FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

**Diplomová práce**

# Algoritmus pro autonomní podvozek realizující vyhýbání překážkám

*Studijní program:* N0714A270010 Mechatronika
*Autor práce:* **Bc. Martin Světlák**
*Vedoucí práce:* Ing. Lukáš Hubka, Ph.D.
Ústav mechatroniky a technické informatiky

Liberec 2024

FAKULTA MECHATRONIKY,
INFORMATIKY A MEZIOBOROVÝCH
STUDIÍ TUL

**Zadání diplomové práce**

# Algoritmus pro autonomní podvozek realizující vyhýbání překážkám

| | |
|---|---|
| *Jméno a příjmení:* | **Bc. Martin Světlák** |
| *Osobní číslo:* | M22000056 |
| *Studijní program:* | N0714A270010 Mechatronika |
| *Zadávající katedra:* | Ústav mechatroniky a technické informatiky |
| *Akademický rok:* | 2023/2024 |

**Zásady pro vypracování:**

1. Seznamte se se současnou podobou řídicího systému vozítka a jeho HW (UWB majáky) pro vyhýbání překážkám. Prostudujte implementovaný VFH+ algoritmus.
2. Rozšiřte detekční schopnost implementovaných lidarů o časový snímek tak, aby vznikla lokální mapa okolí, která omezí náraz na překážky, jež vypadnou ze zorného pole po přiblížení se k nim.
3. Ověřte schopnost vyhýbání se překážkám pomocí lokální mapy u objektů ustupujících ze zorného pole, případně navrhněte alternativní řešení.
4. Vyberte a implementujte alternativní algoritmus k VFH+, např. Smooth Nearness Diagram, Dynamic Window Approach, či jiný tak, aby se zlepšila schopnost jízdy v úzkých prostorech (např. chodbách).

| | |
|---|---|
| *Rozsah grafických prací:* | dle potřeby dokumentace |
| *Rozsah pracovní zprávy:* | 40 až 50 stran |
| *Forma zpracování práce:* | tištěná/elektronická |
| *Jazyk práce:* | čeština |

**Seznam odborné literatury:**


[1] BAE, Kyungbin; Yooha SON; Young-Eun SONG a Hoeryong JUNG. Component-Wise Error Correction Method for UWB-Based Localization in Target-Following Mobile Robot. online. Sensors, roč. 22 (2022), č. 3, s. 1180. Dostupné z: https://doi.org/10.3390/s22031180.

[2] ISLAM, Md Jahidul; Jungseok HONG a Junaed SATTAR. Person Following by Autonomous Robots: A Categorical Overview. online. arXiv. 2019-09-17. Dostupné z: https://doi.org/10.48550/arXiv.1803.08202. arXiv:1803.08202 [cs].

[3] ULRICH, I. a J. BORENSTEIN. VFH+: reliable obstacle avoidance for fast mobile robots. online. In: Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146). 5. 1998. Dostupné z: https://doi.org/10.1109/ROBOT.1998.677362.


| | |
|---|---|
| *Vedoucí práce:* | Ing. Lukáš Hubka, Ph.D.<br>Ústav mechatroniky a technické informatiky |
| *Datum zadání práce:* | 12. října 2023 |
| *Předpokládaný termín odevzdání:* | 14. května 2024 |

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
děkan

doc. Dr. Ing. Jaroslav Hlava
garant studijního programu

V Liberci dne 12. října 2023

# Prohlášení

Prohlašuji, že svou diplomovou práci jsem vypracoval samostatně jako původní dílo s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.
Jsem si vědom toho, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.
Beru na vědomí, že Technická univerzita v Liberci nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu Technické univerzity v Liberci.
Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti Technickou univerzitu v Liberci; v tomto případě má Technická univerzita v Liberci právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.
Současně čestně prohlašuji, že text elektronické podoby práce vložený do IS STAG se shoduje s textem tištěné podoby práce.
Beru na vědomí, že má diplomová práce bude zveřejněna Technickou univerzitou v Liberci v souladu s § 47b zákona č. 111/1998 Sb., o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších předpisů.
Jsem si vědom následků, které podle zákona o vysokých školách mohou vyplývat z porušení tohoto prohlášení.

14. 5. 2024 Bc. Martin Světlák

# Poděkování

Chtěl bych poděkovat svému vedoucímu Ing. Lukáši Hubkovi, Ph.D. za vstřícnost a pomoc při tvorbě této práce. Velké poděkování patří Ing. Danielu Kajzrovi za pomoc s praktickou částí práce. V neposlední řadě bych chtěl poděkovat své rodině a přátelům za neskonalou podporu.

# Algoritmus pro autonomní podvozek realizující vyhýbání překážkám

## Abstrakt

Tato práce se zabývá vylepšením řídicího algoritmu na čtyřkolové mobilní robotické platformě. Hlavním cílem je zvýšení autonomity vozidla vylepšením algoritmů pro jízdní režim Follow-me, který slouží k autonomnímu následování osob či jiných pohyblivých objektů.
Algoritmy jsou nejprve navrženy v prostředí Simulink a následně pomocí funkce generování kódu převedeny do průmyslového PC od firmy B&R. Pro vylepšení chování VFH+ algoritmu je navržen a testován vlastní algoritmus pro zaznamenání překážek v mrtvých úhlech lidarů na vozidle. Na vozidlo je aplikováno Ackermannovo řízení pro zlepšení odometrie. V rámci zlepšení lokální navigace v těsných prostorech je implementován a testován DWA algoritmus.

**Klíčová slova:** autonomní vozidlo, mobilní robotická platforma, Ackermannovo řízení, lokální navigace, lidar, mapa, mrtvé úhly, Simulink, Automation Studio, Target for Simulink

# Obstacle Avoidance Algorithm for Autonomous Chassis

## Abstract

This work focuses on improving the control algorithm for a four-wheeled mobile robotic platform. The main objective is to enhance the autonomy of the vehicle by improving the algorithms for the Follow-me driving mode, which is used to autonomously follow people or other moving objects.
The algorithms are initially designed in the Simulink environment and then converted to the industrial PC from B&R using code generation. To improve the behavior of the VFH+ algorithm, a custom algorithm for recording obstacles in the blind spots of lidars on the vehicle is designed and tested. Ackermann steering is applied to the vehicle to improve odometry. As part of enhancing local navigation in tight spaces, the DWA algorithm is implemented and tested.

**Keywords:** autonomous vehicle, mobile robotic platform, Ackermann steering, local navigation, lidar, map, blind spots, Simulink, Automation Studio, Target for Simulink

# Obsah

# Seznam zkratek

| | |
|---|---|
| **DC** | Direct current, stejnosměrný proud |
| **TWR** | Two-Way Ranging, dvou-cestné měření vzdálenosti |
| **UWB** | Ultra-wideband, ultra-širokopásmový |
| **PC** | Personal computer, osobní počítač |
| **VFH** | Vector Field Histogram, histogram vektorového pole |
| **DWA** | Dynamic window approach, přístup dynamického okénka |
| **SLAM** | Simultaneous Localization and Mapping, současná lokalizace a mapování |

# Seznam obrázků

# Úvod

Diplomová práce se zabývá vylepšením řídicího algoritmu na experimentálním autonomním vozidle s označením Generace 0. Toto vozidlo vzniklo a je provozováno na Technické univerzitě v Liberci. Jedná se o čtyřkolový Ackermannův podvozek s hnanou zadní nápravou a nezávisle natočitelnými předními koly. Podvozek je vybaven řídicím systémem a senzorikou, což mu umožňuje autonomní provoz v různých režimech.

Práce se zaměřuje na vylepšení jízdního režimu Follow-me, při kterém vozidlo následuje pomocí údajů z UWB systému člověka nesoucího UWB vysílač a autonomně se vyhýbá kolizním překážkám pomocí lidarů a algoritmů lokální navigace. Je navržen způsob, jak pokrýt mrtvé úhly lidarů na vozidle pomocí časového záznamu překážek a tvorby virtuálních lidarů. Toto řešení bylo navrženo pro kombinaci s již implementovaným VFH+ algoritmem z důvodu bočních kolizí s překážkami v mrtvém úhlu vozidla. Dalším cílem je vylepšení navigace vozidla v úzkých prostorech navržením alternativního algoritmu k VFH+. Splnění těchto cílů by mělo vést ke zvýšení autonomnosti robotické platformy.

V první kapitole je popsána konstrukce robotické platformy, včetně senzorické výbavy a použitého řídicího systému. V druhé kapitole je vysvětleno dosavadní algoritmizační řešení robotické platformy pro režim Follow-me, včetně VFH+ algoritmu pro lokální navigaci. V této kapitole jsou také zmíněny některé nedostatky dosavadního řešení a důvody pro jeho vylepšení. Ve třetí kapitole je rozebráno a odvozeno Ackermannovo řízení, které bylo aplikované na vozítko. Ve čtvrté kapitole je popsán návrh vlastního algoritmu pro tvorbu časového snímku z lidarů. V páté kapitole je navrženo řešení pro filtraci následované osoby z dat lidaru. Šestá kapitola se zabývá návrhem DWA algoritmu jako alternativy k již použitému VFH+. V sedmé kapitole je popsána implementace navržených algoritmů na robotickou platformu a jsou v ní vyhodnoceny výsledky.

# 1 Robotický podvozek

Experimentální vozidlo, na které se tato práce zaměřuje, bylo vytvořeno v rámci projektu Modulární platforma pro autonomní podvozky specializovaných elektrovozidel pro dopravu nákladu a zařízení na Technické univerzitě v Liberci. Jedná se o modulární elektrickou platformu s označením Generace 0 [1].

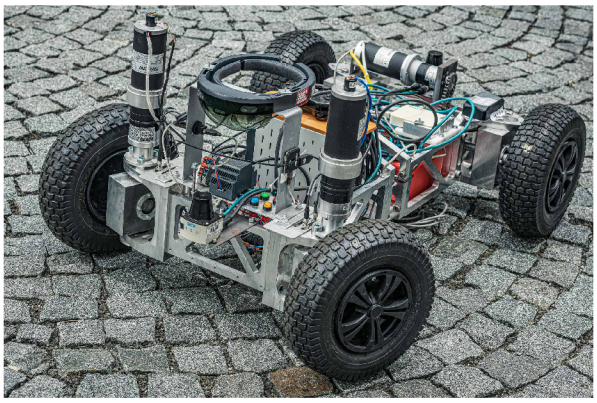
## 1.1 Konstrukce

Vozidlo je konstrukčně řešeno jako čtyřkolová mobilní robotická platforma s poháněnými zadními koly a s natočitelnými předními koly. Toto konstrukční řešení nabízí velmi dobrou stabilitu, ale má omezenou manévrovatelnost danou pouze úhlem natočení předních kol.

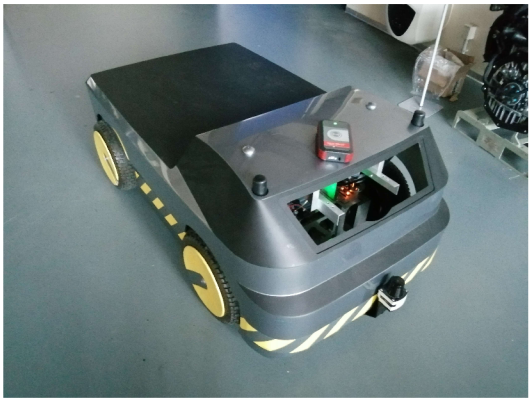
Vozidlo je vybaveno celkově třemi DC motory typu EC070.24E od firmy Transtecno [2]. Zadní kola jsou poháněna jedním společným motorem, který umožňuje pohyb v obou směrech s proměnnými rychlostmi. Moment z motoru je přenášen na hřídel kol pomocí řetězového převodu a ozubených kol.

Přední kola jsou natáčena nezávisle na sobě pomocí dvou servomotorů. To umožňuje softwarovou aplikaci různých řídících geometrií, jako je například Ackermannovo řízení. Nezávisle řízená přední kola mohou pomoci eliminovat prokluz kol a zlepšit tak odometrii robotického systému. Dosud však nebyla na podvozku implementována žádná řídicí geometrie.

Vozidlo měří na délku 120 cm, rozvor kol je 77 cm a rozchod kol 67 cm. Podvozek je osazen koly s pneumatikami s označením 13×5-6. Celé vozidlo včetně senzoriky a kompozitního těla váží 70 kg [3]. Vozidlo je vidět na obrázku 1.1.



(a) Kostra vozidla s komponenty [1]



(b) Vozidlo s kompozitním krytem

Obrázek 1.1: Experimentální vozidlo Generace 0

## 1.2 Senzorika

### Lidary

Vozidlo je vybaveno dvěma lidary, na kterých je založen systém detekce a vyhýbání se překážkám, případně nouzové zastavení při přiblížení k překážce. Jedná se o lidary TIM 781 od značky Sick na obrázku 1.2. Technické parametry lidarů jsou v tabulce 1.1. Kvůli umístění lidarů a konstrukčnímu řešení vozítka dochází k omezení úhlu použitelného zorného pole ze $\pm 135°$ na $\pm 90°$ [3]. Tím vznikají problémy VFH+ algoritmu s mrtvými úhly, kdy lidar není schopen zaznamenat překážky nacházející se vedle vozidla a to může mít za následek kolize s těmito překážkami. Lidar umístěný vpředu vozidla slouží jako vstup pro algoritmy lokální navigace a nouzové zastavení. Zadní lidar slouží pro nouzové zastavení při couvání.

| Lidar TIM 781 | |
|---|---|
| Minimální snímací vzdálenost | 0,05 m |
| Maximální snímací vzdálenost | 25 m |
| Úhel zorného pole | $\pm$ 135° |
| Úhlové rozlišení | 1/3° |
| Frekvence snímání | 15 Hz |

Tabulka 1.1: Tabulka parametrů lidaru TIM 781 [4]

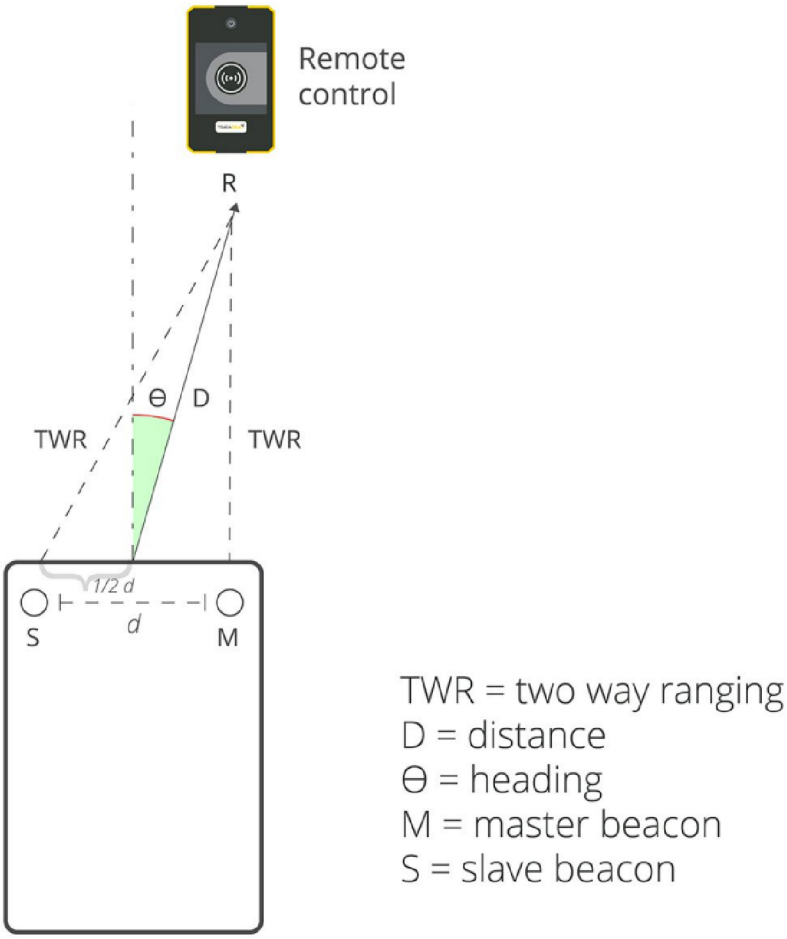

Obrázek 1.2: Lidar TIM 781 [4]

### Ultra-wideband systém

Pro algoritmus Follow-me jsou na předních rozích vozidla umístěny UWB majáky (master a slave), které snímají signál od dálkového UWB ovladače. Ten může být buď nesen člověkem nebo následovaným objektem. Pokud uživatel stiskne tlačítko na dálkovém ovladači, aktivuje se dvoucestné měření vzdálenosti (TWR), jehož princip je znázorněn na obrázku 1.3. TWR technologie měří vzdálenosti mezi ovladačem a dvěma majáky a na základě těchto vzdáleností spočítá polohu ovladače vůči středu mezi majáky a ose vozidla. Výstupem z TWR je vzdálenost a úhel vychýlení popisující polohu ovladače vůči vozidlu [5]. Hlavními výhodami využití UWB systému pro následování objektů oproti využití kamer a rozpoznání obrazu je, že UWB systém nevyžaduje přímou viditelnost objektu. Dalšími výhodami je nízká výpočetní náročnost zpracování dat a malé ovlivnění přesnosti vnějšími vlivy [3]. Parametry UWB systému jsou v tabulce 1.2.

| TB-FMS-RS485 | |
|---|---|
| Technologie pro měření vzdálenosti | Ultra-wideband |
| Snímací vzdálenost | 0,5 m až 60 m |
| Úhel měření | ±50° |
| Přesnost | ±25 cm a ±19° |
| Opakovatelnost | ±5 cm a ±10° |

Tabulka 1.2: Tabulka parametrů Follow-Me systému [6]




Obrázek 1.3: Princip TWR [5]

### Microsoft HoloLens

Vozidlo využívá pro určité funkce brýle Microsoft HoloLens. Microsoft HoloLens slouží primárně jako zařízení pro rozšířenou realitu, je možné ho ale využít i jako inertiální měřící jednotku (IMU), která kombinuje akcelerometr, gyroskop a magnetometr, případně využít funkcí hloubkové kamery a HD kamery. Díky HoloLens je možné získávat informace o pozici vozidla relativně vůči místu zapnutí softwaru. Tyto informace mohou být využity řídicím systémem vozidla, případně pomocí Wi-Fi sítě poslány do nadřazeného kontrolního systému [3]. Umístění HoloLens na vozidle je vidět na obrázku 1.1.

