České vysoké učení technické v Praze
Fakulta dopravní

Katedra aplikované matematiky
Obor: Inteligentní dopravní systémy

Detekce překážek před autonomním vozidlem zpracováním dat z lidaru

Detecting obstacles in front of an autonomous vehicle by processing lidar data

BAKALÁŘSKÁ PRÁCE

Vypracoval: Jan Zmátlo
Vedoucí práce: Ing. Bohumil Kovář, Ph.D.
Rok: 2023

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta dopravní
děkan
Konviktská 20, 110 00 Praha 1

K611 .....**Ústav aplikované matematiky**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení studenta (včetně titulů):

Jan Zmátlo

Studijní program (obor/specializace) studenta:

bakalářský – ITS – Inteligentní dopravní systémy

Název tématu (česky): **Detekce překážek před autonomním vozidlem zpracováním dat z lidaru**

Název tématu (anglicky): Detecting obstacles in front of an autonomous vehicle by processing lidar data

Zásady pro vypracování

Při zpracování bakalářské práce se řiďte následujícími pokyny:

- Zpracujte přehled možných metod pro detekci překážek před autonomním vozítkem. Vysvětlete princip používaných senzorů a metody porovnejte.
- Projektové vozítko JetRacer rozšiřte o Lidar RPLiDAR A1M8. Konstrukční změny náležitě zdokumentujte.
- V jazyce Python naprogramujte aplikaci pro sběr, vizualizaci a vyhodnocení dat z lidaru.
- Ve spolupráci s autorem bakalářské práce Š. Jelínek: "Detekce překážek před autonomním vozidlem zpracováním kamerových dat", vytvořte robustní systém pro detekci překážek před autonomním vozítkem fúzí dat z lidaru a kanery.
- Funkčnost systému ověřte a vyhodnoťte na reálných datech.

Rozsah grafických prací: není specifikovaný

Rozsah průvodní zprávy: minimálně 35 stran textu (včetně obrázků, grafů a tabulek, které jsou součástí průvodní zprávy)

Seznam odborné literatury: dle doporučení vedoucího bakalářské práce

Vedoucí bakalářské práce: **Ing. Bohumil Kovář, Ph.D.**

Datum zadání bakalářské práce: **24. října 2022**
(datum prvního zadání této práce, které musí být nejpozději 10 měsíců před datem prvního předpokládaného odevzdání této práce vyplývajícího ze standardní doby studia)

Datum odevzdání bakalářské práce: **7. srpna 2023**
a) datum prvního předpokládaného odevzdání práce vyplývající ze standardní doby studia a z doporučeného časového plánu studia
b) v případě odkladu odevzdání práce následující datum odevzdání práce vyplývající z doporučeného časového plánu studia

L. S.

....

RNDr. Magdalena Hykšová, Ph.D.
vedoucí
Ústavu aplikované matematiky

prof. Ing. Ondřej Přibyl, Ph.D.
děkan fakulty

Potvrzuji převzetí zadání bakalářské práce.

.....

Jan Zmátlo
jméno a podpis studenta

V Praze dne ..... 24. října 2022

Prohlášení

Předkládám tímto k posouzení a obhajobě bakalářskou práci, zpracovanou na závěr bakalářského studia na ČVUT v Praze, Fakultě dopravní.

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

Nemám závažný důvod proti užívání tohoto školního díla ve smyslu § 60 zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon)

Podpis: ..... Dne: .....

Poděkování

Tímto bych chtěl poděkovat panu Ing. Bohumilu Kovářovi Ph.D. za odborné vedení, trpělivost a ochotu, kterou mi v průběhu zpracování bakalářské práce věnoval.

Název práce:	**Detekce překážek před autonomním vozidlem zpracováním dat z lidaru**
Autor:	Jan Zmátlo
Studijní program:	Technika a technologie v dopravě a spojích
Obor:	Inteligentní dopravní systémy
Druh práce:	Bakalářská
Vedoucí práce:	Ing. Bohumil Kovář Ph.D.

Abstrakt: V této bakalářská práci jsem měl za úkol vyvinout algoritmus pro sběr, vizualizaci a vyhodnocení dat z LIDARu, na kterém bude dále postaven robustní systém pro detekci překážek před autonomním vozítkem fúzí dat z LIDARu a kamery. Cílem bylo tento systém implementovat a experimentálně ověřit jeho funkčnost na vozítku Jetracer řízeném mikropočítačem Jetson Nano. Součástí práce je i text týkající se autonomních vozidel a jejich rozřazení do úrovní podle stupně autonomie. Další část práce je věnována možným senzorickým metodám detekce překážek a jejich porovnání. V práci je zdokumentováno rozšíření vozítka Jetracer o LIDAR RPLiDAR A1 formou 3D tisknutého nástavce, navrženého v grafickém programu Inventor. Detailně vysvětlíme kód pro detekci překážek a mapování prostoru kolem vozítka, který bude poté sloučen s kódem Šimona Jelínka pro detekci překážek zpracováním obrazu z kamery.

Součástí práce je i testování systému na reálné trati, kde bude vozítko reagovat na různé překážky. Funkčnost systému bude na základě těchto experimentů vyhodnocena. Závěr práce shrnuje dosažené výsledky a předkládá náměty pro budoucí rozvoj tohoto systému.

Abstract: In this bachelor's thesis, my task was to develop an algorithm for collecting, visualization, and evaluation of LIDAR data, on which we would build a robust system for detecting obstacles in front of an autonomous vehicle using combination of data from LIDAR and camera. The goal was to implement this system and to experimentally verify its functionality on the Jetracer vehicle controlled by the Jetson Nano microcomputer. Part of this thesis is a text on autonomous vehicles and their division into levels according to degrees of autonomy. Another part of the work is devoted to possible sensory methods of detection of obstacles and their comparison. In the work is documented expension of Jetracer vehicle with LIDAR RPLiDAR A1 in the form of a 3D printed attachment, designed in the graphics program Inventor. Then we will explain in detail the code for detection of obstacles and maping of space around the vehicle, which will then be combined with Simon Jelinek's code for detecting obstacles by processing the image acquired from camera. Part of the work is also a test of the system on a real track, where the car will react to different types of obstacles. The functionality of the system will be evaluated on the basis of these experiments. The conclusion of the work summarizes the results and introduces recommendations and targets for the future development of this system.

Obsah

Seznam obrázků

Seznam zdrojových kódů

Představení práce

Tématem této bakalářské práce je "Detekce překážek před autonomním vozidlem zpracováním dat z LIDARu[1]". Toto téma bylo zvoleno i vzhledem k častému používání lidaru v autonomních vozidelech. Cílem této práce je vyvinout algoritmus, který vyhodnocením dat z LIDARu, spolehlivě detekuje blížící se překážku. Tento algoritmus bude poté sloužit jako základ pro další implementace dalších modulů, které souvisejíc s detekovanými objekty a které se budou řešit v rámci projektu "Autonomní robotické vozítko".

V rámci problematiky detekce překážek před vozidlem bude zapotřebí porozumět možným alternativním senzorickým metodám detekce překážek z hlediska jejich principu, jejich předností a omezení. V tomto ohledu bude i nutné jednotlivé metody mezi sebou porovnat a diskutovat jejich vhodnost pro řešení našeho zadání.

V praktické části se již zaměřím na vytvoření zmíněného algoritmu, který bude implementovaný v programovacím jazyce Python. Tento kód bude dále sloučen s kódem pro detekci překážek zpracováním obrazu z kamery, načež bude jeho funkce otestována na vytvořené reálné trati. Toto testování ověří schopnost navrženého systému detekovat potenciální překážky. Výsledné poznatky potom mohou sloužit jako inspirace či přímo jako základ pro budoucí vývoj.

Motivace

Tento projekt přináší možná řešení dílčích problémů řízení autonomních vozidel se kterými se v moderním světě čím dále více můžeme setkávat. Autonomii vozidla jsou v dnešní době stále více vyvíjena a rozšiřována do po světě. Společnosti jako Tesla, Waymo nebo i Nvidia se stále předhánějí v rychlosti inovací a úrovni autonomie v jimi vyvíjených systémech autonomního řízení.

Spolu s vývojem autonomních vozidel se vyvíjí i LIDARy, jejichž parametry se stále zlepšují a v poslední době dochází i ke snižování jejich ceny. LIDARy jsou široce používány v autonomních vozidlech a můžeme je vidět i v oblastech mapování, zeměměřičství, geologii, vzdušných nebo satelitních systémech a v mnoha dalších.

Podobně jako u skutečných autonomních vozidel je spolupráce více druhů senzorů takřka nutností. Kombinace kamery a LIDARu je jednou z nejčastějších a společnosti

[1] LIDAR – Light Detection and Ranging

jako například Waymo je používají v jejich vlastních vyvíjených autonomních vozidlech.

Přínosy práce

Tato práce přispívá svým obsahem do projektu "Autonomní robotické vozítko"následovně:

- algoritmy pro detekci překážek pomocí LIDARu, vizualizaci prostoru kolem vozítka,
- volně dostupné zdrojové kódy, na které lze navázat příštími studenty tohoto projektu,
- návrhem nástavce, který poskytuje pevné upevnění LIDARu RPLIDAR A1 na vozítko Jetracer a umožňuje kompletní využití jeho funkcí,
- navrhuje systém pro detekci překážek před autonomním vozítkem fúzí dat z LIDARu a kamery.

Struktura bakalářské práce

Tato bakalářské práce začíná teoretickou částí, kde v první kapitole vysvětlím pojem autonomní vozidlo a různé stupně autonomie řízení. V další kapitole zmíním senzory používané pro detekci překážek, zejména se budu věnovat ultrazvukovým senzorům, RADARu, kamerám, LIDARu a porovnáme jejich vlastnosti a vhodnost pro detekci překážek. Zaměřím se na jejich funkční principy a časté využití.

třetí kapitola začíná praktickou část. Nejprve probereme podrobný postup při sestavě vozítka Jetracer. Další částí bude představení použitého LIDARu RPLIDAR A1, jeho specifikací, principu funkce a struktuře výstupních dat. Pak se přesuneme k instalaci operačního systému a dalších knihoven a vytvoříme základní kód pro ovládání autíčka. Budeme diskutovat komplikace, které vznikly při sestavování vozítka a zprovoznění software a přineseme jejich řešení.

Bude vysvětlen postup při návrhu a konstrukci nástavce, který bude spojovat LIDAR s vozítkem. Probereme i zvažované možnosti umístění a uzpůsobení nástavce. Další část bude věnována návrhu nové testovací trati a důvodům její tvorby. V závěru práce se podíváme na samotný algoritmus detekce překážek a mapování terénu. Podrobně si popíšeme význam každého příkazu a funkce. Následná část pak obsahuje kombinaci tohoto kódu s kódem Šimona Jelínka na detekci překážek zpracováním obrazu z kamery. V poslední kapitole pak popíšeme testování kompletní sestavy vozítka na trati a různé experimenty, na základě kterých pak bude projekt vyhodnocen. Na závěr si zhodnotíme výsledky celého projektu a navrhneme potenciální využití a rozvoj do budoucna.

Kapitola 1

Úvod

V této kapitole, budeme psát o důležitých okruzích, která souvisí s problematikou autonomních vozidel, jako jsou úrovně autonomie řízení, funkce LIDARu, případně dalších senzorů a jejich možnosti použití k detekci překážek. Porozumění těchto okruhů je nutné k návrhu a implementaci naší aplikace pro sběr, vizualizaci a vyhodnocení dat z LIDARu a kamery, která bude součástí vyvíjeného systému pro detekci překážek a mapovaní scény před vozítkem Jetson Nano.

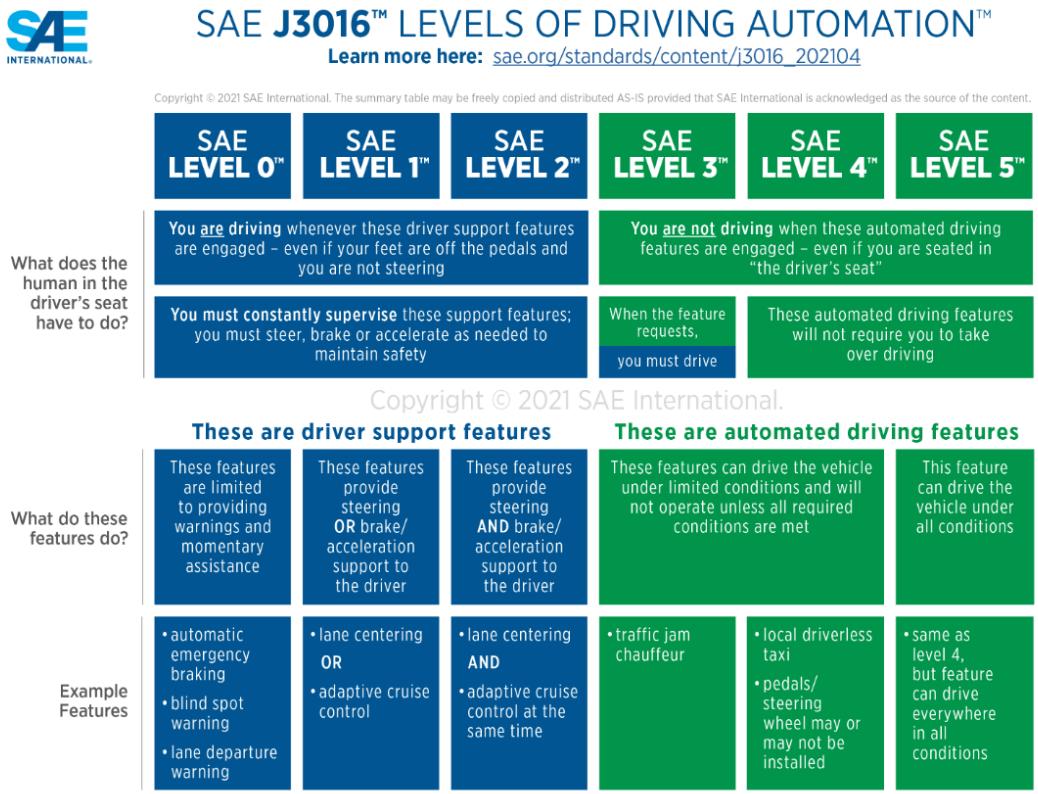
1.1 Co jsou to autonomní vozidla?

Autonomní vozidlo je takové vozidlo, které je schopné řídit v prostoru a reagovat na změny vnějších a vnitřních podmínek bez zásahu člověka. Základem je samozřejmě samotné vozidlo vybavené řadou senzorů, které snímají prostředí kolem vozidla a vytvářejí tak virtuální model světa, ve kterém se vozidlo nachází. Mezi tyto senzory patří například kamery, LIDARy, radary, ultrazvukové senzory, GPS a další. Data z těchto senzorů jsou pak zpracovány počítačem, který má funkci řídící jednotky celého vozidla. Tyto informace, získané zpracováním senzorových dat, jsou pak použity na nastavení parametrů řídících systémů vozidla. Snímání, zpracování a vyhodnocení dat probíhá několikrát za sekundu tak, aby byly zajištěny dostatečně rychlé reakční časy.

