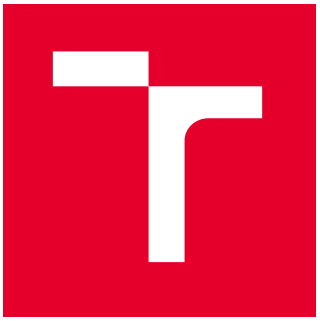
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2024 Bc. Michal Koláčný



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

HARDWAROVÁ PLATFORMA PRO PROPAGAČNÍ ROBOTICKÉ VOZÍTKO

A HARDWARE PLATFORM FOR A PROMOTIONAL ROBOTIC VEHICLE

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Michal Koláčný

AUTHOR

VEDOUCÍ PRÁCE — Ing. Miroslav Jirgl, Ph.D.

SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Michal Koláčný ***ID:*** 220991

Ročník: 2 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Hardwarová platforma pro propagační robotické vozítko

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je realizace funkčního vzorku robotického vozítka pro propagační účely zadávající firmy za účelem studie proveditelnosti řešení. To předpokládá hardwarový návrh a realizaci včetně implementace řídicího software pro pohon vozítka, zpracování obrazových dat a komunikačního rozhraní.

1. Popište koncepci robotického vozítka.
2. Navrhněte vhodnou konstrukci a návrh zadokumentujte.
3. Vyberte vhodné elektrické komponenty pro osazení vozítka.
4. Realizujte elektrické zapojení jednotlivých komponent.
5. Implementujte nezbytný řídicí a komunikační software.
6. Otestujte výsledné řešení, analyzujte proveditelnost možného prototypu a doplňte vhodná doporučení.

DOPORUČENÁ LITERATURA:

SICILIANO, B. a O. KHATIB. Handbook of Robotics [online]. Springer-Verlag Berlin Heidelberg, 2008 [cit. 2023-09-11]. ISBN 978-3-540-30301-5. Dostupné z: https://www.academia.edu/26665877/Handbook_Springer_of_Robotics

Termín zadání: 5.2.2024 ***Termín odevzdání:*** 15.5.2024

Vedoucí práce: Ing. Miroslav Jirgl, Ph.D.

doc. Ing. Petr Fiedler, Ph.D.
předseda rady studijního programu

Abstrakt

Tato práce slouží jako test proveditelnosti pro konstrukci robotického vozítka, kde jeho autonomní pohyb bude zajištěn za pomoci kamery a snímání QR kódu, pro společnost B:TECH. Konkrétně pojednává o součinnosti hardwarové platformy robotického vozítka společně s vybranými elektrickými komponenty. Je zde popsán návrh konstrukce vytištěný na 3D tiskárně a jeho elektrické a mechanické komponenty umožňující pohyb. Dále je práce zaměřena na výběr vhodných elektrických komponentů týkající se bezpečnosti provozu v podobě snímačů. Nezbytnou součástí je také řízení a koordinace pohybu vozítka pomocí řídících mikrokontrolerů. Konkrétní komponenty a výbava je popsána v následujících kapitolách. Výsledkem práce je vytvoření hardwarového konceptu autonomního robotického vozítka.

Klíčová slova

QR kód, HW, robotické vozítko, omni kola, Raspberry

Abstract

This thesis serves as a feasibility test for the construction of a robotic vehicle, where its autonomous movement will be provided by a camera and QR code scanning, for the company B:TECH. Specifically, it deals with the interaction of the hardware platform of the robotic vehicle together with selected electrical components. It describes the design of the structure printed on a 3D printer and its electrical and mechanical components enabling movement. Furthermore, the work focuses on the selection of suitable electrical components related to the safety of operation in the form of sensors. A necessary part is also the control and coordination of the movement of the vehicle using control microcontrollers. Specific components and equipment are described in the following chapters. The result of the thesis is the creation of a hardware concept of an autonomous robotic vehicle.

Keywords

QR code, HW, robotic vehicle, omni wheels, Raspberry

Bibliografická citace

KOLÁČNÝ, Michal. Hardwarová platforma pro propagační robotické vozítko [online]. Brno, 2024 [cit. 2024-05-14]. Dostupné z: https://www.vut.cz/studenti/zav-prace/detail/160031. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Miroslav Jirgl.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	*Bc. Michal Koláčný*
VUT ID studenta:	*220991*
Typ práce:	*Diplomová práce*
Akademický rok:	*2023/24*
Téma závěrečné práce:	*Hardwarová platforma pro propagační robotické vozítko*

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: 15. května 2024 ----------

podpis autora

Poděkování

Děkuji vedoucímu diplomové práce Ing. Miroslavu Jirglovi, Ph.D. za odbornou pomoc a poskytnutí cenných rad a konzultace při tvorbě mé diplomové práce. Dále bych rád poděkoval konzultantovi Ing. Jiřímu Klimešovi za účinnou pomoc při zpracování práce.

V Brně dne: 15.května 2024 ----------

podpis autora

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Úvod

Diplomová práce byla realizována za pomoci společnosti, zabývající se automatizací procesů, B:TECH se sídlem v Havlíčkově Brodě. Zadané téma spadá do odvětví procesní automatizace a robotiky.

Hlavní cíl práce je zaměřen na tvorbu autonomního systému, kterým je robotické vozítko tzv. ARfield (Autonomous Robotic field). Autonomní systém je takový systém, který dokáže plnit požadované úkoly bez lidského zásahu. To znamená, že je schopen se samostatně rozhodovat, dle jeho vnitřní implementace. Autonomní systémy se dnes hojně využívají v oblasti výroby, kde nahrazují přepravní mechanismy, které jsou ovládány člověkem. Systémy ulehčují lidskou práci a umožňují větší bezpečnost při manipulaci s výrobky. Nezbytnou součástí autonomních systémů je jejich hardwarová platforma, která představuje vozítka pohybující se po výrobních halách. Tato vozítka mají specifickou konstrukci, která se odvíjí od konkrétního typu přepravovaného materiálu. Vyrobit jedno z těchto typů vozítek je cílem celé práce, zde se však jedná jen o studii proveditelnosti, tudíž nelze jednoznačně říci, že daný model bude možno použít v reálném výrobním provozu.

Cílem diplomové práce je teoreticky se seznámit s možnostmi návrhu a vzhledy konstrukcí autonomních systémů. Dále je úkolem navrhnout konkrétní konstrukci s následnou realizací. Poté vybrat a osadit vozítko vhodnými elektrickými komponenty nutnými pro oživení a bezpečnost při pohybu. Následně realizovat elektrické zapojení těchto komponentů. Poté vytvořit a implementovat firmware nezbytný pro řízení vozítka a komunikaci s nadřazeným řídícím systémem. Posledním bodem práce je zhodnotit výsledek s cílem analyzovat způsob řešení a navrhnout možná vylepšení.

Tato práce se skládá z osmi hlavních kapitol, kde každá kapitola je následně členěna na podkapitoly, které blíže specifikují danou problematiku. První kapitola se zabývá všeobecným principem robotických vozítek se zaměřením na konkrétní aplikace a využití. Dále je v této kapitole popsán princip pohybu vozítka se zaměřením na snímání QR kódů. Druhá kapitola pojednává o návrhu konstrukce, která je základem pro hardwarovou část aplikace. Třetí kapitola se zabývá výběrem vhodných řídících a elektrických komponentů nezbytných pro pohyb a funkci robotického vozítka. Následující čtvrtá kapitola je zaměřena na popis použitých omni kol vozítka v dané aplikaci. Dále je zde popsána realizace konstrukce s dodatečnými úpravami v návrhu. V páté kapitole je popsán návrh zapojení elektrických komponentů, společně s realizací desky plošného spoje. Šestá kapitola pojednává o tvorbě a implementaci nezbytného řídícího softwaru. Následující sedmá kapitola je zaměřena na zhodnocení dosažených výsledků s možností dodatečných úprav. V poslední kapitole je zhodnocení celkové diplomové práce.

1. KONCEPT ROBOTICKÉHO VOZÍTKA

Následující kapitola popisuje základní koncepci robotického vozítka se zaměřením na hardwarovou část vozítka. V podkapitole 1.1 je blíže popsán základní princip funkce úlohy robotického vozítka. Podkapitola 1.2 pojednává o nezbytném řídícím softwaru, kterým bude vozítko vybaveno. Další podkapitola 1.3 se zabývá typickými aplikacemi použití autonomních robotických vozítek. Poslední podkapitola 1.4 je zaměřena na princip pohybu vozítka pomocí QR kódů. Jsou zde popsány vlastnosti QR kódů. Dále popisuje základní informace o QR kódech, jejich základní strukturu a typy QR kódů. Poslední část této podkapitoly je zaměřena na konkrétní rozložení a použité QR kódy v dané aplikaci.

1.1 Princip funkce

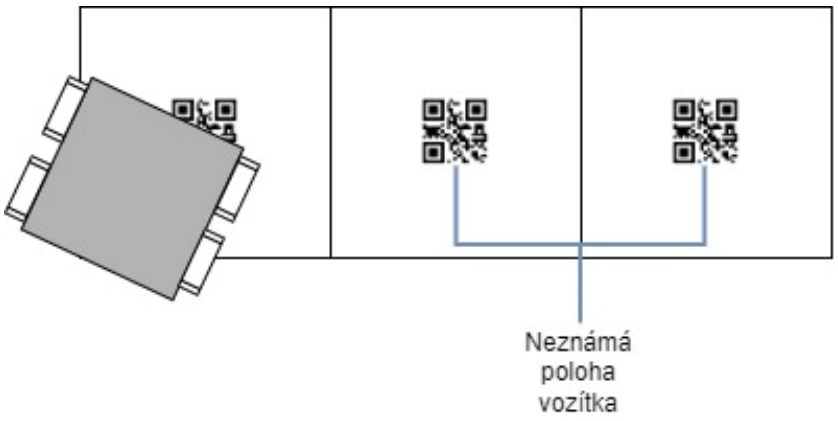
Zadaná práce pojednává o návrhu robotického vozítka, které bude mít za úkol přepravovat potřebné materiály z bodu A do bodu B. Vozítko bude navrženo pro testovací účely a konečný výrobek není nikterak specifický pro konkrétní úlohu. Cílem vozítka je pohybovat se po hale (místnosti) dle předem vytvořených QR kódů, které budou vytištěny a posléze nalepeny na podlahu. QR kódy budou snímány kamerou a získaná data o poloze budou následně odesílána nadřazenému řídícímu softwaru. Lze tedy říci, že jediný okamžik, kdy je známa poloha vozítka je tehdy, když se nachází přímo na určitém QR kódu. Obdobný princip by byl založen na použití RFID (Radio Frequency Identification) kódu. Tento kód však není použit z důvodu nemožnosti rozpoznat natočení vozítka vůči kódu.

Z důvodu zvýšení bezpečnosti při provozu bude zapotřebí vozítko osadit senzory pro detekci překážek. V reálném provozu se může nacházet například i jiná vozítka či pohybující se chodci, tudíž je nutné minimalizovat poškození a zvýšit bezpečnost při provozu. Nadřazený řídící systém bude zajišťovat pohyb po šachovnici a kolaboraci s jinými vozítky.

1.2 Princip řízení

Řízení bude založeno na povelech, které bude vozítko dostávat od nadřazeného řídícího softwaru. Tento řídící software není zahrnut v zadání práce, ale je nezbytný pro celkový pohyb vozítka. Mezi povely bude patřit jakým směr se má vozítko pohybovat. Dle tohoto konkrétního povelu bude následně vozítko schopné pohybu a přesunu. Při povelu bude nutné uzpůsobit ovládání jednotlivých motorů s cílem přímého pohybu všemi směry. Vozítko bude schopné pohybovat se bez natočení konstrukce pomocí omni kol. Z důvodu zvýšení přesnosti přímého pohybu bude vozítko vybaveno MEMS gyroskopem, který bude vracet odchylku od přímého směru a řídící mikrokontroler bude následně ovlivňovat motory vozítka. Pokud se ale vozítko, již nachází na QR kódu, lze pomocí odchylky

načteného kódu srovnat vozítko tak, aby se pro další pohyb vozítko nacházelo v poloze s nulovým natočením vůči QR kódu.



Obrázek 1.1 Snímání QR kódu

Při pohybu vozítka může dojít ke ztrátě lokalizace, kdy vozítko ve stanovený čas nenalezlo QR kód. Důvodem ztráty lokalizace může být například vybitá baterie nebo náhlá překážka vynucující zastavení. Ztráta lokalizace neboli nenalezení QR kódu je hlášeno nadřazenému řídícímu softwaru, který vyšle požadavek na sebelokalizaci. Poté bude vozítko nuceno k sebelokalizaci, tudíž bude nutné projet minimální vzdálenost k nalezení dalšího QR kódu. Při nalezení jakéhokoliv QR kódu bude nahlášena nová poloha a při nenalezení bude vozítko automaticky zastaveno a bude čekat na obsluhu.

1.3 Typické aplikace

Robotická vozítka budou moci být využita v mnoha aplikacích. Díky své konstrukci menších rozměrů jsou velice obratné a vhodné do stísněných prostorů. Mohou být vybaveny koly o velkém průměru, tudíž jsou schopné zdolat i náročnější terén. Jejich konstrukce může být vyrobena z materiálů odolávajících různým typům prostředí.

Mezi typické aplikace použití patří například přepravní mechanismus materiálu po halách. Do dalších aplikací robotických vozítek s autonomním pohybem spadá oblast domácností. V domácnostech mohou být využita v podobě pomocníků při úklidu jako jsou robotické vysavače. Další využití můžou nalézt ve výzkumu či vzdělávání, jakožto učební materiál pro studenty automatizačních oborů na středních a vysokých školách. V neposlední řadě mohou být využity v zabezpečovací technice při monitorování a zabezpečení objektů, či v podobě zařízení, které jsou schopné prozkoumávat objekty před vstupem osob, používaných složkami integrovaného záchranného systému (IZS).

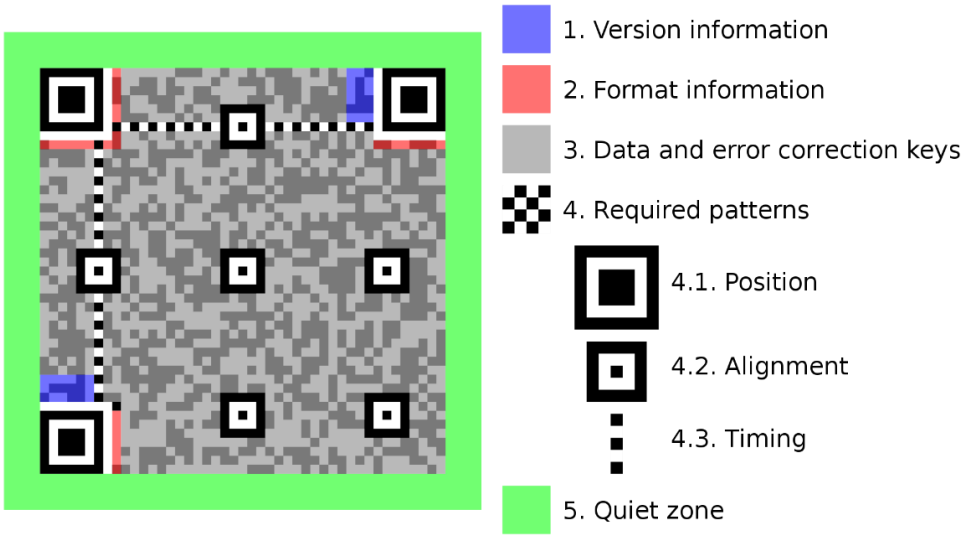
1.4 Snímání QR kódů

1.4.1 Základní informace

QR kód je zkratkou z anglického jazyka „Quick Response“ (neboli „rychlá odezva“). Slouží jako nositel informace a v dnešní době téměř nahrazuje čárové kódy. Jedná se o 2D čtvercový kód, který má několik výhod oproti čárovým kódům. Dokáže uchovat 100x více informací než stejně velký čárový kód a lze ho rychleji načíst. Pro správné načtení není nutné znát jeho úhel natočení, jelikož ho lze načíst z jakékoli strany. [1][2]

První zmínka o QR kódu pochází z roku 1994 z Japonska, kde ho dva japonští vynálezci Takayuki Nagaya a Masahiro Hara použili v automobilovém průmyslu k nahrazení za čárové kódy. Následně ho společnost Denso, vlastnící licenci, dala volně přístupný a může ho používat každý. QR kódy nyní spadají pod normu ISO/IEC 18004:2006. QR kódy jsou velice všestranné a lze je použít téměř všude. Do jednoho QR kódu lze v průměru uchovat až 4296 textových znaků, nebo 7089 číslic závisející však na jeho velikosti. Mezi nejpoužívanější odvětví patří obchodní a gastronomický průmysl. [1][2]

1.4.2 Struktura



Obrázek 1.2 Struktura QR kódu [3]

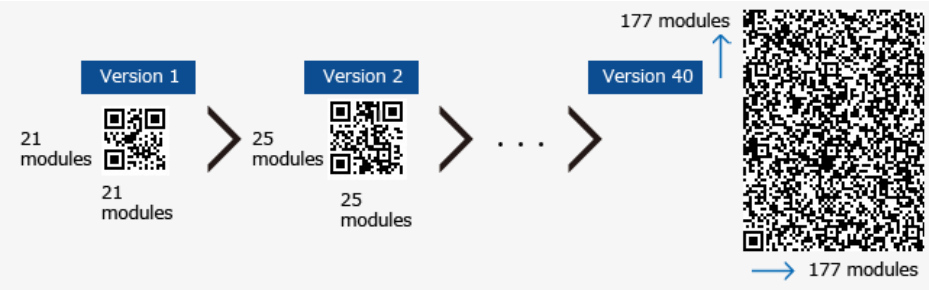
Na obrázku 1.2 je znázorněna struktura QR kódu, která je reprezentována v podobě černobílých objektů. Celý QR kód je složen z několika základních částí, kde každá část má svoji určitou funkci. Tyto funkce jsou:

- **1. Informace o verzi** – tato část kódu označuje konkrétní verzi QR kódu. V současnosti existuje až 40 odlišných verzí struktury QR kódu, nejpoužívanější jsou však verze 1–7. Jednotlivé verze jsou popsány v podkapitole 1.4.3. [4]

- **2. Informace o formátu** – nachází se v okolí pozičních čtverců a určují o jaký typ vzoru masky dat se jedná. Maska říká, jak jsou čtverečky dat uspořádány. Dále představují informaci o toleranci chyb, která může v samotných datech nastat, například při poškození nebo při načítání QR kódu. Slouží ke zjednodušení skenování. [4]
- **3. Data a samoopravné kódy** – tato část pokrývá zbytek kódu a představuje samotná data ve kterých je uložena informace. [4]
- **4. Povinné vzory** – první vzor (4.1) je pozice určující polohu natočení QR kódu. Tedy pokud bude tento vzor jinak umístěný při snímání, jedná se o pootočený kód. Druhý vzor (4.2) je zarovnání. Používá se u rozsáhlejších a větších QR kódů a pomáhá s orientací. Poslední vzor (4.3) slouží ke sdělení informací o velikosti dat uložených v QR kódu. [4]
- **5. Tichá zóna** – jedná se o prostor obklopující samostatný čtvercový kód. Tato zóna neobsahuje žádná data, ale je zde z důvodu správného dekódování kódu a ohraničení oblasti uložených dat. [4]

1.4.3 Verze QR kódů

QR kódy jsou složeny ze stejného počtu řádků a sloupců, ale jejich počet není náhodný. Kolik bude daný kód obsahovat řádků a sloupců je dáno verzí QR kódu. Čím bude verze QR kódu vyšší, tím je samotný QR kód větší a obsahuje více informací. Tedy množství uložených dat v QR kódu je závislé na jeho velikosti. První verze obsahuje 21 x 21 modulů a poslední verze 40 má 177 x 177 modulů. Příkladem je například Mikro QR kód, který umožňuje uložit pouze malá množství informací, maximálně 35 číslic. [5]



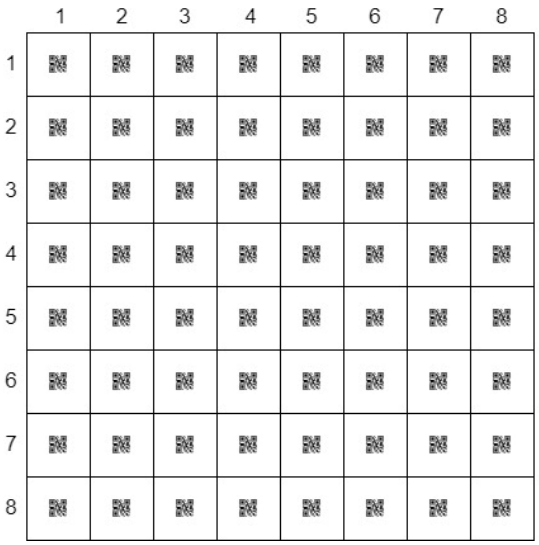
Obrázek 1.3 Verze QR kódů [5]

Další vlastností QR kódů je schopnost samoopravy. Pokud je QR kód nějakým způsobem poškozen či znečištěn, data se nemusí nenávratně ztratit. Každý QR kód má schopnost data do jisté míry opravit pro následující čtení. Množství schopnosti opravit data je dáno 4 úrovněmi: L, M, Q a H. Nejvyšší úroveň je H, která dokáže opravit až 30 % poškozených dat. Následuje úroveň Q s 25 % a M s 15 %. Nejnižší je úroveň L, která je schopna samoopravit jen 7 % ze všech načtených dat. [6]

1.4.4 Uspořádání QR kódů v aplikaci

QR kódy v aplikaci slouží pro detekci aktuální polohy vozítka. Dále umožňují plánování tras a pohyb vozítka po hale (objektu). QR kódy budou v aplikaci rozmístěny v šachovnicovém rozložení o velikosti 8 x 8 QR kódů. Dohromady bude vytištěno a posléze přilepeno páskou na podlahu 64 rozdílných QR kódů s unikátním číslem. Číslo zakódované v QR kódů bude určovat maticový systém. Tento systém bude tvořit řádky a sloupce, kde informace odesílána řídícímu systému bude v podobě řádku a sloupce daného QR kódu. Aby nedocházelo k nechtěným odleskům a tím pádem špatné detekci QR kódů bude muset být páska z matného a nelesklého materiálu či samotný QR bude muset být vytištěn na samolepící štítek.

QR kódy z důvodu ohniskové vzdálenosti kamery od objektu budou mít tu nejmenší možnou velikost, tj. Verze1 (21 x 21 modulů) viz výše. Kódy budou rovnoměrně rozmístěny v předem stanovené vzdálenosti od sebe. Tato vzdálenost nesmí být příliš malá z důvodu vysokého počtu kódů. Naopak také nesmí být příliš velká, jelikož při pohybu vozítka se ztrácí informace o jeho aktuální poloze. Pro testovací účely bude použita místnost s podlahou tvořenou čtvercovými dlaždicemi, kde přesně uprostřed dlaždice bude nalepen QR kód.



Obrázek 1.4 Šachovnicové rozmístění QR kódů

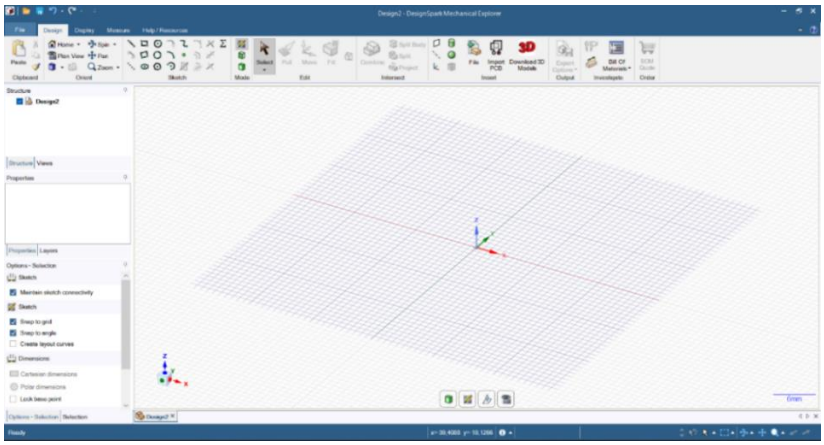
2. NÁVRH ROBOTICKÉHO VOZÍTKA

V následující kapitole je popsán návrh robotického vozítka. Podkapitola 2.1 popisuje vývojové prostředí DesignSpark Mechanical, ve kterém je konstrukce vytvořena a následně připravena pro tisk na 3D tiskárně. Podkapitola 2.2 pojednává o volbě konkrétních komponentů. Podkapitola 2.3 se zabývá konstrukcí vozítka s detailním popisem podvozku, bočních stěn, horního dílu a doplňkových dílů. Dále je v této kapitole popsáno detailní rozmístění vybraných elektrických komponentů.

2.1 DesignSpark Mechanical 6.0.3

Pro návrh a tvorbu konstrukce robotického vozítka bylo použito vývojové prostředí DesignSpark Mechanical. Tento software byl vyvinut společností RS Components výrobcem a dodavatelem elektronických součástek na dnešním trhu. Jedná se ve standartní verzi o bezplatný nástroj pro tvorbu 2D, ale především 3D modelů. Jelikož je výrobcem společnost RS Components software nabízí online knihovnu s více než 38 tisíci 3D modelů součástek, které lze do projektu naimportovat. DesignSpark Mechanical bezprostředně navazuje na vývojové prostředí DesignSpark PCB (Printed Circuit Board), který umožňuje vytvářet desky plošných spojů. Pokud je vytvořena deska plošného spoje v tomto softwaru je zcela kompatibilní s DesignSpark Mechanical a lze ji do projektu naimportovat pro další práci. DesignSpark Mechanical umožňuje import i dalších 3D modelů z jiných vývojových prostředí, pro návrh modelů, s podmínkou toho, že dané soubory musí být typu IDF nebo STEP. [7]

Uživatelské rozhraní (GUI – Graphical User Interface) je navrženo a vytvořeno s odpovídajícími standarty společnosti Microsoft. Má mnohé funkce převzaté z operačního systému Windows, proto může být pro mnohé vývojáře velice intuitivní a jednoduché. [8] Velkou výhodou tohoto vývojového prostředí je nízká náročnost na výpočetní výkon procesoru. [9]



Obrázek 2.1 DesignSpark Mechanical

2.2 Výběr komponentů

Již při tvorbě návrhu konstrukce bylo nutné zohlednit velikost a tvar vozítka v závislosti na elektrických komponentech, kterými bude vozítko osazeno. Pro návrh elektrické části robotického vozítka byly vybrány následující komponenty. Výběr komponentů probíhal po diskusi se zadavatelem v závislosti na náročnosti a požadavcích aplikace. Všechny použité komponenty se základní funkcí jsou vypsány níže a jejich konkrétní popis je v kapitole 3.