## 1.3 Řídicí systém

Pro řízení vozidla je využito průmyslové PC Automation PC 910 od B&R. Jedná se o modulární průmyslové PC, jehož hlavní výhodou je velký výpočetní výkon, který umožňuje rychlé zpracování dat a řízení vozidla v reálném čase [7]. Dalším z důvodů pro využití tohoto PC je podpora generování kódu z MALTAB/Simulinku pro B&R systémy pomocí Automation Studio Target for Simulink. Tento software umožňuje nejprve navrhnout program v rámci prostředí Simulink, což výrazně usnadňuje programování složitějších úloh, a následně program generovat ve formě kódu (strukturovaného textu) do programu Automation Studio [8]. Řídicí systém a popsaná senzorika umožňují vozidlu pracovat ve dvou základních implementovaných režimech.

Prvním režimem je autonomní navigace z bodu A do bodu B s prediktivním regulátorem na základě předem definované trajektorie. V tomto režimu je jako senzor primárně využito zařízení HoloLens pro získání informace o pozici vozidla. I v případě, že není informací z HoloLens využito pro navigaci vozidla, jsou tyto informace posílány na Ante Web pro monitorování výsledků [3]. Tímto režimem se nebude tato práce zabývat.

Druhým režimem, na který se tato práce primárně zaměřuje, je následování osob nebo objektů pomocí Follow-Me algoritmu. Zde je hlavním senzorem UWB sledovací systém a pro detekci a vyhýbání se překážkám jsou využity lidary v kombinaci s VFH+ algoritmem. Tento režim by měl sloužit k následování osob, nebo jiných pohyblivých objektů, za účelem převážení těžkých břemen, případně kvůli jinému druhu asistence osob [1, 3].

# 2 Aktuální algoritmizace robotické platformy

## 2.1 Follow-Me algoritmus

Výstupem z UWB systému je poloha dálkového ovladače vůči vozidlu popsaná vzdáleností a úhlem vychýlení. Jedná se tedy o úlohu zpětnovazebního řízení s regulovanými veličinami v podobě vzdálenosti a úhlu vychýlení dálkového ovladače a dvěma akčními veličinami v podobě rychlosti a úhlu natočení kol. Cílem je udržet vzdálenost vozidla od UWB ovladače na požadované vzdálenosti a úhel vychýlení na nule.

Jednou z možností pro tvorbu formací mobilních robotů je využití virtuální pružiny s tlumičem [9]. Přístup se dá popsat pomocí virtuální síly působící na mobilní platformu v závislosti na vzdálenosti mezi platformou a následovaným objektem. Tuto sílu popisuje rovnice

$$F(t) = k_R \cdot (d_0 - d(t)) + c_R \cdot \frac{d}{dt}(d_0 - d(t)) = k_R \cdot e(t) + c_R \cdot \frac{de(t)}{dt},$$

kde $k_R$ je tuhost virtuální pružiny, $c_R$ je koeficient tlumení virtuálního tlumiče, $d(t)$ je délka pružiny a tedy vzdálenost platformy od objektu a $d_0$ je klidová délka pružiny, neboli požadovaná vzdálenost od objektu. Z této rovnice plyn,e že se jedná o regulátor s proporcionálně-derivačním charakterem. Při využití druhého Newtonova zákona $F(t) = m \cdot \frac{dv(t)}{dt}$ může být silová rovnice pružiny vyjádřena jako proporcionálně-integrační regulátor s rychlostí na výstupu. Na základě toho byl navržen číslicový PI regulátor pro řízení vzdálenosti od cíle na žádanou hodnotu pomocí rychlosti vozidla, který byl však později rozšířen na PID strukturu [3].

Proporcionální člen je realizován jako násobení regulační odchylky konstantou. Regulační odchylka je daná vztahem

$$e(k) = w(k) - y(k),$$

kde $w(k)$ je žádaná vzdálenost od osoby a $y(k)$ je aktuální vzdálenost. Při žádaném pohybu dopředu je tedy $e(k)$ záporná. Aby byl akční zásah kladný, musí být činitel zesílení záporný. Proporcionální složka je zapsána ve tvaru

$$P = r_0 \cdot e(k),$$

kde $r_0$ je činitel zesílení.

Integrační složka je realizována lichoběžníkovou metodou v podobě rekurzivní diferenční rovnice, která je ve tvaru

$$I(k) = I(k-1) + \frac{r_0 \cdot T_v}{2 \cdot T_i}(e(k) + e(k-1)),$$

kde $T_v$ je vzorkovací frekvence a $T_i$ je integrační časová konstanta. Aby nedocházelo při regulačním pochodu ke vzniku Wind-up efektu, je v případě překročení maximální hodnoty akční veličiny použita předchozí hodnota integrační složky. Akční veličina, neboli rychlost vozidla, je omezena na hodnoty $\pm$0,75 m/s.

Derivační složka je realizována jako filtrovaná derivace ve tvaru

$$D(s) = \frac{r_0 \cdot T_d \cdot s}{1 + s \cdot \alpha \cdot T_d}$$

a následně převedená pomocí Tustinovy aproximace na diferenční rovnici ve tvaru

$$d(k) = \frac{2 \cdot r_0 \cdot T_d}{T_v + 2 \cdot T_d \cdot \alpha}(e(k) - e(k-1)) + \frac{2 \cdot T_d \cdot \alpha - T_v}{T_v + 2 \cdot T_d \cdot \alpha} d(k-1),$$

kde $T_d$ je derivační časová konstanta a $\alpha$ je parametr filtru. Výsledný akční zásah PID regulátoru je součtem všech složek. Změna akčního zásahu je omezena maximální hodnotou přírůstku za jeden cyklus. Podrobnější informace o výše zmíněných číslicových regulátorech jsou v [10].

Takto navržený regulátor umožňuje vozidlu následovat osobu s UWB ovladačem jízdou dopředu, ale v případě, že se k ovladači přiblíží na vzdálenost menší než je žádaná, tak i couvat směrem od osoby. Aby nedocházelo ke kmitání vozidla v případě, že se pohybuje na hraně žádané vzdálenosti, musí být reakce regulátoru omezena určitým pásmem. V tomto případě vozidlo stojí, pokud je absolutní hodnota regulační odchylky menší, než velikost pásma [3].

Pro řízení úhlu natočení kol vozidla je využito jednoduchého proporcionální zesílení regulační odchylky. Regulační odchylka může být buďto úhlovým výstupem z UWB systému, nebo výstupem z VFH+ algoritmu, pokud je vyhodnocena nutná změna směru kvůli překážce. Výstupní úhel z UWB systému je filtrován průměrem kvůli velké fluktuaci hodnot.

Tento jízdní režim je doplněn o funkci nouzového zastavení v případě, že se vozidlo přiblíží k překážce na vzdálenost menší, než je vzdálenost při které vozidlo zvládne z maximální rychlosti zastavit za jednu sekundu. Zvažované překážky pro tuto funkci jsou voleny z výseče lidarových dat $\pm$50° v předu i v zadu. Následně je z této výseče nalezeno minimum, které slouží jako údaj o nejbližší překážce před vozidlem.

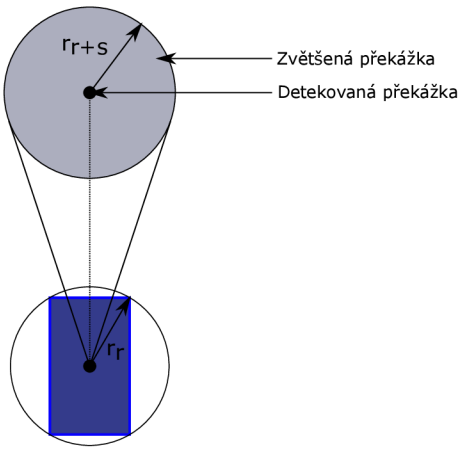
## 2.2 VFH+

Protože v režimu Follow-Me se může stát, že se následovaný objekt dostane za překážku, je nutné doplnit řízení vozidla i o algoritmus pro objíždění překážek. V současném stavu je pro tento účel na vozidle využíván VFH+ algoritmus, který je již implementován v Matlab/Simulinku.

VFH+ algoritmus je založen na původním VFH algoritmu. VFH algoritmus byl vyvinut v roce 1991 pro vyhýbání se překážkám v reálném čase [11]. VFH+ původní algoritmus vylepšuje a doplňuje o funkce, které vylepšují jeho spolehlivost a vyhlazují trajektorii [12].

Vstupem do VFH+ algoritmu je lokální mapa, v tomto případě tvořená snímkem z lidaru. Algoritmus má čtyři hlavní kroky, které vedou ke zvolení nové volné trajektorie.

Prvním krokem je vytvoření tzv. základního polárního histogramu. Každá aktivní oblast, neboli každá detekce lidaru, je zpracována jako náležící vektor daný úhlem a vzdáleností. VFH+ se oproti původnímu VFH liší tím, že bere v potaz rozměry robota. To je provedeno dilatací jednotlivých detekovaných překážek o $r_{r+s} = r_r + d_s$, kde $r_r$ je poloměr robota a $d_s$ je minimální bezpečná vzdálenost mezi robotem a překážkou. Proces je znázorněn na obrázku 2.1. Dilatace překážek umožňuje uvažovat robota jako bod s nulovým rozměrem. Následně je zkonstruován polární histogram s definovaným rozlišením, který udává míru jistoty existence překážky v daném sektoru.




Obrázek 2.1: Příklad dilatace překážky

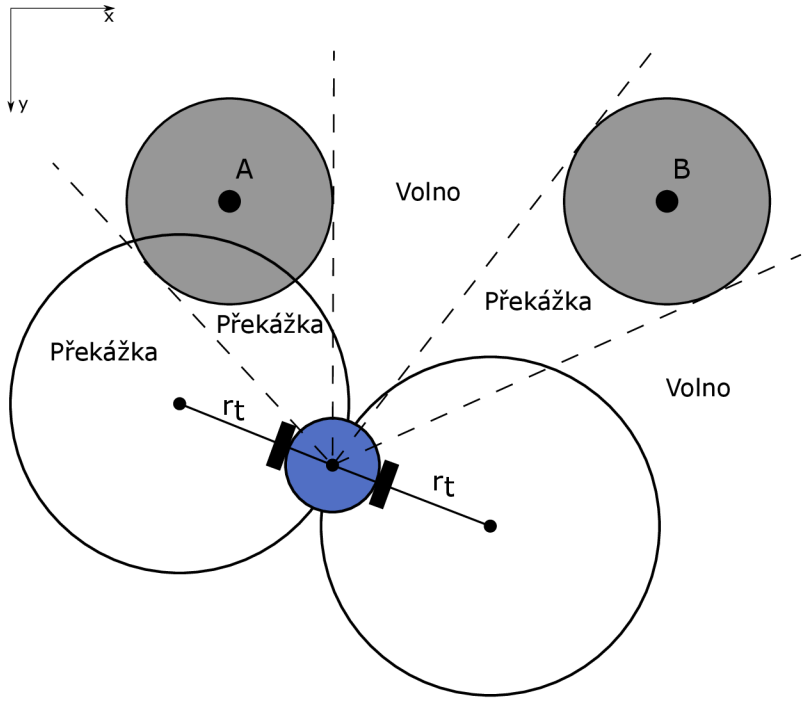
V druhém kroku je v rámci eliminace oscilací a nekontrolovatelného chování, například v situacích, kdy je velikost průchozího prostředí podobná velikosti robotu, vytvořen na základně zvoleného hysterezního pásma binární polární histogram. Převod ze základního histogramu na binární je v čase n

proveden dle následujících pravidel

$$H^p_{k,n} > \tau_{high} \rightarrow H^b_{k,n} = 1$$
$$H^p_{k,n} < \tau_{low} \rightarrow H^b_{k,n} = 0$$
$$jinak : H^b_{k,n} = H^b_{k,n-1},$$

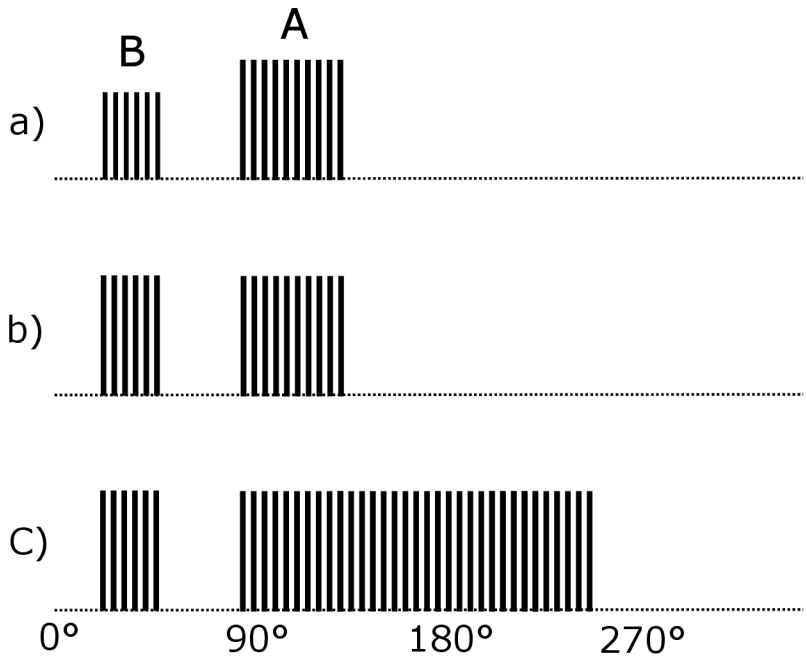
kde $\tau_{high}$ je vrchní práh hystereze a $\tau_{low}$ je spodní práh hystereze. Hodnota 0 binárního histogramu pak představuje volný prostor a hodnota 1 představuje obsazený prostor.

Ve třetím kroku je vytvořen tzv. maskovaný polární histogram, který uvažuje i limitace robota dané minimálním poloměrem otáčení. Situace je naznačena na obrázku 2.2.




Obrázek 2.2: Příklad zablokovaných směrů pohybu [12]

Na obrázku 2.3 jsou znázorněny jednotlivé polární histogramy. Je vidět, že binární histogram indikuje směry vlevo od překážky A jako průchodné, i když to kinematická omezení robota neumožňují. Maskovaný polární histogram tuto chybu eliminuje rozšířením úhlu oblasti překážky.




Obrázek 2.3: a) Základní polární histogram, b) Binární polární histogram, c) Maskovaný polární histogram [12]

V posledním kroku je ze všech volných trajektorií určených z maskovaného polárního histogramu zvolena na základě váhové funkce nejoptimálnější trajektorie, neboli trajektorie s nejmenší hodnotou váhové funkce. Volba trajektorie se dá ovlivnit pomocí váhových koeficientů $\mu_1$, $\mu_1$ a $\mu_3$. Koeficient $\mu_1$ udává, jak moc se ve volbě trajektorie preferuje směr k cíli, $\mu_2$ udává míru efektivity trajektorie s minimální změnou směru pohybu a $\mu_3$ udává snahu o zachování předchozího směru pohybu [12].

## 2.3 Nevýhody a důvody k vylepšení

Vozítko má v současném stavu několik nedostatků, které limitují jeho optimální funkčnost.

Prvním z nedostatků je výše zmíněné konstrukční provedení vozítka společně s umístěním lidarů, což limituje jeho zorné pole. To způsobuje, že vozítko může v některých případech úspěšně objíždět překážku díky VFH+ algoritmu, ale ve chvíli, kdy se překážka dostane mimo zorné pole do ní narazí bokem.

Další nedostatek způsobuje použití výše popsaného VFH+ algoritmu, který nemá nejlepší vlastnosti při projíždění těsnými prostory, jako jsou například úzké chodby. Při průjezdu chodbou může dojít k situaci, kdy algoritmus vyhodnotí, že nelze dodržet bezpečnou vzdálenost, případně manévrovat tak aby prostorem projel.

Dalším problémem je, že VFH+ algoritmus detekuje následovanou osobu (objekt) jako překážku. To někdy vede k nežádoucím změnám směru.

# 3 Algoritmus pro detekci překážek v mrtvých úhlech lidarů

Jelikož na konstrukci vozítka již není možné dělat velké změny, nabízí se řešení pro pokrytí mrtvých úhlů lidarů vytvořením lokální mapy na základě předchozích hodnot z lidarů. K vytvoření lokální mapy bylo zváženo několik možností, nakonec bylo ale s ohledem na již implementovaný VFH+ algoritmus navrženo vlastní řešení.

## 3.1 SLAM

Při vývoji byla nejprve zvažována možnost využití SLAM algoritmů, které by bylo možné efektivně implementovat pomocí toolboxů v Matlab/Simulinku. SLAM algoritmy umožňují současné mapování okolí, aktualizaci mapy a lokalizaci vozidla ve zmapovaném prostoru. K tomu je využita odometrie vozidla v kombinaci se zaznamenanými daty ze senzorů [13]. Takto vytvořená mapa okolí by po úpravách a převedení do binární mapy obsazenosti mohla sloužit s využitím odhadnuté pozice ke zpětnému převedení do bodů lidaru sloužících jako vstup pro VFH+ algoritmus.

Nevýhodou SLAM algoritmů je jejich výpočetní náročnost, která by v kombinaci se zpracováním a následným převedením mapy zpět na hodnoty lidarů mohla výrazně zpomalit výpočetní cyklus průmyslového PC. Další problém by mohl nastat při akumulaci chyb v odometrii a v lokalizaci, kdy by výsledná data lidaru nemusela odpovídat aktuálnímu stavu okolí. K tomu se váže i nedokonalost v mapování, kde především v začátcích tvorby mapy mohou vznikat falešné překážky, nebo jsou překážky posunuty vůči realitě [13]. Další omezení je dané malou manévrovatelností vozidla, díky které by bylo mapování menších prostorů náročné.

Aplikovat SLAM algoritmy by bylo vhodné pro provoz vozidla ve stále stejném prostředí, které by bylo možné předem zmapovat a mapu zpracovat tak, aby nevykazovala výše zmíněné chyby. Pro současný stav a účel vozidla by využití SLAM algoritmů nebylo příliš efektivní kvůli výše zmíněným limitacím a byl proto navržen vlastní algoritmus.

