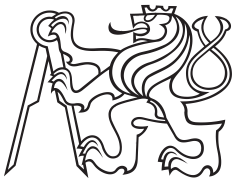
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra kybernetiky

Celoživotní určování polohy mobilního robotu

David Nováček

Vedoucí: Ing. Vladimír Smutný, Ph.D.
Obor: Robotika
Studijní program: Kybernetika a robotika
Květen 2017

Poděkování

V prvé řadě bych rád poděkoval svému vedoucímu Vladimíru Smutnému za trpělivost, cenné připomínky a pozitivní cestu vedení. Dále pak Liboru Wagnerovi především za rady týkající se prostředí ROS. Jarmile Jiraské za uskutečněné měření v provozní hale Phoenix ve Vysokém Mýtě. Děkuji také rodině a všem, kteří mě při studiu podporovali.

Prohlášení

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

V Praze, 26. května 2017

Abstrakt

Tato bakalářská práce se věnuje celoživotnímu určování polohy mobilního robotu, který bude nasazen v industriálním prostředí k přepravování materiálu či zboží. Mobilní robotika pomocí lokalizace zajišťuje robotu potřebnou míru autonomie k samostatné činnosti. Metoda založena na transformaci normálního rozdělení je vhodná do dynamického prostředí s určením polohy v reálném čase a limitem využité paměti.

Cílem práce je uvedení experimentální sestavy do provozu a ověření funkčnosti metody ve vnitřních dynamických prostředí. Důraz je kladen především na sběr dat z reálných prostředí a testování algoritmu lokalizace nad těmito daty. Experimenty byly provedeny na čtyřkolovém pozemním robotu Jackal vybaveném lidarem v prostředí ROS.

Práce zahrnuje popis použitých senzorů, princip metody, popis hardwaru a softwaru a zhodnocuje výsledky experimentů, které potvrzují schopnost dlouhodobé lokalizace pomocí transformace normálního rozdělení.

Klíčová slova: dlouhodobá lokalizace, mapování, robot, lidar, SLAM, NDT, ROS

Vedoucí: Ing. Vladimír Smutný, Ph.D.

Abstract

This bachelor thesis is dedicated to a lifelong localization of a mobile robot, which is designed to be used in an industrial environment for transport of material or goods. Mobile robotics provided by localization ensures sufficient amount of autonomy for the robot to be able to function without further assistance. The method based on transformation of normal distribution is suitable for usage in dynamic environment with real-time localization in appropriate limits of used memory.

The goal of this thesis is to bring the experimental assembly into everyday operation and to verify functionality of the method in inner dynamical area. The emphasis is put on data collecting from real world and localization algorithm testing based on the data. The experiments were performed on four-wheel ground robot Jackal equipped with lidar in ROS environment.

This thesis includes a description of used sensors, the principle of method, hardware and software description and it evaluates the results of experiments which confirm the ability of long time localization using transformation of normal distribution.

Keywords: lifelong localization, mapping, robot, lidar, SLAM, NDT,ROS

Title translation: Lifelong localization of mobile robot

Obsah

Obrázky

Tabulky

Kapitola 1

Úvod

Mobilní robotika se mimo jiné zabývá vývojem autonomních systémů, které jsou schopny vnímat svět kolem sebe a přizpůsobit svoje chování na základě podnětů vnějšího prostředí. Jednou ze základních problematik, kterou mobilní robotika řeší, je určení polohy robotu. Bez informace, kde se v danou dobu robot nachází, nemůže efektivně plnit své úkoly. Schopnost určit svoji polohu jde ruku v ruce se schopností rozpoznávat své okolí a udržovat vnitřní model prostředí, ve kterém se nachází. Aby byl robot v patřičné míře autonomní, musí si uvědomovat svoji polohu i v prostředí, které se s časem mění a tuto informaci poskytovat v reálném čase.

Tato práce si klade za cíl vyhledat a zprovoznit, popřípadě implementovat algoritmus řešící problém Simultánní lokalizace a mapování metodou využívající Transformaci normálního rozdělení a otestovat funkčnost tohoto algoritmu na robotu vybaveného laserovým dálkoměrem v reálném prostředí.

Struktura práce je následující. Druhá kapitola popisuje problematiku lokalizace a mapování, ve třetí kapitole jsou popsány použité senzory, kapitola čtvrtá pojednává o principu použité metody a popisuje implementaci algoritmu, v páté a šesté kapitole je popsán použitý hardware a potřebný software k jeho použití, sedmá kapitola shrnuje výsledky prováděných experimentů. Na konci práce je zařazen návrh zlepšení a závěr.

Kapitola 2

Definice problému

K vykonávání smysluplné činnosti se robot musí bezpečně pohybovat a vnímat prostředí kolem sebe. Inteligentní mobilní robot musí být schopen udržovat vnitřní představu o tomto prostředí, která zahrnuje mimo jiné i znalost polohy, ve které se v danou chvíli nachází. Míra vnímání okolního prostředí je závislá na kvalitě dat ze senzorů poskytující informace o okolním světě a stavu robota.

Vytvoření kvalitního modelu prostředí je závislé na znalosti polohy senzorů v prostředí. Naopak k dostatečně přesnému určení polohy robota je zapotřebí znalost modelu prostředí. Vzájemností určování polohy a vytváření modelu prostředí se tyto dva problémy často řeší jako jeden komplexní problém obecně známý jako **simultánní lokalizace a mapování** (SLAM).

2.1 Lokalizace

Určováním polohy robota se myslí odhad pozice na základě analýzy naměřených dat z různých senzorů. Jedním z nástrojů pro sledování polohy robota je odometrie, která určuje polohu robota na základě ujeté vzdálenosti od referenční polohy. Dalším zdrojem informací je inerciální měřící jednotka skládající se z elektronického gyroskopu, akcelerometru a magnetometru. Tyto tři senzory poskytují informaci o natočení robota v prostoru a jeho zrychlení. Senzorem, který na rozdíl od předchozích neměří vnitřní stav robota, je laserový dálkoměr. Tento senzor měří v jedné rovině vzdálenost od okolních objektů. Fúzí dat ze senzorů lze získat potřebné informace k určení

polohy robotu.

Problém lokalizace můžeme rozdělit do kategorií podle situace, ve které se robot nachází [3]. První situací je sledování známé polohy robotu známá jako **kontinuální (lokální) lokalizace**, která koriguje kumulativní chybu odometrie na základě porovnání senzorických dat s modelem prostředí. Kontinuální lokalizace nehledá polohu v celém statovém prostoru poloh, ale pouze v nejbližším okolí od poslední známé pozice. Díky tomu, že algoritmus prohledává jen část prostoru, je tato lokalizace méně časově náročná. Integrací nepřesných dat ze senzorů je metoda zatížena velkou chybou a často diverguje. Pozice robotu nemusí být nutně vztažena k mapě, často se lokalizuje pomocí relativních souřadnic vztažených k referenční pozici.

V reálném prostředí však často nemáme apriorní informaci o poloze robotu. Tomu odpovídají situace, kdy selže senzorický systém, nebo je robot spuštěn a nemá informaci o svoji poloze. Lokalizace tohoto typu se nazývá **globální (absolutní) lokalizace** [3]. V principu jde o problém, kdy se robot snaží najít svoji pozici hledáním shody mezi naměřenými a referenčními daty modelující svět. Jde o složitější problém, než je kontinuální lokalizace, a to svoji časovou i paměťovou náročností. Při procházení stavového prostoru možných pozic se eleminují ty, kterým naměřená data neodpovídají dokud nezbyde jedna správná pozice.

Počet parametrů určující jednoznačně polohu robotu je stejný jako počet stupňů volnosti volného tělesa v daném prostoru. Polohu pozemního kolového robotu popisují kartézské souřadnice (x,y) a úhel natočení (ϕ). Pomocí těchto tří parametrů jsme schopni popsat libovolnou polohu robotu v jakémkoliv okamžiku.

2.2 Mapování

Druhou stranou mince úspěšné lokalizace je existence modelu prostředí. K plánování bezkolizního pohybu a řešení běžných úkolů je nutné udržovat informace o překážkách a průjezdnosti prostředí. Proces mapování se musí vypořádat s fúzí naměřených dat z různých senzorů, udržováním aktuálního modelu prostředí, nepřesností a zašuměním dat. Důležité je také udržování rozumného objemu dat. Se zvěčujícím se modelem světa zákonitě narůstá objem dat nezbytný k jeho popisu. Není však nutné popisovat prostředí, o kterém již má robot dostatek informací.

K problému mapování lze přistupovat z několika hledisek. Prvním hlediskem je reprezentace světa. Způsobů uchování modelu prostředí je několik. Velmi známým a často používaným modelem je mřížka obsazenosti [1], kde každá buňka této mřížky reprezentuje čtvercovou oblast světa. V buňce je uložena pravděpodobnost, s jakou je buňka obsazena. Další možností jsou geometrické mapy [4], které aproximují překážky v reálném světě sadou geometrických primitiv, jako jsou úsečky, kružnice, oblouky a mnohoúhelníky. Tento typ je paměťově méně náročný a je vhodný k plánování trajektorií pohybu robotu. Existují také modely s vyšší mírou abstrakce, mezi které patří topologické a symbolické mapy [3].

Dalším hlediskem je výpočetní a datová náročnost algoritmu. Od robotu se očekává, že jeho poloha bude k dispozici v reálném čase s dostatečnou přesností. Nelze se však vyhnout kompromisu mezi přesností modelu a časovou náročností výpočtu.

Algoritmus řešící mapování by měl být invariantní vůči různým typům prostředí. Přestože se často volba řešení odvíjí od prostředí, ve kterém se robot bude nacházet, měl by být alespoň se sníženou mírou funkčnosti schopen mapovat prostředí, do kterých nebyl původně určen.

2.3 Omezení řešené úlohy

Vzhledem k rozsáhlosti problematiky Simultánní lokalizace a mapování je potřeba specifikovat, za jakých podmínek bude tato problematika řešena. Mapování bude probíhat ve 2D. Mapa bude odpovídat horizontálnímu řezu budovy v určité výšce. Proto by prostředí nemělo obsahovat více podlaží, schody, vysoké prahy, rampy a nakloněné plochy.

