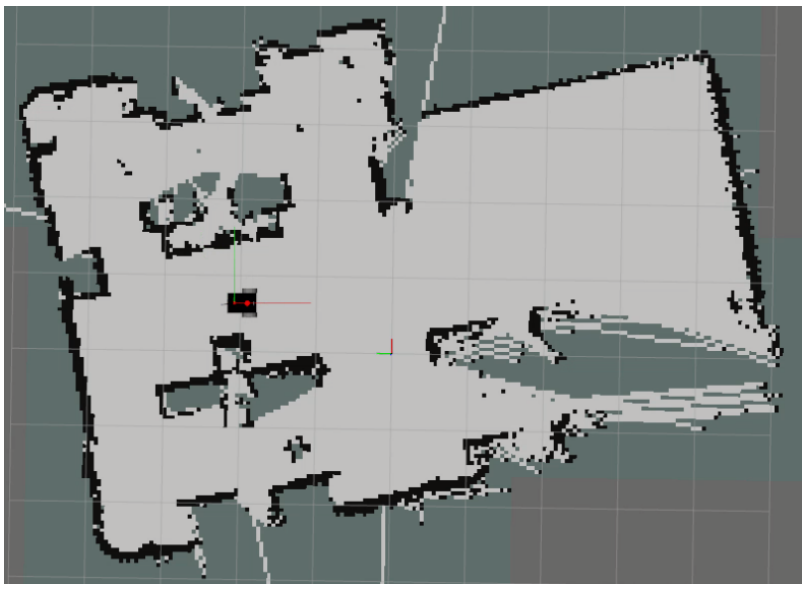
Po robotech se často požaduje aby, svou činnost prováděli autonomně bez dozoru operátora [4]. Proto je potřeba, aby robotická vozidla zvládaly navigaci, mapování okolí a plánování trasy nebo pohyb v předem známém prostředí.
Mezi základní problémy automatizace robotických vozidel patří:

- lokalizace
- mapování
- plánování cesty

Jelikož je pro tvorbu mapy v neznámém prostředí potřeba znát polohu vozidla, jsou to navzájem propojené problémy, které musí být řešeny současně. Po sestavení mapy nebo v průběhu je cílem robota plánovat trasu co nejoptimálněji, tak aby dosáhl svého cíle a vyhl se všem překážkám.

Současné mapování a lokalizace

Jeden ze známých problémů v robotice se nazývá SLAM (anglicky Simultaneous Localization and Mapping, současné mapování a lokalizace), kdy je potřeba vytvářet a aktualizovat mapu neznámého prostředí a zároveň udržovat informaci o poloze vozidla v prostředí.



Obrázek 3.5: Mapa prostředí získaná pomocí SLAM[5]

Řešení tohoto problému je často označováno jako jedeno z nejdůležitějších pro tvorbu plně autonomních vozidel. V současnosti existují metody pro mapování prostředí, které je většinově statické, strukturované a omezené velikosti. Populární metody na řešení tohoto problému využívají filtrování částic jako Kalmanův filtr a různé grafové algoritmy např. FastSLAM.

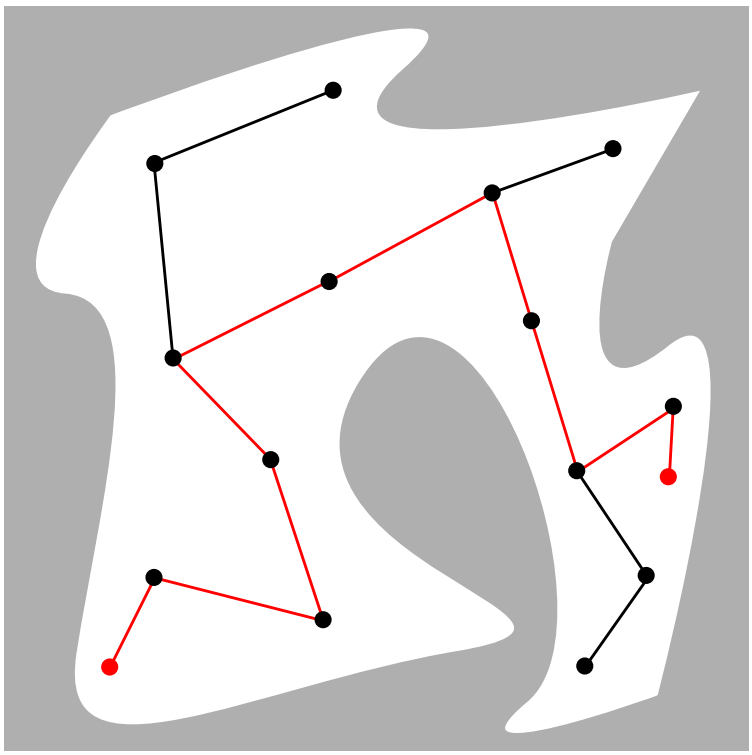
[5]převzato z: https://msadowski.github.io/iris-lama-slam-with-ros/

Plánování trasy a vyhýbání se překážkám

Základním cílem je naplánovat trasu ze začáteční do koncové pozice bez kolize s okolím [4]. Přes to, že je tento problém relativně jednoduchý, je výpočetně náročný. Tento problém je úzce propojený s ovládáním robotických vozidel a jejich geometrií. Velké množství robotů totiž neumožňuje pohyb do všech stran (neholomické vozidla). Tyto limitace je potřeba vzít v úvahu při plánování trasy. Toto dále komplikuje tento problém.

Plánovací algoritmy lze rozdělit do následujících kategorií:

- **plánování pomocí vzorkování** tyto algoritmy rozdělí trasu na podčásti kde kontrolují kolize s prostředím, pro hledání trasy často využívají grafových algoritmů
- **kombinatorické trasy** využívají trasy vytvořené za určitých podmínek, například trasy, které jsou nejdále od všech překážek, nebo trasy mezi vrcholy překážek. Z těchto tras je poté vytvořena vhodná cesta.
- **potenciální pole** využívá gradientního sestupu potenciální funkce, navádějící pohybující-se vozidlo k cíli a od překážek v kombinaci s náhodnými procházkami.



Obrázek 3.6: Plánování pomocí vzorkování[6]

Většina plánovacích algoritmů neplánuje optimální trasy, při plánování to není cílem a často ani nelze jednoduše určit, protože nejkratší trasy nebo trasy s největší vzdáleností od překážek nemusí představovat optimální řešení.

[6] převzato z: https://en.wikipedia.org/wiki/Motion_planning#/media/File:Motion_planning_configuration_space_road_map_path.svg

3.5 3D tisk

3D tisk [25] nebo také aditivní výroba je metoda pro výrobu trojrozměrných objektů z CAD modelu nebo trojrozměrného modelu. Jako 3D tisk je možné označit množství procesů, které jsou ukládány nebo spojovány a pod počítačovou kontrolou vytváří trojrozměrný předmět. Materiál je spojován do celku nejčastěji vrstvu po vrstvě.

Tato technologie výroby se nejčastěji využívá pro prototypování a malosériovou výrobu pro nízké náklady při výrobě malých sérií nebo jednotlivých dílů, jelikož není potřeba pro výrobu připravovat často drahé formy nebo nástroje.

3D tisk je v současnosti velmi dostupnou technologií výroby [15], především díky projektu RepRap, tím umožňuje jednoduchou a levnou tvorbu dílů jednotlivcům nebo univerzitám.

Dále se kapitola zaměřuje na dostupné technologie 3D tisku. Mezi nejdostupněší technologie 3D tisku se řadí:

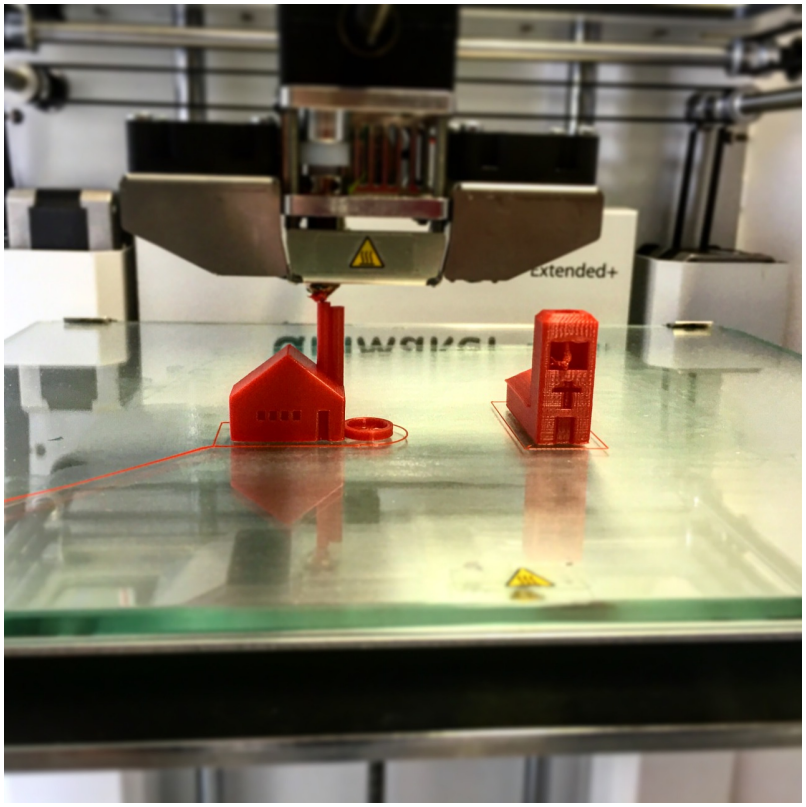
- **FFF** (anglicky Fused Filament Fabrication, tj. výroba spojováním filamentu) známá také jako **FDM** (anglicky Fused Deposition Modeling, modelování depozičním spojováním)
- **SLA** (anglicky Stereo Litography, stereolitografie)

Obrázek 3.7: Předmět vyrobený metodou FFF, tzv. Benchy[7]

[7] převzato z: https://commons.wikimedia.org/wiki/File:3D-printed_3DBenchy_by_Creative_Tools.jpg

Technologie FDM

Technologie FDM [11] využívá materiálu (filament), který postupně po vrstvách ukládá a tím tvoří model. Nejčastěji používané materiály pro 3D tisk jsou termoplasty, především PLA, PETG, ABS, ASA, Nylon nebo flexibilní TPU. Trh s dostupnými materiály se neustále rozšiřuje a umožňuje tak různé tvorbu předmětů pro různé aplikace. 3D tisk pomocí termoplastů probíhá tak, že filament je tlačen pomocí motoru přes horkou trysku, kde se filament taví. Tryska je připevněna na kinematický systém který postupně vrstvu po vrstvě filament ukládá na podložku, která je často nahřívaná. Běžná velikost předmětů, které je možné zhotovit na FDM 3D tiskárně se pohybuje od 15×15×15 cm do 30×30×30 cm, ale existují i tiskárny schopné vyrobit větší předměty.



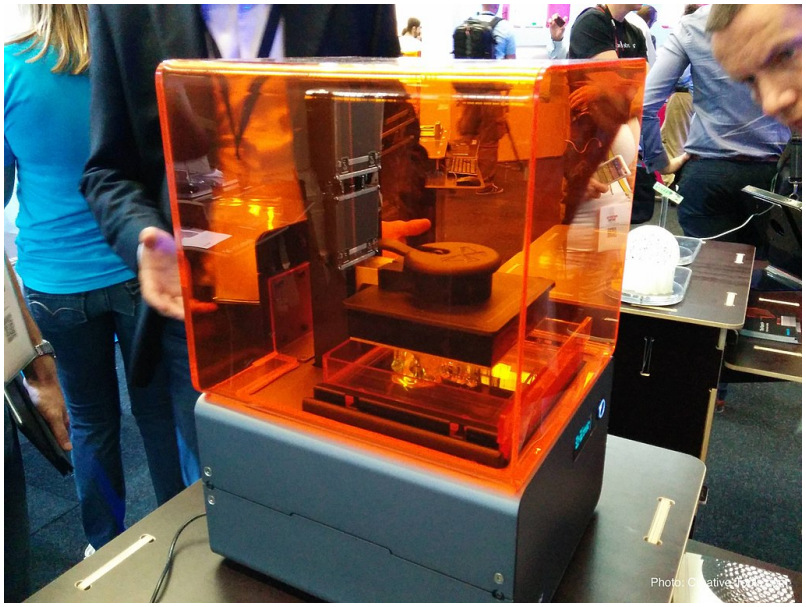
Obrázek 3.8: FDM tiskárna[8]

Rozlišení tisku je limitováno průměrem trysky, která má nejčastěji vnitřní průměr 0,4 mm a výškou vrstvy běžně 0,2 mm. Dalšími omezeními technologie je nemožnost tvorby libovolné geometrie, kdy není možné, aby začínala nová vrstva bez podpory předchozí vrstvy (tzv. ostrov) a stěny měly převis typicky větší než 45°. Toto bývá řešeno podporami. To jsou struktury které umožní tvorbu projektu a po tisku je potřeba mechanicky nebo chemicky, v případě podpor z rozpustitelného materiálu, odstranit. Další omezení je určeno orientací předmětu při tisku, jelikož mezi vrstvami termoplastu předmět nevykazuje takovou pevnost jako samostatný termoplast.

[8] převzato z: https://pixabay.com/photos/printer-3d-making-pressure-1455165/

Technologie SLA

Technologie SLA [11] využívá materiálu, který selektivně vrstvu po vrstvě vytvrzuje světlocitlivý polymer. Pro vytvrzování je využíváno především laseru, nebo displejů a UV světel – mSLA (anglicky Mask Stereolitography, stereolitografie s maskou). 3D tisk typicky probíhá tak, že je fotopolymer umístěn v nádobě, na dně nádoby je umístěn průzor, který umožňuje průnik světla. K průzoru se z hora přisune plocha, na kterou se bude objekt vytvrzovat. Poté se postupně ozařují vrstvy, kdy se s každou vrstvou tisková plocha zvedne.



Obrázek 3.9: SLA tiskárna[9]

Tiskové polymery podobně jako filamenty pro FDM tiskárny mohou mít různé vlastnosti určené aditivy, ty mohou zvyšovat tuhost nebo houževnatost polymeru. Rozlišení tisku je určeno rozlišením displeje nebo přesností laseru, kdy např. velikost pixelu se pohybuje v desítkách µm a výška vrstvy mezi 25 a 100 µm. Velikost tištěného předmětu bývá typicky menší než u FDM tiskáren. Při tisku předmětu podobně jako u FDM tiskáren modely nemohou obsahovat vrstvy bez předchozí podpory, což je také řešeno podporami. Další omezení se vztahuje na veliké změny plochy mezi vrstvami, které mohou způsobit až úplné selhání tisku modelu.

[9] převzato z: https://commons.wikimedia.org/wiki/File:3D_Printshow_2014_London_-_Formlabs_Form_1_SLA_3D_printer_v01_(15150505392).jpg

Kapitola 4

Zhodnocení současného stavu a plán práce

Tato kapitola obsahuje zhodnocení současného stavu ovládání RC serv pomocí počítače a robotických vozidel, jejich ovládání, zhodnocení dostupných technologií 3D tisku, upřesnění specifikace požadavků a plán práce.

4.1 Zhodnocení dosavadního stavu robotických vozidel s RC komponenty

RC komponenty jsou v robotice využívány jako levná alternativa k běžně využívaným aktuátorům. Využívají se v komerčně dostupných kitech využívaných především pro hobby a výukové účely. Ukázalo se, že pro tento účel mohou být vhodnou náhradou, jak se ukázalo při jejich využití při sestavovaní robotických konstrukcí. Ukázalo se, že v případě využití ovladačů kontrolujících proud RC serv lze kontrolovat i jejich kroutivý moment.

Pro ovládání RC serv pomocí počítače je možné v současnosti využít poměrně velké množství řešení. Dále budu porovnávat řešení pro jednodeskový počítač Raspberry Pi 3B+.

Zde lze vybírat z několika variant. Pokud je využito do dvou serv, je nejjednodušší využít integrované periferie pro tvorbu PWM signálu. Pokud je potřeba více serv, je potřeba zvážit, jestli bude potřeba využít zvukový výstup jednodeskového počítače. Pokud ne, je možné využít např. knihovnu Pigpio. Pokud je potřeba ovládat velké množství serv nebo je potřeba využít gpio piny pro jiné účely, je nejjednodušší využít externí ovladače serv připojené přes USB nebo I2C sběrnice.

V současnosti existuje nepřeberné množství konfigurací robotických vozidel, nejčastěji využívané konfigurace využívají diferenciální řízení pro jednoduchost řízení a odhadu polohy. Takovéto konfigurace ale také často nevykazují velikou mobilitu nebo schopnosti překonávat překážky. Ostatní konfigurace nabízí jiné výhody, to ale za cenu větší mechanické složitosti nebo vyššího počtu motorů nebo serv.

Existující robotická vozidla obsahují velké množství kategorií a to od výukových sad po plně autonomní vozidla. V kategorii výukových sad převažují robotická vozidla s malým množstvím senzorů a s vestavěnými systémy s nízkým výpočetním výkonem sloužící spíše jako pomůcka pro výuku programování vestavěných systémů. Nikoli plnohodnotné robotické vozidla. Na druhé straně spektra se objevují první plně autonomní silniční vozidla schopné provozu po zmapovaných silnicích v běžném provozu.

Pro ovládání robotických vozidel se používá různý software. Nejvýznamnější jsou ROS a The Player Project. Z těchto dvou systémů se ROS používá především pro implementační práce a The Player Project více pro simulace a teoretické práce.

V technologii 3D tisku je také možno vybírat mezi dvěma významnými technologiemi, FFF a SLA. SLA je především vhodná pro přesnou reprodukci modelů a detailní modely, není ale již tak vhodná pro mechanické aplikace vzhledem k mechanickým vlastnostem využívaných fotopolymerů. Na druhé straně spektra je technologie FFF, která nabízí nižší rozlišení, ale větší nabídku materiálů včetně mnoha konstrukčních. Tato technologie je dostupnější a rozšířenější než SLA.

4.2 Specifikace požadavků

Po prozkoumání současných řešení a jejich schopností jsem se rozhodl v souladu se zadáním pro návrh robotického vozidla využívající RC komponenty. Mým cílem bylo pokusit se navrhnout a sestrojit robotické vozidlo pouze pomocí "off the shelf" komponent a 3D tisku, využít netradiční podvozek a zpřístupnit robotické vozidla pro studenty mé fakulty. Na základě osobních zkušeností s dálkově ovládanými vozidly, 3D tiskem a metaoperačním systémem ROS a možností prohloubení zkušeností v oblasti návrhu, konstrukce, automatizace vozidel s Ackermannovým řízením jsem se rozhodl pro následující parametry:

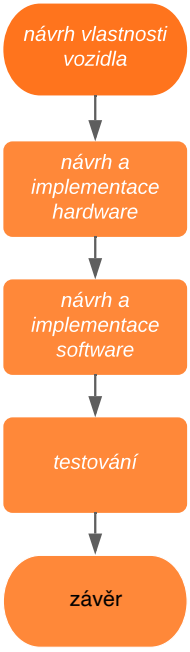
- čtyřkolové vozidlo
- aproximace Ackermannova řízení hřebenovým řízením
- využití "off the shelf" komponent
- využití 3D tisku
- jednoduchost sestavení
- využití lidaru
- výdrž baterie na jedno nabití minimálně 1h
- ovládání pomocí teleoperace a autonomní navigace
- cenová dostupnost
- využití jednodeskového počítače Raspberry Pi 3B+
- využití ROS

Robotické vozidlo s následující specifikací by mělo být schopné pohybu po rovném povrchu, bezdrátového ovládání pomocí počítače nebo mobilního telefonu, a teleoperace a autonomního navigování.