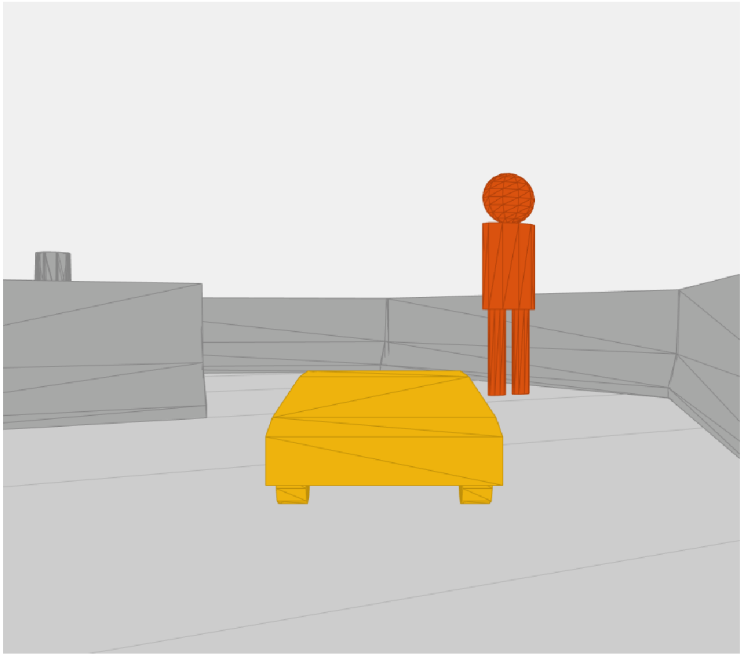
## 3.2 Návrh algoritmu pro tvorbu mapy

Hlavním kritériem pro návrh algoritmu bylo zachovat stávající strukturu s VFH+ algoritmem. K tomu bylo nutné vytvořit lokální mapu ve formě hodnot lidaru, které slouží jako vstup do implementovaného VFH+. Navržený algoritmus se dá rozdělit na následující kroky.

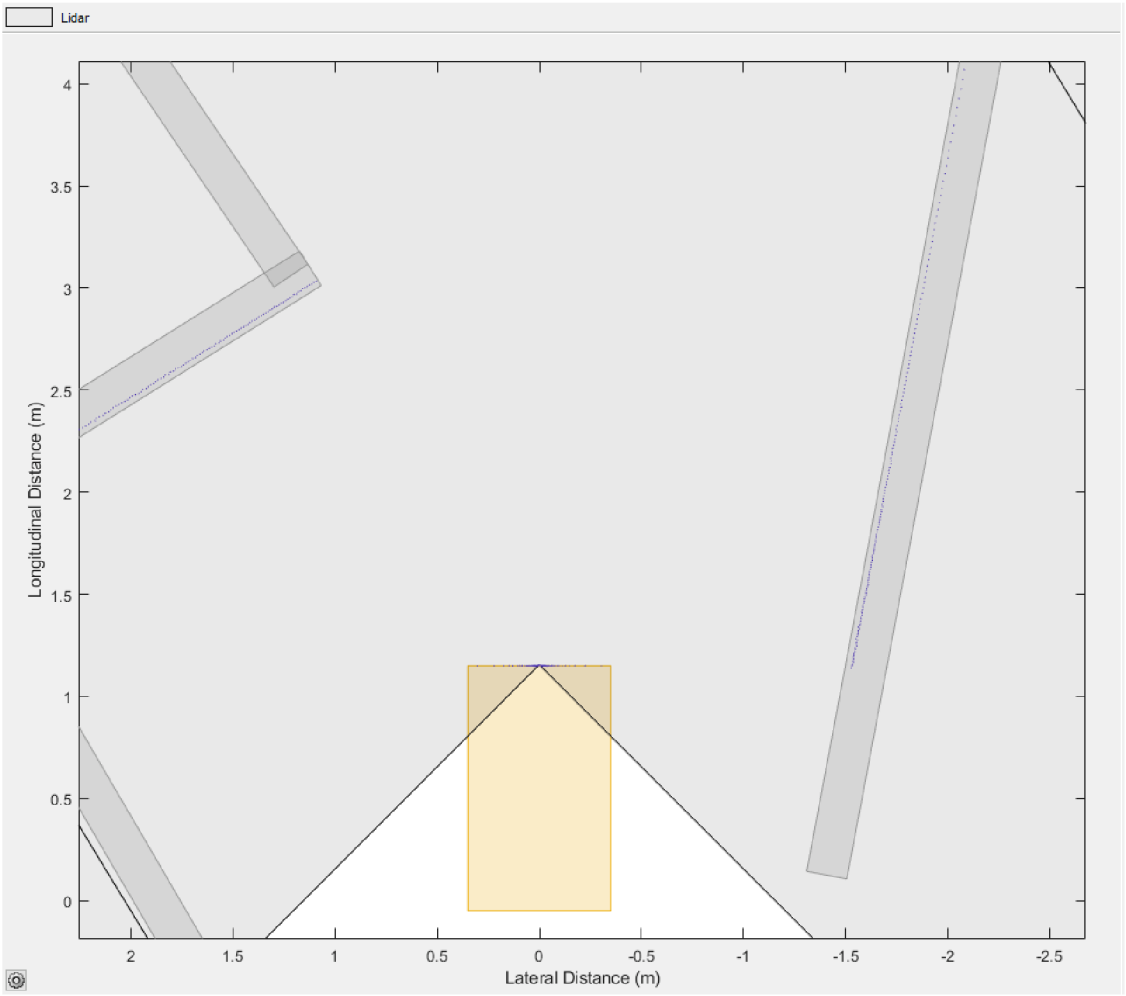
### Zpracování dat lidaru

Prvním krokem je zpracování naměřených hodnot z lidaru. Výstupem z lidaru je pole o 811 hodnotách, z nichž každá představuje vzdálenost naměřenou kladně proti směru hodinových ručiček s úhlovým rozlišením $\frac{1}{3}°$. Pole hodnot je nejprve oříznuto tak, aby v něm nebyly obsaženy hodnoty, kde paprsky lidaru kolidují s konstrukcí vozidla. To kvůli konstrukčnímu řešení vozidla odpovídá rozsahu $\pm 90°$. Omezení díky umístění lidaru jsou vidět na obrázku 3.2, kde část rozsahu přímo zaznamenává konstrukci vozidla. Obrázek pochází z Matlab aplikace Driving Scenario Designer, která umožňuje generovat výstupy lidarů na základě vytvořené simulace [14]. Takto generované hodnoty byly použity pro tvorbu navrženého algoritmu.

Pokud se paprsek lidaru neodrazí od překážky, lidar vrátí nulovou nebo velmi malou hodnotu. Tyto hodnoty jsou nahrazeny za hodnotu maximálního dosahu lidaru, aby nedocházelo k chybné funkci VFH+ algoritmu. V neposlední řadě jsou naměřené vzdálenosti přepočítány na metry.



Obrázek 3.1: Driving Scenario Designer – 3D pohled simulace




Obrázek 3.2: Driving Scenario Designer – Zobrazení bodů lidaru

### Zaznamenání překážek do bufferů

V dalším kroku jsou vytvořeny čtyři buffery, které jsou použity jako FIFO fronty pro ukládání krajních hodnot předního lidaru v podobě souřadnic bodů v kartézském souřadném systému. Pro každou stranu je využita dvojice bufferů pro ukládání souřadnic v osách $x$ a $y$. Tyto souřadnice jsou průměrem dvou krajních hodnot lidaru, aby se omezil vliv chybných měření. Hodnoty se posouvají s každým cyklem průmyslového PC. Délka bufferu se odvíjí od rychlosti vozidla a zvolené doby cyklu průmyslového PC a je nutné ji nastavit tak, aby překážky zůstávaly uloženy minimálně dokud je vozidlo nemine celou délkou. Dalším kritériem pro volbu délky je výpočetní náročnost. Pozice zaznamenaných bodů z předchozího cyklu jsou transformovány na základě změny úhlu směru vozidla a uražené vzdálenosti vždy do nového souřadného systému na nové pozici vozidla. Úhel vozidla a uražená vzdálenost lze popsat pomocí kinematických rovnic robotické platformy. Kinematické rov-

nice pro Ackermannův podvozek v maticovém tvaru lze zapsat jako

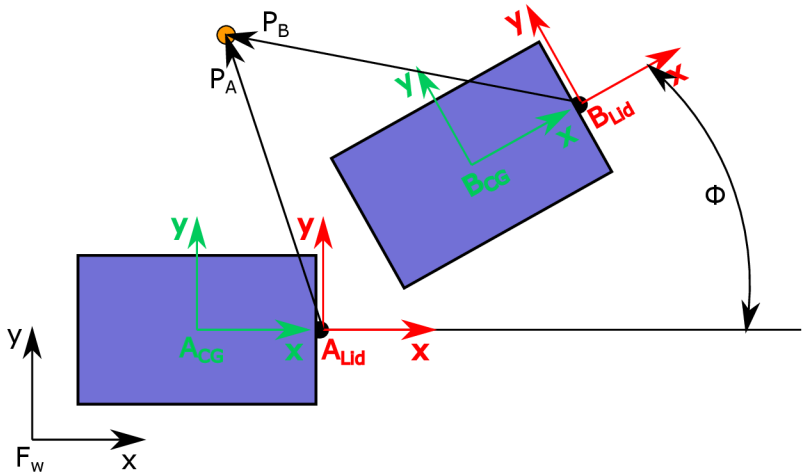
$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \cos\phi & 0 \\ \sin\phi & 0 \\ \frac{\tan\psi}{l} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \dot{\psi} \end{bmatrix},$$

kde $x$ a $y$ jsou polohy v souřadném systému, $\phi$ je úhel směru vozidla, $\psi$ je úhel natočení kol, $l$ je rozvor vozidla a $v$ je rychlost vozidla [15].

Vzhledem k délce cyklu je možné brát rychlost a úhlovou rychlost vozidla jako po částech konstantní. Derivaci úhlu natočení kol není nutné brát v potaz, protože výstupem regulačního pochodu je přímo úhel natočení kol. Díky tomu lze zapsat rovnice pro polohy v souřadném systému a úhel změny směru vozidla jako

$$\begin{aligned} \phi &= \frac{\tan\psi}{l} \cdot v \cdot Tv \\ x &= \cos\phi \cdot v \cdot Tv \\ y &= \sin\phi \cdot v \cdot Tv, \end{aligned}$$

kde $Tv$ je doba trvání cyklu. Takto zjednodušené rovnice umožňují zjistit uraženou vzdálenost v obou osách a změnu úhlu směru vozidla za jeden cyklus na základě předchozích hodnot rychlosti a úhlu natočení kol. Tyto hodnoty translace a rotace slouží pro přepočet mezi souřadnými systémy.




Obrázek 3.3: Souřadné systémy při pohybu vozidla

Souřadnice bodů lidaru uložené v bufferech jsou následně přepočítány pomocí transformační matice. Pokud uvažujeme situaci na obrázku 3.3, kde se vozidlo pohybuje ve 2D souřadném systému, je možné přepočet popsat jako

$${}^{A}\mathbf{P}_{(3\times1)} = {}^{A}_{B}\mathbf{T}_{(3\times3)} \cdot {}^{B}\mathbf{P}_{(3\times1)},$$

kde ${}^{A}\mathbf{P}$ je vektor souřadnic bodu v souřadném systému A doplněný o jedničku na poslední pozici, ${}^{B}\mathbf{P}$ je vektor souřadnic bodu v souřadném systému B doplněný o jedničku na poslední pozici a ${}^{A}_{B}\mathbf{T}$ je transformační matice ze souřadného systému B do souřadného sytému A. Transformační matici složenou z rotace **R** a translace **P** lze popsat vztahem

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{P} \\ \mathbf{0} & 1 \end{bmatrix},$$

kde

$$\mathbf{R} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}, \mathbf{P} = \begin{bmatrix} x \\ y \end{bmatrix}, \mathbf{0} = \begin{bmatrix} 0 & 0 \end{bmatrix}.$$

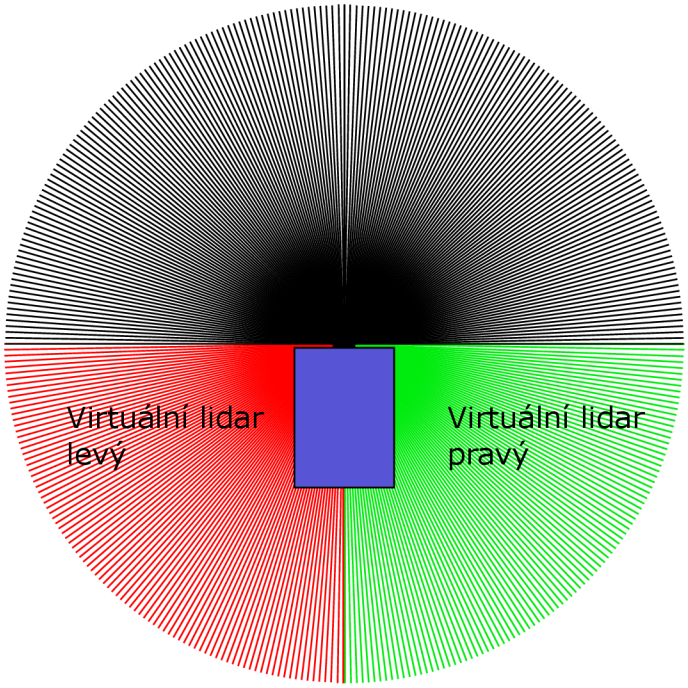
Protože je známá pozice bodů v předchozím kroku, neboli v souřadném systému A, je nutné provést přepočet do aktuálního souřadného systému B pomocí inverzní transformační matice. Výsledný vztah je potom

$${}^{B}\mathbf{P} = {}^{A}_{B}\mathbf{T}^{-1} \cdot {}^{A}\mathbf{P}.$$

Takto přepočtené body tvoří časový snímek překážek, které by jinak byly již mimo zorné pole lidaru. Posun bodů v bufferu je proveden pouze pokud je vozidlo v pohybu.

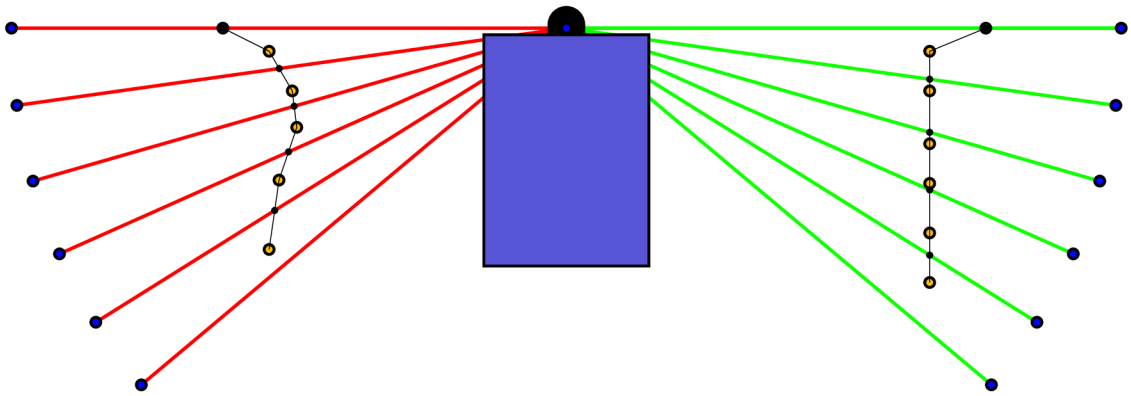
### Tvorba virtuálních lidarů

Protože VFH+ algoritmus pracuje s hodnotami lidaru a nikoliv s mapou je dalším krokem vytvoření virtuálních lidarů, které pokrývají požadovaný prostor kolem vozidla. Pro řešenou úlohu byly vytvořeny dva virtuální lidary s úhlovým rozsahem 90°, úhlovým rozlišením 1° a dosahem 25 m. Rozmístění lidarů je znázorněno na obrázku 3.4.




Obrázek 3.4: Rozmístění virtuálních lidarů na vozidle

Pro každý paprsek virtuálních lidarů jsou v pomocném poli uloženy souřadnice reprezentující jeho koncový bod v souřadném systému lidaru s tím, že počáteční bod je v nule souřadného systému. Mezi úsečkami tvořenými dvojicemi bodů, které definují paprsky lidaru, a úsečkami z po sobě jdoucích bodů zaznamenaných překážek jsou následně spočteny průsečíky. Situace je ve zjednodušené verzi znázorněna na obrázku 3.5.



Obrázek 3.5: Vzniklé průsečíky paprsků lidarů s překážkou

Průsečíky je nutné hledat pouze na segmentech křivek vymezených krajními body. Pro nalezení průsečíků jsou nejprve definovány úsečky $L_1$ a $L_2$ jako Beziérovy křivky prvního stupně

$$L_1 = \begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + t \begin{bmatrix} x_2 - x_1 \\ y_2 - y_1 \end{bmatrix}, \; L_2 = \begin{bmatrix} x_3 \\ y_3 \end{bmatrix} + u \begin{bmatrix} x_4 - x_3 \\ y_4 - y_3 \end{bmatrix},$$

kde $t$ a $u$ jsou reálná čísla. Průsečík dvou úseček je nalezen pomocí hodnot parametrů $t$ a $u$, kde

$$t = \frac{\begin{vmatrix} x_1 - x_3 & x_3 - x_4 \\ y_1 - y_3 & y_3 - y_4 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & x_3 - x_4 \\ y_1 - y_2 & y_3 - y_4 \end{vmatrix}} = \frac{(x_1 - x_3)(y_3 - y_4) - (y_1 - y_3)(x_3 - x_4)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)}$$

a

$$u = \frac{\begin{vmatrix} x_1 - x_2 & x_1 - x_3 \\ y_1 - y_2 & y_1 - y_3 \end{vmatrix}}{\begin{vmatrix} x_1 - x_2 & x_3 - x_4 \\ y_1 - y_2 & y_3 - y_4 \end{vmatrix}} = \frac{(x_1 - x_2)(y_1 - y_3) - (y_1 - y_2)(x_1 - x_3)}{(x_1 - x_2)(y_3 - y_4) - (y_1 - y_2)(x_3 - x_4)}.$$

Souřadnice průsečíku je pak možné zjistit jako

$$(P_x, P_y) = (x_1 + t(x_2 - x_1), y_1 + t(y_2 - y_1)),$$

nebo

$$(P_x, P_y) = (x_3 + u(x_4 - x_3), y_3 + u(y_4 - y_3)),$$

kde průsečík existuje pokud platí, že $0 \leq t \leq 1$ a zároveň $0 \leq u \leq 1$ [16].

Algoritmus postupně prohledává průsečíky paprsků lidarů s uloženými překážkami a nalezené průsečíky jsou následně přepočteny na vzdálenost od lidaru. Paprsky, které neprotnou překážku, mají hodnotu maximálního dosahu. Tyto hodnoty jsou spojeny s hodnotami předního lidaru a vzniká tak lidarové pokrytí okolí vozidla s rozsahem 360°.

Protože VFH+ algoritmus pracuje s hodnotami lidaru vztaženými k souřadnému systému těžiště vozidla, viz souřadný systém $A_{CG}$ na obrázku 3.3, je nutné hodnoty lidarů přepočítat. To je provedeno pomocí funkce transformScan z Navigation Toolboxu v Matlabu [17].

Na konci průchodu programem je do persistentních proměnných uložena aktuální rychlost a úhel natočení, které jsou využity v dalším cyklu jako předchozí hodnota rychlosti a úhlové rychlosti pro výpočet transformace souřadných systémů.