Velikost prostředí musí být adekvátní velikosti robotu a rozměry překážek uzpůsobené rozlišitelné schopnosti senzorů. Předměty musí zajistit dostatečnou odrazivost laserového paprsku z dálkoměru, tudíž se vylučuje přítomnost zrcadel, lesklých předmětů nebo černých těles pohlcující záření.

Kapitola 3

Senzory mobilní robotiky

V této kapitole budou představeny senzory, které mobilní robotika využívá. Budou zde popsány principy, na kterých senzory fungují, jejich přednosti a nevýhody.

3.1 Odometrie

Základním a dostupným způsobem měření polohy robotu je odometrie, která určuje změnu pozice robotu mezi dvěma časovými okamžiky. V principu jde o měření počtu impulzů inkrementálními senzory polohy, které jsou umístěny v kolech. Impulzy jsou na základě znalosti kinematiky robotu (průměr kola, rozvor náprav) přepočítávány na polohu a orientaci. Absolutní polohu lze získat integrací změn pozic, kterou popisuje následující rovnice

$$\boldsymbol{x}_1 = \boldsymbol{x}_0 + \int_{t_0}^{t_1} \frac{d\boldsymbol{x}}{dt} dt, \tag{3.1}$$

kde $\boldsymbol{x}_0$ a $\boldsymbol{x}_1$ jsou pozice robotu v časech t_0, t_1 a $\frac{d\boldsymbol{x}}{dt}$ je rychlost robotu [5].

V důsledku diskrétního zpracování dat můžeme polohu měřit pouze v určitých časových okamžicích. Pozice je zde reprezentována v diskretizované podobě, přestože se robot pohybuje spojitě. Jelikož senzory integrují spolu se změnou pozice i její nepřesnosti, vzniká kumulativní chyba závislá na délce

ujeté dráhy. Z tohoto důvodu odometrie poskytuje dostatečně dobrou a věrohodnou informaci o poloze pouze na krátkou vzdálenost.

Zdroje chyb lze rozdělit do dvou kategorií [6]. Deterministické chyby vznikají nepřesným modelem kinematiky robotu. Typické deterministické chyby jsou rozdílné průměry kol, nepřesná znalost rozvoru nápravy a průměru kol, konečné rozlišení senzorů. Druhým typem jsou chyby náhodné (nedeterministické), které vznikají při nestandardních situacích, jako je jízda po kluzkém povrchu, smyk, deformace kol nebo jednostranné zatížení robotu. Náhodné chyby se špatně detekují a v krátkém čase mohou způsobit velkou odchylku od skutečné hodnoty.

I přes svoje zápory je odometrie důležitým a hlavně levným nástrojem určování pozice robotu. V praxi se spolu s odometrií používají další senzory. Například kompasem a inerciální měřící jednotkou lze měření zpřesnit.

3.2 Inerciální měřící jednotka

Přestože inerciální měřící jednotka (IMU) nebyla v praktické části této práce použita, bude ve stručnosti zmíněna především kvůli možnosti nasazení v budoucí práci. IMU je složena z akcelerometrů a gyroskopů, které jsou umístěny do tří navzájem kolmých os. IMU v této konfiguraci dokáže měřit lineární a úhlové zrychlení a úhel natočení ve všech směrech. Akcelerometry většinou pracují na kyvadlovém nebo vibračním principu [7]. Oba principy měří silové účinky setrvačné síly, které jsou převedeny na zrychlení tělesa, na které je akcelerometr připevněn. Gyroskopy užívané v mobilní robotice pracují nejčastěji na vibračním nebo také optickém principu. Známé gyroskopy s rotující hmotou se vzhledem k jejich velikosti nepoužívají.

Akcelerometry a gyroskopy tvořící IMU mají velmi rychlou odezvu měření, na druhou stranu jsou zatíženy významnými teplotními a časovými drifty. Jednou z příčin driftu je situace, kdy se vychýlené tělísko úplně nevrátí zpět do rovnovážné polohy. Funkční lokalizační metody lze dosáhnout využitím kombinací senzorů jako je odometrie, IMU a kompas, ale především správnou metodou zpracování dat [8].

3.3 Laserový dálkoměr

Dalším hojně užívaným senzorem nejen mobilní robotiky je laserový dálkoměr (lidar). Na rozdíl od předchozích dvou senzorů lidar neměří vnitřní stav robotu. Jedná se o aktivní absolutní senzor pracující na principu měření doby letu paprsku ("Time of flight").

Lidar emituje energii ve formě elektromagnetického záření, konkrétně světla. Laserový zdroj světla emituje paprsek, který se odrazí od překážky, a je zachcen fotocitlivou deskou [3]. Na základě doby letu paprsku lze spočítat vzdálenost mezi senzorem a překážkou.

Narozdíl od sonaru, který pracuje také na principu doby šíření vlnění, není lidar závislý na teplotě vzduchu [5]. Použitý laserový paprsek spadá do 1. třídy nebezpečnosti laserů, je tudíž neškodný lidskému zraku. Další předností je nezávislost vůči okolním světelným podmínkám, je schopen pracovat na přímém světle i v naprosté tmě. Lze ho použít v aplikacích ve vnitřních i vnějších prostorách.

Pro měření v robotice se často používá verze, kdy je laserový paprsek skrze rotující zrcadlo rozmítán do jedné, z pravidla horizontální, roviny. Tímto způsobem proběhne série měření s krokem okolo jednoho stupně. Některé laserové dálkoměry jsou schopny pomocí rozmítaného paprsku pokrýt celý horizont. Pro zachycení tvaru budov existují verze s rozmítaným paprskem ve dvou rovinách. Vytvoření jednoho scanu tohoto typu dálkoměru však trvá dlouho a pro účely lokalizace se nehodí [5]. V případě, že je potřeba mapovat ve třech dimenzích, lze využít dva laserové dálkoměry s rozmítaným paprskem v jedné rovině v konfiguraci, kdy jsou na sebe roviny kolmé.

3.3.1 Bezpečnostní funkce

V mobilní robotice, narozdíl od ostatních odvětví robotiky, dochází ke sdílení pracovního prostoru lidí a robotů. Vzhledem k budoucímu použití robotů v industriálním prostředí, bude zapotřebí dodržet oficiální bezpečnostní evropské standardy. V české verzy jde o normu ČSN EN 1525 (268850) *Bezpečnost motorových vozíků - Vozíky bez řidiče a jejich systémy* [9].

Mobilní robot musí být schopen detekovat osobu v jakékoliv poloze, která

koliduje s naplánovanou trajektorií pohybu robotu. Tuto detekci lze vyřešit umístěním laserového dálkoměru do optimální výšky. Důležité je splnit následující podmínky: [10]

- Vozík musí detekovat osoby nejméně v celé své šířce ve všech směrech pohybu.
- Vozík musí detekovat válec o průměru 200 mm a délce 600 mm ležící kolmo k trajektorii.
- Vozík musí detekovat válec o průměru 70 mm a výšce 400 mm stojící na zemi.

Pro budoucí použití je naplánované umístění lidaru do výšky 150 mm nad zemí.

3.4 Digitální kamera

Bohatým zdrojem informací je kamera, která je v mobilní robotice velmi často nasazována. Kamera umožňuje snímat okolní prostředí podobně jako lidské oko. Zpracování obrazů z kamery v užitečné informace je však komplikovaný proces přesahující rámec této práce, která se bude zabývat pouze uvedením kamery do provozu a sbíraním dat pro budoucí analýzu.

Kapitola 4

Analýza problému a koncept řešení

Problém simultánní lokalizace a mapování, je jednou ze základních výzev mobilní robotiky a řešení nabralo v posledních letech mnoho směrů. Na základě stati (Kapošváry [10]) shrnující metody řešící SLAM se zdá být silným adeptem metoda využívající transformaci normálního rozdělení (NDT). Metodu představili autoři Einhorn & Gross v práci [1]. V porovnání s ostatními metodami je Grafový SLAM využívající hybridní NDT mapy mladou metodou, která ve světě není doposud zažita. Tato metoda byla použita jako řešení SLAM problému a v této kapitole bude blíže popsána.

Nejprve je však potřeba popsat prostředí a specifikovat požadavky, které vyplývají z konceptu budoucího užití. Pro lepší představu nastíním konkrétní situaci, kterou bylo inspirováno zadání této práce. Zákazník z průmyslu má v plánu nasadit ve své továrně autonomní roboty, které budou převážet náklad (materiál, palety se zbožím a podobně). Přirozeným chodem továrny se prostředí mění, kupříkladu přesouváním palet či pohybem ostatních robotů nebo lidí. Z popsané situace plyne hned několik požadavků. Robot musí být schopen zapracovat změny prostředí do svého vnitřního modelu. Jinými slovy, musí fungovat v dynamickém prostředí. Vzhledem k tomu, že robot bude v provozu každý den po dobu několika let, a nejde tedy o jednorázovou expedici zkoumající prostředí, vyvstává nemalá výzva řešení dlouhodobé lokalizace, která musí udržovat model dynamického prostředí za podmínky maximálního objemu dat nezbytného k jeho popisu.

4.1 Dynamika prostředí

Prostředí, ve kterém se robot nachází, nezůstává neměnné a s časem se mění. Avšak ne všechny objekty v dynamickém prostředí musí nutně měnit svoji pozici. K lepšímu pochopení lze zavést následující kategorie [10].

- **Statické objekty** - tato skupina v čase nemění svoji pozici. Do této skupiny patří zdi, velké regály a stroje v továrnách.
- **Polostatické objekty** - tyto objekty se mohou přesouvat, svoji polohu však nemění tak často, jako dynamické objekty. Do této skupiny lze zahrnout nábytek, palety a krabice.
- **Dynamické objekty** - mění svoji polohu velmi často. Příkladem jsou lidé, vysokozdvižné vozíky a ostatní roboty.