Stupeň autonomního řízení definuje společnost SAE[1] (Society of Automotive Engineers) do 5 úrovní. Těchto pět úrovní se ještě rozděluje do 2 sekcí. Do první sekce patří úrovně 0 až 2. V této sekci platí, že řidičem je stále člověk, a to za všech okolností. Dále musí pořád dohlížet na podpůrné systémy a provádět všechny akce jako u obvyklého vozidla. Do druhé sekce patří úrovně 3 až 4. V této sekci se již vozidlo řídí samo a to za všech okolností až na výjimku úrovně 3, kdy vozidlo muže vyžádat manuální řízení pokud projíždí náročným prostředím. V úrovních 4 a 5 toto možnost vyžádání manuálního řízení není.

[1] https://www.sae.org/

Konkrétně v ČR se v současné době stále více posouváme k autorizaci autonomních vozidel na českých silnicích. Existují již strategické dokumenty popisující kroky pro rozvoj autonomních vozidel v ČR. Jedním z dokumentů je "Vize rozvoje autonomní mobility"[2], který byl schválen vládou České republiky v roce 2017, který se věnuje jak přínosům tak problémům autonomní dopravy a definuje případné úlohy, která bude nutné vyřešit. Roce 2017 bylo přijato i "Memorandum o budoucnosti automobilového průmyslu v České republice"[3] týkající se mimo jiné i autonomní mobility jakožto prostředku konkurenceschopnosti českého automobilového průmyslu.



Obrázek 1.1: SEA úrovně[4]

1.1.1 Úroveň 0

Úroveň 0 je nejčastější pro současný typ vozidel. Vozidlo je plně řízeno člověkem. Ve vozidle mohou být varovné systémy, které informují řidiče o stavu vozidla nebo prostředí kolem něj, případně asistenční systémy, jako je například systém nouzového brzdění [1, 2].

[2] https://www.mdcr.cz/getattachment/Uzitecne-odkazy/Autonomni-mobilita/vize_rozvoje_autonomni_mobility.pdf.aspx

[3] https://www.mpo.cz/assets/cz/prumysl/2017/10/memorandum-o-budoucnosti-autoprumyslu-v-CR.pdf

[4] Zdroj: https://www.sae.org

1.1.2 Úroveň 1

Nejnižší úroveň autonomie řízení. Vozidlo je vybavené pouze jedním ADAS[5] systémem. Příkladem může být podpora pro řízení vozidla, ovládaní brzd nebo zrychlování. Součástí první úrovně mohou být i komplexnější systémy, jako je adaptivní tempomat, který přizpůsobí rychlost vozidla k vozidlu jedoucím před ním, nebo systém pro hlídaní jízdy v jízdním pruhu. Řidič ovšem stále musí dohlížet, řídit, brzdit a je zodpovědný za vozidlo [1, 2].

1.1.3 Úroveň 2

V druhé úrovni autonomie je vozidlo vybaveno nejméně dvěma ADAS systémy. Tyto systémy mohou být například kombinací adaptivního tempomatu a systému pro hlídaní jízdy v jízdním pruhu. V této úrovni musí řidič neustále dohlížet nad vozidlem a být připraven zasahovat do řízení. Vozidla této úrovně mohu být vybavena například i automatickým parkováním[1, 2].

1.1.4 Úroveň 3

Na této úrovni je vozidlo výrazně technicky vyspělejší oproti předchozí úrovni. Je schopné dělat samostatná rozhodnutí podle nastalé situace. Řidič již může odvrátit oči od vozovky a například napsat zprávu na telefonu. Může být ovšem stále požádán o převzetí řízení v náročnějších podmínkách, jako například při špatné viditelnosti [1, 2].

1.1.5 Úroveň 4

V této úrovni již řídí vozidlo zcela samo a řidič má možnost plně odvrátit pozornost od řízení a může se například najíst. Vozidlo je schopno převzít kontrolu nad vozidlem ve většině případů, občas tyto vozidla nejsou ani vybavena pedály. Ve velmi vzácných situací může požádat řidiče o převzetí kontroly – například v těžkém terénu, při vysokém sněhu, blátě nebo při špatné viditelnosti. Tyto vozidla jsou bohužel zatím kvůli omezení v legislativě povolena jezdit autonomně pouze v určených zónách a s omezenou rychlostí [1, 2].

K příkladům vozidel se čtvrtou úrovní autonomie patří vozidla francouzské firmy NAVYA[6] a vozidla společnosti Waymo[7], které poskytuje autonomní taxi službu v Arizoně v USA [1, 2].

[5] Advanced driver-assistance system
[6] Zdroj: https://www.navya.tech/en/
[7] Zdroj: https://waymo.com/waymo-one/

1.1.6 Úroveň 5

Nejvyšší a finální úroveň autonomních vozidel. Vozidla úrovně 5 jsou již schopna plně a za jakýchkoliv podmínek ovládat vozidlo a nepotřebují převzetí řízení člověkem. Člověk se může bez obav věnovat čemukoliv jinému než je řízení či dohledu nad vozidlem. Bohužel se zatím takové vozidlo nepodařilo vyvinout[1, 2].

Kapitola 2

Typy senzorů používáních v detekci překážek

Počítač, nebo jiná řídící jednotka v autonomním automobilu neví, kde se v danou chvíli nachází. Proto jsou tato vozidla vybavena senzory, které detekují podněty z okolí a přes elektrické signály je posílají jako data do řídících jednotek. Jedním z těchto podnětu jsou právě překážky, čili objekty, které je potřeba detekovat a reagovat na jejich přítomnost. Detekce objektů a překážek je důležitá úloha v automatizačním průmyslu.

Existuje celá škála různých důmyslných senzorů pro detekci objektů. V této kapitole se zaměříme na ty nejčastěji používané pro detekovaní objektů v autonomních vozidlech, vysvětlíme jejich princip a vlastnosti, načež pak porovnáme jejich vhodnost pro detekci objektů vzhledem ke zvolenému senzoru – LIDARu [3].

2.1 Ultrazvukové senzory

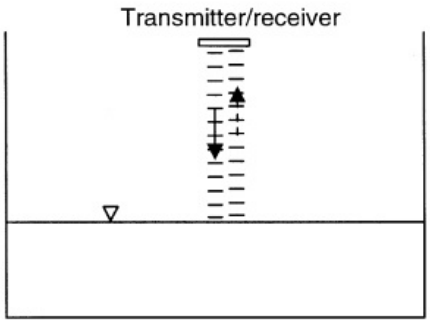
Ultrazvukové senzory jsou aktivní senzory, které se skládají z přijímače a vysílače zvukového signálu. Tento zvukový signál je ultrazvukový, což znamená že jeho frekvence je 20 kHz, nebo více. Tyto frekvence jsou již tak vysoké, že jsou pro lidské ucho neslyšitelné. Většina těchto senzorů používá na přijímaní a vysílaní ultrazvukových vln piezoelektrické, keramické nebo polymerové měniče.

Ultrazvukové senzory se dají použít různými způsoby. Pro využití v problematice detekce objektů se zaměříme na ultrazvukové senzory pro měření vzdálenosti. V tomto případě je vyslaná ultrazvuková vlna odražena od objektu zpět do přijímače senzoru. Doba tohoto procesu je změřena a z výsledného času je pak vypočítána vzdálenost objektu.

$$R = \frac{t \times v}{2} \tag{2.1}$$

Kde R je vzdálenost, t je celkový čas a v je rychlost světla. Tím, že pracujeme s odraženou ultrazvukovou vlnou, tak podmínkou úspěšného záznamu této odražené vlny

je to, že odrazová plocha objektu je rovnoběžná se snímací plochou, tedy paralelní ve směru snímaní.



Obrázek 2.1: Princip funkce ultrazvukového senzoru [4]

Jednou z důležitých vlastností těchto senzorů je jejich neinvazivní princip měření. Jelikož se jedná o ultrazvukové vlny, není třeba se obávat poškození detekovaného předmětu. Další důležitou vlastností ultrazvukové vlny je její útlum. Při použití ultrazvukového senzoru je zapotřebí zvolit správnou frekvenci přenášené vlny. Při vyšších frekvencích ultrazvuku totiž dochází i k vyššímu útlumu (avšak lze dosáhnout vyššího rozlišení signálu). Pokud je tedy zapotřebí detekovat objekty na větší vzdálenosti, je zapotřebí použít senzor vysílající ultrazvukové vlny menších frekvencí. Například pro vzdálenost 30 metrů se používá frekvence 23 kHz, případně pro vzdálenost 12 metrů 40 kHz.

Přenášecí medium signálu, kterým je v našem případě vzduch, je také velice důležité. Například jeho teplota může ovlivnit rychlost ultrazvukové vlny. Dále pak může docházet k vnějšímu rušení, jako je například silný poryv větru a to zvlášť při použití pouze samostatných pulzů ultrazvukového signálu.

Dalším důležitým faktorem je pak povrch detekovaného objektu. Nejvíce odrazivé objekty mají hladký a neporézní povrch, zatímco ty drsné a porézní povrchy jsou méně vhodné, protože na nich při odrazu vznikají nežádoucí ozvěny při nižších frekvencích. Dalším ztěžujícím faktorem těchto senzorů je Dopplerův efekt, při kterém dochází k posunu frekvence, pokud se vysílač pohybuje směrem od nebo k přijímači. Ultrazvukové senzory jsou poměrně levné a při menších nárocích na přenosové parametry i široce dostupné [3, 4].

V dnešních moderních vozidlech se ultrazvukové senzory používají spíše v parkovacích asistentech a na detekci překáže se vzhledem k jejich malému dosahu a rozlišení signálu, takřka nepoužívají. Ovšem pro problém, který řešíme a rozměry našeho autonomního vozidla, tento typ senzoru představuje potenciální řešení. Příkladem konkretního zařízení je ultrazvukový senzor HY-SRF05 SRF05[1], který se dá pořídit za 49 Kč a má dosah až 450 centimetru, což je v našem případě více než dostačující.

[1]Zdroj: https://rpishop.cz/vzdalenost/1321-ultrazvukovy-senzor-hy-srf05-srf05.html

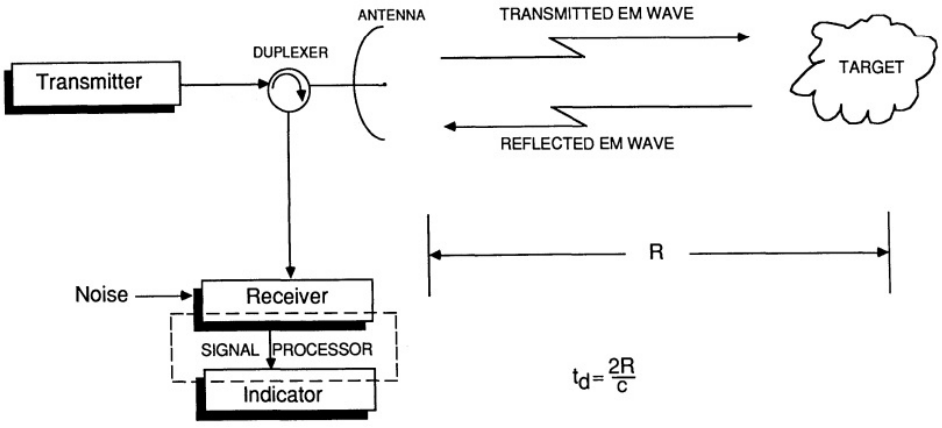
Obrázek 2.2: Ultrazvukový senzor HY-SRF05 SRF05

2.2 RADAR

RADAR (radio detection and ranging) je další aktivní senzor fungující na podobném principu jako ultrazvukové senzory. Namísto ultrazvukových vln se ale používají elektromagnetické pulzy energie. Celý proces měření začíná opět u vysílače signálu. V něm se nejprve vygeneruje signál elektromagnetické energie. Tato energie je pak přivedena na anténu přes duplexor a přenosovou linku. Přes anténu se energie dostane do atmosféry ve formě vlny, kde se pohybuje rychlostí světla. Tato vlna elektromagnetické energie se dá pomocí mechanického a elektronického nastavení antény i nasměrovat do požadovaného směru. Pokud je při tomto procesu vyzařovaná energie zachycena plochou cizího předmětu, tak dochází k jejímu održení. Většina této energie bude odražena do různých směrů, přirozeně ale bude i její část odražená zpět do RADARu. Tato odražená energie vracející se do RADARu se nazývá odražený rozptyl (v angličtině backscatter), zatímco energie odražená do jiných směrů se nazývá bistatický rozptyl (bistatic scatter), případně EM rozptyl (EM scatter). Stejně jako u ultrazvukového signálu jsme schopni spočítat čas návratu odražené energie, jednoduchým vztahem kde c je rychlost světla.

$$t_d = \frac{\text{distance}}{\text{velocity}} = \frac{2R}{c} \tag{2.2}$$

Část této odražené energie směřující k RADARu je zachycena stejnou anténou, která energii vyzářila do atmosféry. Z antény signál putuje opět přes duplexor a přenosovou linku do přijímače RADARu, kde je v tuto chvíli slabý signál ve formě radiové frekvence (RF) zesílen zesilovačem v přijímači a následně zpracován na použitelné informace jako je dosah, rychlost, amplituda, směr atd. Z přijímače data ještě cestují do případného indikátoru, například displeje, kde jsou prezentovány operátorovy daného RADARu.



Obrázek 2.3: Princip funkce RADARu s jeho základními komponenty [5]

Jak jsme se již zmínili RADAR se skládá ze čtyř základních komponentů vysílač, anténa, přijímač a indikátor. Ačkoliv to vypadá že duplexor a procesor signálu jsou také základní komponenty, jedná se spíše o součásti existujících komponentů. Duplexor je správně součástí přenosové linky, kde umožňuje jak přenos, tak příjem elektromagnetických vln pomocí jediné antény. Zatímco procesor signálu existuje často v nějaké formě jako kombinace přijímače a indikátoru.

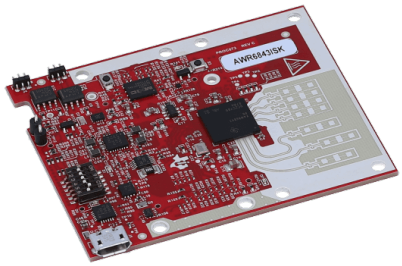
Hlavní funkce vysílače RADARu je vygenerovat RF signál o určitém charakteru a výkonu. Tyto parametry jsou buď vygenerovány oscilátorem přijímače přímo nebo muže být ještě přidán RF zesilovač. Hodnota parametrů je pak určená dle požadavků používaného systému. Může se jednat o nemodulovanou spojitou vlnu nebo o velice komplexní frekvenčně, fázově a časovým kódem modulovanou vlnu. Záleží na náročnosti zadání.