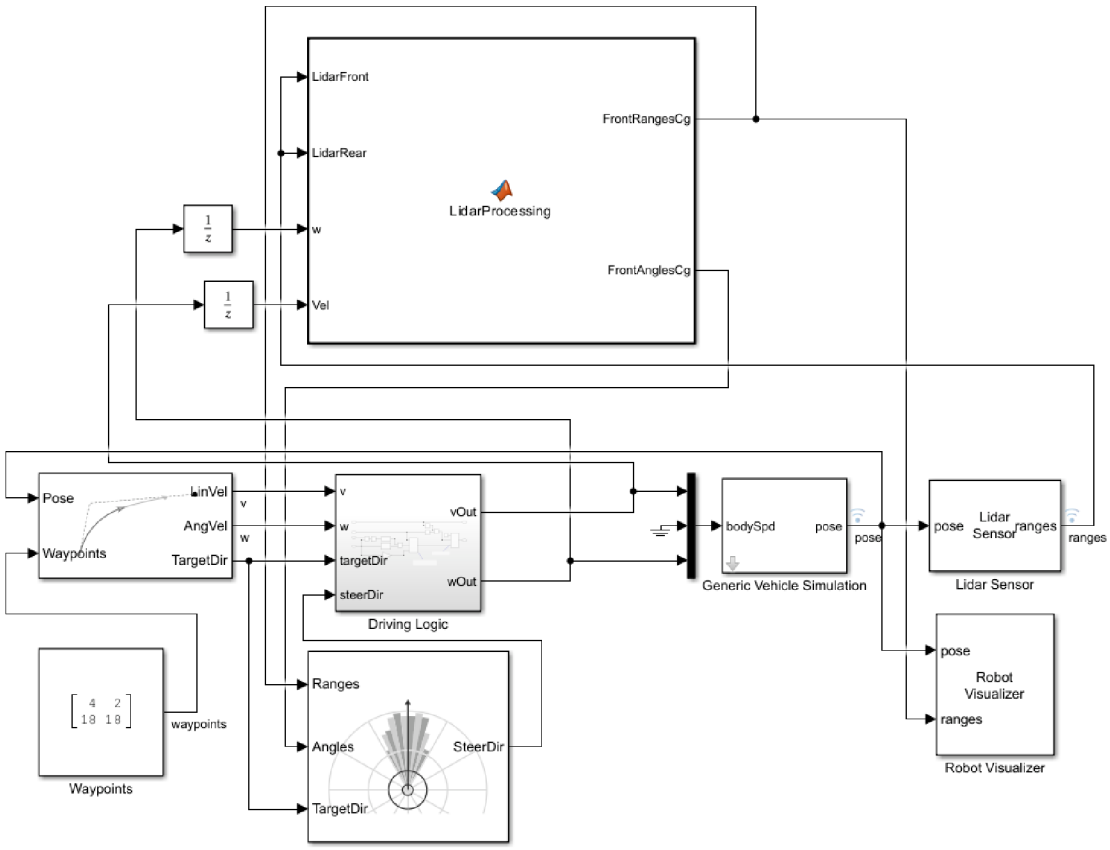
## 3.3 Simulace navrženého algoritmu

Pro prvotní testování algoritmu byla vytvořena simulace v prostředí Simulink. Schéma je vidět na obrázku 3.6. Simulace využívá Mobile Robotics Simulation Toolboxu dostupného z [18]. V toolboxu jsou již implementovány bloky pro simulaci a vizualizaci kinematického modelu vozidla a lidarů v rámci nahrané mapy.

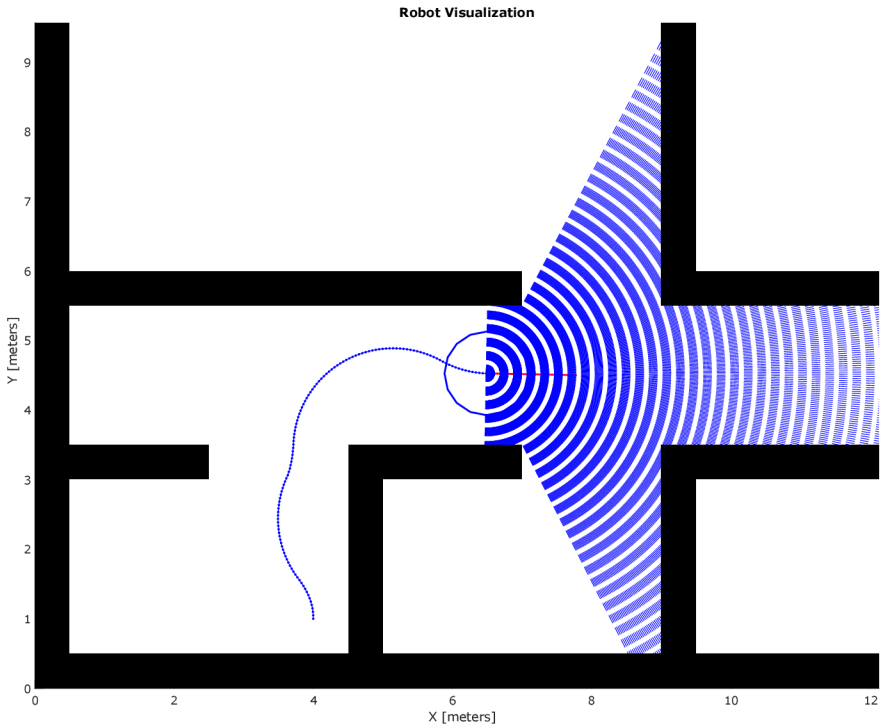
Namísto Follow-me algoritmu byl pro simulační účely využit blok Pure Pursuit, který má na vstupu sadu bodů, které následuje, a pozici vozidla v rámci mapy. Na základě toho vypočítává lineární a úhlovou rychlost a směr k cíli [19]. V subsystému Driving Logic je přepínáno mezi Pure Pursuit algoritmem

a VFH+ algoritmem na základě směrové odchylky od cíle. Blok LidarProcessing je uživatelský blok s navrženým algoritmem popsaným výše. Aby nedošlo ke vzniku algebraické smyčky, jsou signály pro translační a úhlovou rychlost opatřeny bloky Unit Delay, které zpozdí signál o jednu periodu. Bloky Generic Vehicle Simulation a Lidar Sensor jsou z výše zmíněného toolboxu a slouží k simulaci chování vozidla a lidaru. Robot Visualizer slouží k vizualizaci robota s paprsky lidaru v rámci mapy.

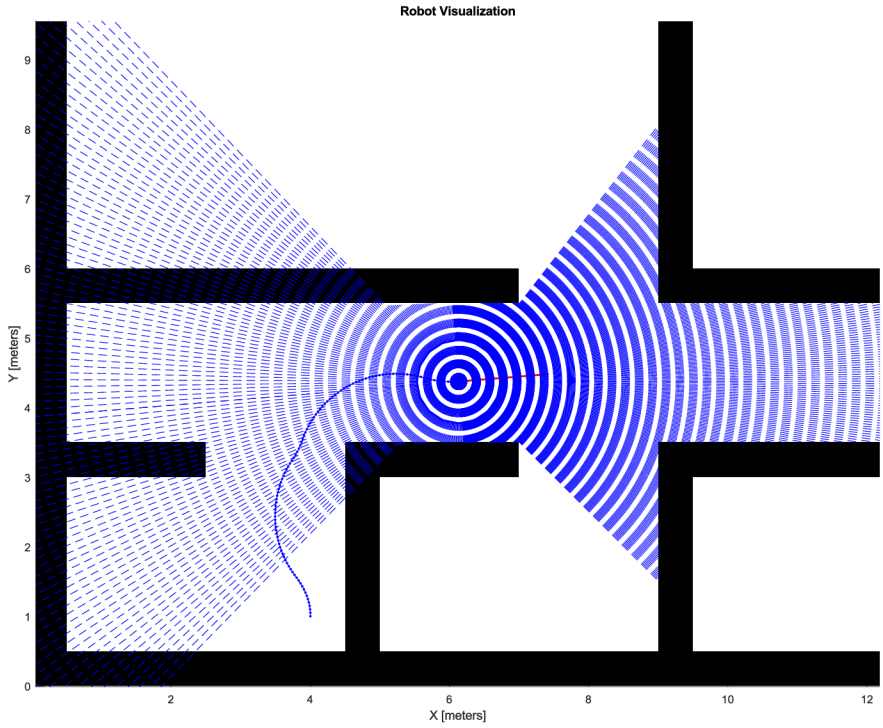
Na obrázcích 3.7 a 3.8 je vidět, že při použití algoritmu byl úspěšně rozšířen rozsah lidaru tak, aby pokrýval mrtvé úhly vozidla.




Obrázek 3.6: Simulační schéma v Simulinku




Obrázek 3.7: Simulace bez navrženého algoritmu




Obrázek 3.8: Simulace s navrženým algoritmem

## 3.4 Ackermannovo řízení

Protože transformace souřadných systémů při pohybu probíhá na základě odometrie vozidla, je vhodné aby byla co nejpřesnější. Proto bylo do řídicí struktury implementováno Ackermannovo řízení.

Ackermannovo řízení umožňuje zajistit správné odvalování kol při zatáčení a snížení jejich smýkání. Ackermannův podvozek se vyznačuje čtyřmi koly, z nichž přední dvě jsou řiditelná. Protože při zatáčení opisuje vnitřní a vnější kolo různé poloměry, musí pro splnění Ackermannovy podmínky být vnitřní kolo natočeno více než vnější [21]. To může být zajištěno buď konstrukčně pomocí tzv. lichoběžníku řízení, nebo pomocí dvou nezávisle řízených servopohonů na předních kolech. Podvozek, kterým se tato práce zabývá, využívá druhé možnosti.

### Odvození rovnic pro Ackermannovo řízení

Pro odvození Ackermannova řízení bude uvažován ideální případ, kdy jsou kola ideálně tuhá, bočně nepoddajná a vozidlo se pohybuje v malých rychlostech. Ackermannova podmínka pak říká, že střed otáčení musí ležet na prodloužené ose zadní nápravy viz obrázek 3.9. Následně můžeme uvažovat, že základní úhel natočení je stejný jako úhel natočení kola pro tříkolku. Na základě těchto úvah vzniknou v obrázku tři trojúhelníky, ze kterých se dají odvodit následující rovnice

$$\begin{aligned} \tan\phi &= \frac{l}{r} \\ \tan\phi_i &= \frac{l}{r - \frac{w}{2}} \\ \tan\phi_o &= \frac{l}{r + \frac{w}{2}}, \end{aligned}$$

kde $l$ je rozvor kol, $w$ je rozchod kol, $r$ je poloměr otáčení vozidla, $\phi$ je základní úhel natočení prostředního kola, $\phi_i$ je úhel natočení vnitřního kola a $\phi_o$ je úhel natočení vnějšího kola.

Odečtením rovnic vnitřního a vnějšího kola je možné odvodit Ackermannovu podmínku

$$\cot\phi_o - \cot\phi_i = \frac{r + \frac{w}{2}}{l} - \frac{r - \frac{w}{2}}{l} = \frac{w}{l}.$$

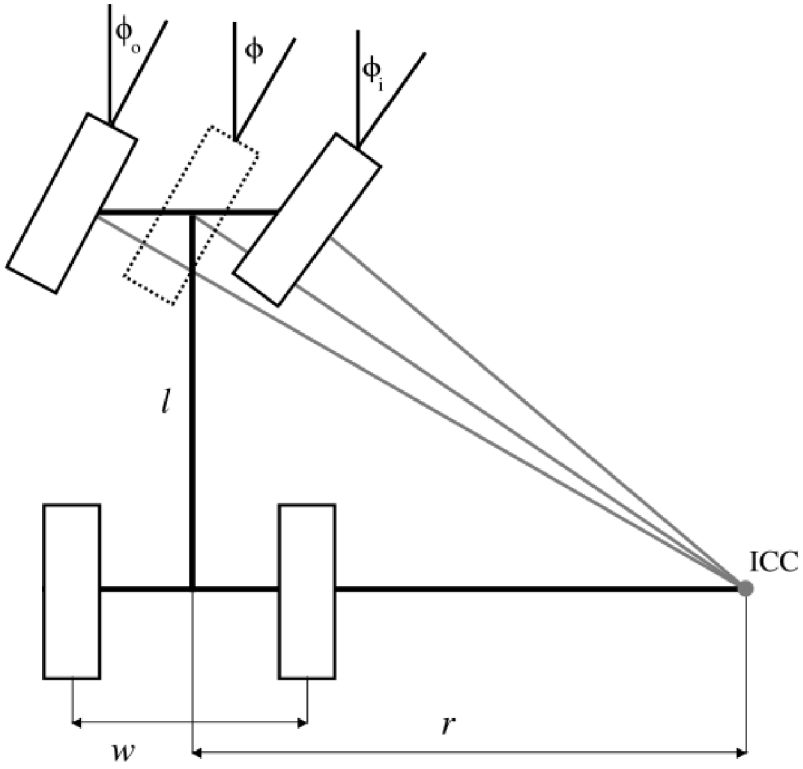
Za využití základního úhlu $\phi$ je možné obdobně odvodit rovnice pro oba úhly

$$\begin{aligned} \cot\phi_i &= \cot\phi - \frac{w}{2 \cdot l} \\ \cot\phi_o &= \cot\phi + \frac{w}{2 \cdot l}. \end{aligned}$$

Protože funkce kotangens není definována v nule, je nutné vztah upravit do podoby

$$\cot \phi_i = \frac{l \cdot \tan \phi}{l + \frac{w}{2} \cdot \tan \phi}$$
$$\cot \phi_o = \frac{l \cdot \tan \phi}{l - \frac{w}{2} \cdot \tan \phi}.$$

Pro použití v algoritmu bylo nutné ošetřit, které kolo je aktuálně vnitřní a které vnější na základě hodnoty základního úhlu [20, 22].




Obrázek 3.9: Ackermannovo řízení [20]

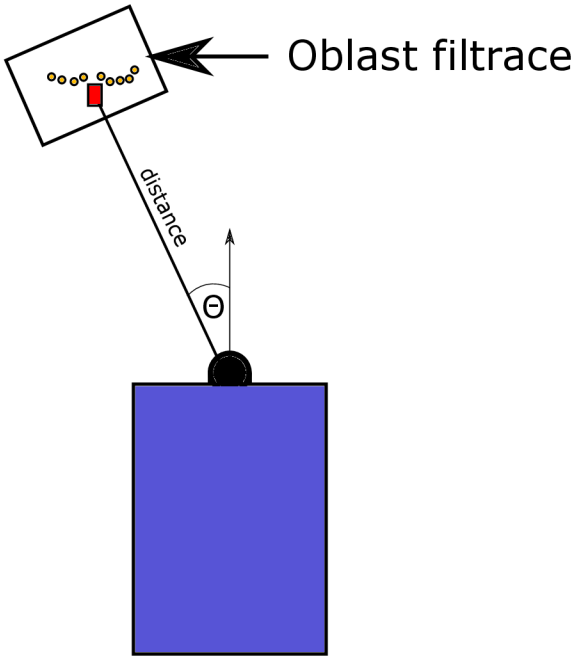
# 4 Filtrace osoby nesoucí UWB ovladač

Jedním z problémů dosavadního algoritmu je, že se vozítko snaží vlivem VFH+ algoritmu objet následovanou osobu. To může mít za následek, že vozidlo zvolí neoptimální trajektorii, nebo že sledovaná osoba vypadne ze zorného pole UWB majáků a vozidlo se zastaví.

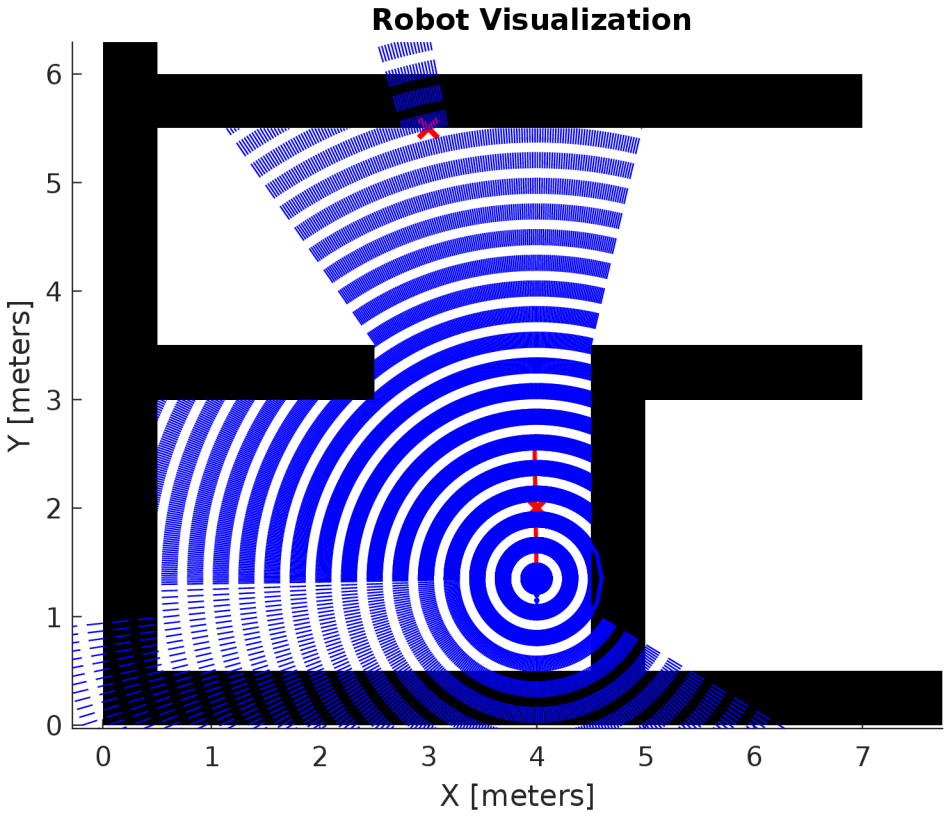
K řešení této situace se nabízí využití detekce osoby na základě dat z 2D lidaru. To lze provést například pomocí detekce pohybu, nebo pomocí podobností geometrických rysů nohou člověka a využití klasifikačních tříd [23]. Vzhledem k proměnnosti snímků nohou zaznamenaných 2D lidarem je taková detekce poměrně náročná a může vykazovat chybovost. Nabízí se proto řešení pomocí informací získaných z UWB systému.

