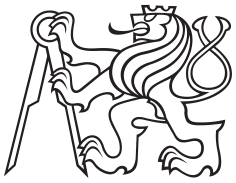
Diplomová práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra řídicí techniky

Přesné dokování mobilního robotu s využitím kamer

Matěj Beránek

Vedoucí: Ing. Karel Košnar, Ph.D.
Obor: Kybernetika a robotika
Studijní program: Kybernetika a robotika
Květen 2019

Poděkování

Rád bych zde poděkoval celému oddělení Inteligentní a mobilní robotiky za to, že jsem mohl tuto práci vytvářet v jejich prostorách a s jejich vybavením. Dále bych chtěl poděkovat za cenné konzultace a připomínky při tvorbě této práce, za což bych chtěl především poděkovat svému vedoucímu Ing. Karlu Košnarovi Ph.D. .

Prohlášení

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

V Praze, dne .....

.....
podpis autora práce

Abstrakt

Práce se zabývá problematikou přesného dokování mobilního robotu. V práci je rozebrána základní teorie této problematiky, dále jsou navrženy dva dokovací algoritmy(algoritmus využívající lokalizaci a algoritmus inspirovaný bearing-only navigací), které jsou experimentálně otestovány na reálném robotu.

Klíčová slova: Turtlebot, autonomní dokování, apriltags, lokalizace pomocí majáků, navigaci podle natočení

Vedoucí: Ing. Karel Košnar, Ph.D.

Abstract

This thesis focuses on problem of precise docking of mobile robot. This thesis summarizes basic theory of this problem, then two algorithms are proposed(localization based algorithm and bearing-only inspired algorithm) and experimentally tested on real robot.

Keywords: Turtlebot, autonomous docking, apriltags, beacon localization, bearing only navigation

Title translation: Precise docking of mobile robot with cameras

Obsah

Obrázky

Tabulky

Úvod

V dnešní době se v průmyslových zónách používá čím dál více robotů. Roboty mají větší pracovní schopnosti než lidé, roboty se nemohou unavit a tím ztratit výkonnost a z dlouhodobého hlediska jsou levnější investicí než lidská pracovní síla. Díky těmto pozitivům se velmi rozvíjí vývoj průmyslových robotů, které by měly v průmyslu člověka nahradit. Tento fakt ovšem přináší mnohé problémy, které se u lidské pracovní síly řešit nemusí. Nastane-li například drobná chyba na pracovní lince v podobě špatně umístěného výrobku, lidský pracovník si s tímto problémem je schopen poradit. S roboty to ale vždy nemusí být takhle jednoduché.

Dnes již existují robotické senzory, které si běžný člověk ani nedokáže představit. Roboty díky nim získávají více a více dat o okolním prostředí, které následně zpracují - často s vyšší účinností než obyčejný člověk. Problémem je ale cena těchto senzorů. Většina lidí v průmyslu se snaží ušetřit všude, kde je to jen možné, proto je důležité zabývat se zjednodušováním a zlevňováním robotů. Je nutné si položit otázky: Je potřeba používat takhle drahý robot s těmito senzory na tuhle úlohu, nebo existuje i jiná, levnější, cesta? Je možné, že by tuto úlohu zvládly i jednodušší a levnější roboty?

Nastiňme si jeden konkrétní případ, například ve skladu, kde jsou využita robotická ramena pro nakládání věcí z regálů, a robotické vozíčky, které tyto naložené věci převážejí. Představme si, že sklad je rozlohou obrovský a nakládacích ramen se zde vyskytují stovky. Vozíčků, které jezdí po skladu, jsou ale pouhé desítky. Samozřejmě je možné mít drahé senzory na vozíčkách i na ramenech, zjednodušila by se tím práce robotů a zamezilo by se tak mnoha chybám. Problém je ale v ceně, kterou lze zredukovat omezením počtu senzorů. Pokud si tedy musíme rozhodnout, které roboty budou "chytré"a které budou ty "hloupé", je výhodnější mít co nejméně těch chytrých, dokud

funkčnost celku zůstává stejná. A přesně na tento případ je tato práce zaměřena - hloupá ramena, která nakládají věci neustále na stejné místo, a chytré vozíčky, které vždy dokáží přesně přijet na určené nakládací místo. Práce se především zaměřuje na prozkoumání možností přesného dokování při použití vizuálních senzorů.

V první kapitole je rozebrána teorie zabývající se lokalizací a navigací pomocí kamer. Druhá kapitola popisuje vybrané algoritmy pro dokování. Třetí kapitola popisuje softwarovou implementaci a čtvrtá kapitola prezentuje experimentální výsledky.

Kapitola 1

Úvodní teorie

1.1 Definice úlohy

Cílem práce je navrhnout a otestovat algoritmus dokování robotu s co největší přesností (nejlépe pod 1 cm z hlediska polohy, je-li to možné).

Pro řešení musejí být využity vizuální senzory (tj. kamery). Určení pozice probíhá mimo jiné pomocí navrženého doku, který je vytvořen při řešení úlohy. Pro dok nejsou určena žádná pravidla, lze tedy využít například značek, jednoduchých či složitých tvarů a různých velikostí doku. Robot je uvažován jako všesměrový podvozek (lze pohybovat robotem ve směrech os x i y).

Výsledná poloha je pak definována pomocí souřadnic $[x, y, \varphi]$, kde x a y jsou relativní souřadnice vůči poloze doku a φ je orientace robotu vůči doku.

1.2 State of the art

V této části bude zmíněno několik základních postupů přesné lokalizace robotu a jejich navigace v prostoru. Tato práce je vytvářena s myšlenkou využití na všesměrových podvozcích, je tedy nutné přihlédnout k tomuto faktu při vybírání metody navigace robotu. Uvažujeme-li, že algoritmus dokování má fungovat hlavně pro přesné dokování, nikoliv pro globální pohyb například ve

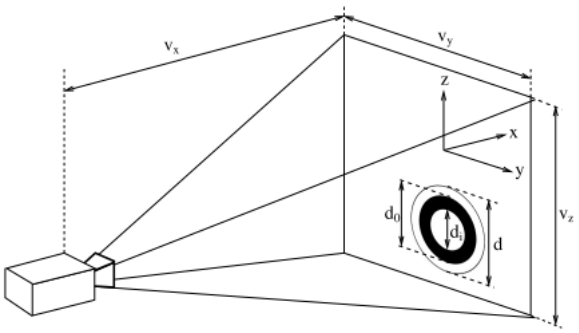
skladu, nemusíme se zaměřovat na algoritmus zabývající se reakcí na možné překážky. V této situaci má všesměrový podvozek obrovskou výhodu, nejdříve můžeme vyřešit orientaci robotu a potom můžeme řešit jeho pohyb v prostoru, jelikož je tento pohyb nezávislý na orientaci robotu.

1.2.1 Lokalizace robotu z vizuálních dat

V dnešní době se pro lokalizaci robotů z vizuálních dat používá několik přístupů.

Předem známé prostředí

Pokud má být robot lokalizován v místech, která jsou předem známá, vyplatí se použít značky. V anglicky psaných textech se těmto značkám říká hlavně majáky (beacons). Pod tímto majákem si člověk může představit cokoliv, co lze jednoduše a přesně lokalizovat v obraze kamery. Majákem tedy může být například papír s tištěnou značkou, mnohobarevný objekt či objekt se zvláštním tvarem. Poté si stačí napsat detektor, pokud je vybrán typ majáku, který ještě nikdo jiný nepoužíval. V dnešní době je ale vyvinuto několik velmi dobře fungujících detektorů, takže existuje možnost využití již vytvořených detektorů. V těchto případech se převážně jedná o tištěné majáky jako například AprilTags [10] a Whycon [16].



Obrázek 1.1: Ukázka detekce značky Whycon v obraze. Převzato z : [16]

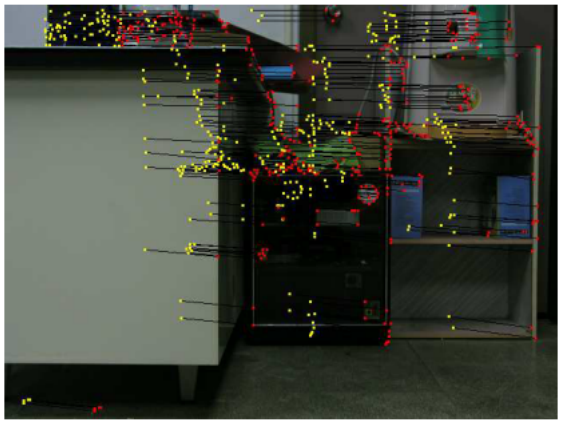
Problémem tohoto přístupu lokalizace je nemožnost využití v neznámých prostorech. K lokalizaci pomocí majáků totiž potřebujeme co nejpřesnější informace o jejich poloze v globálních souřadnicích, ke kterým je následně vztažena měřená poloha robotu. Přesnost této lokalizace závisí na kvalitě informací o jednotlivých majících, na kvalitě vizuálního systému a na kvalitě

zvoleného detektoru těchto majáků.

Pro lokalizaci pomocí majáků je možné dokonce využít i omnikamery. Výhoda oproti soustavě kamer je v tom, že je použita pouze jedna kamera, všechny detekované informace jsou tedy vztaženy ke stejnému bodu v souřadné soustavě. Problémem je ale vysoká cena takovéto kamery a velmi často je složité zpracování jejího obrazu. Samotný obraz pak při těchto úpravách obsahuje méně informací než obraz z obyčejné kamery, je proto lepší používat jednoduché vzory na značkách nebo jednoduché tvary majáků.

Neznámé prostředí

Pro lokalizaci v neznámém prostředí bylo nutné vymyslet jiné postupy, které nespoléhají na jakékoliv apriorní informace. Mnohem jednodušším postupem je využití dálkoměrů, z jejichž informací o vzdálenosti objektů je možné tvořit mapy prostředí. Problémem těchto senzorů je jejich vysoká cena, proto se vědci zabývají i jinými postupy, kterými by mohlo být neznámé prostředí mapováno za použití jiných senzorů.



Obrázek 1.2: Příklad hledání obrazových příznaků. Převzato z : [2]

Existuje velké množství algoritmů, které hledají v obraze z kamery tzv. obrazové příznaky. Těmito příznaky mohou být různé body, hrany a objekty v obraze. Obrazové příznaky ale nejsou dostatečnou informací, se kterou by bylo možné vytvořit mapu. Využijeme-li ale stereokamery, je možné dopočítat informace o vzdálenosti jednotlivých příznaků a nahradit si tak dálkoměr. Důležitým aspektem tohoto přístupu je kvalitní algoritmus, který hledá příznaky v obraze. Je totiž nutné zajistit, aby jednotlivé kamery ve stereokameře identifikovaly stejné příznaky. Poté je nutné spárovat tyto příznaky, díky čemuž jsme schopni získat informace o jednotlivých vzdálenostech. Výhodou vizuálního přístupu je možnost zachování informace o jednotlivých příznacích

v čase, tzn. pokud se robot pohybuje, je možné stejné příznaky porovnávat s předchozími informacemi a lépe odhadovat polohu.

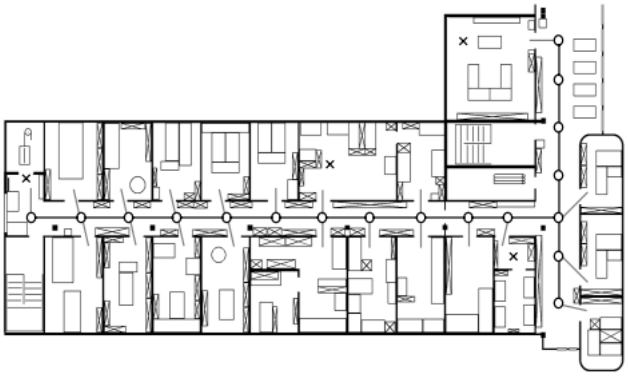
Reálné využití

Oba výše popsané generalizované přístupy jsou využívány v reálných experimentech. Například články [1], [4], [7] a [2] se zabývají lokalizací robotů, ať už pomocí majáků, nebo pomocí obrazových příznaků.

V článku [1] autoři představují zlepšený algoritmus lokalizace pomocí triangulace pro zašuměná data využívající majáky. Triangulace je relativně jednoduchým geometrickým přístupem, kterým lze počítat polohu robotu, pokud známe údaje o hledaných majácích. Problém nastane při nepřesných datech, kdy naměřená data nevytvoří přesný trojúhelník a proto je nutné počítat přeurčené soustavy rovnic a minimalizovat chybu odhadu polohy. Navržený algoritmus je následně testován ve velkém množství situací v simulátoru i s reálnými daty. Výsledky pokusů shrnují do grafů a chyb, které se pohybují v jednotkách centimetrů.