Anténa pak přebírá od přenosové linky RADARu RF signál, kde je pak tento signál vyzářen do přenosového media a naopak. Mimo této základní funkce nám také anténa umožňuje RF signál přijímat a vysílat do požadovaných směrů. Pokud je RADAR vybavený zmíněním duplexorem, pak je na tuto funkci potřeba pouze jedna anténa. Při vysílání do požadovaného směru je energie soustředěna do předem určeného tvaru paprsku, který je pak namířen do tohoto směru. Při příjmu signálu je opět vytvořen paprsek, který zachycuje energii odraženou od objektů. Konstrukce těchto antén se může drasticky lišit podle výkonu a složitosti zadání.

Dalším základním komponentem je přijímač RADARu. Jeho hlavní funkcí je přijmout slabý odražený signál přes přenosovou linku a zesílit ho do takové úrovně, ve které je zpracovatelný na informace obsažené v signálu. Existují různé konfigurace přijímače z nichž nejčastěji používaná je superheterodyne. Toto zesílení je optimalizované na průběh signálu na vysílači. Čím věrnější je tato optimalizace, tím přesnější jsou potom přijímané informace.

Nakonec se informace dostávají do indikátoru kde jsou prezentovány uživateli. Vzhled a rozhraní indikátoru se liší podle zadání [3, 5].

Obrázek 2.4: RADARU s 25 metrovou anténou v Chilboltonské observatoři[2]



Obrázek 2.5: Anténa RADARového senzoru na PCB[3]

RADAR má díky charakteru své funkce výhodné vlastnosti. To že se jedná o elektromagnetickou energii cestující vzduchem znamená, že RADAR muže správně fungovat i za špatného počasí – například mračna, mlha, tma nebo sníh. Toto je například výhoda oproti ultrazvukovým senzorům, které jsou oproti RADARu ovlivněny například i silným větrem. Dále je RADAR schopný identifikovat detailnější informace o detekovaném objektu či objektech, kterých muže být i víc. Podobně jako například u LIDARu, byť mnohem méně přesněji, jsme ze získaných dat schopni vytvořit mapu prostoru kolem RADARu a to i ve 3D.

Také se v případě konstrukčně větších projektů jedná o levnější zařízení. Tím, že signál cestuje rychlostí světla, tak jsme schopni velice rychle měřit čas mezi vyzářením elektromagnetické energie a detekcí její odražené části.

Dosah RADARu je také značně vyšší než v případě ultrazvukových senzorů. Jako většina těchto senzorů fungujících na principu odrazu od snímaného objektu, může dochá-

[2]Zdroj: https://www.ralspace.stfc.ac.uk/Pages/RADAR-Systems.aspx

[3]Zdroj:https://www.edn.com/antenna-on-package-technology-simplifies-automotive-in-cabin-radar-sensor-design/

zet k saturaci přijímače pokud je objet příliš blízko vysílači (anténě). Přesnost radarových systémů může dále ovlivnit rušení. Toto rušení muže být způsobeno například vodivými materiály, které obklopují snímané objekty. V takovém případně RADAR nedokáže objekt zaznamenat. Problémem jsou také cizí RF signály, které mohou interferovat s RF signálem RADARU. To má za následek zkreslená a nepřesná data. Ačkoliv jsou RADARy ve velkém měřítku levnější než jiné podobné systémy, v málem měřítku jsou relativně drahé [6].

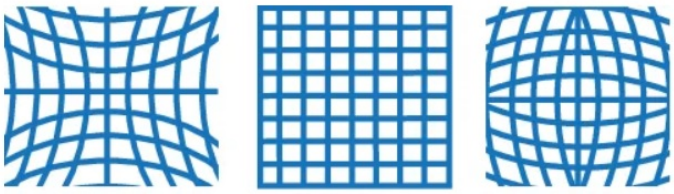
Co se týče použití RADARu v moderních autonomních vozidlech vznikají jisté nevýhody. Jak jsme se již zmínili, muže docházet například k rušení od cizích RF signálů. RADARem vykreslené objekty také nejsou nejpřesnější a uživatel tak může být ochuzen o důležité informace jako je například rozměr detekovaného objektu. Komplikace také může přinést vlastní umístění antény. RADAR by tedy namísto samostatného senzoru fungoval lépe spíše ve fúzi s ostatními senzory, jako je pravě LIDAR nebo kamera. V takové konfiguraci by mohl svými přednostmi redukovat nedostatky jiných senzorů jako jsou například špatná viditelnost nebo průhledný povrch objektu. Tím by mohl představovat potenciální cestu řešení naší problematiky v oblasti detekce objektů před autonomním vozítkem [3, 7].

2.3 Kamera

Dalším vhodným a velice populárním senzorem pro detekci objektů je kamera. Kamery jsou pasivní senzory. Zjednodušeně fungují na principu, při kterém je světlo vyzářené (odražené) z objektu zachyceno na fotocitlivý povrch přes čočku uvnitř konstrukce kamery. Potom co je daný paprsek světla zachycen na fotocitlivý povrch, který se také nazývá obrazová rovina (image plane), je světlo uloženo ve formě obrazu. Paprsky světla které prošly čočkou jsou totiž přes fotocitlivý povrch přeměny na elektrony, ze kterých pak vzniká měřitelné napětí. Toto napětí je zesíleno, převedeno přes ADC (analogově-digitální převodník), načež získáme hodnotu v metrických jednotkách. Velice zjednodušeně jsme z toho výstupu dále schopni zjistit polohu objektu formou vektoru se třemi prvky $[x, y, z]$. Tento vektor se potom v prostoru bude promítat přes čočku do bodu v obrazové rovině $[x', y']$. Jakmile máme promítnuté body a jejich souřadnice jako metrické jednotky, je možný i jejich převod na pixely. Tím nám vznikne obraz, který se dá dále zpracovávat na použitelné informace jako je detekce objektů či jejich klasifikace.

Nutným krokem při zpracovaní obrazu zachyceného kamerou je její kalibrace. Tato nutnost vzniká kvůli použité čočce, které svým tvarem způsobuje zkreslení snímku. Toto zkreslení se řeší korekcí s použitím známých objektů tj. šachovnic. Pomocí těchto šachovnic můžeme poté spočítat kalibrační parametry, které budou zohledňovat zkreslení. Příklady těchto zkreslení jsou Barrel a Pincushion [8].

[4]Zdroj: https://medium.com/@BabakShah/camera-technology-in-self-driving-cars-610371db4b0b



Obrázek 2.6: Zkreslení snímku v pořadí Barrel, žadné zkreslení, Pincushion[4]

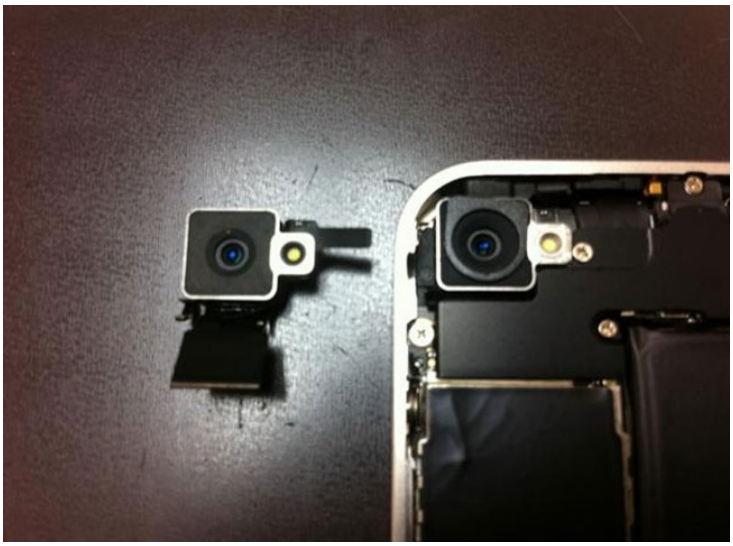
Jak jsme si již řekli, kamery jsou velice populárním řešením detekce objektů. Motivací této skutečnosti je fakt, že člověk používá zrak jakožto hlavní a nejvýkonnější smysl. Dopravní prostory jsou navrhovány pro lidské řidiče a pro jejich smysl zraku. Tato skutečnost je vidět například ve tvaru, odrazivosti pruhového značení nebo barvy dopravního značení, a to do té míry, že je i regulována standardy.

Jak jsme si již řekli, kamera je pasivní senzor. Tímto se odlišuje například od LIDARu a RADARu, u kterých se naopak jedná o senzory aktivní. Tím se rozumí, že na vnímaní prostředí kolem senzoru je zapotřebí přenos přes elektromagnetické vlny. To znamená větší spotřebu energie oproti pasivním senzorům, což značně snižuje efektivitu. Tento problém ještě narůstá u mobilních systémů, které mají typicky omezenou kapacitu energie a každý další energetický výdaj znamená zkrácení doby provozu. Další potenciální problém aktivních senzoru je vznik vzájemných rušení. Toto nastává když více senzorů operuje ve stejném prostředí, přičemž toto riziko narůstá s počtem přítomných senzorů. Například při přítomnosti více autonomních vozidel na vozovce.

Co se týká ekonomické stránky, jsou kamery, oproti senzorů jako je LIDAR, víceméně levným řešením detekce objektů. To dále zvyšuje jejich popularitu a počet systémů, které jsou na nich založené. Další výhody kamer jsou spojené s jejich konstrukcí. Kamery, oproti například LIDARu, mohu totiž být velice kompaktní. To můžeme vidět například na běžných mobilních telefonech obsahujících výkoné kamery, nepřesahující rozměry jednoho centimetru krychlového. Dále, jelikož kamery neobsahují žádné větší mechanicky pohybující se části (na rozdíl od LIDARu), mohou být konstruovány mnohem robustněji, což v kombinací s jejich malými rozměry znamená, že mohu být integrovány do systému mnohem snadněji.

Kamery mají ovšem i své nevýhody. K těm patří například jejich závislost na na dobrém osvětlení. Pokud není přítomný optimální zdroj světla, kvůli špatnému počasí či denní době, tak kamery nejsou schopné cokoliv zaznamenat. Na druhou stranu je takto závislý na světle i řidič, který to řeší zapnutím integrovaných světel vozidla. Navíc s problémy s počasím se potýkají i ostatní senzory zmíněné v kapitole 2. Dále je třeba zmínit, že kamery vyžadují velké výpočetní výkony k extrahování požadovaných informací z obrazu, a to kvůli obrovskému množství dat poskytnutých kamerou, které je třeba

[5]Zdroj: https://www.geeky-gadgets.com/white-iphone-4-gets-taken-apart-reveals-modified-camera-lens-02-05-2011/



Obrázek 2.7: Modul kamery iPhone 4[5]

zpracovat. Proto je potřeba správně nastavit parametry systému a tím optimalizovat množství dat ke zpracování.

Pro tyto faktory je kamera jako samostatný senzor tím nejlepším řešením pro realizaci reaktivního instinktivního chování pro autonomní vozidla. V kombinaci s ostatními senzory (LIDAR) představuje robustní a efektivní řešení pro detekci objektů [3].

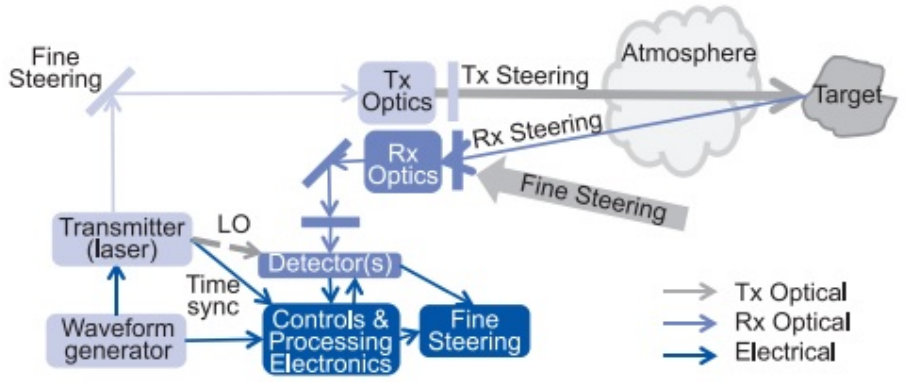
2.4 LIDAR

Posledním často používaným senzorem v autonomních vozidlech, na který se zaměříme, je LIDAR. LIDAR je aktivní senzor, který opět funguje na podobném principu jako RADAR a ultrazvukové senzory. Kde je v nějaké formě vyslán signál (v případě LIDARu laser), který se odrazí od předmětu a z vlastností tohoto odrazu jsou pak získány informace.

Nejprve si tedy ukážeme jak funguje systém LIDARu. Vše začíná u generátoru tvaru vlny, který generuje tvar vlny laserového signálu. Ten může mít více konfigurací, počínající od jednoho laseru, až po oscilátor vybavený více lasery či laserovými zesilovači. Pro správné nasměrování laserového signálu na objekt se používá vysílací optika, kterou laserový signál prochází po tom co je vyzářen z vysílače. Během cesty k objektu musí laserový signál projít médiem, kterým je ve většině případů atmosféra, ale muže jím být například i voda. Jakmile se signál dostane k objektu je odražen. Pak opět cestuje stejným médiem do přijímací optiky. Odtud je pak signál nasměrován do detektoru či detektorového pole a pak je zpracován na data. S detektorovým polem je možné provádět *flash imaging*, což znamená vytváření obrazu pouze s jedním pulzem signálu.

V následujícím schematu je zobrazen Bistatický LIDAR, ve kterém je přijímač a vysílač oddělen. Pokud by tyto zařízení byli v jedné součástce, pak takový LIDAR nazýváme

Monostatický [9, 10].



Obrázek 2.8: Schema systému Bistatického LIDARu [9]

Čas od vyzáření jednoho pulzu laserového signálu, následné cesty a odražení od objektu a nakonec zachycení přijímačem LIDARu se dá spočítat pomocí následujícího vzorce:

$$\Delta R = \frac{C}{2B} \tag{2.3}$$

Kde delta R je rozlišení dosahu, c je rychlost světla a B je šířka pásma systému, která se typicky odvíjí od šířky pásma přenosového signálu.