Zavedení skupiny polostatických objektů lépe popisuje dynamické prostředí, ve kterém je občas těžké určit, zda jde o statický, nebo dynamický objekt. Je důležité zmínit, že dynamika prostředí se v procesu mapování projeví, až po uplynutí určité doby. Většina řešení SLAM problematiky je koncipována do statického prostředí. Vzhledem k časovému horizontu použití, kdy je robot vyslán na jednorázovou expedici, se dynamika prostředí nestihne projevit. V dlouhodobém užití je nutné zapracovat dynamiku prostředí do konceptu řešení SLAMu a adekvátně měnit vnitřní model tohoto prostředí. Vynechání dynamiky by vedlo k nekonzistentnímu porovnávání modelu a naměřených dat a proces lokalizace by začal selhávat.

4.2 Dlouhodobá lokalizace

O dlouhodobé lokalizaci se hovoří v souvislosti s délkou nasazení robotu v daném prostředí. Robot, který bude nasazován dennodenně v provozní hale, musí být schopný kontinuálně měnit mapu prostředí a poskytovat údaj o své poloze v reálném čase. Oproti běžné SLAM problematice je dlouhodobý SLAM více obtížnější, především kvůli nárokům na použitou paměť a výpočetní čas.

4.2.1 Matematická formulace

Určování polohy robotu lze chápat jako míru pravděpodobnosti, se kterou se v daném místě nachází. Pravděpobnostní formulace dlouhodobého SLAMu umožňuje formální reprezentaci nejistot zašumělých dat [10]. Dlouhodobý SLAM lze definovat jako

$$p(\boldsymbol{x}_{1:t}, m_t | \boldsymbol{z}_{1:t}, \boldsymbol{u}_{1,t-1}, m_0, \boldsymbol{x}_0), \tag{4.1}$$

kde $\boldsymbol{x}_t$ je odhad pozice robotu a m_t je mapa prostředí v čase t, $\boldsymbol{z}_{1:t}$ jsou současná pozorování, $\boldsymbol{u}_{1:t}$ jsou řídicí vstupy, m_0 je dlouhodobá mapa a $\boldsymbol{x}_0$ je počáteční poloha [11].

4.3 Koncept řešení

Přestože algoritmů řešících SLAM je mnoho, většina z nich vychází ze tří hlavních metod, kterými jsou rozšířené Kalmanovy filtry (Extended Kalman filter - EKF), částicové metody (Particle methods) a grafově založené techniky (Graph-based SLAM) [10].

Metoda použita v této práci je založena na grafové technice a využívá hybridní NDT mapy. Velmi užitečnou vlastností této techniky je schopnost tvorby mapy ve 2D a 3D bez jakékoliv úpravy algoritmu [1]. Kromě laserového dálkoměru lze použít libovolný hloubkový senzor (Kinect, 3D laserový dálkoměr, RGB-D kamera, sonar). Mapovací proces využívající mřížku obsazenosti (occupancy grid) dokáže modelovat překážky i volný prostor.

4.3.1 Grafový SLAM

Graf je tvořen hranami a uzly. Uzly grafu představují odhad pozice robotu a hrany, tvořené odometrií, reprezentují transformaci mezi uzly.

Tvorba grafu pozic

Úkolem této části je vytvoření grafu, která zaznamená trajektorii robotu. Na vstup algoritmu přicházejí informace z odometrie nebo IMU. S narůstající chybou odometrie klesá věrohodnost odhadu pozice [1]. Při překročení určité meze věrohodnosti se utvoří uzel grafu v_i, který je spojený s předchozím vrcholem v_{i-1} odometrickou hranou. Hrana představuje transformaci $\boldsymbol{\delta}_{i,i-1}(x, y, \theta)$ popisující změnu pozice z uzlu v_{i-1} do v_i. Tímto způsobem vzniká řetězec uzlů spojených hranami reprezentující pohyb robotu.

Přirozeným chodem robotu může nastat situace, kdy robot navštíví pozici, ve které se nacházel již dříve. V grafu pozic je tento fakt reprezentován smyčkou. Ve smyčce se nachází hrana spojující dva uzly, které nesou stejné informace o okolí [2]. Kdyby smyčky nebyly detekovány, uzly v grafu by se stále řetězily za sebe, což by mělo za následek zbytečné narůstání dat popisujících již zmapovanou oblast. Navíc by jedné pozici odpovídal nejeden uzel. Následkem by byla nejednoznačnost v procesu určování pozice.

Detekce uzavřených smyček

Cílem detekce uzavřené smyčky je najít takovou dvojici uzlů, které se svými naměřenými daty překrývají a nesou tak společné informace popisující prostředí. Jde o místa, která již robot navštívil, ne však v nejbližší minulosti. Místa navštívena v blízké minulosti evidentně budou naměřenými daty překrývat současnou polohu, avšak uzavřenou smyčku netvoří.

Pro daný uzel a je potřeba najít všechny uzly $b_1, b_2, \ldots, b_n; n \in \mathbb{N}$, které naměřenými daty překrývají uzel a [2]. Přesněji jde o problém nalezení relativních pozic uzlů a a $b_i, i \in \langle 1, n \rangle$.

Řešením je Dijkstra projekce, která začíná v uzlu a, a podle minimální nejistoty zřetězí kovarianční matice a transformace. Minimální nejistota je určena na základě determinantu kovarianční matice. Kovarianční matice s malými prvky má nižší determinant než kovarianční matice s velkými [2].

Po generaci překrývajících se uzlů je každá potenciální dvojice uzlů testována na základě vzájemné topologické vzdálenosti. Pokud není dvojice zamítnuta je hrana mezi nimi zařazena do grafu. Přidávání hran v procesu hledání uzavřených smyček zanáší do grafu chyby. Špatně utvořené uza-

vřené smyčky mají katastrofický dopad na tvorbu mapy. Hledání nesprávně zařazených hran má na starosti neustále pracující optimalizační sekce.

Optimalizace

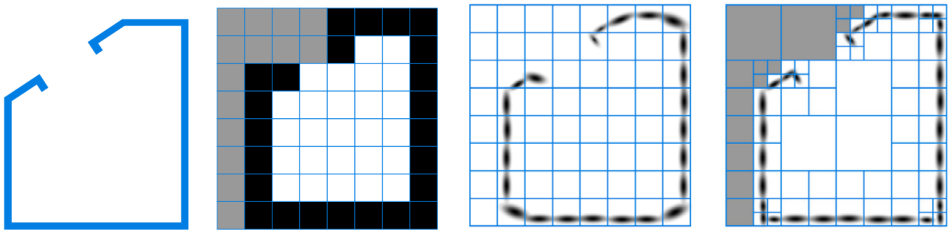
Pozice robotu $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n$ jsou reprezentovány uzly $v_1, v_2, \ldots, v_n$. Dvě po sobě jdoucí pozice $\boldsymbol{x}_i, \boldsymbol{x}_j$ jsou spojeny odometrií a senzorickými daty. V grafu pozic tomu odpovídá transformace $\boldsymbol{\delta}_{j,i}$, která spojuje uzly v_i a v_j. Vyvstává optimalizační problém [1]

$$\boldsymbol{X}^* = \underset{\boldsymbol{X}}{\operatorname{argmin}} \sum_{i,j} \boldsymbol{e}(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{\delta}_{j,i})^T \boldsymbol{\Omega}_{j,i} \boldsymbol{e}(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{\delta}_{j,i}), \tag{4.2}$$

kde $\boldsymbol{X} = (\boldsymbol{X}_1^T, \boldsymbol{X}_2^T, \ldots, \boldsymbol{X}_n^T)^T$ je vektor odhadu pozic všech vrcholů v grafu, $\boldsymbol{e}(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{\delta}_{j,i})$ je chybová funkce, která popisuje jak dobře zapadají uzly v_i a v_j do transformace $\boldsymbol{\delta}_{j,i}$, a $\boldsymbol{\Omega}_{j,i}$ je informační matice, která je inverzí kovarianční matice. Cílem je najít takové $\boldsymbol{X}$, při kterém je rovnice 4.2 minimální. Výsledkem optimalizace je nejvěrohodnější vektor odhadů pozic $\boldsymbol{X}^*_{1:n}$, který reprezentuje trajektorii robotu [1].

4.3.2 Mapování transformací normálního rozdělení

Mřížka obsazenosti slouží k zasazení dat z laserového dálkoměru do mřížky s buňkami. Na rozdíl od běžných metod založených na mřížce obsazenosti, kde buňka nabývá dvou stavů (obsazená, prázdná), je obsah NDT buňky aproximován normálním rozdělením. Diskrétní buňky jsou uspořádány do stromové struktury, která umožňuje tvorbu mapy s proměnnou velikostí buněk v závislosti na členitosti prostředí. Pro 2D mapování je použita stromová struktura (quadtree), kde každá rodičovská buňka má čtyři potomky [1].



Obrázek 4.1: Mapy využívající mřížku obsazenosti. Zleva: tvar prostředí, dvoustavová mapa, NDT mapa, hybridní NDT mapa využívající stromovou strukturu; převzato z [1]

Naměřené body z laserového dálkoměru ležící v jedné rovině jsou rozděleny do odpovídajících buněk. NDT mřížka dovoluje sloučení nových měření se starými, čímž přidává nové informace o okolí. K popsání každé buňky je zapotřebí získat parametry dvourozměrného normálního rozdělení $N(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$, kde $\boldsymbol{\mu}_c$ je sloupcový vektor středních hodnot buňky z prostoru $\mathbb{R}^2$, a $\boldsymbol{\Sigma}_c$ je kovarianční matice $\mathbb{R}^2 \times \mathbb{R}^2$. Kovarianční matice je složená z kovariancí mezi složkami vektoru, je symetrická, pozitivně definitní a prvky na diagonále odpovídají rozptylům složek středních hodnot [12]. Pravděpodobnost obsazenosti buněk mapy mapy M lze definovat jako

$$p(\boldsymbol{x} \in s) = \sum_{c \in M} \omega_c N(\boldsymbol{x} | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c), \tag{4.3}$$

kde naměřený bod $\boldsymbol{x} \in \mathbb{R}^2$ náleží množině bodů s, které leží na povrchu objektů nacházejících se v prostředí. K reprezentování volného místa je v každé buňce přidána hodnota obsazenosti o_c. Reprezentování volného místa explicitně je důležité pro plánování trasy robotu. Konečné pravděpodobnostní rozdělení povrchu bodů objektu je vyjádřeno jako $o_c N(\boldsymbol{x} | \boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$ [1]. Prázdné buňky jsou modelovány s rovnoměrným rozdělením pravděpodobnosti a na rozdíl od obsazených buněk neobsahují Gausián.