Autoři [7] měřili kvalitu lokalizace robotu pomocí kamer a jejich vlastního detektoru majáků. Majáky jsou v tomto případě plakáty na zdech místností (jedná se pouze o experiment uvnitř místnosti). Lokalizace poté probíhá pomocí rohů plakátů (quadrangle), což je velmi podobný postup, jako výše zmíněná triangulace. Výsledky tohoto experimentu jsou velmi podobné předchozím pokusům, jelikož se chyby lokalizace pohybují v jednotkách centimetrů.

V [4] je rozebrán algoritmus využívající obrazové příznaky pro lokalizaci v lokální mapě. Tato mapa je složena pouze z rovných úseků, zjednodušeně si ji lze představit jako množinu čar pro každou lokální metrickou mapu. Zároveň je zde využita topologická mapa pro delší jízdu (např. mezi místnostmi). Algoritmus tedy chytře přepíná mezi topologickou a metrickou mapou pro lokalizaci. Opět byl algoritmus testován pouze uvnitř budovy. Výsledky algoritmu uvedené v článku jsou velmi uspokojivé, v experimentech ujel robot přes 1 km a průměrná chyba lokalizace v jeho konečném bodě se pohybovala kolem 1 cm. Je nutné poznamenat, že obě mapy prostředí jsou předem známé, nejedná se tedy o SLAM(Simultaneous localization and mapping) ale pouze o lokalizaci.



Obrázek 1.3: Spojení topologické a lokální mapy. Převzato z : [4]

Článek [2] pojednává o algoritmu lokalizace pomocí obrazových příznaků. V tomto algoritmu není potřeba mapa. Lokalizace probíhá pomocí porovnání příznaků v různých časových okamžicích. Nejdříve je nutné určit páry obrazových příznaků, které si odpovídají. Po tomto kroku už je lokalizace velmi podobná algoritmu ICP, který využívá bodové mraky (pointcloud). Tento algoritmus byl testován ve vnitřním i venkovním prostředí. Kvalitu algoritmu ovlivňuje hlavně hledání odpovídajících dvojic příznaků. Výsledky uvedené v článku jsou horší, než v předchozích článcích, chyba se pohybuje kolem 9 cm. Zároveň je to ale jediný přístup, který nevyužívá apriorní informaci o okolí.

1.2.2 Navigace robotu v prostředí

Jak již bylo zmíněno výše, v této práci je uvažován všesměrový podvozek. Zabýváme-li se navigací, s největší pravděpodobností pracujeme se souřadnicovými systémy. Pro navigaci všesměrového podvozku lze využít rychlosti v jednotlivých směrech os, tedy v_x a v_y, nezávisle na sobě. Díky tomuto faktu je navigace všesměrového podvozku mnohem jednodušší, než navigace jiných podvozků. Pokud je k dispozici jiný podvozek (diferenciální, ackermannův nebo např. podvozek automobilu), je nutné přepočítat rychlosti v_x a v_y na rychlosti $v_{\texttt{linear}}$ a $v_{\texttt{angular}}$.

Stejně jako předchozí kapitola o lokalizaci byla rozdělena na dvě větší sekce, i navigaci lze podobným způsobem rozdělit. Navigaci lze rozdělit na navigaci s mapou (navigace plánováním) a na reaktivní navigaci.

Navigace s mapou

Pro navigaci s mapou je nutné vytváření mapy a lokalizace v ní. Pro tento typ navigace existuje několik algoritmů, které hledají cesty v mapách (např. A-Star, DFS (hloubkové prohledávání grafu), BFS (šířkové prohledávání grafu) nebo třeba Dijkstrův algoritmus). Dále záleží na typu map, které jsou použity, aby byl pro ně vybrán nejvhodnější způsob hledání cest. Dokáže-li se robot lokalizovat v mapě, je možné ho navigovat podle vypočítané trajektorie na správné místo.

Pro převedení trajektorie do reálných příkazů robotu lze využít například přístup následování mrkve (follow the carrot), kde je vždy vybrán nejbližší bod z trajektorie k aktuální poloze robotu. Jakmile se robot přiblíží na dostatečnou vzdálenost, je automaticky vybrán další bod v pořadí, na který je robot navigován. Tuto vzdálenost určuje programátor, neměla by být příliš velká (trajektorie robotu bude velmi rozdílná oproti trajektorii vypočítané) ani příliš malá (robot není přesný a v jednom bodu může vzniknout uzel, kolem kterého se robot bude neustále pohybovat a nikdy se nedostane dostatečně blízko). Tímto přístupem by robot měl přesně následovat vypočítanou trajektorii (přesnost zde závisí na kvalitě lokalizace a na zvolené vzdálenosti pro přepínání bodů).

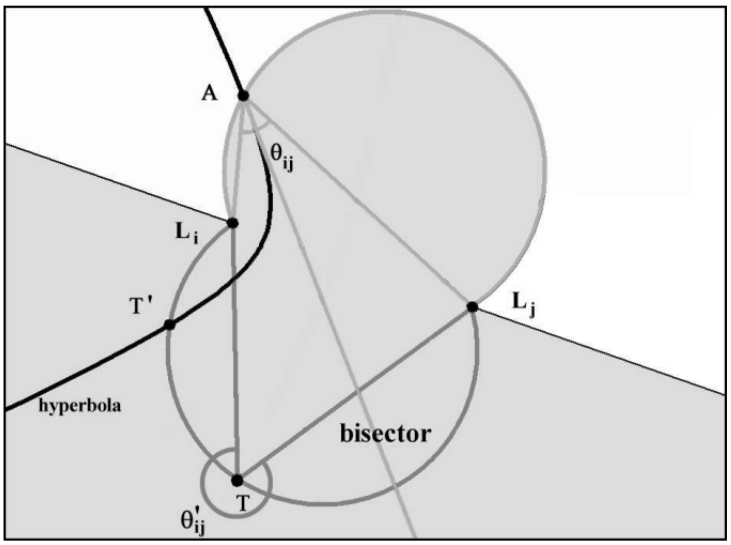
Reaktivní navigace

Pro reaktivní navigaci není nutná mapa prostředí ani lokalizace v ní. Reaktivní navigace je založena na momentálním stavu robotu a prostředí, ve kterém se robot pohybuje. Dále musí být určena pravidla, podle kterých se robot má pohybovat. Příkladem této navigace je například následování zvoleného objektu (pomocí kamer i pomocí senzorů vzdálenosti), jízda na konkrétní místo určené pomocí značek v okolí nebo pohyb po prostředí pomocí pravidla pravé (levé) ruky (robot se drží v určité vzdálenosti od stěn a překážek, pokud je moc vzdálený, zatáčí doprava (doleva)). Příklad této navigace bez mapy je v článcích [6], [11], [12] a [13].

Při hledání informací o vizuální navigaci robotu bez mapy jsem se pak především zaměřil na tzv. "bearing-only"navigaci (jedinou informací z kamer je úhel k různým značkám). Články [6], [12] a [13] se zabývají různými implementacemi tohoto přístupu. Bearing-only navigace byla vybrána, protože informace o úhlech ke značkám je nejpřesnější informací, kterou jsme schopni z kamer získat (bavíme-li se o vypočítání úhlů rovnou z polohy značky v obraze), záleží zde hlavně na kvalitní kalibraci kamer.

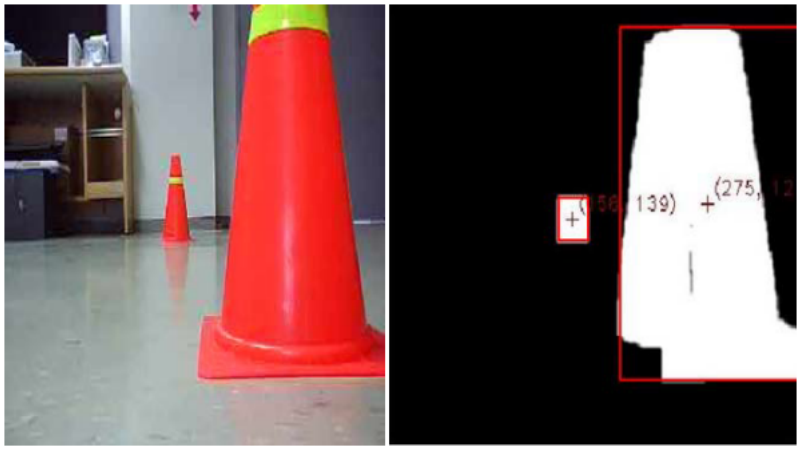
Reálné použití

V článku [6] je popsáno řešení problému navigace, pokud jsou v prostoru alespoň dvě značky a robot je schopen měřit úhel mezi nimi. Algoritmus, jenž je navržen, využívá jízdu po osách úhlů, díky kterým se robot dostane na oblouky, ve kterých jsou značky vidět pod danými úhly. Pokud se počet značek zvýší, cílová poloha je jednoznačně určena. S tímto přístupem bylo dosaženo přesnosti kolem 3 cm v koncových polohách robotu. Nutné podotknout, že experimenty byly provozovány s objektivem se záběrem 360° (rybí oko), nicméně je v článku uvedeno, že data byla měřena s přesností až 0.1°.



Obrázek 1.4: Ukázka teorie jízdy po ose úhlu. Převzato z : [6]

Článek [11] neimplementoval navigaci bearing-only, nicméně také pojednává o navigaci bez mapy. Jedná se o navigaci všesměrového robotu pomocí dopravních kuželů. Autoři si napsali vlastní detektor, který dokáže tyto kužely rozpoznávat. Algoritmus je potom založen na rozpoznávání jednotlivých kuželů a jejich barev. V článku jsou rozebrány dvě barvy kuželů - červená (značí pouze záchytný bod trajektorie) a zelená (koncový bod trajektorie). Algoritmus je v článku popsán pouze pomocí stavů :



Obrázek 1.5: Detektor kuželů. Převzato z : [11]

- Robot nic nevidí - otáčí se kolem své osy dokud nezachytí kužel
- Robot vidí kužel daleko od sebe - robot se vydá směrem ke kuželu
- Robot vidí kužel dostatečně blízko sebe - je-li tento kužel zelený, robot je na místě určení, je-li kužel červený, robot ho objede a hledá další kužel v obraze

V článku bohužel není popsáno žádné vyhodnocení přístupu z hlediska přesnosti nebo spolehlivosti. Algoritmus zde navržený je každopádně zajímavým a velmi jednoduchým přístupem k navigaci pomocí značek.

V [13] autoři navrhují vylepšený algoritmus v porovnání s [6]. Algoritmus není závislý na používání značek. Pro navigaci je zde využito obrazových příznaků, které splňují podmínku, že jsou viditelné z cílové pozice (musejí být opět minimálně 3). Tyto příznaky jsou poté nalezeny i z aktuální pozice robotu a využity jako značky. Algoritmus je pak velmi podobný, jelikož se také jedná o bearing-only navigaci. Jsou vypočítány úhly k jednotlivým značkám (potažmo příznakům) a navigace je pak vypočítána z výslednice jednotlivých os úhlů. V článku je zároveň důkaz konvergence tohoto přístupu navigace.

[12] navazuje na předchozí přístup a obohacuje ho o pozorovatele stavů. Algoritmus je opět založený na bearing-only přístupu, přidává však pozorovatele stavů, kterými jsou vzdálenost k cíli, orientace robotu a natočení vůči cíli. Tento algoritmus opět pracuje se třemi značkami, předchozí stavy je schopen pozorovatel počítat z informace o úhlech k jednotlivým značkám. Navigace je zde zjednodušena díky informacím o orientaci, natočení k cíli a vzdálenosti robotu k cíli. Pro pozorovatele byl použit rozšířený kalmanův filtr (EKF). Tento algoritmus dosahuje vynikajících výsledků, robot je tímto schopen dosáhnout cílových poloh s přesností desetin centimetrů. Důležité je však zmínit, že algoritmus byl testován pouze v simulacích (i s reálnými daty), nebyl tedy vyzkoušen s praktickým robotem.

Kapitola 2

Teoretický popis řešení problému

Po domluvě s vedoucím práce byly vytvořeny dva algoritmy pro dokování robotu. Jeden algoritmus je založen na lokalizaci (bez mapy) a jízdě podle lokalizovaných souřadnic, druhý je algoritmem implementujícím bearing-only navigaci.