Po tom co laserový signál docestuje po odrazu do přijímače LIDARu, jsou na základě jeho parametrů získány informace o objektu. Tento proces nastává při dopadu světla na detektor načež je generováno elektrické napětí, které je v ideální přímé detekci rovné druhé mocnině intenzity dopadajícího světla. Tyto optické detektory reagují pouze na intenzitu dopadajícího světla, nejsou například schopné detekovat rychlé kolísání nosné frekvence kolem 200 až 600 THz s vlnovou délkou 0,5 až 1,5 μm [9, 10].

LIDARy se dají rozdělují na více typů. Například sem patří *Range-only* LIDAR neboli 1D-LIDAR, který měří pouze jednu dimenzi – vzdálenost. Pak jsou tu 2D-LIDARy, které měří vzdálenost v horizontální 2D rovině. Posledním významným typem LIDARů je 3D-LIDAR. Ten dokáže, jak již jméno napovídá, měřit vzdálenost v jak horizontální rovině tak i vertikální. Díky tomu jsme pak schopni z výstupu zkonstruovat 3D obraz objektu či okolí. Existují i jiné typy LIDARů jako například *Synthetic aperture LIDAR* nebo *Coherent LIDAR*, který dokáže měřit rychlost pohybujícího se objektu podle Dopplerova posunu a frekvence, případně *Synthetic aperture LIDAR* [9, 10].

LIDARy obecně jsou velmi komplexní senzory, které mají široké možnosti použití. Disponují vlastnostmi, které je dělají velice účinnými nástroji pro vizualizaci a detekci objektů. K těm patří například velice přesné vnímaní. Toto platí zejména u 3D-LIDARů, které jsou schopné vytvořit 3D mračna bodů s vysokým rozlišením, což umožňuje přesné a velice detailní vnímaní objektů a prostředí. Díky tomuto detailnímu vnímaní jsme i schopni velice přesně rozeznávat tvary objektů a dokonce i vytvářet mapy prostředí.

Mimo autonomní vozidla se LIDARy, kvůli jejich schopnosti vytvářet detailní mapy, používají v oborech jako archeologie, geografie, geologie, geomorfologie a seismologie. Dále, na rozdíl od kamer, je LIDAR plnohodnotně funkční jak ve dne, tak i v noci. LIDARu nedělají problém ani různé druhy nepříznivého počasí, jako je mlha nebo sníh. Zřejmou výhodou LIDARů je jejich rychlost. Doba návratu laserového signálu se pohybuje pouze v řádech nanosekund, což umožňuje získat velké množství dat během krátkých okamžiků. Toto je zvlášť výhodné při mapování venkovních prostor během leteckého mapování.

LIDAR je tedy velice výkoný a užitečný senzor, na velké množství aplikací, zahrnující i detekci objektů v okolí autonomních vozidlech. S jeho výhodnými vlastnostmi ovšem přichází i pár nevýhod. Jedno z hlavní a asi nejznámější nevýhodnou vlastností LIDARů je jejich vysoká cena, která je dána nákladnými komponenty v jejich konstrukci. Ve srovnání s podobnými snímacími technologiemi jako je kamera nebo RADAR, je LIDAR značně dražší a implementace více LIDARů, do jednoho autonomního vozidla, je z ekonomické stránky takřka nemožná. Dále kvůli malé vlnové délce laserového signálu, která se pohybuje v řádech mikrometrů, má LIDAR problémy s přesnou detekcí na delší vzdálenosti. Tedy, ačkoliv LIDAR dokáže přesně detekovat blízké objekty, tak jeho výkon klesá se vzdáleností a není schopný v těchto situacích fungovat tak spolehlivě jako RADAR, jehož vlnová délka signálu může být i v řádech centimetrů. LIDARy na delší vzdálenosti existují, ovšem jejich cena je značně vysoká a kvalita dat ne vždy uspokojující. Dalším problémem je náchylnost na rušení. Některé materiály mohu mít negativní vliv na kvalitu získaných dat. Průhledné materiály jako sklo nebo naopak velmi reflektivní materiály můžou zkreslovat nebo zeslabovat laserový signál a tak ovlivnit spolehlivost a přesnost detekování objektů [11, 12].

Na vzdory těmto překážkám je LIDAR stále velice používaný v autonomních vozidlech kvůli jeho schopnostem vnímat, detekovat a mapovat objekty a prostory. Dále, tak jako u předešlých senzorů, LIDAR je velice efektivní senzor pro detekci objektů, avšak funguje nejlíp v kombinaci s ostatními senzory jako je kamera a RADAR. Také stojí za zmínku, že se LIDARové technologie stále rapidně vyvíjejí, aby se dále vylepšila tato technologie a vyřešila některá omezení [3].

2.5 Porovnání senzorů

Nyní, když jsme si vysvětlili principy jednotlivých senzorů a vyjmenovali jejich přednosti a nedostatky, tak se zaměříme na jejich porovnání s LIDARem s ohledem pro detekci objektů před autonomním vozítkem.

Prvním kandidátem, který by mohl nahradit LIDAR ve funkci detekce objektů je jednoduchý ultrazvukový senzor. Ultrazvukový senzor na předku našeho autonomního vozítka by zcela jistě splnit úkol zadání detekce překážek. Ultrazvukové senzory jsou v celku jednoduché, kompaktní a jejich velmi nízká cena je nepopíratelný důvod jejich

použití v autonomních vozidlech. Nicméně jak již bylo zmíněno v části 2.1, ultrazvukové senzory mají značně limitované funkce. Měření vzdáleností je velice důležité pro funkci detekce překážek a ultrazvukové senzory se v přesnosti nemohou rovnat LIDARu. Čím přesněji víme kde se objekt nachází, tím spolehlivěji můžeme podniknout kroky pro vyhýbací manévr a nebo zastavení.

Za zmínku také stojí kratší dosah ultrazvukových senzorů (typicky pouze pár metrů) oproti LIDARu, který je viditelný hlavně v měření na dlouhé vzdálenosti, se kterými si LIDAR typicky snadno poradí. Toto ovšem momentálně nepředstavuje velké omezení, jelikož se naše zadaní tyká pouze malého vozítka. Další výhodnou funkcí, kterou ultrazvukové senzory postrádají je schopnost vytvářet mapy prostor, ve kterém se vozítko nachází. Například ani nejsme schopni od sebe rozeznávat jednotlivé objekty. Bez těchto funkcí limitujeme možnosti autonomního vozítka a tak omezujeme potenciální navazující projekty. Problémy může také představovat špatné počasí proti kterému jsou ultrazvukové senzory mnohem náchylnější než LIDARy. Celkově jsou ultrazvukové senzory dobrým způsobem jak levně a jednoduše detekovat, že se před nimi nachází objekt. Nicméně postrádají adaptabilitu na to zvládnout komplexnější situace, které mohou v budoucnu tohoto projektu vzniknout.

Dalším možným senzorem, který by mohl nahradit LIDAR je RADAR (část 2.2). Nejprve si tedy řekněme v jakých oblastech RADAR výhodnější. Dosah je u RADARů větší než u LIDARů. To tedy znamená přesnější detekce na větší vzdálenosti a tím pádem i spolehlivější předvídaní situací. Nicméně stejně jako u ultrazvukového senzoru se v našem případě jedna o malé autonomní vozítko, pro které je dosavadní dosah LIDARů dostačující pro drtivou většinu úloh. Důležitou výhodu je ovšem pořizovací cena. RADARy jsou takřka vždy levnější řešení než LIDARy. RADARy nemají funkční problémy při špatném osvětleni, jako například v noci či mlze, problémy ale mohou nastat s rozptylem signálu či jeho rušením při silném poryvu větru nebo kontaktu s vodivým materiálem. Nesporno výhodou LIDARu je samozřejmě jeho přesnost. Ačkoliv v přesnosti jsou RADARy přesnější než ultrazvukové senzory, nemohou se rovnat s přesností LIDARu a to zejména při detekci menších objektů, což konkrétně u RADARů muže znamenat, že menší objekty nebudou detekované a tím může dojít k nárazu do daného objektu. Pro naše účely by nicméně RADAR byl dostačujícím senzorem pro detekci objektů a představoval by alternativní řešení.

Posledním možným senzorem, který porovnáme s LIDARem je kamera. Nutno zmínit že používané vozítko JetRacer je již vybaveno jednou funkční kamerou, která slouží hlavně jako detektor vodících linií a zároveň bude fungovat jako sekundární senzor pro detekci objektů, pro který bude LIDAR sloužit jako ověřovací senzor pro vyloučení chybných záznamů. Proto je potřeba se zamyslet, zda by tato kamera mohla sloužit jako samostatný senzor, případně zvážit přidání další kamery do systému a použít stereo vidění. Pokud jsou v systému přítomny dvě kamery, pak je pomocí triangulace možné rekonstruovat hloubku

obrazu ze dvou současně pořízených snímků. Tato konfigurace by nicméně byla velmi náročná, jelikož by se jednalo o dvě separátní kamery a jejich nastavení by vyžadovalo kalibraci a přesnou synchronizaci. Tento problém by šel vyřešit osazením vozítka modulem stereo-kamery, kompatibilním se základní deskou Jetson Nano, riskujeme tím ovšem možnost nekompatibility s API výrobce vozítka.

Pokud tedy použijeme řešení pouze s jednou kamerou, která je součástí vozítka, tak nejsme schopni měřit vzdálenosti, což jak již bylo zmíněno je velice podstatné pro naše zadání. Dalším problémem je závislost na osvětlení a příznivém počasí. Výhodnou vlastností kamer je ovšem jejich schopnost rozeznávat barvy, které je odlišuje od všech zmíněných senzorů a může být vítanou vlastností pro budoucí navazujíc projekty. Kamery jsou velice výkoným senzorem pro detekci objektů a jsou takřka vždy implementovány v autonomiích vozidle pro různé účely, nicméně pro naše zadání disponují omezeními, které je dělají jakožto samostatný senzor nedostačujícími.

Ačkoliv by splnění našeho zadání bylo více méně možné s kterýmkoliv ze zmíněných senzorů, nejlepším řešením se zdá být kombinace více senzorů. Tímto způsob se jednotlivé nedostatky a omezení jednoho senzoru vykompenzují přednostmi druhého a naopak. Kombinace kamera a LIDAR je velmi robustní systém detekce překážek, jelikož kamera, jakožto pasivní senzor efektivněji využívá energii a pokud by kamera přišla do úskalí při například špatném osvětlení nebo obecně při špatně detekovaném záznamu, tak LIDAR může sloužit jako kontrolní a spolehlivý senzor detekce objektů. S touto myšlenkou bude v následujících častí systém detekce překážek implementován a naprogramován.

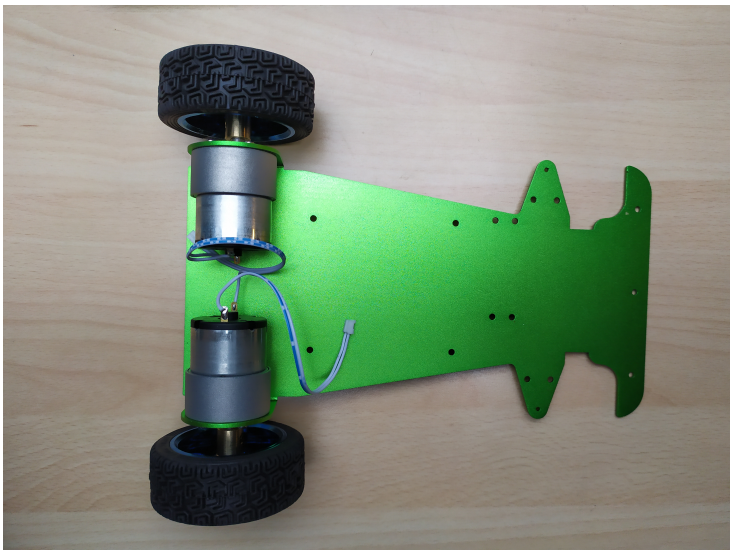
Kapitola 3

Sestavení vozítka JetRacer

V této kapitole popíšeme postup při sestavení vozítka Waveshare JetRacer[1]. Představíme použitý LIDAR - *Slamtec RPLIDAR A1* a jeho specifikace. Pak budeme diskutovat možnosti montáže LIDARu na vozítko JetRacer a popíšeme design možných nástavců a jejich umístění.

3.1 Konstrukce vozítka

Nejprve byly na spodní desku přimontovány dva převodové motory (obr. 3.1), které slouží jako pohon zadních kol vozítka. Do plastových koleček byly vloženy spojky, které byly následně upevněny šroubky. Kolečka pak byla nasazena na hřídele motorů s pomocí šroubků s vnitřním šestihranem.

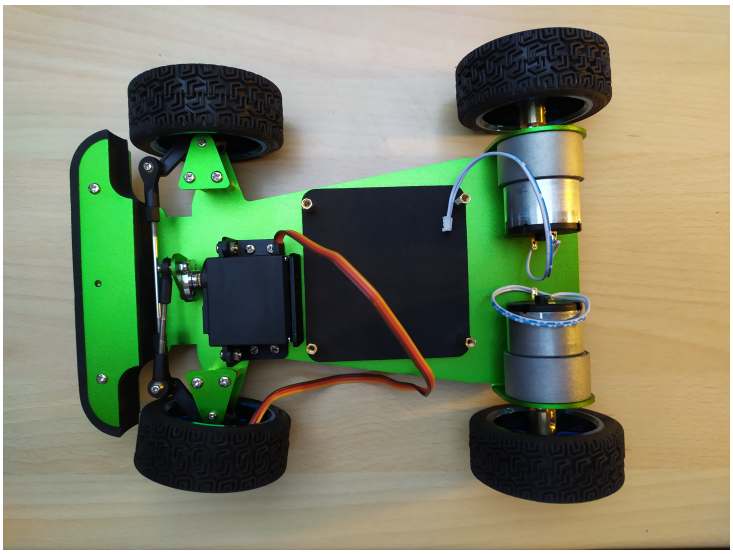


Obrázek 3.1: Spodní deska s motory a koly

V dalším kroku byl přišroubován servo motor spolu s jeho držákem k podvozku vozítka. Tento servo motor slouží jako volant pro ovládaní předních kol a umožňuje řídit směr jízdy

[1]Zdroj: https://www.waveshare.com/jetracer-ai-kit.htm/

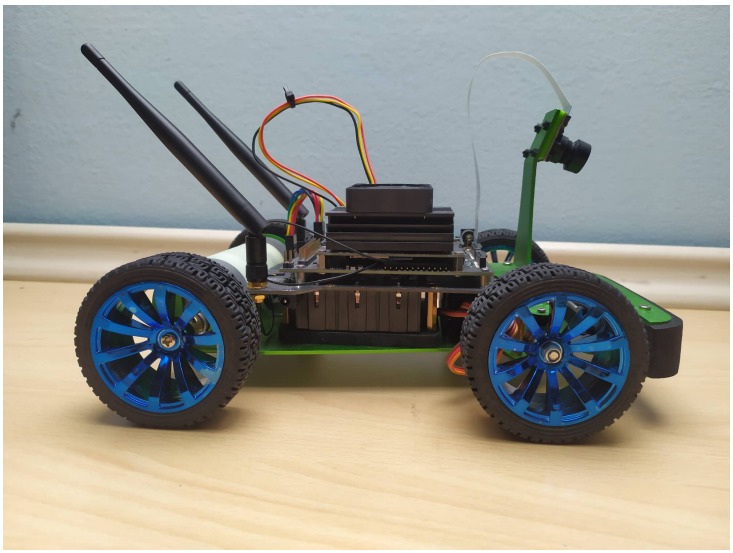
vozítka. K tomuto servo motoru byl následně přišroubován systém hřídelí s držákem, spojující samotný servo motor s předními koly. Přední kola byla navlečena na kovové tyče, což umožnilo jejich otáčení kolem jejich os. Pak byly na obě kola našroubovány dvě šestiúhelníkové vzpěry s trojúhelníkovou deskou , čímž byla ukončena konstrukce obou předních kol. Předek podvozku byl dále vybaven molitanovým nárazníkem. Tím bylo dokončeno sestavení podvozku (obr. 3.2).