4.3 Plán práce

Na základě zvolených parametrů byl vytvořen plán práce, který zněl:

"Vyber vhodné senzory a řídící elektroniku, software a způsob napájení. Při výběru ber ohled na dostupnost komponent a jednoduchost zapojení. Na základě vybraných komponentů navrhni strukturu vozidla ideálně tak, aby bylo jednoduše sestavitelné a díly tištěné na 3D tiskárně nepotřebovaly speciální materiály nebo manuální úpravy. Sestav model vozidla, osaď komponent a testuj vlastností vozidla. Navrhni a implementuj aplikace demonstrující vlastnosti robotické platformy. Testuj dosažených vlastností robotického vozidla a proveď jejich demonstraci. Zhodnoť a diskutuj dosažené výsledky."



Obrázek 4.1: Plán práce

Kapitola 5

Popis vlastní práce

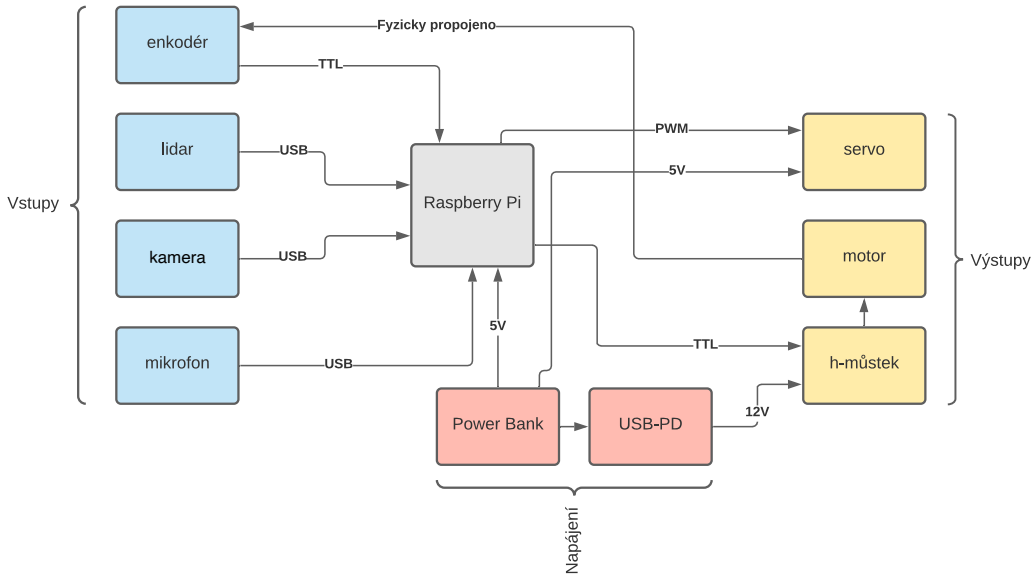
Tato kapitola popisuje návrh robotického vozidla, výběr potřebné elektroniky, senzorů, řídících systémů, jeho konstrukci a demonstrační aplikace. Dále kapitola obsahuje testování a hodnocení robotického vozidla.

5.1 Návrh vozidla

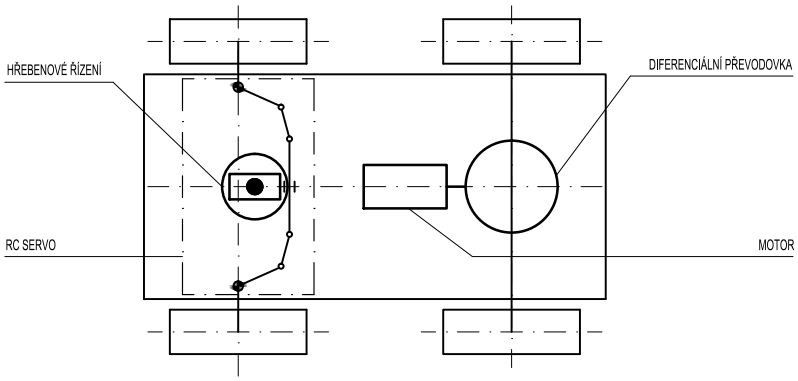
Vozidlo bylo koncipováno jako čtyřkolové vozidlo s konfigurací dvou spojených poháněných kol a dvou řiditelných kol pomocí Ackermannova řízení. Pro účely napájení byla zvolena záložní baterie (anglicky powerbank, záložní zdroj energie, využíván především pro dobíjení přenosné elektroniky skrz USB port). Jedná se o velmi dostupný zdroj energie, který je velmi rozšířený. Tímto odpadá nutnost navrhovat nabíjecí obvody pro vestavěné baterie, které často zahrnuje pájení ochranných obvodů nebo využití nepraktických jednorázových baterií. Pro ovládání motoru byl zvolen H-můstek a servo ovládající kola vozidla je řízeno pomocí vestavěného PWM generátoru. Pro účely demonstrace využití vozidla byla přidána kamera a mikrofon.

Výběr komponent

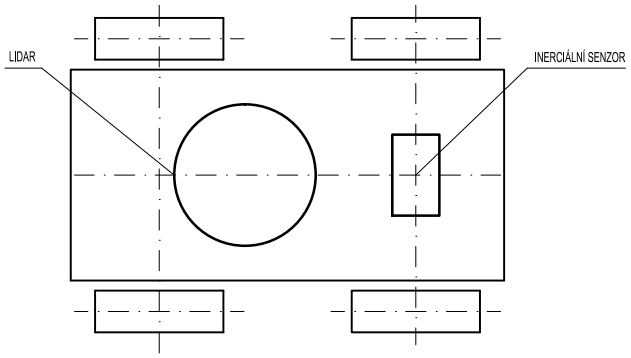
Z uvedených parametrů vozidla, hmotnosti daných komponentů a jejich velikosti byla odhadnuta celková velikost a hmotnost vozidla. Poté byl z dostupných motorů vybrán stejnosměrný motor s převodovkou a enkodérem, tento motor při napájecím napětí vykazuje dle specifikací točivý moment 16,671 $\mathrm{N\,cm^{-1}}$ při napětí 1,3 A. Proto je potřeba zajistit 12V napájení pro motory. To je zařízeno výběrem power banky s třemi USB výstupy, kde alespoň jeden port podporuje standard USB-PD (USB Power Delivery) a napětí minimálně 12V. Pak lze motor připojit pomocí power delivery modulu. Raspberry Pi je poté napájeno z druhého konektoru power banky a třetí port je využit pro napájení serva. Senzory jsou napájeny z 5V pinu Raspberry Pi a data jsou přenášena pomocí převodníku úrovní a GPIO pinů.



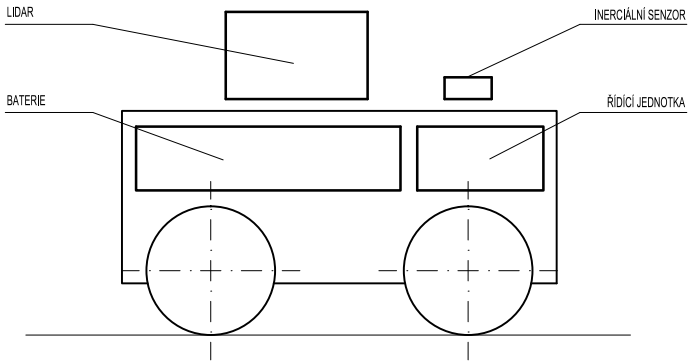
Obrázek 5.1: Navržená architektura robota



(a) Koncepční návrh podvozku vozidla



(b) Koncepční návrh vozidla, pohled shora



(c) Koncepční návrh vozidla, pohled zboku

Obrázek 5.2: Koncepční návrh vozidla

Návrh Ackermannova řízení

Pro vozidlo bylo potřeba navrhnout hřebenové řízení Ackermannova řízení pro vlastní geometrii. Pro určení parametrů jsem naprogramoval skript aproximující optimální řídící úhly pomocí optimalizačního algoritmu.

Jedná se o problém s prohledáváním stavového prostoru s lokálními minimy, kde je cílem nalézt řešení nejvíce se blížící optimálnímu. Předpokládejme symetrické hřebenové řízení, které je celkově určeno 7 parametry viz obr. 5.3:

- rozchodem **r**
- vzdáleností kola od bodu otáčení **v**
- délkou první části spoje s hřebenem $\boldsymbol{l_1}$
- délkou druhé části spoje s hřebenem $\boldsymbol{l_2}$
- úhlem mezi osou kola a druhé části spoje ($\boldsymbol{l_2}$) s hřebenem $\boldsymbol{\alpha}$
- vzdáleností hřebene od nápravy **d**
- délkou hřebene **h**

Pro kontrolu vůči ideálním parametrům je potřeba znát také vzdálenost náprav **n**. Parametry rozchod, vzdálenost kola od bodu otáčení a vzdálenost náprav jsou určeny velikostí vozidla a jsou proto konstantní. Vzdálenost hřebenu od přední osy lze z konstrukčního hlediska určit jako konstantu a zmenšit tak prohledávaný prostor. Pro omezení prohledávaného prostoru lze dále stanovit následující podmínku: v neutrální poloze jsou osy předních kol rovnoběžné s zadní nápravou. Tím lze eliminovat další parametr, protože pro splnění této podmínky existuje pouze jedno řešení, uvažujeme-li, že chceme docílit co nejkratší délky hřebene.

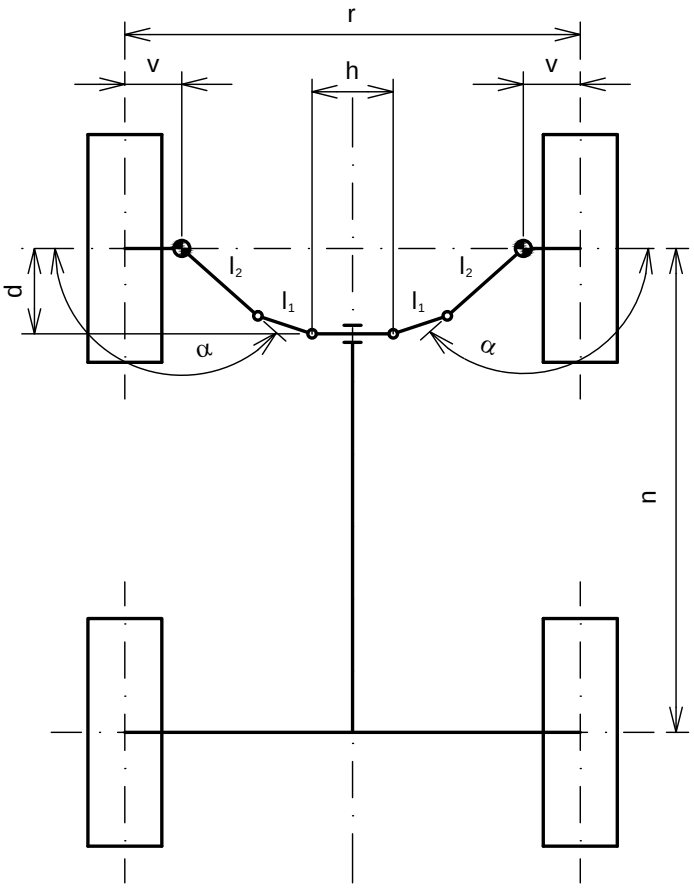
Jako optimalizační algoritmus byl využit algoritmus globálního prohledávání SHGO[7] (Anglicky simplicial homology global optimisation, globální optimalizace zjednodušené homologie), který umožňuje prohledávat stavový prostor s lokálními extrémy.

Pro porovnávání s optimální geometrií byl zvolen průměrný rozdíl druhých mocnin chyby geometrie pro řadu úhlů natočení kol. Pro určení chyby byl spočítán ideální úhel natočení levého kola pro natočení pravého kola. Ten byl porovnán s úhlem natočení kola hřebenového řízení s danými parametry. Byly porovnávány úhly od 0° do požadovaného úhlu $\boldsymbol{\varphi}$ zatočení. Záporné úhly není potřeba porovnávat díky předpokládané symetrii hřebenového řízení.

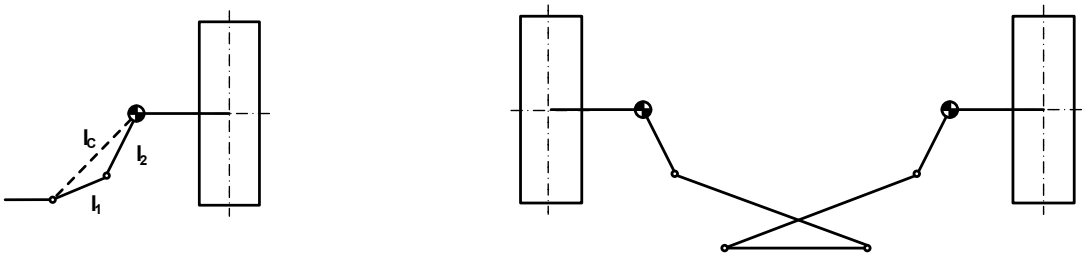
Pro výpočet úhlů natočení kol ideální geometrie byly využity rovnice Ackermannova řízení 2.2 a 2.3.

Pro výpočet úhlů kol hřebenového řízení je nejprve potřeba zkontrolovat, jestli je daná geometrie platná, to je provedeno ověřením řady podmínek tak, aby geometrie splňovala trojúhelníkové nerovnosti trojúhelníku se stranami l_1, l_2 a l_c pro všechny úhly natočení kol viz obr. 5.4(a) a zamezit křížení spojů viz obr. 5.4(b) z konstrukčních důvodů. Poté je postupně počítána poloha samostatných spojů zprava doleva. Tímto je spočítán úhel natočení levého kola v závislosti na úhlu kola pravého.

Toto je provedeno pro dané úhly a spočítána průměrná hodnota rozdílu vůči ideálnímu natočení pro dané úhly kola. Byly voleny úhly od 0 od φ s rozestupy **s** stupňů.



Obrázek 5.3: Ackermannova geometrie a popis parametrů



(a) Trojúhelníková nerovnost hřebenového řízení

(b) Překřížení spojů hřebenového řízení

Obrázek 5.4: Podmínky pro platnost geometrie hřebenového řízení

Tento algoritmus byl implementován v programovacím jazyce Python, především pro veliké množství dostupných knihoven a nástrojů při řešení optimalizačních problémů. Byly využity knihovny SciPy, Numpy s a Matplotlib. Implementace optimalizačního algoritmu vyžaduje plně definovanou funkci, proto je při neplatné geometrii navrácena maximální možná hodnota. Algorimus také umožňuje definovat funkci omezující prohledávající prostor. Tato funkce byla definována jako minimální vzdálenost nutná pro platnou geometrii.

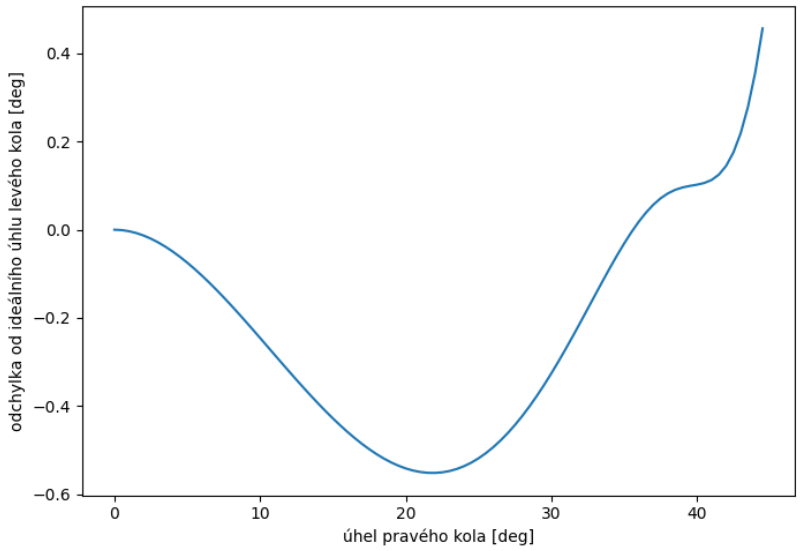
Pro samostatný výpočet byly zvoleny následující parametry:

r	170 mm
n	160 mm
v	20 mm
d	30 mm
$\boldsymbol{\varphi}$	45°
s	0,5°

Pro tyto parametry a optimalizací s 9 iteracemi bylo dosaženo následujících výsledků:

$\boldsymbol{\alpha}$	114,9505°
$\boldsymbol{l_2}$	62,8266 mm
$\boldsymbol{l_1}$	32,9186 mm

Průměrná druhá mocnina odchylky od ideálního úhlu kola je **0.1089** $[deg^2]$.



Obrázek 5.5: Odchylka hřebenového řízení vůči optimální geometrii

Získaná přesnost geometrie je pro potřebný model dostatečná a vzhledem k tolerancím při sestrojování vozidla je zanedbatelná. Při testování sestrojeného vozidla se ukázalo že příliš velké úhly mezi spoji l_1 a l_2 a výrobní tolerance způsobují vůli v pohybu kol viz kapitola 5.6.

Obrázky výsledné geometrie hřebenového řízení se nachází v příloze A.

5.2 Mechanická konstrukce

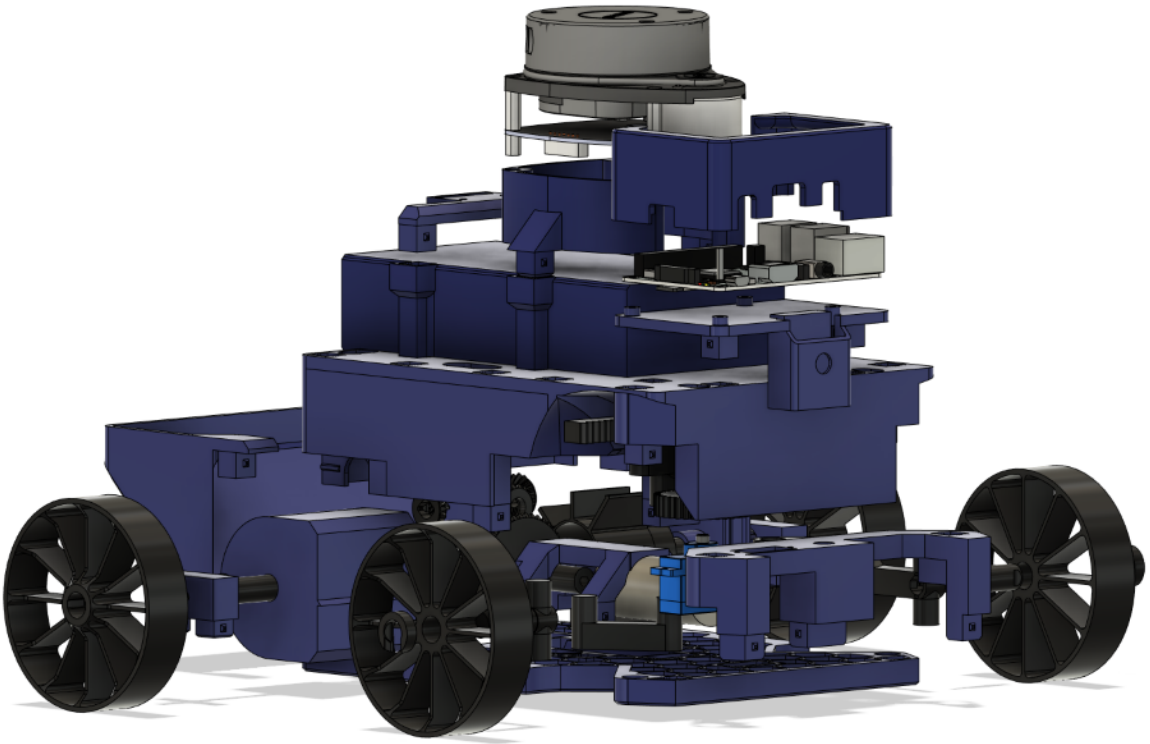
Jedná se o čtyřkolové vozidlo, které má dvě poháněné kola s diferenciálem a dvě řiditelná kola s Ackermannovým řízením. Celé vozidlo je navrženo jako modulární s možností přidání senzorů, nebo využití podobných komponentů.