2.1 Vstupní data

Za vstupní data do obou algoritmů jsou považovány informace o úhlech, pod kterými jsou vidět značky v obraze kamery. Tato data jsou totiž považována za nejpřesnější informace, které jsme schopni z kamer získat (ostatní informace jsou například vzdálenost podle velikosti značky v obrázku). Máme-li informaci o poloze značky v obrázku a ohniskovou vzdálenost kamery, je velmi jednoduché vypočítat úhel, pod kterým je značka viditelná. Lze pozorovat v 2.1, úhel lze vypočítat rovnicí

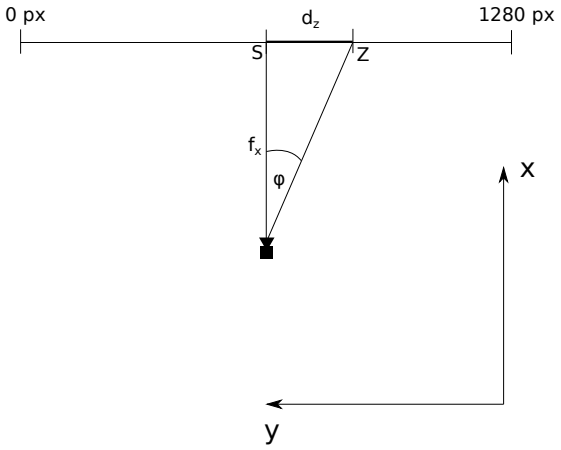
$$\varphi = \texttt{atan2}(-d_z, f_x), \tag{2.1}$$

je-li

$$d_z = Z - S \tag{2.2}$$

a chceme-li úhel s orientací podle os uvedených vpravo na 2.1.

Dále je uvažována co nejpřesnější kalibrace kamer a co nejpřesnější informace o umístění jednotlivých kamer.



Obrázek 2.1: Nákres geometrie získání úhlu z kamery

2.2 Algoritmus s lokalizací

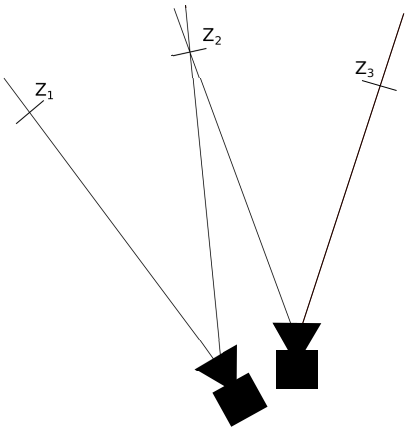
2.2.1 Triangulace

Triangulace je pro lokalizaci pomocí vizuálních senzorů jednou z nejčastějších metod. Triangulace, jak už slovo napovídá, je metodou, která využívá měření úhlů z kamer, díky kterým je možné dopočítat konečný bod. V praxi není nikdy možné určit jeden konkrétní bod, proto je nutné na konci metody implementovat metodu řešící minimalizaci chyby výpočtu (třeba metoda nejmenších čtverců). Například článek [3] se zabývá touto metodou lokalizace a jejím zlepšením. Problémem triangulace je závislost na značkách a v základním znění i na pořadí jednotlivých značek (je nutné mít úhly orientované). Při triangulaci zároveň mohou nastávat problémy s různými kvadranty souřadnic značek, které je nutné ošetřit.

2.2.2 Navržený algoritmus

Prvním přístupem k dokování je algoritmus, kterýje založen na lokalizaci pomocí značek umístěných na doku. Jedná se pouze o 2D lokalizaci, zajímá nás tedy jen x, y a θ. Přestože byla čerpána inspirace z mnoha článků, jen velmi malá množina se zabývala využitím více kamer s malým úhlem záběru. Pro algoritmy mapování pomocí značek využívající triangulaci je velmi důležitá přesnost informací o kamerách (umístění, natočení), kterou (zejména umístění) je při využití více kamer velmi obtížné získat.

Ze vstupních dat, která jsou popsána výše, jsme schopni zkonstruovat rovnice přímek procházející jednotlivými kamerami se směrem podle určených úhlů (2.2).



Obrázek 2.2: Přímky jednotlivých kamer ke značkám

Rovnice přímky je základní

$$y = ax + b, \tag{2.3}$$

kde je možné vypočítat

$$a = \tan(\varphi + \alpha) \tag{2.4}$$

a

$$b = y_C - ax_C, \tag{2.5}$$

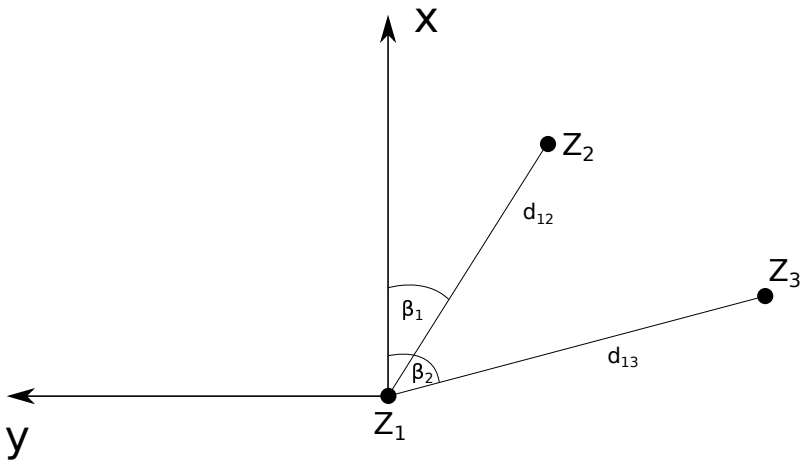
pokud φ je úhel vypočítaný v rovnici 2.1, α je natočení konkrétní kamery a x_C a y_C jsou souřadnice kamery, kterou má přímka procházet.

Po vytvoření těchto přímek máme soustavu rovnic, která může mít až dvakrát více proměnných, než je počet rovnic (pokud každá značka je viděna právě jednou). Proto je nutné získat apriorní znalost o doku a poloze značek vůči sobě. Pokud je tato informace známá, je možné všechny značky vztáhnout ke konkretnímu bodu v prostoru (ideální vztáhnout k jedné ze značek), čímž se tato soustava rovnic stane přeurčenou soustavou rovnic se třemi proměnnými (x_D, y_D a θ). Apriorní znalost o doku je možné psát jako parametry d (vzdálenost značky od hledaného bodu) a β (úhel od hledaného bodu) 2.3. Pak je možné zapsat souřadnice každé značky jako

$$y_Z = y_D + d\sin(\beta) \qquad x_Z = x_D + d\cos(\beta). \tag{2.6}$$

Po vyjádření proměnných je možné každou rovnici přímky z jedné kamery k jedné značce zapsat jako

$$y_D + d \cdot \sin(\theta + \beta) = a(x_D + d \cdot \cos(\theta + \beta)) + b. \tag{2.7}$$



Obrázek 2.3: Informace o doku

Další postup je řešen v materiálu [9]. Pro řešení soustavy nelineárních rovnic je využita nelineární metoda nejmenších čtverců (NLSQ). Tato metoda spočívá na principu lineární regrese, je nutné vyjádřit si Jakobián soustavy rovnic a z ní vypočítat řešení. Metoda potřebuje původní odhad a poté iteruje k řešení. Nejprve je nutné zapsat soustavu rovnic jako

$$h(x) = z \tag{2.8}$$

kde

$$h(x) = y_D + d \cdot \sin(\theta + \beta) - a(x_D + d \cdot \cos(\theta + \beta)) \tag{2.9}$$

a $z = b$. Obě proměnné jsou nyní již vektory se všemi rovnicemi ze soustavy rovnic. Zároveň x je již vektorem hledaných proměnných (x_D, y_D a θ). Po získání těchto proměnných je nutné vypočítat Jakobián nelineární funkce $h(x)$.

$$\nabla H_{\hat{x}} = \frac{\partial h}{\partial x} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_m} \\ \vdots & & \vdots \\ \frac{\partial h_n}{\partial x_1} & \cdots & \frac{\partial h_n}{\partial x_m} \end{bmatrix}. \tag{2.10}$$

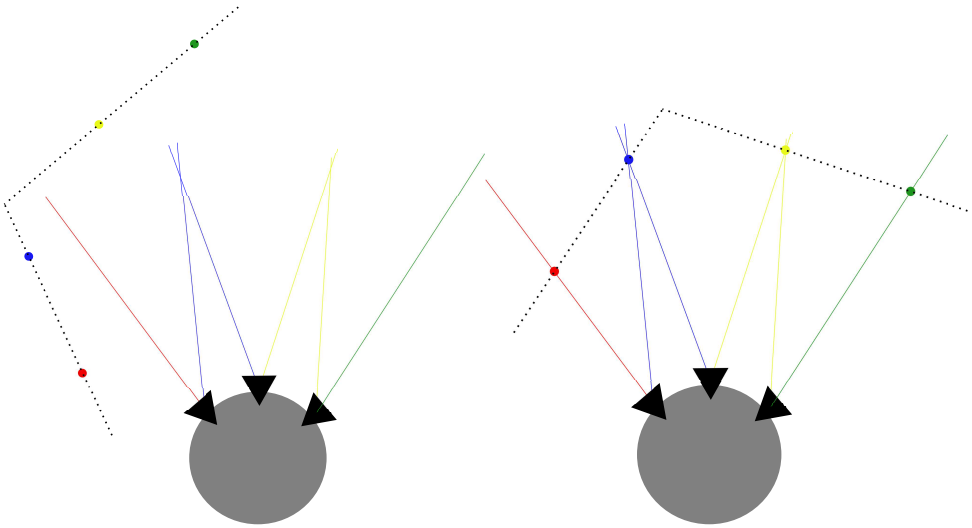
Díky tomuto Jakobiánu jsme schopni vypočítat

$$\delta = (\nabla H_{\hat{x}}^T \nabla H_{\hat{x}})^{-1} \nabla H_{\hat{x}}^T (z - h(\hat{x})), \tag{2.11}$$

jenž se aplikuje jako přírustek k původnímu odhadu řešení

$$\hat{x} = \delta + \hat{x}. \tag{2.12}$$

Tento algoritmus se cyklicky opakuje, dokud není splněna končící podmínka přesnosti řešení. Podmínka má většinou tvar $(h(\hat{x}) - z)^T \cdot (h(\hat{x}) - z) < \texttt{konst.}$, kde konstanta je ta kýžená přesnost řešení.



(a) : Start NLSQ **(b) :** Konec NLSQ

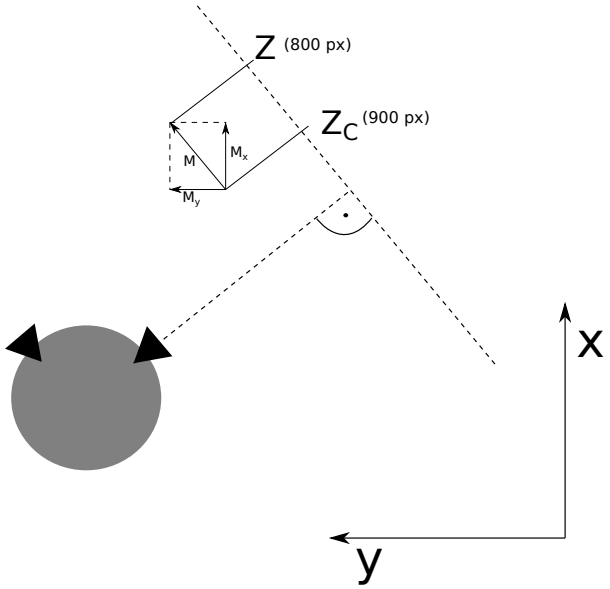
Jako počáteční odhad je vhodné použít předchozí řešení této lokalizace, pro úplně první lokalizační cyklus je možné použít prakticky cokoliv, je ale možné, že první cyklus zabere delší čas.

Pro navigaci může být v tomto přístupu použit jakýkoliv regulátor počínaje P-regulátorem. Nejvhodnější by bylo využít minimálně PI regulátor, aby byla zajištěna nulová odchylka od reference.

2.3 Algoritmus bez lokalizace

Jako druhý algoritmus byl zvolen jednoduchý přístup inspirovaný bearing-only navigací. Jako vstupní data jsou využita podobná data, která jsou popsána výše. Tento algoritmus však nevyužívá informace o úhlech ke značkám, ale samotnou polohu značky v obraze.

Problémem tohoto přístupu a klasického bearing-only přístupu je absence jednoho senzoru se širokým záběrem snímání. Většina bearing-only navigací využívá širokoúhlé kamery, algoritmy jsou konstruovány tak, že se nemusí řešit viditelnost značek, podle kterých je následně robot naváděn. V našem rozvržení kamer je problémem to, že nejsou všechny na stejném místě, tudíž nemají jeden střed, ke kterému by bylo možné úhly mezi jednotlivými značkami vztáhnout. Proto by bylo samotné měření úhlů mezi značkami nepřesné a složité na výpočet.