Obrázek 3.2: Spodní deska rozšířená o přední kola

K podvozku byla přišroubována rozšiřovací deska JetRacer s držákem kamery a vývody na antény. Hlavním úkolem rozšiřovací desky je snížit kabeláž a zjednodušit montáž. Rozšiřovací deska je poháněna třemi Li-ionovým bateriemi s napětím 3,6 V a kapacitou 3000 mAh. Baterie a kabely obou hnacích motorů pak byly připojeny k desce. Na rozšiřovací desku pak byl zapojen a přišroubován mikropočítač NVIDIA Jetson Nano[2], pomocí něhož je vozítko možné programovat. K Jetson Nano a k anténám byla připojena bezdrátová NIC karta s duálním režimem, která nám spolu s anténami umožní programovat a ovládat vozítko přes internet. Nakonec byl na pasivní chladič mikropočítače přišroubován chladící větrák. Tám byla montáž vozítka JetRacer dokončena (obr. 3.3).

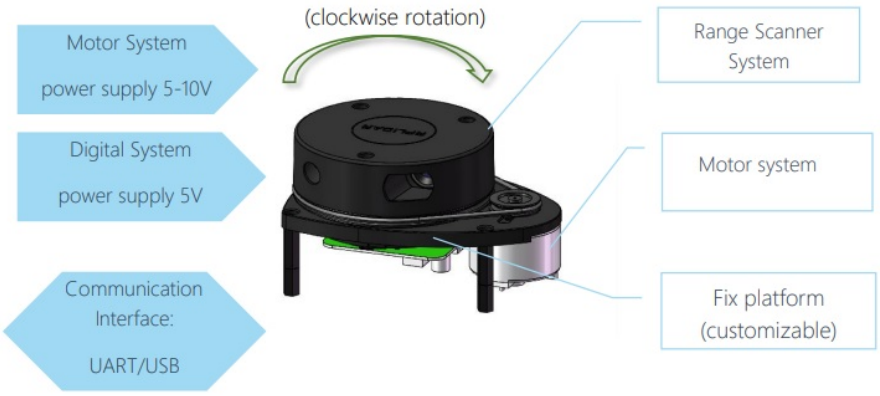
[2]Zdroj: https://www.nvidia.com/cs-cz/autonomous-machines/embedded-systems/jetson-nano/



Obrázek 3.3: Kompletní konstrukce

3.2 RPLIDAR A1

Jedním se zásadních kroků byl výběr LIDARu, který by nebyl příliš nákladný a zároveň by splnil naše požadavky. V tomto ohledu byl zvolen 360 úhlový 2D LIDAR *RPLIDAR A1* od společnosti *Slamtec*. Mimo to, že se jedná o 2D LIDAR, který dokáže měřit vzdálenost v horizontální rovině, je vybaven i motorem, díky kterému je schopný pořizovat záznamy ve 360 stupních kolem své osy. Jeho maximální dosah 12 metrů. Dále je vybaven komunikačním rozhraním USB/UART.

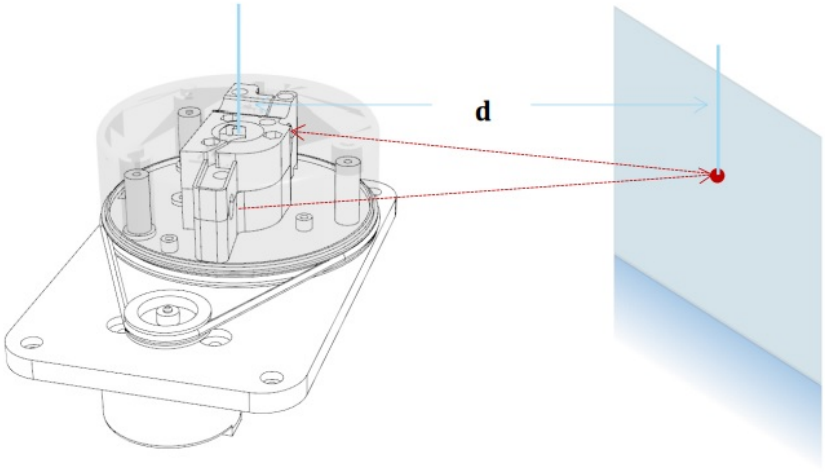


Obrázek 3.4: Konstrukční schéma LIDARu RPLIDAR A1[3]

Má skenovací frekvenci 5.5 Hz při vzorkování 1450 bodů za otáčku s možností nastavení až na maximálně 10 Hz. Systém je důmyslně navržen tak, aby se skenovací frekvence přizpůsobovala rychlosti motoru.

[3]Zdroj:https://bucket-download.slamtec.com/d1e428e7efbdcd65a8ea111061794fb8d4ccd3a0/LD108_SLAMTEC_rplidar_datasheet_A1M8_v3.0_en.pdf

Systém funguje na principu laserové triangulace a používá vysokorychlostní systém na zpracování a sběr dat od společnosti *Slamtec*. Po vyslání modulovaného laserového signálu, je tento signál odražen od objektu a zachycen vizuálním sběrným systémem, kde jsou s pomocí digitálního procesoru získána data o objektu. Tyto data obsahují informace o vzdálenosti od objektu, úhlu ve kterém se objekt nachází od LIDARu, kvality záznamu a začátku nového měření.



Obrázek 3.5: Princip měření RPLIDAR A1[4]

3.3 Návrh a konstrukce nástavce LIDARu

Při návrhu nástavce, na kterém bude umístěn LIDAR *RPLIDAR A1*, bylo nejprve zapotřebí zvážit jeho lokaci na JetRaceru. První možností bylo umístit LIDAR na nárazník JetRaceru. Toto řešení by bylo v celku jednoduché, protože bychom nemuseli konstruovat žádné složité nástavce a postačila by jednoduchá deska spojující LIDAR s nárazníkem. Hlavní výhodou tohoto řešení by byla možnost snímat objekty vysoké od 2 centimetrů. Nevýhodou tohoto řešení je omezený úhel snímaní LIDARu. Jelikož by LIDAR byl umístěn na nárazníku JetRaceru, tak by byl úhel snímají omezen zhruba na třetinu jeho potenciálu (LIDAR by nedetekoval objekty skryté tělem JetRaceru). Toto omezení by nás při našem zadaní (detekce objektů před vozidlem) sice neomezovalo, ale limitovalo by budoucí projekty, ve kterých by bylo nutné detekovat objekty v celém okolí vozítka.

Druhou možností bylo umístit LIDAR mezi rozšiřovací desku s bateriemi a mikropočítač Jetson Nano. LIDAR by byl umístěn na rozšiřovací desce s pomocí šroubků a nad ním by byla uchycena Jetson Nano deska, která by stála na sloupcích připevněných k rozšiřovací desce. Tento způsob umístění by nám dovolil snímat objekty vysoké 10 centimetrů a výš. Dále by nám to umožnilo, oproti prvnímu řešení, snímat objekty v téměř

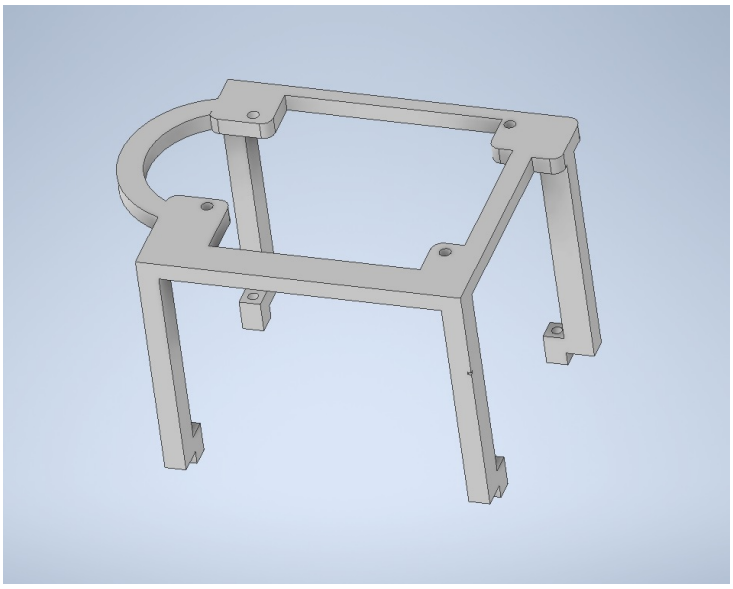
[4]Zdroj:https://bucket-download.slamtec.com/d1e428e7efbdcd65a8ea111061794fb8d4ccd3a0/LD108_SLAMTEC_rplidar_datasheet_A1M8_v3.0_en.pdf

v plném snímacím úhlu. Problém způsobují sloupky držící Jetson Nano. Jelikož by byly součástí jeho snímacího úhlu, nebylo by možné zachytit objekty, které se nacházejí za nimi. Dalším problémem by byla nutnost předělání držáku na kameru, jelikož by, stejně jako sloupky na Jetson Nano, blokoval dohled LIDARu.

Třetí možností bylo umístit LIDAR nad větrák s chladičem, který reguluje teplotu procesoru desky Jetson Nano. V tomto řešení máme neomezený 360 stupňový výhled kolem vozítka a jsme schopni snímat objekty vysoké 15 centimetrů a výš.

Po zvážení všech možností jsme se rozhodli zvolit třetí variantu uchycení a to zejména z důvodu její jednoduchosti provedení a největšímu potenciálnímu využití v budoucích projektech.

Další krokem bylo navrhnout nástavec (Obr. 3.6), který by šel spolehlivě umístit nad chladič a Jetson Nano, a na který půjde nainstalovat i samotný LIDAR. Pro návrh jsem použil program *Inventor* od společnosti *Autodesk*. Při návrhu bylo nejprve důležité přesně stanovit rozměry jednotlivých součástek, na které bude nástavec upevněn. Z toho důvodu jsme namísto měření zvolili kratší cestu a volně dostupné modely jednotlivcích součástí jsme stáhli ze stránek výrobců používaných zařízení[5].

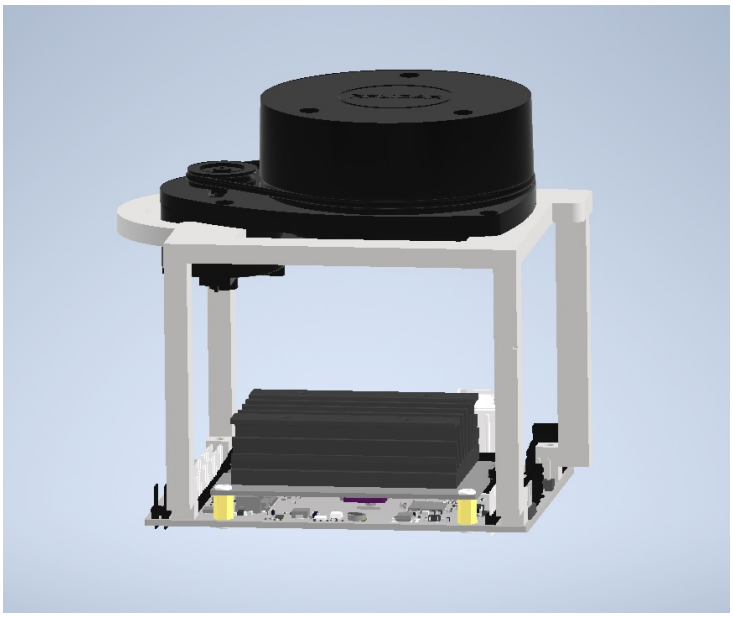


Obrázek 3.6: Design nástavce pro montáž LIDARu

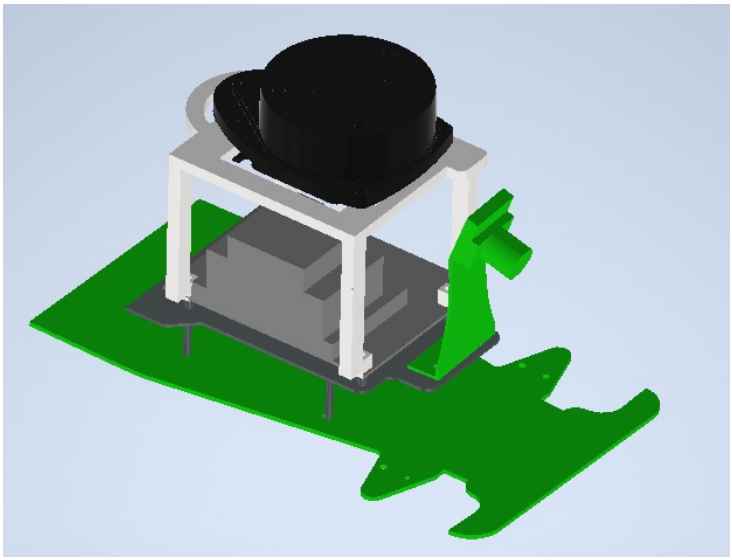
Dále už jen následoval jednoduchý proces designování nástavce. Při tomto procesu bylo potřeba zvážit výšku nástavce. Jelikož se před LIDARem nachází sloupek s kamerou, je nutné, aby snímací rovina LIDARu byla nad touto kamerou. Zároveň ale chceme mít tuto rovinu co možno nejníže k zemi, protože vše, co by se nacházelo pod touto rovinou by nebylo zaznamenáno a tím by hrozilo nebezpeční nárazu do nízkých překážek. Možným řešením tohoto problému by bylo navrhnout nový kratší sloupek kamery. Takto bychom

[5]Zdroje: https://grabcad.com/library/nvidia-jetson-nano-2, https://bucket-download.slamtec.com/893feee8a83ebfa6e6a816f8e31e38f413a9b736/RPLIDAR%20A1M8%20.STL

mohli i zkrátit výšku nástavce. Toto řešení, by ale ovlivnilo existující konfigurace kódů a proto jsme od něj odstoupili.