Konstrukce robotického vozidla je omezena cílenými metodami 3D tisku, složitostí stavby a potřebnými nástroji. Proto jsou díly omezeny běžnou tiskovou plochou 3D tiskárny, co nejmenším využití spojovacího materiálu jako např. šrouby nebo lepení dílů, případně upravováním dílů po tisku jako sundávání podpor a začišťování.

Pro tvorbu 3D modelu vozidla byl využit CAD software Fusion 360. Jako návrhová strategie byl využit agilní přístup, kdy byly vymodelovány a vyzkoušeny důležité mechanické celky jako diferenciální převodovka a hřebenové řízení, podle kterých byl navržen zbytek vozidla.

Při návrhu všech dílů bylo dbáno na to aby byly dodrženy kritéria pro výrobu 3D tiskem pomocí FDM technologií viz. 3.5. Pro spoje mezi díly byly využity zacvakávací spoje, které umožňují jednoduché skládání bez spojovacího materiálu lepidla a také jednoduché rozebrání.

Během modelování bylo využito již existujících modelů Raspberry Pi 3B+[1], serva SG90[2] a lidaru RPLidar A1[3]. Zbytek komponentů byl vymodelován podle reálné předlohy.

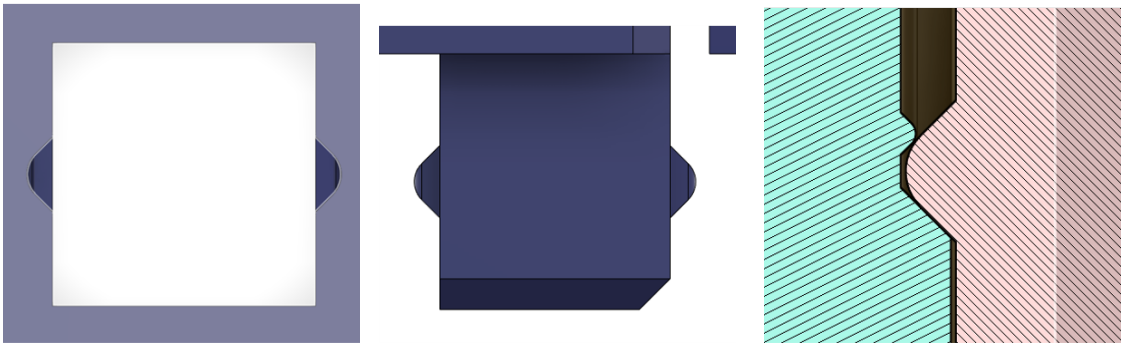


Obrázek 5.6: Rozložené vozidlo

[1]dostupné z: https://grabcad.com/library/raspberry-pi-3b-with-sd-card-1
[2]dostupné z: https://grabcad.com/library/sg90-servo-motor-2
[3]dostupné z: https://grabcad.com/library/rp-lidar-a1-inventor-reassembly-1

Zacvakávací spoje

Tyto spoje jsou v různých variacích využívány pro spojování většiny dílu vozidla. Nejčastější variantou je čtvercová díra a čtvercová zástrčka. Tato zástrčka má výstupky, které se zacvaknou do zarážek ve čtvercové díře a drží tak na místě. Pro pevné spojení je využito vlastností materiálu, který se při spojení dočasně stlačí a poté opět vrátí do původní polohy čímž pevně drží na místě. Tento zacvakávací mechanismus je navržen tak, aby jej bylo možné vytisknout v mnoha orientacích s minimálními převisy, proto je možné jej opakovaně využít v celém modelu s minimálními úpravami.

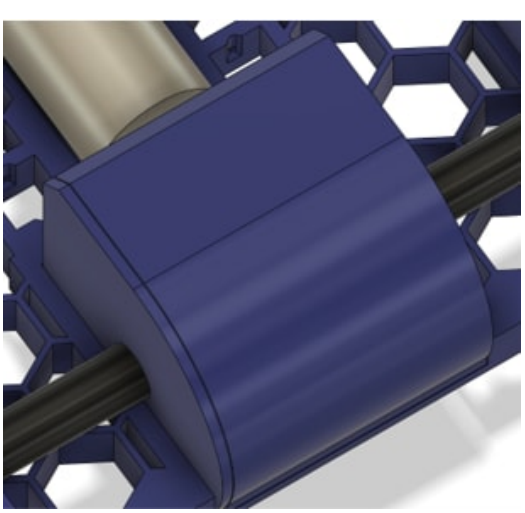


(a) Díra pro zacvakávací spoj (b) Protikus zacvakávacího spoje (c) Detail zacvakávacího spoje

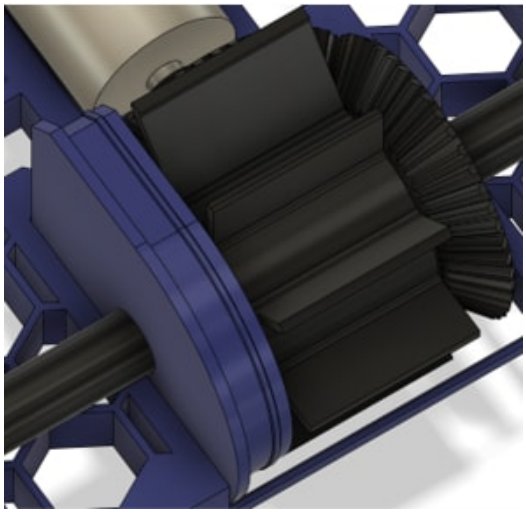
Obrázek 5.7: Zacvakávací spoj

Diferenciální převodovka

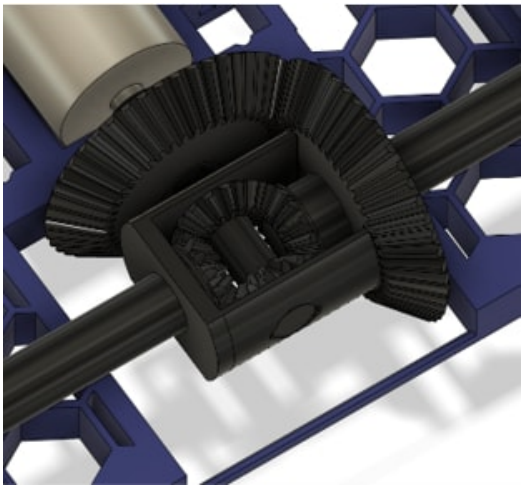
Diferenciální převodovka byla navržena jako plně 3D tisknutelná. Pro tvorbu ozubených kol bylo využito rozšíření FM Gears. Při návrhu jsem postupoval pomocí postupu zevnitř ven, jelikož interní ozubené kola jsou klíčovou součástí celého mechanismu. Po úspěšném návrhu tisknutelných ozubených kol s co nejmenší velikostí jsem se snažil zbytek převodovky navrhnout tak aby byla výsledná převodovka co nejkompaktnější. Celý návrh využívá mnoho zacvakávacích spojů umožňujících jednoduché složení.



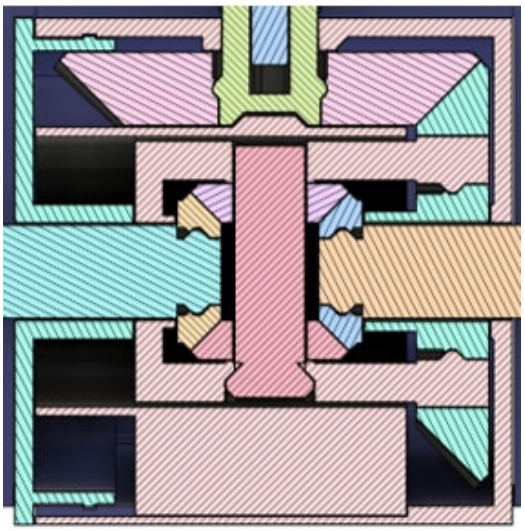
(a) Diferenciální převodovka s krytem



(b) Diferenciální převodovka bez krytu



(c) Ozubené kola diferenciální převodovky

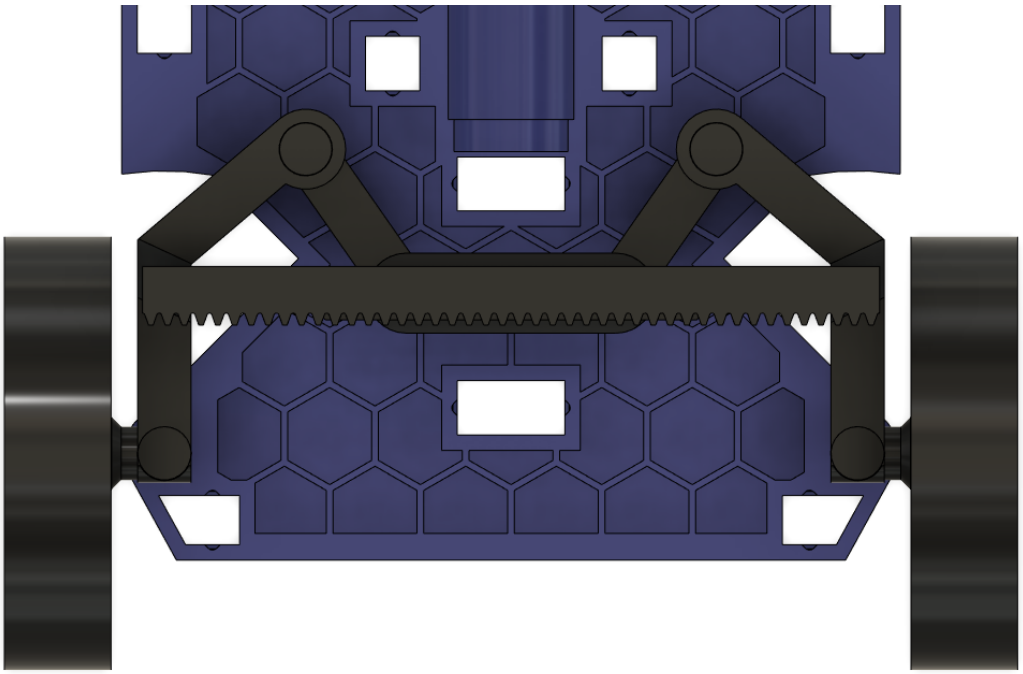


(d) Průřez diferenciální převodovkou

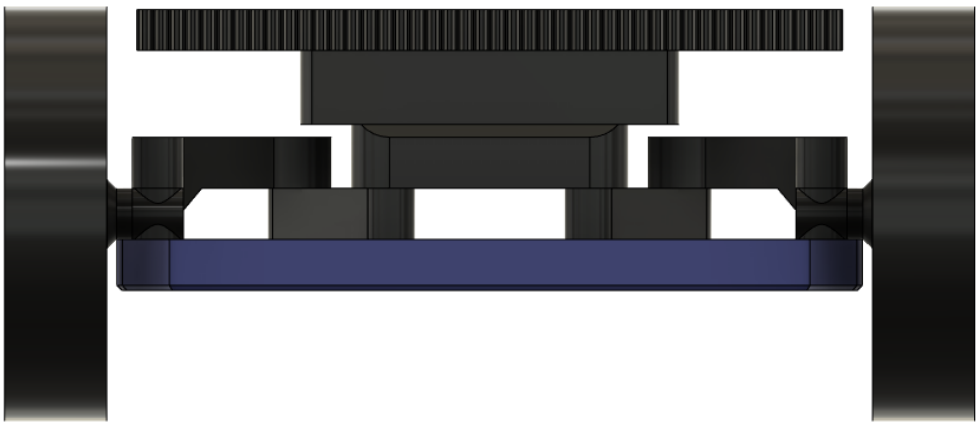
Obrázek 5.8: Diferenciální převodovka

Hřebenové řízení

Hřebenové řízení bylo navrženo dle spočtené geometrie viz. 5.1. Daná geometrie byla převedena na 3D model a to tak, aby při pohybu nekolidovala s ostatní geometrií vozidla a splňovala podmínky pro 3D tisk. Toho bylo docíleno rozdělením některých součástí na více dílů tak aby nebylo potřeba využít podpor a byla dodržena kvalita povrchu kontaktních ploch a jejich pevnost. Jelikož řídící servo umožňuje pohyb pouze o 180° byla navržena převodovka umožňující pohyb předních kol v plném rozsahu.

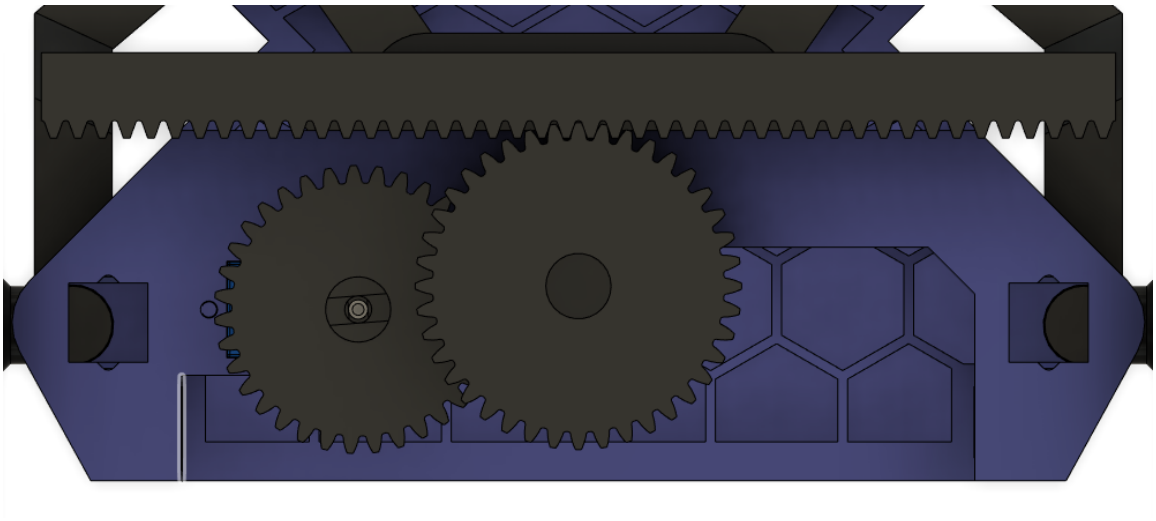


(a) Hřebenové řízení, pohled shora

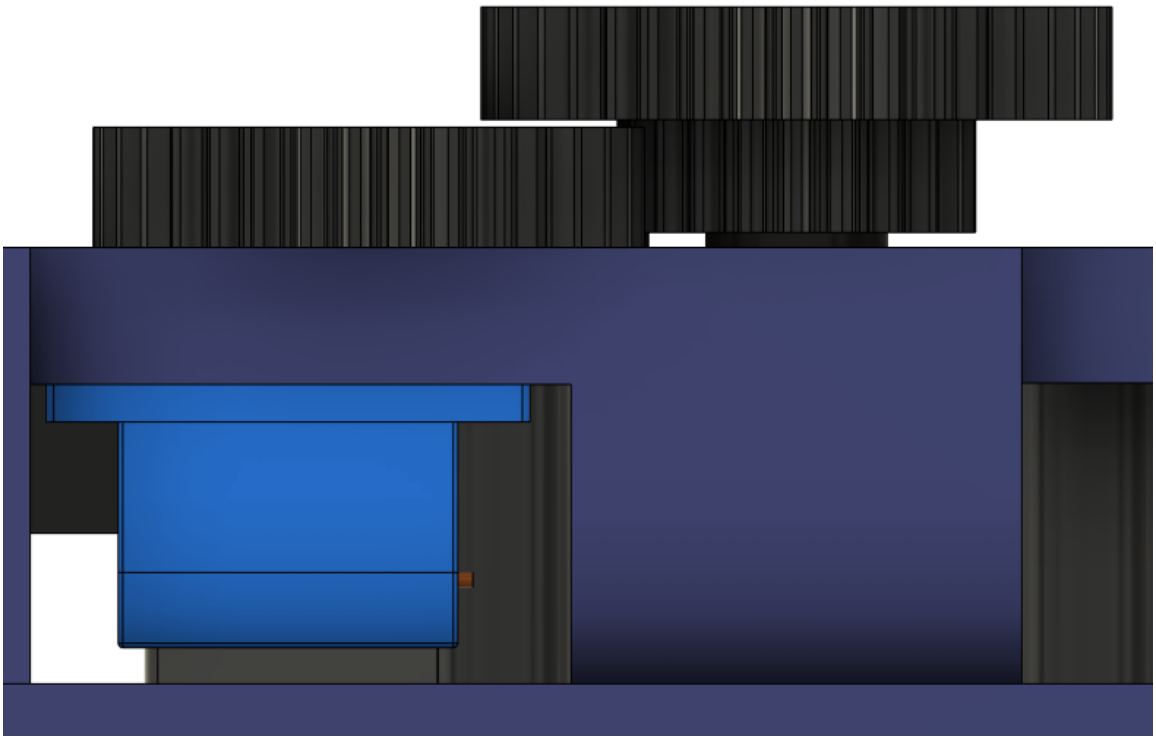


(b) Hřebenové řízení, pohled zepředu

Obrázek 5.9: Vytvořené hřebenové řízení



(a) Převodovka hřebenového řízení, pohled shora

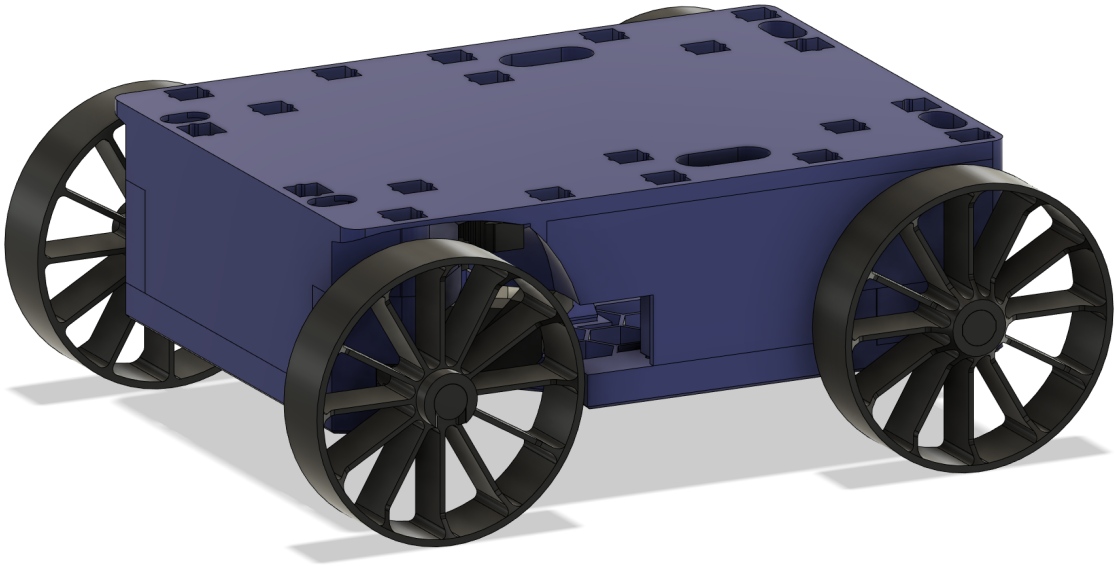


(b) Převodovka hřebenového řízení, pohled zepředu

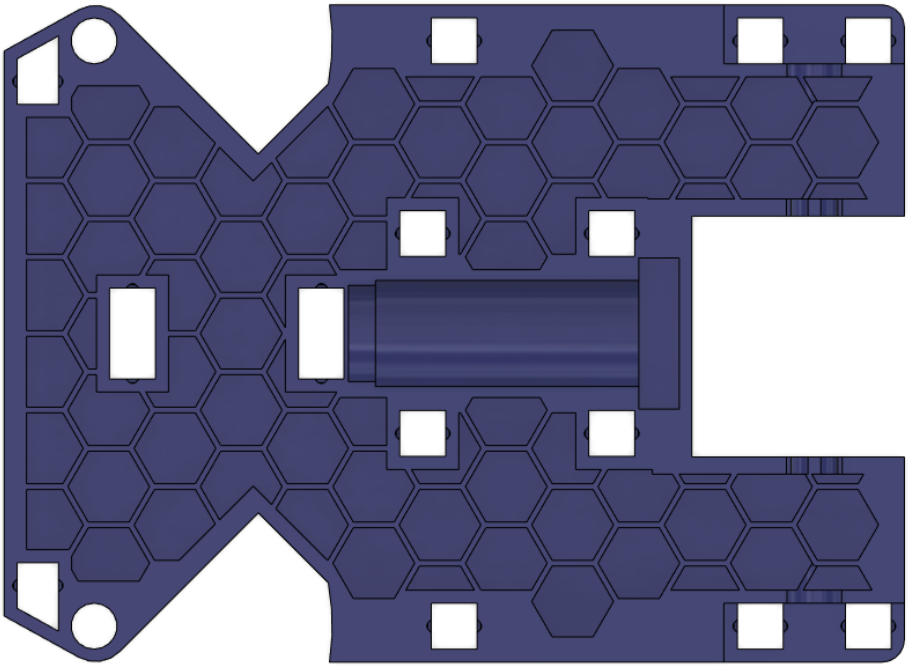
Obrázek 5.10: Převodovka hřebenového řízení