### Filtrace na základě údajů z UWB

Pro filtraci osoby byl vytvořen algoritmus, který na základě informací z UWB systému odstraní osobu z dat lidarů, které vstupují do VFH+ algoritmu. Oblast filtrace je omezena úhlovým rozsahem UWB systému a také rozsahem vzdáleností daných maximální vzdáleností reakce VFH+ algoritmu a požadované vzdálenosti Follow-me algoritmu. To odpovídá rozsahu 1,5–4 m. Z UWB systému vystupují informace o vzdálenosti a úhlu mezi předkem vozidla a UWB ovladačem. Díky těmto informacím je možné určit oblast, ve které se budou nacházet nohy osoby nesoucí UWB ovladač. Oblast byla určena experimentálně pro vzdálenost 1,5 m a 4 m a její velikost je přepočítávána lineárně na základě údaje o vzdálenosti od vozidla. Body lidaru nacházející se v určené oblasti jsou vyfiltrovány, takže na ně VFH+ algoritmus nereaguje. Situace je znázorněna na obrázku 4.1, kde žluté body představují zaznamenané nohy osoby a červený obdélník je UWB ovladač. V rámci bezpečnosti byl separátní lidarový snímek vstupující do funkce nouzového zastavení zachován bez filtrace osoby. Tento přístup byl ověřen simulačně, jak je vidět na obrázku 4.2, kde se podařilo vyfiltrovat cílovou pozici z dat lidaru.




Obrázek 4.1: Určení oblasti pro filtraci




Obrázek 4.2: Simulace filtrace cíle z lidarových hodnot

# 5 DWA algoritmus jako alternativa k VFH+

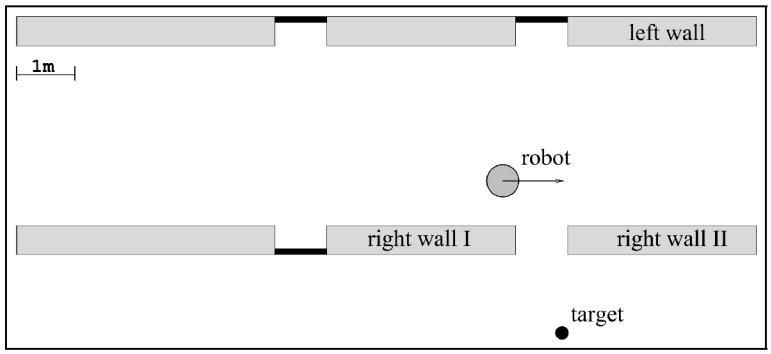
Jelikož VFH+ algoritmus někdy selhává, a to zejména v úzkých chodbách a průchodech, byl zvolen Dynamic Window Approach (DWA) algoritmus, jako alternativa pro lokální navigaci ve snaze vylepšit chování vozidla v těchto situacích.

## 5.1 Princip DWA algoritmu

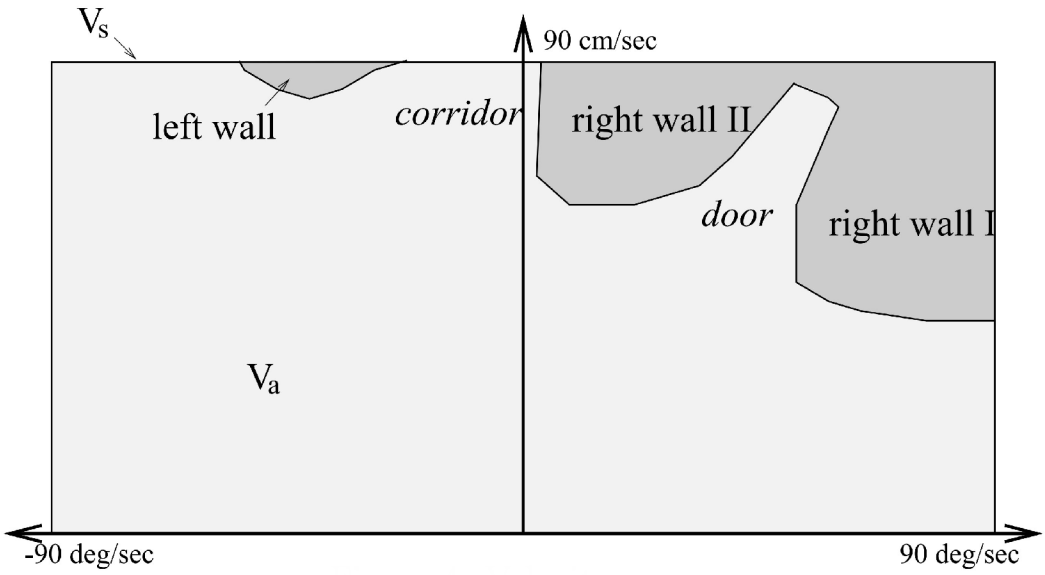
DWA algoritmus slouží pro lokální plánování trajektorie a vyhýbání se překážkám. Jeho princip se dá rozdělit na dvě základní části. Nejprve je vytvořen prohledávaný prostor na základě dosažitelných úhlových a translačních rychlostí a následně je zvolena optimální trajektorie pomocí maximalizace cílové funkce. DWA algoritmus bere v potaz kinematická a dynamická omezení vozidla [24].

### Generování prohledávaného prostoru

Algoritmus aproximuje trajektorie jako kruhové křivky dané dvojicí rychlostí $(v, \omega)$, kde $v$ je translační rychlost a $\omega$ je úhlová rychlost vozidla. Tak vznikne dvojrozměrný prostor $V_s$ tvořený ze všech dosažitelných rychlostí [24]. Pro znázornění bude využita situace na obrázku 5.1.




Obrázek 5.1: Vzorová situace [24]




Obrázek 5.2: Prostor rychlostí [24]

Překážky nacházející se v okolí robota limitují jeho translační a úhlové rychlosti. Maximální rychlosti závisí na vzdálenosti nejbližší překážky na trajektorii. Pokud je zvolena trajektorie daná rychlostmi $(v, \omega)$, pak lze definovat $dist(v, \omega)$ jako vzdálenost k nejbližší překážce na této trajektorii. Rychlosti jsou považovány za přípustné, pokud je robot schopen zastavit před překážkou bez způsobení kolize. Množinu přípustných rychlostí $V_a$ je možné popsat jako

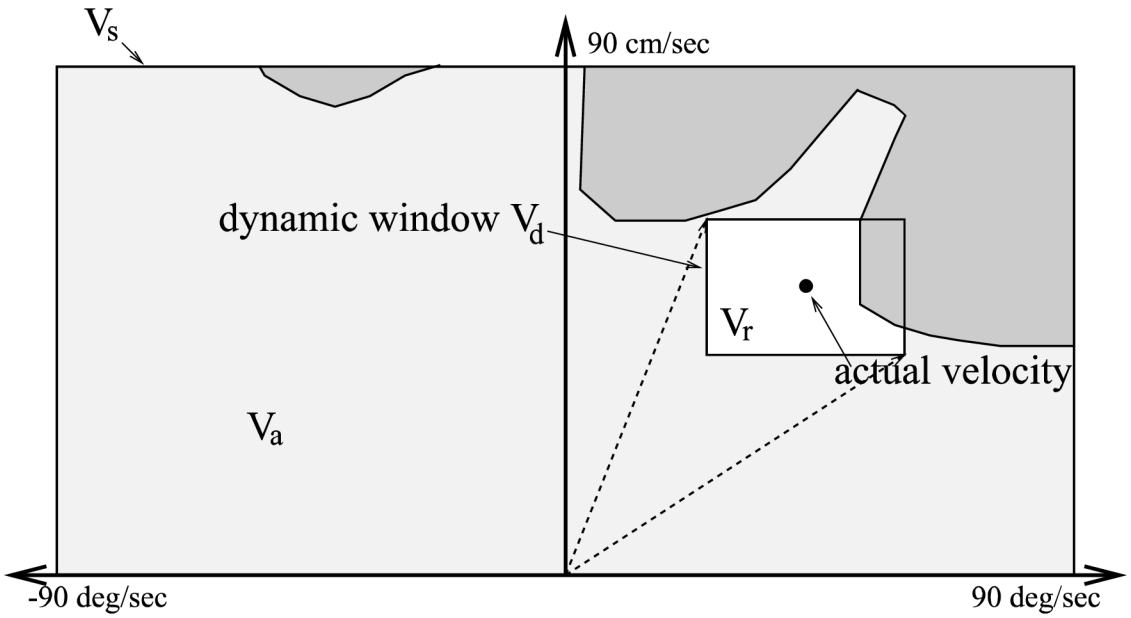
$$V_a = \left\{ (v, \omega) \mid v \leq \sqrt{2 \cdot dist(v, \omega) \cdot \dot{v}_b} \wedge \omega \leq \sqrt{2 \cdot dist(v, \omega) \cdot \dot{\omega}_b} \right\},$$

kde $\dot{v}_b$ a $\dot{\omega}_b$ jsou zpomalení robota při brzdění elektromotorem [24]. Celkový prostor rychlostí $V_s$ a prostor tvořený přípustnými rychlostmi $V_a$ je vidět na obrázku 5.2.

Aby bylo bráno v potaz omezené zrychlení motorů, je celkový prostor redukován do dynamického okénka na rychlosti, které je možné dosáhnout v následujícím časovém intervalu. Dynamické okénko $V_d$ je možné popsat výrazem

$$V_d = \{ (v, \omega) \mid v \epsilon [v_a - \dot{v} \cdot t, v_a + \dot{v} \cdot t] \wedge \omega \epsilon [\omega_a - \dot{\omega} \cdot t, \omega_a + \dot{\omega} \cdot t] \},$$

kde $t$ je časový interval, během kterého působí na vozidlo zrychlení $\dot{v}$ a $\dot{\omega}$ , a $(v_a, \omega_a)$ je vektor aktuálních rychlostí [24].




Obrázek 5.3: Dynamické okénko [24]

Výsledný prohledávaný prostor $V_r$ daný výše popsanými omezeními je pak možné definovat jako průnik všech oblastí

$$V_r = V_s \cap V_a \cap V_d$$

a je znázorněn na obrázku 5.3 bílou barvou [24].

### Hledání maxima cílové funkce

Po nalezení výsledného prohledávaného prostoru $V_r$ je na základě cílové funkce zvolena optimální dvojice rychlostí $(v, \omega)$. Prohledávaný prostor rychlostí je diskretizován s nastavitelnou velikostí kroku ovlivňující výpočetní náročnost. Pro jednotlivé trajektorie v diskretizovaném prostoru $V_r$ je spočtena cílová funkce $G(v, \omega)$ pomocí rovnice

$$G(v, \omega) = \sigma(\alpha \cdot angle(v, \omega) + \beta \cdot dist(v, \omega) + \gamma \cdot velocity(v, \omega)).$$

Hodnota funkce $angle(v, \omega)$ udává, jak velká je odchylka výsledného úhlu natočení od cíle a je přepočtená tak, aby nulová odchylka od cíle měla v cílové funkci maximální hodnotu. Funkce $dist(v, \omega)$ udává vzdálenost nejbližší překážky od zvolené trajektorie. Funkce $velocity(v, \omega)$ představuje velikost translační rychlosti. Parametry $\alpha$, $\beta$ a $\gamma$ slouží k nastavení vah jednotlivých funkcí, díky kterým je možné řídit preference ve volbě trajektorie. Všechny funkce v cílové funkci $G(v, \omega)$ jsou normalizovány na rozsah hodnot [0,1], díky čemuž je průběh cílové funkce vyhlazenější [24].

## 5.2 Návrh DWA algoritmu pro řešenou úlohu

Pro vyhýbání se překážkám byl vytvořen DWA algoritmus v prostředí Simulink pomocí bloku Matlab function a byl upraven aby vyhovoval řešené problematice. Vytvořený program čerpal ze zdrojového kódu dostupného z [25].

Jako vstup do algoritmu slouží data lidaru vystupující z algoritmu popsaného v 3.2. Data jsou přepočteny do souřadného systému vozidla tak, aby body lidaru tvořily mapu překážek v okolí vozidla. Dalšími vstupy do algoritmu jsou aktuální translační rychlost a úhel natočení kol. Protože algoritmus pracuje s translační a úhlovou rychlostí vozidla, je úhlová rychlost vypočtena pomocí vztahu

$$\omega_a = \frac{v_a}{l} \cdot \tan \psi_a,$$

kde $\phi_a$ je aktuální úhel natočení kol, $v_a$ je aktuální translační rychlost a $l$ je rozvor kol. Časový interval pro vyhodnocení následujících rychlostí byl nastaven na 0,02 s, což odpovídá jednomu cyklu použitého průmyslového PC.

Pro tvorbu dynamického okénka je nutné zadefinovat fyzikální omezení vozidla. Ty odpovídají hodnotám v tabulce 5.1. Hodnoty byly nastaveny experimentálně na základě maximální rychlosti dovolené regulátorem ve Follow-me algoritmu.

| Parametry vozidla | |
|---|---|
| Maximální rychlost | $0,75\ m/s$ |
| Maximální zrychlení | $0,5\ m/s^2$ |
| Maximální úhlová rychlost | $0,52\ rad/s$ |
| Maximální úhlové zrychlení | $0,8\ rad/s^2$ |

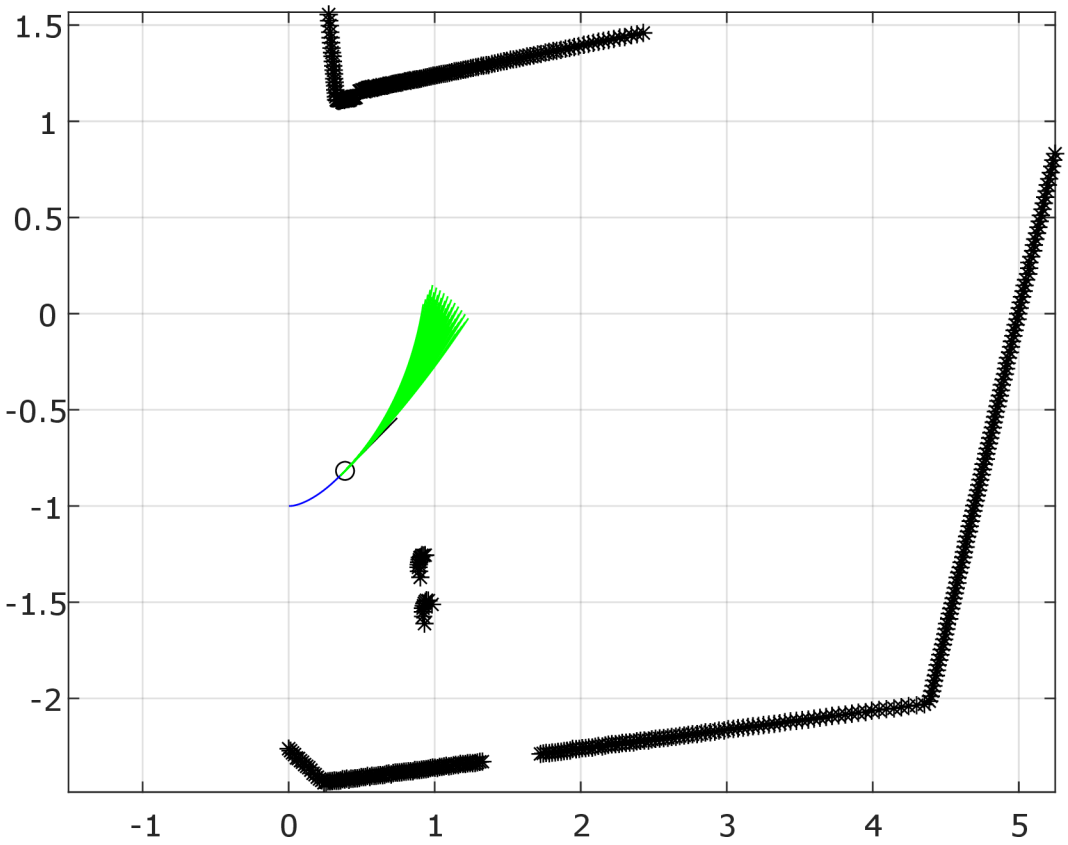
Tabulka 5.1: Tabulka fyzikálních omezení vozidla

Dynamické okénko je pomocí těchto hodnot vytvořeno stejně jako je popsáno výše v 5.1. Diskretizace okna je provedena s krokem 0,005 m/s pro translační rychlost a 0,0087 rad/s pro úhlovou rychlost.

Aby byla minimalizována šance, že se vozidlo dostane do situace, kde se kvůli omezené manévrovatelnosti zasekne, byl výpočet trajektorie rozšířen o predikční horizont s časem 3 s. To umožní predikovat, kam by se vozidlo dostalo v případě, že se vydá zvolenou trajektorií, a preferovat tak trajektorie, které nevedou k překážkám.

Následně jsou spočteny jednotlivé hodnoty vstupující do cílové funkce pro vyhodnocení. Pro zamezení naražení do překážky jsou z uvažovaných rychlostí odstraněny ty, které by mohly vést ke kolizi. Všechny body lidaru tvořící mapu jsou brány jako překážky, a protože jsou vztaženy k těžišti vozidla, jsou dilatovány o poloměr vozidla a bezpečnou vzdálenost od překážky podobně jako je popsáno v 2.2. Tato vzdálenost byla zvolena jako 0,6 m. V potaz jsou brány jen trajektorie, které přímo neporuší tuto vzdálenost od překážek. Bezpečná vzdálenost je nadále rozšířena o vzdálenost, za kterou

vozidlo stihne zastavit, pokud bude v dané trajektorii pokračovat zvolenou rychlostí. Do výsledné cílové funkce jsou uvažovány jen hodnoty $dist(v, \omega)$ a k ním příslušící trajektorie, které splní tyto podmínky. Pro výpočet funkce $angle(v, \omega)$ je využita informace o úhlové odchylce k cíli vystupující z UWB systému. Tento úhel je porovnávaný s predikovaným úhlem natočení vozidla a výstupem je hodnota, která při nulové odchylce od cíle odpovídá maximální hodnotě funkce. Do hodnoty $velocity(v, \omega)$ jsou uloženy translační rychlosti příslušící vyhovujícím trajektoriím. Všechny uložené hodnoty v cílové funkci jsou následně normalizovány.