Obrázek 2.5: Ukázka získání momentu z jedné značky

Navržený algoritmus funguje na principu momentů, které skládáním směřují robot do cílové polohy. Z obrázku 2.5 můžeme vidět princip výpočtu těchto momentů. Cílová poloha robotu musí být zapsána jako pozice Z_C jednotlivých značek v jednotlivých kamerách (v pixelech). Při aktuální poloze změříme polohu Z značky v obraze a zjistíme, jestli se liší od požadované polohy Z_C. Moment lze jednoduše vypočítat jako

$$M = c(Z_C - Z), \tag{2.13}$$

kde c je konstanta robotu, která přepočítává pixely na data, která lze rozumně využívat jako rychlosti robotu.

Jelikož víme, že poloha obrazu kamery je kolmá na její směr, můžeme vypočítat úhel momentu jako

$$\varphi_M = \alpha + 90° \tag{2.14}$$

(α je úhel natočení kamery). Díky definici M a úhlu φ_M je zajištěn správný směr momentu pro $Z_C > Z$ i pro $Z_C < Z$. Pomocí znalosti o úhlu φ_M je jednoduché získat jednotlivé části M ve směru os M_x a M_y

$$M_x = M\cos(\varphi_M) \qquad M_y = M\sin(\varphi_M). \tag{2.15}$$

Tyto složky momentu lze rovnou považovat za rychlosti v_x a v_y pro všesměrový podvozek. Pokud není k dispozici všesměrový podvozek, lze si úlohu zjednodušit a považovat rovnosti $v_x = v_{\texttt{linear}}$ a $v_y = v_{\texttt{angular}}$.

Tento algoritmus nespoléhá na lokalizaci ve 2D prostoru, pouze se lokalizuje pomocí jednotlivých obrazů kamer. Z hlediska teorie by měl i tento

algoritmus být velmi přesný, pokud bude k dispozici dostatečný počet značek a značky budou rozumně rozmístěny v prostoru (stejně jako u jiných bearing-only navigací).

Kapitola 3

Implementace algoritmů

V této části je popsáno softwarové složení řídicí jednotky robotu. Operačním systémem počítače je Ubuntu 16.04. Ke komunikaci mezi jednotlivými částmi hardwaru (podvozek, kamery, počítač) je použit systém ROS (Robot Operating System).

3.1 Robot Operating System

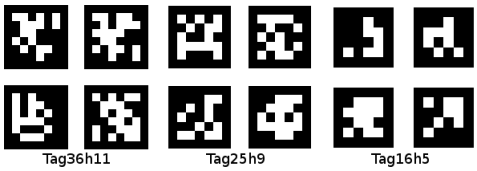
ROS je robotickým operačním systémem, který zajišťuje přenos dat mezi jednotlivými komponentami sítě [14]. Systém funguje na principu jednotlivých uzlů (nodes), které pracují nezávisle na sobě ve vlastních procesech. Tyto uzly mezi sebou mohou komunikovat pomocí komunikačních témat (topics). Každé komunikační téma musí mít předepsaný typ zpráv (messages), které se budou skrz toto téma posílat. Každý uzel má následně definováno, ke kterým komunikačním uzlům se aktivně nebo pasivně připojí. Aktivní připojení (advertise) umožňuje uzlu vysílat zprávy (se správným typem) po komunikačním tématu, nemůže z něj však zprávy číst. Pasivní připojení funguje naopak, uzel nemůže zprávy tvořit a posílat, ale může zprávy poslouchat a zpracovat. ROS dále nabízí okamžité zpracovávání zpráv pomocí funkcí, které se zavolají pokaždé, když se na daném tématu objeví nová zpráva.

Velkou výhodou ROSu je jednoduchost použití. ROS podporuje dva jazyky - C++ a Python. Oba tyto jazyky jsou tzv. high-level programovací jazyky a pro průměrného uživatele jsou přijatelnější, protože se nemusí starat o věci jako je například správa paměti. Zároveň ROS umožňuje velmi jednoduché

vytvoření sítě mezi několika počítači, čímž se značně zjednodušuje vytváření komunikace mezi více roboty.

AprilTags

Pro detekci v obraze byly vybrány značky AprilTags. Značky AprilTags [10] byly využity kvůli doporučení vedoucího práce a dlouholeté zkušenosti s těmito značkami. Značky fungují na podobném principu jako QR kódy. Značky poskytují informaci o svém stavu ve 3D prostoru (poloha a natočení). Detektor je zároveň kódově relativně transparentní a není těžké si detektor upravit podle vlastních potřeb. Důležitou vlastností detektoru je detekce všech rohů značky v obraze, díky čemuž lze vypočítat střed značky. Jak je zmíněno v předchozí kapitole, poloha značky v obraze a následné počítání úhlu by měla být nepřesnější informace, kterou je možné z kamery získat.



Obrázek 3.1: Ukázka značek AprilTags. Převzato z : [18]

3.2 Zdrojové kódy

V této části bude popis jednotlivých zdrojových kódů. Algoritmy pro dokování byly vytvořeny pomocí několika uzlů kvůli opakovatelné použitelnosti a přehlednosti. V hierarchii systému ROS se používají balíčky, které obsahují zdrojové kódy. Každý balíček je většinou zaměřen na jednu konkrétní úlohu, v každém balíčku pak může být vytvořeno několik spustitelných souborů které se chovají jako jednotlivé uzly v síti systému ROS. Jako příklad balíčku lze uvádět balíček pro obsluhu podvozku TurtleBot2, který obsahuje několik různých spustitelných souborů, ze kterých lze vybírat požadovanou funkčnost (uzel pro spuštění pouze motorů, uzel pro spuštění motorů a vestavěných senzorů atd.). Při tvorbě této práce byl vytvořen jeden balíček obsahující uzly algoritmů dokování, jeden balíček obsahující obsluhu kamer a jejich čtení, který byl převzat z minulých let a upraven pro potřebu práce.

3.2.1 Balíček camera_node

Tento balíček pro obsluhu kamer Basler vznikl již při předchozí spolupráci na jiném projektu [17]. Pro potřebu této práce byl rozšířen o několik důležitých částí (například přidání možnosti zadání parametrů z kalibrace kamery).

Složka **src** obsahuje všechny zdrojové kódy potřebné pro obsluhu kamery, detekci značek AprilTags v obraze a následné odeslání naměřených dat na komunikační téma (topic). V souboru **ImageCapture.cpp** se nachází kód obsluhující kameru a snímání jejího obrazu, soubor **TagRecognition.cpp** obsahuje detektor značek AprilTags a soubor **CameraNode.cpp** je hlavním programem implementujícím komunikaci se systémem ROS (tedy odesílání zpráv).

Algorithm 1: Princip funkce algoritmu na obsluhu kamer

```
1 initialization (ROS, TagDetection, ImageCapture);
2 while ROS ok, camera ok do
3 |   ImageCapture ← get new image;
4 |   apply calibration parameters on image;
5 |   TagDetection ← detect all tags in the image;
6 |   Send tags info on ROS topic;
7 end
```

Jelikož není předem známé, kolik značek bude v obraze detekováno, bylo nutné vytvořit vlastní typ zprávy. Definice této zprávy je ve složce **msg**. Soubory pro generování zpráv jsou pouze výčtem proměnných, které se mají ve zprávě vyskytovat, o zbytek se postará ROS.

3.2.2 Balíček docking

Balíček docking obsahuje všechny potřebné kódy pro fungování navržených algoritmů ve složce **src**. První navrhovaný algoritmus je rozdělen na tři soubory a tudíž i tři uzly v systému ROS(**camera_reader.cpp**, **localization.cpp** a **navigation.cpp**), zatímco druhý algoritmus je pouze jediným souborem (**servoing.cpp**).

Zde je stručně popsaná funkce jednotlivých kódů

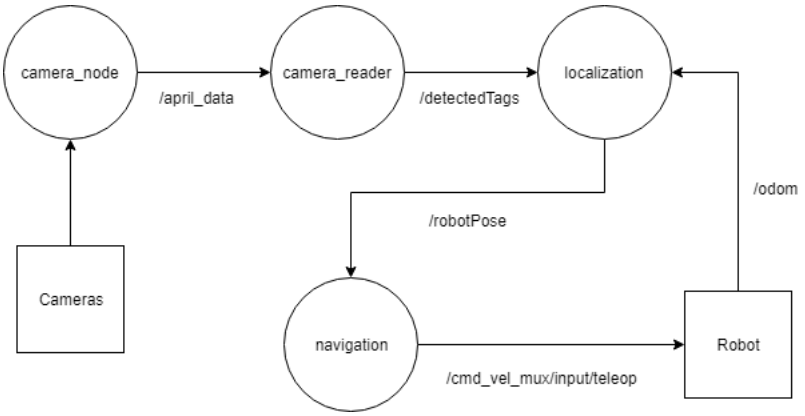
- **camera_reader.cpp** - Tento kód implementuje čtení kamer a zpraco-

vání dat ze všech kamer do kompaktní struktury se všemi potřebnými informacemi. Vstupem jsou data o všech značkách ze všech kamer, výstupem jsou vypočítané úhly z jednotlivých kamer do detekovaných značek

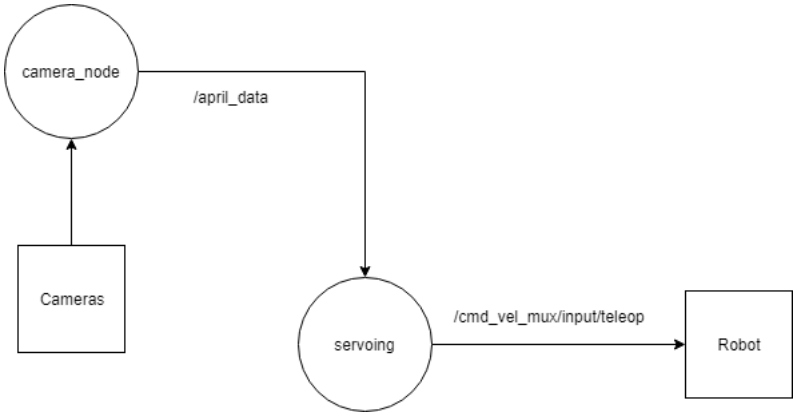
- **localization.cpp** - Tento kód implementuje samotný algoritmus NLSQ 2.2.2, pomocí kterého je dok lokalizován vůči robotu. Vstupem jsou data o úhlech k jednotlivým značkám, výstupem jsou souřadnice polohy bodu, ke kterému je dok vztažený (značka s ID 0).

- **navigation.cpp** - Zde je implementovaná samotná navigace na konkrétní místo. Vstupem je poloha doku a odometrie robotu, výstupem jsou rychlosti $v_{\texttt{linear}}$ a $v_{\texttt{angular}}$ pro řízení robotu.

- **servoing.cpp** - Tento kód je implementací druhého algoritmu. Vstupem jsou nezpracovaná data o detekovaných značkách a výstupem jsou opět rychlosti $v_{\texttt{linear}}$ a $v_{\texttt{angular}}$ pro řízení robotu.



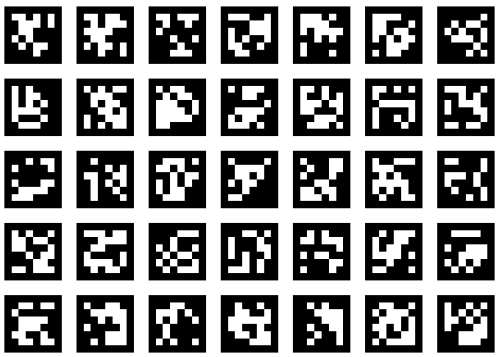
Obrázek 3.2: Schéma propojení algoritmu s lokalizací



Obrázek 3.3: Schéma propojení algoritmu bez lokalizace

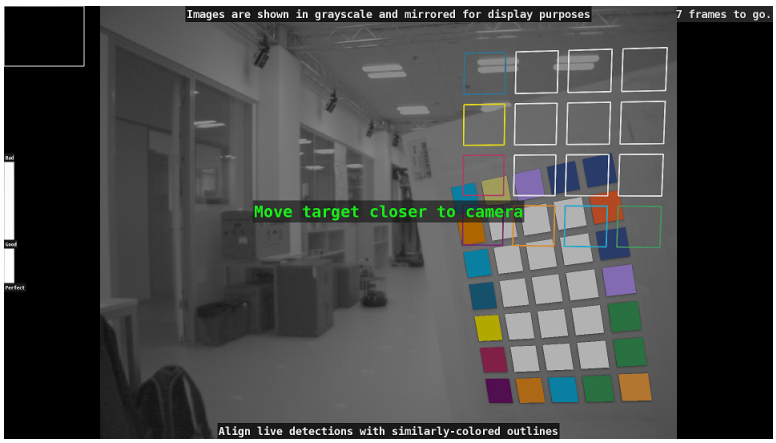
3.3 Kalibrace kamer

Pro kalibraci kamer byl využit software April Camera suite [15]. Tento kalibrátor je vytvořen stejnými autory, jako použitý detektor značek AprilTags. Hlavní unikátností tohoto kalibrátoru je skutečnost, že pro detekci v obraze nevyužívá šachovnici, ale definovanou soustavu značek AprilTags. Tímto vzniká obrovská výhoda oproti ostatním kalibrátorům - detekovaný terč nemusí být v obraze neustále celý. Tento přístup přináší lepší pokrytí okrajů obrazu, čímž se zvyšuje kvalita kalibrace.