- **Motor JGB37-550 12 V s převodovkou** – k zajištění pohybu bude vozítko osazeno motory ovládanými pomocí řídící jednotky Raspberry Pi Pico. Přívodní vodiče motorů budou připojeny k řadiči pro DC motory.
- **Řadič pro DC motory 7 A 160 W** – slouží pro převod PWM (Pulse Width Modulation) signálu z Raspberry Pi Pico na signál vhodný pro ovládání motorů.
- **Raspberry Pi NoIR kamera V2** – tato komponenta bude v aplikaci použita pro snímání QR kódů a tím i k detekci aktuální polohy. Kamera bude doplněna o přisvětlovací LED a bude připojena k Raspberry Pi 4 přes CSI (Pulse Width Modulation) port.
- **MEMS Gyroskop** – bude sloužit pro detekci odklonu od přímé trajektorie pohybu při přesouvání se mezi QR kódy. Data z tohoto senzoru budou zpracovávána a budou použita pro ovládání motorů. Tato komponenta bude připojena k Raspberry Pi Pico a jeho výstup bude připojen na port I2C.
- **Laserový senzor vzdálenosti GY-VL53L0X I2C** – vozítko bude z pohledu zajištění bezpečnosti vybaveno senzory pro detekci překážek. Senzory budou připojeny k Raspberry Pi Pico. V aplikaci budou použity 4 takovéto senzory na každé straně vozítka. Senzory budou připojeny k I2C portu.
- **I2C INA 3221 Senzor napětí a proudu** – tato komponenta bude měřit napětí a proud procházející drivery z důvodu přetížení. Dále zde bude pro měření napětí baterie z důvodu možného odeslání zprávy o nízkém napětí a následné nutnosti nabíjení. Každý driver a baterie má svůj senzor, tudíž aplikace bude obsahovat pouze jeden takovýto senzor, jelikož jeden senzor umožňuje sledovat až tři kanály. Senzor bude připojen k Raspberry Pi Pico přes port I2C.
- **TCA9548 1xI2C master – 8xI2C slave expander** – jedná se o přepínač mezi I2C signály, jelikož Raspbery Pi Pico disponuje pouze dvěma I2C porty a je nezbytné připojit 6 komponentů s I2C porty. K této komponentě budou připojeny senzory pro detekci překážek. Přepínač bude připojen k Raspberry Pi Pico.
- **Raspberry Pi Pico** – je řídící mikrokontroler, který bude dle příkazů od nadřazeného řídícího softwaru vhodně ovládat motory. Dále bude snímat data

z výše uvedených senzorů a tyto data odesílat Raspberry Pi 4, se kterým bude propojen pomocí UART (Universal asynchronous receiver/transmitter) portu.

- **Raspberry Pi 4** – jedná se o mikrokontroler, který bude propojen s Raspberry Pi Pico a bude přijímat od něho data ze senzorů. Dále bude sloužit pro sběr a zpracování dat z kamery, jelikož k němu bude přes CSI port připojena kamera. Všechna tato data následně budou odesílána přes Wi-Fi (Wireless Fidility) modul nadřazenému řídícímu softwaru. Od řídícího softwaru bude dostávat příkazy nutné pro zpracování a následné ovládání vozítka.
- **Baterie 3S –** bude v aplikaci použita pro napájení. Jedná se o vytvořenou baterii, která je složena ze 3S článků a napájí nezbytné komponenty. Tato baterie byla přidělena jakožto hotový výrobek zadavatelem.
- **Doplňkové komponenty –** aplikace bude vybavena pro převod napětí DC/DC převodníkem z 12 V na 5 V, z důvodu napájení komponentů. Dále bude v aplikaci použito antivandal tlačítko s LED indikátorem pro spínání napětí.

Konkrétní schéma propojení výše uvedených komponentů je v kapitole 5.

2.3 Konstrukce

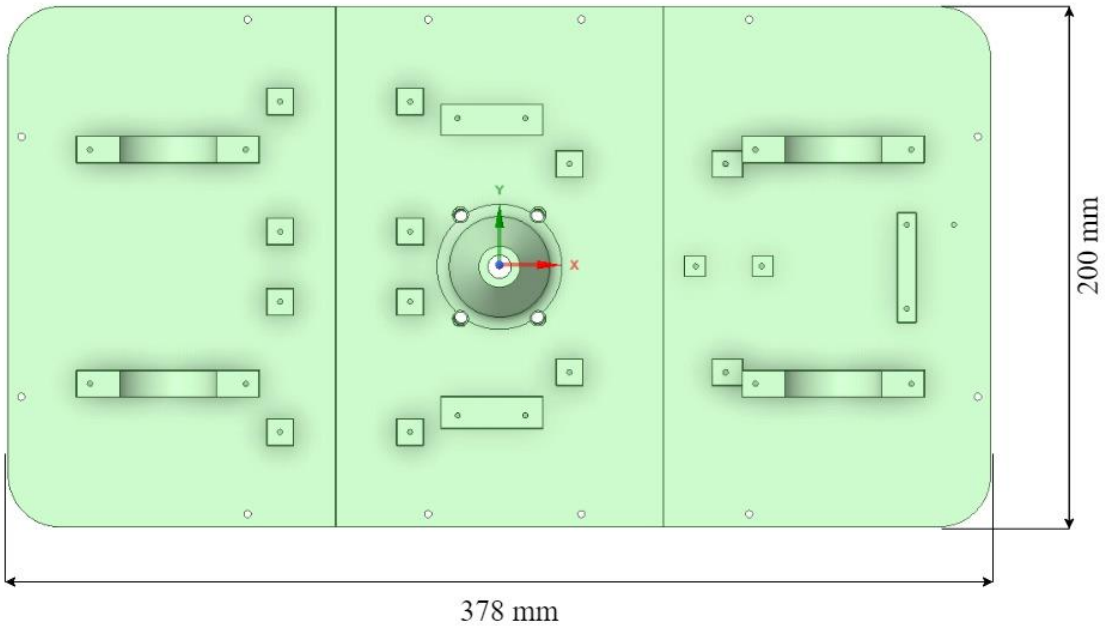
Tělo vozítka bude navrženo tak, aby bylo schopné převážet předměty nižších hmotností a také s ohledem na velikost komponent, kterými bude vozítko osazeno. Z důvodu vyšší stability při pohybu bude nutné vozítko osadit 4 koly, které umožňují všesměrový pohyb. Celé vozítko je namodelováno v prostředí DesignSpark Mechanical, které je popsáno výše v podkapitole 2.1.

2.3.1 Podvozek

Model podvozku vozítka bude následně vytištěn na 3D tiskárně, kde tloušťka podvozku je 5 mm z důvodu namáhání vozítka při přesunu materiálu a hmotnosti komponentů jimiž je vozítko osazeno.

Základem vozítka je podstava ve tvaru obdélníku o rozměru 378 x 200 mm. Podvozek má zaoblené rohy s poloměrem 20 mm. Omni kola jsou umístěna tak, aby jejich středy tvořily čtverec, jelikož při tomto rozmístění kol je vozítko nejstabilnější. Středy kol jsou od sebe vzdáleny 256 mm, aby byl zajištěn minimální prostor mezi dvěma protějšími motory pro jejich přívodní vodiče. Podvozek se nachází ve výšce 19,18 mm od zemského povrchu z důvodu velikosti kol a motorů.

Jelikož je vozítko rozsáhlé velikosti bylo zapotřebí rozdělit podvozek na tři části, aby byla možná realizace tisku. Postranní díly jsou široké 125,9 mm a prostřední díl má šířku 125,8 mm. Mezi jednotlivými díly podvozku se nachází dilatační mezery o velikosti 0,2 mm z důvodu nepřesnosti tisku a pro následné bezproblémové smontování.



Obrázek 2.2 Půdorys podvozku robotického vozítka

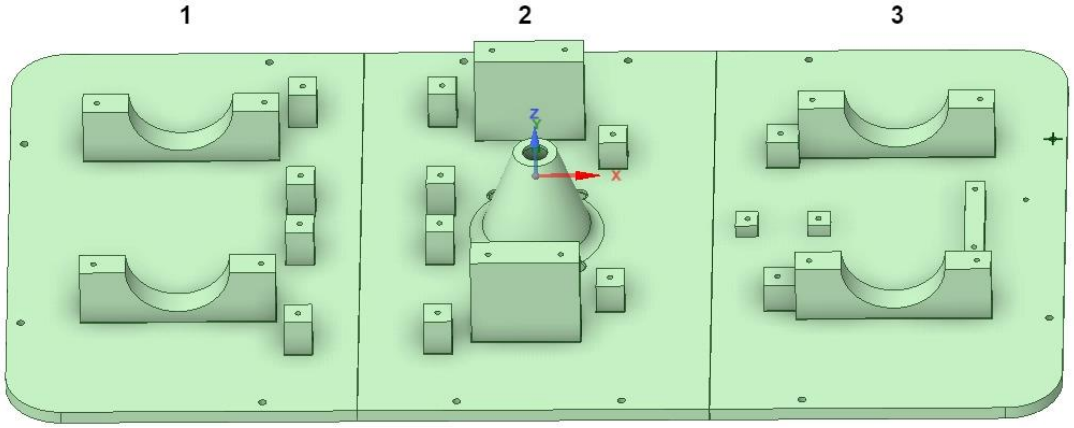
Na středním dílu podvozku č.2 dle obrázku 2.3, přímo ve středu celého vozítka, se nachází otvor pro kameru. Otvor je kuželovitého tvaru se zaoblenými hranami s podstavou o velikosti 40,05 mm a sklonem 32,78°. Tento otvor je nezbytný pro správné snímání QR kódů kamerou. Jelikož je otvor přímo ve středu podvozku, je možné pak lépe detekovat, že vozítko se nachází přesně ve středu čtverce, kde je nalepený QR kód. Pokud by kamera snímala QR kódy jinde než ve středu vozítka, mohlo by dojít k tomu, že část konstrukce vozítka by mohla zasahovat i mimo vyznačený čtverec kde je umístěn snímaný QR kód. Znamenalo by to, že vozítko obsazuje více čtverců najednou a tyto čtverce by nemohly využívat i jiná vozítka. Dále by nebylo jednoduché zjistit do jakého čtverce vozítko zasahuje, a který další čtverec je obsazen, jelikož vozítko hlásí svou polohu z nasnímaného QR kódu. Pokud by nešlo umístit kameru do středu vozítka, musely by se čtverce, kde se nachází QR kód zvětšit. Nevýhodou tohoto řešení by bylo, že se zvýší časový úsek ztráty informace o poloze vozítka.

Na spodní straně prostředního dílu se v okolí podstavy kužele nachází otvory pro přisvětlovací LED, které umožnují lepší detekci QR kódu. LED jsou zde z důvodu zastíněného prostoru pod vozítkem a tím toto okolí přisvětlují. LED jsou umístěny tak, že jejich středy tvoří obdélník o velikosti 38,02 x 28,97 mm. Dále se na středním dílu podvozku nacházejí podpěrné nohy sloužící jako držák pro modul Raspberry Pi 4 s kamerou. Na obou stranách od kamery se nachází další podpěrné nohy pro drivery a pro desku s plošným spojem. Všechny tyto nohy budou tisknuty jako součást středního dílu.

Postranní díly obsahují vždy dva půlkruhové držáky pro uchycení motorů. Držáky bylo nutné konstruovat s ohledem na pozici motoru. Jelikož mají motory na svém těle otvory pro chlazení bylo nutné držáky umístit před tyto chlazení a tím zajistit, aby

nedocházelo k přehřívání motorů. Dále je možné na dílu č. 1 vidět, že se zde nacházejí další čtyři podpěrné nohy pro uchycení driverů pro řízení motorů. Na dílu č. 3 je dle obrázku 2.3 patrné, že vlevo vedle držáků pro motory se nachází další dvě podpěrné nohy pro uchycení desky plošného spoje. Uprostřed dílu č. 3 jsou dále podpěrné nohy pro uchycení komponenty TCA9548 1xI2C master – 8xI2C slave expander. V pravé části tohoto dílu jsou také vytvořeny podpěry pro snímač napětí a proudu.

Po obvodu celého podvozku jsou připraveny otvory pro šrouby. Tyto šrouby budou sloužit pro uchycení podvozku k rámu vozítka, který je popsán níže. Celkem se na každém z dílů podvozku nacházejí vždy 4 otvory pro šrouby. Jednotlivé díly podvozku mezi sebou nejsou nikterak spojeny, jelikož jako spojovací element slouží rám vozítka společně s komponenty, kde se jejich tělo nachází současně na dvou dílech podvozku.



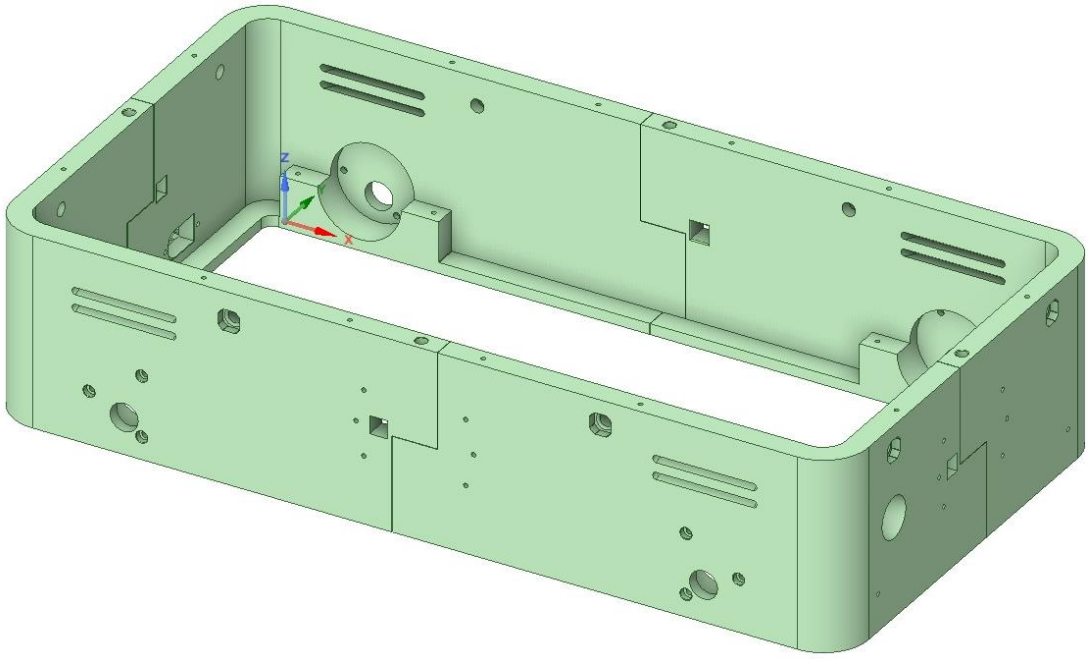
Obrázek 2.3 Podvozek robotického vozítka

2.3.2 Boční stěny

Tloušťka bočních stěn je z důvodu hmotnosti možných převážených předmětů zvolena na 10 mm. Výška bočních stěn je 84,8 mm a je zvolena s ohledem na množství komponentů umístěných uvnitř vozítka. Stejně jako podvozek jsou boční stěny rozděleny do čtyř shodně velkých dílů. Tvar těchto bočních stěn je shodný s tvarem celé podstavy, a tak podvozek přesně zapadá pod boční stěny. Vozítko je navrženo tak, že není určena přední a zadní část vozítka, tudíž vzhled vozítka je z obou stran shodný. Orientace vozítka bude ovlivněna ovládáním motorů. Díly jsou navrženy tak, aby každý bok měl funkci zámku (výřez či výstupek), a tudíž do sebe mohli zapadnout. Tímto mechanismem je zajištěna vyšší pevnost celého rámu. V zámku jsou vytvořeny otvory pro následné sešroubování. Spodní hrany bočních stěn jsou širší, ve tvaru L, což zajišťuje větší pevnost vozítka s ohledem na větší dosedací plochu boční stěny. Tyto vzniklé doplňkové plochy slouží také jako držáky pro přední části všech motorů. Pod motory se nachází opět půlkruhy odpovídající průměru motorů, které jsou součástí bočních stěn. Motory jsou poté usazeny až do poloviny tloušťky stěny, což snižuje pravděpodobnost nechtěnému protočení motoru v držácích. Z vnější strany bočních stěn jsou v oblasti motorů vytvořeny

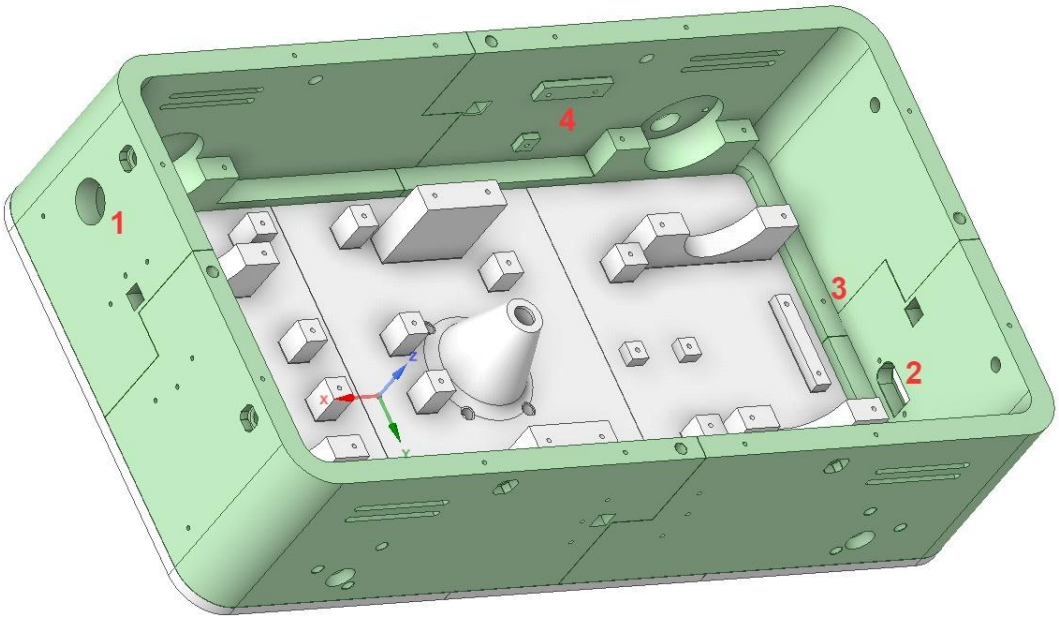
otvory pro uchycení motorů i v příčném směru, jelikož motory použité v aplikaci mají ze přední strany montážní otvory pro uchycení.

Po všech bočních stěnách jsou také otvory na spodní hraně a horní hraně, které přesně odpovídají otvorům na podvozku a víku. Tyto otvory slouží pro spojení bočních stěn společně s podvozkem a k připevnění víka. V oblasti, kde se nachází motory jsou v každém z dílu vytvořeny dva otvory pro odvětrávání. Odvětrávání na bočních stěnách je nezbytné pro správný chod elektrických komponentů umístěných uvnitř vozítka s ohledem na jejich vyzařované teplo. Dále jsou na každém z dílu umístěny vždy 2 otvory pro LED, které indikují aktuální směr pohybu či stav vozítka. Celkem vozítko obsahuje 8 takovýchto LED. Posledním společným prvkem všech bočních stěn jsou montážní otvory pro uchycení krytu senzorů detekující překážky společně s otvorem pro vývod kabeláže od senzoru.



Obrázek 2.4 Boční stěny robotického vozítka

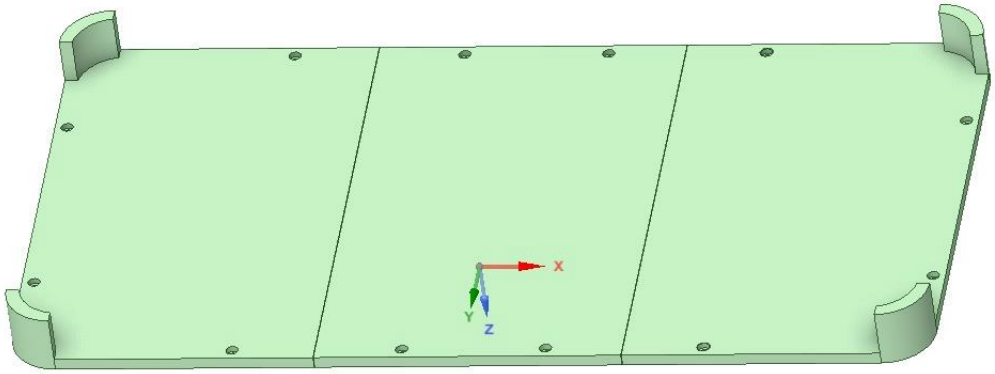
Na bočních stěnách se mimo jiné nachází i další úpravy, které lze vidět na obrázku 2.5. Na tomto obrázku je patrné že na jedné z bočních stěn, konkrétně na kratší straně vozítka se nachází otvor pro tlačítko (1) s průměrem 19 mm, které slouží pro odpojení napájení a uvedení vozítka do stavu nečinnosti. Naproti tomuto tlačítku se nachází další otvor, který může sloužit pro umístění zapuštěného konektoru pro nabíjení baterie (2), aby nebyla nutná demontáž horního dílu při každém nabíjení. V této aplikaci bude tento otvor zaslepen záslepkou. Na jedné z kratších stěn je také vytvořen otvor, sloužící pro uchycení třetího šroubu senzoru napětí a proudu (3). Posledním prvkem jsou montážní otvory s podpěrami (4) na delší stěně, které slouží pro upevnění DC/DC měniče napětí.



Obrázek 2.5 Boční stěny robotického vozítka

2.3.3 Horní díl

K zamezení styku obsluhy s elektrickými komponentami uvnitř vozítka je vozítko vybaveno ochranným horním dílem. Tento díl má tvar shodný s podvozkem s výškou 5 mm. Je stejně jako podvozek rozdělen na 3 díly o shodných rozměrech jako u podvozku. Na každé z rohů celé desky jsou půlkruhové zábrany o poloměru 20 mm a výšky 20 mm. Tyto zábrany se nachází pouze na rohách, aby nemohlo dojít k nechtěnému pohybu materiálu, ale také aby bylo možné tento materiál bezproblémově na vozítko naložit či z vozítka složit. Dále jsou na horním dílu rozmístěny montážní otvory pro přichycení horního dílu k bočním stěnám. Tyto otvory jsou do poloviny výšky víka širší, aby mohly mít montážní šrouby zapuštěnou hlavu a nepřekáželi tak převáženým materiálům.



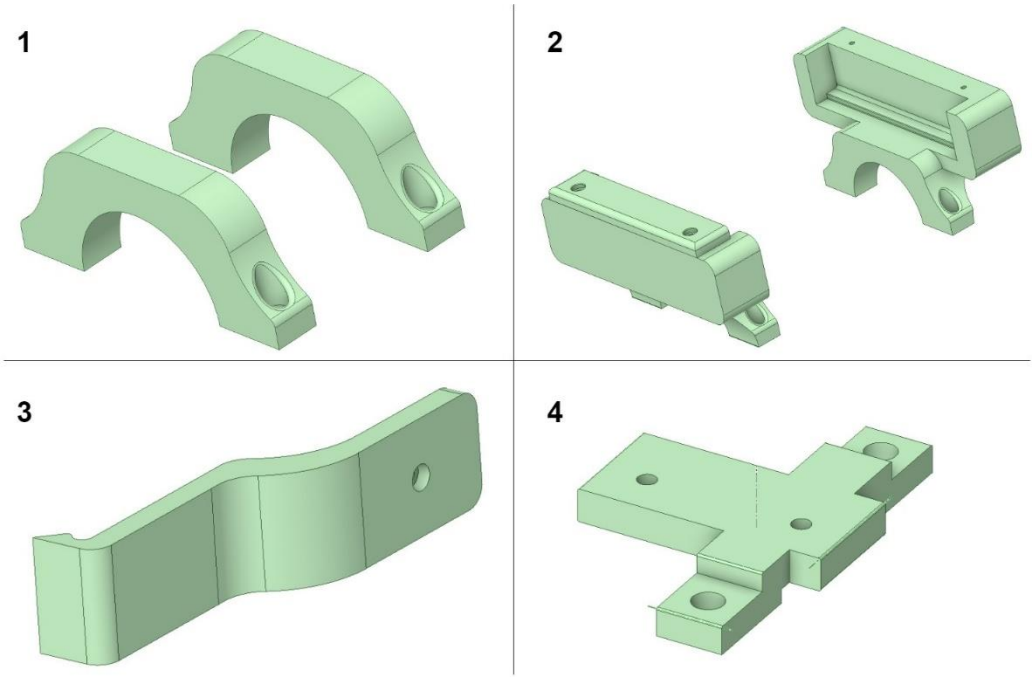
Obrázek 2.6 Horní díl robotického vozítka

2.3.4 Doplňkové díly

Jelikož bylo nutné při návrhu vozítka zohlednit i uchycení některých komponentů či doplnit některé hlavní komponenty o rozšiřující části byly realizovaná i doplňkové komponenty. Mezi tyto doplňkové komponenty patří:

- **Kryt motoru** – jak je již zmíněno výše, motory jsou uloženy a posazeny na předpřipravený úchyt nacházející se na desce podvozku. Dále bylo nutné realizovat a vytvořit také horní dílu tohoto úchytu, aby se co nejvíce minimalizovala pravděpodobnost nechtěného pohybu při provozu vozítka. Z tohoto důvodu byly vytvořeny obdobné díly ve tvaru půlkruhu, které přesně kopírují velikost poloměru motoru. Jelikož má motor v přední části (blíže k hřídeli) užší poloměr než v zadní části (blíže ke kontaktům), bylo nutné vytvořit dva odlišné úchyty vždy pro jeden motor. Z důvodu nepřesnosti tisku bylo nutné po celém vnitřním obvodu úchytu vytvořit opět dilatační mezeru 0,2 mm, aby se dal do těchto úchytů motor upevnit. Výška úchytu byla v oblasti otvorů pro šroub snížena o 1,5 mm aby při použití montážních šroubů úchyty pevně sevřely motory. Úchyty jsou vyobrazeny na obrázku č. 2.7 a konkrétně se jedná o díly č.1.
- **Držáky pro baterii** – jelikož je baterie umístěna v oblasti nad motory, bylo nutné vytvořit vhodné držáky, které baterii umístí do této polohy a také zamezí nechtěnému pohybu baterie při provozu vozítka. Jedná se o díl č.2 na obrázku 2.7. Držák baterie je vytvořen jako součást úchytů pro motory s větším poloměrem. Tedy spodní část držáku tvoří úchyty pro motor a horní část držáku tvoří úchyt pro baterii. Držák baterie přesně kopíruje tvar použitého pouzdra baterie a je opět vytvořena 0,2 mm rezerva na každé straně, aby bylo vložení baterie do držáku snadnější. Baterie je tedy položena na úchytu motoru a ze všech bočních stran je okolo ní vytvořen rámeček. Baterii lze tedy vyjmout směrem svisle vzhůru. Tento způsob uchycení baterie šetří místo ve vozítku oproti tomu, kdyby se baterie vyndávala ze strany. Aby nedošlo k nechtěnému vypadnutí baterie z držáku, jsou po stranách přes baterii namontovány plíšky pro její uchycení.
- **Nárazník** – na každé straně vozítka je také připevněn ochranný nárazník, který eliminuje následky případného nárazu vozítka do překážky. Tyto nárazníky chrání omni kola a motory v podélném směru pohybu. Jsou zde proto, kdyby z jakéhokoliv důvodu, došlo ke kontaktu s cizím předmětem, aby minimalizovali poškození hřídele motoru a kol. Nárazníky jsou připevněny v oblasti každého kola na kratší stěně rámu. Jsou navrženy tak, že jejich tvar obklopuje celé dané omni kolo i s motorem. Jeden tento nárazník lze vidět na obrázku 2.7 č. 3.
- **Držák pro gyroskop** – aby bylo možné umístit MEMS gyroskop přesně do středu vozítka bylo zapotřebí vytvořit pro tento senzor držák, jelikož se střed

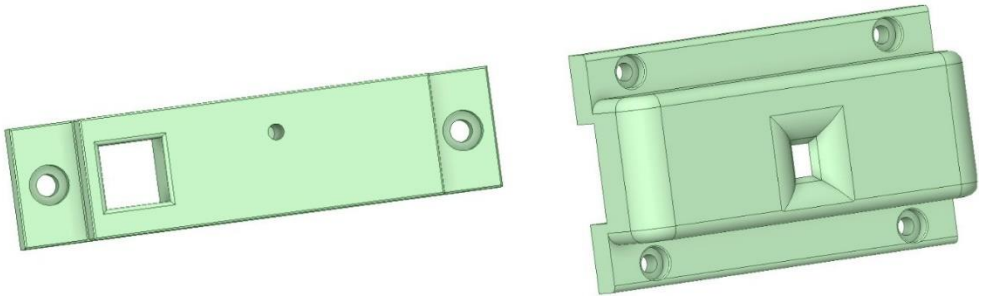
vozítka nachází v oblasti modulu s kamerou. Držák je ve tvaru obráceného písmena U s podpěrnými nohami a je pomocí šroubů připevněn k modulu s kamerou. Senzor je k tomuto držáku připevněn taktéž pomocí šroubů, které však mají ze spodní strany držáku zapuštěné matice. Tyto matice jsou zde z důvodu nízké výšky držáku, a tudíž by šrouby nebyly pevně spojeny s držákem. Držáku je ve výšce 7 mm nad modulem s kamerou, jelikož bylo zapotřebí dbát na odvětrávání modulu a celkovou výšku vozítka. Na obrázku 2.7 se jedná o díl č.4.