Podvozek vozidla

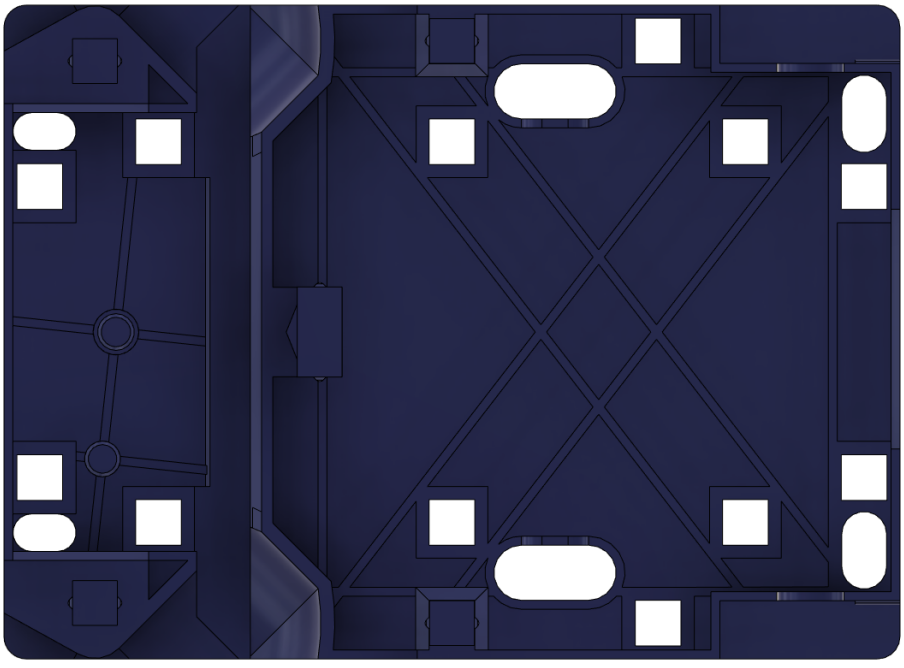
Podvozek vozidla byl navržen jako dva ucelené kusy spojující hřebenové řízení, diferenciální převodovku a motor v jeden funkční celek. Velikost tohoto dílu je omezena tak, aby jej bylo možno vytisknout na většině FDM 3D tiskáren, proto byla zvolena velikost půdorysu $200\,\text{mm} \times 146\,\text{mm}$. Díl podvozku zároveň slouží jako mechanická součást hřebenového řízení, kde omezuje jeho pohyb tak aby se pohybovalo pouze v určených směrech.



Obrázek 5.11: Podvozek vozidla

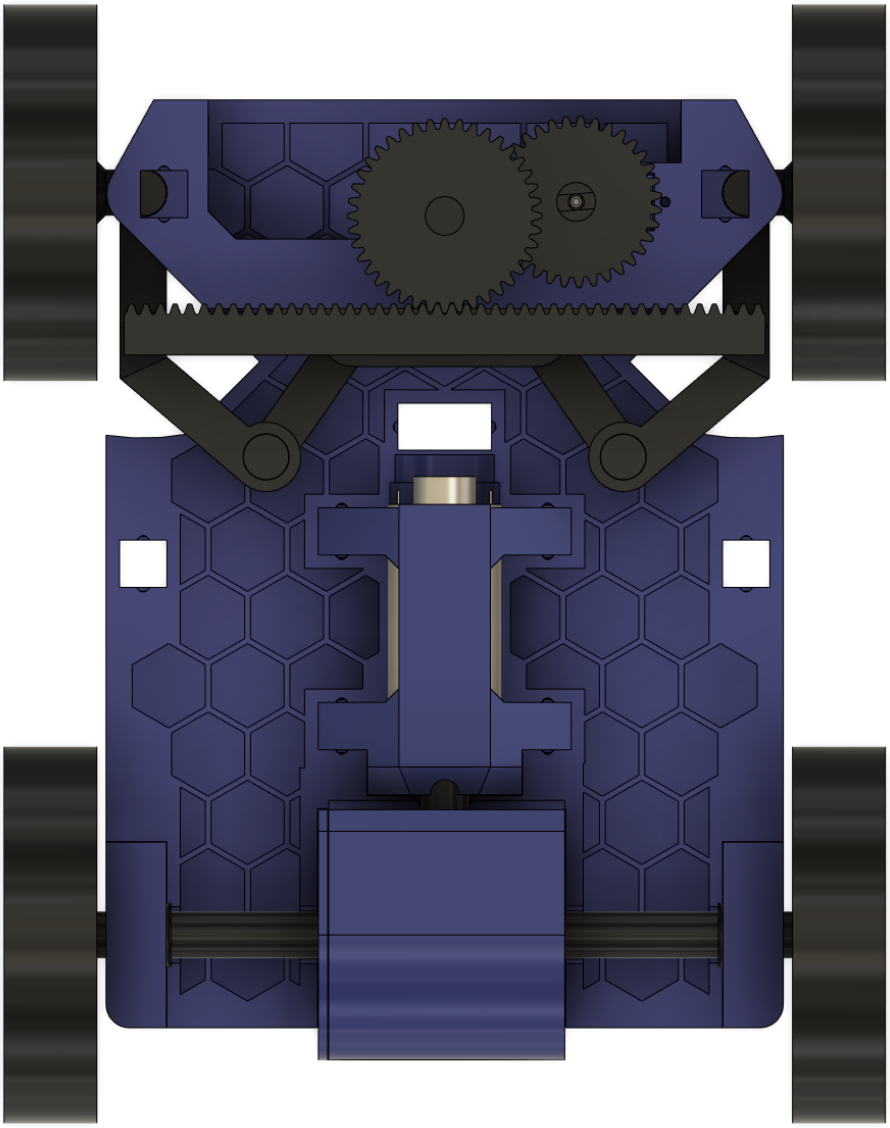


(a) Podvozek vozidla, spodní část



(b) Podvozek vozidla, horní část

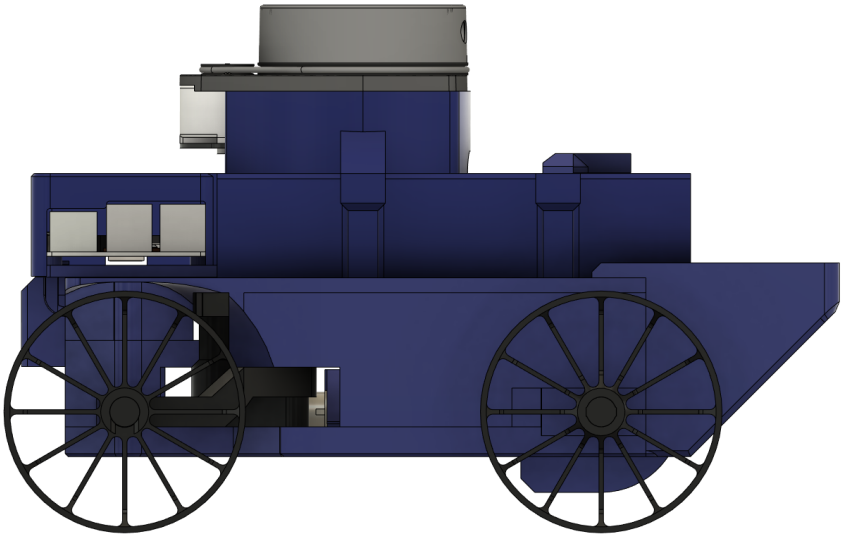
Obrázek 5.12: Části podvozku vozidla



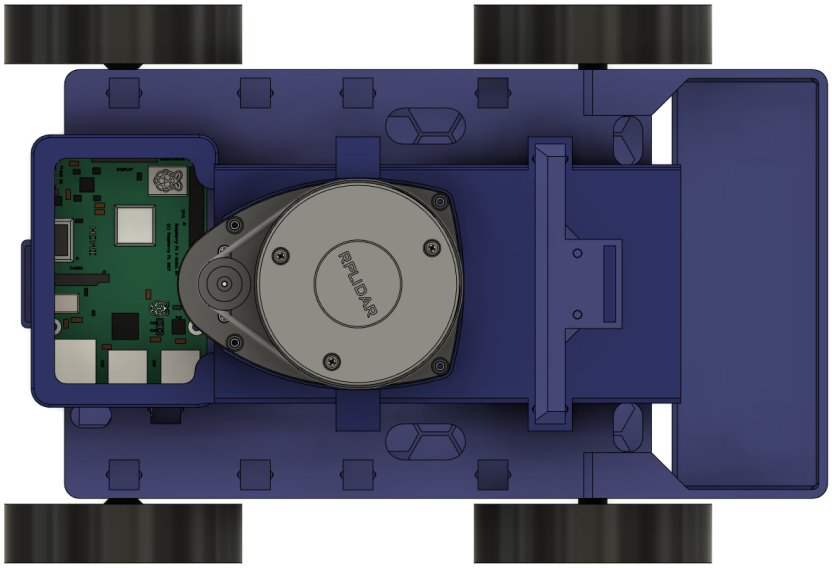
Obrázek 5.13: Spodní část podvozku vozidla včetně vnitřních komponentů

Připevnění elektronických komponent

Pro připevnění elektronických komponent jsou využity zacvakávací spoje na horní části podvozku. Jedná se o baterii, jednodeskový počítač, kameru a inerciální senzor. Tím, že jsou tyto díly odděleny od zbytku vozidla v případě změny elektronických součástí je lze poměrně jednoduše upravit bez změn na součástech podvozku. Součástí robota je také oddělení pro schování přebytečné délky USB kabelů.



(a) Připevnění elektrických komponent, boční pohled



(b) Připevnění elektrických komponent, pohled shora

Obrázek 5.14: Části podvozku vozidla

Elektrické prvky

Robotické vozidlo se skládá z mnoha dílčích komponentů, které operují s různými napájecími napětími a pro ovládání nebo komunikaci vyžadují různé druhy signálů a komunikačních sběrnic.
Vozidlo obsahuje následující komponenty:

- Raspberry Pi 3B+
- RPLidar A1M8
- MPU 9250 SPI/IIC Modul
- modul H-můstek L9110S
- motor Machifit 25GA370 s enkodérem
- USB mikrofon
- servo PDI-1109MG
- USB-C přepínatelný napěťový zdroj ZY12PDN
- kamera Raspberry Pi Camera Module V2
- záložní baterie Xiaomi MI Power Bank 3

Pro napájení robotického vozidla je využito záložní baterie Xiaomi MI Power Bank 3, jedná se o zdroj s kapacitou 74 Wh poskytující pro napájení 2 konektory USB-A poskytující při jejich současném využití napájecí napětí 5 V a proud 3 A a jeden konektor USB-C podporující standard USB-PD nabízející řadu napětí od 5 do 20 V s výstupem až 45 W. Tento konektor lze také využít k nabíjení záložní baterie také pomocí standardu USB-PD nabíječkou s výkonem až 45 W s dobou celkového nabíjení 4,5 h.

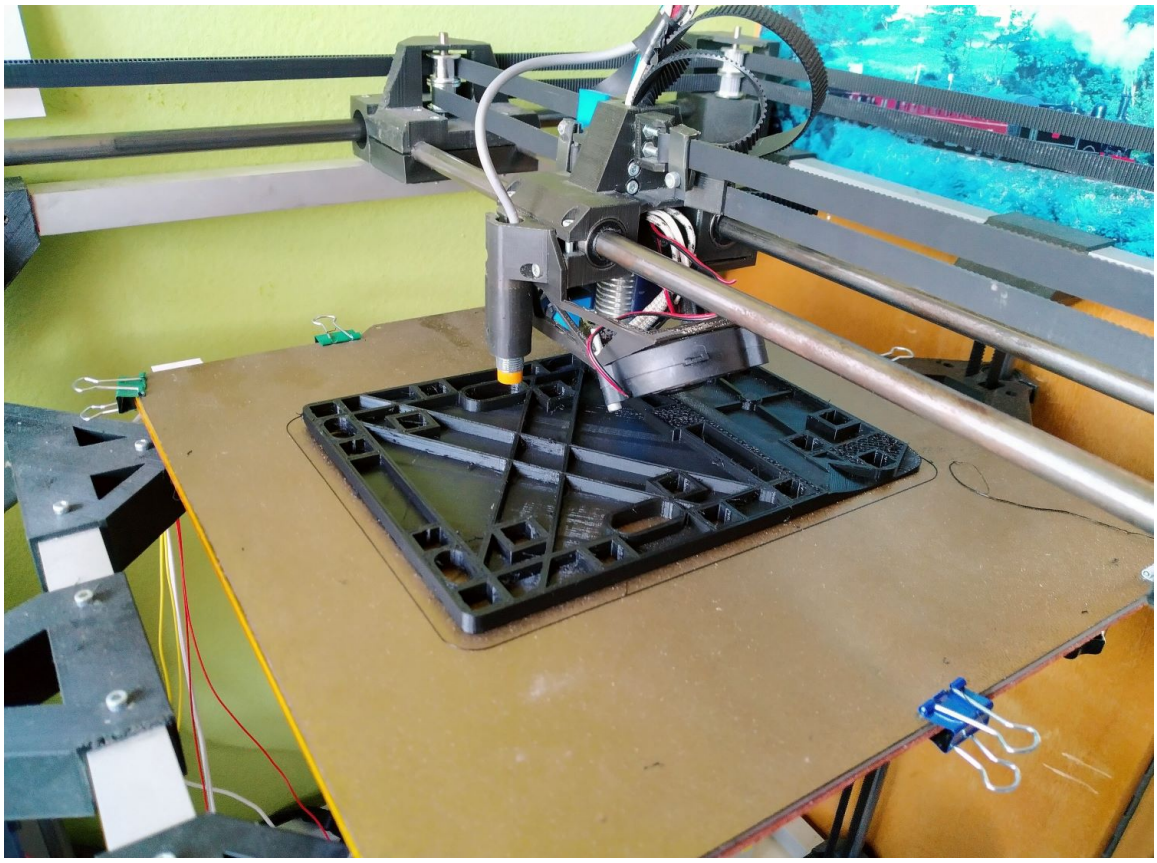
Tento záložní zdroj napájí z USB-A konektoru Raspberry Pi 3B+. Druhý USB-A port je využit pro napájení serva, jelikož je potřeba zajistit uzemnění serva, je uzemňovací vodič spojen také s výstupním pinem Raspberry Pi 3B+. Konektor USB-C je využit k napájení motoru. Je zapojen do modulu poskytujícího 12 V napětí pomocí standardu USB-PD. Tento modul je pak propojen s modulem s dvěma čipy L9110S ovládající napájení motoru.

K jednodeskovému počítači je připojen akcelerometr, ovladač motoru, enkodér, kamera, mikrofon a lidar. Pro připojení modulu akcelerometru MPU9250 byla využita I2C sběrnice na GPIO pinech 2 a 3, pro ozámemení přerušní byl využit GPIO pin 18. Napájení je realizováno pomocí 5 V pinu a uzemňovacího pinu. Pro připojení ovladače motoru byly využity piny 13 a 19. Servo je řízeno pomocí pinu 22. Enkodér využívá k přenosu dat piny 17 a 27 společně a je napájen napájecím pinem s napětím 3,3 V a uzemňovacím pinem. Mikrofon je zapojen do USB portu Raspberry Pi 3B+. Kamera Raspberry Pi Camera Module V2 je připojena pomocí CSI portu. Ovládání a napájení lidaru RPLidar A1M8 je realizováno pomocí USB portu Raspberry Pi 3B+. Podrobnější popis zapojení se nachází v příloze B.