Na začátku procesu mapování je stav buňky neznámý. Nejjednodušší cestou k modelování neznámých stavů buněk je nastavení hodnoty obsazenosti na $o_c = 0,5$. S každým novým měřením se upravují hodnoty buněk, následkem čehož se začne formovat mapa. Každé měření je popsáno pozicí senzoru $\boldsymbol{p}$, směru paprsku $\boldsymbol{d}$ a naměřené vzdálenosti z. Pomocí těchto parametrů je koncový bod $\boldsymbol{x} \in \mathbb{R}^2$ definován jako $\boldsymbol{x} = \boldsymbol{p} + z\boldsymbol{d}$ [1]. K zapracování naměřených dat do mapy je zapotřebí najít všechny buňky, kterým náleží naměřené koncové body, a poté upravit hodnoty normálního rozdělení v buňkách pomocí následujících přírůstkových pravidel:

$$\begin{aligned} \boldsymbol{\mu}'_c &= \alpha \boldsymbol{\mu}_c + (1 - \alpha)\boldsymbol{x}, \\ \boldsymbol{\Sigma}'_c &= \alpha \boldsymbol{\Sigma}_c + \alpha(1 - \alpha)(\boldsymbol{\mu}_c - \boldsymbol{x})(\boldsymbol{\mu}_c - \boldsymbol{x})^T, \\ k'_c &= k_c + 1, \end{aligned} \tag{4.4}$$

kde $\alpha = \frac{k_c}{k_c+1}$ a k_c představuje počet změn buňky [1]. V dynamickém prostředí je třeba hodnotu k_c každé buňky limitovat. Díky tomuto omezení mají nově naměřená data vyšší dopad na tvorbu mapy, která se rychle přizpůsobí změnám prostředí.

Po každém měření je třeba upravit parametry normálního rozdělení buněk umístěných ve stromové struktuře. Parametry rodiče p jsou rekurzivně

počítány z jeho potomků

$$\begin{aligned} \boldsymbol{\mu}_p &= \sum_i \omega_i \boldsymbol{\mu}_i, \\ \boldsymbol{\Sigma}_p &= \sum_i \omega_i \boldsymbol{\Sigma}_i + \beta_i (\boldsymbol{\mu}_p - \boldsymbol{\mu}_i)^T, \\ \omega_i &= \frac{k_i}{k_p}, k_p = \sum_i k_i. \end{aligned} \tag{4.5}$$

Stromová struktura umožňuje generování map s různým rozlišením odpovídajícímu hloubce zanoření ve stromu [1].

4.3.3 NDT registrační algoritmus

Registrační algoritmus řeší spojování starších snímků z lidaru s právě naměřenými. Úkolem je najít 2D transformaci

$$\begin{bmatrix} x' \\ y \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}, \tag{4.6}$$

která popisuje vztah mezi dvěma snímky [2]. V rovnici 4.6 $[t_x, t_y]^T$ představuje translaci a úhel θ rotaci transformace.

Obvykle by nyní následovalo počítání hodnotící funkce pro každou dvojici bodů, která by našla nejvěrohodnější transformaci mezi dvěma snímky. V metodě NDT je však obecné pravděpodobností rozdělení transformováno na rozdělení normální. Na místo porovnávání každé dvojice bodů, lze porovnávat rozdělení v dvojici buněk. Algoritmus minimalizuje součet L_2 (Lebesgueových) vzdáleností mezi dvojicí pravděpodobnostních rozdělení. Transformace mezi dvěma množinami buněk X a Y je definována jako

$$f(\boldsymbol{p}) = \sum_{i=1,j=1}^{n_x.n_y} -d_1 \exp\left(-\frac{d_2}{2} \boldsymbol{\mu}_{ij}^T (\boldsymbol{R}^T \boldsymbol{\Sigma}_i \boldsymbol{R} + \boldsymbol{\Sigma}_j)^{-1} \boldsymbol{\mu}_{ij} \right), \tag{4.7}$$

$$\boldsymbol{\mu}_{ij} = \boldsymbol{R}\boldsymbol{\mu}_i + \boldsymbol{t} - \boldsymbol{\mu}_j, \tag{4.8}$$

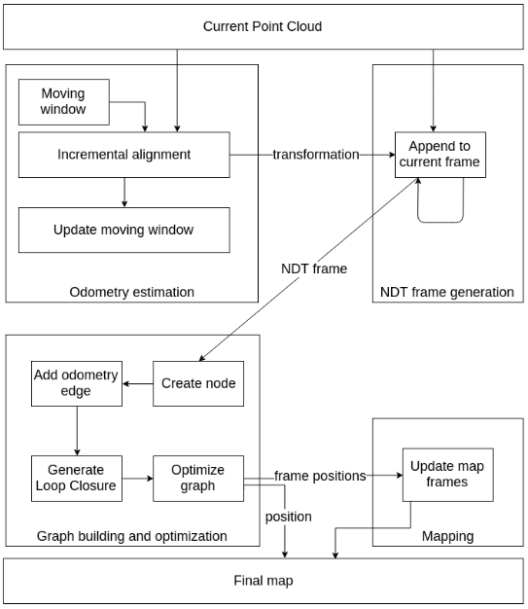
kde $\boldsymbol{p} = (t_x, t_y, \theta)$ je transformace ve 2D prostoru, $X(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ a $Y(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$ jsou funkce pravděpodobnostního rozdělení jednotlivých dvojic buněk množin X a Y a $(\boldsymbol{R}, \boldsymbol{t})$ reprezentují rotační matici vytvořenou z úhlu θ a translační vektor $\boldsymbol{t} = [t_x, t_y]^T$. Rovnice 4.7 obsahuje dva regulační parametry $d_1 = 1$ a $d_2 = 0.05$. Rovnice 4.8 představuje rozdíl vektoru středních hodnot po transformaci do nové pozice. Způsob maximalizace parametru $\boldsymbol{p}$ rovnice 4.7 je blíže popsán v práci [2]. S vytvořením mřížky je starý Point Cloud zahozen, což významně šetří paměť. Výhodou porovnávání celého rozdělení v buňce namísto jednotlivých bodů je schopnost provádět registraci téměř desetkrát rychleji [2].

4.4 Struktura implementace

Implementace řešící problematiku NDT-SLAM je převzata z od kolegy Lukáše Jelínka z Matematicko-fyzikální fakulty Univerzity Karlovy. Původní práce [2] popisuje implementaci detalněji, přesto je vhodné zde popsat hlavní třídy programu. Program je velmi dobře členěn a jednoduše spolupracuje s ostatním softwarem. Hlavní třídou je **NDTGrid2D**, ve které se tvoří mřížka s diskrétními buňkami, probíhá zde registrace snímků z lidaru. Jsou zde začleněny funkce na změnu velikosti buněk, správu použité paměti, vyhledávání, slučování a aktualizaci buněk.

Vyšší abstrakci obecné mřížkové struktury zajištuje třída **VoxelGrid2D**. Mřížka je reprezentována jako jednorozměrný vektor, který v sobě uchovává ukazatele na buňky. Prázdným buňkám odpovídá nulový ukazatel. Mřížka je inicializována prázdná a je schopna dynamicky měnit svoji velikost.

Důležitou třídou je **NDTCell**, která uchovává kovarianční matici a vektor středních hodnot buňky. V této třídě probíhá aktualizace parametrů normálního rozdělení podle přírůstkových pravidel 4.4. Na obrázku níže je popsán proces grafového NDT-SLAMu.



Obrázek 4.2: Diagram popisující proces NDT-SLAMu; převzato z [2]

Kapitola 5

Hardware

V této kapitole bude popsán použitý hardware. Bude představen výzkumný robot Jackal, lidar Sick TiM361 a kamera Pylon.

5.1 Lidar Sick TiM361

V této práci je lidar stěžejním senzorem pro tvorbu mapy a kontinuální lokalizaci. V kapitole 3.3 byl popsán princip fungování lidaru a pro jeho vlastnosti je vhodným vstupním senzorem do procesu mapování popsaného v kapitole 4.3. Společnost SICK se prakticky stala synonymem inteligentních senzorů. Lidar TiM361 je kvůli svým parametrům, vlastnostem a účelu budoucího použití rozumnou volbou.

Tento model rozmítá laserový paprsek v horizontální rovině. S rozsahem 270 ° a rozlyšením 0.33 ° je schopný emitovat více než 800 paprsků v jednom snímku [13]. Díky odolnosti vůči přímému slunečnímu světlu ho lze použít ve vnitřních i vnějších aplikacích. K robotu Jackal je připojen skrze ethernetový kabel. Napájení lidaru zajišťuje robot Jackal skrze připojení na napájecí port s napětím 12 V.

Parametry lidaru Sick TiM361	
vlnová délka laseru	850 nm
třída nebezpečnosti	1, eye-safe
úhel rozsahu	270 °
frekvence snímků	15 Hz
úhlové rozlišení	0.33 °
rozsah měření	0.05 - 10 m
systematická chyba	±60 mm
statická chyba	20 mm
připojení	ethernet, USB
pracovní napětí	9 - 28 V DC
spotřeba energie	4 W
váha	250 g bez kabeláže
rozměry	60 x 60 x 86 mm
odolnost vůči okolnímu osvětlení	80000 lx

Tabulka 5.1: Parametry lidaru Sick TiM361

5.2 Kamera Pylon

Kamera DSL215 je umístěna nad lidarem a míří vzhůru. Pro budoucí práce se pomocí obrazů z kamery bude řešit globální lokalizace s hrubší přesností a bude podchycovat situace, kdy robotu selže senzorický systém. Kamera poskytuje barevný RGB obraz s rozlišením 5 Mpx a je vybavena rybím okem se zorným úhlem 185 ° [14]. Přestože ji lze připojit pomocí USB 3.0, k robotu Jackal je připojena skrze nižší, robotem podporovanou verzi USB 2.0.