Obrázek 2.7 Doplňkové díly pro realizaci vozítka

- **Kryt pro uchycení senzoru** – tento kryt slouží pro připevnění senzoru přesně ve středu z každé strany vozítka. Bylo nutné vytvořit kryt, jelikož přímo uprostřed každé stěny se nachází zámek dvou dílů s otvorem pro montážní šroub. Kryt je složen ze dvou částí, a to spodního dílu (levý díl na obrázku 2.8) a horního dílu v podobě víka (pravý díl na obrázku 2.8). Spodní díl má na svém těle otvory, kterými je připevněn k bočním stěnám vozítka a otvor pro připevnění senzoru. Otvory pro připevnění k bočním stěnám jsou do poloviny výšky dílu širšího průměru, aby bylo možné montážní šroub zapustit a aby tento šroub následně nebránil při montáži horního dílu. Dále tento díl obsahuje díru ve tvaru čtverce, pomocí níž je možné do vnitřní části vozítka přivést přívodní vodiče senzoru. Horní díl je vytvořen tak, aby po připevnění k bočním stěnám a spodnímu dílu, uvnitř vytvořil prostor pro senzor a přívodní vodiče. Lze tedy říci, že mezi spodním a horním dílem vznikne tzv. kapsa. Horní díl obsahuje, v oblasti čipu pro snímání, hranol, kterým bude

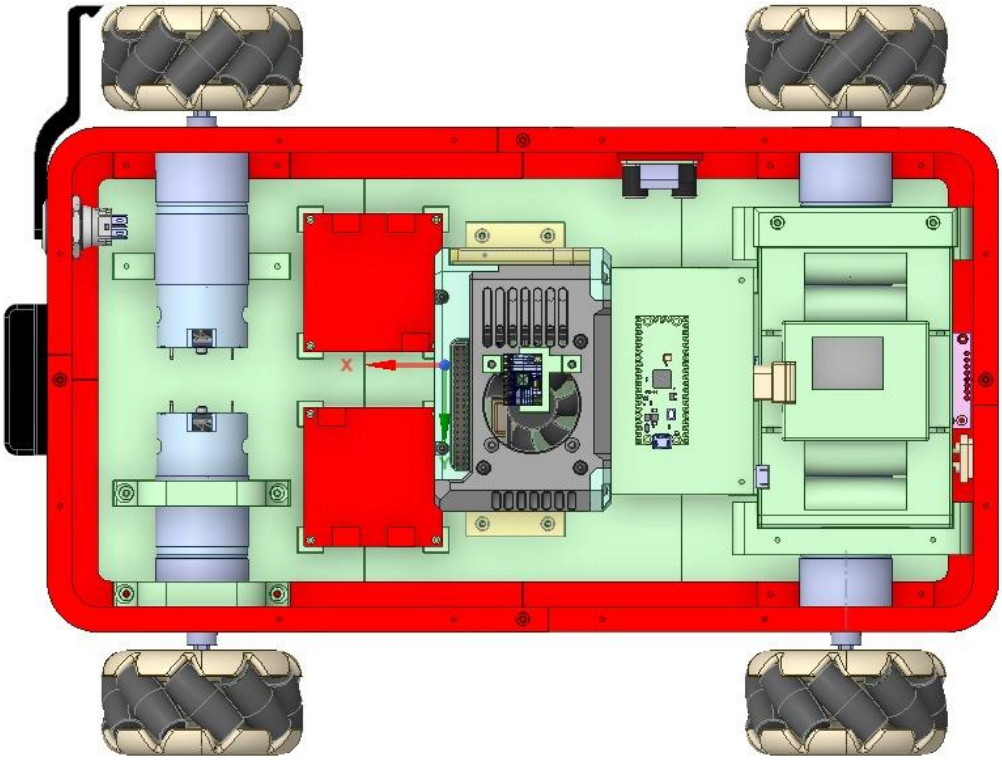
senzor moci snímat okolní předměty. Pro montáž horního dílu jsou využity boční stěny, kam je obdobně jako spodní díl připevněn. Celý tento kryt je z estetického hlediska a z důvodu nedokonalosti tisku ostrých hran, zaoblen na hranách.



Obrázek 2.8 Kryt pro uchycení senzoru

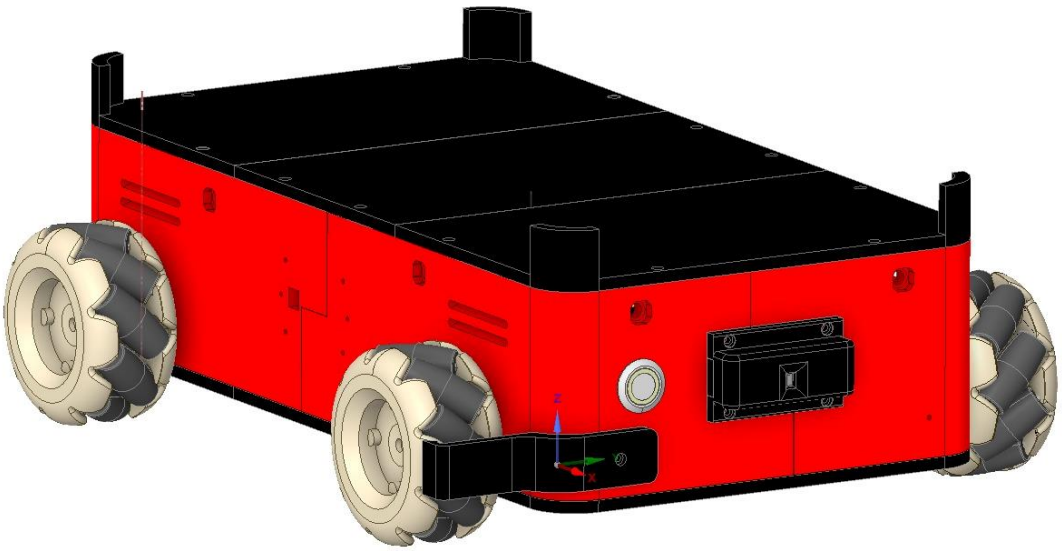
2.3.5 Rozmístění komponentů v konstrukci

Elektrické komponenty jsou v rámu vozítka rozmístěny v závislosti na propojení s jinými prvky a také dle jejich velikostí. Uprostřed vozítka se nachází ochranné pouzdro obsahující řídící mikrokontroler Raspberry Pi 4 společně s připojenou kamerou. Tyto komponenty musí být umístěny na vyvýšeném kuželu, jelikož má kamera minimální snímatelnou vzdálenost, ze které je schopna číst QR kódy. Dle obrázku č. 2.9 se v pravé části vozítka, poblíž kamery, nachází druhý řídící mikrokontroler Raspberry Pi Pico umístěný na plošném spoji. Tyto dva prvky bylo nutné umístit do středu vozítka s ohledem na to, že všechny ostatní komponenty k nim musí být připojeny. Na boční stěně poblíž Raspberry Pi Pico se nachází DC/DC měnič pro převod napětí. Pod deskou s plošným spojem se nachází I2C modul rozšiřující porty I2C. V pravé části vozítka se nachází baterie složená ze 3S článků. Tato baterie je umístěna na držácích motorů a nachází se tedy nad motory. Motory jsou osazeny do konstrukce tak, aby středy kol nasazených na motoru tvořily výše zmiňovaný čtverec pro lepší ovladatelnost. Mezera mezi motory je 28 mm z důvodu manipulačního prostoru pro přívodní vodiče. Za motory v pravé části, blíže k hraně rámu je umístěn senzor pro měření napětí či proudu (růžová destička), který měří napětí baterie a proud řadičů motorů. Řadiče pro motory (červené destičky) jsou umístěny na pozici blíže k levé straně, jelikož je to s ohledem na jejich velikost jediné volné místo na vozítku. Na středu z každé strany vozítka jsou umístěny senzory pro detekci překážek, které jsou na obrázku znázorněny pomocí černého obdélníku. Poslední komponenta, MEMS gyroskop, je umístěn na držáku na modulu s kamerou, jelikož z důvodu přesné detekce úhlu natočení je nutné, aby se tento senzor nacházel na středu celého vozítka.



Obrázek 2.9 Rozmístění komponentů

Na obrázku 2.10 je vyobrazen návrh celého vozítka včetně elektrický komponentů, omni kol a doplňkových dílů navržených ve vývojovém prostředí DesignSpark Mechanical.



Obrázek 2.10 Výsledný vzhled vozítka

3. ŘÍDÍCÍ A ELEKTRICKÉ KOMPONENTY

V následující kapitole jsou níže popsány zvolené řídící mikrokontrolery společně s elektrickými komponenty, které jsou osazeny na konstrukci vozítka. Podkapitola 3.1 je zaměřena na jednodušší mikrokontroler Raspberry Pi Pico. Kde v první podkapitole 3.1.1 je blíže specifikováno v jaké souvislosti je tento mikrokontroler použit v dané úloze. Podkapitola 3.1.2 je zaměřena na vlastnosti daného mikrokontroleru. V podkapitole 3.2 je blíže popsán mikrokontroler Raspberry Pi 4. Podkapitola 3.2.1 je zaměřena na použití daného mikrokontroleru a druhá část 3.2.2 pojednává o specifických vlastnostech Raspberry Pi 4 jakožto zařízení. Podkapitola 3.2.3 popisuje programovací jazyk Python, který je v dané aplikaci použit.

Podkapitola 3.3 pojednává o pohonu vozítka, a to konkrétně o motoru JGB37-550 12 V s převodovkou a driverem pro řízení. V podkapitole 3.4 je popsán konkrétní typ kamery, kterými jsou kódy detekovány. Podkapitola 3.5 je zaměřena na MEMS gyroskop pro detekci odklonu od přímého směru. Podkapitola 3.6 pojednává o snímači polohy společně s multiplexorem pro spojení dat ze snímačů. Dále jsou v kapitole 3.7 popsány baterie pro napájení elektrických komponentů na vozítku. Poslední podkapitola 3.8 je zaměřena na doplňující elektrickou komponentu, která měří velikost proudu a napětí na každém z motorů.

3.1 Raspberry Pi Pico

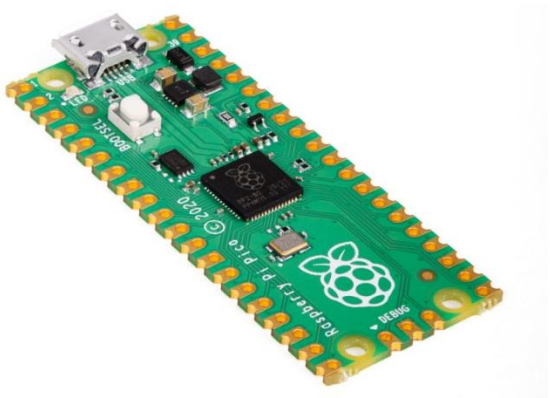
3.1.1 Použití

Raspberry Pi Pico je využit pro řízení motorů a snímaní dat ze snímače napětí a proudu. Pro snímaní dat by nebylo možné použít Raspberry Pi 4, jelikož není vybaveno analogovými vstupy. Dále je k Raspberry Pi Pico připojen snímač uhlového natočení z důvodu udržení přímého směru. Přímý směr je nutné sledovat z důvodu vybavení vozítka všesměrovými koly, které neumožňují přesné pohyby přímým směrem. Také zde jsou připojeny snímače pro detekci překážek.

3.1.2 Klíčové vlastnosti

Jedná se o jednočipový počítač, který nemá vlastní operační systém a lze ho většinou použít pouze na jednu konkrétní úlohu. Je vybaven čipem RP2040, kde „RP“ znamená označení destičky Raspberry Pi a jednotlivé číslovky mají následující význam. „2“ označuje počet jader procesoru, „0“ označuje typ jádra, konkrétně jádro Dual ARM Cortex-MH0 s taktovací frekvencí 133 MHz. Číslo „4“ značí vzorec pro výpočet paměti SRAM (Static Random Access Memory). A poslední číslice „0“, slouží jakožto vzorec pro výpočet energeticky nezávislé kapacity neboli nevolativní paměti uvnitř čipu. [10][11][12][13]

Mikrokontroler obsahuje celkem 26 GPIO (General-purpose input/output) pinů a 8 PIO (Programmable input/output) pinů. Dále na něm lze nalézt 16 pinů umožňují generovat PWM signál. Pro komunikaci jsou zde piny pro UART, I2C a SPI (Serial Peripheral Interface). Lze napájet pomocí USB1.1 (Universal Serial Bus) nebo externího napájecího zdroje o velikosti 3,3 V. Nezbytnou součástí je také FLASH paměť o velikosti 2 MB a SRAM paměť o velikosti 264 kB. Je vybaven přesným hodinovým signálem a časovačem na čipu mikrokontroleru. USB port lze použít nejen pro napájení, ale je primárně určen pro programování tohoto mikrokontroleru. Raspberry Pi Pico je možné programovat v jazycích C a MircoPython. [10][11][12][13]



Obrázek 3.1 Mikrokontroler Raspberry Pi Pico [11]

3.2 Raspberry Pi 4

3.2.1 Použití

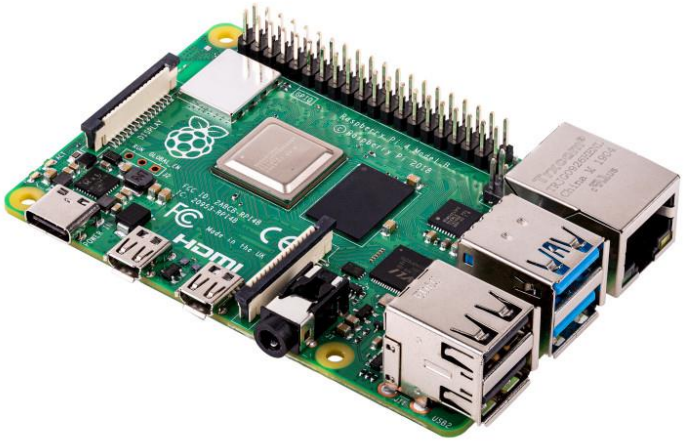
Tento mikrokontroler zajišťuje čtení QR kódů snímaných z kamery. Dále slouží ke komunikaci přes Wi-Fi modul. Data přečtená z kamery jsou odesílána do PC, kde jsou poté zpracována. Raspberry Pi 4 bylo na tuto funkci vybráno z důvodu rychlosti procesoru, což umožňuje rychlé zpracování dat z kamery. Jelikož se jedná jen o koncept a práce není koncipována jako průmyslové zařízení je čtení a přenos dat řešen tímto způsobem. Pokud by se jednalo o finální výrobek určený do průmyslového provozu, čtení a následný přenos dat by zajišťovala například průmyslová kamera.

3.2.2 Klíčové vlastnosti

Raspberry Pi 4 je také jako Raspberry Pi Pico jednodeskový mikropočítač. Tento model se liší tím, že má již svůj vlastní integrovaný operační systém Raspberry Pi OS (Operating System) dříve nazývaný Raspbian. Jedná se o Linuxový systém založený na platformě Debian. Tento operační systém má již mnoho předinstalovaných softwarů, například Python, který umožňuje programování [14]. Pro správný chod operačního systému je nutná pravidelná aktualizace firmwaru. V případě, že operační systém na Raspberry ještě

není nainstalovaný, je na mikrokontroleru slot na microSD (Secure Digital) kartu, která slouží pro načtení operačního systému a pro uložení dat [15].

Mikrokontroler je již vybaven portem USB – C, pomocí něhož je možné mikrokontroler napájet a programovat. Pro napájení je nutné minimálně stejnosměrné napětí o velikosti 5 V s proudem 3 A. Při nižším napájení může dojít ke smazání dat z SD karty či k jejímu poškození. Možné je i napájení mikrokontroleru přes Ethernet, avšak s nutností přídavného hardwarového modulu Raspberry Pi PoE HAT. Mikrokontroler obsahuje dva microHDMI (High – Definition Multi-media Interface) konektory pro připojení monitorů. Dále má 4 jádrový procesor Broadcom BCM2711 ARM Cortex-A72 s taktovací frekvencí 1,5 GHz. [16] Raspberry Pi 4 má několik verzí velikosti operační paměti RAM typu LPDDR4. V daném případě je použito Raspberry s RAM o velikosti 4 GB.



Obrázek 3.2 Mikrokontroler Raspberry Pi 4 [16]

3.2.3 Programovací jazyk Python

Mezi nejpoužívanější programovací jazyky dnešní doby patří Python. Tento programovací jazyk byl sestaven v roce 1991, kde je za jeho tvůrce považován jistý Guido van Rossum.[17] Python je univerzální jazyk a není specializovaný na žádný konkrétní problém. Lze tedy říci, že je vůči ostatním programovacím jazykům velice všestranný. Používá se pro vývoj webových stránek, softwarů a automatizačních úloh. Dále pomocí Pythonu lze analyzovat data a zabývat se strojovým učení. Obsahuje velké množství knihoven, které umožňují rychlejší a efektivnější psaní kódu. Má velmi jednoduchý programovací jazyk podobný angličtině a lze jej používat na několika platformách (Windows, Linux, Mac, Raspberry atd.). [18]

Jelikož je Python určen převážně pro mikrokontrolery s výkonnějšími procesory, je Raspberry Pi Pico programováno v MicroPythonu. MicroPython je jedna z verzí Pythonu, která je navržena pro běh na jednodušších a menších mikrokontrolerech.[18]

3.3 Motor JGB37-550 12 V s převodovkou

Pro pohon robotického vozítka byl vybrán stejnosměrný motor JGB37-550 s převodovkou. Každé kolo pohání jeden motor, tudíž vozítko je osazeno celkem 4 motory. Tento motor je dodáván v několika variantách s různými otáčkami hřídele. Mezi varianty patří 18, 28, 53, 86, 160, 260, 480, 770 rpm. Pro vhodnou rychlost pohybu vozítka je nutné vybrat motor s vhodnými otáčkami hřídele, kde rychlost vozítka závisí na velikosti kol. Vozítko je vybaveno omni koly o poloměru 80 mm.

Skutečná rychlost pohybu vozítka je vypočtena dle vztahu

$$v = \frac{2 \cdot \pi \cdot r \cdot v_{rpm}}{60} \; [m/s] \; , \tag{3.1}$$

kde v je rychlost vozítka, r je poloměr kol v metrech a v_{rpm} představuje maximální jmenovitou rychlost při zatížení.

Pro každou variantu nabízenou výrobcem je níže uvedena tabulka 3.1 se srovnáním. Výběr je však omezen pouze na varianty, které mají maximální proud (Stop Current) 6 A, a nelze ho z důvodu ostatních součástek překročit.

Tabulka 3.1 Porovnání jednotlivých variant motoru JGB37-550

Otáčky hřídele bez zátěže [rpm]	**Maximální otáčky hřídele při zatížení [rpm]**	**Vypočtená rychlost vozítka [m/s]**
160	130	0,545
260	200	0,838
480	380	1,592
770	600	2,514

Z důvodu čtení QR kódu je vhodné zvolit nižší rychlost pohybu, jelikož kdyby bylo vozítko příliš rychlé nebylo by možné správně a včasně detekovat QR kód. Je tedy zvolen motor s otáčkami hřídele 160 rpm (revolutions per minute). Tento motor je napájen stejnosměrným napětím 12 V z baterie, která je popsána níže. Motor je se svojí velikostí a konstrukcí určen do modelářské techniky, což umožňuje použít jej v dané aplikaci k demonstraci úlohy. [19] Průměr hřídele je 6 mm ve tvaru kruhu, s výřezem pro uchycení pomocí šroubů.

Obrázek 3.3 Motor JGB37-550 12 V s převodovkou [19]

3.3.1 Řadič pro DC motory 7 A 160 W

Další nezbytnou elektronickou součástkou je řadič pro řízení motorů. Modul je navržen pro ovládání dvou na sobě nezávislých motorů, kde ovládání umožňuje i změnu směru otáčení. V dané aplikaci je pro řízení 4 motorů nutné použít dva takovéto řadiče, kde každý obstarává 2 motory. Otáčky motoru jsou řízeny pomocí změny PWM signálu.

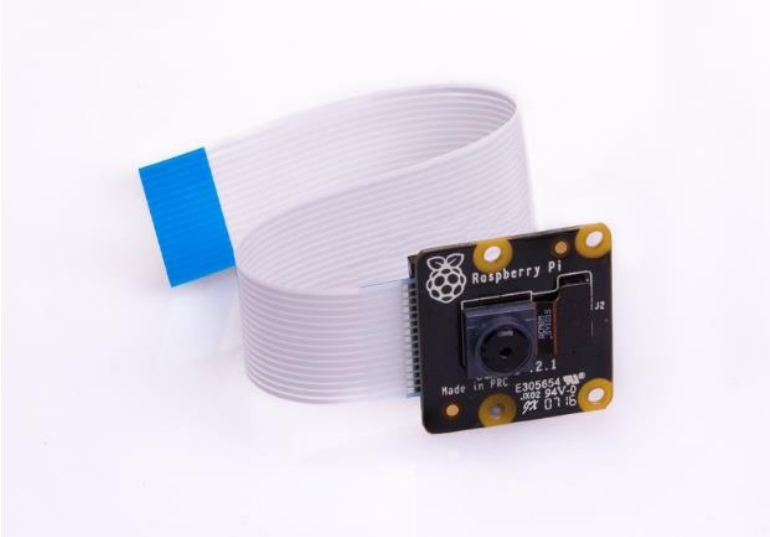
Řadič je napájen stejnosměrným napětím v rozsahu 6,5 – 27 V. Je vybaven třemi výstupy na kanál, kde dva výstupy umožnují napájení motoru a třetí je řídící. Jeho maximální provozní proud je až 7 A na výstup jednoho z kanálů a špičkově je schopný dodat proud až 50 A. Jeho logika je založena na MOSFET tranzistoru s optoelektrickým členem, který zajištuje řízení. Může řídit s logikou 3,3 až 5 V, kde pro úroveň LOW je napětí v rozsahu 0 – 0,8 V a pro úroveň HIGH 3 – 6,5 V. Na těchto řídících signálech je schopen dodat proud od 3–11 mA na každém z kanálu. Na řídícím kanálu umožňuje vytvořit PWM signál v rozsahu 0–10 kHz, kde minimální šířka pulzu je stanovena na 50 µs. Pracovní teplota se pohybuje v rozmezí teplot od -25–80 °C. [20]

3.4 Raspberry Pi NoIR kamera V2

Pro snímání QR kódu, který slouží k orientaci a pohybu vozítka, je použit kamerový modul Raspberry Pi NoIR kamera V2. Tento modul je do zařízení vybrán z důvodu vhodného rozlišení a příznivé ceny dostačující pro tuto aplikaci. Výrobcem kamerového modulu je Raspberry, tudíž je vhodný a kompatibilní s výrobky od téže společnosti. Lze tedy bez nutnosti doplňkového hardwaru připojit kamerový modul k Raspberry, které mají CSI port, přes flex kabel a následně kamerový modul programovat pomocí jazyka Python.

Kamerový modul je osazen kamerou Sony IMX219, která má rozlišení snímače 10 Mpx a ohniskovou vzdálenost 3,04 mm. Touto kamerou je možné pořizovat fotografie a natáčet HD (High-Definition) videa s rozlišením 3280 x 2464 px. Je se svými malými

rozměry 25 x 20 x 9 mm a 4 montážními otvory vhodná pro použití v modelářské technice. Dále je vybavena speciální funkcí nočního vidění, umožňující používat kameru i za snížené světelné viditelnosti. Objektiv je vybaven pouze pevným ohniskem s manuálním ostřením kamery, což ale není problém při použití v dané aplikaci, jelikož se snímané objekty (QR kódy) bodu nacházet vždy ve stejné vzdálenosti od kamery. Jelikož kamera snímá objekty pod vozítkem, ve zastíněném prostoru, bylo nutné dovybavit vozítko přisvětlovacími LED. [21]



Obrázek 3.4 Raspberry Pi NoIR kamera V2 [21]

3.5 MEMS Gyroskop

Jelikož je vozítko vybaveno omni koly zajišťující pohyb, je zapotřebí z důvodu konstrukce kol dovybavit vozítko gyroskopem. Data z gyroskopu slouží jako mechanismus pro detekci odklonu od přímé trajektorie. Jeho výstupní signál má funkci zpětné vazby pro ovládání motorů. Pro danou aplikaci byl vybrán cenově dostupný a výkonově dostačující modul pro detekci pohybu s 3osým gyroskopem a akcelerometrem GY-521.

Destička je vybavena čipy GY-521 a MPU6050, které umožňují měřit úhel natočení a zrychlení. Čipy s řídícím modulem (Raspberry Pi Pico) komunikují přes I2C sběrnici. Destička je napájena stejnosměrným napětím v rozsahu 3–5 V. Dále je vybaven 16bitovým analogově digitálním převodníkem a umožňuje snímat úhel natočení ve všech třech osách (X, Y a Z). Je schopen měřit v rozsahu ± 250, 500, 1000, 2000 °/s. Provozní proud gyroskopu je 3,6 mA. Z důvodu ověření funkce destičky umožňuje uživatelský test. Modul je použitelný při teplotách okolí v rozsahu -40–85°C. [22]

3.6 Laserový senzor vzdálenosti GY-VL53L0X I2C

Pro vyšší bezpečnost provozu vozítka v reálném prostoru je nutné dovybavit vozítko senzory pro detekci překážek nebo osob v blízkosti vozítka. Pro tento případ je vozítko osazeno 4 senzory polohy na každé straně. V dané aplikaci je z důvodu vhodného principu funkce použit laserový senzor vzdálenosti GY-VL53L0X I2C.