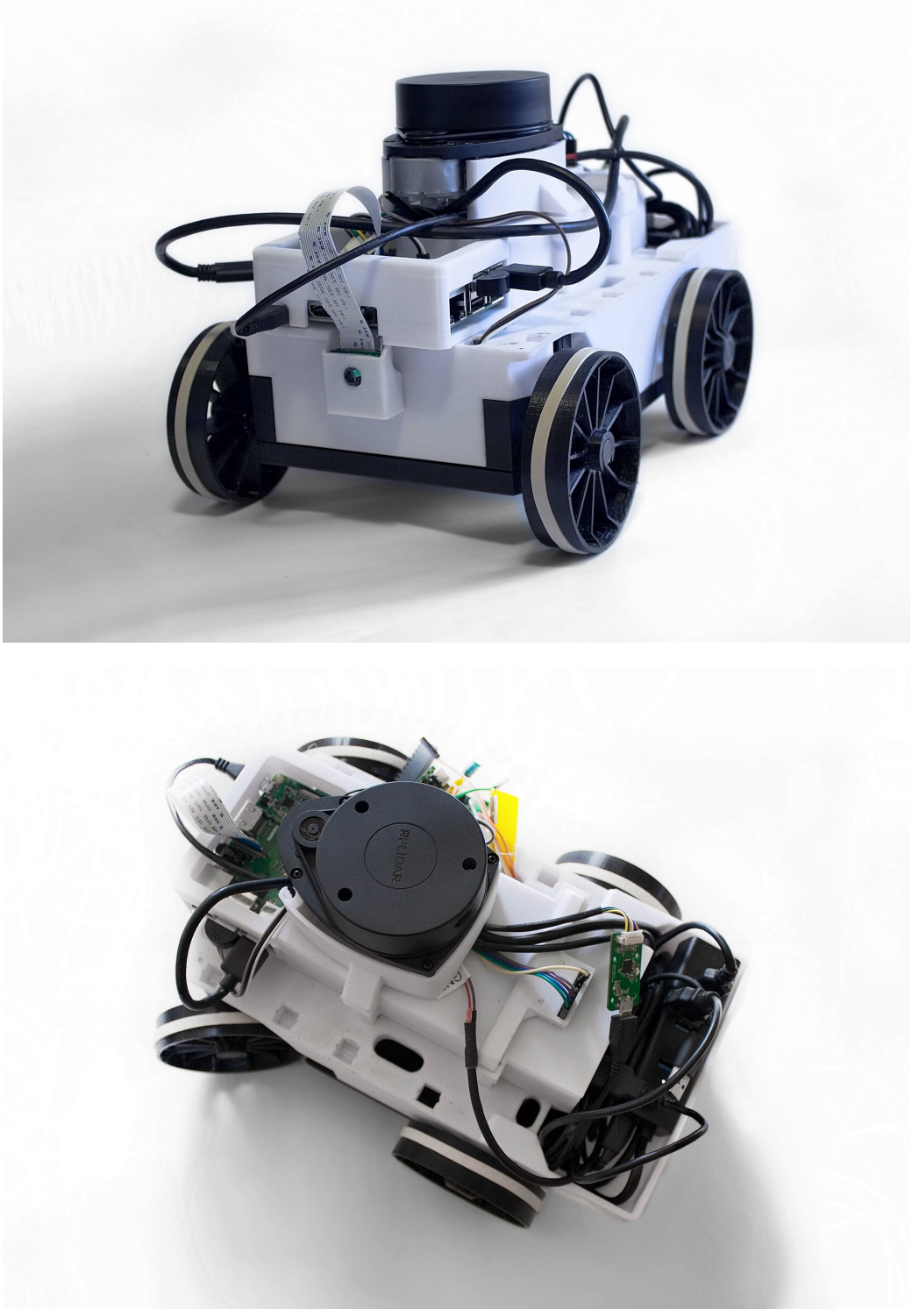
Sestavení vozidla

Pro sestavení vozidla bylo potřeba všechny potřebné díly vytisknout na 3D tiskárně. K tomu byly využity FDM tiskárny Anet A8 s modifikacemi a tiskárna vlastní konstrukce. Jelikož každá z 3D tiskáren tiskne s jinými tolerancemi, bylo také možné částečně ověřit kompatibilitu dílů vytištěných na různých 3D tiskárnách.

Při sestavování podvozku bylo využito pouze dílů vytištěných na 3D tiskárnách a šroubů dodaných v balení serva. Některé kabely např. kabel pro připojení motoru nebyl ukončen konektorem, proto byly přidány vlastní ukončení. Toto jde ale také řešit spojkami bez pájení nebo speciálního vybavení. Některé díly bylo nutno po tisku kvůli nepřesnostem mírně upravit skalpelem nebo smirkovým papírem, to je ale při tvorbě mechanických komponent s přesnou tolerancí na 3D tiskárnách poměrně běžné.



Obrázek 5.15: 3D tisk dílu podvozku



Obrázek 5.16: Sestavené robotické vozidlo

5.3 Řízení elektromechanických komponentů

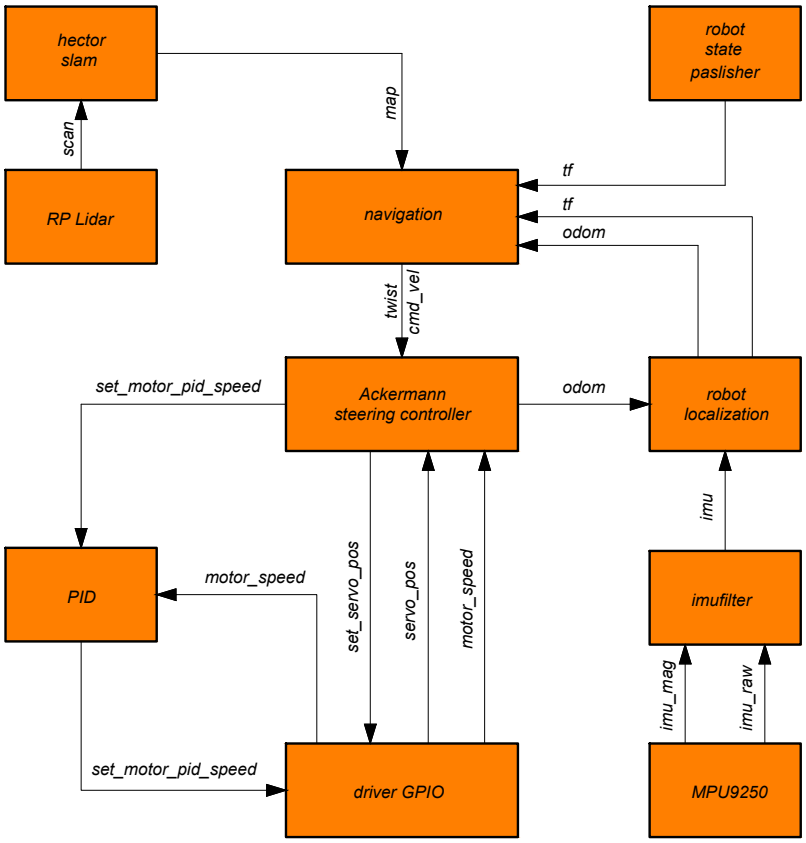
Pro řízení všech komponentů v robotickém vozidle bylo využito Raspberry 3B+, které poskytuje mnoho možností pro připojení vstupních i výstupních periférií.

Jako software byl vybrán ROS viz. 3.3, který umožňuje jednoduchou integraci mnoha komponent díky mnoha existujícím balíčkům, implementovaných algoritmů a mnoha dostupných nástrojů. ROS zařizuje ovládání motoru, serva, čtení senzorů, mapování prostředí a navigaci v něm.

Struktura Aplikace v ROS

Při tvorbě sortware bylo snahou vytvořit software co nejvíce kompatibilní s existujícími balíčky a typy zpráv. To umožňuje využít veliké množství existujících navigačních a plánovacích algoritmů.

Byl implementován uzel starající se o ovládání natočení přeních kol a rychlosti otáčení motoru. Byl modifikován existující balíček pro práci s inerciálním senzorem. Dále byly využity balíčky pro kontrolu Ackermannova podvozku, lidaru, balíčky pro mapování a navigaci. Bližší implementační detaily popisuje příloha B.



Obrázek 5.17: Schéma aplikace

Řízení rychlosti zadních kol a natočení předních kol

Pro ovládání rychlosti vozidla je potřeba kontrolovat otáčky motoru. K tomu je využito h-můstku, konkrétně modulu obsahujícího dva čipy L9110S, kde byl využit pouze jeden z nich. Ovládání motoru je řízeno pomocí čtyř signálů viz. tabulka 5.1.

Vstup				Výstup
IA	IB	OA	OB	Popis
L	L	L	L	Vypnuto
H	L	H	L	Otáčení vpřed
L	H	L	H	Otáčení zpět
H	H	H	H	Vypnuto

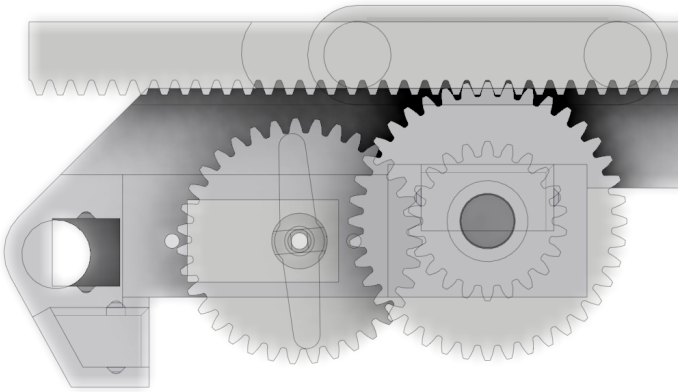
Tabulka 5.1: Pravdivostní tabulka pro řízení motorů pomocí modulu L9110S

Pro přesné ovládání otáček motoru jsou otáčky měřeny pomocí enkodéru, který je součástí motoru viz obr. 5.18. Jedná se o enkodér s dvěma senzory magnetického pole využívající hallův jev, které snímají pohyb 11 magnetů kolem osy motoru. Jelikož mají senzory úhel 90° jedná se o běžný inkrementální enkodér s detekcí směru pohybu. Enkodér je připojen na osu motoru před jeho převodovkou, proto pro je potřeba při výpočtu otáček dbát ohled také na převodový poměr převodovky.

Obrázek 5.18: Enkodér s hallovými senzory na motoru

Pro ovládání natočení kol je využito elektronické servo s kovovými převody PDI-1109MG. Jedná se o běžné modelářské servo velikosti mikro s úhly natočení 0 - 180°. Pro jeho ovládání je využito PWM signálu s frekvencí 330 Hz. Servo je spojeno s předními koly pomocí ozubených kol s převodovým poměrem $\frac{35}{22}$, kde kolo spojené s hřebenem má roztečnou kružnici o poloměru 20 mm. Tímto převodovým poměrem je docíleno že je 180° pohybu serva dostatečných pro plný pohyb předních kol.

Pro řízení úhlu natočení předních kol bylo vytvořeno virtuální střední kolo, které dále zjednodušuje ovládání robota jelikož je třeba počítat pouze s jedním kolem umístěným ve středu vozidla, což zjednodušuje další výpočty a generalizuje danou geometrii.



Obrázek 5.19: Převody serva a ozubeného hřebenu

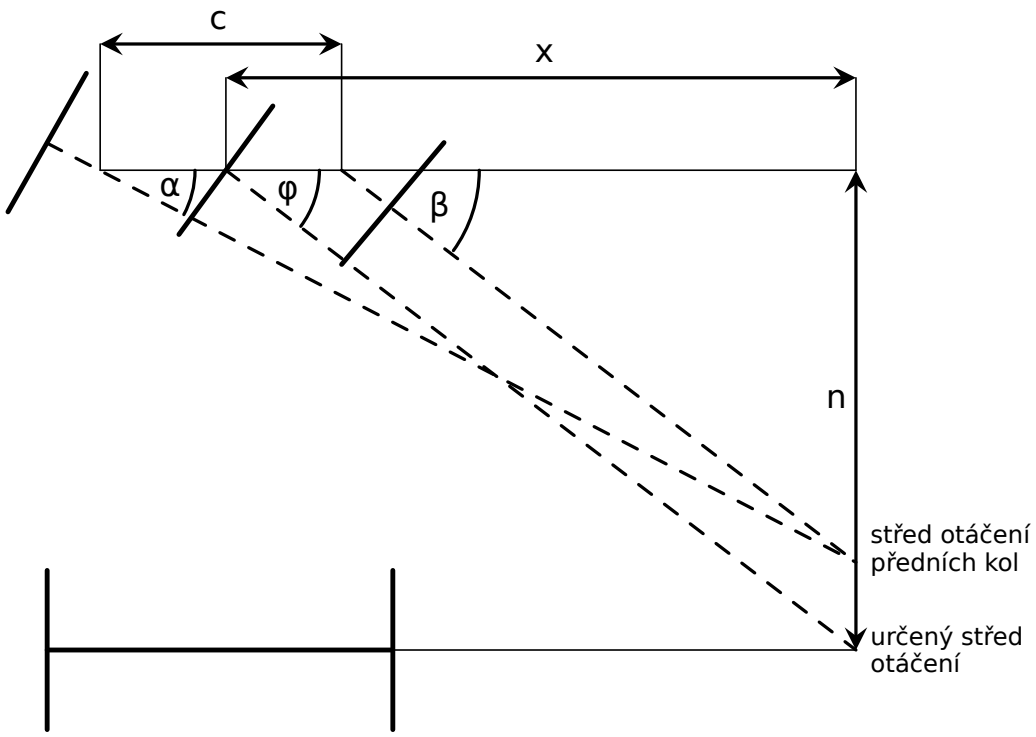
Pro přepočet úhlu natočení serva na úhel natočení virtuálního středního kola bylo využito lineární aproximace. Pro výpočet hodnot jsem vytvořil skript počítající odpovídající dvojice hodnot. Bylo využito výpočtu přepočítavajícího úhel pravého kola na úhel levého kola hřebenového řízení z optimalizačního algoritmu 5.1.

Dále byl vytvořen algoritmus pro aproximaci úhlu zatáčení virtuálního středního kola z úhlu levého a pravého kola. Jelikož úhel natočení levého a pravého kola nebude ideální, střed otáčení nebude ležet na ose zadních kol viz obrázek 2.8. Chyba natočení kol oproti ideální ale není není vysoká, proto lze posunutí středu otáčení v ose vozidla zanedbat. Pro výpočet středu otáčení vozidla jsem využil střed otáčení předních kol, který vznikne průsečíkem jejich os. Vzdálenost středu otáčení předních kol od osy vozidla byla přenesena na zadní nápravu. Tento bod byl zvolen jako střed otáčení vozidla viz obrázek 5.20. Pro tento střed otáčení byl vypočten úhel natočení virtuálního středního kola. Úhel natočení vituálního předního kola je popsán následujícími rovnicemi.

$$x = \frac{2\cos(\beta)\sin(\beta-\alpha) + c^2\sin(\alpha)}{2c\sin(\alpha)} \tag{5.1}$$

$$\tan\varphi = \frac{x}{n} \tag{5.2}$$

$$\varphi = \arctan\left(\frac{2\cos(\beta)\sin(\beta-\alpha) + c^2\sin(\alpha)}{2cn\sin(\alpha)}\right) \tag{5.3}$$



Obrázek 5.20: Určení středu otáčení pro neideální úhly zatočení předních kol

Servo i motor jsou ovládány pomocí vstupně výstupních pinů. Jelikož se jedná o komponenty s vysokým odběrem, jejich napájení je realizováno mimo jednodeskový počítač pomocí samostatných napájecích obvodů viz obrázek 5.1. Oba komponenty jsou řízeny PWM signálem generovaným pomocí knihovny Pigpio 3.2.

5.4 Aplikace ovládající elektrické komponenty

Byla vytvořena aplikace, která slouží jako ovladač serva a motoru, tato aplikace přijímá řídící zprávy ovládající rychlost motoru a natočení předních kol a odesílá zprávy o aktuální poloze serva a rychlosti motoru.

Pro ovládání rychlosti motoru je využíváno PWM signálu. Hodnota řídící zprávy představuje úsilí se kterým je motor řízen. Může nabývat hodnot od -1 do 1, kde -1 je maximální

úsilí v jednom směru otáčení, 0 je žádné úsilí a 1 maximální úsilí v druhém směru otáčení. Úsilí je převáděno na střídu PWM signálu přímo úměrně.

Pro určení rychlosti motoru jsou měřeny změny logických úrovní na hallových senzorech. Knihovna Pigpio nabízí využít přerušení na libovolném pinu jednodeskového počítače Raspberry Pi. Na rozdíl od běžných přerušení se ale jedná o pravidelné kontrolování stavu pinů při pravidelně generovaném přerušení pomocí periférií. Proto je možné, že velmi krátké pulzy nebudou zaznamenány. Jelikož se ale jedná o fyzický systém omezený maximální rychlostí motoru, je rychlost snímání pinů dostatečná. Pro zjišťování stavu enkodéru jsou využity 3 proměnné, které uchovávají poslední stav signálů ze senzorů a celkový počet kroků enkodéru.

Implementace

Pro měření enkodéru byla implementována třída Encoder, která zapouzdřuje veškerou obsluhu příchozích signálů a poskytuje metody pro čtení stavu enkodéru. Tato třída komunikuje se serverem knihovny Pigpio, obsluhuje příchozí přerušení a čítá počet kroků. Při inicializaci třídy nastaví vstupní piny, pull-up rezistory a zaregistruje obsluhu přerušení na vzestupné i sestupné hrany signálu. Obsluha přerušení zkontroluje pin přerušení a podle předchozího stavu inkrementuje nebo dekrementuje čítač kroků enkodéru. Při čtení stavu enkodéru je třeba dbát na to, že čítač kroků může přetéct je proto s jeho hodnotou vhodně pracovat.

Ovládání h-můstku s obvodem L9110S bylo také řešeno třídním přístupem. Třída HBridge umožňuje nastavit směr otáčení střídu PWM signálu a směr otáčení až dvou připojených motorů. Při inicializaci je navázána komunikace s se serverem Pigpio a nastaveny výstupní piny. Rychlost motoru je nastavována pomocí proměnné s hodnotou 0-255 určující střídu PWM signálu. PWM signál je generován knihovnou Pigpio. Modul s h-můstky je ovládán signály viz tabulka 5.1.

Pro ovládání serva je také vytvořena třída zapouzdřující veškerou komunikaci s hardware. Třída Servo při inicializaci získá pin pro ovládání serva a jeho limity pohybu. Nastaví daný pin na výstupní. Při zadání úhlu přepočítá zadaný úhel dle limitů na délku řídícího signálu viz kapitola 3.2.

V případě změny jednodeskového počítače na počítač nepodporující knihovnu Pigpio je potřeba pouze znovu implementovat tyto třídy. Zbylá část programu je nezávislá na hardware.

Aplikace funguje jako nekonečná smyčka která běží až do příchodu ukončovacího signálu. Při startu vytvoří instance objektů ovládající servo, h-můstek a motor, poté inicializuje uzel v ROS a registruje vysílané a přijímané zprávy. Jako obsluhy přijetí zprávy jsou použity jednoduché funkce aktualizující stav proměnných pro ovládání motoru a předních kol. V každé iteraci nekonečné smyčky jsou aktualizovány hodnoty pro ovládání motoru a předních kol, přepočten požadovaný úhel předních kol na natočení serva a zadány nové řídící hodnoty. Je zjištěn počet kroků na enkodéru, který je porovnán s předchozím stavem a z tohoto rozdílu je vypočtena aktuální rychlost motoru. Poté jsou publikovány aktuální rychlost a úhel natočení kol, smyčka je poté uspána po zbytek cyklu tak, aby byla dodržena její běhová frekvence. V případě ukončení smyčky je před ukončením programu vypnut motor.

5.5 Možnosti ovládání robotického vozidla

Jelikož robotické vozidlo pro ovládání podporuje standardní rozhraní, existuje velké množství aplikací pro ovládání vozidla. Vozidlo lze ovládat dvěma možnými způsoby.