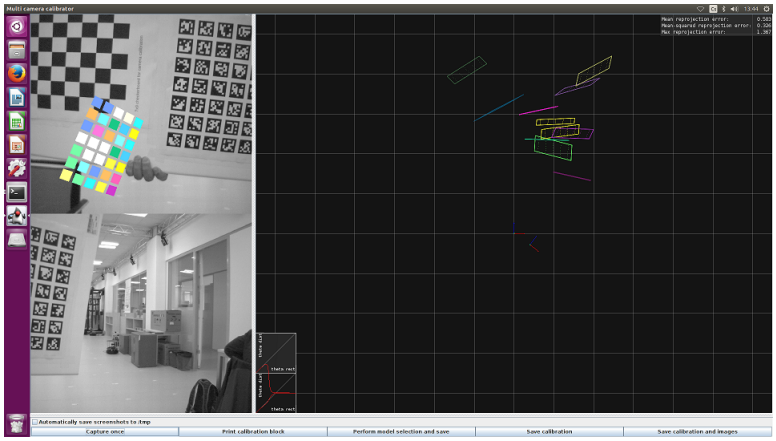
Obrázek 3.4: Ukázka soustavy značek AprilTags. Převzato z : [15]

Samotná kalibrace má vytvořené velmi příjemné uživatelské prostředí. Pokud jsou značky v obraze detekovány, jsou pro uživatele na monitoru obarveny různými barvami. Tyto barvy jsou zároveň využity pro zobrazení ideálního umístění, do kterého je nutné soustavu značek v dalším snímku dostat. Tímto je zaručeno, že kamera bude nakalibrovaná celá správně. S tímto kalibrátorem tedy není nutná žádná předchozí zkušenost s kalibrací, program uživatele provede od začátku do konce. Dále je možné vybrat z několika kamerových modelů (např. Caltech, RadialPolynomial, AngularPolynomial atd.). U modelů je nutné vybírat s ohledem na další využití kalibrace, ne všechny jsou například kompatibilní s OpenCV knihovnou.



Obrázek 3.5: Ukázka uživatelského prostředí

Nejdůležitější funkcí tohoto kalibrátoru je ale možnost kalibrace více kamer najednou. V této možnosti již není pomocný ukazatel s polohou značek, nicméně uživatelské prostředí je pořád přehledné. Při této kalibraci se kromě kalibrace kamer také počítá poloha kamer vůči sobě (je nutné mít překrývající se záběry). Na monitor se následně vykreslují pozice kamer v kartézských souřadnicích i s jejich natočením, takže si uživatel může jednoduše zkontrolovat, jestli spočítaná poloha kamer odpovídá jejich reálnému umístění.



Obrázek 3.6: Ukázka uživatelského prostředí multi-kalibrátoru

Kapitola 4

Experimentální ověření funkčnosti algoritmů

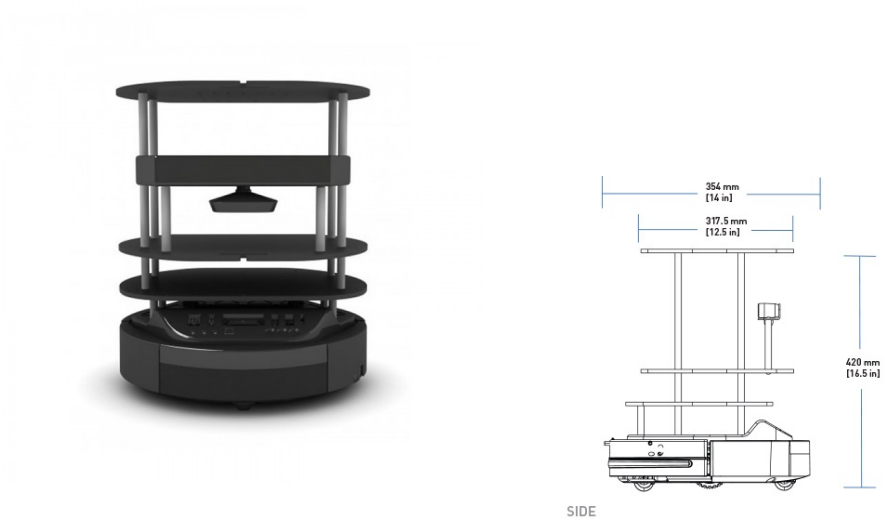
4.1 Setup

Pro potřeby práce byl sestaven robot, který je využit pro všechny experimenty v reálném prostředí. Od původně avizovaného všesměrového podvozku muselo být upuštěno, jelikož tyto podvozky nejsou jednoduše a levně dostupné, proto byl zvolen diferenciální podvozek TurtleBot2, který byl již zakoupený v testovací laboratoři. Podvozek je dále osazen palubním počítačem a čtyřmi kamerami.

4.1.1 TurtleBot2

Cenová dostupnost je jednou z nejdůležitějších pozitivních vlastností tohoto podvozku. Zároveň je podvozek opatřen plnou podporou operačního systému ROS, který umožňuje jednoduchou komunikaci počítače s robotem. Podvozek je připraven na komunikaci pomocí sběrnice USB. Ve výbavě podvozku jsou dále uváděny palubní notebook a stereokamera, které ovšem nebyly v práci využity.

TurtleBot2 je diferenciálním podvozkem. Je opatřen pouze dvěma koly na stranách, vepředu a vzadu jsou pouze neovládaná pomocná kola, aby



(a) : TurtleBot2 v základní výbavě **(b) :** Rozměry podvozku TurtleBot2

Obrázek 4.1: Převzato z : [19] a [20]

byl podvozek stabilní. Díky vlastnostem diferenciálních podvozků je možné TurtleBot2 řídit rychlostmi $v_{\texttt{linear}}$ a $v_{\texttt{angular}}$, je dokonce možné podvozek otáčet na místě. Diferenciální podvozky jsou lépe řiditelné, než podvozky čtyřkolové (ať už smykové nebo podvozky podobné autu), nicméně se zde vyskytují mnohá omezení, která se nevztahují na podvozky všesměrové.

4.1.2 Intel NUC

Jako palubní počítač byl zvolen Intel NUC. Tyto počítače mají obrovskou výhodu ve své velikosti a velké možnosti modifikace. Jelikož je počítač od značky Intel, jedinou pevnou součástí je jejich procesor. Ostatní komponenty si uživatel dokoupí sám dle potřeby. Počítač je opatřen čtyřmi porty USB 3.0, portem HDMI, portem pro Ethernet a síťovou kartou umožňující WiFi připojení.

Kvůli nutnosti připojení více kamer, robotu a periferií k počítači, bylo nutné využít i USB hub, který díky portům USB 3.0 neomezuje funkčnost hardwaru.

Obrázek 4.2: Intel NUC. Převzato z : [21]

4.1.3 Kamera Basler daA1280-54uc

Jako vizuální senzory byly využity kamery Basler daA1280-54uc. Kamery jsou schopny funkce s frekvencí až 60fps. Rozlišení těchto kamer je 1280x960 pixelů. Objektiv na kamery byl zvolen s ohniskovou vzdáleností 4 mm a úhlem záběru asi 60°. Výhodou těchto kamer je opět jejich nízká cena, díky čemuž byly přístupné v laboratoři.

(a) : Kamera Basler daA1280-54uc **(b) :** Objektiv Computar Lens 4mm

Obrázek 4.3: Převzato z : [22] a [23]

Ovladačem ke kamerám je software Pylon, který byl zakomponován do operačního systému ROS. Kvůli nízkému úhlu záběru bylo nutné využít 4 kamery, aby byl pokryt prostor před a po stranách robotu.

4.1.4 Vicon

Pro přesné měření a vyhodnocování experimentů bylo využito systému Vicon. Vicon je systémem používaným na celém světě pro motion-capture, což je využíváno v robotice, počítačové grafice atd. Laboratoř, ve které byla práce testována, je opatřena tímto systémem kamer, což výrazně zjednodušuje přesnost lokalizace a měření pozic robotů.

(a) : Vicon kamera **(b) :** Vicon reflexní kulička

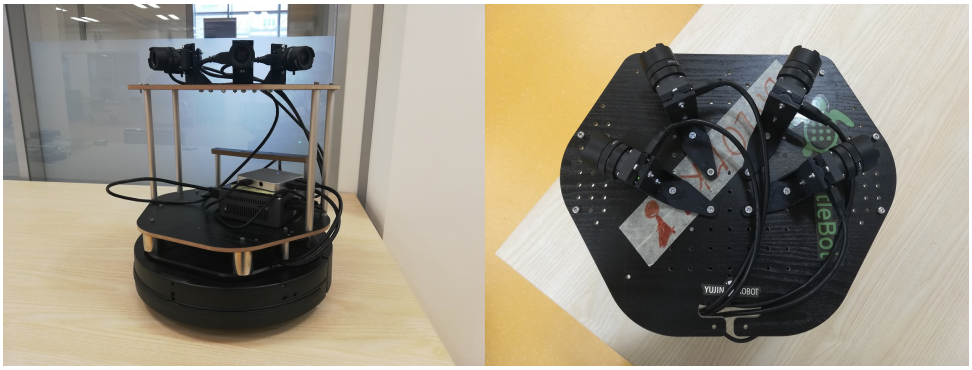
Obrázek 4.4: Převzato z : [24]

Systém funguje na principu několika kamer (myšleno desítky kamer) pokrývající detekovaný prostor. Díky následnému zpracování dat ze všech kamer je možné dosáhnout vysoké přesnosti. Pro fungování systému je nutné využít speciálních reflexních kuliček, které jsou dodávány se systémem. Systém Vicon je schopen tyto kuličky následně detekovat s přesností až na desetiny milimetrů.

4.2 Vytvořený hardware

4.2.1 Sestavený robot

Celkový robot se skládá z podvozku TurtleBot2, čtyř kamer Basler a počítače Intel NUC s připojeným USB hubem.



(a) : Sestavený robot **(b) :** Konfigurace kamer

Kamery jsou připevněny tak, jak je vidět na obrázku 4.5b. Za tímto účelem byly vytisknuty úchyty kamer na 3D tiskárně. Pro upevnění těchto úchytů byly využity předvrtané otvory na deskách k podvozku TurtleBot2, kvůli čemuž konfigurace kamer není ideální (kamery neleží na jedné kružnici se středem ve středu podvozku). Jejich polohu i natočení bylo naštěstí možné získat ze softwaru 3.3. Kamery jsou číslovány zleva, natočení a poloha kamer jsou uváděny vůči pravotočivé souřadnicové soustavě, která má počátek ve středu podvozku a směr osy x je ve směru pohybu podvozku.

Číslo kamery	Natočení α	Pozice X	Pozice Y
0	64°	0.06 cm	11.3 cm
1	26.6°	8.73 cm	6.66 cm
2	−26.6°	8.51 cm	−6.57 cm
3	−64°	−0.06 cm	−11.3 cm

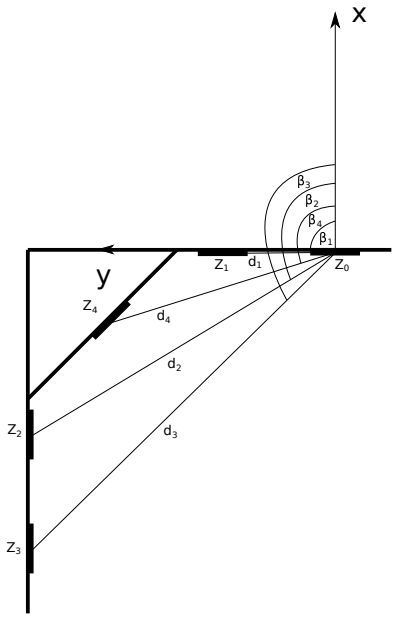
Obrázek 4.6: Konfigurace kamer robotu

Konfigurace kamer byla při experimentech zhodnocena jako ne úplně vhodná, jelikož kvůli jejich nízkému zornému poli robot nemá ideální pokrytí prostoru přímo před sebou. Nejideálnějším řešením by bylo přidání páté kamery, která by směřovala dopředu ve směru osy x.