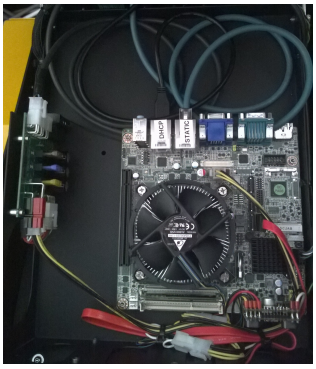
5.3 Robot Jackal

Robot Jackal od společnosti Clearpath Robotics je čtyřkolové pozemní vozítko určené k výzkumu a vývoji nových aplikací v oblasti mobilní robotiky. Voděodolný kryt umožňuje provoz robotu i v nehostinných venkovních podmínkách. Pohyb zajišťují dva motory s celkovým výkonem 500 W, kde každý pohání současně dvě kola vždy na jedné straně (náhon 4x4). Diferenciální kinematika umožňuje otáčení robotu na jednom místě.

Uvnitř robotu je umístěn lithium-iontový akumulátor s kapacitou 270 Wh. Nabíjení trvá okolo 4 hodin a v provozu běžně vydrží 5-6 hodin v závislosti na použití. Zpřístupněné jsou napájecí konektory s napětím 5 V, 12 V a 24 V

pro napájení dalších periferních zařízení. Mimo baterii je v robotu umístěný řídicí počítač (Celeron J1800, Dual core 2.4GHz, 2BG RAM, 32 GB Hard Drive) s operačním systémem Ubuntu 14.04, ve kterém běží ROS Indigo. K počítači lze připojit další zařízení skrze dva ethernetové porty - statický port s adresou 192.168.1.11 a DHCP port, nebo skrze USB 3.0. (Pozn.: Připojení periferního zařízení skrze USB 3.0 způsobí neschopnost spuštění počítače. Doporučuji použít nižší verzi USB 2.0.) Lze také připojit zobrazovací zařízení skrze VGA nebo HDMI port. Běžná komunikace je zajištěna skrze Wi-Fi adaptér. Existuje také možnost ovládání pohybu robotu pomocí bezdrátového herního ovladače využívajícího technologii Bluetooth [15].

Ve standardní výbavě robot Jackal obsahuje gyroskopy a akcelerometry tvořící IMU, odometrii a poskytuje informaci o napětí na baterii, napájecích pinech a na motorech. Se svojí velikostí (508 x 430 x 250 mm) a hmotností (17 kg) se dá převážet do různých testovacích prostředí, popřípadě přenést přes nezdolatelné překážky a současně zajišťuje určitou míru robustnosti a stability (například ve srovnání s robotem Turtlebot 2).



(a) : Vnitřek robotu

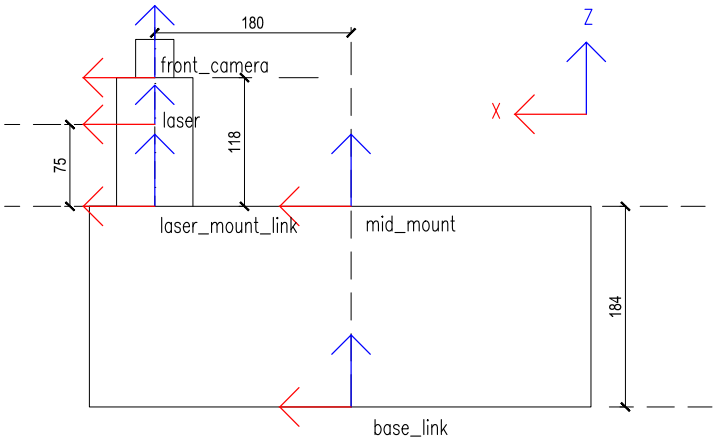
(b) : Kompletní experimentální sestava

(c) : Lidar s kamerou

Obrázek 5.1: Fotky použitého hardwaru.

5.3.1 Experimentální sestava

Pro účely lokalizace je důležité znát polohu senzorů. Je tedy třeba změřit vzdálenosti mezi souřadnicovými systémy. Na obrázku níže je vidět řez experimentální sestavy rovinou xz, která robot, lidar i kameru protíná středem. Vzájemné vztahy mezi nimi lze popsat planárně.



Obrázek 5.2: Vztahy souřadnicových systémů robotu, lidaru a kamery v rovině xz.

Kapitola 6

Software

Tato kapitola se zabývá popisem použitého softwaru. Bude zde představen operační systém pro roboty (ROS), programy určené robotu Jackal, laserovému dálkoměru Sick TiM361 a kameře Pylon. V neposlední řadě bude také popsán program řešící 2D SLAM problematiku.

6.1 Robot Operating System

Robot Operating System (ROS) je volně dostupný operační systém určený pro roboty obsahující mnohé užitečné nástroje, knihovny, hardwarové ovladače a balíky programů. V posledních letech ROS nabyl na významnosti. Díky velkému počtu programátorů používajících ROS vzniklo mnoho výukových lekcí a aktivních fór, kde se diskutuje nad řešením každodenních problémů vývojářů. ROS v současné době čítá přes 3000 balíků [16], běží na Linuxu, umožňuje chod dalších programů a snadno spolupracuje s dalšími knihovnami jako je OpenCV, MoveIT, PCL, MRPT a další.

ROS je tvořen balíky ("packages"), uzly ("nodes"), komunikačními kanály ("topics"), zprávami ("messages") a službami ("services"). Balíky jsou logicky ucelné části softwaru obsahující zpravidla několik uzlů. Uzly jsou spustitelné soubory provádějící výpočty. Výměna informací mezi uzly probíhá pomocí kanálů nebo služeb. Komunikační kanály jsou prvky spojující uzly. Je možné, aby jeden uzel vysílal data do více kanálů, nebo je přijímal z více kanálů. Data jsou v komunikačním kanále přenášena pomocí zpráv, které jsou různě

strukturované v závislosti na typu přenášených dat. Zpráva může využívat jiný typ zprávy jako svoji součást. Výhodou je ustálená struktura, která se používá standardně ve většině balíků, a proto lze jednoduše propojovat různé kusy softwaru bez větších úprav. Posledním prvkem ROSu jsou služby zajišťující druhý způsob předání informací mezi uzly. Pokud uzel potřebuje danou službu, pošle žádost, cílový uzel ji obdrží, vykoná danou službu a pošle odpověď žadateli. Program nepokračuje ve své činnosti, dokud není služba vyřízena.

ROS existuje v několika verzích. Každá verze odpovídá písmeni v abecedě doplněného o kódové slovo - Fuerte, Hydro, Indigo, Jade, Kinetic [17]. V této práci je stěžejně použit ROS Kinetic, ve kterém probíhají všechny výpočty. Verze Kinetic je primárně určena operačnímu systému Ubuntu 16.04 (Xenial). Jelikož se firma Clearpath Robotics doposud nerozhodla přestoupit na novější verzi, zůstává v robotu Jackal ROS Indigo běžící v operačním systému Ubuntu 14.04 (Trusty Tahr). V této starší verzi probíhá sběr dat ze senzorů (laserový dálkoměr, kamera, odometrie).

V této práci ROS představuje hlavní dějiště veškerých procesů od ovládání motorů a sběr dat ze senzorů, přes jejich zpracování, až po optimalizační algoritmy nebo prováděné simulace.

6.1.1 Rviz

Při běžné práci nevyhnutelně nastane situace, kdy je potřeba zkoumat naměřená data. Je možné si obsah komunikačního kanálu vypsat do terminálu. Touto cestou jsou data prezentována v syrové formě hodící se pro výpočetní algoritmy. Rychlým a jednoduchým způsobem zobrazení dat je 3D vizualizační nástroj Rviz. Rviz slouží k vizualizaci jakéhokoliv dění v ROSu. V reálném čase vykresluje 3D model robotu, změny báze, naměřená data, obraz z kamery a další. V Rvizu lze interaktivně nastavovat trajektorii robotu a plánovat tak jeho pohyb [18]. V neposlední řadě umožňuje vizualizovat časové řady naměřených dat (kompas, odometrie). V této práci je Rviz použit jako hlavní vizualizační nástroj především pro zobrazení naměřených dat z laserového dálkoměru, sestavené mapy a pro simulace nad naměřenými daty. Rviz se spouští následujícím příkazem.

```
rosrun rviz rviz
```

6.1.2 Rosbag

Cenným nástrojem pro uchování naměřených dat a zaznamenávání stavu robotu je Rosbag [19]. Pomocí nástroje Rosbag lze zaznamenávat jakékoliv zprávy libovolného komunikačního kanálu. Naměřená data jsou k dispozici pro pozdější přehrání a další zpracování. Tento postup je velmi praktický, neboť pokaždé, kdy je třeba testovat algoritmus a zkoušet různé nastavení parametrů, není nutné znovu provádět měření.

Komunikační kanály vhodné k nahrávání a pozdější zpracování jsou uvedeny v tabulce níže. Příkaz k nahrávání komunikačních kanálů:

```
rosbag record odometry/filtered scan tf
    pylon_camera_node/image_raw
```

Výčet komunikačních kanálů vhodných k nahrávání	
/scan	data z laserového dálkoměru
/odometry/filtered	odometrická data
/pylon_camera_node/image_raw	data z kamery
/tf	transformace bází

Tabulka 6.1: Nahrávané komunikační kanály

6.2 Použité balíky

6.2.1 Laserový dálkoměr Sick TiM361

Pro zajištění komunikace s laserovým dálkoměrem, nastavením parametrů a sbíráním dat je použit balík *sick_tim*. V tomto balíku se nachází několik uzlů odpovídajících modelům laserového dálkoměru. V naší práci byl použit model Sick TiM361 5.1 z rodiny TiM3xx. Odpovídající uzel je *sick_tim551_2050001*, který nastaví patřičné parametry a založí komunikační kanál */scan*. Vizualizace dat z laserového dálkoměru je ukázána na obrázku níže.