Vyhodnocovací jednotka pro výpočet vzdálenosti pracující na principu odraženého a přijatého laserového paprsku, vypočítává vzdálenost z prodlevy mezi vyslaným a přijatým světelným signálem. Je napájen stejnosměrným napětím 3–5 V. Je schopný detekovat předměty až do vzdálenosti 2 m s přesností ± 3 %. Je vhodný do jakéhokoli prostředí, jelikož přesnost signálu není ovlivněna žádnou odrazivostí povrchu. Není rušen žádným okolním zářením, jelikož je paprsek vysílán ve vysokém infračerveném záření, což je pro lidské oko neviditelné. Je vybaven komunikačním rozhraním I2C, pomocí něhož odesílá data řídící jednotce. Svými malými rozměry je hojně využíván v technice a mezi konkrétní aplikace patří například robotika, počítačové systémy či bílá technika (automatické pračky, myčky nádobí, …). [23][24]

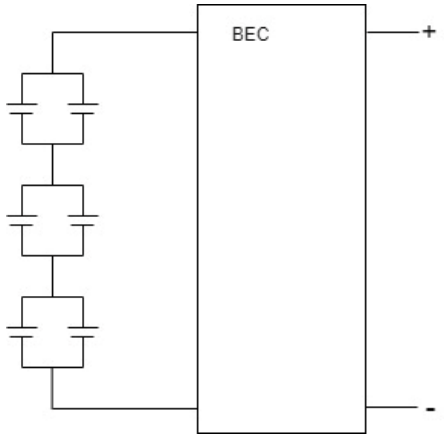
3.6.1 TCA9548 1xI2C master – 8xI2C slave expander

TCA9548 1xI2C master – 8xI2C slave expander je rozšiřující modul sloužící pro hardwarové rozšíření I2C sběrnice. Je v aplikaci použit z důvodu omezeného počtu portů I2C sběrnice na mikrokontroleru Raspberry Pi Pico. Raspberry Pi Pico má pouze dva I2C porty, kde jeden z nich je již obsazen MEMS gyroskopem a snímačem napětí a proudu. Tento modul má funkci multiplexoru, který umožňuje připojit až osm I2C portů, kde výstupem z modulu je pouze jeden I2C port. Tímto způsobem je možné k Raspberry Pi Pico připojit 4 optické senzory pro snímání okolních předmětů, které jsou umístěny na každé straně vozítka, i přesto že Raspberry Pi Pico má již pouze jeden volný I2C port.

Modul je vybaven 8 obousměrnými překladači umožňující ovládat pouze jeden I2C port. Jedná se tedy o jeden řídící master (SCL – Serial clock /SDA – Serial data) a osm řízených slave. V jeden okamžik lze ovládat buď jeden nebo více kanálu zároveň pomocí programovatelného registru v řídícím modulu. Použití je poměrně jednoduché, kdy samostatný multiplexor je na adrese 0x070 (lze adresu změnit od 0x070 do 0x077) kde na tento port je odesílán jeden bajt s konkrétním číslem odpovídající konkrétnímu slave I2C portu. Teoreticky lze ovládat až 64 I2C portů při použití na všech adresách. Master I2C modulu umožňuje RESET modulu při neočekávané funkci nebo nesprávné operaci, tím že nastaví vstup RESET do úrovně LOW. Modul lze napájet stejnosměrným napětím v rozsahu 1,65 – 5,5 V s maximálním proudovým odběrem 100 mA. Ovládat jednotlivý I2C porty je možné s frekvencí 400 kHz. [25][26]

3.7 Baterie

Pro napájení komponentů je v aplikaci použita baterie složená z bateriových článků. Celá baterie je složena ze tří 3S baterií, paralelně spojených, kde každá má napětí 3,7 V stejnosměrných. Celkové napětí baterie je přibližně 11,1 V stejnosměrných. Baterie je opatřena ochranou proti pod vybití a přebití baterie a slouží jako nadproudová ochrana (BEC – Battery Eliminator Circuit). Z důvodu napájení i komponentů s napájecím napětí aplikace obsahuje i DC/DC převodník na napětí stejnosměrných 5 V.



Obrázek 3.5 Vnitřní zapojení bateriových článků

3.8 I2C INA3221 senzor napětí a proudu

Další součástkou je senzor pro detekci přetížení a odběru proudu motorů při provozu a napětí na baterii. Modul umožňuje snímat hodnoty elektrického napětí a proudu na 3 kanálech. Jelikož má vozítko 2 řadiče pro ovládání motorů a jednu baterii je zapotřebí vybavit aplikaci 2 takovými moduly pro kontrolu všech motorů. Může být napájen stejnosměrným napětím v rozsahu 2,7 – 5,5 V a jeho proudový odběr při zatížení je 350 µA. Modul komunikuje s mikrokontrolerem přes I2C sběrnici nebo SMBUS (System Management Bus), a tudíž je možné při připojení k Raspberry Pi Pico využívat I2C sběrnici. Pracovní teplota modulu je v rozsahu -40–125 °C. Svými malými rozměry lze modul použít v aplikacích pro detekci softwarových elektrických obvodů nebo modelářské techniky. [27][28]

4. REALIZACE KONSTRUKCE

Následující kapitola pojednává o realizaci konstrukce robotického vozítka. V podkapitole 4.1 je popsána pohybová jednotka robotického vozítka v podobě omni kol. Podkapitola 4.1.1 pojednává o základním principu a důvodech použití všesměrových kol. Konkrétní typ omni kol, který je použit v této práci je popsán v podkapitole 4.1.2. V podkapitole 4.2 je blíže specifikován materiál, který byl použit pro 3D tisk se závěrečným zhodnocením výběru filamentu. Další podkapitola 4.3 pojednává o kompletaci konstrukce společně s potřebnými změny v návrhu.

4.1 Všesměrová kola (omni)

4.1.1 Základní popis

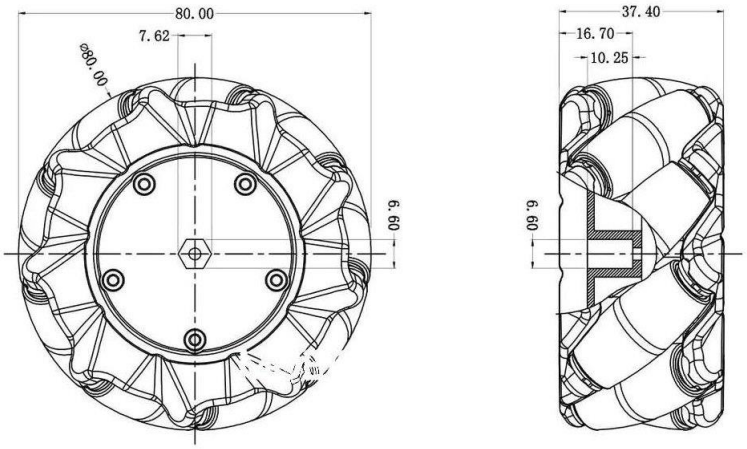
Cílem přepravního vozítka je co nejrychleji a nejefektivněji přepravit daný materiál (zakázku) na určené místo. Kdyby bylo zapotřebí pohybovat se s vozítkem nejen směrem dopředu a dozadu, ale i do stran (doprava či doleva, nebo příčně), bylo by nutné při použití kol s pevnou osou otočit vozítko požadovaným směrem. Tento problém pro usnadnění pohybu v prostoru řeší omni kola, která se bez nutnosti natočení vozítka, umí pohybovat všemi směry. Natočit se do libovolného úhlu je možné, bez vykonání kruhového pohybu. Omni kola byla vyvinuta a vyrobena tak, že umožňují pohyb všemi směry jen za pomoci jednoho rotujícího prvku, na kterém jsou osazeny pohyblivé valivé elementy ve tvaru válečků.

Jednotlivé válečky jsou umístěny po celém obvodu otáčejícího se kola. Tyto válečky jsou vůči směru kola mírně natočeny, čímž samotné kolo není schopno přímého pohybu. Pohyb kola není dán jen rotující částí vnitřní kola, nýbrž pohybujícími se válečky. Pokud je zařízení vybaveno pouze jedním takovým to omni kolem, není možné dosáhnout požadovaného cíle s přesností. Řešení je použít alespoň dvě takováto omni kola, která budou umístěna kolmo vůči sobě. Takovéto uspořádání zajišťuje přímý pohyb vozítka, z důvodu toho, že protilehlá omni kola mají válečky natočeny opačně. Tudíž lze říci, že se omni kola rozdělují na levá a pravá. Pokud však nelze zařízení vybavit více než jedním omni kolem je zde také řešení. V předchozím případě byli válečky pasivní (bez nutnosti pohonu), kdežto v tomto případě by musely být válečky aktivní. Zde by však mohl vzniknout problém s pohonem válečků, jelikož válečky rotují v pohyblivém pevném kole.[29] Konkrétní navržená omni kola použitá v této práci jsou popsány níže.

Obrázek 4.1 Ilustrační obrázek všesměrových kol [30]

4.1.2 Mecanum Omni kol ø80mm

Pro dané vozítko bylo z důvodu velikosti konstrukce a velikosti motorů vybrána omni kola o průměru 80 mm. Každé kolo je osazeno 9 pohyblivými válečky rovnoměrně rozmístěnými po celém obvodu kola. Válečky jsou uloženy pod úhlem 45° a nosností 10 kg. Kolo i válečky jsou vyrobeny z plastového materiálu s povrchovou úpravou se silikonové pryže. Povrchová úprava je velice odolná vůči nárazům a chod kola je tichý z důvodu vysokého koeficientu tření. Princip pohybu je založen na tzv. šestihranné všesměrové technologii, což umožňuje pohyb všemi směry (dopředu, dozadu, doleva, doprava a diagonálně všemi čtyřmi směry). Kola jsou označena příslušnými stranami (pravá a levá) z důvodu připevnění k vozítku a správnému chodu. Konkrétní rozměry daných kol jsou na obrázku 4.2. [31]



Obrázek 4.2 Mecanum Omni kol ø80mm [30]

4.2 Materiál pro tisk

Při tvorbě robotického vozítka bylo nutné vybrat vhodný materiál ze kterého budou jednotlivé části vytištěny. Níže jsou podrobně popsány zadavatelem dostupné filamenty, které byly použity pro realizaci autonomního robotického vozítka.

4.2.1 ASA

ASA je jeden z technických materiálů používaných pro 3D tisk. Jeho celý název je Acrylic Styrene Acrylonitrile, kde každé jeho slovo v názvu značí materiál, ze kterého je sloučen. Je považován za nástupce ABS, ale oproti tomuto materiálu je méně náchylnější na kroucení a dokáže více odolávat UV záření. Materiál je schopen odolávat vnějším teplotám dosahujících až 93 °C, kde do této teploty nedochází k deformaci materiálu. Dále je tento materiál vhodný pro použití při výrobě prototypů a součástek, či při tvorbě produktů, které budou používány ve venkovním prostředí.

Při tisku je nutné dodržet teplotu trysky okolo 260 °C. Další doporučenou podmínkou pro tisk je teplota podložky, která musí být během celého tisku udržována okolo 110 °C s výjimkou prvním základové vrstvy, kde doporučená teplota desky je 105 °C. Během celého tisku je vhodné mít umístěnou tiskárnu v místností s větráním, jelikož při teplotách tisku dochází k uvolňování škodlivých výparů z materiálu. Mezi hlavní nevýhody tohoto materiálu je jeho vysoké kroucení, které je způsobeno velkým rozdílem teplot při tisku a následném chladnutí. Tudíž ASA není vhodná pro tisk rozměrných výtisků, jelikož by při chladnutí mohlo dojít ke smrštění vrstev a následnému poškození výrobku. [32][33]

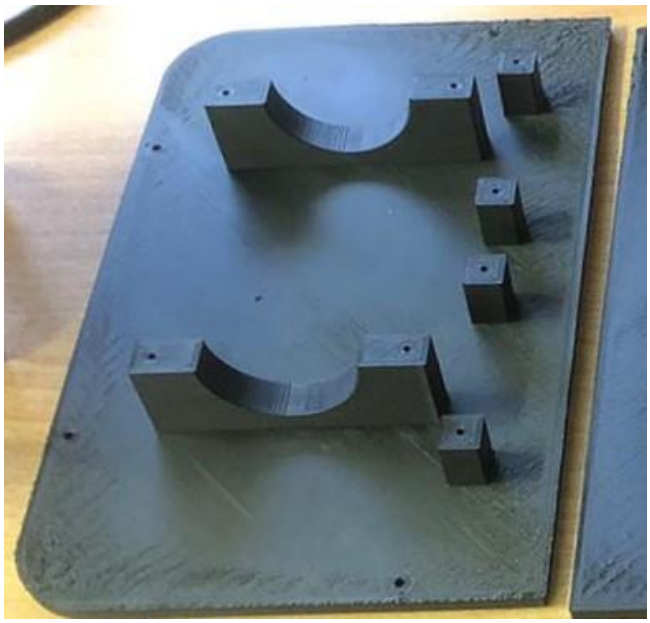
4.2.2 PCTG

PCTG je název pro jeden z materiálů používaných především pro 3D tisk. Jedná se o sloučeninu ze skupiny termoplastických polyesterů zvanou polycyklohexylendimethylen tereftalát. Má spoustu společných vlastností s materiálem PETG, a to především vnitřní strukturu materiálu. Jeho nespornou výhodou oproti výše zmíněnému materiálu ASA je vysoká soudržnost materiálu a zanedbatelné smrštění. Teplota, které jsou schopné předměty z tohoto materiálu odolávat je 76 °C. Používá se hlavně při tisku prototypů a modelů rozsáhlejších rozměrů které vyžadují dlouhodobou stálost struktury. Výrobky z tohoto materiálu se používají především v automobilovém průmyslu například pro přívod kapalin a další, jelikož je tento materiál dobře tvarovatelný.

Teplotu trysky během celého tisku je nutné udržovat okolo 270 °C, přičemž teplota stolu je shodná s materiálem ASA a to 90–110 °C. Jelikož tento materiál není tolik toxický, není nutné při tisku dbát na odvětrávání místnosti, ve které se nachází tiskárna. Dále je tento materiál považován za ekologický, jelikož je plně recyklovatelný. [34][35]

4.2.3 Tisk komponentů

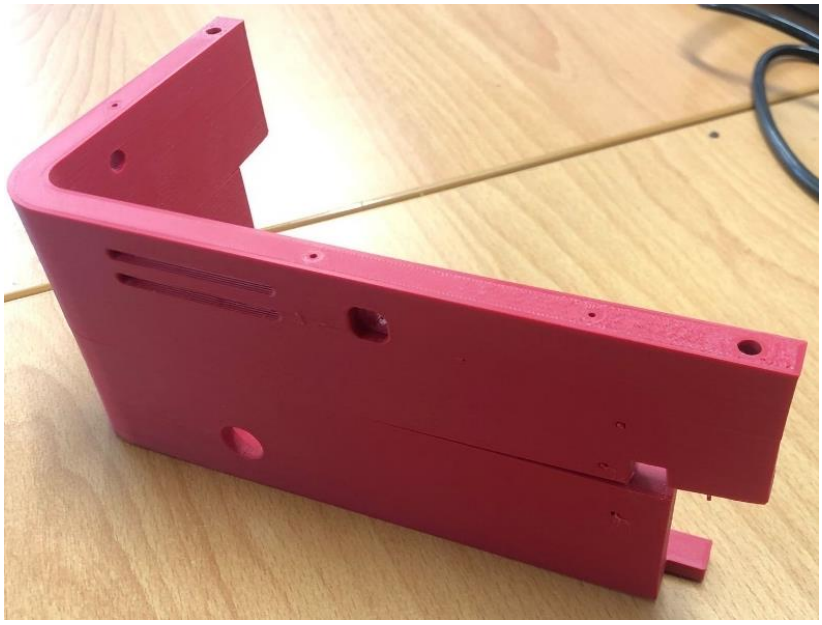
Při výběru z výše uvedených materiálů byl stěžejním parametrem materiálová stálost při chladnutí. Tedy jak moc se materiál bude deformovat při realizaci prvků větších rozměrů. Z důvodu dostupnosti filamentů ASA ve firmě byla konstrukce vozítka vytištěna z tohoto materiálu. Pro spodní díly byl použit více náchylný a méně stálý materiál ASA. Při tvorbě těchto dílů nedošlo k žádné deformaci a díly byly vytištěny v pořádku. Pro spodní díly byla použita ASA černé barvy. Volba barvy byla diskutována s konzultantem firmy. Výsledek tisku jednoho z těchto dílů je na obrázku 4.3.



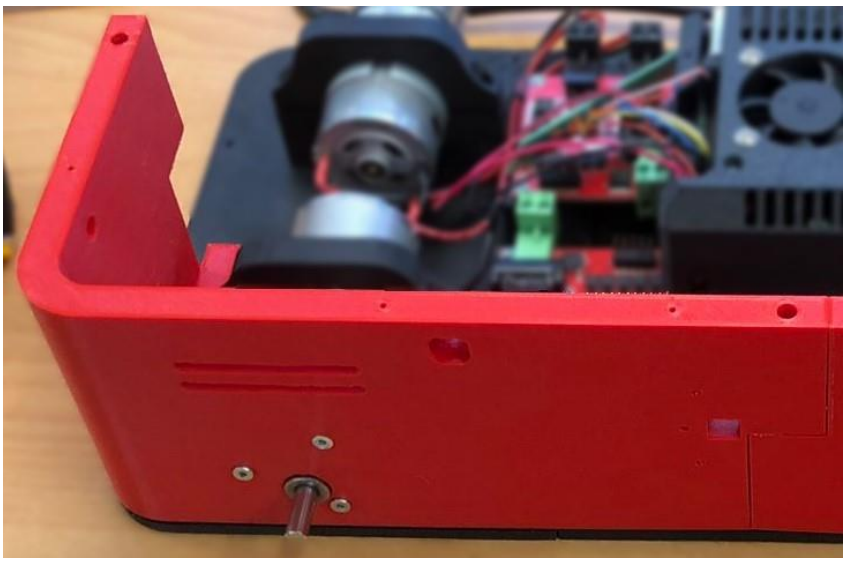
Obrázek 4.3 Podvozek – materiál ASA

Po vytištění a kontrole předchozích dílů byl pro tisk bočních stěn vybrán také materiál ASA, ale tentokrát ve variantě červené s označením cherry red. Tento materiál byl po vytisknutí jedné z bočních stran zhodnocen jako nevhodný, jelikož při chladnutí došlo k poškození dílu. Díl byl poškozen v úzkém spoji mezi otvorem pro snímač a boční hranou dílu. Rozsah poškození je vyobrazen na obrázku 4.4.

Nakonec byla po konzultaci se zadavatel zvolen filament PCTG v červené barvě s označením traffic red, který byl použit pro opětovné vytištění jedné boční stěny. Po tisku byla opět provedena kontrola a výsledek tisku byl již uspokojivý a nedošlo k žádnému poškození. Proto byly i další boční stěny vytištěny z materiálu PCTG. Výsledek tisku je možné vidět na obrázku 4.5.



Obrázek 4.4 Boční díl – materiál ASA



Obrázek 4.5 Boční díl – materiál PCTG

Jelikož vozítko obsahuje i další doplňkové komponenty menších rozměrů shodné barvy se stěnami (držák pro senzor, držáky motoru s užším poloměrem), byl i na tisk těchto komponentů použit materiál PCTG. Důvodem použití PCTG na tyto komponenty byl odstín a vnější struktura červené barvy oproti ASA v červené barvě. Na ostatní doplňkové komponenty (Držák pro gyroskop, držáky pro baterii, …) byl použit materiál ASA černé barvy.

4.3 Kompletace a úpravy

Při kompletaci výše uvedených vytištěných dílů společně s elektrickými a mechanickými komponenty potřebnými pro provoz vozítka, bylo nutné některé z částí konstrukce upravit či předělat a přetisknout. Mezi upravované díly patří držák pro senzor vzdálenosti, prostřední díl s kuželem pro kameru a přídavný nástavec na kolo, který nebyl do návrhu zahrnut.Pro kompletaci a spojení jednotlivých dílů konstrukce byly převážně použity šrouby velikosti B3x12 a 2.2x10. Jediným rozdílem bylo uchycení snímačů vzdálenosti na držák vždy pomocí jednoho šroubu velikosti B18x6. Pro uchycení komponentů v konstrukci vozítka (řadiče pro motory, DC/DC měnič, deska s plošným spojem, …) byly při kompletaci použity šrouby s označením B2.2x6. Všechny tyto šrouby jsou určeny primárně pro upevnění a pospojení plastů.

4.3.1 Nástavec na kolo

Při montáži výše uvedených omni kol na hřídel motoru bylo zjištěno, že montážní otvor v kole má jiný poloměr otvoru, než udává výrobce. Tudíž nebylo možné omni kolo připevnit na hřídel motoru. Po dohodě se zadavatelem práce bylo kolo demontováno a bylo nutné vytvořit mezikus pro přichycení redukce mezi motor a kolo. Na vnitřní stranu omni kola byl navržen a vytvořen mezikus ve tvaru kruhu, kde byly namodelovány mezery pro stávající části kola. První zkušební vzorek byl shledán jakožto nevhodný, jelikož při zpětné montáži kola, nebylo z důvodu velikosti mezikusu možné zkompletovat kolo. Problémem bylo vsunout válečky pod úhlem vhodném ke spojení obou stran kola. Druhý díl byl tedy zmenšen a následně byl tento mezikus připevněn k plastové části kola pomocí dvousložkového lepidla. Dále byly v mezikuse a do omni kola provrtáno 6 děr pro uchycení zakoupené redukce v podobě hliníkového náboje, který má vnitřní poloměr shodný s poloměrem hřídele. Výsledek navrženého mechanismu společně s vnitřním mezikusem je možné vidět na obrázku 4.6.

Obrázek 4.6 Mezikus a náboj na omni kolo

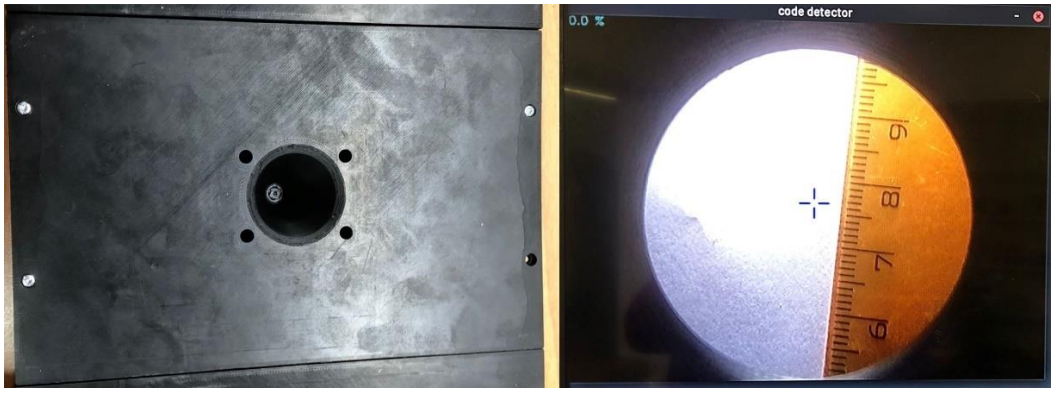
4.3.2 Držák na senzor vzdálenosti

Pro uchycení senzoru vzdálenosti byl navrhnut a vytvořen držící mechanismu, na kterém měl být upevněn snímač pomocí jednoho šroubu. Tento držák byl však vytvořen bez přesného 3D modelu daného senzoru. Tudíž při návrhu držáku nebyly zohledněny SMD komponenty na spodní straně destičky se senzorem, které na původním držáku způsobily nevyváženost destičky a nechtěný pohyb. Také nebylo možné připevnit horní díl s otvorem pro senzor, jelikož destička byla více vystouplá a pod jiným úhlem, než bylo navrženo. Z tohoto důvodu byla navržen a vytvořen nový držák pro senzor, který má v místě těchto komponentů vyhloubení čímž je zamezen pohyb destičky a také nyní lze připevnit na držák horní díl. Na obrázku 4.7 je možné vidět porovnání předchozího (horní díl) a nynějšího (spodní díl) držáku pro senzor vzdálenosti. Z obrázku je také patrné, že tvar a velikost společně s montážními otvory a otvorem pro přívodní vodiče zůstaly shodné jako v původním návrhu.

Obrázek 4.7 Spodní část držáku pro senzor vzdálenosti

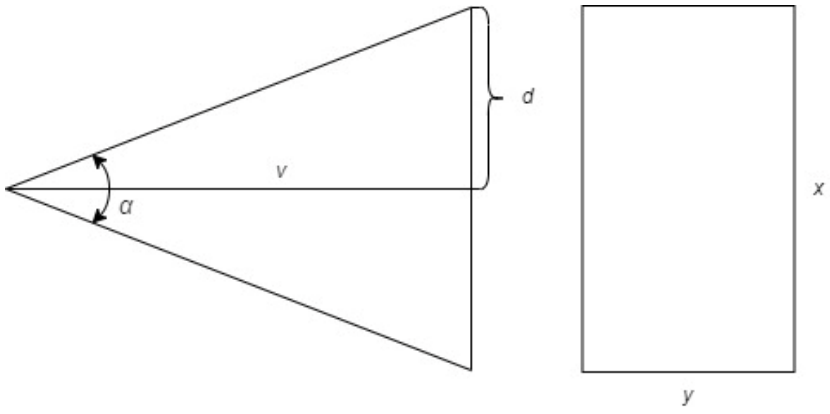
4.3.3 Střední díl podvozku

Poslední změnou při kompletaci robotického vozítka byla úprava středního dílu podvozku. Důvodem úpravy byl úzký otvor a nevhodně zvolený tvar otvoru pro kameru. Tato nevhodná volba se projevila po připojení a připevnění modulu s kamerou, kdy kamera měla zastíněnou část svého zorné pole. Tedy viděla jen část objektů pod vozítkem a zbytek byl zastíněn hranou kužele. Na obrázku 4.8 je znázorněna tato situace. Pro snímání QR kódů by byl kužel této velikosti dostačující, ale z důvodu vyšší pravděpodobnosti nalezení QR kódu je vhodnější, aby bylo kameře umožněno vidět co největší možný prostor.



Obrázek 4.8 Původní otvor pro kameru společně se zorným polem kamery

Z tohoto důvodu byl prostřední díl přepracován za účelem zvýšení pravděpodobnosti detekce QR kódu. Při realizaci nového dílu přicházela v potaz varianta stávajícímu kuželu zvětšit úhel náklonu. Toto řešení však nebylo možné realizovat, jelikož by kužel větší velikosti zasahoval do okolních komponentů, a tedy bylo by zapotřebí přeorganizovat vnitřní rozložení komponentů ve vozítku. Další variantou bylo změnit tvar stávajícího otvoru pro kameru. Zorné pole kamery by nebylo ve tvaru kruhu nýbrž obdélníku. Toto je možné pozorovat i na obrázku 4.10 na pravé straně. Výrobce udává, že zorný úhel po úhlopříčce je 79°. Byla přidána rezerva 1°, takže výsledný úhel je 80°. Výška kamery od spodní hrany podvozku je 41,5 mm. Názorná situace a postup výpočtu s konkrétními hodnotami je níže.



Obrázek 4.9 Označení parametrů k výpočtu otvoru pro kameru

$$d = v \cdot tan\frac{\alpha}{2}\ [mm]\ , \tag{4.1}$$

kde d je polovina délky daného obdélníku, v je výška kamery od podstavy rovna 41,5 mm a α je zorný úhel kamery 80°.

Poté polovina délky d obdélníku činí 34,8226 mm.