4.2.2 Dok

Jako dok byly využity pouze dřevotřískové desky přibité k sobě. Dok je konstruován jako velké písmeno L kvůli zvýšené manévrovatelnosti robotu. Kvůli využívání doku i pro jiné diplomové práce s laserovými dálkoměry bylo nutné opatřit dok nerovnostmi pro lepší detekci v datech z laserových dálkoměrů. Zároveň byl dok označen pěti značkami AprilTags pro vizuální

detekci z kamer. Dále byla každá značka opatřena kuličkou pro detekci pomocí systému Vicon, díky čemuž bylo možné změřit vzájemné polohy značek, což je nutné pro zajištění co nejlepší funkčnosti algoritmu v části 2.2.2. Veličiny

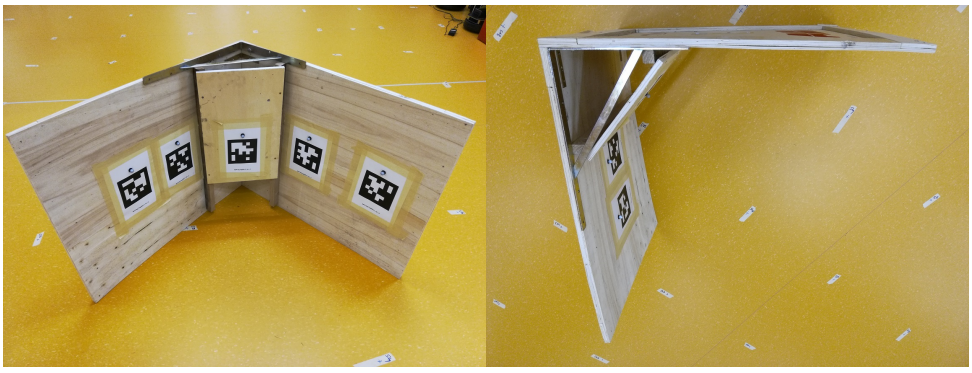


Obrázek 4.7: Popis doku s příslušnými veličinami

popisující dok byly naměřeny v tabulce 4.8, podle obrázku 4.7

ID značky	Vzdálenost [d]	Úhel [β]
0	0 m	0°
1	0.3379 m	90°
2	0.8233 m	120.67°
3	0.9523 m	133.01°
4	0.5913 m	107.69°

Obrázek 4.8: Veličiny popisující dok

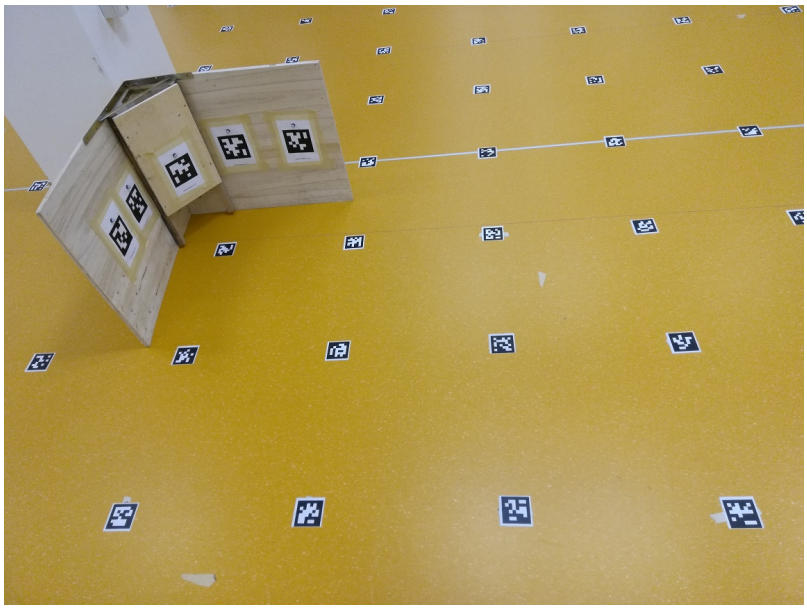


(a) : Dok zepředu **(b) :** Dok seshora

4.3 Reálný experiment

4.3.1 Setup

Reálný experiment byl proveden v laboratoři, ve které byla celá práce vytvořena. Pro měření a vyhodnocování experimentů byl využit systém Vicon 4.1.4. Dok byl postaven na volném místě, kde se robot nemohl setkat s žádnými překážkami. Osvětlení místnosti bylo běžné denní světlo s rozsvícenými zářivkami. Pro hrubou orientaci o poloze robotu (např. z jakého bodu robot pouštět) a doku byly využity značky na zemi, které se v laboratoři vyskytují pro jiné projekty.



Obrázek 4.10: Foto místa experimentu

4.3.2 Popis experimentu

Pro vyhodnocení kvality algoritmů byl využit kvantitativní test. Při vytváření práce bylo premisou, že algoritmus má obstarávat pouze přesné dokování, je tedy předpokladem, že naváděcí algoritmus navede robot do rozumné vzdálenosti (po konzultaci s vedoucím byla tato vzdálenost určena na maximálně 2 m) od doku. Zároveň je předpokládáno, že robot se vždy bude vyskytovat v místech, ze kterých je dok vidět ze správné strany. Díky tvaru doku je tedy experiment omezen pouze na 1 kvadrant vůči poloze doku.

Byly proto vybrány 3 body tak, aby dok byl ze všech plně viditelný. Jeden bod je určen jako střední, tzn. představíme-li si dok jako pravoúhlý, tento bod leží velmi blízko ose tohoto úhlu. Zbylé dva body byly určeny jako bod vpravo a vlevo od osy úhlu doku, zároveň však musela být splněna podmínka viditelnosti všech značek doku (jinak algoritmy selžou). Ze všech 3 bodů bylo provedeno 5 dokování s oběma algoritmy a s proměnlivým počtem detekovaných značek AprilTags. Oba navrhované algoritmy byly otestovány ve stejných podmínkách, jedinou změnou byly počáteční a koncové body:

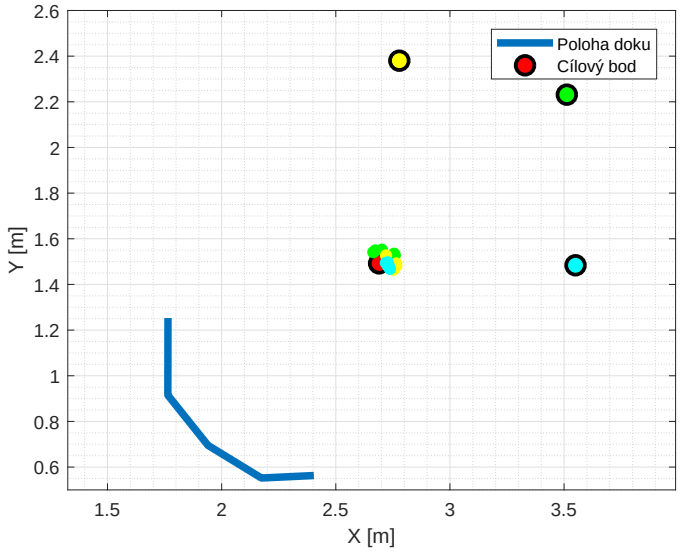
- Algoritmus s lokalizací funguje lépe, je-li koncový bod více vzdálen od doku.
- Algoritmus bez lokalizace funguje lépe v nižší vzdálenosti od doku.

Výsledky experimentu jsou polohy robotu při "zadokování". Pro vyhodnocení experimentů je následně využita vzdálenost od koncového bodu. Jelikož jsou experimenty provedeny kvantitativně, je ze všech těchto vzdáleností vypočítána střední hodnota chyby, minimální a maximální hodnota chyby a její rozptyl. Všechna naměřená data jsou pro názornost vykreslena v grafech.

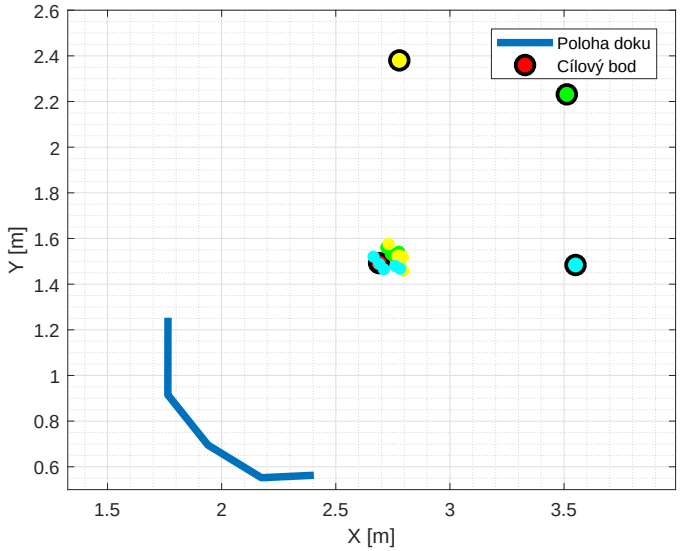
4.3.3 Algoritmus s lokalizací

Při testování lokalizace se projevil problém s použitím gradientní metody pro řešení přeurčené soustavy rovnic. Pokud je každá značka viděna pouze jednou kamerou, žádné 2 přímky se neprotínají v místě jedné značky 2.2, čímž není žádná značka pevně určena. Kvůli této skutečnosti se stává, že algoritmus nalezne jiné dostatečně dobré řešení, což velmi změní lokalizaci doku a také jízdu robotu. Bohužel kamery na robotu nemají tak velký překryv svých zorných úhlů, takže se občas tento problém vyskytuje. Kvůli této skutečnosti byla zvolena dokovací pozice dále od doku (cca 1 m), protože pokud se robot vyskytuje blízko u doku (vzdálenost 0.5 m a nižší), je prakticky nereálné, aby značky byly v zorných úhlech více kamer a tudíž je lokalizace velmi špatná.

Algoritmus byl vyzkoušen při detekci všech pěti značek doku s podmínkou, že je nutné v jeden moment vidět minimálně čtyři z nich. Pro porovnání byl algoritmus vyzkoušen i při detekci pouze čtyř značek (algoritmus ignoruje prostřední značku (ID = 4) 4.8) s podmínkou, že jsou v jeden moment detekovány minimálně tři. Pokud jsou však detekovány pouze 3 značky, ze soustavy rovnic přestává být soustava přeurčená, což značně snižuje přesnost lokalizace.



Obrázek 4.11: Detekce 5 značek AprilTags

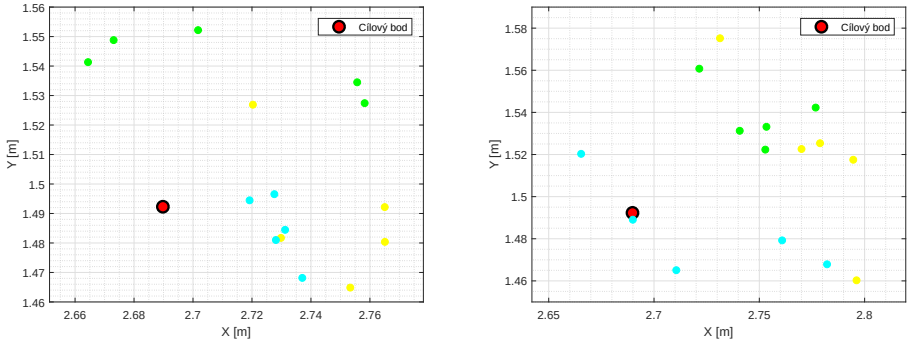


Obrázek 4.12: Detekce 4 značek AprilTags

V grafech 4.11 a 4.12 je červeně označen cílový bod, dále různými barvami s černými okraji jsou označeny startovní body. Body s příslušnými barvami bez černých okrajů jsou dojezdové pozice z příslušných startovních bodů.