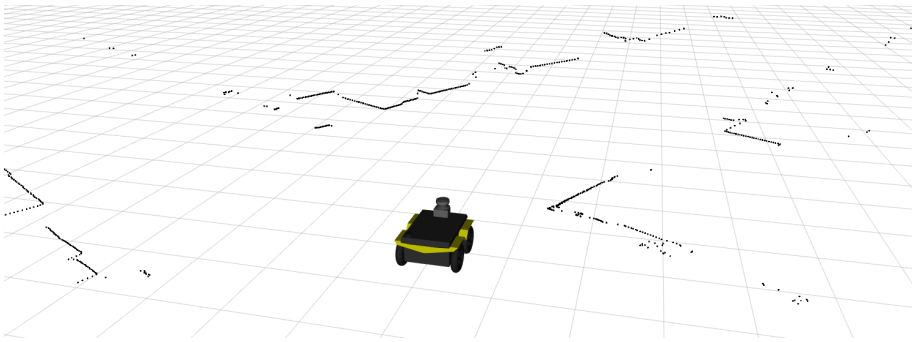
Podle návodu [20] je třeba v programu SOPAS-ET (pouze pro Windows) povolit výstup RSSI (Recieved Signal Strength Indicator). V tomto programu lze mimo jiné změnit IP adresu lidaru, pod kterou bude vystupovat v síti. Pro správné fungování je třeba zajistit, aby lidar a zařízení, ke kterému je skrze standardní ethernetové rozhraní připojeno, měly stejnou adresu podsítě.

V našem případě je lidar připojen na statický port robotu Jackal s adresou 192.168.1.11, proto byla původní adresa laserového dálkoměru 192.168.2.1 změněna na 192.168.1.22. Po přenastavení IP adresy konfiguračním programem SOPAS je potřeba nastavit IP adresu a port v uzlu *sick_tim551_2050001* editací poslední sekce. Uzel musí být spuštěn přímo na robotu Jackalovi, nikoliv na pracovním notebooku. Příkaz ke spuštění:

```
roslaunch sick_tim sick_tim551_2050001.launch
```

Nastavené parametry	
frekvence měření	15 Hz
úhel rozlišení	1 °
rozsah měření	0.05 - 10 m
počáteční úhel	-2.35 rad (-135 °)
koncový úhel	2.35 rad (135 °)
IP adresa	192.168.1.22

Tabulka 6.2: Ukázka důležitých parametrů nastavených laserovému dálkoměru Sick TiM561



Obrázek 6.1: Ukázka vizualizace dat z laserové dálkoměru spolu s modelem robotu v nástroji Rviz

6.2.2 Robot Jackal

Důležitá kolekce je *jackal_simulation*, která obsahuje balíky: *driver_common, gazebo_ros_pkgs, jackal a jackal_simulator*. Soubor experimentálních balíků pracující v ROS Kinetic nahrazuje software od Clerpath robotics, který je podporován pouze verzí Indigo. Tato kolekce zajišťuje veškerou práci s Jackalem. Umožňuje nejen základní operace jako je zasílání příkazů do motoru nebo sledování výstupního napětí na napájecích pinech konektorů, ale poskytuje i sofistikovanější služby jako je 3D model robotu [21] nebo připravené balíky pro navigaci a mapování. Správný chod zajišťuje spolupráce s dalšími balíky: *control_msgs, control_toolbox, realtime_tools, ros_control, ros_controllers a ros_vizualization*, na kterých je tato kolekce závislá.

Nastavení sítě

Po prvním spuštění je potřeba nastavit síť, ke které se bude robot Jackal automaticky připojovat. Na této síti bude probíhat veškerá komunikace. Je vhodné promyslet dopředu, v jakých místech bude později robot pracovat a nastavit adekvátní síť. Doporučuji vytvořit na notebooku nový přístupový bod (AP). Takové nastavení není vázáno na konkrétní oblast závislou na pokrytí Wi-Fi signálu, ale umožňuje nasazovat robota do různých prostředí.

K nastavení sítě vedou dvě cesty. První cestou je připojení klávesnice a monitoru k robotu, druhou je připojení notebooku ke statickému portu s IP adresou 192.168.1.11. V takovém případě je potřeba nastavit IP adresu notebooku na stejnou podsíť (např. 192.168.1.87). Po přihlášení se pomocí příkazu

```
wicd-curses
```

dostaneme do síťového nastavení. Vybereme síť, ve které chceme s Jackalem pracovat, tlačítkem F10 uložíme nastavení a klávesou C opustíme nastavení.

	klávesnice + monitor	ssh spojení
uživatelské jméno	administrator	ssh administrator@192.168.1.11
heslo	clearpath	clearpath

Tabulka 6.3: Přihlašovací údaje k robotu Jackal

Nastavením sítě však práce nekončí. Nyní je potřeba nastavit ROS Master [22], který zajistí spojení s Jackalem. Dle doporučení [23] vytvoříme soubor *remote-jackal.sh* s následujícím obsahem.

```
# Jackal's hostname : port number
export ROS_MASTER_URI=http://cpr-j100-0093:11311
# Your laptop's wireless IP address
export ROS_IP=<laptop IP> #example: 10.42.0.1
```

Používané uzly

Robot Jackal poskytuje mnoho uzlů, pro naši práci je důležitý pouze uzel popisující model robotu. Modelu byly upraveny báze laserového dálkoměru a kamery, která odpovídá skutečné experimentální sestavě popsané v sekci 5.3.1. Model robotu lze vyvolat pomocí příkazu:

```
roslaunch jackal_description description.launch
```

Pro pohodlnější ovládání robotu byl přidán uzel *jackal_teleop* převzatý z Turtlebot 2 [24], kterému byly upraveny komunikační kanály a parametry vzhledem k rozdílným rozměrům, maximálním rychlostem apod. Byla přidána funkce tempomatu, která udržuje nastavenou rychlost. Tuto rychlost lze zvyšovat a snižovat při pohybu robotu. Řízení je snazší a plynulejší, neboť uživatel pouze koriguje směr. Tempomat se dá vypnout ručně i automaticky, když uživatel začne brzdit. Příkaz pro spuštění:

```
roslaunch jackal_teleop keyboard_teleop.launch
```

6.2.3 Kamera Pylon

Ovládání kamery Pylon zajišťuje balík *pylon_camera*. Po instalaci popsané v návodu [25] je dostupný uzel *pylon_camera_node*, který založí hned několik komunikačních kanálů. Nejdůležitější je kanál */pylon_camera_node/image_raw*, který přenáší obraz z kamery. Balík zajistí automatické nastavení parametrů jako jsou expozice, zesílení, gamma korekce intenzity pixelů a světlost. Důležitými parametry nastavitelnými uživatelem jsou počet snímků za sekundu a báze, ve které bude kamera vysílat.

V naší konfiguraci je frekvence snímků nastavena na jeden snímek za sekundu. Při používané rychlosti nepřesahující 0,5 ms^{-1} kamera poskytuje dva snímky na metr ujeté vzdálenosti, což je pro naše využití více než dostačující. Uzel je nutné spouštět přímo na robotu Jackal, ke kterému je kamera připojena skrze USB 2.0 kabel. Příkaz pro spuštění:

```
roslaunch pylon_camera pylon_camera_node.launch
```

Komprese obrazu

Přenášení a vizualizace obrazu z kamery zatěžuje komunikační kanál, což má za následek zpoždění příchozích obrazů a neplynulý chod. Řešením tohoto problému jsou balíky *image_common* [18] a *image_transformation* [26] zajišťující transformace obrazu (compressed, compressedDepth, raw a theora).

Plynulejší přenos obrazů zajišťuje transformace theora, která komprimuje obraz z kamery s malou šířkou pásma. Komprimovaná data slouží pouze k přenosu v komunikačním kanálu, pro výpočty a další zpracování se používá původní nekomprimovaný obraz.

NDT-SLAM

Samotné řešení 2D SLAM problému je obsaženo v balíku *dynamic_slam* [27]. Tomuto balíku byly upraveny komunikační kanály, tak aby implementace fungovala na robotu Jackal. Byl vytvořen nový uzel *graph_slam_jackal*, který řeší SLAM s parametry uvedenými v tabulce níže. Většinu těchto parametrů určuje prostředí, ve kterém je robot nasazen.

Program může využívat odometrická data, ale je schopný pracovat pouze s laserovým dálkoměrem. Poskytuje dvě mapy. První mapa uchovává pouze okolí robotu a může být užitečná pro vyhýbání se překážkám. Druhá mapa obsahuje informace o celém prostředí [2].

Implementace požaduje komunikační kanály s daty z laserového dálkoměru (*/scan*) a odometrie (*/odometry/filtered*). Po spuštění se založí komunikační kanály. Důležité jsou */slam/graph, /slam/map, /slam/map_pcl*, které popisují vizualizaci grafu pozic robotu, vytvořenou mapu prostředí a PCL reprezentaci mapy. Příkaz pro spuštění:

```
roslaunch ndt_gslam graph_slam_jackal.launch
```

Parametry SLAM		
parametr	hodnota	popis
scanmatch_window_radius	30.0	Poloměr (v metrech) oblasti, ve které probíhá inkrementální registrace snímků z lidaru.
node_gen_distance	1.0	Euklidovská vzdálenost (v metrech) dvou uzlů v grafu pozic.
loop_max_distance	30.0	Vnější poloměr (v metrech) oblasti, ve které bude probíhat detekce uzavřených smyček.
loop_min_distance	2.5	Vnitřní poloměr (v metrech) oblasti, ve které bude probíhat detekce uzavřených smyček.
loop_score_threshold	0.85	Hranice zamítnutí uzavřené smyčky v rozsahu [0,1]. V případě vyššího ohodnocení smyčky, než je hranice, je smyčka zařazena do grafu

Tabulka 6.4: Přehled důležitých parametrů uzlu *graph_slam_jackal*.