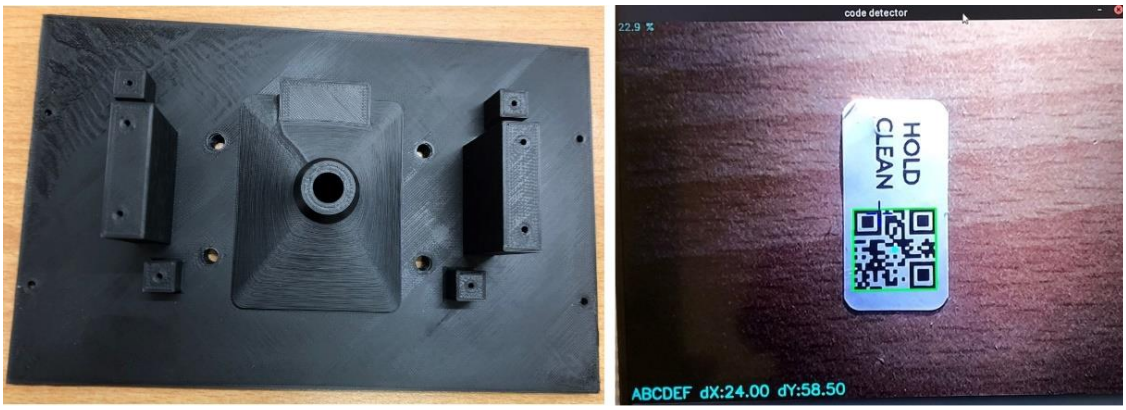
$$x = 2 \cdot d\ [mm]\ , \tag{4.2}$$

kde x představuje celkovou délku daného obdélníku.
Po dosazení je x rovno 69,6452 mm.

$$y = x \cdot \frac{r_s}{r_d}\ [mm]\ , \tag{4.3}$$

kde y je celková šířka daného obdélníku, r_s a r_d představují hodnoty aktuálně nastaveného rozlišení kamery 600 x 800 px.

Po dosazení je y šířka obdélníku rovna 52,2339 mm. Po zaokrouhlení rozměrů na nejbližší vyšší celou hodnotu byl vytvořen otvor s obdélníkovou podstavou 70 x 53 mm. Při této změně bylo ještě zapotřebí poupravit podpěrné nohy řadiče motorů, protože při rozšířeném otvoru pro kameru by z důvodu tloušťky materiálu nebylo možné do stěny upevnit šrouby. Proto zde vnitřní podpěrné nohy budou sloužit jen jako podpěry, ale nebude možné do těchto nohou upevnit šroub. Tedy řadiče budou v konstrukci upevněny pouze za pomoci tří šroubů ze čtyř možných. Avšak po odzkoušení bylo zjištěno, že tyto tři upevňovací body jsou zcela dostačující pro danou aplikaci.



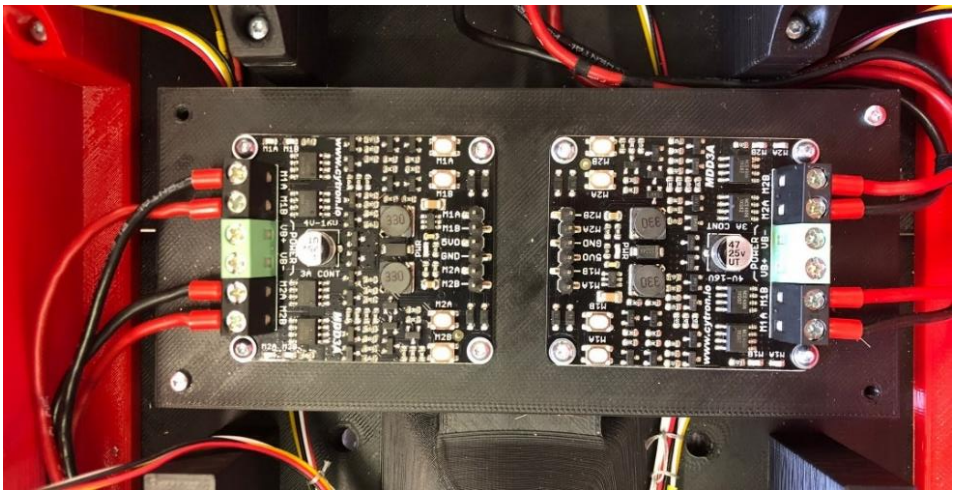
Obrázek 4.10 Nový otvor pro kameru společně se zorným polem kamery

4.3.4 Řadič MDD3A 3A 4V-16V DC

Při oživení elektrických komponentů bylo zjištěno, že výše uvedené řadiče (Řadiče pro DC motory 7 A 160 W), které měly být v aplikaci použity jsou z neznámých důvodů nefunkční. Nefunkčnost řadičů se projevovala špatným ovládáním. Zapojení a inicializace pinů byla provedena přesně tak, jak uvádí výrobce v manuálu, ale nebylo možné provést reverzaci motoru. Dále by řadič měl být schopen ovládat vždy dva motory, ale jeden z řadičů ovládal pouze jeden motor. Z těchto důvodů po konzultaci se zadavatelem bylo shledáno vyměnit dosavadní řadiče za nové. Byl vybrán jiný typ řadiče, jelikož pořizovací cena byla nižší, řízení jednodušší a pro danou aplikace dostačující.

Dále jsou tedy v aplikaci pro ovládání motorů použity řadiče MDD3A, které jsou vhodné pro ovládání stejnosměrných motorů s napětím 4-16 V. Každý z řadičů je

schopen ovládat 2 motory. Řadiče obsahují 5 vstupních signálů a jeden výstupní 5 V, který není připojen, jelikož slouží pro napájení mikrokontroleru. Řadiče jsou k řídícímu mikrokontroleru připojeny tedy pouze pomocí 5 signálových pinů. Původní řadiče byly k Raspberry Pi Pico připojeny přes 8 pinový konektor. Tento konektor i deska plošného spoje byla zachována, pouze do konektoru bylo připojeno jen 5 signálových vodičů. Jedná se o signály M1A, M1B, M2A, M2B a GND. Signály M1A a M1B slouží pro ovládání jednoho motoru a M2A a M2B ovládají druhý motor. Pokud jsou oba signály (MxA a MxB) v logické 1 nebo 0, tak je motor zabrzděn. Chod obou motoru vpřed je umožněn tak, že na signál M1A a M2A je přivedena logická 1 (PWM signál) a na M1B a M2B logická 0. Chod motorů vzad je realizován prohozením logických úrovní na těchto signálech. [36]

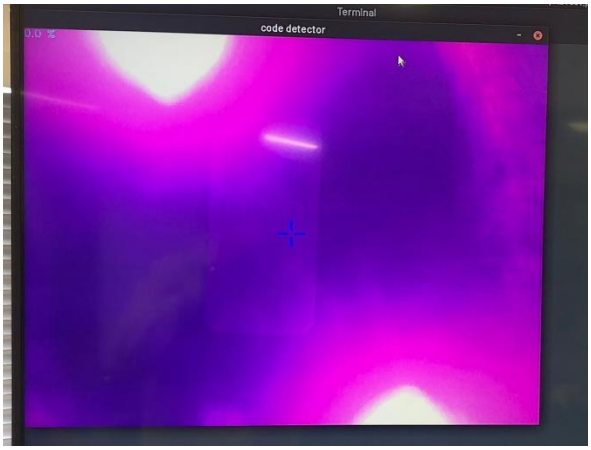


Obrázek 4.11 Řadič MDD3A s upevněním

Jak je z obrázku 4.11 patrné, nové řadiče mají odlišnou velikost oproti původně vybraným řadičům. Tedy bylo zapotřebí vyřešit problém s montáží, jelikož podpěrné nohy pro uchycení řadičů byly navrženy na původní řadiče. Řešení bylo vytvoření desky s rozměry vhodnými k uchycení do původních 4 podpěrných noh a možností upevnění dvou nových řadičů.

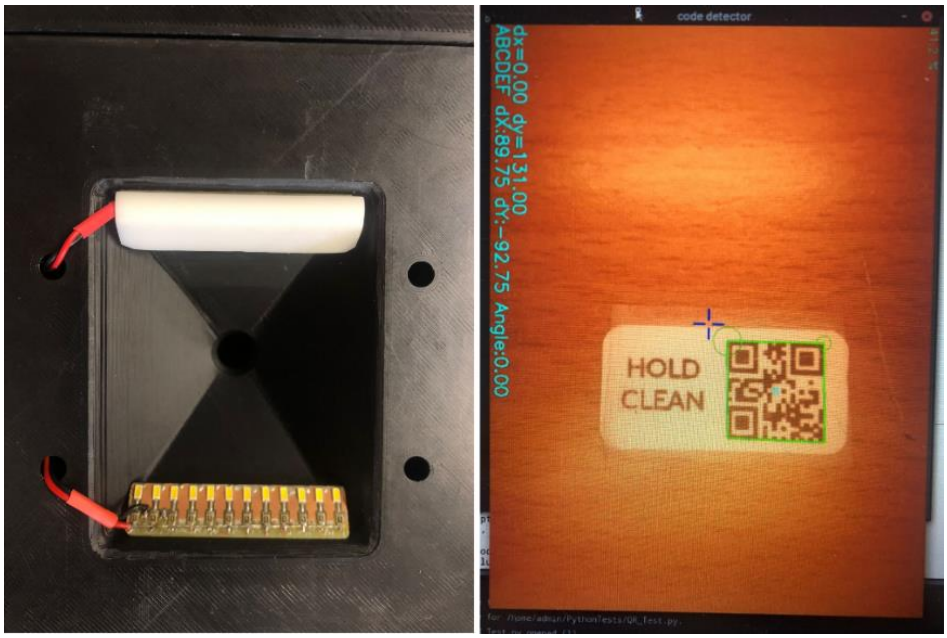
4.3.5 Přisvětlovací LED

Aby byla kamera schopna snímat QR kódy umístěné na podlaze, bylo nutné vozítko vybavit přisvětlovacími LED. Tyto LED slouží pro osvětlení prostoru pod vozítkem, snímatelnou oblast kamery, jelikož vozítko vytváří zastíněnou oblast pod podvozkem. Původně byla navrhnuta varianta použití 4 infračervených, popřípadě klasických bílých LED. Pro tyto LED byly na podvozku vytvořeny výše zmíněné otvory. Nicméně při použití této varianty bylo zjištěno, že LED tvoří bodový zdroj světla. Tedy oblast pod těmito LED byla v zorném poli kamery příliš přesvícená a ostatní oblasti naopak tmavé. Výsledek použití této varianty je zobrazen na obrázku č. 4.12.



Obrázek 4.12 Snímek z kamery – IR LED

Po dohodě se zadavatel byla zvolena varianta použití LED pásku. Tento LED pásek je umístěný na podvozku v otvoru pro kameru a připevněn k podvozku pomocí oboustranné samolepící pásky. Během testování bylo zjištěno, že pro danou úlohu bude dostačující použití dvou takovýchto LED pásků, které budou umístěny naproti sobě. Dále bylo zjištěno, že obraz z kamery je taktéž lehce přesvětlen, a proto byly na tyto pásky vytištěny ochranné kryty vytištěny z bílého filamentu PCTG. Tento filament je při osazení průsvitný a ve výsledku zmatňuje a rozptyluje světlo vyzářené z LED pásku. Předpřipravené otvory pro umístění LED byly využity pro připojení přívodních vodičů. Aby byla zajištěna vyšší pravděpodobnost snímání objektu, byly QR kódy přelepeny matnou pásku za účelem snížení odlesků.



Obrázek 4.13 Snímek z kamery – LED pásek

5. ELEKTRICKÉ SCHÉMA ZAPOJENÍ

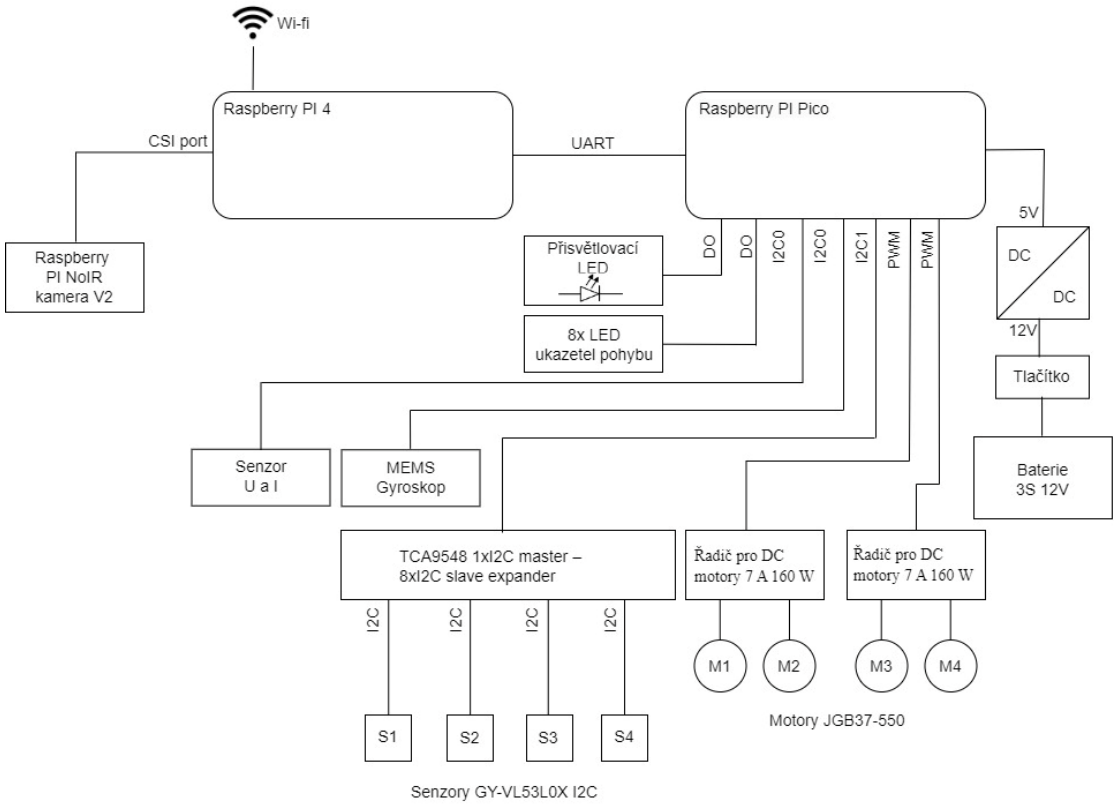
Následující kapitola pojednává o elektrickém zapojení jednotlivých komponentů v aplikaci. V podkapitole 5.1 je navrhnut koncept tohoto zapojení. V podkapitole 5.2 je popsáno elektrické schéma zapojení komponentů společně s návrhem a realizací plošného spoje.

5.1 Koncept

Celá soustava je napájena z baterie s dodávaným stejnosměrným napětím 12 V. Toto napětí musí být přetransformováno na 5 V stejnosměrných pomocí DC/DC měniče napětí. Důvodem transformace napájecího napětí je napájení řídících mikrokontrolerů Raspberry a dalších elektrických komponentů.

O snímání dat z kamery a odesílání dat přes Wi-Fi se stará mikrokontroler Raspberry Pi 4. Tento mikrokontroler je narozdíl od Raspberry Pi Pico vybaven Wi-Fi modulem a CSI portem, pomocí něhož je možné připojit výše uvedenou kameru. O zpracování dat z MEMS gyroskopu, senzoru polohy a napětí/proudu se naopak stará Raspberry Pi Pico, které je vybaveno I2C porty. Jelikož Raspberry Pi Pico nemá dostatečné množství I2C portů bylo nutné aplikaci dovybavit rozšiřovacím modulem, který umožňuje připojit až 8 nezávislých I2C modulů a následně pomocí multiplexoru odesílat data pouze přes jeden I2C port. Dále tento mikrokontroler ovládá drivery (řadiče) motorů pomocí PWM signálu. Na Raspberry Pi Pico jsou dále přes DO signál připojeny LED, které indukují pohyb vozítka.

Aby nadřazený řídící software mohl přijímat a zpracovávat data, je nutné propojit oba řídící mikrokontrolery (Raspberry Pi 4 a Raspberry Pi Pico). Nejjednodušší variantou je propojit tyto dva mikrokontrolery mezi sebou prostřednictvím portu UART. Veškerá předzpracovaná data poté budou pro další zpracování předávána nadřazenému řídícímu softwaru. Tyto data se budou předávat přes Wi-Fi připojení, které umožňuje Raspberry Pi 4. Blokové schéma zapojení celé elektrické části je na obrázku 5.1.



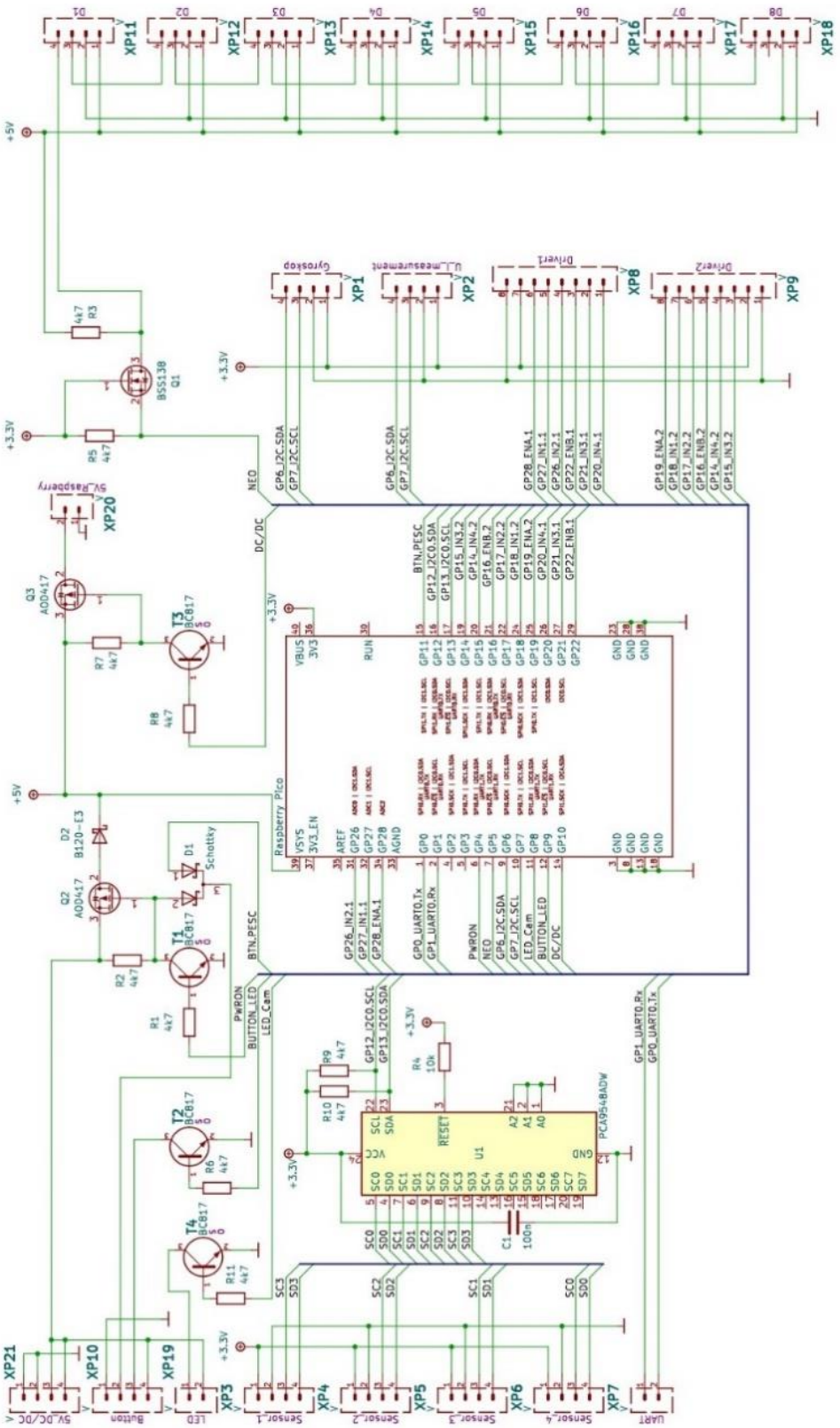
Obrázek 5.1 Blokové schéma zapojení

5.2 Realizace elektrického zapojení

Pro návrh a konečnou realizaci elektrického schéma zapojení byl použit otevřený a bezplatný nástroj KiCad 7.0. Jedná se o nástroj vhodný pro realizaci elektrických zapojení a následného vytvoření návrhu pro tvorbu plošného spoje (PCB). Mezi dané aplikace, které byly využity pro realizaci plošného spoje patří:[37]

- **KiCad Schematic Editor** – tato aplikace slouží pro návrh elektrického schématu v podobě výběru a následného umístění a propojení elektrických komponentů. Tento editor nabízí již zmiňovanou knihovnou symbolů, kterou si uživatel může dle svých požadavků upravovat a měnit s možností přidání svých vlastních symbolů.[38]
- **Editor PCB** – tato aplikace se zabývá realizací nakresleného schématu v Schematic Editoru na desku plošného spoje. Pomocí tlačítka update lze rychle nahrát změny, které se provedly v návrhu schématu. V tomto editoru je možností omezit velikost výsledného schématu v podobě reálné velikosti plošného spoje. Výsledný plošný spoj lze zobrazit jako 3D model pomocí 3D vieweru. Při realizace PCB lze měnit tloušťku měděného spoje a mezeru mezi spoji. Také tento nástroj umožňuje navrhovat vícevrstvé plošné spoje.[38]

5.2.1 Schéma zapojení



Obrázek 5.2 Elektrické schéma zapojení

Po vhodném výběru komponentů a blokovém schématu zapojení bylo realizováno pomocí nástroje KiCad elektrické schéma zapojení. Výkres elektrického schématu je patrný z obrázku 5.2, který slouží pro základní popis funkce.

Základem celého schématu je Raspberry Pi Pico, které je hlavním řídícím prvkem celé soustavy. Je se všemi komponenty ve schématu propojen pomocí BUS sběrnice, která je zde použita pro větší přehlednost celého schématu. Raspberry Pi Pico je napájeno 5 V a napájení je připojeno na pin VSYS. Řídící mikrokontroler je s druhým mikrokontrolerem Raspberry Pi 4 propojeno pomocí konektoru XP7 přes UART komunikaci. V horní části schématu se nacházejí SMD součástky s potřebnou funkcí pro realizaci a funkci daného zapojení. Tranzistor Q2 a T1 společně s rezistory R1, R2 a s diodami D1 a D2 tvoří elektrický obvod sloužící jakožto přidržovací mechanismus pro spínací kontakt použitý k zapnutí napájení celé soustavy. Tento stav detekuje pin na Raspberry Pi Pico GP11 (BTN.PESC). Jako řídící pin pro tuto funkci a pin pro zapnutí napájení slouží GP4 (PWRON). K propojení DC/DC měniče s tímto obvodem slouží 4 pinový konektor XP21. Pro zapnutí a vypnutí celého napájení a uvedení vozítka do stavu nečinnosti slouží tlačítko připojené ke konektoru XP10, který je vybaven přisvětlovacími LED. Tyto LED se ovládají pomocí pinu GP9 (BUTTON_LED) a tranzistoru T2 společně s rezistorem R6. Pro spouštění přisvětlovacích LED ke kameře slouží port XP19, který je ovládán pomocí pinu GP8 (LED_Cam) společně s T4 a R11. Dále je zde část obvodu, která je zaměřena na napájení Raspberry Pi 4. Je nutné, aby bylo nejdříve zapnuto napájení pro Raspberry Pi Pico a až poté bylo napájeno Raspberry Pi 4. Tato část obvodu je realizována přes pin GP10 (DC/DC) a součástek R7, R8, T3 a Q3. K tomuto obvodu je připojen konektor XP20.

Další část elektrického zapojení je tvořena tranzistorem Q1 a rezistory R3 a R5, která pomocí pinu GP 5 (NEO) na Raspberry Pi Pico slouží k ovládání LED indikujících směr pohybu vozítka. Tyto LED jsou připojeny na konektory XP11 až XP18, kde jsou mezi sebou propojeny pomocí signálů DI a DO. Jednotlivé LED jsou napájeny pomocí 5 V.

Gyroskop (konektor XP1) a snímač proudu či napětí (konektor XP2) jsou připojeny k Raspberry Pi Pico pomocí signálů SDA a SCL na I2C sběrnici. Tato sběrnice se na řídícím mikrokontroleru nachází na pinech GP6 a GP7. Na pinech GP14 – GP22 a GP26 – GP28 jsou k Raspberry Pi Pico připojeny řadiče k motorům. Jsou připojeny pomocí 8 pinového konektoru.

Dále je ve schématu možné vidět přídavný I2C multiplexor, který je k řídícímu mikrokontroleru připojen přes piny GP12 a GP13 I2C sběrnice. Tomuto multiplexoru bylo dle datového listu nutné připojit RESET pin přes pull-up rezistor na + 3,3 V. K multiplexoru jsou připojeny všechny 4 snímače vzdálenosti na kanálech 0–3. Senzory jsou připojeny přes 4 pinové konektory XP3 – XP7.

Všechny výše uvedené elektrické komponenty společně s konektory byly do schématu vloženy z knihovny. Tato knihovna byla poskytnuta zadavatelem firmy. Při vkládání komponentů bylo nutné u každé ze součástek nastavit hodnotu a footprint.

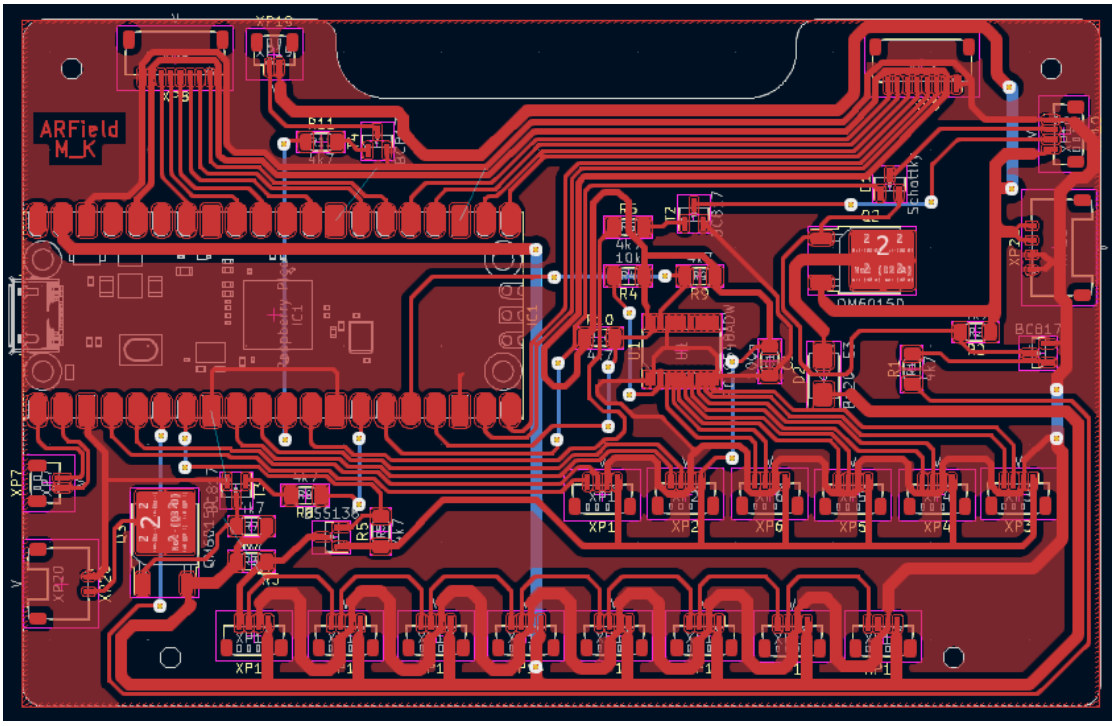
Footprint je nezbytný pro realizaci plošného spoje, aby byla zajištěna shodnost reálných dílů s navrženou deskou pro následnou kompletaci a pájení.