- žlutý bod = bod vpravo od osy úhlu
- zelený bod = bod u osy úhlu

- modrý bod = bod vlevo od osy úhlu

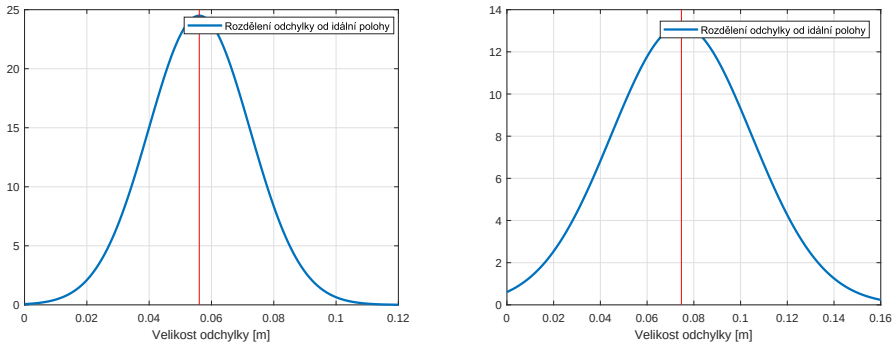


(a) : Detekce 5 značek AprilTags **(b) :** Detekce 4 značek AprilTags

Obrázek 4.13: Detail okolí cílové pozice

Z každé pozice bylo provedeno 5 dokovacích měření. Dok v grafu je pouze z orientačního důvodu, vykreslena je pouze spojnice všech pěti značek doku, reálný dok je větší.

Pro větší názornost je v obrázku 4.14 vykresleno rozdělení chybové vzdálenosti definované střední hodnotou μ a směrodatnou odchylkou hodnot σ. Pokud je střední hodnota nenulová, ale rozptyl velmi malý, není algoritmus nekvalitní, jen je nutné počítat s offsetem algoritmu v podobě střední hodnoty chyby.



(a) : Detekce 5 značek AprilTags **(b) :** Detekce 4 značek AprilTags

Obrázek 4.14: Rozdělení chybové vzdálenosti od koncové pozice

Typ hodnoty	Velikost
μ	0.0561 m
σ	0.0163 m
min	0.0295 m
max	0.0783 m

(a) : Detekce 5 značek AprilTags

Typ hodnoty	Velikost
μ	0.0747 m
σ	0.0301 m
min	0.0032 m
max	0.1111 m

(b) : Detekce 4 značek AprilTags

Obrázek 4.15: Výsledky dokování při využití algoritmu s lokalizací

Z tabulek 4.15 je zřejmé, že počet detekovaných značek má vliv na kvalitu lokalizace robotu. Průměrně robot dokoval s chybou o 2 cm větší, než při dokování s detekcí všech značek. Zároveň při detekci pouze čtyř značek je směrodatná odchylka dvakrát větší, než při detekci všech značek, což naznačuje nižší konzistentnost dokování.

Za povšimnutí dále stojí skutečnost, že dojezdy z "levého"bodu vypadají přesnější, než z ostatních bodů. Kvůli této skutečnosti byla vyhotovena tabulka vzdáleností pro jednotlivé startovní body.

Pravý bod		Střední bod		Levý bod	
Typ hodnoty	Velikost	Typ hodnoty	Velikost	Typ hodnoty	Velikost
μ	0.0616 m	μ	0.0661 m	μ	0.0405 m
σ	0.0166 m	σ	0.0107 m	σ	0.0085 m
min	0.0414 m	min	0.0552 m	min	0.0295 m
max	0.0762 m	max	0.0783 m	max	0.0531 m

Obrázek 4.16: Srovnání kvality dokování v závislosti na startovním bodu při detekci 5 značek

Pravý bod		Střední bod		Levý bod	
Typ hodnoty	Velikost	Typ hodnoty	Velikost	Typ hodnoty	Velikost
μ	0.0985 m	μ	0.0771 m	μ	0.0485 m
σ	0.0106 m	σ	0.0138 m	σ	0.0359 m
min	0.0858 m	min	0.0641 m	min	0.0032 m
max	0.1111 m	max	0.1003 m	max	0.0955 m

Obrázek 4.17: Srovnání kvality dokování v závislosti na startovním bodu při detekci 4 značek

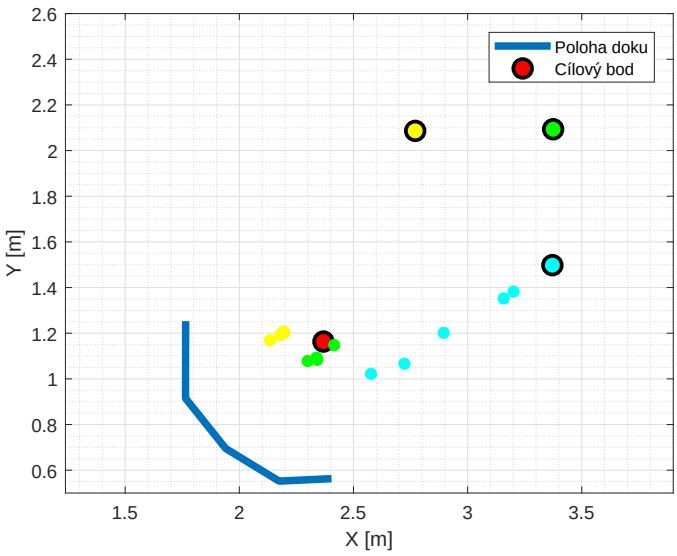
Z tabulek 4.16 a 4.17 je zřejmé, že robot opravdu dokuje lépe z levého bodu než z bodů ostatních. Z toho vyplývá, že metoda bohužel není nezávislá na startovní pozici. Nejvíce je tato skutečnost ovlivněna natočením robotu vůči doku při jízdě do cílové pozice. Jak je uvedeno výše, robot nemá ideální vizuální pokrytí prostoru před sebou. Jede-li zleva, pravá strana doku se vyskytuje před ním a naopak. Jelikož jsou značky na levé straně doku více u sebe než na straně pravé, při jízdě z levého bodu jsou detekovány na straně robotu a díky jejich vzdálenosti od sebe je více pravděpodobné, že alespoň jedna je detekována více kamerami.

Při pokusech nastalo několik případů, kdy robot ztratil značky ze všech zorných polí kamer a díky tomu nezadokoval vůbec. Tato skutečnost je způsobena využitím diferenciálního podvozku místo podvozku všesměrového.

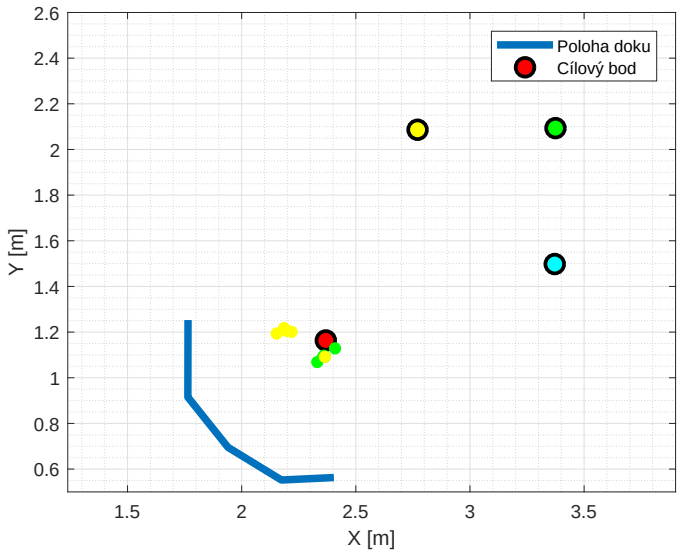
4.3.4 Algoritmus bez lokalizace

Pro testování algoritmu bez lokalizace je nutné, aby značky byly v zorném poli kamer neustále. Kvůli této skutečnosti je nutné pohybovat se v menších vzdálenostech od doku než při předchozích měřeních druhého algoritmu. Jakmile je ztracena nějaká část značek, algoritmus nemá dostatečnou zpětnou vazbu a přestává fungovat. Tento algoritmus je zároveň velmi ovlivněn nevyužitím všesměrového podvozku, jelikož hlavní a jedinou informací z rozložení momentů (2.5) jsou rychlosti v_x a v_y. Jakmile jsou tyto rychlosti přepočítávány na $v_{\texttt{linear}}$ a $v_{\texttt{angular}}$, jsou obdržené informace z algoritmu značně aproximovány a funkčnost algoritmu rapidně klesá.

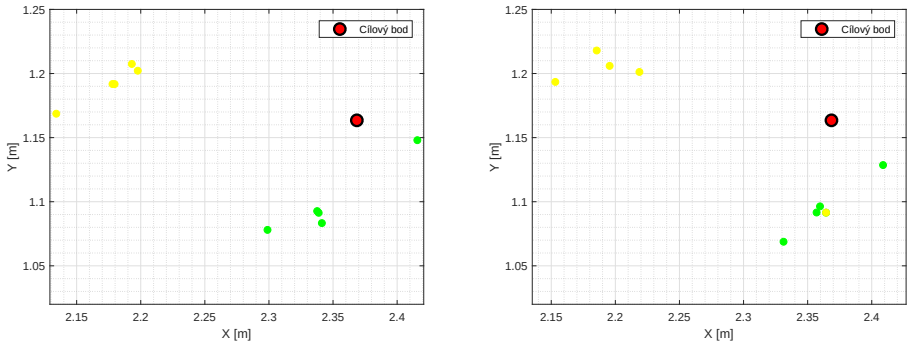
Tak jako předchozí algoritmus byl i tento vyzkoušen při detekci všech pěti značek, tak i při detekci pouze čtyř, kde nedetekovaná značka je opět prostřední (ID = 4).



Obrázek 4.18: Detekce 5 značek AprilTags



Obrázek 4.19: Detekce 4 značek AprilTags

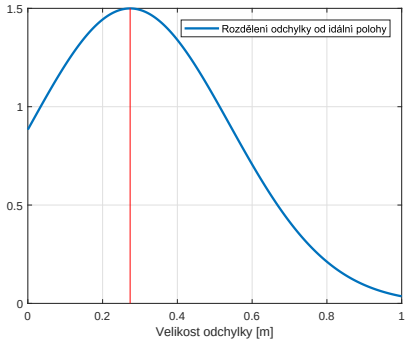


(a) : Detekce 5 značek AprilTags **(b) :** Detekce 4 značek AprilTags

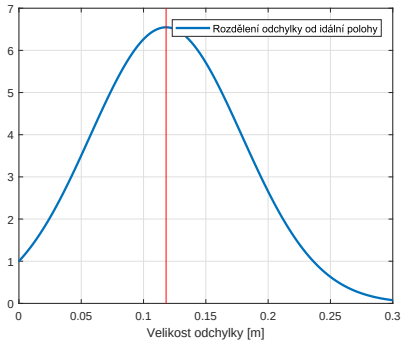
Obrázek 4.20: Detail okolí cílové pozice

V grafech 4.18 a 4.19 je využito stejné značení jako při testování předchozího algoritmu. Jak lze vidět z grafů na první pohled, algoritmus je značně ovlivněn startovní pozicí. Při detekci pouze čtyř značek se robotu nepodařilo dokovat z levého startovního bodu ani jednou ze všech pěti pokusů (pouze se otáčel a popojížděl v jednotkách centimetrů) kvůli nedostatečné informaci o značkách. Při detekci všech značek je evidentní, že z levého startovního bodu byl taktéž velký problém dokovat (2 nejvzdálenější pokusy jsou velmi podobné výsledkům pokusu se čtyřmi značkami). Pro srovnání s předchozím algoritmem byly vykresleny rozdělení vzdáleností od koncové polohy.

Je nutné uvést, že ve statistikách 4.22 algoritmu při detekci pouze čtyř značek nefiguruje nepovedené dokování z levého bodu, které značně ovlivňuje hodnoty vypočítané pro algoritmus při detekci všech značek. Vynecháme-li



(a) : Detekce 5 značek AprilTags



(b) : Detekce 4 značek AprilTags

Obrázek 4.21: Rozdělení chybové vzdálenosti od koncové pozice

Typ hodnoty	Velikost
μ	0.2736 m
σ	0.2660 m
min	0.0497 m
max	0.9262 m

(a) : Detekce 5 značek AprilTags

Typ hodnoty	Velikost
μ	0.1181 m
σ	0.0609 m
min	0.0533 m
max	0.2175 m

(b) : Detekce 4 značek AprilTags

Obrázek 4.22: Výsledky dokování při využití algoritmu bez lokalizace

ze statistik dojezdy z levého startovního bodu, vychází kvalita algoritmu při detekci všech značek v tabulce 4.23.

Typ hodnoty	Velikost
μ	0.1374 m
σ	0.0642 m
min	0.0497 m
max	0.2345 m

Obrázek 4.23: Detekce 5 značek AprilTags (bez levého startovního bodu)

Porovnáme-li tabulky 4.23 a 4.22b, je zřejmé, že počet detekovaných značek nemá na kvalitu algoritmu prakticky žádný význam. Z naměřených výsledků dokonce vychází, že při detekci pouze čtyř značek je algoritmus lepší, nicméně tato skutečnost je ovlivněna malým počtem měření (vynecháme-li levý startovní bod, statistika je počítána pouze z deseti hodnot). Se zvýšeným počtem experimentů by výsledky algoritmů vycházely velice pravděpodobně stejně.