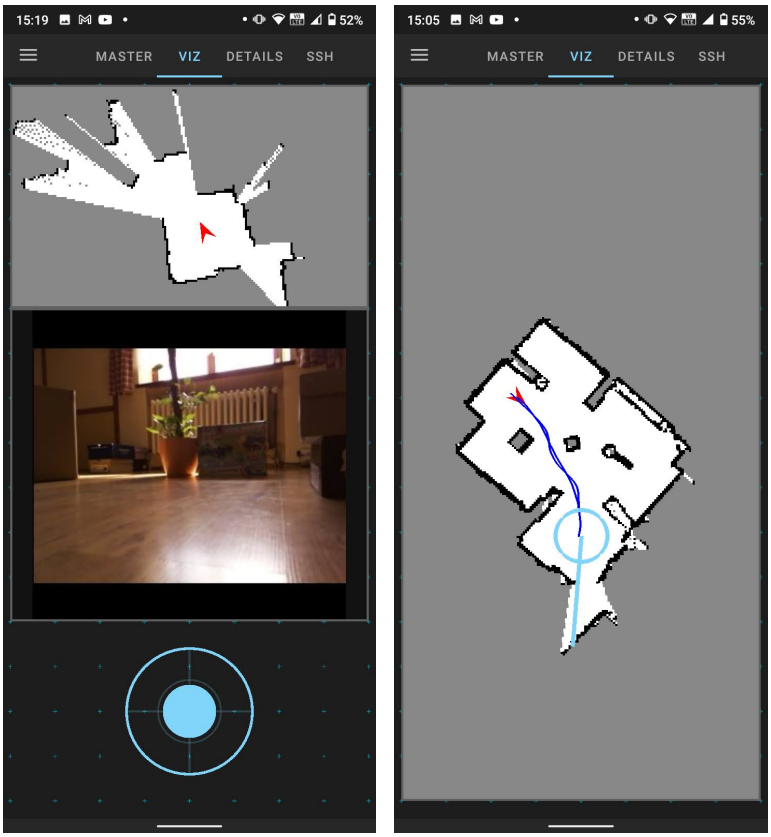
- teleoperace
- pomocí navigačního cíle

Při teleoperaci jsou vozidlu posílány informace o požadované lineární a úhlové rychlosti. Jedná se o zprávy typu `geometry_msgs/Twist`. Všehny ovládací aplikace podporující tento druh zpráv mohou vozidlo ovládat.

Dále následují příklady aplikací umožňující ovládat toto robotické vozidlo.

Teleoperace

Pro teleoperaci lze využít například aplikaci ROS-Mobile[4] dostupnou pro mobilní telefony s operačím systémem Android. Tato aplikace umožňuje dálkově ovládat vozidlo pomocí virtuálního joysticku, sledovat video z kamery a mapu vytvořenou pomocí mapování.



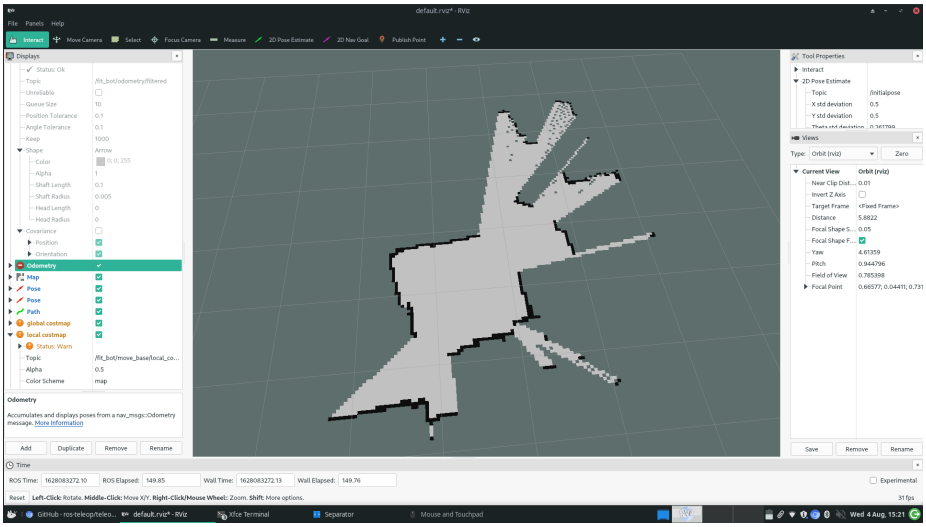
Obrázek 5.21: Snímky obrazovky během ovládání robotického vozidla

Další možností je například balíček ROS `teleop_twist_keyboard` umožňující ovládání robotického vozidla pomocí klávesnice v terminálové aplikaci.

[4] dostupné z: https://github.com/ROS-Mobile/ROS-Mobile-Android

Ovládání pomocí navigačního cíle

Robotické vozidlo také umožňuje navigaci pomocí navigačního cíle. Stačí odeslat zprávu s cílovou polohou a robot zařídí plánování trasy a navigaci. Pro odesílání těchto zpráv lze využít například aplikaci rviz, která také umožňuje zobrazit mapu získanou robotem, jeho aktuální polohu a naplánovanou trasu.



Obrázek 5.22: Snímek obrazovky z aplikace rviz, zobrazující aktuální mapu okolí

5.6 Ověření funkčnosti a interpretace výsledků

Tato sekce se zabývá způsobem ověření funkčnosti, provedenými testy a obsahuje získané výsledky.

V případě uvedení robotického vozidla na trh je potřeba provést veliké množství testů, například:

- test vlastností vozidla
- analýza chemických, fyzikálních, mechanických a elektrických nebezpečí a nebezpečí souvisejících s hořlavostí, hygienou a radioaktivitou
- test elektromagnetické kompatibility
- zátěžové testy
- uživatelské testování

Jelikož toto testování přesahuje obsah práce, zaměřil jsem se na testy, které měly za cíl ověřit vlastnosti vozidla a jeho možnosti. Jejich cílem je také demonstrovat schopnosti vozidla, především jeho jízdní vlastnosti a schopnost se autonomně navigovat.

Jízdní vlastnosti

První testy vozidla zkoušely jízdní vlastnosti vozidla. Testování probíhalo pomocí mobilní aplikace ROS-Mobile, která umožňuje vozidlu odesílat zprávy typu `geometry_msgs/Twist`, ovládající robot pomocí dopředné a úhlové rychlosti.

Po vyzkoušení jízdních vlastností bylo odhaleno, že kola vytištěné na 3D tiskárně nemají dostatečné tření pro pohyb na kluzkém povrchu, proto byly na kola přidány gumy a následující testy probíhaly s jejich využitím. Dále testy jízdních vlastností odhalily, že při pohybu po nerovném povrchu diferenciální převodovka v kombinaci s pevným podvozkem způsobuje uváznutí vozidla na místě v případě ztráty kontaktu s povrchem.

Dále byla testována maximální rychlost vozidla, skutečný minimální poloměr zatáčení, maximální úhel stoupání a schopnost překonávat schod.

Testování maximální rychlosti

Na zemi byla vyznačena vzdálenost 1 m, kde vzdálenost byla rozdělena na 4 m a 1 m. Vozidlo bylo umístěno na začátek vyznačené vzdálenosti, první metr dráhy byl počítán jako rozjezd, následující po další 4 metry pohybu vozidla byl měřen čas pomocí stopek. Z času pohybu mezi těmito značkami byla vypočítána průměrná rychlost. Toto měření bylo prováděno na rovné laminátové podlaze. Bylo provedeno celkově 20 měření. Data z měření se nachází v příloze C.

Pro výpočet rychlosti byl použit následující vzorec

$$v = \frac{s}{t} \tag{5.4}$$

Maximální rychlost vozidla je $0{,}2132\,\mathrm{m\,s^{-1}} \pm 0{,}0005\,\mathrm{m\,s^{-1}}$

Tato maximální rychlost odpovídá technickým parametrům vybraných komponent a je pro pohyb vozidla naprosto dostačující.

Testování minimálního poloměru zatáčení

Pro testování minimálního úhlu zatáčení byla na zem vyznačena rovná čára. Vozidlo bylo umístěno kolmo k čáře. Kola byly nastaveny na maximální výchylku. Vozidlo bylo uvedeno do pohybu a byla měřena vzdálenost, kde vozidlo opět protne čáru, byla měřena hodnota minimálního průměru dráhy vnějšího kola. Z této vzdálenosti byl vypočítán maximální poloměr zatáčení. Bylo provedeno celkově 20 měření. Polovina měření byla prováděna při zatáčení doleva a polovina při zatáčení doprava. Data z měření se nachází v příloze C.

Pro výpočet minimálního poloměru zatáčení byl použit následující vzorec:

$$x = \frac{d - r}{2} \tag{5.5}$$

Kde x je minimální poloměr zatáčení, d měřená vzdálenost a r vzdálenost kol.

Minimální poloměr zatáčení vozidla je: $356{,}9\,\mathrm{mm} \pm 6{,}7\,\mathrm{mm}$

Teoretický poloměr zatáčení je 225,2 mm. Rozdíl těchto hodnot může být způsoben skluzem kol při zatáčení. Po bližší inspekci hřebenového řízení také bylo zjištěno, že v případě natočení kol, kdy spoje hřebenového řízení svírají vysoký úhel, způsobují nepřesnosti vůli v natočení předních kol. Tento jev je nejvíce patrný při maximálním natočení předních kol.

Měření maximálního úhlu stoupání

Testování maximálního úhlu stoupání bylo provedeno na nakloněné rovině, úhel stoupání byl zvyšován dokud vozidlo bylo schopné stoupat, poté byl úhel stoupání změřen. Bylo provedeno 20 měření. Data z měření se nachází v příloze C.

Maximální úhel stoupání vozidla je: $5{,}0° \pm 0{,}1°$

Byl zjištěn poměrně nízký úhel stoupání vozidla, to je způsobeno volbou motoru, pro zvýšení úhlu stoupání je potřeba využít výkonnější motor.

Měření schopnosti překonávat schod

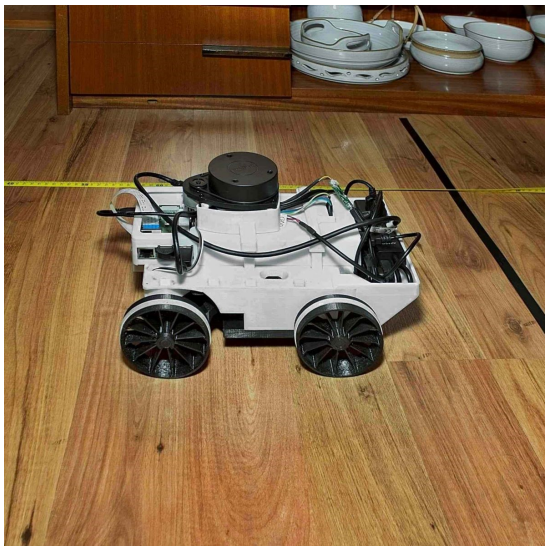
Měření překonávání schodu bylo měřeno na schodu proměnlivé velikosti. Byla zvyšována výška schodu, dokud byl robot schopen jej překonat po každém pokusu byla změřena výška schodu. Jako úspěšné překonání schodu bylo počítáno, když robot překonal schod přední i zadní nápravou. Bylo provedeno 20 měření. Data z měření se nachází v příloze C.

Maximální úhel stoupání vozidla je: $4{,}8\,\text{mm} \pm 0{,}2\,\text{mm}$

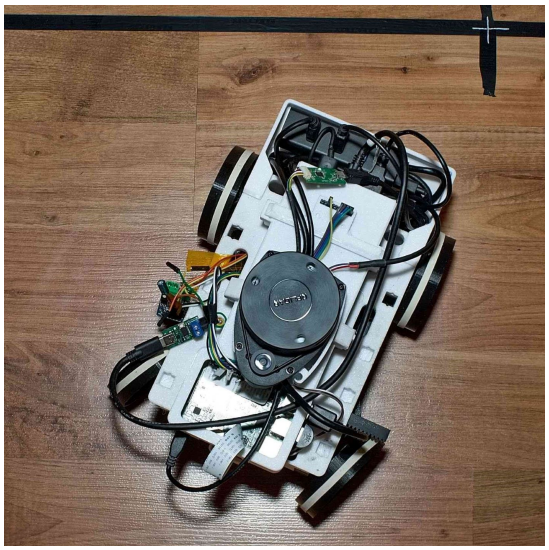
Robot dokáže úspěšně překonávat malé překážky, což umožňuje jeho bezproblémový pohyb po rovné zemi. Pro zlepšení této schopnosti by bylo potřeba vyměnit motor za výkonnější, případně využít větší kola.

Závěr testování

Při testování bylo zjištěno že se robot dokáže pohybovat po rovné zemi a překovávat drobné překážky. Problémy způsoboval nerovný terén, kde ztrácel kontakt s oběma koly nebo vysoký úhel stoupání. Navržená geometrie řízení měla oproti teoretickému návrhu odchylku. Tato odchylka byla způsobena především mechanickými vlastnostmi, které nebyly zohledněny během návrhu. Pro zlepšení těchto vlastností by bylo potřeba upravit algoritmus pro návrh hřebenového řízení, případně vyměnit motor za výkonnější.

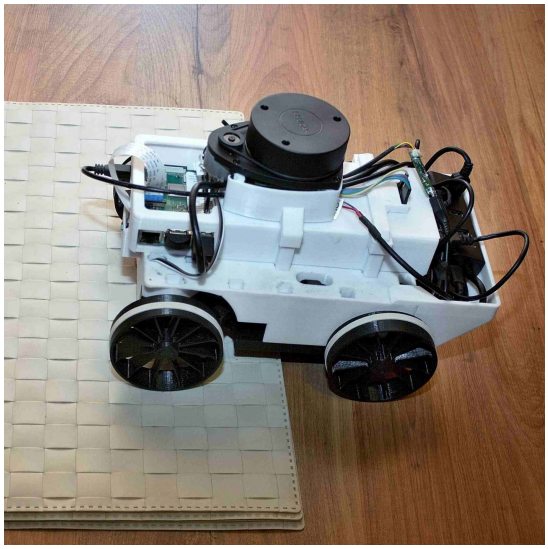


(a) Měření maximální rychlosti



(b) Měření minimálního poloměru zatočení

(c) Měření maximálního úhlu stoupání



(d) Měření schopnosti překonávat schod

Obrázek 5.23: Obrázky z měření

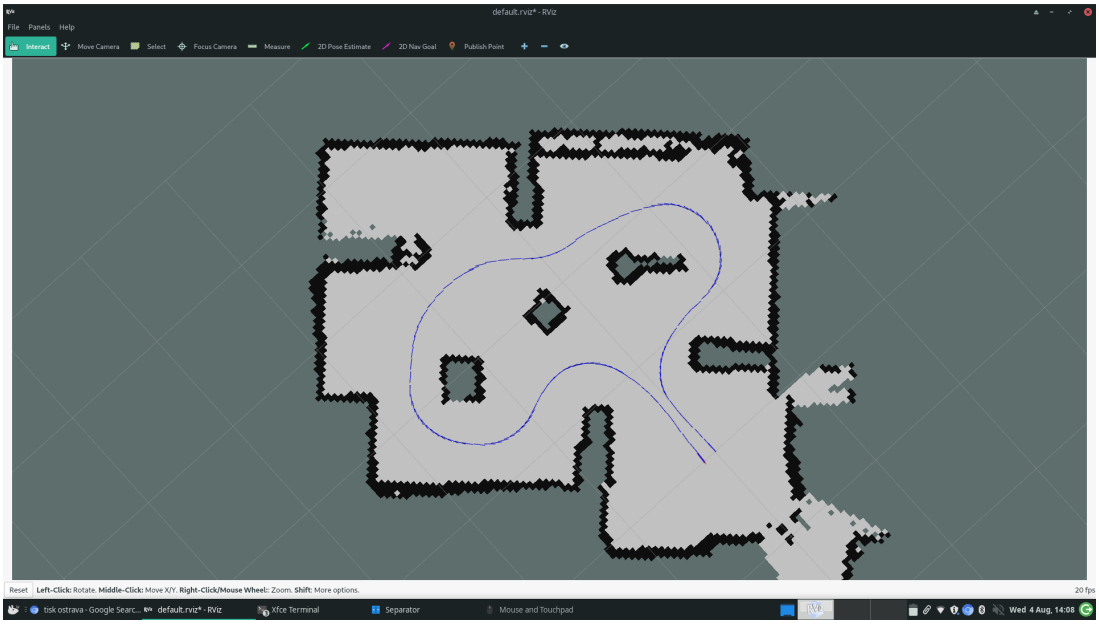
Demonstrace mapování a navigace

Byla provedena demonstrace mapování neznámého prostředí, navigace ve zmapovaném prostředí a navigace v částečně zmapovaném prostředí. Testy probíhaly v místnosti o rozměrech 3,5 m × 4 m. V místnosti byly rozmístěny překážky, které byly voleny tak, aby nebyly nižší než výška umístění laserového senzoru.

Obrázek 5.24: Fotografie místnosti kde bylo prováděna demonstrace

Demonstrace mapování vozidlem

Pro demonstrování schopností mapovat prostředí byla provedena jedna okružní jízda místností, kdy byl robot navigován dálkovým ovladačem za přímé viditelnosti robota. Po jedné okružní jízdě byla zaznamenána výsledná mapa.

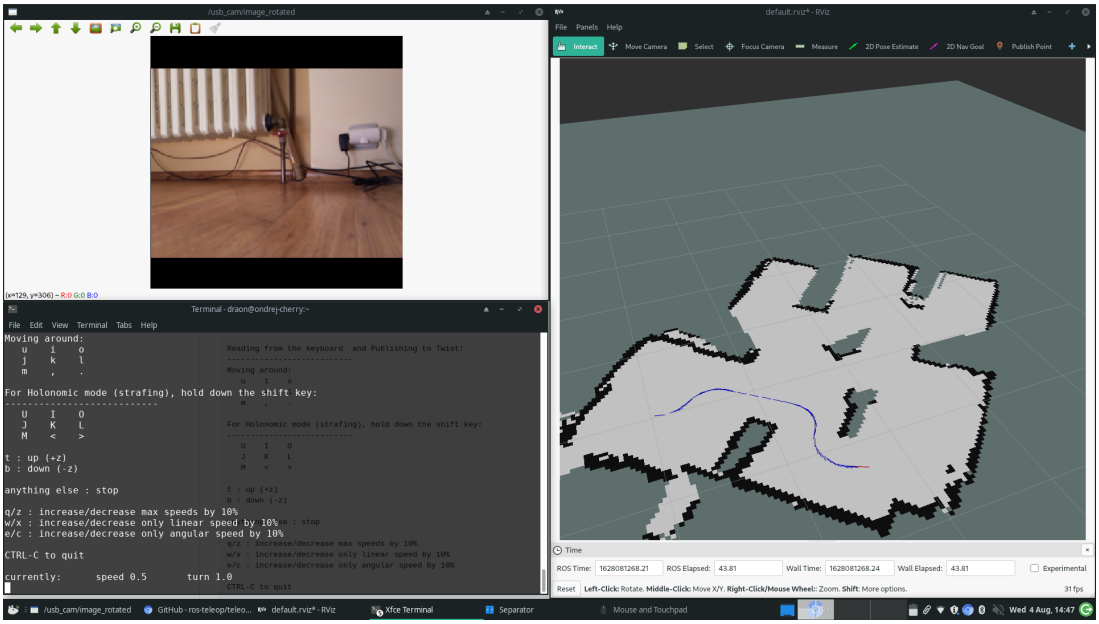


Obrázek 5.25: Výsledná mapa po okružní jízdě po místnosti

Výsledná mapa věrně odpovídá místnosti kde bylo mapováno, v mapě se nenachází zkreslení nebo jiné artefakty. Dvojitá zeď v horní části mapy je ve skutečnosti topení, "díry"ve zdech jsou způsobeny špatným zakrytím mezer v předělu místnosti.

Demonstrace teleoperace

Pro demonstrování teleoperace byl robot navigován po určené trase mimo přímou viditelnost z jiné místnosti. Pouze za pomocí generované mapy a dat z kamery.