5.2.2 Návrh plošného spoje

Základem pro návrh plošného spoje bylo vytvoření reálného tvaru a velikosti výsledné desky na kterou bude plošný spoj vyroben. Velikost a tvar byl navržen již při tvorbě konstrukce vozítka s ohledem na možnou velikost desky. Tato deska má tvar obdélníku s rozměry 110 x 70 mm s výřezem na jedné z delší stran. Důvodem výřezu byl omezený prostor způsobený změnou tvaru otvoru pro kameru. Deska byla z DesignSpark Mechanical exportována jako soubor .dxf a následně vložena do prostředí KiCad jako vrstva Edge.Cuts. Tím byla vytvořena ohraničující 2D část, která sloužila jako omezující prostor pro umístění komponentů a propojení mezi nimi.

Rozmístění komponentů společně s tvarem desky a propojkami je možné vidět na obrázku 5.3. Základním a hlavním prvkem byl opět řídící mikrokontroler Raspberry Pi Pico, který byl umístěn na jednu z kratších stran desky tak, aby bylo možné i při montáži desky plošného spoje do vozítka připojit USB konektor pro připojení k PC. Dále byl z destičky I2C multiplexoru použit jen čip. Tudíž bylo možné umístit multiplexor přímo na plošný spoj a nemusel být umístěn na přídavných nohách v konstrukci vozítka, jak bylo původně v plánu. Konektory pro připojení řadičů motorů jsou umístěny na delší hraně desky plošného spoje, tudíž je možné vést propojovací kabeláž pod modulem s kamerou (okolo otvoru pro kameru). Další dva konektory XP21 a XP20 jsou umístěny na kratší straně destičky. Všechny tyto 4 konektory mají horizontální rozložení, tudíž lze k nimi připojit konektor pouze v horizontální směru (z boku).

Všechny ostatní konektory jsou již ve vertikální směru připojení (shora). Pod tento typ konektoru patří konektor sloužící k připojení přisvětlovacích LED pod kamerou umístěný vedle konektoru pro řadič. Tato pozice byla zvolena z důvodu umístění LED v konstrukci vozítka. Dále jde o konektory pro LED indikátory směru umístěné na protější delší straně plošného spoje. V poslední řadě se jedná o konektory pro připojení gyroskopu, snímače napětí a proudu, a nakonec o senzory vzdálenosti. Všechny ostatní elektrické komponenty byly rozmístěny s ohledem na propojení mezi nimi.



Obrázek 5.3 Návrh desky plošného spoje

Při realizaci a propojení jednotlivých komponent byly na propojovací spoje použity tři různé tloušťky spojů. Jednalo se o spoje 0,406 mm, 0,610 mm a 1,270 mm. Tloušťka spoje byla volena vždy s ohledem na velikost protékajícího proudu spojem. Tedy například všechny spoje pod napětím 5 V byly taženy spoji s tloušťkou 1,270 mm. Na všechny spoje pod napětím 3,3 V byla použita tloušťka spoje 0,610 mm a všechny ostatní spoje (datové spoje) mají tloušťku 0,406 mm. Všechny tyto spoje jsou na obrázku 5.3 znázorněny červenou barvou a jsou nakresleny ve vrstvě F.Cu.

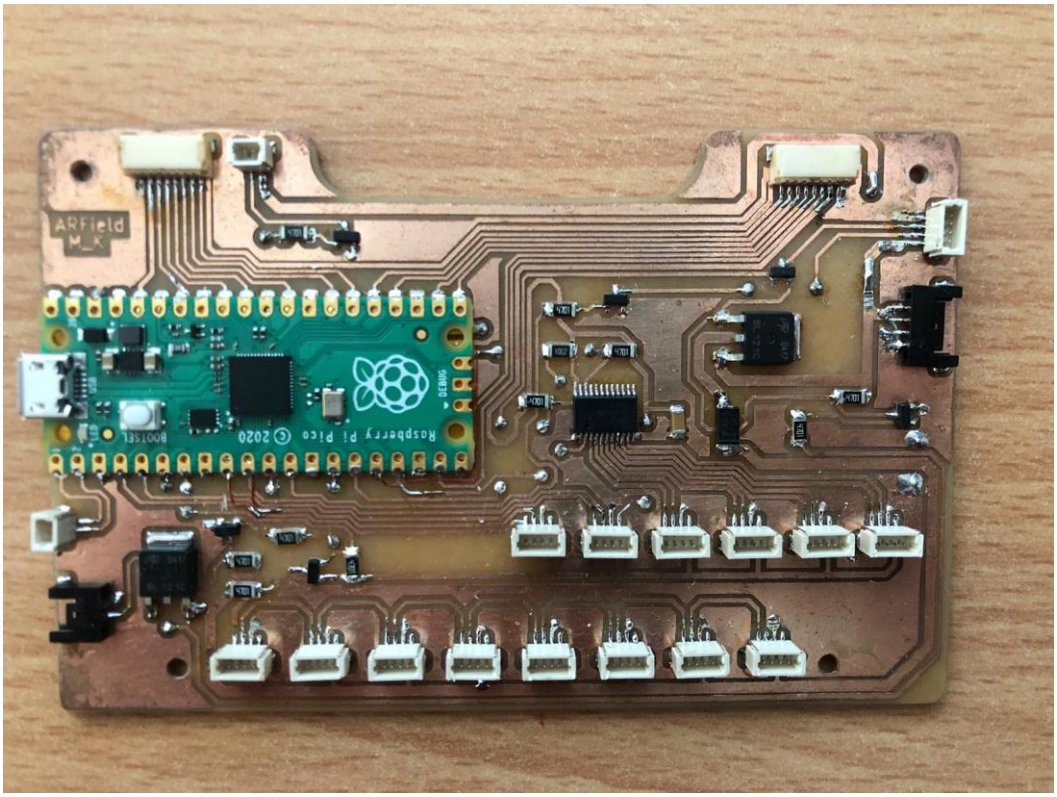
Jelikož došlo ke křížení některých ze spojů a nebyla zadavatelem možná realizace vícevrstvé desky, bylo zapotřebí tyto spoje propojit měděnými propojkami na druhé straně desky plošného spoje. Jedná se o propojky na vrstvě B.Cu, které jsou znázorněny na obrázku 5.3 modrou barvou.

5.2.3 Realizace plošného spoje

Při montáži a osazování plošného spoje došlo k mírným úpravám týkajících se desky plošného spoje. Následné úpravy se týkaly I2C sběrnice. První nedostatek desky vznikl při připojování gyroskopu a snímače napětí a proudu na jednotlivé konektory, které se nacházejí na desce. Tyto dva snímače komunikují na stejné sběrnici I2C1 a aby bylo docíleno této komunikace je zapotřebí připojit oba snímače na stejný pin na Raspberry Pi Pico. Na Raspberry se nachází tato stejná sběrnice (I2C1) na více pinech, které mezi sebou ale nejsou propojeny. Tento problém se projevoval tak, že bylo zapotřebí sběrnici inicializovat dvakrát, kdy při druhé inicializaci však došlo ke ztrátě již získaných dat

z prvního ze snímačů. Druhá změna, která byla provedena byla z důvodu prohození signálu SCL a SDA u I2C multiplexoru. Kde byl pin SDA na multiplexoru připojen na pin SCL na Raspberry Pi Pico a pin SCL na multiplexoru byl připojen na SDA. Toto zapojení by však nebylo funkční a bylo taktéž zapotřebí úpravy.

Po dohodě se zadavatel bylo na řešení těchto nedostatků použita metoda přerušení spoje a následné propojení pomocí měděného drátu. Potřebné spoje byly tedy přepojeny přímo na vytištěné desce a jelikož se jedná pouze o prototyp vozítka, tak nová deska se již nevyráběla. Pokud se ale pro budoucí použití bude jednat o sériovou výrobu, bude vyrobena nová již upravená deska. Možné řešení úprav a výsledná realizace plošného spoje je na obrázku 5.4.



Obrázek 5.4 Výsledná realizovaná deska plošného spoje

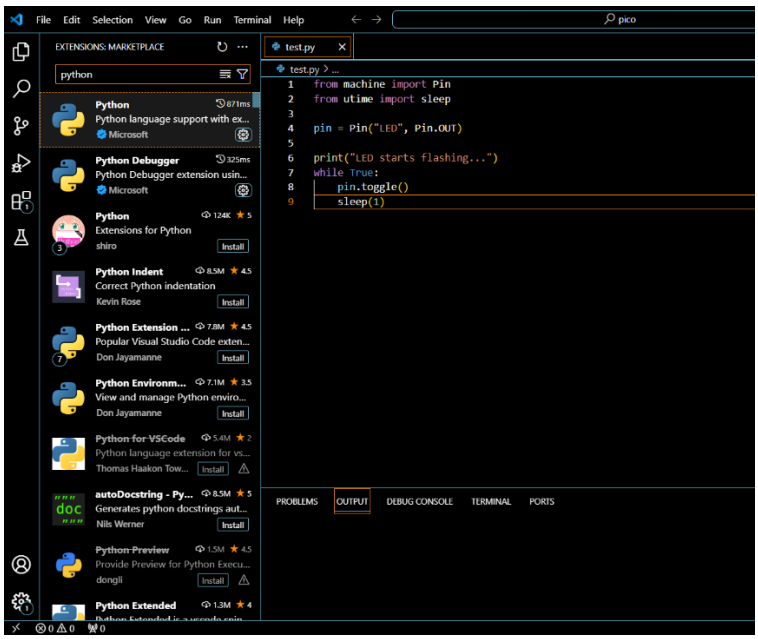
6. IMPLEMENTACE ŘÍDÍCÍHO A KOMUNIKAČNÍHO SOFTWARU

Tato kapitola je zaměřena na řídící a komunikační software pro robotické vozítko. První podkapitola 6.1 popisuje editor Visual Studio Code, ve kterém je napsán firmware pro robotické vozítko. V následující podkapitole 6.2 je podrobně představeno řešení řídícího softwaru v mikrokontroleru Raspberry Pi Pico. Poslední podkapitola 6.3 pojednává o softwaru vytvořeném pro obsluhu Raspberry Pi 4.

6.1 Visual Studio Code

Pro tvorbu a realizaci řídícího a komunikačního softwaru byl po dohodě se zadavatelem zvolen editor Visual Studio Code. Tento editor je vyvíjen společností Microsoft a první verze byla zveřejněna v roce 2015. Jedná se o volně dostupný editor, který lze používat na operačních systémech jako jsou například Linux, Windows nebo MacOS. Podporuje programovací jazyky typu Java, C++, C#, Python a další. [39][40]

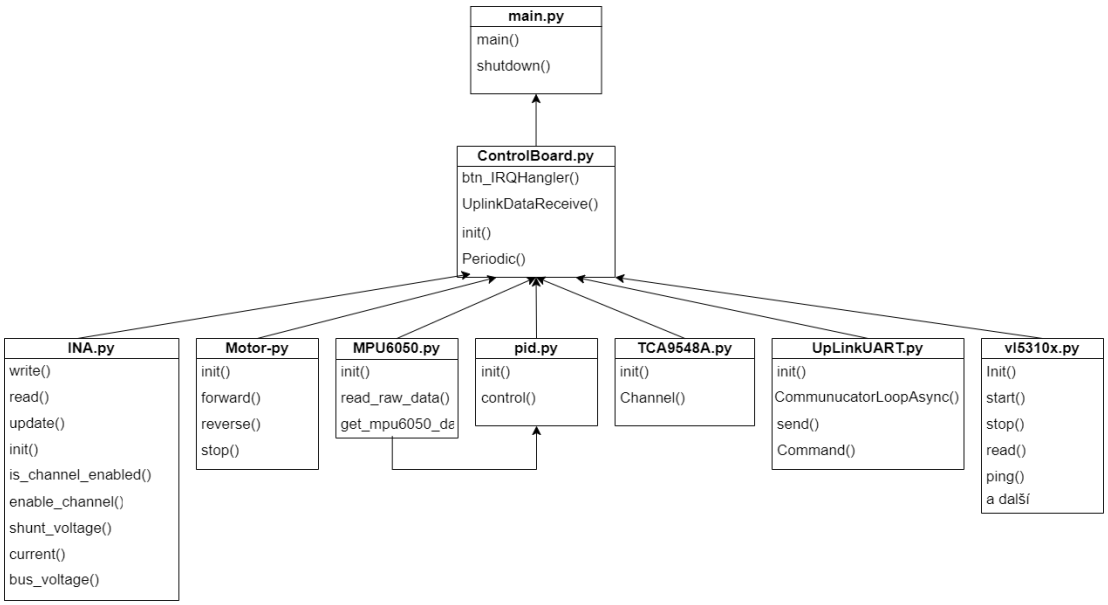
Visual Studio Code lze rozšířit o další doplňkové moduly což umožní uživateli přizpůsobit si své programovací prostředí podle svých potřeb. Tato rozšíření lze instalovat přímo ve Visual Studio Code a hlavním přínosem je podpora pro různé programovací jazyky [41]. Nejvýznamnějším prvkem tohoto rozšíření je automatické dokončování. Tato vlastnost umožňuje doplnit uživateli již rozepsaný příkaz, který se doplňuje dle kontextu v předchozí již vytvořené části. Na obrázku 6.1 níže je zobrazeno prostředí Visual Studio Code.



Obrázek 6.1 Editor Visual Studio Code

6.2 Software v Raspberry Pi Pico

Níže v následujících podkapitolách 6.2.1 – 6.2.9 jsou popsány jednotlivé skripty vytvořené pro obsluhu mikrokontroleru Raspberry Pi Pico. Tyto soubory jsou psané v jazyce MicroPython. Následující podkapitoly byly rozděleny dle jednotlivých .py souborů a jejich struktura je zobrazena na obrázku 6.2.



Obrázek 6.2 Struktura projektu v Raspberry Pi Pico

6.2.1 INA.py

Tento skript slouží pro inicializaci a čtení dat ze snímače napětí a proudu. Část kódu byla převzata z již vytvořeného skriptu umístěného na GitHubu. Tato část byla upravena pro potřeby a využití v dané aplikaci [42]. Jelikož byl původní skript napsán v jazyce CircuitPythonu, bylo zapotřebí ho přepsat do jazyka MicroPython. Třída INA3221 obsahuje tyto nezbytné funkce:

- **write** – tato funkce umožňuje zápis dat do paměti zařízení. Je zde použit příkaz writeto_mem namísto příkazu write, aby byl skript spustitelný v jazyce Micropython. Tento příkaz potřebuje tři parametry. První parametr určuje adresu zařízení, kam se zapisují data, druhý parametr obsahuje adresu registru a poslední parametr je pole dat, které se zapisuje.
- **read** – tato funkce je ve skriptu použita pro čtení z registru paměti zařízení, kde jsou uložená naměřená data. Je zde použit příkaz readfrom_mem, aby mohl být skript spuštěn v MicroPythonu. Tento příkaz také obsahuje tři parametry. První dva parametry určují adresu zařízení a registr, ze kterého mají být data vyčtena. Poslední parametr určuje počet bajtů, které mají být vyčítány.

- **update** – tato funkce využívá funkcí read a write, aby byla možná aktualizace bitu v registru. Nejdříve jsou načteny hodnoty v registru, který má být aktualizován a poté se pomocí masky určuje, které bity mají být změněny. Nakonec je této funkci předán parametr value, kde je nová hodnota, která má být do registru zapsána.
- **init** – nastavuje se zde komunikace přes I2C sběrnici a adresa zařízení. Dále se zde nastavuje hodnota shunt rezistorů na jednotlivých měřících kanálech. Na kanálu CH1 je hodnota rezistoru 0,1 Ω a na kanálech CH2 a CH3 je hodnota odporu 0,02 Ω. Hodnota odporu je závislá na použitých rezistorech na destičce senzoru. Konfigurace senzoru je zapsána do zařízení přes funkci write.
- **enable_channel** – tato funkce umožňuje povolení nebo zakázání kanálu tím, že aktualizuje příslušný bit v registru konfigurace zařízení pomocí metody update. Kontroluje se zde zda vybraný kanál pro povolení či zakázání je platný v rozsahu 1-3.
- **shunt_voltage** – tato funkce měří úbytek napětí na jednotlivých kanálech při připojení zátěže. Naměřená data se dále využívají pro výpočet proudu (current funkce).
- **current** – tato funkce získává proudy na jednotlivých kanálech zařízení. Na kanál CH1 je připojen DC/DC měnič, na kanálu CH2 je připojen jeden řadič pro motory a na posledním kanálu CH3 je připojen druhý řadič. Proud je získáván přes výpočet. Při výpočtu proudu je nejprve zavolána funkce shunt_voltage pro získání napětí na shunt rezistoru pro daný kanál. Poté se toto napětí vydělí shunt rezistorem, který odpovídá tomuto kanálu.
- **bus_voltage** – tato funkce vrací z registu naměřené napětí na sběrnici pro daný kanál. Pomocí tohoto napětí je možné identifikovat napětí na baterii, která napájí celé robotické vozítko.

6.2.2 Motor.py

Následující skript Motor.py slouží v aplikaci pro inicializaci a ovládání řadičů pomocí PWM. Konkrétní přiřazení pinů, na kterých jsou řadiče připojeny se provádí v konstruktoru skriptu ControlBoard.py. Základem je zde konstruktor init(), který definuje signály, kterými je řadič ovládán. Jsou zde vytvořeny dvě PWM, které slouží pro řízení rychlosti motorů. Při inicializaci jsou PWM nastaveny na nulu. Dále tento skript obsahují dvě metody forward() a reverse(), které mají parametr speed. Pomocí tohoto parametru lze v ControlBoardu.py měnit rychlost otáčení motorů. Metoda forward() složí pro pohyb vpřed a reverse() pro pohyb vzad. Jak nastavit výstupy (MxA, MxB) u řadičů, aby byl chod motoru dopředný či zpětný je popsán v podkapitole 4.3.4. Pro zastavení motoru je volána metoda Stop(), kde jsou PWM výstupy nastaveny na nulu.

6.2.3 MPU6050.py

Tento skript slouží pro inicializaci a čtení dat z gyroskopu MPU6050, skládá se z konstruktoru Init() a ze dvou funkcí read_raw_data() a get_mpu6050_data(). V konstruktoru probíhá inicializace jak I2C sběrnice, tak se zde nastavují i potřebné registry pro správu napájení a konfiguraci senzoru. Funkce read_raw_data() slouží pro získání dat, která byla neměřena ze senzoru. A poslední část skriptu obsahuje funkci get_mpu6050_data(), která umožňuje získaná data ze senzoru převést na reálné hodnoty. Převod těchto hodnot je uveden v dokumentaci k senzoru. Všechna získaná data jsou zabalena do slovníku, která jsou přístupná pod jednotlivými klíči. V aplikaci jsou stěžejní hodnota úhlové rychlosti v ose z.

6.2.4 pid.py

Aby při přesunu vozítka mezi jednotlivými QR kódy nedocházelo k nechtěné změně směru způsobené nedokonalostí použitých omni kol na vozítku, je aplikace vybavena MEMS gyroskopem. Tento gyroskop snímá úhlovou rychlost vozítka, čímž ze získané odchylky lze ovládat a měnit rychlost otáčení motorů a tím zajistit přímý směr pohybu. Aby data z gyroskopu mohla být použita a vhodně ovlivňovala změnu rychlosti otáčení motorů, je v aplikaci použit P regulátor. Tento regulátor má tedy jako vstupní veličinu úhlovou rychlost z MEMS gyroskopu a dle nastavených parametrů mění rychlost (frekvenci PWM) otáčení motorů.

Tento skript nese název pid, jelikož je implementován tak, aby mohl v případě potřeby být použit PID regulátor. Skript je složen ze dvou hlavních metod init() a control(). Metoda init() je volána při vytváření instance třídy PID. Mezi její parametry patří koeficienty PID regulátoru (r_0, r_i, r_d), čas vzorkování, požadovaná hodnota, minimální a maximální hodnota výstupu. V metodě control() je implementován samotný výpočet výstupní hodnoty dle vstupní hodnoty. Nejprve je zde vypočítávána chyba jako rozdíl mezi požadovanou a naměřenou hodnotou. Poté dochází k výpočtu jednotlivých složek pro získání celkového výstupu z regulátoru neboli akčního zásahu.

Výpočet byl proveden dle následujícího vzorce

$$x(t) = r_0 \cdot e(t) + r_i \int_0^t e(t)dt + r_d \cdot \frac{de(t)}{dt}, \tag{6.1}$$

kde $x(t)$ je akční zásah regulátoru, r_0 ve vztahu reprezentuje proporcionální konstantu (zesílení) regulátoru, r_i je integrační konstanta regulátoru, r_d je derivační konstanta regulátoru a e značí regulační odchylku

Samotný výstup z regulátoru, akční zásah, je omezen dle nastavení minimální a maximální hodnoty.

6.2.5 TCA9548A.py

Jedná se o skript sloužící pro inicializaci a nastavení multiplexoru TCA9548A, který je v aplikaci použit pro snímání dat z jednotlivých snímačů vzdálenosti. Mezi jeho hlavní

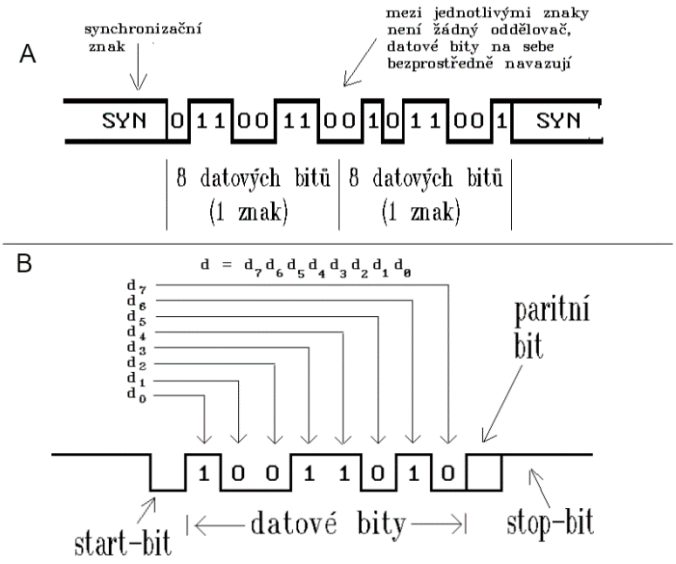
části patří konstruktor init, kde probíhá inicializace a vlastnost Channel. V konstruktoru se nastavuje I2C sběrnice pro komunikaci a také se nastavuje výchozí stav kanálu na vypnuté. Dále je v inicializaci nastavuje volba kanálu na -1 což znamená, že není vybrán žádný kanál.

Dále je zde vytvořena vlastnost pro Channel, aby bylo možno kontrolovat, co se s daným atributem děje. Touto vlastností je umožněno nastavovat jednotlivé kanály multiplexoru. Je zde ošetřeno překročení rozsahu vybraných kanálů (0-7). Pokud je vybrán kanál v tomto rozsahu provede se zápis do registru. A po zápisu se aktualizuje změna na nový kanál.

6.2.6 UpLinkUART.py

Tento skript slouží jako komunikační protokol pro odesílání dat mezi Raspberry Pi Pico a Raspberry Pi 4 přes rozhraní UART. Při návrhu tohoto protokolu bylo zapotřebí vybrat vhodnou sériovou komunikaci. Pro přenos dat přes sériovou komunikaci lze využít synchronního nebo asynchronního přenosu dat. Základní popis těchto přenosů je následující:

- **Synchronní přenos** – při tomto přenosu dat jsou přenášeny celé datové bloky najednou a následují po sobě. Tyto datové bloky nejsou vybaveny žádnými start či stop bity. Pro stanovení začátku bloku je použit tzv. SYN (synchronizační) znak, který je zde pro zajištění časové synchronizace odesílatele s příjemcem. Při použití tohoto znaku poté příjemci stanoví přesné datové okamžiky, ve kterých číst jednotlivé datové bity. Synchronní přenos je rychlejší než asynchronní, nicméně je náročnější na realizaci z důvodu nutnosti synchronizace odesílatele a příjemce. Tento přenos dat je patrný z obrázku 6.3 Synchronní a asynchronní přenos dat6.3 A.
- **Asynchronní přenos** – při použití asynchronního přenosu nemusí být data nikterak synchronizována a mohou být odesílána náhodně. To znamená že v tomto případě data můžou být přenášena s libovolnými časovými rozestupy. Jelikož by ale příjemce nemohl přesně stanovit kdy má začít vyhodnocovat přijatá data, jsou tyto datové bloky vybaveny pomocnými start a stop bity. Tyto bity slouží k identifikace začátku a konce datového bloku a jejich délka odpovídá jednomu či dvou datovým bitům. Dále může být tento přenos vybaven tzv. Paritním bitem, který zajišťuje rozpoznání chyb. Jedná se o nejjednodušší metodu k hledání chyb při přenosu dat, která doplní přenášený blok dat o jeden bit s hodnotou 0 nebo 1. Dodatečný bit má takovou logickou úroveň, aby se mohlo v datovém bloku nacházel bud´ sudý počet 1 („sudá parita“), nebo lichý počet 1 („lichá parita“). Příjemce tedy musí vědět jakou paritu přenášená data využívají, aby mohl následně vyhodnotit chybu. Příjemce však může vyhodnotit chybu jen při lichém počtu chybných bitů datového bloku. Tento přenos dat je patrný z obrázku 6.3 B.



Obrázek 6.3 Synchronní a asynchronní přenos dat

Pro danou aplikaci a přenos dat byla zvolena metoda asynchronního přenosu dat. Důvodem tohoto výběru bylo, že asynchronní komunikace umožňuje procesoru, aby prováděl jiné úkony během čekání na odezvu z druhého zařízení.

Mezi hlavní části tohoto skriptu patří konstruktor init. V initu probíhá inicializace sériové linky UART, kde se předává nastavení portů rx a tx pro komunikaci, které je vytvořeno ve skriptu ControlBoard.py. Pro zajištění správného přenosu dat je zapotřebí, aby proti komunikující strana měla stejné nastavení této přenosové sériové linky (baudrate = 115200, parity = None, stopbits = 1, bytesize = 8). Pro asynchronní komunikace zde probíhá inicializace StreamReader a StreamWriter. Tyto dva objekty umožňují přijímat a odesílat data přes sériovou linku asynchronně.

Další částí skriptu je asynchronní metoda CommunicatorLoopAsync() soužící pro asynchronní příjem dat. Metoda obsahuje jednu nekonečnou smyčku, která čeká na příjem dat. Přijatá data jsou přečtena pomocí metody readline(), která je součástí objektu StreamReader. Dále je pro tuto metodu vytvořen callback, aby přijatá data mohla být využívána a zpracovávána v jiné části projektu. Další metodou je send(), která umožňuje odesílat data přes metodu write(), která je součástí objektu StreamWriter. Vytvořená zpráva, která má být odeslána je předávána do metody Command(). V metodě Command() je pak volána metoda send() pro odeslání zprávy.

6.2.7 Vl5310x.py

Tento kód byl převzat z GitHubu a slouží ke konfiguraci senzorů vzdálenosti pro sběr dat [44]. V tomto skriptu je zahrnuto několik metod, avšak zde jsou popsány pouze hlavní metody. První hlavní metodou tohoto skriptu je Init(), zde probíhá kompletní nastavování registrů, inicializace pro nastavení I2C komunikace. Dále zde probíhá kalibrace senzorů a nastavování rychlosti měření. Metoda start() inicializuje senzory pro spuštění měření.