Obrázek 5.4: Zvažované trajektorie v dynamickém okně [25]

Parametry, kterými jsou jednotlivé složky cílové funkce násobeny, slouží k ladění chování algoritmu. Jejich zvolené hodnoty jsou v tabulce 5.2.

| Parametry cílové funkce | |
|---|---|
| $\alpha$ | 0.05 |
| $\beta$ | 0.2 |
| $\gamma$ | 0.1 |

Tabulka 5.2: Hodnoty parametrů cílové funkce

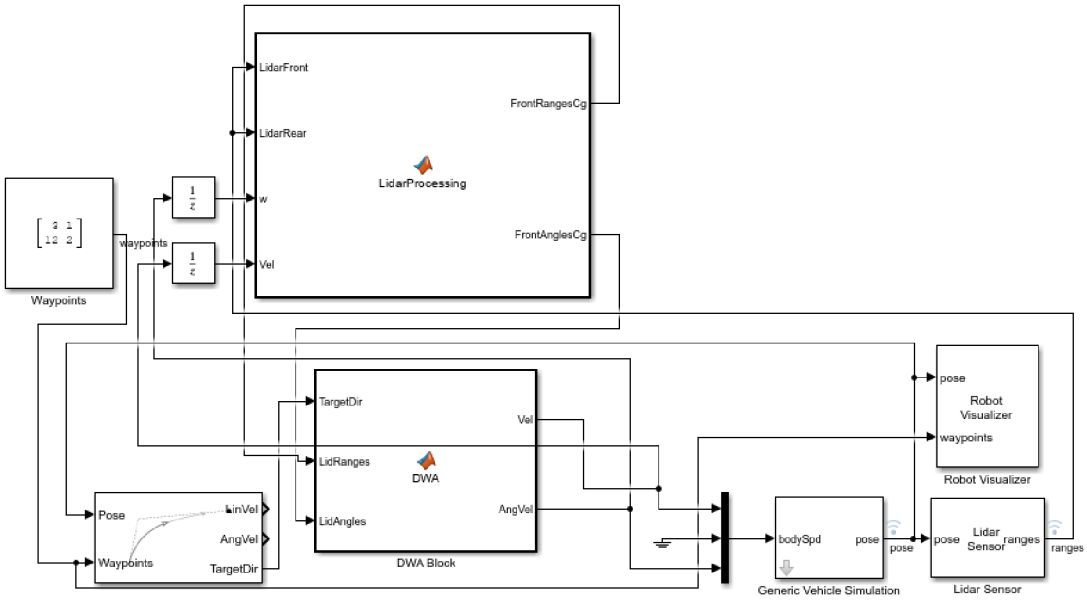
Z hodnot cílové funkce je nalezeno maximum, čímž je zjištěna optimální trajektorie. Výstupem z algoritmu je optimální translační a úhlová rychlost vozidla. Úhlová rychlost je přepočtena zpět na požadovaný úhel natočení kol pomocí vztahu

$$\psi = \arctan\left(\frac{\omega \cdot l}{v}\right).$$

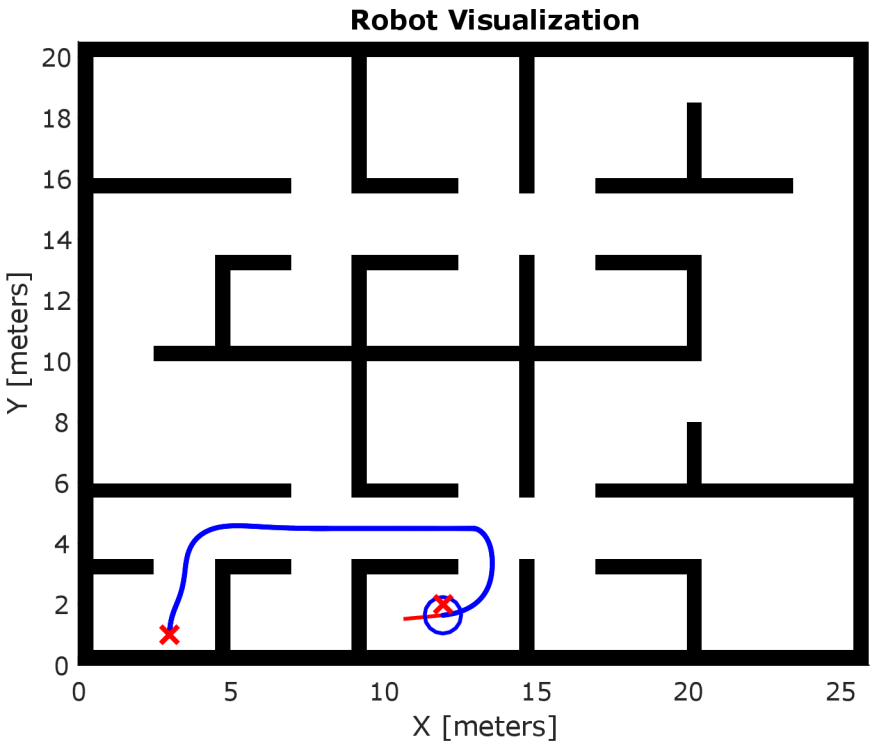
Množina vyhodnocovaných trajektorií v testovacím prostředí je znázorněna na obrázku 5.4.

## 5.3 Simulační ověření funkčnosti a porovnání s VFH+

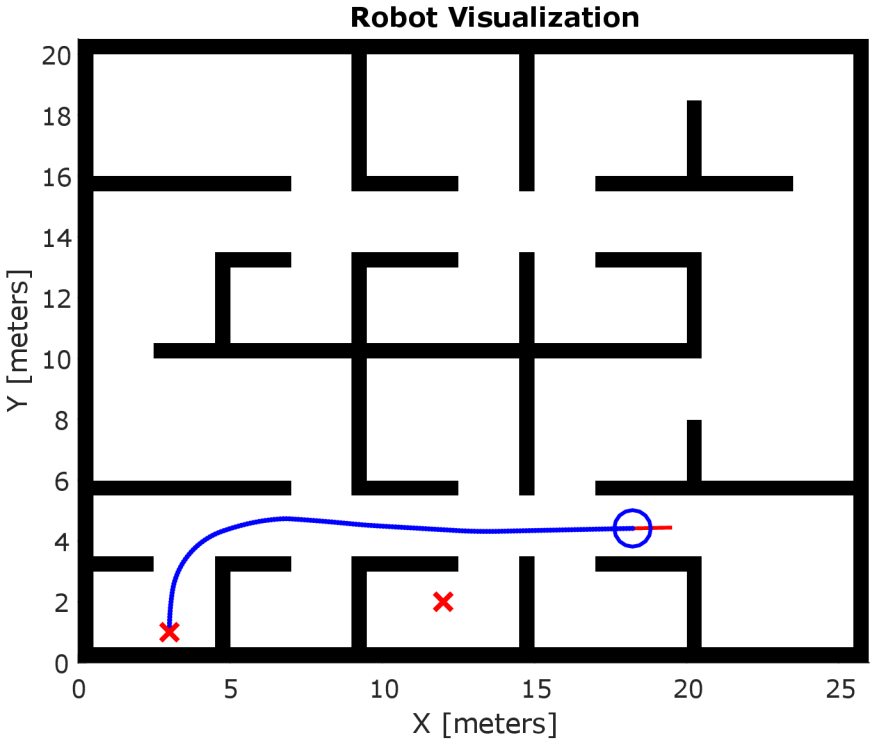
Pro testování DWA algoritmu byla vytvořena simulace v prostředí Simulink obdobně jako v 3.3. Jediným rozdílem je, že blok Pure Pursuit zde byl využit pouze pro zjištění směru k cíli, protože z DWA algoritmu vystupují přímo volené rychlosti na základě předem nastavených fyzikálních omezení. VFH+ algoritmus rychlosti nebere vůbec v potaz a jsou proto simulačně brány jako konstantní. Tato odlišná funkčnost obou algoritmů může při zvoleném simulačním porovnání zkreslovat výsledky.



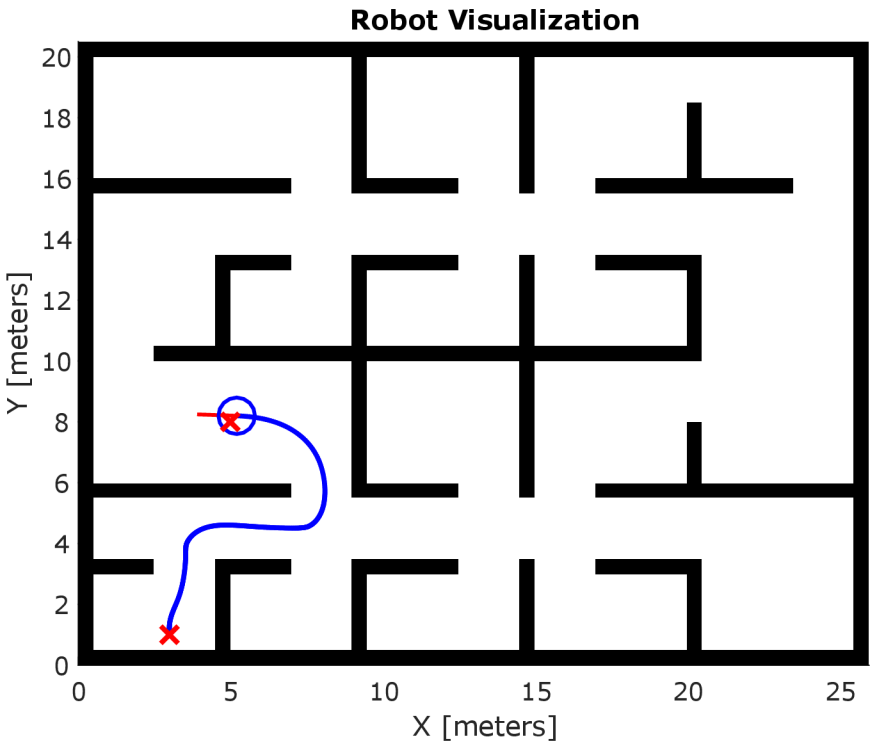

Obrázek 5.5: Simulační schéma v Simulinku



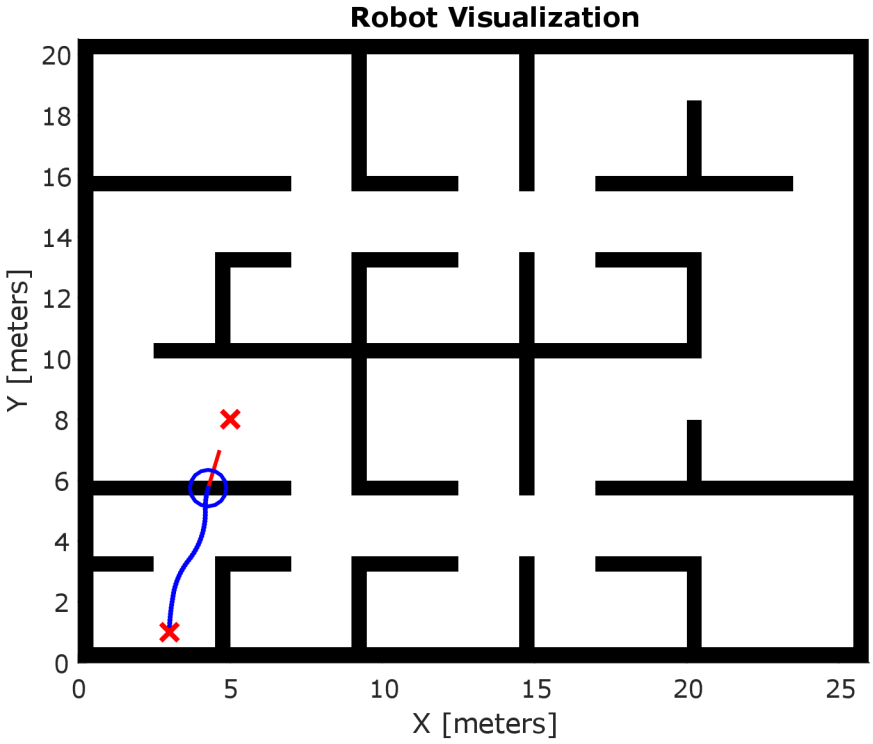

Obrázek 5.6: Simulace DWA algoritmu 1




Obrázek 5.7: Simulace VFH algoritmu 1




Obrázek 5.8: Simulace DWA algoritmu 2




Obrázek 5.9: Simulace VFH algoritmu 2

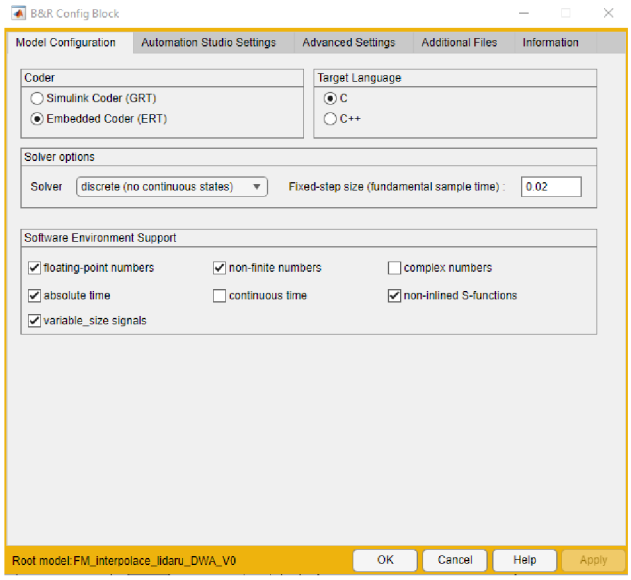
Algoritmy byly testovány na několika situacích. Na základě simulací se ukázalo, že DWA algoritmus dokázal vyřešit i situace, kde VFH+ algoritmus ve stejné konfiguraci jako na vozítku selhal. DWA by tedy mohl vykazovat lepší chování, zejména pak v odbočování do úzkých průchodů a při jejich průjezdech. Je důležité podotknout, že simulace neodpovídají stoprocentně reálné Follow-me aplikaci, ale slouží spíše k odzkoušení obou algoritmů a porovnání jejich obecného chování. Zároveň může mít robotická platforma jiné chování než, použitý simulační blok Generic Vehicle Simulation, který zanedbává některé parametry.

# 6 Implementace na robotické platformě a výsledky testování

## 6.1 Generování kódu do průmyslového PC

Pro implementaci algoritmů na robotickou platformu bylo využito generování kódu z prostředí Simulink do prostředí Automation Studio pomocí softwaru Automation Studio Target for Simulink. I přesto že Simulink podporuje dynamickou alokaci paměti, je pro generování kódu do Automation Studia nutné alokovat všechny proměnné staticky.

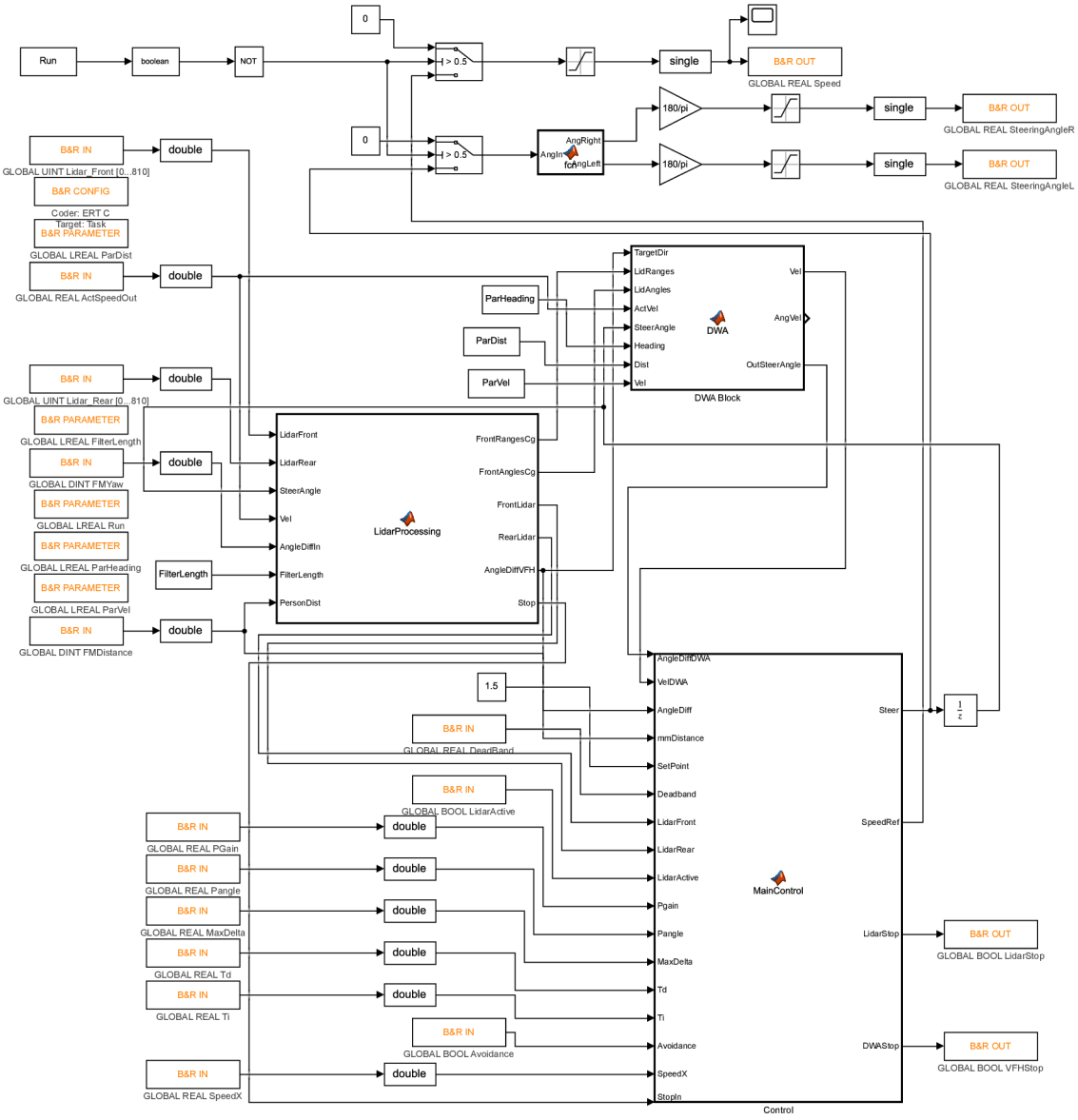
V bloku konfigurace pro AS Target for Simulink je možné nastavit využitý Coder pro generování kódu, jazyk, ve kterém bude kód generován, a parametry kódu. Pro generování kódu byl využit Embedded Coder v prostředí Simulink, který umožňuje generovat optimalizovaný kód pro embedded aplikace v jazycích C a C++ [26]. Nastavení jsou vidět na obrázku 6.1.




Obrázek 6.1: Nastavení pro generování kódu

Výsledné blokové schéma programu pro DWA algoritmus je vidět na obrázku 6.2, kde je v bloku LidarProcessing algoritmus pro pokrytí mrtvých úhlů popsaný v 3.2 a v bloku MainControl je řízení Follow-me algoritmu. Schéma

pro VFH+ je obdobné s rozdílem, že namísto bloku DWA je blok VFH implementovaný v Matlabu. Target for Simulink umožňuje pomocí bloků definovat vstupní a výstupní proměnné, které se vygenerují do programu v Automation Studia společně s kódem programu.