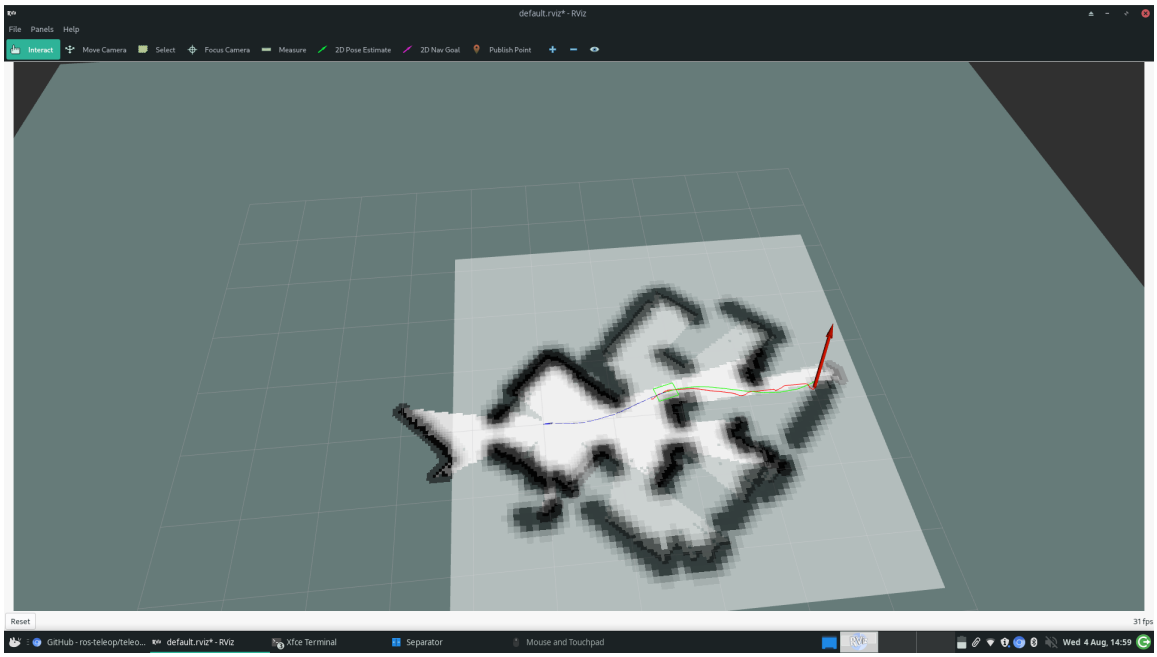
Obrázek 5.26: Teleoperace robota při demonstraci

Na obrázku 5.26 se nachází terminálová ovládací aplikace, obraz z kamery robota a aktuální získaná mapa. Na mapě se nachází uražená trasa (modře) a aktuální pozice a orientace vozidla (červená šipka).

Vozidlo bylo možné díky obrazu z kamery a mapy ovládat i mimo přímý dohled, ovládání vozidla ztěžovala přibližně 0,5 s latence obrazu oproti skutečnosti.

Demonstrace navigace v částečně zmapovaném prostředí

Poslední demonstrační úloha byla provedena tak, že byl robot umístěn do neznámého prostředí, kde mu byl zadán navigační cíl na místě, které je zakryté překážkou.



Obrázek 5.27: Navigace v částečně zmapovaném prostředí

Na obrázku 5.27 můžeme vidět robota v polovině cesty za svým cílem, na obrázku se nachází globální mapa, globální cenová mapa, lokální cenová mapa, uražená trasa robota (modře), globální naplánovaná trasa (červeně), lokální naplánovaná trasa (zeleně), obrys vozidla (zelený obdélník) aktuální polohu vozidla (malá červená šipka v obrysu vozidla) a cíl vozidla (velká červená šipka).

Kapitola 6

Závěr práce

Cílem práce bylo navrhnout implementaci/konstrukci vozidla, jeho způsob ovládání s ohledem na to, aby získalo využitím počítače nové lepší vlastnosti. Popsat možnosti konstrukce, ovládání, automatizace vozidla a demonstrovat implementaci na vhodné úloze. Tento cíl byl splněn.

Po prostudování dostupných materiálů k řízení modelářských serv, možností jejich ovládání počítačem a existujících robotických konstrukcí s RC komponenty byly nejdůležitější poznatky, ze kterých jsem vycházel při návrhu robotického vozidla, popsány v teoretické části. Byl vybrán čtyřkolový robot s Ackermannovým řízením s laserovým senzorem. Vytvořil jsem technický návrh robotického vozidla, které je snadno opakovatelně sestrojitelné s využitím "off the shelf" komponent a 3D tisku. Během návrhu bylo dbáno na to, aby sestavování robota bylo co nejméně technicky náročné. Popsal jsem možnosti konstrukce, ovládání a automatizace vozidla. Pro vozidlo byla zvolena aproximace Ackermannova řízení pomocí hřebenového řízení. Pro tento druh řízení byl vytvořen algoritmus zjišťující ideální parametry řízení pro danou geometrii vozidla pomocí optimalizační metody SHGO (Anglicky simplicial homology global optimisation, globální optimalizace zjednodušené homologie).

Na základě mého technického návrhu jsem vozidlo sestrojil. Po sestrojení vozidla byl navržen a implementován software převádějící ovládání vozidla na standardní rozhraní. To umožňuje využití mnoha implementovaných algoritmů pro ovládání a navigaci vozidla. Byly testovány jízdní vlastnosti vozidla, které byly vyhodnoceny jako vyhovující pro pohyb pro rovné ploše. Vozidlo umožňuje současnou navigaci a mapování v neznámém prostředí nebo teleoperaci. Tyto vlastnosti byly předvedeny na demonstračních úlohách.

Dalším pokračováním práce by mohla být úprava optimalizačního algoritmu pro zlepšení mechanických vlastností řízení, ladění parametrů odometrie a navigace, přidání dalších senzorů pro detekování malých překážek, využití dat z kamery pro navigaci v prostředí nebo připevnění dalších aktuátorů pro interakci s prostředím.

Literatura

[1] Fried, L. *Adafruit 16-Channel PWM/Servo HAT & Bonnet for Raspberry Pi* [online]. Leden 2015 [cit. 2021-07-30]. Dostupné z: https://learn.adafruit.com/adafruit-16-channel-pwm-servo-hat-for-raspberry-pi/downloads.

[2] Bae, J.-H., Park, S.-W., Park, J.-H. et al. Development of a low cost anthropomorphic robot hand with high capability. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. Vilamoura: IEEE, 2012, s. 4776–4782. DOI: 10.1109/IROS.2012.6386063. ISBN 978-1-4673-1736-8.

[3] Bishop, B., Piepmeier, J., Piper, G. et al. The use of low-cost RC servos in a robotics curriculum. *AAAI Spring Symposium - Technical Report*. Palo Alto: The AAAI Press. Leden 2004, sv. 1, SS-04-01, s. 52–56.

[4] Bruno, S. a Oussama, K. *Springer Handbook of Robotics*. 2. vyd. New York City: Springer International Publishing, 2016. ISBN 978-3-319-32550-7,978-3-319-32552-1.

[5] Collett, T., Macdonald, B. a Gerkey, B. Player 2.0: Toward a Practical Robot Programming Framework. *Proceedings of the 2005 Australasian Conference on Robotics and Automation, ACRA 2005*. Sydney: [b.n.]. Srpen 2008.

[6] Deingruber, O. *Model vozidla řízený počítačem*. Brno, 2019. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií.

[7] Endres, S., Sandrock, C. a Focke, W. A simplicial homology algorithm for Lipschitz optimisation. *Journal of Global Optimization*. Heidelberg: Springer Netherlands. Říjen 2018, sv. 72. DOI: 10.1007/s10898-018-0645-y.

[8] Gandhinathan, R. a Joseph, L. *ROS Robotics Projects: Build and control robots powered by the Robot Operating System, machine learning, and virtual reality, 2nd Edition*. Birmingham: Packt Publishing, 2019. ISBN 9781838645199.

[9] Gerkey, B. a Vaughan, R. The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems. *Proceedings of the International Conference on Advanced Robotics*. Coimbra: IEEE. Srpen 2003.

[10] Ishiwatari, S. *Rapiro* [online]. Květen 2013 [cit. 2021-07-30]. Dostupné z: http://www.rapiro.com/.

[11] Kruth, J.-P., Leu, M. a Nakagawa, T. Progress in Additive Manufacturing and Rapid Prototyping. *CIRP Annals - Manufacturing Technology*. Amsterdam: Elsevier Ltd. Leden 1998, sv. 47, s. 525–540. DOI: 10.1016/S0007-8506(07)63240-5.

[12] Masood, M. U. a Haghshenas Jaryani, M. A Study on the Feasibility of Robotic Harvesting for Chile Pepper. *Robotics*. Thousand Oaks: SAGE Publications. 2021, sv. 10, č. 3. DOI: 10.3390/robotics10030094. ISSN 2218-6581. Dostupné z: https://www.mdpi.com/2218-6581/10/3/94.

[13] *The pigpio library* [online]. Abyz.me.uk, říjen 2013 [cit. 2021-07-30]. Dostupné z: http://abyz.me.uk/rpi/pigpio/index.html.

[14] *The Player Project* [online]. playerproject [cit. 2021-07-30]. Dostupné z: https://playerproject.github.io/.

[15] Průša J., B. M. *Basics of 3D printing* [online]. Praha: Prusa Research s.r.o., 2019. Dostupné z: https://www.prusa3d.com/wp-content/uploads/basics-of-3D-printing.pdf.

[16] *ROS.org | Powering the world's robots* [online]. ROS.org [cit. 2021-07-30]. Dostupné z: https://www.ros.org/.

[17] *Documentation - ROS Wiki* [online]. San Francisco: ROS.org. Dostupné z: http://wiki.ros.org/.

[18] Thomas, K. *SMARS modular robot by tristomietitoredeituit* [online]. Thingiverse.com, listopad 2017 [cit. 2021-07-30]. Dostupné z: https://www.thingiverse.com/thing:2662828.

[19] *Tinkerkit braccio robot* [online]. Arduino SA, říjen 2005 [cit. 2021-07-30]. Dostupné z: https://store.arduino.cc/tinkerkit-braccio-robot.

[20] Todoran, H. G. *Optimal Local Path-Planning and Control for Mobile Robotics*. Vídeň, 2018. Disertační práce.

[21] *TurtleBot3* [online]. Robotis [cit. 2021-07-30]. Dostupné z: https://emanual.robotis.com/docs/en/platform/turtlebot3/overview/.

[22] Vlk, F. *Podvozky motorových vozidel*. 1. vyd. Brno: František Vlk, 2006. ISBN 9788023964646.

[23] Waymo. *Waypoint - The official Waymo blog: Introducing Waymo's suite of custom-built, self-driving hardware* [online]. Únor 2017 [cit. 2021-07-30]. Dostupné z: https://blog.waymo.com/2019/08/introducing-waymos-suite-of-custom.html.

[24] *Waymo — Wikipedia, The Free Encyclopedia* [online]. San Francisco: Wikipedia, 2021 [cit. 2021-07-30]. Dostupné z: http://en.wikipedia.org/w/index.php?title=Waymo&oldid=1033435030.

[25] *3D printing — Wikipedia, The Free Encyclopedia*. San Francisco: Wikipedia, 2021 [cit. 2021-01-15]. Dostupné z: http://en.wikipedia.org/w/index.php?title=3D%20printing&oldid=1000505775.

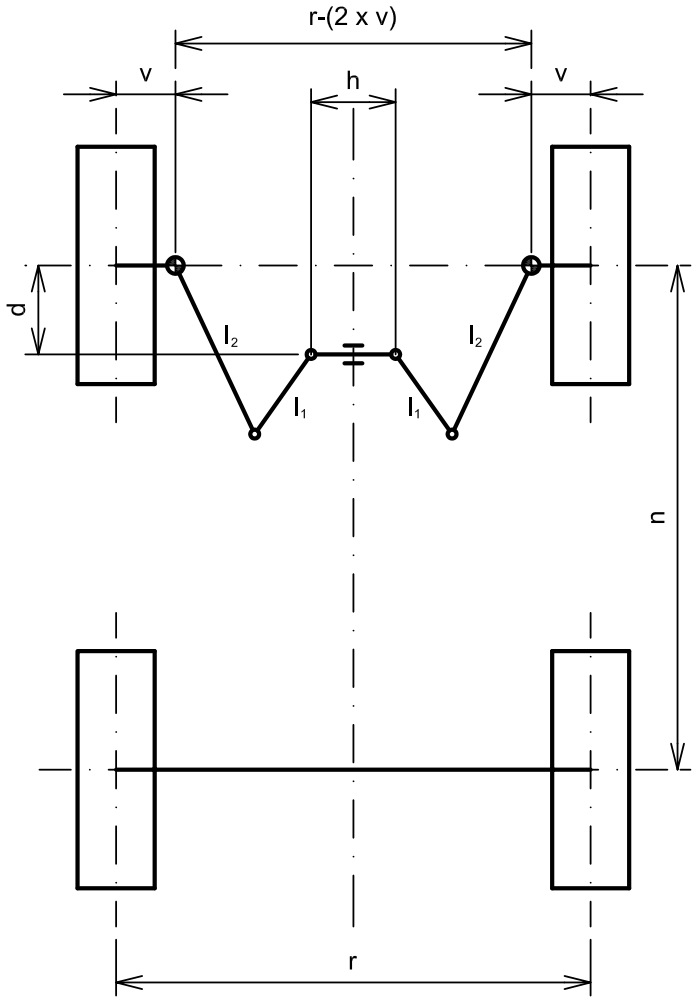
[26] *Mobile robot — Wikipedia, The Free Encyclopedia* [online]. San Francisco: Wikipedia, 2021 [cit. 2021-01-15]. Dostupné z: http://en.wikipedia.org/w/index.php?title=Mobile%20robot&oldid=996559778.

[27] *Vestavěný systém — Wikipedia, The Free Encyclopedia* [online]. San Francisco: Wikipedia, 2021 [cit. 2021-07-30]. Dostupné z: http://cs.wikipedia.org/w/index.php?title=Vestav%C4%9Bn%C3%BD%20syst%C3%A9m&oldid=19848000.

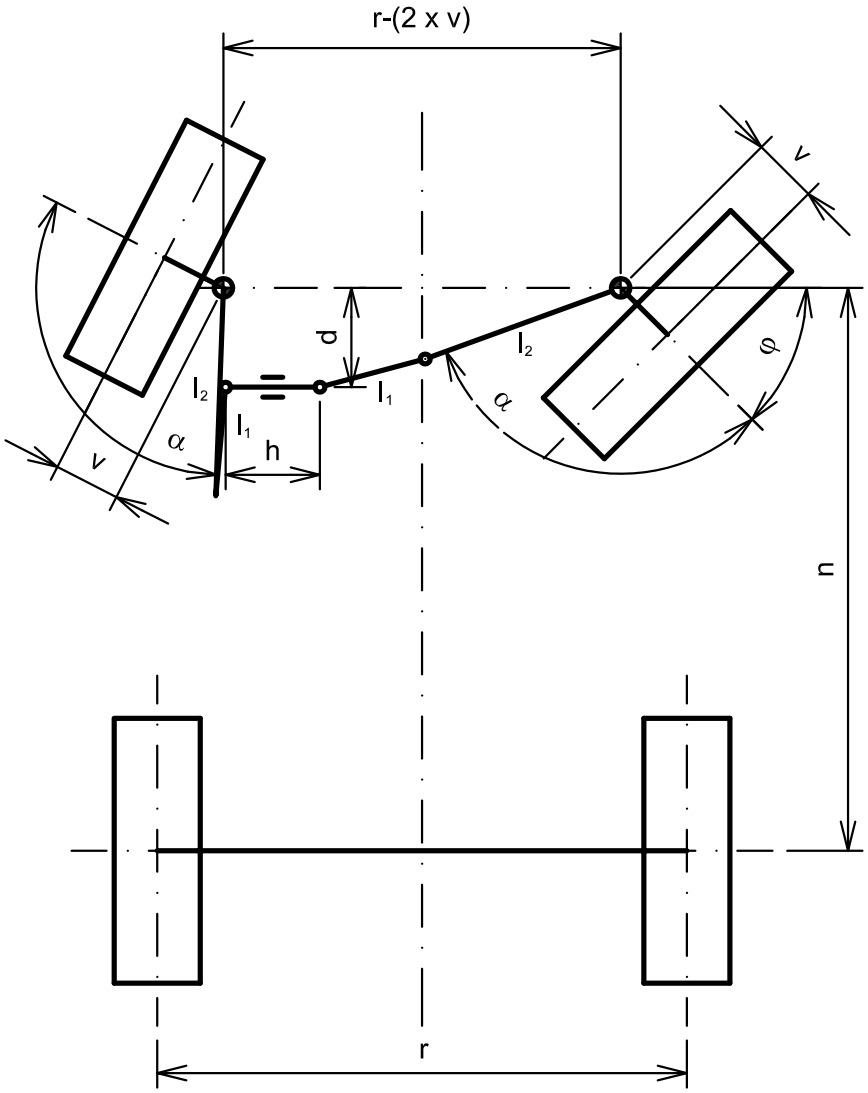
[28] Zhang, H., Gonzalez Gomez, J., Me, Z. et al. Development of a Low-cost Flexible Modular Robot GZ-I. In:. Xi'an: IEEE, Srpen 2008, s. 223 – 228. DOI: 10.1109/AIM.2008.4601663. ISBN 978-1-4244-2494-8.

Příloha A

Výsledná geometrie hřebenového řízení



Obrázek A.1: Výsledná geometrie získaná optimalizační metodou, kola rovně



Obrázek A.2: Výsledná geometrie získaná optimalizační metodou, maximální zatočení kol

Příloha B

Implementační detaily

V této kapitole naleznete další implementační detaily blíže popisující fungování robotického vozidla a jeho zapojení.

Struktura Aplikace v ROS a konfigurace systému

Tato sekce popisuje strukturu aplikace ROS její uzly, jejich vzájemnou komunikaci a seznam využitých balíčků.

Popis komunikace uzlů v ROS

Cílem implementace bylo podporovat soustavu balíčků s názvem navigation, které poskytují plánování trasy a mapování. Pro správné fungování potřebují obdržet zprávy typu `tf/Message` (strom transformací), `nav_msgs/Odometry` (odometrie), `sensor_msgs/LaserScan` nebo `sensor_msg/PointCloud` (laser scan nebo mrak bodů ze senzorů) a volitelně `nav_msgs/GetMap` (server s mapou). Dále musí robot podporovat ovládání pomocí zpráv `geometry_msgs/Twist` (příkaz k pohybu), konkrétně rychlost v ose x, y a úhlovou rychlost θ.

Pro ovládání pohybu robota byl využit framework `ros_control`, který umožňuje mapovat vlastní aktuátory na standardní rozhraní a umožňuje tak využít velkou knihovnu standardních ovladačů. Ovladač `ackermann_steering_controller` nabízí ovládání podvozků s ackermannovou geometrií definovaných pomocí virtuálního zadního a předního kola ve středu vozidla. Tento balíček po správné konfiguraci poskytuje řízení virtuálního předního a zadního kola a publikuje zprávy s odometrií.

Pro ovládání natočení kol je využito zpráv, konkrétně `std_msgs/Float32`, které odesílá implementace ovladače aktuátoru uzlu, který se stará o komunikaci s hardware pomocí vstupně/výstupních pinů (dále IO uzel). Pro ovládání rychlosti motoru je využito PID regulátoru. Jedná se o softwarový PID regulátor poskytovaný v ROS balíčku **PID**, tento balíček dostává zprávy s požadovanou rychlostí od implementace ovladače aktuátoru, aktuální rychlostí od I/O uzlu. Na základě obdržených zpráv a nastavených parametrů řídí rychlost motoru pomocí úsilí, reprezentující střídu PWM signálu, které odesílá I/O uzlu pomocí zpráv.