Dále se zde nastavuje časový interval, jak často má senzor provádět měření. Pokud je tento interval nastaven na nulu, senzor provádí měření v pravidelných intervalech, což umožňuje pravidelné získávání dat o vzdálenosti bez nutnosti manuálního spouštění každého měření. Nakonec je příznakem started senzor označen jako spuštěný. Další metoda je stop(), která provádí zastavení měření a nastavuje příznak started na false pro označení, že senzor již není spuštěný. Další důležitou metodou tohoto skriptu je metoda read(), která slouží pro čtení dat ze senzoru. Nejprve probíhá kontrola, zda je senzor spuštěný. Pokud není, provede se konfigurace senzoru a spustí se měření. Po dokončení měření se získaná hodnota uloží. Poslední hlavní metodou je ping(), kde se volají jednotlivé metody start(), read() a stop(). Tato metoda slouží pro jednoduchou obsluhu senzoru v podobě získávání aktuální naměřené vzdálenosti ze senzoru.

6.2.8 ControlBoard.py

V tomto skriptu je obsažena veškerá logika celého projektu. Jsou zde importovány a řízeny všechny podružné .py soubory. Tento skript se skládá z několik hlavních metod.

První metodou je btn_IRQHandler, která je určena pro obsluhu přerušení (IRQ) vyvolaného stisknutím tlačítka. Toto tlačítko slouží pro zapnutí a vypnutí napájení celého robotického vozítka. Poté co proběhne kontrola, zda přerušení vyvolalo tlačítko uloží se čas stisku tlačítka. Pokud je tlačítko uvolněno a bylo již dříve stisknuto, spočítá se rozdíl času mezi nynějším časem a časem posledního stisku tlačítka. Pokud je rozdíl času mezi stisky v definovaném rozmezí pro dlouhý stisk, zapne se napájení zařízení. Další částí je detekování krátkého stisku, která probíhá v asynchronní metodě Periodic(). Pokud se jedná o krátký stisk vypne se napájení zařízení.

Druhá metoda tohoto skriptu je UpLinkDataReceive(), Tato metoda zpracovává přijatá data z přenosu Uplink přes callback. Dochází zde k parsování přijaté zprávy a k rozdělení podle typu zprávy. První typ zprávy definuje pohybové příkazy a druhý typ zprávy obsahuje data týkající se úhlu natočení a odchylek zařízení od QR kódu.

Další metodou je init(), kde probíhá veškerá inicializace. Nejdříve jsou zde nastavovány jednotlivé piny na Raspberry Pi Pico podle elektrického schématu projektu. Dále je inicializována I2C0 sběrnice, která slouží pro komunikaci mezi senzory vzdálenosti a multiplexorem. Tedy pokud je na sběrnici I2C0 nalezeno zařízení s adresou 0x70, což odpovídá adrese multiplexoru, jsou v tomto zařízení postupně procházeny a přepínány 4 kanály. Na těchto kanálech se opět skenují a hledají zařízení odpovídající adrese 0x29 (senzory vzdálenosti), kterým je poté přiřazena instance třídy VL53L0X. Tato třída je popsána níže. Další inicializovanou sběrnicí je I2C1 sběrnice. Po této sběrnici probíhá komunikace s gyroskopem a se snímačem napětí a proudu. Opět se nejprve skenuje sběrnice a hledá se na ní zařízení s adresou 0x68. Pokud je zařízení s touto adresou nalezeno, provede se jeho inicializace. Tato adresa patří gyroskopu MPU6050. Dále se na této sběrnici hledá zařízení s adresou 0x40, které odpovídá zařízení pro snímaní napětí a proudu INA3221. Pokud je zařízení nalezeno opět dojde k jeho

inicializaci. Dále zde probíhá inicializace komunikace přes UART, která je použita ke komunikace mezi Raspberry Pi 4 a Raspberry Pi Pico. V poslední části této metody se nachází inicializace výstupního pinu pro napájení, řízení LED indikující stavy vozítka a také přisvětlovacích LED, které jsou řízeny PWM signálem. Dále je zde inicializace DCDC měniče a inicializace pinu tlačítka. Dále se zde inicializují jednotlivá signály pro řízení motorů. Poslední inicializací je PID regulátor, kde se nastavuje integrační a derivační konstanta na nulu a definuje se tak, že se jedná pouze o P regulátor.

Poslední metodou je asynchronní metoda Periodic(), kde probíhá periodické vyčítání jednotlivých dat ze všech snímačů (snímače vzdálenosti, snímač napětí a proudů a gyroskop). Při vyčítání dat ze senzoru vzdálenosti je použita pomocná proměnná SenID, která umožňuje měřit data postupně. Jak funguje vyčítání dat je popsáno níže v spodkapitole 6.2.7. Z každého senzoru jsou data ukládána do příslušné proměnné (_DistLeft, DistFront, …) v závislosti na umístění senzoru v robotickém vozítku. Vyčítání dat z gyroskopu je umožněno přes metodu get.mpu6050_data(), kde data jsou ukládána do slovníku Poté jsou jednotlivé hodnoty úhlové rychlosti získány ze slovníku a uloženy do odpovídajících instančních proměnných. K jednotlivým hodnotám, které snímá snímač napětí a proudu se přistupuje přes metody bus_voltage() a current(), kde se natavuje kanál, na kterém se tyto hodnoty měří. Jednotlivá získaná data se dále skládají do zprávy přes extend, bytearray a struct.pack. Tato zpráva je složena jako bajtové pole, které je však pro snadnější zpracování převedeno přes binascii.hexlifly na řetězec, který obsahuje hexadecimální reprezentaci těchto dat. Zpráva je poté odesílána do Raspberry Pi 4 přes sériovou linku UART.

Řízení vozítka je zde realizováno dle příchozích příkazů od nadřazeného řídícího softwaru. Aby mohlo být vozítko uvedeno do pohybu je zde nejdříve kontrolováno, zda se v definované vzdálenosti nenachází překážka. Pokud se žádná překážka nenachází v blízkosti vozítka dojde k výpočtu PID regulátoru pro řízení rychlostí motorů. Vozítko dostává příkazy o pohybu vždy když nalezne QR kód. Pro zpracování příkazu je vozítko pozastaveno na načteném QR kódu po dobu 1 s. Dále je směr pohybu na vozítkách indikován LED. Pokud vozítko stojí svítí všechny LED zelenou barvou. Když je vozítko v pohybu, jeho aktuální směr pohybu je signalizován pomocí příslušných blikajících bílých LED.

6.2.9 main.py

Jedná se o hlavní část programu, kde dochází ke spouštění celé aplikace. Main.py obsahuje dvě hlavní funkce main() a shutdown(). Funkce main() je asynchronní funkce starající se o správu paměti pomocí garbage collectoru. Tento collector umožňuje uvolňování paměti, kterou proces nepoužívá. Slouží pro efektivní využívání paměti. Dále se zde kontroluje stav aplikace. Pokud program běží, tak se z ControlBoardu.py volá funkce Periodic(). Funkce shutdown() je také definována jako asynchronní a slouží k ukončení aplikace. Volá se zde metoda deinit(), která je definována

v ControlBoardu.py. Tato metoda slouží k vyčištění a ukončení zařízení připojených k mikrokontroleru. Následně je proveden garbage collector, který se zde také používá pro uvolnění paměti.

Hlavní blok kódu Main.py nejprve testuje, zda je tento skript spouštěn přímo nebo z jiného skriptu. Pokud je tento skript spouštěn přímo, volají se funkce main() a shutdown(). Dále tento blok zajišťuje správné ukončení programu, pokud dojde k jeho přerušení nebo k chybě. Požadavky na přerušení jsou zde zpracovány pomocí výjimek.

6.3 Software v Raspberry Pi 4

Software vytvořený v mikrokontroleru Raspberry Pi 4 obsahuje pouze jeden skript. Tento skript obsluhuje celý řídící mikrokontroler. Celý skript je složen ze tří hlavních částí komunikace se Raspberry Pi Pico, komunikace s PC (nadřazeným řídícím softwarem) a detekce QR kódu z kamery. Tyto části jsou vytvořeny jako tasky, aby bylo možné spouštět tyto části asynchronně. Pro zajištění dostupnosti nezbytných dat pro spuštění tasku byl v tomto skriptu zaveden globální slovník. Tento slovník slouží jako úložiště pro data, která jsou vytvořena až v průběhu běhu jiných tásků. Níže jsou popsány jednotlivé části rozdělené dle funkcionality.

V podkapitole 6.3.2 je popsána komunikace Raspberry Pi 4 s nadřazeným řídícím systémem (PC). Zprávy, které si jednotlivá zařízení mezi sebou posílají jsou ve formě paketů. Struktura těchto paketů byla realizována v souladu s předem definovanou strukturou komunikace specifikovanou nadřeným systémem. Struktura jednotlivých zpráv této komunikace byla převzata z jiné diplomové práce, pro zajištění kompatibility systému [43].

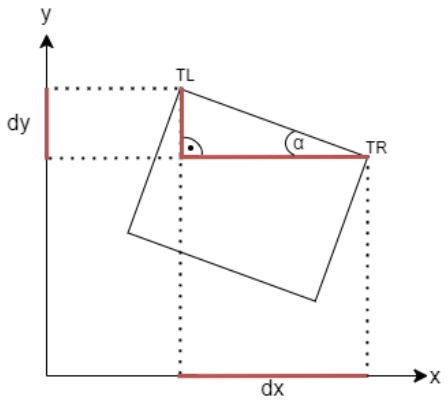
6.3.1 Detekce QR kódu

Nejdříve je provedena základní konfigurace kamery kde se nastavuje adresa video zařízení. Touto adresou je /dev/video0 a je typická pro první dostupnou kameru. Dále se zde nastavuje rozměr snímaného obrazu na 800 x 600 px. Poslední částí konfigurace je vytvoření instance pro třídu QRCodeDetector(), kterou obsahuje knihovna OpenCV. Tato třída obsahuje funkce, které umožňují nalezení QR kódu v obraze a jeho následné dekódování.

Když je kamera nakonfigurovaná následuje čtení dat z kamery pomocí funkce cap.read(), která vrací dva parametry. Prvním parametrem je boolean hodnota, která značí, zda byl úspěšně načten snímek a druhým parametrem je samotný snímek z kamery. Poté je uložena šířka a výška snímku pro následný vypočet středu snímku, který je popsán níže. Pro detekci a dekódování je použita funkce detectAndDecode(), která ve snímku detekuje a dekóduje nalezený QR kód. Vyčítají se zde dva parametry. Do prvního parametru data je uložen dekódovaný obsah QR kódu. Do druhého parametru bbox jsou uloženy souřadnice obdélníku, který ohraničuje nalezený QR kód. Pokud jsou uložena nějaká data, tak se provádí vykreslení obdélníku kolem nalezeného QR kódu podle

souřadnic uložených v parametru bbox. Při vykreslování obdélníku jsou použity souřadnice jednotlivých rohů QR kódu a poté jsou mezi těmito rohy nakresleny úsečky pomocí cv2.line.

Dále bylo zapotřebí zjistit jaký je úhel mezi QR kódem a středem vozítka, kde se nachází střed kamery. Tato informace je dále použita pro vycentrování vozítka. Pro výpočet úhlu natočení bylo nejprve nutné vypočítat vzdálenosti rohů pro možné použít trigonometrické funkce. Jednotlivé vzdálenosti byly získaný jako rozdíl jednotlivých souřadnic dvou rohů (TL a TR). Požadovaný úhel lze poté vypočítat použitím arkus tangens funkce. Příklad je znázorněn na obrázku 6.4. Jelikož funkce math.atan použitá ve skriptu vrací úhel v radiánech, bylo zapotřebí úhel převést na stupně a poté upravit výsledek tak, aby vracel hodnoty v rozsahu $\pm$ 180 °.



Obrázek 6.4 Znázornění pro výpočet úhlu natočení QR kódu

Další částí skriptu je výpočet středu QR kódu. Tato část je vypočtena opět za použití souřadnic jednotlivých rohů obdélníku. Kdy jsou vypočteny vždy středy dvou sousedících stran obdélníku. Pro první stranu se určují souřadnice středu (Cent1X, Cent1Y). Pokud souřadnice x-ového bodu pravého dolního rohu je menší než souřadnice x-ového bodu levého dolního rohu, pak je střed vypočten jako polovina rozdílu x-ových souřadnic obou bodů. Pokud je souřadnice x-ového bodu levého dolního rohu menší, pak je střed vypočten analogicky, ale s prohozenými body. Pro druhou stranu se postupuje obdobně jako pro první stranu. Poté se nekreslí dvě kružnice vždy do středu strany o poloměru půlky strany. Tam kde se kružnice protnou je střed QR kódu.

Další střed, který je dále počítán je střed snímku. Zde jsou použité pomocné proměnné, do kterých byly uloženy rozměry obrázku. Střed je na snímku vykreslen pomocí křížku. Tato informace je nezbytná pro výpočet vzdálenosti středu kamery od středu QR kódu. Vzdálenost společně s úhlem natočení slouží k tomu, aby mohlo být vozítko navedeno tak, aby se střed vozítka nacházel na středu QR kódu.

V poslední části skriptu se vypočítává procentuální úspěšnost detekce QR kódů a je zobrazována na snímku společně s úhlem natočení. Celkový snímek je zobrazen v okně s názvem "code detector" a stisknutím klávesy "q" lze ukončit program.

6.3.2 Komunikace s nadřazeným řídícím systémem

Komunikace mezi Raspberry Pi 4 a nadřazeným řídícím softwarem (PC) probíhá přes bezdrátové připojení Wi-Fi. Pro komunikaci se serverem (PC) je použit UDP protokol, kde Raspberry Pi 4 je v pozici klienta a obě zařízení jsou připojena do stejné sítě. Pro realizaci komunikace bylo nezbytné nastavit na Raspberry Pi 4 statickou IP adresu. Před začátkem komunikace se serverem bylo zapotřebí nastavit a otevřít pouze komunikační port (PORT_NUMBER = 5000). Při znalosti tohoto portu posléze server může zahájit komunikaci s klientem. Příjem dat není vázán na konkrétní IP adresu serveru (nadřazeného systému), ale čte data ze všech zařízení, která klienta kontaktovala. Poté klient odpovídá a posílá zprávy na IP adresu zařízení, ze kterého byl kontaktován.

V nekonečné smyčce probíhá čekání na příchozí zprávy z portu. Pomocí metody recvfrom lze zjisit, kdo byl odesílatelem zprávy. Data obsažená ve zprávě jsou uložena do proměnné data a IP adresa odesílatele do proměnná addr. Ze zprávy jsou extrahována potřebné informace dle předdefinované struktury. První dva bajty zprávy obsahují ID vozítka. Další přijaté bajty zprávy se týkají pohybových příkazů a jsou datového typu bool. Pokud je některý z příkazů true, je tento příkaz poslán do Raspberry Pi Pico přes UART, aby mohl být vykonán. Poslední dva bajty jsou predikce QR kódu. Tyto dva poslední bajty však nejsou v dané aplikace využívány a jejich hodnota je nula.

Raspberry Pi 4 přijímá data také od Raspberry Pi Pico (způsob a princip přijímání dat je popsán v podkapitole 6.3.3), která jsou následně dekódována a kompilována do struktury vhodné k odeslání do nadřazeného řídícího softwaru. Tato kompilace zpráv je prováděna pomocí metody bytearray.extend(), která umožnuje postupné přidávání dat do vytvořeného bytearray. V této aplikaci jsou vytvořeny dva různé typy paketů (0x01 a 0x02) dle typového čísla obsahující zprávu. Oba pakety obsahují číslo vozítka, od kterého jsou data odesílána. První paket obsahuje statusové zprávy týkající se vozítka a jsou odesílána cyklicky při obdržení žádosti od nadřazeného systému. Druhý typ zprávy obsahuje informace týkající se aktuálně načteného QR kódu a je odesílán jen při detekci QR kódu. Tyto typy zpráv jsou odesílány pomocí příkazu sock.sendto().

6.3.3 Komunikace s Raspberry Pi Pico

V této části skriptu dochází nejdříve k inicializaci sériového portu pro komunikaci s Raspberry Pi Pico. Pro komunikaci je použit sériový port ttyAMA0. Nejdříve se ověřuje, zda je sériový port již otevřen. Pokud ano, dojde k jeho uzavření a následnému otevření. Po opětovném ověření, zda je port otevřen se z toho portu začnou číst data. V nekonečné smyčce se čeká na příchozí data z portu. Když data přijdou, jsou přečtena pomocí metody readline(). Přijatá data jsou přidána do řetězce rxMessage a následně jsou

dekódována. Přijatá zpráva je přijata v podobě pole stringu, proto je pro správné rozparsování zprávy nejprve nutné si jednotlivá data zprávy převést na pole bajtů. Data jsou uložena do klíčových slov pro následné zpracování a možnou volbu použití těchto dat v celém projektu.

Kromě přijímání dat jsou zde také vytvořeny dva typy zpráv pro odesílání do Raspberry Pi Pico. První typ zprávy 0x01 jsou data obsahující příkazy od nadřazeného systému. Tato data jsou po obdržení Rasperry Pi 4 pouze předána do Raspberry Pi Pico. Druhý typ zprávy 0x02 obsahuje informace o aktuálně načteném QR kódu v podobě odchylky od středu QR kódu a úhel natočení. Oba typy zpráv s jejich obsahem jsou nezbytné pro řízení vozítka a jsou zpracovávány v Raspberry Pi Pico. Pro tvorbu zprávy je použit opět bytearray(), který z těchto jednotlivých dat složí zprávu a před odesláním je na celou zprávu použita funkce binascii.hexlify(). Tato funkce konvertuje binární data do jejich reprezentace v hexadecimálním formátu vhodná pro odeslání. Pro odeslání dat je použita funkce ser.write(), která data odešle. Dále bylo zapotřebí použít funkci ser.flush(), která po odeslání zprávy vyčistí a vyprázdní buffer sériového portu.

7. TESTOVÁNÍ

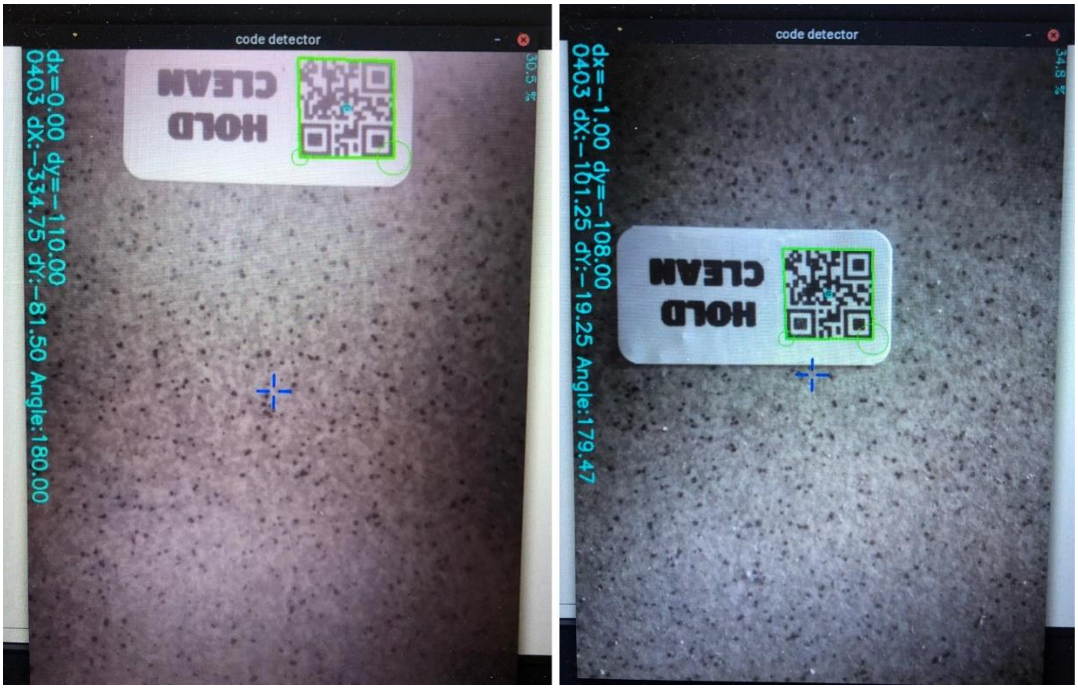
V následující kapitole jsou popsány výsledky testování realizovaného výrobku společně s návrhem dodatečných úprav. Podkapitola 7.1 pojednává o testováních konkrétních částí autonomního robotického vozítka se zaměřením na pohyb, komunikaci, čtení dat z kamery a testování hardwaru robotického vozítka. Druhá podkapitola 7.2 je zaměřena na popis možných dodatečných úprav výsledného vzorku s odůvodněním.

7.1 Testování realizace

7.1.1 Čtení dat z kamery

Při četní dat z kamery bylo zjištěno, že kamera je schopna detekovat objekty v celém svém zorném poli. Jelikož je pro přisvícení oblasti pod kamerou použit LED pásek se stínítkem, je v oblasti přímo pod tímto LED páskem procentuálním úspěšnost snímání nižší než mimo něj. Toto je způsobeno zaostřením kamery na méně nasvícené plochy čímž je oblast pod LED páskem přesvícená. Při použití nižšího jasu LED se kamera také zaostřuje na tmavší části snímku. I přes tento nedostatek je aktivní celá snímatelná oblast pod kamerou a kamera je schopna číst data z QR kódu na kterémkoliv místě. Tento procentuální rozdíl však není velký a pro danou aplikaci je tato varianta zcela použitelná a plně dostačuje požadavkům aplikace. Varianty snímatelných oblastí s procentuální úspěšností snímání jsou na obrázku 7.1.

Kromě detekce dat uložených v QR kódu je kamera také schopna snímat a vyhodnocovat odchylku středu vozítka (kamery) od středu QR kódu společně s úhlem natočení. Pří výpočtu odchylky natočení QR kódu se nepředpokládá chyba, jelikož jsou při výpočtu využívány rohy načatého QR kódů. Na základě tří jeho rohů je poté vždy stejným způsobem vypočítávána odchylka natočení. Tudíž když bude detekován QR kód bude vždy určen úhel natočení. U výpočtu odchylek v ose x a y je způsob obdobný s tím rozdílem, že při tomto výpočtu se využívá středu QR kódu. Všechna načtená a vypočtená data jsou poté odesílána ke zpracování.



Obrázek 7.1 Čtení QR kódů kamerou

7.1.2 Komunikace a přenos dat

V dané aplikaci je komunikace pro přenos rozdělena dvou samostatných částí. První část se zabývá přenosem dat mezi Raspberry Pi 4 a nadřazeným řídícím software. Druhá část je zaměřena na výměnu dat mezi Raspberry Pi 4 a Raspberry Pi Pico.

Při testování bylo zjištěno, že pokud Raspberry Pi Pico odešle data s aktuálními informacemi o vozítku (stav baterie, naměřené vzdálenosti od senzorů, …) jsou tyto data pomocí Raspberry Pi 4 přeposlána nadřazenému řídícímu systému. Naopak vždy když Raspberry Pi 4 přijme data o povelech k pohybu jsou tato data přeposlána do Raspberry Pi Pico pro následné zpracování. Aby však bylo možné obdržet příkaz k pohybu od nadřazeného řídícího systému je nejprve nutné odeslat data o aktuální poloze. Tedy pokud dojde k detekci QR kódu kamerou jsou data zpracována a odeslána jak nadřazenému řídícímu systému, tak i Raspberry Pi Pico, aby mohla být zahájen pohyb potřebný pro nalezení optimální polohy na QR kódu.

Pro komunikaci po sériové lince byla nastavena rychlost přenosu na 115200 bit/s, kde tato rychlost byla shledána jako nejoptimálnější. Aby však nedocházelo v Rasbperry Pi 4 k přehlcení bufferu s daty, byl celý kód ošetřen podmínkami určující čas komunikace. Tedy pokud zařízení neodešle či nepřijme ve stanovený čas, je obsluhována jiná událost. Konečná výměna dat mezi jednotlivými zařízeními probíhala dle očekávání a dle požadavků zadavatele.

7.1.3 Pohyb vozítka

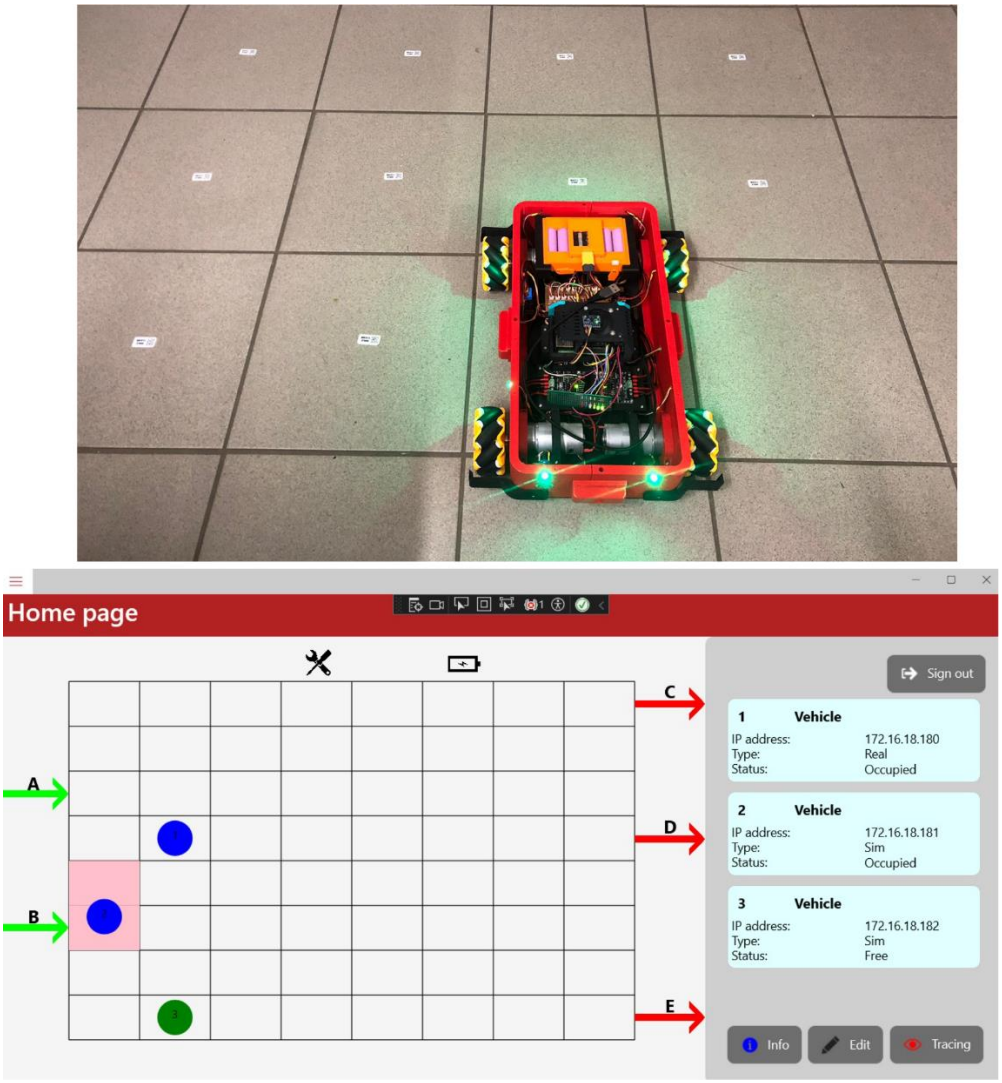
Aby při pohybu vozítka byla dodržena bezpečnost provozu je celý pohyb ošetřen podmínkou, že v okolí vozítka se nesmí nacházet žádné překážky v určité vzdálenosti. Pokud nějaký ze čtyř senzorů detekuje překážku ve vzdálenosti 100 mm bude pohyb automaticky pozastaven či nebude vůbec spuštěn.