Obrázek 6.2: Blokové schéma programu pro DWA algoritmus

V Automation Studiu byl vygenerovaný program přiřazen do 20ms taskové třídy. Již existující řídicí struktura byla upravena tak, aby bylo možné řídit natočení obou kol zvlášť, a využít úhlů pro Ackermannovo řízení.

## 6.2 Algoritmus pro tvorbu mapy

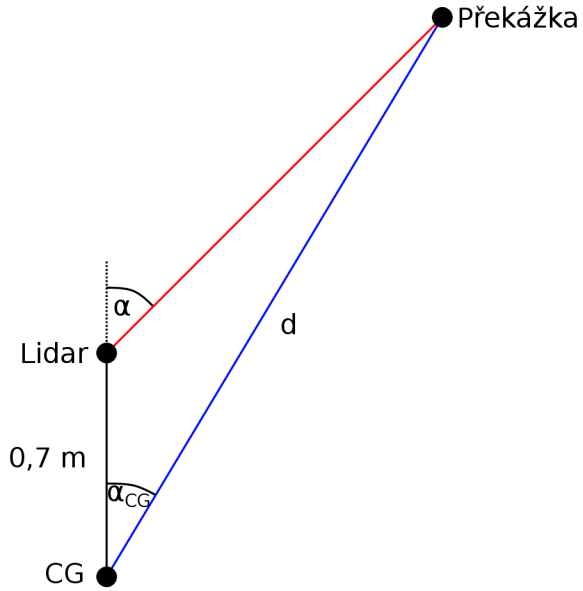
Navržený algoritmus byl testován primárně v kombinaci s VFH+ algoritmem, za účelem vylepšení jeho chování. Na základě testování a doby cyklu 20 ms byla zvolena délka bufferu pro ukládání bodů překážek 200 hodnot. Tato délka bufferu spolehlivě pokryje mrtvé úhly vozidla i v případě pomalejší jízdy za osobou.

### Export a zpracování dat

Pro export dat lidaru z vozítka pro ověření funkčnosti bylo využito extérního módu Simulinku podporovaného softwarem AS Target for Simulink. Tento režim umožňuje v reálném čase sledovat, upravovat a ukládat data z připojeného zařízení pomocí programu v Simulinku [27]. Data byla uložena pomocí Simulink bloku To Workspace. Kvůli paměťovým omezením přenosu bylo možné exportovat pouze údaje o naměřené vzdálenosti, nikoliv však úhly. Protože hodnoty lidaru jsou transformovány do těžiště vozidla, bylo nutné dopočítat korespondující transformované úhly pro správné zobrazení výsledků. Přepočty byly provedeny za využití sinové věty a obecných goniometrických úprav a výsledný vzorec vypadá následnovně

$$\alpha_{CG} = \pi - \arcsin\left(\frac{0,7 \cdot \sin(\pi - \alpha)}{d}\right) - \pi + \alpha,$$

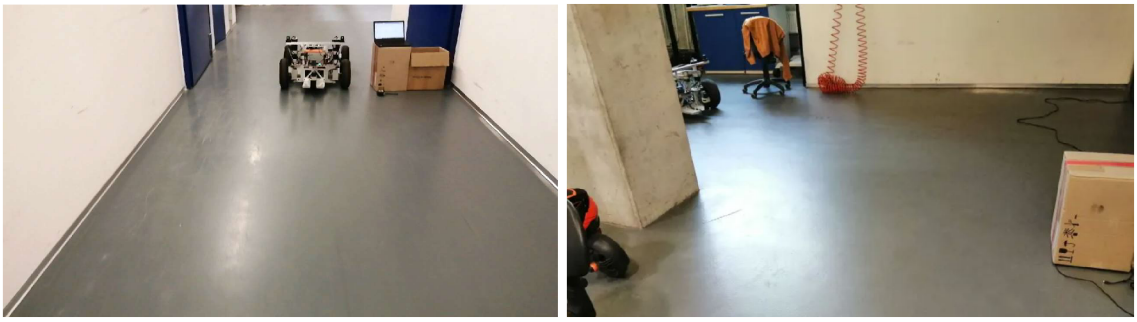
kde $\alpha_{CG}$ je výsledný úhel z těžiště vozidla, $\alpha$ je původní úhel paprsků lidaru před převedením a $d$ je změřená vzdálenost. Hodnoty jsou znázorněny na obrázku 6.3.




Obrázek 6.3: Znázornění přepočtu úhlů

### Vyhodnocení výsledků algoritmu

Pro vyhodnocení výsledků byly vybrány dvě lokace na obrázcích 6.4. Z vykreslených dat lidaru na obrázcích 6.6 a 6.8 je vidět, že algoritmus úspěšně zaznamenává překážky i v mrtvých úhlech vozidla. Při testování s VFH+ algoritmem nedošlo k boční kolizi vozidla s překážkami v mrtvém úhlu a algoritmus tedy plní svůj praktický účel.



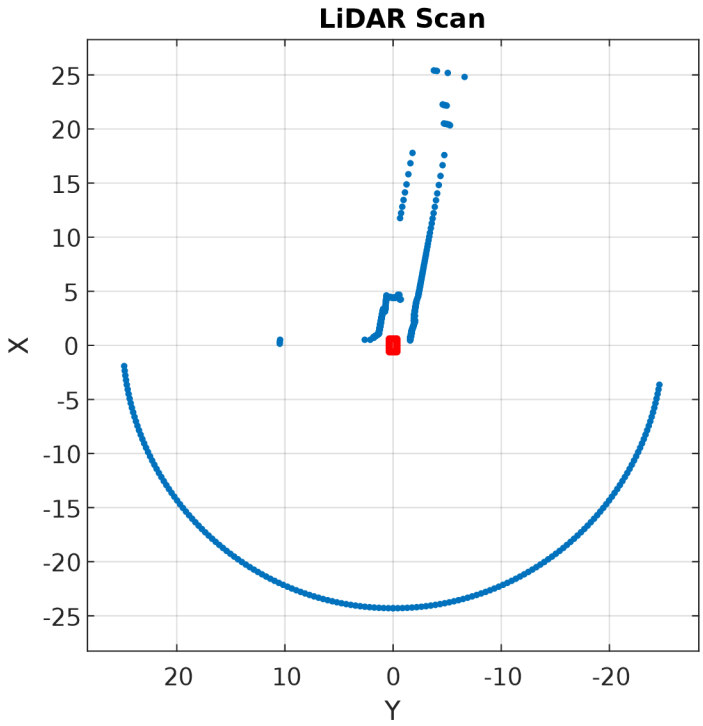
(a) Testování v podlouhlé chodbě (b) Testování v rohu místnosti

Obrázek 6.4: Testovací oblasti

Na dodatečně vygenerovaném výstupu z VFH+ algoritmu pro situaci na obrázku 6.8 (pro znázornění je uvažován cílový úhel jako -50° a stejně i maximální úhel zatočení) je vidět, že i přesto, že je překážka mimo zorné pole lidaru, je VFH+ algoritmus díky záznamu překážky schopen vyhodnotit trajektorii tak, aby nedošlo k boční kolizi. Na obrázku 6.10 je vidět, že bez zaznamenání překážky by ke kolizi dojít mohlo.

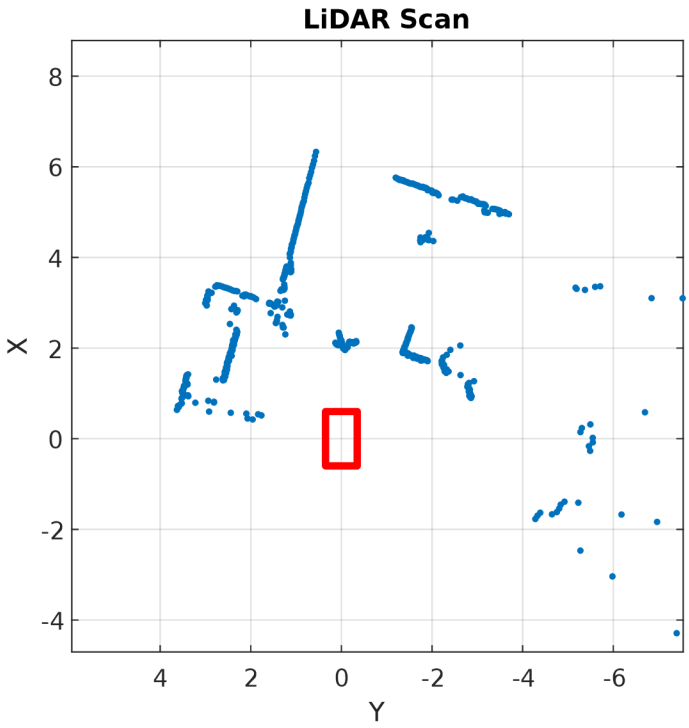
Při bližším pohledu na obrázek 6.6 je vidět, že při zatáčení dochází k mírnému zkreslení záznamu pozice překážek vůči realitě. To je očekávané a může být způsobeno nepřesnou odometrií. Chyba odometrie na testovaném vozítku může vznikat nedokonalou tuhostí kol a s nimi spojené konstrukce, případně i odchylkou v kalibraci nulové pozice natočení kol. Další chyby mohou vznikat vlivem uvažování po částech konstantní rychlosti a úhlu natočení kol a jinými aproximacemi ve výpočtech. Tato chyba se akumuluje s uraženou vzdáleností a hodnoty ke konci bufferu mohou být výrazněji posunuty vůči realitě. Pro delší časový úsek mapování by bylo vhodné tyto chyby co nejvíce minimalizovat.

Hlavním cílem bylo zmapovat blízké překážky nacházející se v mrtvých úhlech na bocích vozidla. V těchto oblastech nedochází k tak velkému zkreslení, protože naakumulovaná chyba je malá. Dále jsou navíc tyto údaje zkresleny vlivem rozlišení VFH+ algoritmu. Navržený přístup se tedy pro řešenou problematiku jeví jako dostačující a výpočetně méně náročná alternativa k jiným složitějším algoritmům pro tvorbu mapy.

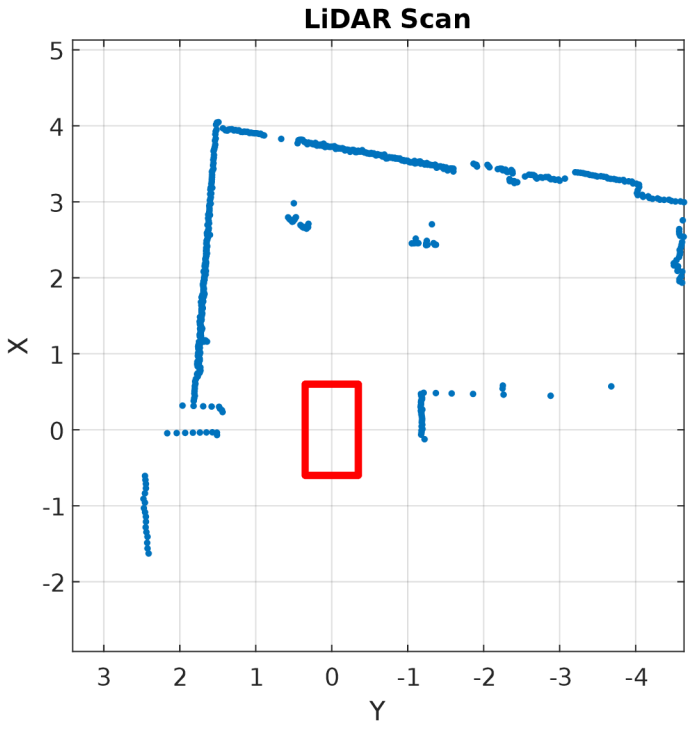



Obrázek 6.5: Výchozí pozice pro průjezd chodbou

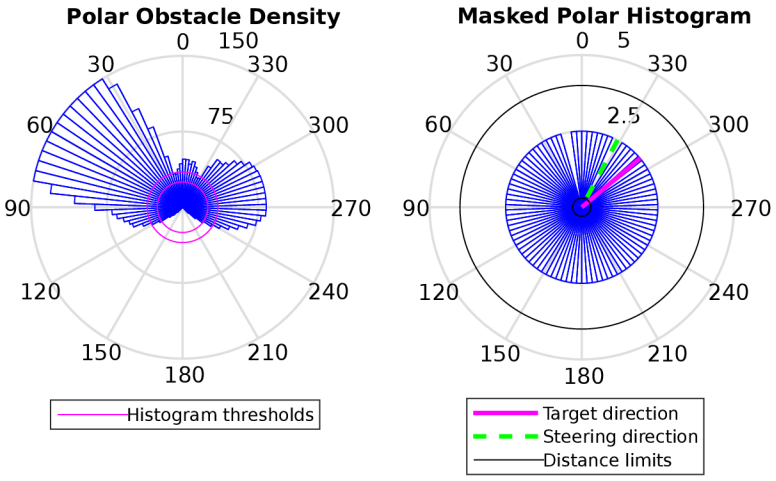

Obrázek 6.6: Pozice se zaznamenanými body předchozích překážek



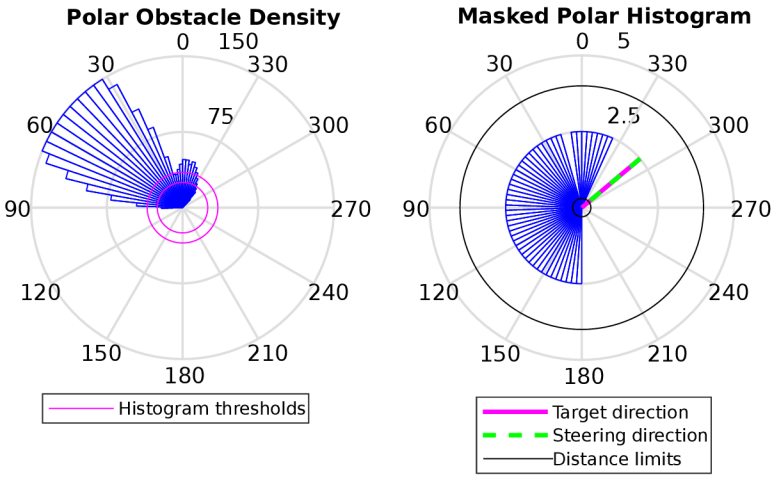

Obrázek 6.7: Výchozí pozice pro vjezd do rohu místnosti




Obrázek 6.8: Pozice se zaznamenanými body předchozích překážek



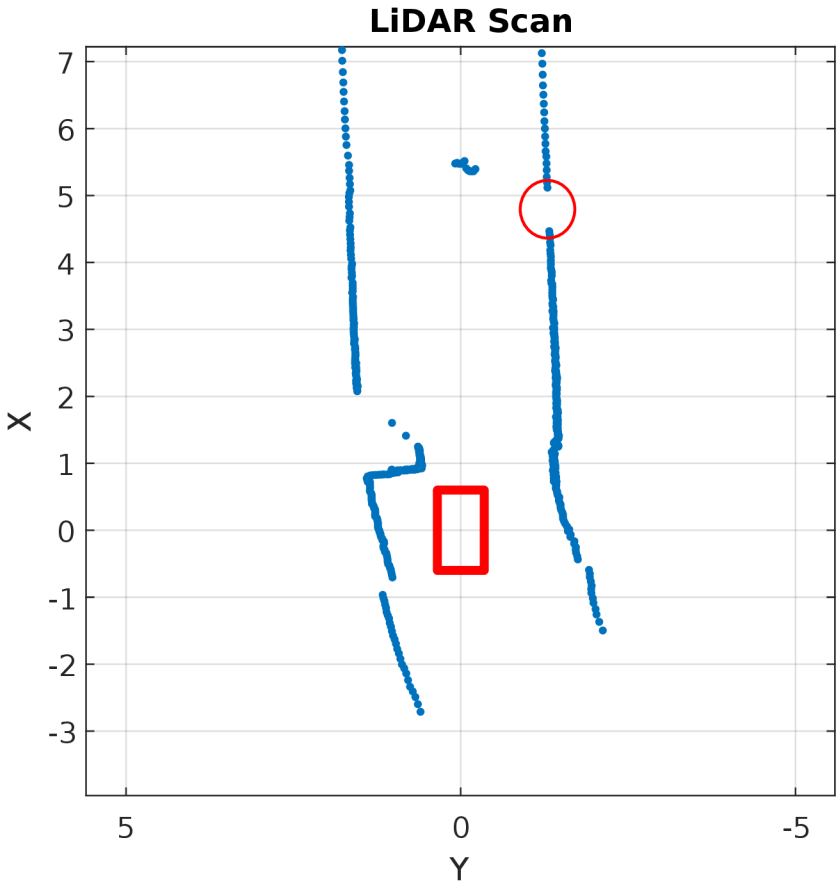

Obrázek 6.9: Výstup z VFH+ algoritmu pro situaci na obrázku 6.8




Obrázek 6.10: Výstup z VFH+ algoritmu pro situaci na obrázku 6.8 bez záznamu překážek

## 6.3 Filtrace osoby z dat lidaru

Jak je vidět z lidarového snímku na obrázku 6.11, osobu se z lidarových dat pomocí přístupu navrženého v 4 spolehlivě odstranit nepodařilo. Důvodem je velká fluktuace údajů z UWB systému, především pak velký rozptyl úhlových hodnot. Díky tomu i napříč filtraci souhlasí měřená pozice osoby s opravdovou jen málokdy. Navíc občasně dojde k odstranění bodů v oblasti, kde odstraněny být nemají, viz červený kruh na obrázku 6.11.




Obrázek 6.11: Lidarový snímek po pokusu o filtraci

## 6.4 Testování DWA algoritmu

### Úprava řídicí struktury pro implementaci DWA

Pro implementaci DWA algoritmu bylo nutné udělat poměrně velké zásahy do stávající řídicí struktury a jejího principu. Původní funkce Follow-me algoritmu popsaná v 2.1 musela být pozměněna tak, aby vyhovovala výstupům z DWA algoritmu. V prvotní verzi návrhu bylo plánováno pouze přepínat mezi výstupy z Follow-me algoritmu a výstupy z DWA algoritmu. Problémem je,

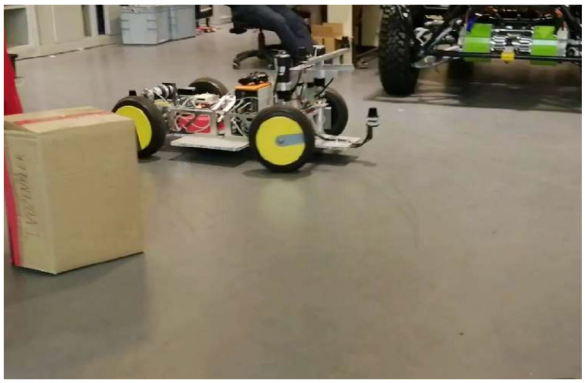
že není možné spolehlivě rozlišit, zda vozidlo následuje osobu nebo se DWA algoritmus snaží vyhnout překážce. To je způsobeno skutečností, že DWA algoritmus má na výstupu úhel natočení kol v podobě inkrementů dosažitelných za další výpočetní periodu na základě definovaných zrychlení. Rozlišení mezi algoritmy na základě výstupních úhlu tedy není možné zrealizovat.