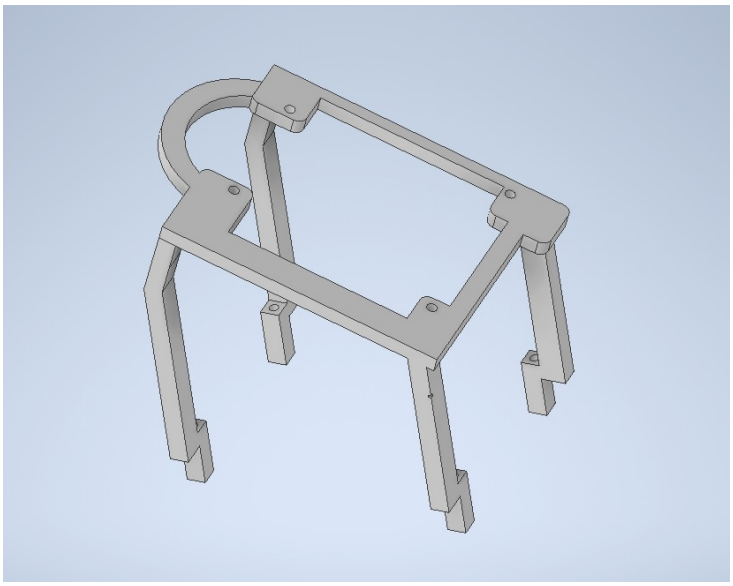
Obrázek 3.7: Sestava z modelů Jetson Nano, LIDARu a nástavce



Obrázek 3.8: Kompletní sestava na rámu vozítka Jetracer

Nakonec jsme ještě přišli na alternativní verzi nástavce, který by držel LIDAR ve stejném úhlu jako sloupek kameru (Obr. 3.9, 3.10). Takto bychom sice přišli od schopnost zmapovaní prostoru v 360 stupňovém úhlu, ale umožnilo by nám to detekovat i velice nízké překážky.

Horizontální nástavec je zhruba 75 milimetrů vysoký, zatímco nástavec v úhlu je v nižší části 70 milimetrů a ve výší části 100 milimetrů vysoký. Sloupce nástavce, ve tvaru obdélníku, mají rozměr 8×6 milimetrů. Nástavec v úhlu má stejný sklon jako sloupek

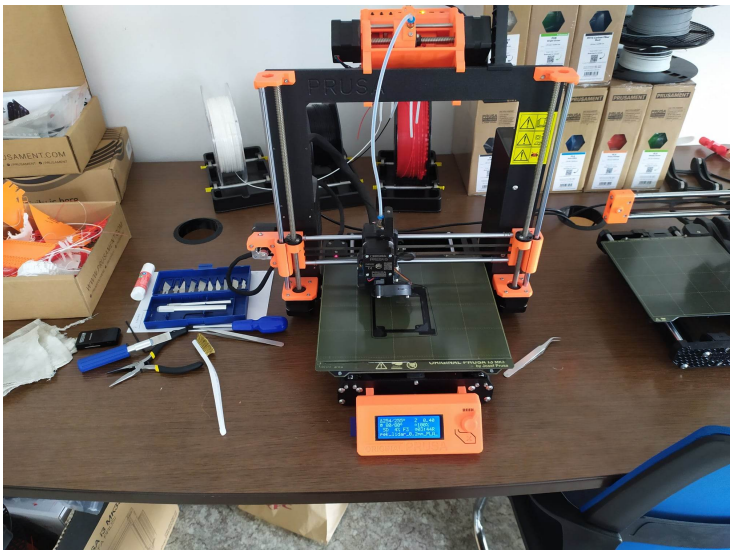


Obrázek 3.9: Samostatný nástavec v úhlu

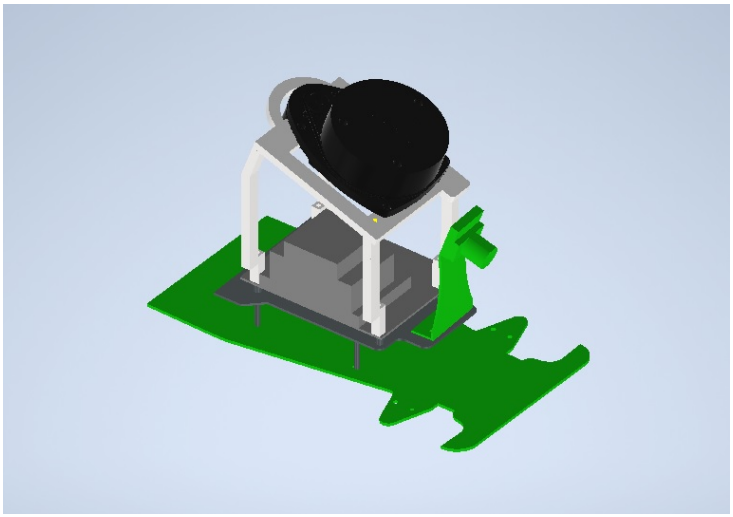
kamery a to 70 stupňů. Všechny tyto rozměry byly zvoleny tak, aby byl získán plný rozhled LIDARu nestínění kameru a zároveň aby bylo možné detekovat co možná nejnižší objekty.

Obrázek 3.10: Sestava z poskytnutých modelů a nástavce v úhlu

Oba nástavce byly vytisknuty na 3D tiskárně Prusa MK3 (Obr. 3.11) na Florenci v místnosti F208. Nástavec byl upevněn k desce Jetson Nano a k LIDARu šroubky M2 5×16 a M2,5 5×20 s maticemi M2 a M2,5.



Obrázek 3.11: Fotka 3D tiskárny v procesu tisku nástavce



Obrázek 3.12: Kompletní sestava v úhlu na rámu vozítka Jetracer

Kapitola 4

Implementace

V této kapitole předposlední kapitole se zaměřím na princip aplikace pro sběr, vizualizaci a vyhodnocení dat z LIDARu a na algoritmus systému pro mapování prostoru a detekci překážek. Fotodokumentace a zdrojové kódy software budou součástí práce, včetně vysvětlení jejich funkce. Nakonec si vysvětlíme postup při kombinaci kódu pro detekovaní překážek použitím dat z LIDARu s kódem Šimona Jelínka, který řeší stejnou problematiku zpracováním obrazu z kamery.

4.1 Příprava a instalace software

Pro instalaci softwaru do Jetracer byla použita SD karta s kapacitou 128 GB, na kterou byla pomocí software Etcher nahrán obraz nejnovější verze operačního systému Jetpack 4.5.1 s konfigurací pro Jetracer. Karta pak byla vložena do slotu na sd kartu na desce Jetson nano. Deska byla pomocí micro USB kabelu připojena k počítači. Vozítko JetRacer bylo připojeno k naší wifi síti a bylo dostupné přes stránku JupyterLab. Nakonec byl nainstalován překladač jazyka Python a deska byla nastavena do 5 W režimu, který zajistí, že Jetson nano nebude odebírat více energie než jsou baterie schopné dodat.

Dále bylo potřeba nainstalovat knihovny a driver pro LIDAR *RPLIDAR A1*. Jetpacku pro Jetracer byl v době instalace jen limitovaně funkční a jediné co fungovalo byl servomotor pro ovládaní směru jízdy. Bylo zapotřebí chybějící funkce doplnit. Jednou z komplikací, kterou v té době použitá verze Jetpacku měla, byla nefunkční verze knihoven pro ovládaní motorů kol. Pro vozítko Jetracer bylo potřeba použít alternativní verzi knihoven od Waveshere, namísto nainstalované verze NVIDIA-AI-IOT.

Dalším problémem byla nainstalovaná verze Pythonu (2.7). Vzhledem k tomu, že jedena z knihoven *adafruit-servokit* pro ovládání motorů používala funkci implementovanou ve verzi Pythonu 3, bylo zapotřebí vytvořit virtuální prostředí, ve kterém se tato verze Pythonu používá jako výchozí. Tím vznikl další problém, protože nainstalovaná verze knihovny OpenCV byla určena pro Python 2.7. Přes veškeré snahy se nám nepodařilo zprovoznit OpenCV s verzí Pythonu 3 a to i přes vlastní kompilaci OpenCV.

Tento problém byl odstraněn v nejnovější verzi Jetpacku, kde je OpenCV pro straší verzi Pythonu 2.7, ale i novější verzi 3.

Poslední překážkou před vlastní tvorbu kódu, bylo nastavení komunikace Jetraceru s LIDARem. Jako první bylo potřeba nainstalovat driver, tak aby Jetson Nano byl schopný komunikovat s LIDARem. USB signály konvertujeme na sériové tak, aby jim Jetson Nano rozuměl. Po připojení LIDARu přes poskytnutý micro usb adaptér, jsme pomocí příkazu `lsusb` zjistili že LIDAR od společnosti *Slamtec* vyžaduje *Cygnal Integrated CP210x UART bridge driver*. Tento driver jsme snadno získali z volně dostupného úložiště installACMModule[1]. V nejnovější verzi Jetpack je tento driver už nainstalován a tím pádem je tento krok zbytečný.

Následně bylo zapotřebí získat oprávnění pro použití portu, který obsluhuje LIDAR. Tento problém jsme vyrašili přihlášením do dvou korespondujících skupin *tty* a *dialout* pomocí příkazu `sudo usermod -a -G tty dialout {username}`. Po tomto úkonu již bylo pouze třeba nainstalovat vhodnou knihovnu usnadňující práci s LIDARem. První zvažovanou knihovnou byl *rplidar* z uložiště SkoltechRobotics[2]. Tato jednoduchá knihovna posloužila jako testovací pro ověření funkčnosti LIDARu, nicméně vzhledem k limitované dokumentaci jsme se rozhodli knihovnu nevyužít. Dále jsme ještě testovali *RPLidar* knihovnu z uložiště Roboticia[3], kde vznikl obdobný problém s dokumentací. Nakonec jsme našli knihovnu *PyRPlidar* z uložiště Hyun-je[4], která splnila všechny naše nároky – byla snadná na použití a měla patřičnou dokumentaci.

4.2 Algoritmus kódu na detekci překážek a mapovaní prostoru

Při tvorbě programu, který by pomocí dat z LIDARu spolehlivě detekovat potenciální překážku před vozítkem a zároveň dokázal vytvořit 2D mapu 360 úhlového prostoru okolo vozítka jsme použili programovací jazyk Python. Python je velice populární dynamicky interpretovaný programovací jazyk, který nabízí veliké množství podporujících knihoven pro různé aplikace. Je poměrně jednoduchý a dá se rychle naučit. Dále je na internetu dostupné velké množství již existujících kódů, ze kterých se dá čerpat. Nevýhodou je že není tak výkony jako jako například C++ nebo Java, což představuje určité omezení a to zejména pokud je program spuštěn na relativně pomalém procesoru Jetson Nano.

Ze všeho nejdříve je tedy zapotřebí načíst potřebné knihovny, které budou využívány ve zbytku kódu.

[1] Zdroj: https://github.com/jetsonhacks/installACMModule
[2] Zdroj: https://github.com/SkoltechRobotics/rplidar
[3] Zdroj: https://github.com/Roboticia/RPLidar
[4] Zdroj: https://github.com/Hyun-je/pyrplidar

```
import math
import numpy as np
import matplotlib.pyplot as plt
from math import cos, sin, radians, pi
import lidar_to_grid_map as lg
import json
from collections import deque
```

Zdrojový kód 4.1: Import potřebných knihoven

Abychom z datového výstupu LIDARu získali potřebné informace, na základě kterých budeme detekovat překážky a mapovat prostor okolo vozítka, tak bylo zapotřebí data nejprve načíst a zpracovat. Z již zmíněné knihovny *pyrplidar* jsme tedy využili již existující funkci `simplescan` a upravily ji pro naše potřeby. Nejdříve se připojíme k LIDARu.

```
lidar = PyRPlidar()
lidar.connect(port="/dev/ttyUSB0", baudrate=115200, timeout=3)
```

Zdrojový kód 4.2: Připojení k LIDARu

Potom nastavíme, aby se získaná data zaznamenala do textového souboru a pomocí další funkce z knihovny *pyrplidar* `ForceScan` zahájíme skenování prostoru kolem LIDARu.

```
f = open(f, 'w')
scan_generator = lidar.force_scan()
orig_stdout = sys.stdout
sys.stdout = f
```

Zdrojový kód 4.3: Funkce na zahájení snímaní

Výstupem jsou potom iterace skenování, z nichž každá má 4 sekce v *json* formátu. První sekcí je `Start Flag`, která je omezená na boolovská data *True* a *False* a pouze označuje začátky nových skenů při stavu *True*. Druhou sekcí je kvalita měření. Třetí a čtvrtá sekce, které jsou pro nás nejdůležitější, obsahují úhel vyslaného paprsku a vzdálenost LIDARu od objektu, od kterého se paprsek odrazil. Tyto iterace jsou postupně nahrávany pomocí cyklu *for* do souboru. V cyklu do souboru zaznamenáváme pouze sekce úhel a vzdálenost. Sekce `Start Flag` a kvalita jsou pro nás nepotřebné a zbytečně by komplikovaly další práci s daty. Tento cyklus proběhne ve výchozím stavu 360 krát, aby se zajistili záznamy ze alespoň zhruba jednoho celého otoční LIDARu. Hodnota určující počet záznamu se dá jednoduše zvýšit(snížit), podle toho jak si chceme následně počínat. To by nám například zajistilo dostatek redundantních náhradních iterací pro případ, že by některá z předchozích byla chybně zaznamenaná toto by avšak způsobilo i o něco delší průběh cyklu.

```
for count, scan in enumerate(scan_generator()):
    print(json.dumps({"angle": scan.angle, "distance": scan.distance}))
    if count == 360:
        break
```

Zdrojový kód 4.4: Cyklus pro nahrávaní a filtrování dat

Jakmile doběhne cyklus a naměřená data jsou uložena v textovém souboru můžeme zahájit jejich zpracování a analýzu. K tom nám slouží funkce `file_read`. Tato funkce nejprve načte všechny řádky naměřených dat ze souboru do proměnné typu *pole – measures*. Dále odstraníme chybná měření, ve kterých je vzdálenost od LIDARu rovna nule. Nakonec provedeme ještě jedno filtrovaní, pomocí kterého v proměnné *measures* zůstanou pouze hodnoty zaznamenané v úhlu 300 až 70 stupňů a zároveň se vzdáleností odrazu od 140 do 290 milimetrů.

```
def file_read(f):
    measures = [json.loads(line) for line in open(f)]
    measures = [measure for measure in measures if measure['distance'] != 0]
    measures = [measure for measure in measures if 140 <= measure['distance'] <= 290 and
    (300 <= measure['angle'] <= 360 or 0 <= measure['angle'] <= 70)]
    return measures
```