6.3 Použité knihovny

6.3.1 Point Cloud Library

Point Cloud Library (PCL) [28] je volně dostupná knihovna usnadňující práci s naměřenými daty z laserového dálkoměru. Je dostupná na všech běžných operačních systémech (Windows, MacOS, Linux, Android). PCL snadno umožňuje vizualizaci a filtrování dat, povrchovou rekonstrukci, modelování, planární segmentaci a další často používané operace. PCL je napsána v jazyce C++ a skládá se z menších knihoven, které lze kompilovat odděleně. PCL spolu s ROS umožňuje vizualizaci dat v reálném čase. V této práci jsou Point Cloudy používaný jako základní struktura dat a PCL je využita k jejich registraci. Registrace je řešena pomocí minimalizace součtu kvadrátů vzdáleností jednotlivých dvojic bodů metodou bod po bodu ("point-to-point").

6.3.2 G2O

Knihovna G2O [29] napsaná v jazyce C++ je známá a velmi často používaná knihovna v problematice grafového SLAMu. Lze ji používat jako součást ROS. Slouží k optimalizaci grafu pozic, konkrétněji hledá nelineární chybovou funkci, při které je rovnice 4.2 minimální. V této práci je G2O knihovna použita jako hlavní nástroj optimalizace grafu pozic.

6.3.3 Eigen

Knihovna Eigen [30] je standardní knihovnou ROS, která dokáže numericky řešit rovnice, umožňuje běžné operace lineární algebry a práci s maticemi. V této práci je použita především pro výpočet transformačních matic.

Kapitola 7

Experimenty a výsledky měření

V této kapitole bude popsáno testování funkčnosti algoritmu v různých prostředích, ve kterých byl robot nasazen. Práce [2], odkud pochází implementace, je zaměřena na naprogramování metody využívající transformaci normálního rozdělení a funkčnost algoritmu byla testována pouze na veřejně dostupných datech z univerzity MIT (Massachusetts Institute of Technology). Jedním z vytyčených cílů této práce je ověřit funkčnost algoritmu v reálných podmínkách. Nejde tedy jen o schopnost vytvořit mapu prostředí, ale také o zpracování dat a poskytnutí informace o poloze v reálném čase.

Všechna měření byla prováděna na experimentální soustavě popsané v sekci 5.3.1. Použité parametry lidaru, kamery a implementace jsou popsány v sekcích 6.2.1 až 6.2.3. S ohledem na požadavek zpracování dat v reálném čase rychlost robotu v experimentech nepřesahuje 0,5 ms^{-1}. V budoucích aplikacích nebude použita rychlost o moc vyšší, především z důvodu zajištění bezpečnosti lidí na pracovišti.

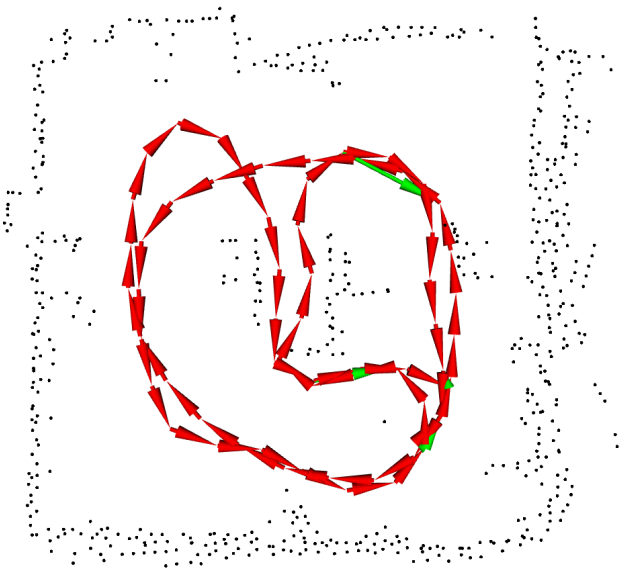
7.1 Ověření funkčnosti ve vnitřních prostředích

Nejprve bude popsána funkčnost ve vnitřních prostředích jako jsou kanceláře, tělocvičny, chodby, vestibuly a podobně. Přestože se tato prostředí liší od prostředí budoucího nasazení, spadají do kategorie vnitřních prostor a lze na nich testovat přesnost lokalizace a detailnost tvorby mapy, nebo detekci uzavřených smyček.

7.1.1 Tělocvična

Základní funkčnost byla ověřena v tělocvičně (10 x 10 m), která je dobrým prostředím pro prvotní testování především díky své velikosti, jednoduchému tvaru a modularitě překážek, jako jsou krabice, nebo menší kusy nábytku. Různou konfigurací překážek lze pozorovat, jak dobře robot zapracovává změny prostředí.

V tomto prostředí neměl robot sebemenší problém udržovat kontinuální lokalizaci. Adekvátně reagoval na změny prostředí, které zapracoval do mapy a správně detekoval uzavřené smyčky v místech, kde se již dříve nacházel. Na obrázku níže je vidět vytvořená mapa, červenými šipkami je znázorněná trajektorie robotu a zelené šipky reprezentují detekované uzavřené smyčky.



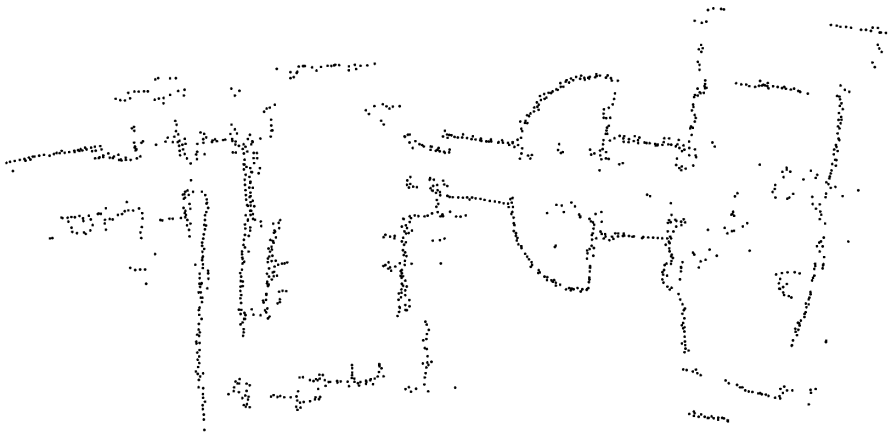
Obrázek 7.1: Mapa tělocvičny, budova C, Karlovo náměstí

7.1.2 Chodby

O poznání hůře je na tom funkčnost v monotónních chodbách a koridorech. V takovýchto prostorách vypadají snímky z laserového dálkoměru téměř stejně, proto často nesprávně zapracuje detekce uzavřených smyček, která má později negativní dopad na tvorbu mapy. V těchto případech mapa neodpovídá skutečnému tvaru prostředí, fragmenty mapy se často začnou nesprávně překrývat a celý proces mapování a lokalizace kolabuje.

7.1.3 Vestibul školy

Členitějším terénem, ve kterém byl robot testován je vestibul školy na Karlově náměstí. V tomto prostředí šlo především o vyzkoušení funkčnosti mapování s dynamickými objekty. Rušnost vestibulu zaplněného pohybujícími se lidmi modeluje velmi rychle měnící se prostředí. Proces mapování si velmi dobře poradil s dynamickými objekty a do mapy je nezahrnul. Ihned po přesunutí objektu se buňka odpovídající staré pozici uvolnila. Na obrázku níže je vidět mapa vestibulu.

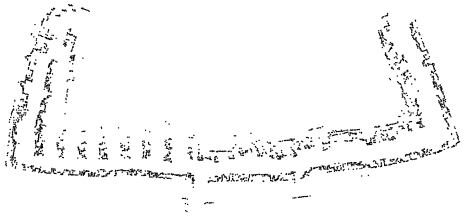


Obrázek 7.2: Mapa vestibulu ČVUT na Karlově náměstí; budova A zprava, skleněná kostka uprostřed, budova B zleva.

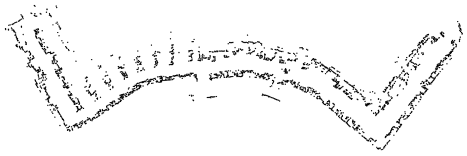
7.2 Experimenty v provozní hale Phoenix

Samostatnou sekci si zaslouží popis experimentů prováděných v provozní hale firmy Phoenix ve Vysokém Mýtě. Vyskytují se zde statické objekty - regály a zdi, polostatické objekty - palety, bedny, krabice i dynamické objekty - vysokozdvižné vozíky, lidé s nízkozdvižnými vozíky. Přirozenost změn provozní haly by se obtížně napodobovaly, a proto jsou data pořízená v takovém prostředí velkým přínosem.

Díky předchozí pracovní zkušenosti znám vnitřní strukturu haly, vytíženost jednotlivých cest, důležitá místa překladu beden, umístění nabíjecích stanic vysokozdvižných vozíků a podobně. Tyto informace byly využity k naplánování co nejvěrohodnějších experimentů připodobněných skutečnému nasazení robotu do provozu. V průběhu měření nastaly modelové situace jako například setkání s člověkem, vyhýbání se vysokozdvižným vozíkům (střed s dynamickými objekty) nebo přirozené změny prostředí (přesunutí palet).