I/O uzel se stará o ovládání motoru poháněnícího zadní kola a serva řídícího úhel natočení předních kol. Tato aplikace přijímá zprávy typu `std_msgs/Float32` pro úhel natočení kol a `std_msgs/Float64` pro úsilí ovládající motor. Publikuje zprávy typu `std_msgs/Float32` pro aktuální nastavený úhel kol a `std_msgs/Float64` pro aktuální rychlost motoru.

Strom transformací publikuje balíček `robot_state_publisher` který čte statické transformace z konfiguračního souboru ve formátu urdf.

Pro zpřesnění odometrie vozidla je využíváno kombinace odometrie z podvozku společně se zprávami z IMU (anglicky Inertial Measurment Unit, inerciální senzor). Tyto zprávy jsou zpracovány kalmanovým filtrem který implementuje balíček `robot_localization`.

Pro získání dat z IMU je využito balíčku `ros-driver_mpu9250` dostupného z repozitáře git[1]. Jelikož balíček starající se o komunikaci s IMU definuje nestandardní formát zpráv `sensor_msgs_ext`, byly upraveny zdrojové soubory balíčku tak aby používaly komptabilní zprávy. Balíček `robot_localization` vyžaduje pro zprávy z IMU jeden z typů:

- `geometry_msgs/PoseWithCovarianceStamped`
- `geometry_msgs/TwistWithCovarianceStamped`
- `sensor_msgs/Imu`

Tyto zprávy obsahují informaci o orientaci, tu ale čip MPU9250 neposkytuje. To je vyřešeno pomocí balíčku `imu_tools`, konkrétně `imu_filter_madgwick`, který pomocí lineárního, úhlového zrychlení a síly magnetického pole pozici senzoru aproximuje.

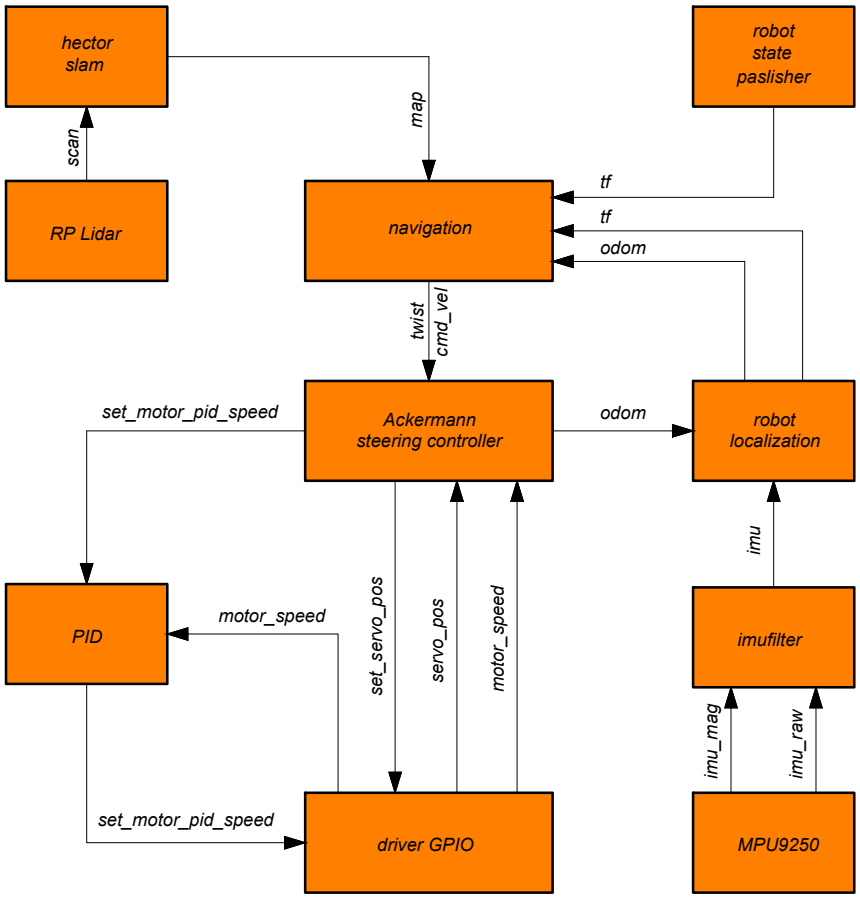
Pro získávání dat z laserového skenneru je využit balíček poskytovaný výrobcem. Tento balíček byl sestaven ze zdrojových kódů.

Současnou lokalizaci a mapování (SLAM) zajišťuje balíček `hector_mapping`. Balíček získává data z odometrie a laserového skeneru poskytuje funkcionalitu mapového serveru a zprávy transformace mezi rámcem mapy odometrií vozidla.

Pro navigaci bylo využito souboru balíčků `navigation`. Protože robotické vozidlo má nestandardní podvozek neumožňující otáčení na místě, nebylo možné využít výchozí plánovací algoritmy. Proto byl využit zásuvný modul `teb_local_planner`, který při plánování trasy zohledňuje minimální poloměr zatáčení. Pro globální cenovou mapu byla využita mapa s rozšířením překážek, globální cenová mapa využívá dat z laserového skeneru a rozšíření těchto bodů. Pro úsporu výpočetního času jsou překážky lokální cenové mapy převáděny na jednoduché geometrie pomocí zásuvného modulu `costmap_converter`.

Publikování zpráv z kamery vozidla poskytuje balíček `usb_cam`. Jelikož je kamera na vozidle umístěna v obrácené orientaci, je jeho otočení provedeno pomocí balíčku `image_rotate`. Tento balíček využívá velké množství výpočetních zdrojů, proto je využívám pouze pří teleoperaci vozidla, kdy není potřeba plánovat trasu vozidla.

[1] dostupné z: https://github.com/pcdangio/ros-driver_mpu9250



Obrázek B.1: Struktura uzlů v ROS

Propojení s ros_control

Pro propojení s balíčkem `ros_control` byla vytvořena implementace rozhraní `hardware_interface::RobotHW`, která se při ovládání aktuátorů stará o komunikaci s hardware. Byly konfigurovány jejich limity a nastaveny ovladače využívající daný hardware. Jelikož je komunikace s hardware implementována pomocí zpráv v ROS jedná se o přeposílání zpráv uzlu starajícího se o ovládání hardware.

Ovladač `ackermann_steering_controller` vyžaduje rozhraní typu velocity interface pro kontrolu rychlosti zadního kola a rozhraní typu position interface pro kontrolu natočení předních kol.

Pro implementaci uvedených rozhraní byl vytvořen nový uzel. Tento uzel se stará o komunikaci s ostatními uzly komunikujícími s hardware. V třídě implementující rozhraní `hardware_interface::RobotHW` jsou zaregistrovány dvě rozhraní typu `hardware_interface::JointStateHandle` pro aktualizace rychlosti motoru a natočení předních kol. Poté jsou registrovány rozhraní `hardware_interface::VelocityJointInterface` pro řízení rychlosti motoru a rozhraní `hardware_interface::PositionJointInterface` pro ovládání pozice předních kol.

Uzel běží v nekonečné smyčce, kdy v pravidelných intervalech aktualizuje stav rychlosti motoru, pozici předních kol a odesílá řídící zprávy pro jejich ovládání.

Úprava balíčku ovladače inerciálního senzoru

Balíček `ros-mpu9250_driver` využívá nestandardní zprávy definované v balíčku `ros-sensor_msgs_ext`. Pro další využití především balíčkem `robot_localization` je potřeba zprávy typu `sensor_msgs/Imu`. Dalším problémem je, že balíček poskytuje pouze zprávy o síle magnetického pole nikoliv už zprávy o orientaci senzoru. Proto bylo využito balíčku `imu_filter_madgwick`, který filtruje data a aproximuje orientaci senzoru pomocí měřeného lineárního, úhlového zrychlení a měřeného magnetického pole. Jako zdroj dat využívá zprávy typu `sensor_msgs/Imu` a `sensor_msgs/MagneticField`. Proto byly upraveny zdrojové soubory balíčku `ros-mpu9250_driver` tak, aby publikoval potřebné zprávy.

Konfigurace systému

Pro jednodeskový počítač Raspberry Pi 3B+ jsem zvolil operační systém Linux, distribuci Ubuntu Server 20.04.2 LTS. Tato distribuce byla zvolena jelikož se jedná o oficiálně podporovanou distribuci pro ROS Noetic. Tento operační systém byl nainstalován v 64-bitové verzi.

Do repozitáře aplikací apt byly přidány balíčky ROS a byly nainstalován základní balíček `ros_base`. V domovském adresáři byla vytvořena složka `catkin_ws` která byla nastavena jako pracovní adresář pro nástroj catkin. Tento adresář obsahuje všechny vytvořené balíčky a balíčky které bylo potřeba sestavit ze zdrojového kódu. Pro ulehčení práce na počítači byl upraven soubor .bashrc který při vždy při tvorbě nového terminálu nastaví systémové proměnné pro práci s ROS.

Jelikož je sestavování aplikací výpočetně velmi náročné a pro překlad aplikací může být vyžadováno velké množství operační paměti, byl vytvořen soubor swap o velikosti 2 GB.

Pro jednoduchý síťový přístup byl DHCP klient nastaven tak ať žádá o pevnou IP adresu. Toto umožňuje nastavit systémové proměnné tak, aby bylo možné se připojit k prostředí ROS i ze vzdáleného systému.

Byla sestavena a nainstalována knihovna Pigpio. Byly upraveny konfigurační soubory systému tak aby se server knihovny spouštěl vždy při zapnutí počítače.

Pro přístup ke kameře byl upraven konfigurační soubor `/boot/firmware/config.txt`, kde bylo přidána konfigurace `start_x=1`.

Seznam použitých balíčků ROS

V projektu byly využity následující balíčky z repozitáře ROS:

- `sensor_msgs`
- `pid`
- `robot_localization`
- `ros_control`
- `robot_state_publisher`
- `joint_state_publisher`
- `joint_state_controller`
- `velocity_controllers`
- `position_controllers`
- `ackermann_steering_controller`
- `imu_tools`
- `hector_slam`
- `teb_local_planner`
- `costmap_converter`
- `usb_camera`
- `image_rotate`

Dále byly využity následující balíčky:

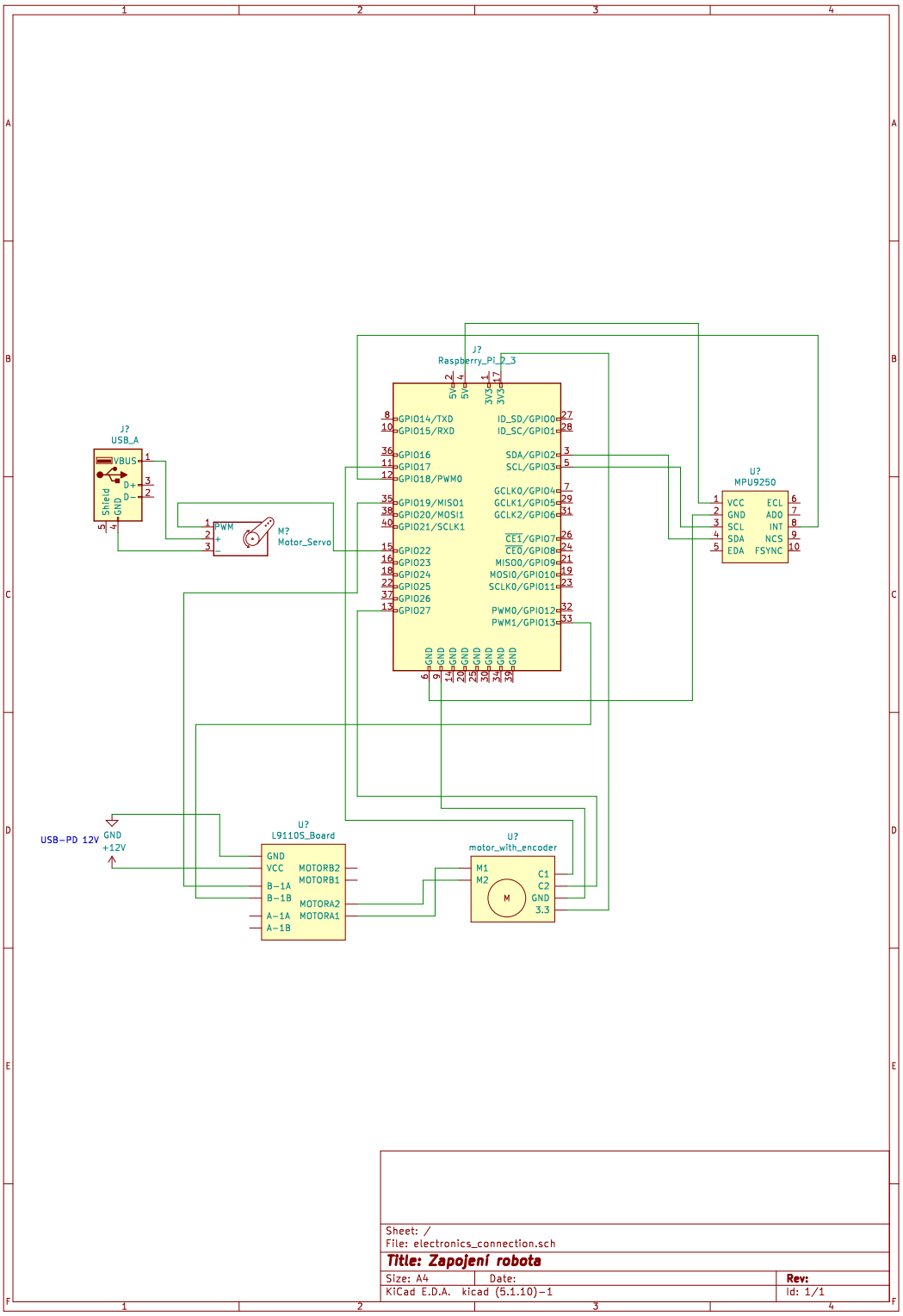
- `ros-sensor_msgs_ext`[2]
- `ros-driver_mpu9250`[3]
- `rplidar_ros`[4]

[2] dostupné z https://github.com/pcdangio/ros-sensor_msgs_ext
[3] dostupné z https://github.com/pcdangio/ros-driver_mpu9250
[4] dostupné z https://github.com/Slamtec/rplidar_ros

Elektrické zapojení komponentů Raspberry Pi 3B+

číslo pinu	název pinu	komponenta	pin zařízení
1	3V3 power	-	-
2	5V power	-	-
3	GPIO 2 (SDA)	MPU9250	SDA
4	5V power	MPU9250	VCC
5	GPIO 3 (SCL)	MPU9250	SCL
6	Ground	MPU9250	GND
7	GPIO 4	-	-
8	GPIO 14	-	-
9	Ground	enkodér	GND
10	GPIO 15	-	-
11	GPIO 17	enkodér	C2
12	GPIO 18	MPU9250	INT
13	GPIO 27	enkodér	C1
14	Ground	-	-
15	GPIO 22	servo	SIG
16	GPIO 23	-	-
17	3V3 power	enkodér	3.3V
18	GPIO 24	-	-
19	GPIO 10 (MOSI)	-	-
20	Ground	servo	GND
21	GPIO 9 (MISO)	-	-
22	GPIO 25	-	-
23	GPIO 11 (SCLK)	-	-
24	GPIO 8 (CE0)	-	-
25	Ground	-	-
26	GPIO 7 (CE1)	-	-
27	GPIO 0 (ID_SD)	-	-
28	GPIO 1 (ID_SC)	-	-
29	GPIO 5	-	-
30	Ground	-	-
31	GPIO 6	-	-
32	GPIO 12 (PWM0)	-	-
33	GPIO 13 (PWM1)	L9110S	A-1B
34	Ground	-	-
35	GPIO 19 (PCM_FS)	L9110S	A-1A
36	GPIO 16	-	-
37	GPIO 26	-	-
38	GPIO 20 (PCM_DIN)	-	-
39	Ground	-	-
40	GPIO 21 (PCM_DOUT)	-	-

Tabulka B.1: Tabulka zapojení I/O pinů na Raspberry Pi 3B+



Obrázek B.2: Zapojení Raspberry Pi 3B+

Příloha C

Hodnoty měření

Tato kapitola obsahuje naměřené data získané při testování vozidla.

P.č. měření	Naměřená hodnota Čas [s]	Odchylka aritmetického průměru [s]
1	19.30	0.54
2	18.81	0.05
3	18.77	0.01
4	18.83	0.07
5	18.85	0.09
6	18.76	0.00
7	19.11	0.35
8	18.84	0.08
9	18.59	-0.17
10	18.70	-0.06
11	18.51	-0.25
12	18.67	-0.09
13	18.75	-0.01
14	18.74	-0.02
15	18.66	-0.10
16	18.80	0.04
17	18.53	-0.23
18	18.66	-0.10
19	18.74	-0.02
20	18.62	-0.14

Tabulka C.1: Hodnoty maximální rychlosti

- $\overline{v} = 0{,}2132\,\mathrm{m\,s^{-1}}$
- $\sigma = 0{,}0005\,\mathrm{m\,s^{-1}}$

Průměr zatáčení

P.č. měření	Naměřená hodnota - Min. průměr dráhy zadního kola [mm]	Odchylka aritmetického průměru [mm]
1	914	30
2	936	52
3	945	61
4	930	46
5	927	43
6	950	66
7	933	49
8	933	49
9	942	58
10	925	41
11	863	-21
12	842	-42
13	918	34
14	848	-36
15	815	-69
16	840	-44
17	836	-48
18	754	-130
19	777	-107
20	848	-36

Tabulka C.2: Hodnoty průměru zatáčení

- $\overline{d} = 844\,\mathrm{mm}$
- $\sigma = 13\,\mathrm{mm}$

Maximální úhel stoupání

P.č. měření	Naměřená hodnota - úhel stoupání	Odchylka aritmetického průměru
1	4.0	-1.0
2	5.0	0.0
3	5.0	0.0
4	6.0	1.0
5	5.0	0.0
6	6.0	1.0
7	5.0	0.0
8	5.0	0.0
9	4.0	-1.0
10	5.0	0.0
11	5.0	0.0
12	6.0	1.0
13	4.0	-1.0
14	5.0	0.0
15	5.0	0.0
16	6.0	1.0
17	5.0	0.0
18	4.0	-1.0
19	5.0	0.0
20	5.0	0.0

Tabulka C.3: Hodnoty maximálního úhlu stoupání

- $\overline{\alpha} = 5{,}0°$
- $\sigma = 0{,}1°$

Schopnost překonávat schod

P.č. měření	Naměřená hodnota - max. výška překážky [mm]	Odchylka aritmetického průměru [mm]
1	3.42	-1.38
2	5.97	1.17
3	5.09	0.29
4	5.23	0.43
5	5.19	0.39
6	5.28	0.48
7	3.09	-1.71
8	5.44	0.64
9	4.94	0.14
10	5.25	0.45
11	4.92	0.12
12	4.87	0.07
13	3.59	-1.21
14	4.87	0.07
15	5.65	0.85
16	5.29	0.49
17	4.97	0.17
18	3.23	-1.57
19	5.13	0.33
20	5.28	0.48

Tabulka C.4: Hodnoty výšky překonaného schodu

- $\overline{h} = 4{,}8\,\mathrm{mm}$
- $\sigma = 0{,}2\,\mathrm{mm}$