Zdrojový kód 4.5: Funkce pro analýzu dat

Z takto filtrovaných dat je pak velice snadné určit zda LIDAR detekoval přakážku či nikoliv. Pokud se před JetRacerem nachází překážka v rozmezí definované vzdálenosti a úhlu je proměnná *measures* neprázdná, což značí přítomnost alespoň jednoho záznamu splňující kriteria překážky. Pokud je proměnná *measures* prázdná, znamená to, že žádná překážka nebyla detekována a Jetracer může pokračovat v jízdě. Tyto alternativy ověříme jednoduchou podmínkou `if`. V případě překážky se tato skutečnost vypíše a budou nastaveny nulové hodnoty do proměnných, které ovládají otáčky obou motorů.

```
if __name__ == "__main__":
        f = "out.txt"
        simple_scan(f)
        measures = file_read(f)
        if len(measures) > 0:
            print("je tam objekt, zastav!")
            car.throttle_gain = 0
            car.throttle = 0
```

Zdrojový kód 4.6: Funkce pro analýzu dat

Tento způsob ukládání výstupu z LIDARu do souboru a následné načtení ze soubor byl použit zejména kvůli snazší kontrole formátu dat a samotných hodnot úhlu a vzdáleností. Dále nám to umožnilo pracovat se stejnými daty ve více funkcích, které byly v různých samostatně spustitelných programech. Nicméně ve finální verzi programu byl tento mezikrok vynechán a namísto uložení do souboru, byl výstup uložen do proměnných v rámci jednoho kódu. Tato úprava, ve funkci `simplescan`, zahrnovala vytvoření prázdné proměnné typu pole pod názvem *measures*, do které budou načteny výstupy z LIDARu. To proběhne v upraveném cyklu *for* kde namísto příkazu *dump* použijeme příkaz *append*, který přidává prvek (v našem případě jeden záznam) na konec proměnné typu list. Proměnnou *measures* pak poskytneme pro další zpracovaní příkazem *return*

```
 measures = []
    for count, scan in enumerate(scan_generator()):
        measures.append(({"angle": scan.angle, "distance": scan.distance}))
        if count == 360:
            break
...
return measures
```

Zdrojový kód 4.7: Funkce pro analýzu dat

Další změna nastala ve funkci `fileread`, kde byl pouze vynechán řádek ve kterém se v cyklu *for* načítají hodnoty ze souboru do proměnné. Proměnná s těmito hodnotami je vstupní parametr této funkce. Zbytek kódu zůstal nezměněn. Tyto změny přinášejí rychlejší běh programu, protože není nutní otevírat vícekrát stejný soubor a ukládat do něj, případně číst z něj data.

```
 def file_read(measures):
    measures = [measure for measure in measures if measure['distance'] != 0]
    measures = [measure for measure in measures if 140 <= measure['distance'] <= 290 and
    (
            250 <= measure['angle'] <= 360 or 0 <= measure['angle'] <= 120)]
    return measures
```

Zdrojový kód 4.8: Funkce pro analýzu dat

Mimo detekci překážek jsme z načtených dat dále schopni vytvořit 2D-mapu prostoru, ve které se Jetracer nachází. Pro tento úkol jsme použili existující algoritmus[5], který byl modifikován pro naše potřeby. Nejprve je potřeba data opět zpracovat. To uděláme pomocí funkce `mapping_area`, která nejdříve data, stejně jako funkce `file_read` uloží do proměnné *measures*. Z dat v této proměnné jsou odstraněny chybné záznamy, které obsahují vzdálenost rovnou nule. Dále naměřené záznamů úhlů a vzdáleností uložíme do jejich vlastních proměnných *angles* a *distances*, které jsou typu *pole*.

```
def mapping_area(f):
    measures = [json.loads(line) for line in open(f)]
    measures = [measure for measure in measures if measure['distance'] != 0]
    angles = []
    distances = []
    for measure in measures:
        angles.append(measure["angle"])
        distances.append(measure["distance"])
    angles = np.array(angles)
    distances = np.array(distances)
    return angles, distances
```

Zdrojový kód 4.9: Funkce rozřazení dat do proměných

Potom co máme tyto dvě sady párových dat, tak můžeme pokračovat jejich vizualizací. Před tím, než zaznamenané body vykreslíme do grafu, tak musíme spočítat jejich

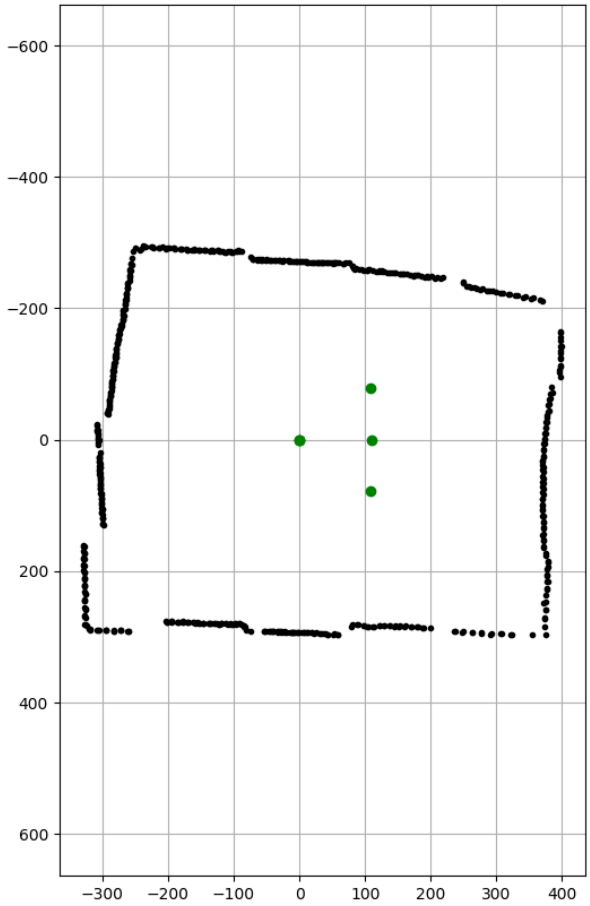
[5]Zdroj: https://atsushisakai.github.io/PythonRobotics/modules/mapping/lidar_to_grid_map_tutorial/lidar_to_grid_map_tutorial.html

souřadnice v prostoru. Pro přehlednost a budoucí snadnou práci s daty jsme se rozhodli používat kartézské souřadnice na místo existujících polárních. Tyto souřadnice získáme jednoduchým převodem.

$$x = r \cdot \cos(\theta), \tag{4.1}$$

$$y = r \cdot \sin(\theta). \tag{4.2}$$

Kde r je vzdálenost k objektu a θ je úhel, ve kterém se bod nachází. Poté stačí pouze definovat rozměr grafu a vykreslit data nastavenou barvou a vzhledm bodů. Aby byla mapa ještě názornější, tak jsme do grafu přidali další sadu dat, která reprezentuje vozítko JetRacer a jeho umístění v prostoru (zelené body).



Obrázek 4.1: 2D mapa prostoru kolem Jetraceru

```
ang, dist = mapping_area("out.txt")
oy = np.sin(np.deg2rad(ang)) * dist
ox = np.cos(np.deg2rad(ang)) * dist(figsize=(6, 10))
plt.plot([ox, np.zeros(np.size(ox))], [oy, np.zeros(np.size(oy))], ".",c='black',
    linestyle = 'None')
ang, dist = mapping_area("vozitko.txt")
oy = np.sin(np.deg2rad(ang)) * dist
ox = np.cos(np.deg2rad(ang)) * dist
plt.plot([ox, np.zeros(np.size(ox))], [oy, np.zeros(np.size(oy))], "o-",c='green',
    linestyle = 'None')
```

Zdrojový kód 4.10: Náhradní souřadnic

Nakonec můžeme ještě konfigurovat parametry grafu a obrázek grafu uložit ve formátu *png* pro další použití, případně graf můžeme i zobrazit. Tímto jednoduchým způsobem jsme dostali 2D mapu prostoru kolem našeho vozítka (Obr. 4.1).

```
plt.axis("equal")
bottom, top = plt.ylim()
plt.ylim((top, bottom))
plt.grid(True)
plt.savefig('prostor.png', bbox_inches='tight')
#plt.show()
```

Zdrojový kód 4.11: Vykreslení grafu

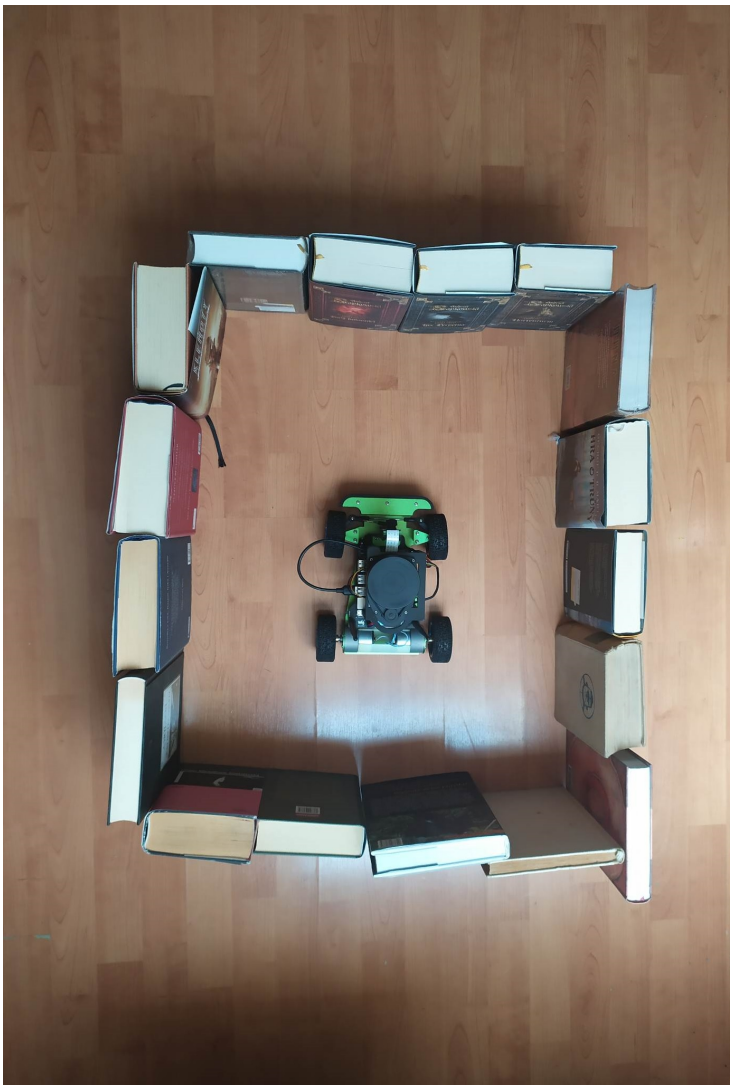
Takto vytvořená 2D-mapa prostoru nám mimo jiné umožní identifikovat rozměry překážky a jejich přesnou lokaci vůči Jetraceru. Dále muže sloužit jako startovací bod pro další navazující práce, které budou analyzovat prostředí v okolí vozítka. Alternativně bychom dále pomocí zmíněného algoritmu a funkcí z kódu `lidar_to_grid_map`, který lze volně stáhnout z úložiště PythonRobotics od Atsushi Sakai[6], mohli tuto mapu transformovat do mřížkové mapy. Ta nám umožní vidět volná a obsazená místa na mapě, které jsou vizualizovány jako zelené a červené buňky. Tato varianta je velice populární způsob reprezentace dat a dá se použít v různých dalších aplikacích.

4.3 Kombinace kódů

Posledním krokem implementační části této práce je kombinace popsaného kódu pro detekci překážek pomocí dat z LIDARu s kódem Šimona Jelínka, jehož úkolem bylo vytvořit funkci na detekci překážek zpracováním obrazových dat z kamery.

Díky této spolupráci jsme byli schopni prozkoumat dva různé přístupy k řešení společné problematiky detekce překážek. Navíc, kombinací více senzorů, podobně jako v reálných autonomních vozidlech, jsme schopni zmírnit nebo úplně odstranit některé nedostatky, kterými jednotlivé senzory (LIDAR a kamera) samostatně trpí. K těmto nedostatkům patří například problematické vnímání hloubky obrazu při použití jedné kamery. To

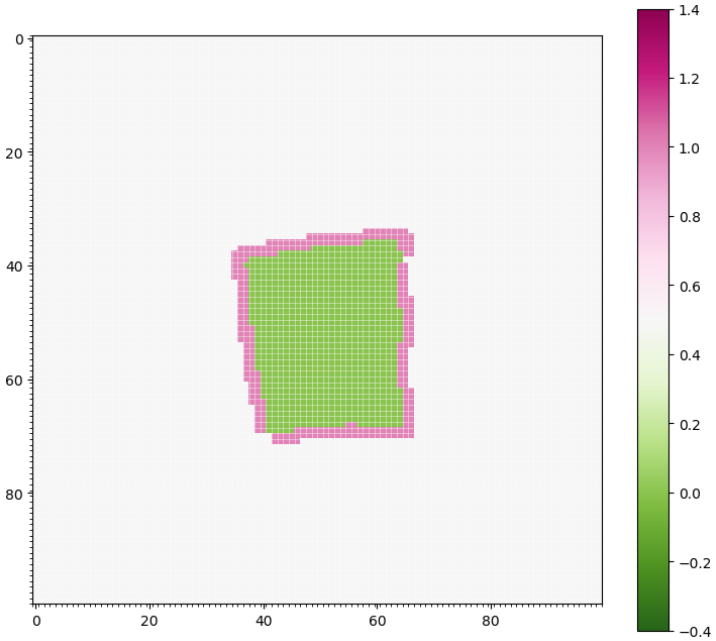
[6] Zdroj:https://github.com/AtsushiSakai/PythonRobotics



Obrázek 4.2: Situace, která odpovídá 2D-mapě prostoru na obr. 4.1

představuje zásadní problém při detekci překážek. Systém založený pouze na vyhodnocování obrazu z kamery tak nedokáže posoudit zda se v cestě nachází fyzická překážka, či pouze piktogram na vozovce nebo jiná ne konzistence povrchu. Tato omezení nám pomůže odstranit instalovaný LIDAR. Jelikož se laserové paprsky odráží pouze od fyzických objektů, tak se nemůže stát že by se nechal zmást obrazcem na vozovce. Naopak, kamera nám zase poskytuje velké množství dat a tím i informací, které se dají použít v různých dalších aplikacích, jako je například rozpoznávání dopravního značení, případně detekce vodících čar.