Pohyb vozítka lze rozdělit na dvě hlavní částí. První částí je pohyb mezi jednotlivými QR kódy. Pohyb mezi těmito kódy je řízen pomocí příkazů od nadřazeného řídícího softwaru, kdy tento software dle předchozí získané pozice určuje směr pohybu. Pokud se vozítko pohybuje mezi jednotlivými QR kódy nadřazený systém ztrácí informaci o jeho poloze. Aby bylo možné při tomto pohybu udržet co nejpřímější směr je k tomuto účelu využíván MEMS gyroskop. Tento gyroskop snímá aktuální hodnotu úhlové rychlosti ve směru pohybu (v ose z). Hodnoty z gyroskopu jsou použity jako vstup do regulátoru, který reguluje akčním zásah rychlost otáčení kol, čímž vyrovnává úhlovou rychlost na 0. Jelikož při pohybu vozítko nejeví velké výchylky od přímého směru a jelikož jsou QR kódy umístěny tak, že vozítka urazí pouze malou vzdálenost, je v úloze použit pouze P regulátor. P regulátor tedy násobí odchylku úměrnou vychýlení a reguluje rychlost otáčení kol. Jelikož z naměřených dat není možné nalézt přesný matematický model vozítka, byla pro návrh regulátoru a nalezení příslušných parametrů použita metoda náhodného výběru. Kde při testování bylo zjištěno, že nejoptimálnějších výsledků při pohybu dosahuje nastavení parametru r0 na hodnotu 22. Při tomto nastavení je vozítko schopné detekovat všechny QR kódy na stanovené trase. Při akčním zásahu regulátoru, který ovlivňuje pohyb vozítka, dochází ke změnám v rychlosti kol. Pro korigování směru se snižuje rychlost motorů na potřebné straně vozítka. Pokud by naopak docházelo ke zvyšování rychlosti a vozítko již dosáhlo svého maximální rychlosti, nebylo by možné dále zvyšovat rychlost motorů a tím pádem korigovat směr pohybu vozítka.

Druhá část pohybu spočívá v nastavení vozítka do optimální polohy nad QR kódem pro následný přesun. V kódu je pouze realizována příprava pro tento typ pohybu, která by sloužila pro snížení pravděpodobnosti nenalezení následujícího QR kódu způsobeným nepřímým pohybem mezi QR kódy. Tato varianta pohybu nakonec nebyla realizována z důvodu vysoké latence způsobené snímáním a přenosem dat. Zadavateli byla navržena jedna z možných variant, avšak nakonec nebylo o žádném z řešení definitivně rozhodnuto. V tomto případě by se vozítko pohybovalo pomaleji než při pohybu přímým směrem. Pro srovnání se na QR kódu by bylo nastaveno rozmezí ± 10 px v odchylkách a $\pm 5°$ v úhlu natočení, ve kterém by se vozítko považovalo za srovnané. Vzhledem k existenci latence při přenosu dat by mohl být při metodě narovnávání využíván princip krokování pro minimalizaci rizika přejetí QR kódu. Tento princip by spočíval v tom, že při určité rychlosti pohybu vozítka a znalosti odchylky v pixelech lze určit, jak daleko se vozítko posune, než obdrží další zprávu. Na základě této informace by vozítko provedlo pouze určitý pohyb, odpovídající dané odchylce, aby se minimalizovalo riziko, že nastane přejetí QR kódu, a to tak, že počká na další zprávu před dalším pohybem. Tímto

způsobem by se mohlo dosáhnout vyváženého a přesného sledování QR kódu při kompenzaci za vzniklou latenci. Tímto způsobem by mohly být nalezeny všechny optimální pozice v x a y ose. Úhel natočení by byl prováděn obdobným způsobem, kde odchylka je v tomto případě dána ve stupních. společně s úhlem natočení.

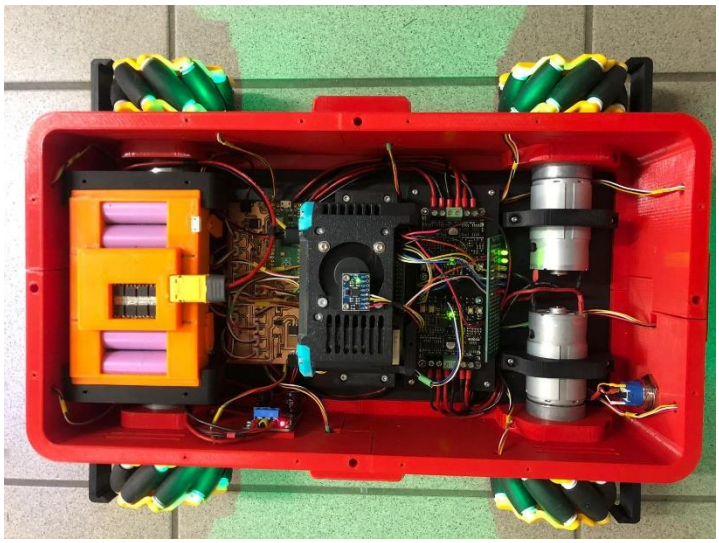
V aplikaci není zahrnuta varianta toho, že vozítko bude manuálně přemístěno a tím se změní směr jeho pohybu. Tedy vozítko poté pojede přímý směrem ve směru, do kterého bylo načteno, čímž ale nenalezne požadovaný QR kód. Dále se při spuštění vozítka uvažuje, že vozítko je do prostoru s QR kódy umístěno tak, aby byl schopen bez jakéhokoliv dalšího pohybu detekovat QR kód a odeslat data nadřazenému řídícímu softwaru.



Obrázek 7.2 Testování reálného vozítka s aplikací [43]

7.1.4 Hardware vozítka

Během celého testování bylo shledáno, že vozítko vybavené výše uvedenými elektrickými komponenty je schopno na jedno plné nabití baterie jezdit v průměru 140 min, při kapacitě baterie 7000 mAh a odběrovém proudu 3 A. Při testování a přerušovaném pohybu bylo vozítko schopné provozu až 6 h. Dále bylo zjištěno, že materiál konstrukce vozítka je dostatečně pevný a stabilní při převážení předmětů. Vozítko je ve výsledku schopné převážet předměty o velikosti 200 a 378 mm a nízké hmotnosti. Výsledná realizace robotického vozítka plně dostačuje pro danou aplikaci a je vhodné pro použití jakožto testovacího vzorku pro nadřazený řídící software. Konkrétní realizovaný prototyp je na obrázku 7.3.



Obrázek 7.3 Prototyp vozítka – elektrická část

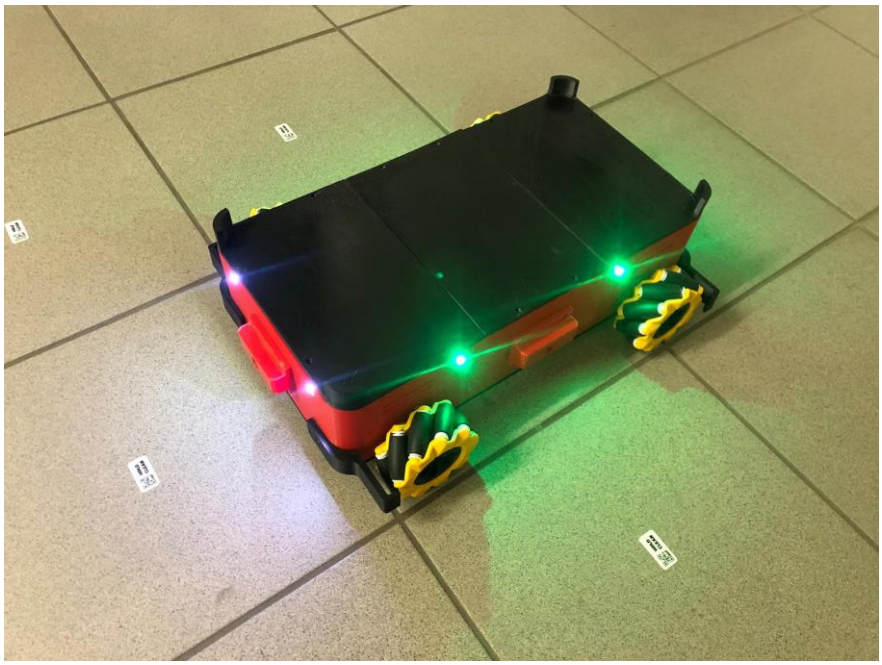
7.2 Možné úpravy výsledné realizace

Při testování robotického vozítka v provozu a při následném zhodnocení výrobku je nutné zohlednit, že vozítko bylo navrženo a zkonstruováno s ohledem na typ použití. Vozítko bylo navrženo za účelem testování nadřazeného řídícího softwaru s ohledem na požadavky zadavatele. Tedy vozítko slouží jakožto model, který není primárně určen do reálného průmyslového provozu, avšak může představovat jakýsi základ pro následnou realizaci. Pokud by zadavatel chtěl vytvořit vozítko vhodné pro nasazení do provozu a k výkonu konkrétní úlohy by mohlo být vozítko zkonstruováno a vybaveno následujícími komponenty.

Dle aplikace a použití by mohl být použit pevnější a odolnější materiál na realizaci konstrukce povozku a rámu. Dále by bylo nutné zohlednit výsledné rozměry s ohledem na převážený typ materiálu. Pro lepší odolnost a stabilitu vozítka by bylo možné použít jiný typ kol, jelikož při pohyb vozítka dochází k mírným otřesům celé konstrukce a tím i otřesům převáženého materiálu. Dále by pro prodloužení doby provozu mohlo být

vozítko vybaveno jiným typem baterii, popřípadě druhou baterií. Tato baterie by mohla být umístěna například v přední části vozítka, kde se nachází volný prostor. Také by vozítko mohlo dle požadavků zákazníka být rozšířeno o větší bezpečnost. Senzory vzdálenosti by byly nahrazeny jiným typem snímačů. Například pro snížení rizika by mohl být v aplikaci použit LIDAR (Light Detection And Ranging). Pomocí něhož by bylo možné detekovat překážky po celém obvodu pohybujícího se vozítka. Pro snímání dat by bylo možné namísto výše uvedené kamery použít průmyslovou kameru, která by umožnovala stabilnější, rychlejší a přesnější snímání QR kódů.

Jelikož byla úloha navržena pouze pro testovací účely a s ohledem na požadavky zadavatele je výsledná realizace plně dostačující pro daný typ úlohy a použití.



Obrázek 7.4 Výsledná realizace vozítka

8. ZÁVĚR

Cílem této diplomové práce bylo navrhnout konstrukci robotického vozítka určeného převážně pro přepravu materiálu. Dáne vozítko slouží pouze jako prototyp pro testování nadřazeného řídícího software. Základem práce bylo seznámit se s problematikou a použitím robotických vozítek. Dále bylo úkolem navrhnout jednu konkrétní konstrukci a vybrat vhodné elektrické komponenty nezbytné pro řízení a zajištění bezpečnosti během provozu. Dalším bodem zadání bylo navrhnout a vhodně realizovat elektrické zapojení těchto komponentů. A nakonec vytvořit nezbytný řídící a komunikační software.

Prvním krokem pro návrh celé koncepce robotického vozítka bylo seznámení se s konkrétními typy robotických vozítek s ohledem na jejich princip funkce. Pro danou úlohu dle požadovaných vlastností od zadavatele byl pro pohyb vozítka vybrán princip snímání kamerou. V úloze jsou snímány QR kódy, které jsou připevněny na podlaze a dle jejich rozmístění se následně vozítko pohybuje. Důvodem tohoto výběru byl jednoduchý princip snímání a také následné zpracování dat z těchto kódů. Princip pohybu vozítka dle QR kódů a jeho nadřazený řídící software nebyl součástí této práce. Dále byla dle stanoveného principu funkce navržena konkrétní konstrukce vozítka. Pro návrh vozítka a jeho následnou realizaci bylo vybráno vývojové prostředí DesignSpark Mechanical, kde byla celá konstrukce namodelována a poté vytištěna na 3D tiskárně. Posléze byly vybrány konkrétní typy elektrických součástek, nezbytné pro pohyb a bezpečnost. Z důvodu nároků a použití výsledného produktu byly po dohodě se zadavatelem vybrány komponenty převážně určené pro modelářskou techniku. Pro řízení a zpracování nasnímaných dat byly jako řídící mikrokontrolery vybrány součástky Rasbperry Pi Pico a Raspberry Pi 4. Pro snímání QR kódů byla vybrána kamera Raspberry Pi NoIR kamera V2, která je s mikrokontrolery kompatibilní. Z důvodu nenáročnosti aplikace byly pro pohon vybrány stejnosměrné motory a následně osazeny omni koly, které díky své konstrukci a vhodnému rozmístění na vozítku umožňují všesměrový pohyb.

Při realizaci elektrického zapojení bylo nutné dovybavit vozítko dalšími elektrickými komponenty, z důvodu nízkého počtu portů na řídících mikrokontrolerech. Posléze bylo vytvořeno elektrické schéma zapojení ve vývojové prostředí KiCad 7.0, ve kterém byl také navrhnut plošný spoj. Při kompletaci a montáži bylo shledáno několik nedostatků, které jsou i s jejich výsledný řešením popsány výše v práci.

Po kompletaci a propojení komponentů byl vytvořen řídící software pro pohyb vozítka a snímání dat kamerou. Dále byl realizován komunikační software pro možné přijímání a odesílání dat nadřazenému řídícímu softwaru. Všechny nezbytné části práce byly posléze otestovány a zhodnoceny dosažené výsledky s možným rozšířením aplikace. Dle zadavatele byla ve výsledku sestavena reálná konstrukce robotického vozítka odpovídající požadavkům a účelům použití firmy B:tech.

Literatura

[1] QR kód slaví 25 let. Jak funguje a k čemu všemu je dobrý? Inteligentnisvet [online]. 2019 [cit. 2023-12-11]. Dostupné z: https://inteligentnisvet.cz/clanky/qr-kod-slavi-25-let-jak-funguje-a-k-cemu-vsemu-je-dobry?fbclid=IwAR1E3tOaMsiDm0LxPeotNDE2n90He4rszqd4OGuYv_6rKXCuWN_FDrQI1ns

[2] QR kód slaví 25 let. Víte, jak vám tečkovaný čtverec může usnadnit život? Mobilmania [online]. 2019 [cit. 2023-12-11]. Dostupné z: https://mobilmania.zive.cz/clanky/qr-kod-slavi-25-let-vite-jak-vam-teckovany-ctverec-muze-usnadnit-zivot/sc-3-a-1346523/default.aspx

[3] QR kód. In: Wikipedie [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://cs.wikipedia.org/wiki/QR_k%C3%B3d#/media/Soubor:QR_Code_Structure_Example_3.svg

[4] Structure of The QR Code: How Is The Data Coded? Digital business card [online]. [cit. 2023-12-11]. Dostupné z: https://myqrbc.com/structure-of-the-qr-code-how-is-the-data-coded/?fbclid=IwAR2oRLcZLTRrxH5F7LD1XQtzRlB_oclRQVEnpQAO2fX_TXgT0FwwSOFeMwE

[5] QR Code Structure. Scanova Blog [online]. 2015 [cit. 2023-12-11]. Dostupné z: https://scanova.io/blog/how-qr-codes-work/?fbclid=IwAR1sGbwc9s3AToAeZu8ebGIot1pWJawwj6LWK_d2lAITFLkUOhrLF3rBBew

[6] QR kód. WikiKnihovna [online]. 2012 [cit. 2023-12-11]. Dostupné z: https://wiki.knihovna.cz/index.php/QR_k%C3%B3d?fbclid=IwAR2F6tnBWK_m-wVd0RQxlaA6GZimYFgbS3__-DjvRTxraWBy6PZgYXW-LNY

[7] DESIGNSPARK MECHANICAL: PROGRAM PRO NÁVRH VE 3D ZDARMA. DPS [online]. 2013 [cit. 2023-12-11]. Dostupné z: https://www.dps-az.cz/clanky/id:4592/designspark-mechanical-program-pro-navrh-ve-3d-zdarma

[8] The DesignSpark Mechanical Interface. DS Mechanical [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://help.spaceclaim.com/dsm/6.0/en/Discovery/user_manual/the_spaceclaim_interface.html

[9] DesignSpark Mechanical. Konstrukter [online]. 2014 [cit. 2023-12-11]. Dostupné z: https://www.konstrukter.cz/recenzujeme-designspark-mechanical-strojirensky-3d-cad-ktery-muzete-mit-zadarmo/

[10] Raspberry Pi Pico a Pico W. Raspberry Pi [online]. [cit. 2023-12-11]. Dostupné z: https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html

[11] Raspberry Pi Pico. RPishop [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://rpishop.cz/raspberry-pi-pico/5117-raspberry-pi-pico-0617588405587.html#description-anchor

[12] Seznamte Se S Deskou Raspberry Pi Pico Za 4 $ S Dvoujádrovým Mikrokontrolérem RP2040 Cortex-M0+. Electronics-lab [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://www.electronics-lab.com/meet-the-4-raspberry-pi-pico-board-with-rp2040-dual-core-cortex-m0-microcontroller/?fbclid=IwAR1T8d-TkRyZ8IykyOeQB9E6ht55DAmWV7x7qYy3HPgH8Fxwfht0I8RHzeI#google_vignette

[13] RP2040 DATASHEET [online]. In: . s. 639 [cit. 2023-12-11]. Dostupné z: https://datasheets.raspberrypi.com/rp2040/rp2040-datasheet.pdf?_gl=1*278dl*_ga*OTc4NjA5NDI4LjE2OTY2ODQxMzc.*_ga_22FD70LWDS*MTY5NjY4NDEzNi4xLjEuMTY5NjY4NDE3MS4wLjAuMA&fbclid=IwAR2GX7_qZE6idK6MNNSfqBsXedG7JGI5xjE4Pw-XceDkp4plTwg26ygZNRw

[14] Instalace systému Raspbian PIXEL. RPIShop.cz [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://rpishop.cz/micro-sd/576-instalace-systemu-raspbian-pixel.html?fbclid=IwAR01fwhMlki0z-Ztl0L5KNjMzIW7l2jBlNZCEGcJ5Mhvxyat9jOmAzol4pY

[15] Raspberry Pi 4 Model B [online]. In: . 2023, s. 6 [cit. 2023-12-11]. Dostupné z: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-product-brief.pdf?fbclid=IwAR01MA0A-4gBedAu4iUcnGN1AmDBz4zDAuSCv5rwyOtV1xzGv1yRcD94duI

[16] Raspberry Pi 4 Model B - 4GB RAM. RPIShop.cz [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://rpishop.cz/raspberry-pi-4/1598-raspberry-pi-4-model-b-4gb-ram-765756931182.html#description-anchor

[17] Úvod do Pythonu. W3 schools [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://www.w3schools.com/python/python_intro.asp

[18] Co se používá python pro průvodce pro začátečníky k používání pythonu. Cousera [online]. 2023 [cit. 2023-10-15]. Dostupné z: https://www.coursera.org/articles/what-is-python-used-for-a-beginners-guide-to-using-python https://www.educative.io/answers/what- je-rozdíl-mezi-pythonem-a-mikropythonem

[19] Motor JGB37-550 12V s převodovkou. LaskaKit [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://www.laskakit.cz/motor-jgb37-550-12v-s-prevodovkou/?variantId=9213¤cy=eur&gad_source=1&gclid=CjwKCAiAg9urBhB_EiwAgw88mdI8WChv2Fjp6Hop6 j19ZsUb7hoCIhwQAvD_BwE

[20] 7A/160W Dual H-Bridge Motor Controller. In: Handson Technology [online], s. 9. [cit. 2023-12-11]. Dostupné z: https://www.laskakit.cz/user/related_files/7a-160w_motor_control.pdf

[21] Raspberry Pi NoIR kamera V2. RPIShop.cz [online]. [cit. 2023-12-11]. Dostupné z: https://rpishop.cz/mipi-kamerove-moduly/331-raspberry-pi-noir-kamera-modul-v2.html#productcomments-anchor

[22] Gyroskop a akcelerometr gy 521 MPU 6050. In: LaskaKit [online]. [cit. 2023-11-13]. Dostupné z: https://www.laskakit.cz/user/related_files/mpu-6050_datasheet_v3_4.pdf https://www.laskakit.cz/arduino-gyroskop-a-akcelerometr-gy-521--mpu6050/#relatedFiles
[23] VL5310x. In: ST life [online]. 2016, s. 40 [cit. 2023-12-11]. Dostupné z: https://www.laskakit.cz/user/related_files/vl53l0x.pdf
[24] Laserový senzor vzdálenosti GY-53 VL53L0X + STM32. LaskaKit [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://www.laskakit.cz/laserovy-senzor-vzdalenosti-gy-53-vl53l0x-stm32/
[25] TCA9548A Low-Voltage 8-Channel. In: LaskaKit [online]. 2019, s. 39 [cit. 2023-12-11]. Dostupné z: https://www.laskakit.cz/user/related_files/tca9548a.pdf
[26] TCA9548 1xI2C master - 8xI2C slave expander. LaskaKit [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://www.laskakit.cz/tca9548-1xi2c-master-8xi2c-slave-expander/#relatedFiles
[27] Senzor napětí a proudu s I2C sběrnicí. In: ECLIPSERA Modules [online]. 2018, s.3 [cit. 2023-12-11]. Dostupné z: https://dratek.cz/docs/produkty/0/708/1523338818.pdf
[28] 2C INA3221 senzor napětí a proudu - 3 kanály. Drátek [online]. [cit. 2023-12-11]. Dostupné z: https://dratek.cz/arduino/3080-i2c-senzor-napeti-a-proudu-3kanalovy-ina3221.html
[29] Active omni wheel capable of active motion in arbitrary direction and omnidirectional vehicle. In: Bulletin of the JSME [online]. 2016, s. 20 [cit. 2023-12-11]. Dostupné z: https://www.jstage.jst.go.jp/article/jamdsm/10/6/10_2016jamdsm0086/_pdf?fbclid=IwAR1mMh5dX5BhOLLb8cTWRPNYqtLGw6y2VEedCNo2b2n2gK6b2_06CMCHKqk
[30] Sada 4ks Mecanum Omni kol ø80mm. In: LaskaKit [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://www.laskakit.cz/sada-4ks-mecanum-omni-wheel-kol-60mm-s-motory/?gclid=CjwKCAjw-KipBhBtEiwAWjgwrEOzi3TbCDPKkd0Veft2a1QiFkjLPEb9vCAWlJ-0X4mH6MVpOoiQVxoCdCYQAvD_BwE#ratingTab
[31] Sada kol Mecanum - 80 mm - 4 kusy - černá / žlutá s válečky. Botland [online]. 2023 [cit. 2023-12-11]. Dostupné z: https://botland.cz/omni-wheels/19385-sada-kol-mecanum-80-mm-4-kusy-cerna-zluta-s-valecky-dfrobot-fit0654-1-5904422346522.html
[32] ASA. PRUSA RESEARCH by JOSEF PRUSA [online]. 2023 [cit. 2024-03-23]. Dostupné z: https://help.prusa3d.com/cs/article/asa_1809
[33] ASA. MATERIALPRO 3D [online]. 2024 [cit. 2024-03-23]. Dostupné z: https://www.materialpro3d.cz/materialovy-slovnik/asa/
[34] PCTG – co to je? Vlastnosti a škodlivost. Botland [online]. 2023 [cit. 2024-03-23]. Dostupné z: https://botland.cz/blog/pctg-co-to-je-vlastnosti-a-skodlivost/

[35] Filament SPECTRUM / PCTG CF10 / ČERNÁ / 1,75 mm / 1 kg. Smart3d [online]. 2024 [cit. 2024-03-23]. Dostupné z: https://www.smart3d.cz/p/filament-spectrum-pctg-cf10-cerna-1-75-mm-1-kg?gad_source=1&gclid=Cj0KCQjwqdqvBhCPARIsANrmZhPwE9RISXnjGZ-a3ZhPotODrE6fWUEWO-uU8PlTIM0qbIohyE_YsTQaAmMSEALw_wcB
[36] MDD3A 3Amp 4V-16V DC Motor Driver(2 Channels) [online]. In: . 2019, s. 6 [cit. 2024-04-12]. Dostupné z: https://download.kamami.pl/p578534-MDD3A%20Datasheet.pdf
[37] KiCaD. KiCad Docs [online]. 2023 [cit. 2024-03-23]. Dostupné z: https://docs.kicad.org/7.0/en/introduction/introduction.html
[38] KiCad. KiCad Docs [online]. [cit. 2024-03-23]. Dostupné z: https://docs.kicad.org/7.0/en/eeschema/eeschema.html
[39] Lekce 1 - Visual Studio Code - Úvod do editoru zdrojového kódu. Itnetwork [online]. 2024 [cit. 2024-03-23]. Dostupné z: https://www.itnetwork.cz/csharp/vscode/visual-studio-code-uvod-do-editoru-zdrojoveho-kodu
[40] Writing Your Own Debugger and Language Extensions with Visual Studio Code. CODE Magazine [online]. 2021 [cit. 2024-03-23]. Dostupné z: https://www.codemag.com/article/1809051/Writing-Your-Own-Debugger-and-Language-Extensions-with-Visual-Studio-Code
[41] Použití rozšíření Visual Studio Code pro portály. Microsoft [online]. 2023 [cit. 2024-03-23]. Dostupné z: https://learn.microsoft.com/cs-cz/power-apps/maker/portals/vs-code-extension
[42] MicroPython_INA3221 [online]. In: . 2021 [cit. 2024-04-19]. Dostupné z: https://github.com/neaxi/MicroPython_INA3221/blob/master/ina3221/ina3221.py
[43] LÁZNIČKOVÁ, Jana. Řídicí a vizualizační software pro propagační robotické vozítko [online]. Brno, 2024 [cit. 2024-05-04]. Dostupné z: https://www.vut.cz/studenti/zav-prace/detail/160032. Diplomová práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky. Vedoucí práce Miroslav Jirgl.
[44] Vl53l0x. In: GitHub [online]. 2021 [cit. 2024-05-08]. Dostupné z: https://github.com/kevinmcaleer/vl53l0x

SEZNAM PŘÍLOH

Příloha A - Elektronická příloha

Obsah přílohy CD:
RoboticVehicle.zip

- Diplomová práce
  - **xkolac16.pdf**
- Designspark Mechanical
  - **ARF_RBT.rsdocx**
- KiCad 7.0 – kolacny_DP
  - **kolacny_DP.kicad_pro**
- Zdrojový soubor Raspberry Pi Pico – pico
  - **ControlBoard.py**
  - **INA.py**
  - **main.py**
  - **Motor.py**
  - **MPU6050.py**
  - **pid.py**
  - **TCA9548A.py**
  - **UpLinkUART.py**
  - **vl5310x.py**
- Zdrojový soubor Raspberry Pi 4
  - **QR_Test.py**
- Blokové schéma zapojení – drawio
  - **schema_blokove.drawio**
  - **diagram_pico.drawio**