Protože z DWA algoritmu vystupuje oproti VFH+ algoritmu i rychlost, byla struktura Follow-me pozměněna tak, že při jízdě dopředu je pohyb řízen pouze DWA algoritmem až do doby, kdy dojede na požadovanou vzdálenost od osoby. V tu chvíli je pohyb řízen opět regulátory původního Follow-me algoritmu. To umožňuje následování osoby s konstantní volbou trajektorie bez překážek, ale také zastavení na požadované vzdálenosti před osobou a couvání díky původnímu algoritmu.

### Ladění parametrů algoritmu

Jako výchozí hodnoty parametrů cílové funkce algoritmu byly použity simulačně ověřené hodnoty popsané v 5.2. Takto nastavený algoritmus vykazoval dobré chování z hlediska vyhýbání se překážkám v prostoru s hustější koncentrací překážek a v uzavřeném prostoru, avšak v otevřeném prostoru měl tendence po objetí překážky volit trajektorie s větší rychlostí, které vedly k výpadku osoby ze zorného pole UWB systému a zastavení vozidla.

Toto chování v otevřených prostorech vykazovaly i další zkoušené konfigurace ať už kvůli volbě větší rychlosti, či kvůli dodržení bezpečné vzdálenosti od překážek. Toto chování je možné do určité míry ovlivnit výrazným zvětšením váhy pro směřování k cíli. To ale může někdy vést na volbu neoptimální trajektorie, případně ke vzniku kolizní trajektorie, kdy už vozidlo není schopno manévrovat mimo překážky. Vhodným řešením by bylo zaznamenávat předchozí pozice osoby a při výpadku ze zorného pole nastavit tuto pozici jako cílovou. Problém však nastával díky velké fluktuaci úhlových údajů UWB systému a také občasným nesmyslným údajům po výpadku ze zorného pole. Díky tomu vykazovalo DWA poměrně nepředvídatelné chování a od tohoto přístupu bylo upuštěno.

(a) Vozidlo se úspěšně vyhne překážce (b) Osoba se dostane mimo zorné pole UWB systému

Obrázek 6.12: Chování DWA algoritmu v otevřeném prostoru

Pro průjezdy úzkými prostory byla na základě testování vybrána konfigurace parametrů v tabulce 6.1. Tato konfigurace vykazovala velmi dobré výsledky při manévrování v uzavřené chodbě s překážkami.

| Parametry cílové funkce | |
|---|---|
| $\alpha$ | 0.1 |
| $\beta$ | 0.1 |
| $\gamma$ | 0.2 |

Tabulka 6.1: Hodnoty parametrů cílové funkce pro průjezd chodbou

## 6.5 Porovnání DWA a VFH+ algoritmů

### Porovnání na základě experimentu

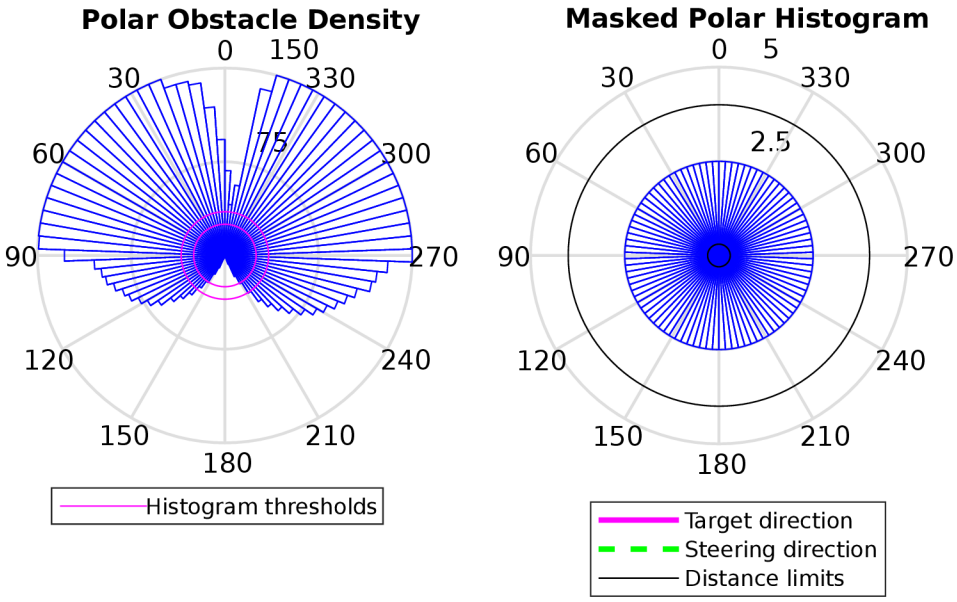
Pro porovnání navrženého DWA algoritmu v konfiguraci z tabulky 6.1 a již implementovaného VFH+ algoritmu byl realizován praktický experiment na otestování schopnosti projet úzkým prostorem. Experiment je vidět na obrázcích 6.13. Šířka prostoru mezi futry dveří a krabicí je v tomto případě 125 cm pro oba algoritmy. Zatímco VFH+ algotimus v tomto případě selhal, a vozidlo zastavilo až díky funkci nouzového zastavení, DWA algoritmus byl schopen manévrovat tak, aby prostorem bezpečně projel. Je nutno podotknout, že vozidlo projíždělo prostorem velmi dlouho na téměř nejnižší možnou rychlost danou DWA algoritmem. To je způsobeno tím, že DWA vyhodnocuje dosažitelnou rychlost na základě toho, zda stihne zastavit před kolizí s překážkou. Bezpečná vzdálenost od středu vozidla k překážce je nastavena na 0,6 m, a tedy bezpečný prostor je v tomto případě roven 5 cm, ve kterých musí algoritmus držet střed vozidla. Právě fakt, že DWA algorimus tímto způsobem volí rychlost vozidla, umožňuje nastavit menší bezpečnou

vzdálenost od překážky. I díky tomu je schopen manévrovat v užších prostorech.

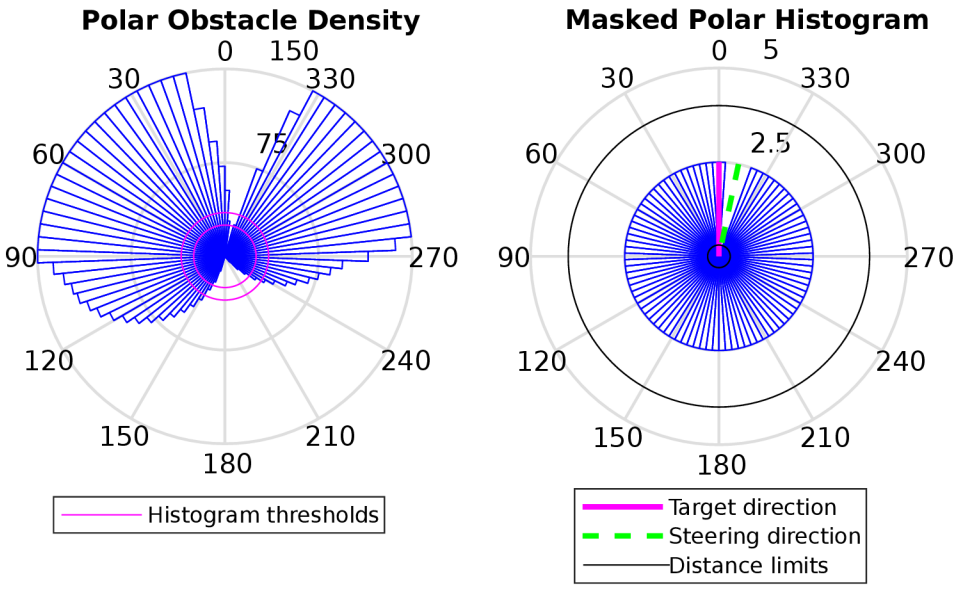
VFH+ byl schopen projet teprve prostorem širokým 165 cm. Výstupy z VFH+ pro obě situace jsou na obrázcích 6.14 a 6.15.

(a) DWA algoritmus (b) VFH+ algoritmus

Obrázek 6.13: 1. Porovnávací experiment




Obrázek 6.14: Výstup VFH+ algoritmu pro průchod 125 cm




Obrázek 6.15: Výstup VFH+ algoritmu pro průchod 165 cm

### Vyhodnocení výsledků

Na základě testování se dá usoudit, že v uzavřených prostorech s více překážkami a v úzkých průchodech vykazuje DWA v testované konfiguraci lepší chování oproti VFH+. To je způsobeno především díky tomu, že rychlost je volena společně s trajektorií a algoritmus je schopen při objíždění překážek zpomalit natolik, aby bylo objetí proveditelné. To ale může způsobovat, že při projíždění některých oblastí je DWA algoritmus pomalejší, než původní řídicí algoritmus s VFH+. Jedná se zejména o pohyb ve volných prostorech s malým množstvím překážek, kdy zpomalení k objetí překážky není nezbytně nutné. Dalším problémem DWA algoritmu je větší kmitání úhlu natočení kol vlivem fluktuace úhlu UWB systému, protože dochází neustále k vyhodnocování nové trajektorie.

Při testování DWA v otevřených prostorech docházelo při objíždění překážky k volbě trajektorie, která vedla k výpadku osoby ze zorného pole UWB systému častěji, než tomu bylo u VFH+ algoritmu. Je nutno podotknout, že chování obou algoritmů se dá výrazně ovlivnit nastavením parametrů a dosáhnout tak různých výsledků při řízení. Protože nastavit parametry tak, aby algoritmy vykazovaly dobré chování ve většině situací je velice složité, měla by se volba algoritmů odvíjet především od zamýšlené aplikace.

# 7 Závěr

Diplomová práce se zabývala rozšířením a vylepšením některých funkcí Follow-me režimu na již existující robotické mobilní platformě s názvem Generace 0 vytvořenou na Technické univerzitě v Liberci.

Pro vylepšení funkce již implementovaného VFH+ algoritmu pro lokální navigaci byl vytvořen a úspěšně otestován algoritmus, který dokáže vytvářet krátkodobý záznam překážek v podobě bodů virtuálních lidarů. Funkčnost algoritmu byla ověřena jak simulačně, tak na reálné robotické platformě. Díky časovému záznamu překážek se podařilo omezit boční kolize vozidla s překážkami, které již byly mimo zorné pole předního lidaru.

V rámci zlepšení odometrie bylo pro natočení předních kol vozidla úspěšně implementováno do řídicí struktury Ackermannovo řízení.

Protože jsou nohy následované osoby zaznamenány lidarem, může se stát že se vozidlo sledovanou osobu díky VFH+ algoritmu snaží nesprávně objet. Z tohoto důvodu byl navržen algoritmus filtrující body lidaru nacházející se v bezprostřední blízkosti neseného UWB ovladače. Tento algoritmus byl úspěšně otestován simulačně, avšak v reálné aplikaci kvůli fluktuaci úhlových údajů z UWB systému selhal.

Z důvodu zlepšení lokální navigace v úzkých prostorech byl navržen pro robotickou platformu DWA algoritmus. DWA algoritmus byl úspěšně otestován na simulaci i na robotické platformě. Při praktických experimentech se povedlo z hlediska manévrování v těsných prostorech předčít původně implementovaný VFH+ algoritmus. Při testování v otevřených prostorech docházelo při některých případech k volbě trajektorie, která vedla k výpadku následované osoby ze zorného pole. V některých případech také docházelo ke kmitání úhlu natočení kol vlivem fluktuace úhlových údajů z UWB systému.

K vylepšení chování obou algoritmů pro lokální navigaci by bylo vhodné provést extenzivní testování jejich parametrů a určit, které mají nejlepší chování pro co nejširší variaci provozovaných prostředí. Dalším zlepšením pro navržené funkce by mohlo být využití pokročilejších filtrovacích metod pro úhlové údaje vystupující z UWB systému, například Kalmanova filtru. Pokud by výstupní úhel odpovídal reálnému postavení osoby, bylo by možné ji vyfiltrovat z dat lidaru a obecně vylepšit volbu trajektorie použitých algoritmů lokální navigace.

# Použitá literatura


[1] Řídí samo. Vyvíjíme vozidlo s vlastními smysly. *T-UNI* [online]. [cit. 2024-02-25]. Dostupné z: https://tuni.tul.cz/rubriky/veda-a-vyzkum/id:134569/ridi-samo-vyvijime-vozidlo-s-vlastnimi-smysly

[2] DC electric motors. *TRANSTECNO* [online]. [cit. 2024-04-05]. Dostupné z: https://www.transtecno.com/wordpress/wp-content/uploads/2015/06/H_EC-DC-electric-motors_210513_0521.pdf

[3] HLAVA, J., Cýrus, J., Kajzr, D. a Kočí, J. *Concept of autonomous vehicle control and data acquisition* [interní materiál]. Technická Univerzita v Liberci.

[4] TIM781-2174101. *SICK* [online]. [cit. 2024-02-28]. Dostupné z: www.sick.com

[5] Terabee Follow-Me User Manual. *Terabee* [online]. [cit. 2024-03--4]. Dostupné z: https://terabee.b-cdn.net/wp-content/uploads/2020/04/Terabee-Follow-Me-user-manual.pdf

[6] Terabee Follow-Me Specification sheet. *Terabee* [online]. [cit. 2024-03-04]. Dostupné z: https://www.terabee.com/shop/mobile-robotics/terabee-follow-me/

[7] Automation PC 910. *B&R* [online]. [cit. 2024-04-05]. Dostupné z: https://www.br-automation.com/cs/produkty/prumyslove-pocitace/automation-pc-910/

[8] Automation Studio Target for Simulink. *B&R* [online]. [cit. 2024-04-05]. Dostupné z: https://www.br-automation.com/en/products/software/modeling-and-simulation/automation-studio-target-for-simulink/

[9] WIECH, Jakub a HENDZEL, Zenon. Robotic Swarm Shape Control Based on Virtual Viscoelastic Chain. Online. In: SZEWCZYK, Roman; ZIELIŃSKI, Cezary a KALICZYŃSKA, Małgorzata (ed.). *Automation 2021: Recent Achievements in Automation, Robotics and*

*Measurement Techniques. Advances in Intelligent Systems and Computing*. Cham: Springer International Publishing, 2021, s. 209--218. ISBN 978-3-030-74892-0. Dostupné z:https://doi.org/10.1007/978-3-030-74893-7_20. [cit. 2024-03-14].

[10] HLAVA, J. *Prostředky automatického řízení II: analogové a číselné regulátory, elektrické pohony, průmyslové komunikační systémy*. Praha: České vysoké učení technické, 2000. ISBN 80-01-02221-8.

[11] BORENSTEIN, J. a KOREN, Y. The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Transactions on Robotics and Automation*. Roč. 7, č. 3, s. 278-288. ISSN 1042296X. Dostupné z: https://doi.org/10.1109/70.88137.

[12] ULRICH, I. a BORENSTEIN, J. VFH+: reliable obstacle avoidance for fast mobile robots. Online. In: *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*. IEEE, 1998, s. 1572-1577. ISBN 0-7803-4300-X. Dostupné z: https://doi.org/10.1109/ROBOT.1998.677362. [cit. 2024-03-14].

[13] SLAM (Simultaneous Localization and Mapping). *MathWorks* [online]. [cit. 2024-04-09]. Dostupné z: https://uk.mathworks.com/discovery/slam.html

[14] Driving Scenario Designer. *MathWorks* [online]. [cit. 2024-04-04]. Dostupné z: https://uk.mathworks.com/help/driving/ref/drivingscenariodesigner-app.html

[15] Mobile Robot Kinematics Equations. *MathWorks* [online]. [cit. 2024-04-04]. Dostupné z: https://uk.mathworks.com/help/robotics/ug/mobile-robot-kinematics-equations.html

[16] Line–line intersection. *Wikipedia* [online]. [cit. 2024-04-05]. Dostupné z: https://en.wikipedia.org/wiki/Line%E2%80%93line_intersection

[17] transformScan. *MathWorks* [online]. [cit. 2024-04-04]. Dostupné z: https://uk.mathworks.com/help/robotics/ref/transformscan.html

[18] Mobile Robotics Simulation Toolbox. *MathWorks* [online]. [cit. 2024-04-04]. Dostupné z: https://uk.mathworks.com/matlabcentral/fileexchange/66586-mobile-robotics-simulation-toolbox

[19] Pure Pursuit Controller. *MathWorks* [online]. [cit. 2024-04-09]. Dostupné z: https://uk.mathworks.com/help/nav/ug/pure-pursuit-controller.html

[20] Ackerman Steering. *XARG* [online]. [cit. 2024-03-10]. Dostupné z: https://www.xarg.org/book/kinematics/ackerman-steering/

[21] Ackermannova podmínka. *autolexicon.net* [online]. [cit. 2024--03-10]. Dostupné z: https://www.autolexicon.net/cs/articles/ackermannova-podminka/

[22] Kinematic Steering. *MathWorks* [online]. [cit. 2024-03-10]. Dostupné z: https://uk.mathworks.com/help/vdynblks/ref/kinematicsteering.html

[23] ARRAS, Kai O.; MOZOS, Oscar Martinez a BURGARD, Wolfram. Using Boosted Features for the Detection of People in 2D Range Data. Online. In: *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE, 2007, s. 3402-3407. ISBN 1--4244-0602-1. ISSN 1050-4729. Dostupné z: https://doi.org/10.1109/ROBOT.2007.363998. [cit. 2024-04-13].

[24] FOX, D.; BURGARD, W. a THRUN, S. The dynamic window approach to collision avoidance. Online.*IEEE Robotics & Automation Magazine*. Roč. 4, č. 1, s. 23-33. ISSN 10709932. Dostupné z: https://doi.org/10.1109/100.580977. [cit. 2024-04-13].

[25] DynamicWindowApproachSample.m zdrojový kód. *GitHub* [online]. [cit. 2024-04-14]. Dostupné z: https://github.com/AtsushiSakai/MATLABRobotics/blob/master/PathPlanning/DynamicWindowApproach/DynamicWindowApproachSample.m

[26] Embedded Coder. *MathWorks* [online]. [cit. 2024-05-11]. Dostupné z: https://uk.mathworks.com/products/embedded-coder.html

[27] Host-Target Communication with External Mode Simulation. *MathWorks* [online]. [cit. 2024-05-11]. Dostupné z: https://uk.mathworks.com/help/ecoder/armcortexa/ug/set-up-and-use-hosttarget-communication-channel.html