Kód Šimona Jelínka, mimo funkce detekce překážek, obsahuje i funkce pro detekci vodících čar a jízdu mezi liniemi. Algoritmus pro detekci překážek funguje na principu detekce změny intenzity jasu mezi vodícími liniemi. Identifikace takovéto změny zmámena, že na vozovce může být cizí předmět, tedy překážka, před kterou je nutno zastavit. Nicméně nadprahovou změnu intenzity jasu v jízdním pruhu může způsobit i piktogram vodorovného dopravního značení, případně výrazná změna textury povrchu vozovky, a



Obrázek 4.3: Mřížková mapa předešlé situace

v takových případech by zastavení bylo nežádoucí. Proto je v takovém případě namísto okamžitého zastavení, spuštěno detekování překážky aktivním senzorem – LIDARem. Pokud je LIDARem detekován objekt, znamená to, že na cestě je skutečně překážka a je spuštěna funkce pro zastavení vozítka. V opačném případě to znamená že změna intenzit jasu v jízdním pruhu neodpovídá překážce a vozítko pokračuje v jízdě. Pokud dojde ke stavu, kdy bylo vozítko Jetracer zastaveno, a poté byla překážka manuálně odstraněna, dojde k znovu rozjetí vozítka a inicializaci primární funkce, která detekuje překážky a zajišťuje jízdu mezi vodícími pruhy.

Struktury kódu detekce překážek pomocí LIDARu, která je uvedená v kapitole 4, zůstává ve větší míře nezměněna. Došlo pouze k přidání `if` funkce, která spouští celý kód pokud kamera detekuje překážku. V případě, že byly detekovány potenciální překážky, tak jsou odpovídající data zapsána do proměnné typu pole po názvem *objects*. Pokud je toto pole neprázdné, tak se spustí kód, který detekuje překážky pomocí LIDARu.

```
1             if len(objects)>0:
```

Zdrojový kód 4.12: Podmínka spuštění detekce objektů lidarem

V další části kódu, pokud kamera chybně detekovala překážku, tak je příkazem `else`, vyprázdněna proměnná *objects*, čímž jsou všechny nepravdivé záznamy o detekovaných překážkách odstraněny. V opačném případě jsou záznamy v *objects* ponechány, což znamená přítomnost alespoň jednoho záznamu fyzické překážky.

```
1             else:
2                 objects = []
```

Zdrojový kód 4.13: Vyprázdnění pole objects

Kapitola 5

Testování a vyhodnocení experimentů

5.1 Tvorba testovací trati

Pro testování bylo nutné získat nahrávku, která obsahuje obraz z kamery, na které by bylo možné "offline"testovat a optimalizovat program pro jízdu mezi pruhy. Po prvních testech jsme zjistili, že plachta s natištěnu tratí (Obr. 5.1), dodaná jako součást stavebnice JetRacer, má nedostačující kvalitu jak povrchu, tak i tištěného motivu. Plastová plachta s rozměry 3×2 metrů a váhou 1.43 kilogramů, byla po roztažení značně pomačkaná a tím pádem i nerovná. To znamenalo komplikace při budoucím projíždění trati a samozřejmě i při nahrávaní testovacích sekvemcí. Jednotlivé nerovnosti byly totiž programem vyhodnocovány jako samostatné pruhy, což dělalo nahrávku nepoužitelnou. Dalším problémem byla barva plachty a tištěného motivu dráhy.

Obrázek 5.1: Fotografie rozložené dodané plachty

Jelikož je žlutý potisky na bílém povrchu už z principu hůře rozlišitelný, program často nedokázal ve snímku najít správnou dráhu, zejména na místech, která byla výrazněji nasvícená. To představovalo v kombinaci s lesknoucím se bílím plastem velkou komplikaci.

Z těchto důvodů jsme se rozhodli vytvořit svou vlastní trať pro testování. Pro návrh jsme použili stejné parametry jako u dodané trati. Šířka pruhů je 263 milimetrů a vodící čáry jsou široké 35 milimetrů. Nejvhodnějšími barvami pro naší testovací trať by byla bílá na černém povrchu. Tato kombinace barev je lehce rozeznatelná na nahrávce a program by s ní neměl mít potíže. Abychom ovšem nemuseli vytisknou velkou černou plochu, což je finančně náročnější řešení, navrhli jsme proto klasickou variantu černý motiv na bílem pozadí a o inverzi barev se postará software. Využili jsme program *Inkscape*[1], a to zejména kvůli jeho jednoduchému použití, přehlednému rozhraní a navíc jedná se o software, který je dostupný zdarma. *Inkscape* nám ale především umožnil nastavit rozměry tištěné plochy, což bylo nutností, jelikož standardní formáty nejsou pro naše zadání vhodné. Trať byla vytištěna na ploteru *HP Designjet T2500* v copy centru FD ČVUT v místnosti B112.

5.2 Testování

Posledním krokem v naší práci bylo pomocí vytištěné trati experimentálně otestovat funkčnost systému, který detekuje překážky před vozítkem zpracováním dat z kamery a LIDARu. Nejdříve byl nainstalován potřebný software pro správnou funkci obou částí programu do jednoho vozítka Jetracer. Následně započalo samotné testování.

Vozítko bylo vždy manuálně postaveno před začátek trati a spuštěno. Pokud vozítko správně detekovalo obě vodící linie tak se dalo do pohybu.

Obrázek 5.2: Foto z testovaní

[1] Zdroj: https://inkscape.org/

Správná funkce systému byla ověřena tím, že jsme na trati vytvářeli různé situace, které mohou nastat. Mezi tyto situace patří:

- *trať bez jakýchkoliv objektů, které by mohli spustit zastavovací funkce* – testujeme správnou funkci primární funkce pro detekci vodících čar a jízdu mezi liniemi,
- *trať s překážkou na jejím konci* – testujeme správnou funkci detekce této překážky kamerou i LIDARem a zastavení vozítka před překážkou,
- *trať s obrazcem černého čtverce* – testujeme detekci falešného objektu kamerou, která nebude potvrzena LIDARem a vozítko nezastaví,
- *trať s překážkou, která je po zastavení vozítka odstraněna* – testujeme znovu rozjetí vozítka po odstranění překážky.

Po otestování základních funkcí systému jsme přidali další experimenty, ve kterých bylo testováno víc funkcí najednou:

- *trať obsahuje jak obrazec čtverce tak reálnou překážku* – vozítko má přejet obrazec čtverce, nezastavit a pokračovat v jízdě až k překážce, před kterou zastaví,
- *trať obsahuje jak obrazec čtverce tak reálnou překážku, která je po zastavení vozítka odstraněna* – testujeme znovu rozjetí vozítka na trati z předchozího experimentu.

Obrázek 5.3: Vozítko před rozjezdem na testovací trati

Abychom důkladně prověřili všechny funkce testovaného systému detekce překážek, tak byl každý experiment několikrát opakován. Z každého experimentu vznikly nahrávky, které jsou součástí této práce.

Na základě provedených experimentů nebyly Výsledky funkce systému vždy uspokojivé. Vozítko mělo častokrát probléme detekovat obě vodící linie pokud se při jízdě příliš odchýlilo od ideálního směru jízdy. V těchto případech zastavilo. Tento problém způsobovala nesprávně kalibrovaná kamera.

Další problém nastával v rychlosti detekce. Objekt musí být nejprve detekován kamerou, poté je aktivována detekce LIDARem a až po vyhodnocení dat z LIDARu je možné vozítko zastavit. Tento proces netrvá déle než 1 sekundu nicméně i takto krátká doba ne vždy stačí na včasné zastavení.

Obě uvedené komplikace jsme vyřešili správnou kalibrací kamery a úpravami kódu. Nepodařilo se je zcela eliminovat, ale výrazně jsme redukovali jejich výskyt. K dalšímu zlepšení by vedla lepší optimalizace kódů, případně jejich implementace v jazyce Java nebo C++.

I přes tyto komplikace považujeme výsledek jako úspěšný. Většina komplikací totiž vznikla ve funkci, která zajišťuje jízdu vozítka mezi jízdními pruhy, která nebyla hlavním cílem tohoto projektu. Náš hlavní cíl byla funkce na detekci překážek pomocí fúze dat z LIDARu a kamery. Tato funkce systému byla až na drobné výjimky spolehlivá. Vozítko téměř vždy zastavilo před reálnou překážku a nikdy se nenechalo zmást obrazcem na trati.

Obrázek 5.4: Foto z testovací místnosti, Florenc

Veškeré testování proběhlo v budově ČVUT fakulty dopravní na adrese Na Florenci 25, v knihovně Ústavu aplikované matematiky.

Závěr a další vývoj

Cílem této práce bylo vytvořit robustní systém pro detekci překážek před autonomním vozítkem JetRacer fúzí dat z LIDARu a kamery a tento systém ověřit na reálných datech. V textu jsme se nejprve věnovali vysvětlení konceptu autonomního vozidla načež jsme zmínili všech 5 úrovní autonomie podle SEA.

Dále je v práci zpracován přehled potenciálních senzorů pro detekci překážek, které se používají v tradičních autonomních vozidlech a vysvětlili jsme jejich principy. Tyto senzory jsme následně porovnali s LIDARem a uvedli důvody jeho použití v tomto projektu.

V další, praktické části, jsme nejdříve vysvětlili postup při montáži vozítka JetRacer a poté jsme představili parametry a princip funkce LIDARu *RPLIDARu A1*. Práce pokračuje instalací potřebného softwaru, popisem komplikací a jejich řešení, na které může čtenář narazit.

Dále jsme se věnovali úkolu rozšíření vozítka JetRacer o LIDAR *RPLiDAR A1* pomocí 3D vytisknutého nástavce, který byl navržený v programu *Inventor*. V práci diskutujeme parametry nástavce, způsob uchycení a další možné designy.

Kvůli nekvalitní trati, která byla dodána s vozítkem JetRacer popisujeme návrh nové v programu Inkscape.

V závěru praktické části jsme detailně rozepsali implementaci algoritmu pro detekci překážek a mapování prostoru v programovacím jazyce Python. Vysvětlili jsme každou funkci a příkaz a navrhli možná vylepšení. Implementační část končí kombinací kódu na detekci překážek pomocí LIDARu s kódem Šímona Jelínka pro detekci překážek zpracováním obrazu z kamery.

Na závěr jsme popsali postup při testování systému na reálných datech, různé konfigurace testovacích tratí a funkcí. Uvádíme nejčastěji vznikající komplikace a způsoby jejich řešení. Načež proběhlo vyhodnocení výsledku celého projektu detekce překážek fúzí dat z LIDARu a kamery.

Mezi hlavní přínosy práce patří:

1. Byla tedy vyvinuta aplikace pro sběr, vizualizaci a vyhodnocení dat z LIDARu. Na základě této aplikace byly vytvořeny dvě funkce, z nichž první mapuje prostoru před vozítkem Jetracer pomocí LIDARu. Data získaná touto funkcí nám umožňují vidět prostorový řez okolí vozítka horizontální rovinou. Funkce je spolehlivá pro

prostory od přibližně 20 cm do 2 metrů. Mimo tento interval vznikají občasné chybné záznamy, nicméně většina bodů je stále zaznamenaná správně. Řešením pro zvýšení výkonosti a vyřešení těchto problému by mohl být systém ROS, což znamená Robot Operating System[2], který by mohl být součástí řešení navazujících projektů.

2. Byla vyvinuta funkce pro detekci překážek pomocí LIDARu. Vývoj této funkce byl takřka bezproblémový. Je to dáno tím, že funkce je založena na jednoduchém principu, což jí dělá velice robustní. Objekty jsou vždy v předem určeném rozmezí (vzdálenost a úhel) detekovány a svou funkci dobře plní i jako kontrolní detekce překážek v kombinaci kódu s detekcí objektů pomocí kamery.

Budoucím rozvojem popsaného systému by mohla být například funkce, která místo okamžitého zastavení před překážkou, bude pozvolně zpomalovat podle vzdálenosti k překážce. Toto by umožnilo plynulejší jízdu a více času na zpracování a vyhodnocení nadcházející situace. Dalším krokem by pak mohlo být objetí překážky, pokud by to bylo možné. Toto by již mohlo být provedeno s pomocí ROS softwaru, který by značně zjednodušil složité robotické funkce a dovolil snazší vývoj dalších funkcí vozítka bez nutností detailních znalostí komplexních fyzikálních a programovacích metod.

[2] Zdroj:http://wiki.ros.org/

Bibliografie

[1] SAE International. *SAE Levels of Driving Automation*. Květ. 2023. URL: https://www.sae.org/blog/sae-j3016-update.

[2] Synopsys. *The 6 Levels of Vehicle Autonomy Explained*. Čvc. 2023. URL: https://www.synopsys.com/automotive/autonomous-driving-levels.html?fbclid=IwAR23g-ErvF0ex5J1XqOYGiE-uIoosM0OJpEpl-bvyArKFtoeZTJhsPOeU-o.

[3] Alexander Schaub. „Sensors for Autonomous Vehicle“. In: *Robust Perception from Optical Sensors for Reactive Behaviors in Autonomous Robotic Vehicles*. 2017, s. 28–44. DOI: 10.1007/978-3-658-19087-3.

[4] John G. Webster Ramón PallásAreny a Ramón PallásAreny. „Ultrasonic-Based Sensors“. In: *Sensors and Signal Conditioning*. 2000, s. 538–544.

[5] Edward K. Reedy Jerry L. Eaves. „INTRODUCTION TO RADAR“. In: *PRINCIPLES OF MODERN RADAR*. 1987, s. 1–3. DOI: 10.1007/978-1-4613-1971-9.

[6] Admin. *Advantages And Disadvantages Of RADAR Systems*. 2022. URL: https://lidarradar.com/info/advantages-and-disadvantages-of-radar-systems.

[7] Emily Newton. *Radar for Autonomous Vehicles Could Be the Key to Self-Driving Cars*. 2022. URL: https://revolutionized.com/radar-for-autonomous-vehicles/.

[8] Babak Shahian Jahromi. *Camera Technology in Self-Driving Cars*. 2019. URL: https://medium.com/@BabakShah/camera-technology-in-self-driving-cars-610371db4b0b.

[9] Paul MCManamon. „Introduction“. In: *Field guide to Lidar*. 2015, s. 1–13.

[10] Qi Chen Pinliang Dong. „Principles of LIDAR Remote Sensing“. In: *LiDAR technologies and systems*. 2018, s. 19–26.

[11] Dubizzle Blog. *LIDAR System in Self-driving Cars*. 2023. URL: https://www.dubizzle.com/blog/cars/lidar-autonomous-driving/.

[12] Matt Domas Matthew Berry. *Advantages and Disadvantages of LiDAR Technology*. 2020. URL: https://flyguys.com/advantages-disadvantages-lidar-technology/.