Pro možnost porovnání s předchozím algoritmem jsou uvedeny v tabulkách 4.24 a 4.25 hodnoty pro jednotlivé startovní body.

Z tabulek 4.24 a 4.25 je vidět, že závislost na startovní pozici je velmi vysoká. Zároveň je z obou tabulek patrné, že nezáleží na počtu detekovaných

Pravý bod		Střední bod		Levý bod	
Typ hodnoty	Velikost	Typ hodnoty	Velikost	Typ hodnoty	Velikost
μ	0.1949 m	μ	0.0800 m	μ	0.5638 m
σ	0.0233 m	σ	0.0216 m	σ	0.2678 m
min	0.1752 m	min	0.0497 m	min	0.2514 m
max	0.2345 m	max	0.1103 m	max	0.8608 m

Obrázek 4.24: Srovnání kvality dokování v závislosti na startovním bodu při detekci 5 značek

Pravý bod		Střední bod		Levý bod	
Typ hodnoty	Velikost	Typ hodnoty	Velikost	Typ hodnoty	Velikost
μ	0.1627 m	μ	0.0735 m	μ	NaN
σ	0.0555 m	σ	0.0177 m	σ	NaN
min	0.0721 m	min	0.0533 m	min	NaN
max	0.2175 m	max	0.1018 m	max	NaN

Obrázek 4.25: Srovnání kvality dokování v závislosti na startovním bodu při detekci 4 značek

značek, pouze na startovní pozici. Výsledky algoritmu jsou nejlepší, pokud se startovní pozice vyskytuje blízko osy úhlu doku, jelikož jsou značky rozmístěny rovnoměrně na obou stranách robotu.

Tento algoritmus je také závislý na kvalitě řídicího regulátoru. Algoritmus v této práci je řízen pouhým PI regulátorem. Vlastnosti algoritmu by mohly být do jisté míry zlepšeny s použitím lepších regulátorů, jehož optimálním návrhem by mohla být pokryta celá diplomová práce.

4.3.5 Zhodnocení

Oba dva algoritmy byly otestovány za stejných podmínek ve stejný den, při experimentování byly stejné světelné podmínky a robot byl po celou dobu experimentů zapojen v nabíječce, aby byl potlačen vliv klesajícího napětí napájecí baterie.

Z experimentů vychází lépe algoritmus s používáním lokalizace při detekci všech značek. Průměrná odchylka od cílové pozice je lehce pod 6 cm. U algoritmu mírně záleželo na startovní pozici, odchylky mezi startovními pozicemi se ale pohybovaly v jednotkách centimetrů (cca 2 cm).

Při využití algoritmu bez lokalizace se hodnoty odchylek od cílové pozice pohybovaly až v desítkách centimetrů. Velký vliv u tohoto algoritmu má

počáteční pozice vůči doku. Pokud se tato poloha vyskytuje u osy úhlu doku, algoritmus dosahuje kvalit předchozího algoritmu. Za použití lepšího regulátoru by možná mohl být i přesnější. Markantní vliv počáteční pozice však tento algoritmus činí naprosto nepoužitelným v kombinaci s hardwarem, na kterém byly experimenty provedeny.

Závěr

Cílem práce bylo seznámit se z lokalizací robotu pomocí kamer a navigací robotu pomocí kamer. Následně měl být vytvořen dokovací algoritmus, který bude co nejpřesněji dokovat všesměrový robot na předem určené místo v doku. Podle potřeby měl být vytvořen dok, který měl být využit pro vyzkoušení algoritmu v reálném prostředí.

Při návrhu algoritmu bylo využito nejen nově nabytých vědomostí, ale i vědomostí z různých školních předmětů. Při prvních testech s reálným hardwarem byly zjištěny potíže, a proto byl navržen i další algoritmus využívající naprosto odlišný přístup. Oba algoritmy byly implementovány a vyzkoušeny v reálném prostředí.

V zadání práce je uvedeno slovo "přesné dokování", jehož význam se může v různých oblastech lišit. Po konzultaci s vedoucím práce jsme význam slova pro tuto práci stanovili na jednotky centimetrů, nejlépe přesnost v rámci milimetrů. Tohoto výsledku bohužel nebylo dosaženo, nejlepší algoritmus se pohybuje v odchylkách kolem 6 cm. Odchylky druhého algoritmu byly značně vyšší. Oba algoritmy mají svá omezení, algoritmus s lokalizací přestává fungovat při nízké vzdálenosti od doku, algoritmus bez lokalizace funguje hlavně v poloze robotu u osy úhlu doku.

Přestože se může jevit, že algoritmy nejsou dostatečně přesné, byly vyvíjeny pro použití na všesměrovém podvozku. Tento podvozek bohužel nebyl obstarán pro reálný experiment, proto musely být algoritmy upraveny pro fungování s podvozkem diferenciálním. Zároveň by značně zlepšilo funkčnost algoritmu přidání více kamer, aby bylo dosaženo lepšího pokrytí prostoru jejich zornými poli. Pokud by se na robotu vyskytovalo více kamer s většími překryvy zorných polí, zvýšila by se kvalita lokalizace v prvním algoritmu a

zvýšil by se počet momentů, ze kterých je počítán pohyb v algoritmu druhém. Při použití všesměrového podvozku by byl zrušen vliv otáčení robotu na kvalitu algoritmu (hlavně algoritmus bez lokalizace).

Závěrem lze říci, že v budoucnu by se měly algoritmy otestovat s všesměrovým podvozkem. Zároveň by bylo dobré zlepšit vizuální pokrytí prostoru (ať už přidáním kamer, nebo zvolením jiné konfigurace kamer). Dále by bylo dobré zlepšit regulátor v algoritmu bez lokalizace, aby byl vliv řízení na konečnou chybu dokování skoro nulový.

Příloha A

Literatura

[1] M. Betke and L. Gurvits, "Mobile robot localization using landmarks," IEEE Transactions on Robotics and Automation, vol. 13, no. 2, pp. 251–263, Apr. 1997.

[2] H.-Y. Lin, J.-H. Lin, and M.-L. Wang, "A visual positioning system for vehicle navigation," Proceedings. 2005 IEEE Intelligent Transportation Systems, 2005., Sep. 2005.

[3] J. Esteves, A. Carvalho, and C. Couto, "Generalized geometric triangulation algorithm for mobile robot absolute self-localization," 2003 IEEE International Symposium on Industrial Electronics (Cat. No.03TH8692).

[4] N. Tomatis, I. Nourbakhsh, K. Arras, and R. Siegwart, "A hybrid approach for robust and precise mobile robot navigation with compact environment modeling," Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164), 2001.

[5] D. Szaloki, N. Koszo, K. Csorba, and G. Tevesz, "Marker localization with a multi-camera system," 2013 International Conference on System Science and Engineering (ICSSE), 2013.

[6] K. E. Bekris, A. A. Argyros, and L. E. Kavraki, "Exploiting Panoramic Vision for Bearing-Only Robot Homing," Imaging Beyond the Pinhole Camera, pp. 229–251, 2006.

[7] V. Ayala, J. Hayet, F. Lerasle, and M. Devy, "Visual localization of a mobile robot in indoor environments using planar landmarks," Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000) (Cat. No.00CH37113).

[8] D. Yuen and B. Macdonald, "Vision-based localization algorithm based on landmark matching, triangulation, reconstruction, and comparison," IEEE Transactions on Robotics, vol. 21, no. 2, pp. 217–226, 2005.

[9] Reinštein, M. (2018). From LSQ to NLSQ. https://cw.fel.cvut.cz/b172/_media/courses/bxaro/iro_lsq-nlsq.pdf

[10] Olson, Edwin. "AprilTag: A robust and flexible visual fiducial system."2011 IEEE International Conference on Robotics and Automation. IEEE, 2011.

[11] Begum, Afroza, et al. "A Simple Visual Servoing and Navigation Algorithm for an Omnidirectional Robot." 2010 3rd International Conference on Human-Centric Computing, 2010, doi:10.1109/humancom.2010.5563325.

[12] Gupta, Misha, et al. "Bearing Only Visual Homing: Observer Based Approach." 2017 25th Mediterranean Conference on Control and Automation (MED), 2017, doi:10.1109/med.2017.7984144.

[13] M. Liu, C. Pradalier, Q. Chen, and R. Siegwart, "A bearing-only 2D/3D-homing method under a visual servoing framework," in Robotics and Automation (ICRA), 2010 IEEE International Conference on, pp. 4062–4067, IEEE, 2010.

[14] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, "ROS: an open-source Robot Operating System," ICRA Workshop on Open Source Software, 2009.

[15] Camera suite. [online] April robotics laboratory. [cit. 20.5.2019] Dostupné z: https://april.eecs.umich.edu/wiki/Camera_suite

[16] Krajník, T., Nitsche, M., Faigl, J., Duckett, T., Mejail, M., & Přeučil, L. (2013, November). External localization system for mobile robotics. In 2013 16th International Conference on Advanced Robotics (ICAR) (pp. 1-6). IEEE.

[17] Beránek Matěj. Experimentální prostředí pro skupinovou robotiku. Praha, 2017. Bakalářská práce. ČVUT FEL. Vedoucí práce Ing. Libor Přeučil CSc.

[18] AprilTag. [online] April robotics laboratory. [cit. 20.5.2019] Dostupné z: https://april.eecs.umich.edu/software/apriltag.html

[19] Turtlebot 2 - mobile robotic platform. [online] Clearpath robotics. [cit. 20.5.2019] Dostupné z: https://www.clearpathrobotics.com/turtlebot-2-open-source-robot/

[20] Turtlebot 2. [online] Ros components. [cit. 20.5.2019] Dostupné z: https://www.roscomponents.com/en/mobile-robots/9-turtlebot-2.html#/3d_sensor-no/controller-no/docking_station-no/power_cods-europa_cee_7_16/additional_battery-no/disk_extra_nuc-no/assembly_and_configuration-no/arm_robot-no

[21] Intel NUC. [online] [cit. 20.5.2019] Dostupné z: https://www.intel.com/content/www/us/en/products/boards-kits/nuc.html

[22] Kamera Basler dart DAA1280-54UM-S/CS/B. [online] SodaVision. [cit. 20.5.2019] Dostupné z: https://www.sodavision.com/product/daa1280-54um-scsb/

[23] Computar 4MM F1.2 MANUAL IRIS CS-MOUNT LENS [online] RMA Electronics. [cit. 20.5.2019] Dostupné z: https://www.rmaelectronics.com/computar-t0412fics-1-3-4mm-f1-2-manual-iris-cs-mount-lens/

[24] VICON [online] VICON [cit. 20.5.2019] Dostupné z: https://www.vicon.com

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Beránek** Jméno: **Matěj** Osobní číslo: **434817**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra řídicí techniky**

Studijní program: **Kybernetika a robotika**

Studijní obor: **Kybernetika a robotika**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Přesné dokování mobilního robotu s využitím kamer

Název diplomové práce anglicky:

Precise docking of mobile robot with cameras

Pokyny pro vypracování:

1. Seznamte se s metodami přesné lokalizace z vizuálních dat
2. Seznamte se s metodami navigace mobilního robotu (s důrazem na všesměrový podvozek)
3. Navrhněte a implementujte metodu pro přesné dokování mobilního robotu
4. Vyhodnoťte vlastnosti navržené metody

Seznam doporučené literatury:

1.Visual localization of a mobile robot in indoor environments using planar landmarks. Ayala, V., Hayet, J. B., Lerasle, F., & Devy, M. International Conference on In Intelligent Robots and Systems, (IROS) 2000
2. A hybrid approach for robust and precise mobile robot navigation with compact environment modeling. Tomatis, N., Nourbakhsh, I., Arras, K., & Siegwart, R. Conference on Robotics and Automation, ICRA. 2001
3. A visual positioning system for vehicle or mobile robot navigation, LIN, Huei-Yung; LIN, Jen-Hung Transactions on Information and Systems, 2006

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Karel Košnar, Ph.D., inteligentní a mobilní robotika CIIRC

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

Datum zadání diplomové práce: **14.02.2019** Termín odevzdání diplomové práce: **24.05.2019**

Platnost zadání diplomové práce:
do konce letního semestru 2019/2020

Ing. Karel Košnar, Ph.D.	prof. Ing. Michael Šebek, DrSc.	prof. Ing. Pavel Ripka, CSc.
podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

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

. | Datum převzetí zadání

Podpis studenta