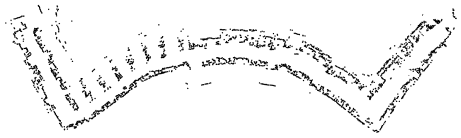


(a) : mapa 1

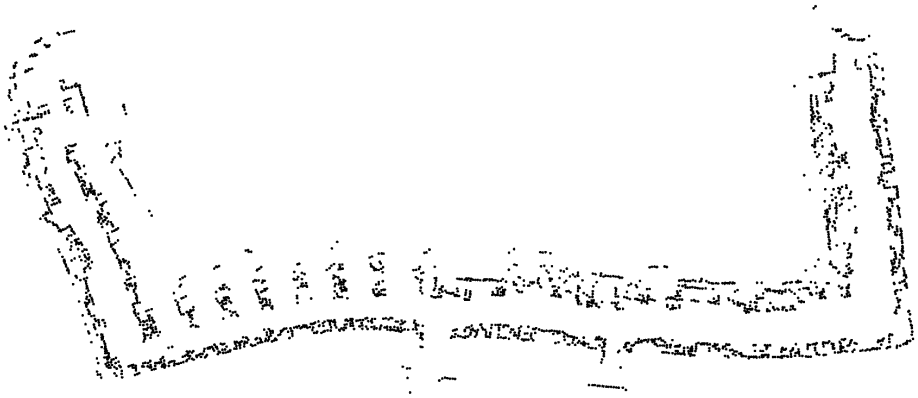


(b) : mapa 2

(c) : mapa 3



(d) : mapa 4



(e) : mapa 5

Obrázek 7.3: Vytvořené mapy z pěti nezávislých měření ve shodné části haly.

Na souboru obrázků 7.3 vidíme výsledné mapy z pěti samostatných měření. Experiment měl ukázat, do jaké míry jsou si mapy z jednotlivých měření podobné. Mapy 7.3b a 7.3d vykazují značné zakřivení. Obě tyto mapy byly pořízeny v opačném směru pohybu než mapy ostatní. Za toto zkreslení může zřejmě rozdílné nahuštění pneumatik, nebo proklouznutí kol. Chyby takového typu však odometrie není schopna zachytit.

Další experiment modeluje situaci, kdy je původní cesta zablokována z důvodu probíhajícího zaskladňování. Lze si představit, že neprůchodnost cesty detekoval sám robot, nebo tento fakt zaznamenal jiný robot a informoval o překážce ostatní. Robot tedy zvolí jinou cestu se stejným startovním i cílovým bodem, jako tomu bylo v předchozím měření. Tuto situaci popisuje následující obrázek.

Obrázek 7.4: Mapa s odlišnou trajektorií a stejnými výchozími body.

Výše popsané experimenty a mnohé další prováděné potvrdily schopnost mapovat dynamické industriální prostředí, v reálném čase zpracovávat data a poskytovat informaci o poloze. Nejslabším článkem procesu SLAM je nekorigovaná chyba odometrie, která způsobuje deformaci mapy patrnou na obrázku 7.3b.

Kapitola 8

Návrh budoucí práce a shrnutí

8.1 Návrh budoucí práce

Budoucí práce spočívá především ve vylepšování metody a odstranění současných nedostatků. V prvé řadě jde o korekci chyby odometrie, která způsobuje deformaci mapy. Zpřesnění odometrie lze řešit pomocí dalších podpůrných senzorů jako je IMU nebo kompas. Například práce [8] dosahuje velice dobrých výsledků lokalizace s využitím IMU. Bylo by také vhodné zvýšit váhu odometrie v situacích, kdy se robot nachází v monotónní chodbě a snímky z laserového dálkoměru jsou příliš podobné. V těchto situacích má větší vliv detekce uzavřených smyček, která zde vyhodnocuje nesprávné smyčky.

Dalším vylepšením a nemalou výzvou je zapracování globální lokalizace, která podchytí špatnou interpretaci dat ze senzorickcých systémů. Pro tyto účely byla již přimontována a zprovozněna kamera s rybím okem mířící do stropu. Dobrým startem budou naměřená data z provozní haly Phoenix, která obsahují databázi obrazů celého prostředí.

V neposlední řadě bude třeba vyřešit kooperaci více robotů nasazených do stejného prostředí. Především půjde o koncept sdílené mapy, ke které bude mít přístup každý robot. Tímto způsobem lze dopředu optimálně plánovat trasu robotu a vyhnout se překážkám ještě dříve, než by na ně konkrétní robot narazil.

V plánu je také zapracovat CAD výkres prostředí do procesu mapování. Na rozdíl od průzkumných misí, kdy je prostředí předem neznámé, je výhodou znalost industriálního prostředí, do kterého budou roboty nasazeny. Do počáteční mapy by se tak daly zasadit informace o zdech a statických objektech.

8.2 Shrnutí

Tato práce se zabývá dlouhodobou lokalizací mobilních robotů. V principu jde o řešení problematiky známé jako SLAM (Simultánní lokalizace a mapování). Mapování probíhá ve 2D za pomoci lidaru jako stěžejního senzoru. Nejprve byla popsána problematika lokalizace a mapování, poté výběr použitých senzorů a jejich princip, dále byl popsán princip metody grafového SLAMu využívající hybridní NDT mapy, byla představena experimentální sestava a potřebný software ke správnému provozu a nakonec byly popsány provedené experimenty.

Jedním z cílů bylo zprovoznit použitý hardware (robot Jackal, laserový dálkoměr Sick TiM361, kamera Pylon) a zajistit spolupráci jednotlivých komponent. I přes projevené komplikace, například v podobě staré verze ROSu v robotu Jackal, nebo zablokování počítače připojením USB 3.0, se podařilo celou experimentální sestavu zprovoznit a otestovat v prostředích, ve kterých se plánuje budoucí nasazení autonomních vozíků.

Dalším cílem bylo najít a zprovoznit, popřípadě naimplementovat metodu řešící SLAM za pomoci transformace normálního rozdělení (NDT). Implementace byla převzata od kolegy Lukáše Jelínka z Matematicko-fyzikální fakulty Univerzity Karlovy [2] a otestována na datech z reálných prostředí. Funkčnost algorimu celkově prokázala schopnost řešit kontinuální lokalizaci a mapování v reálném čase. Současná implementace však selhává v monotónních chodbách a koridorech.

Literatura

[1] E. EINHORN and H. GROSS, "Generic ndt mapping in dynamic environments and its application for lifelong slam," *Elsevier*, 2015. Ilmenau University of Technology, Germany, 28 - 39.

[2] L. JELÍNEK, "Graph-based slam on normal distributions transform occupancy map," 2016. Department of Theoretical Computer Science and Mathematical Logic, Faculty of mathematics and physics, Charles University.

[3] M. KULICH, "Lokalizace a tvorba modelu prostředí v inteligentní robotice," 2003. Czech Technical University in Prague.

[4] A. JELÍNEK, "Vector maps in mobile robotics," 2011. Brno University of Technology.

[5] R. MÁZL, "Lokalizace pro autonomní systémy," 2007. Czech Technical University in Prague.

[6] R. SIEGWART and I. NOURBAHSH, *Introduction to Autonomous Mobile Robots*. 2004. Massachutsetts institute of Technology Cambridge, 197 - 206.

[7] A. LAWRENCE, *Modern inertial technology: Navigation, Guidance, and Control*. 1998.

[8] M. IBRAHIM and O. MOSELHI, "Inertial measurement unit based indoor localization for construction applications," 2016. 13 - 20.

[9] Český normalizační institut, *ČSN EN 1525 Bezpečnost motorových vozíků - vozíky bez řidiče a jejich systémy*, 1998.

[10] M. KAPOŠVÁRY, "Lifelong localization of mobile robot," 2016.

[11] G. D. TIPALDI, D. MEYER-DELIUS, and W. BURGARD, "Lifelong localization in changing environments," 2013. The International Journal of Robotics Research.

[12] M. FRIESL, "Pravděpodobnost a statistika hypertextově [online]. dostupné z:." http://home.zcu.cz/~friesl/hpsb/varvec.html, 2014. 38 -48.

[13] "Tim361-2134101 | detection and ranging solutions | sick. 301 moved permanently [online]. dostupné z:." https://www.sick.com/cz/en/detection-and-ranging-solutions/2d-lidar-sensors/tim3xx/tim361-2134101/p/p369447. cit. 2017-05-22.

[14] "Dsl215 - lenses - optics. framos | industrial imaging and machine vision [online]. dostupné z:." https://www.framos.com/products/en/optics/lenses/dsl215-15406.html. cit. 2017-05-22.

[15] "Jackal ugv - small weatherproof robot - clearpath. clearpath robotics: Autonomous mobile robots [online]. dostupné z:." https://www.clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/. cit. 2017-05-22.

[16] "Ros.org | is ros for me?. ros.org | powering the world's robots [online]. dostupné z:." http://www.ros.org/is-ros-for-me/, 2016-12-20. cit. 2017-05-22.

[17] "Distributions - ros wiki. documentation - ros wiki [online]. dostupné z:." http://wiki.ros.org/Distributions, 2017-04-25. cit. 2017-05-22.

[18] "image_common - ros wiki. documentation - ros wiki [online]. dostupné z:." http://wiki.ros.org/image_common?distro=kinetic, 2013-10-09. cit. 2017-05-22.

[19] "rosbag/commandline - ros wiki. documentation - ros wiki [online]. dostupné z:." http://wiki.ros.org/rosbag/Commandline, 2017-02-19. cit. 2017-05-22.

[20] "sick_tim - ros wiki. documentation - ros wiki [online]. dostupné z:." http://wiki.ros.org/sick_tim, 2016-12-20. cit. 2017-05-22.

[21] "urdf - ros wiki. documentation - ros wiki [online]. dostupné z:." http://wiki.ros.org/urdf, 2014-10-12. cit. 2017-05-22.

[22] "Master - ros wiki. documentation - ros wiki [online]. dostupné z:." http://wiki.ros.org/Master, 2012-02-03. cit. 2017-05-22.

[23] "Setting up jackal's network — jackal tutorials 0.5.1 documentation. clearpath robotics: Autonomous mobile robots [online]. dostupné z." https://www.clearpathrobotics.com/assets/guides/jackal/network.html. cit. 2017-05-22.

[24] "turtlebot_teleop - ros wiki. documentation - ros wiki [online]. dostupné z:." http://wiki.ros.org/turtlebot_teleop, 2015-01-08. cit. 2017-05-22.

[25] "pylon_camera - ros wiki. documentation - ros wiki [online]. dostupné z:." http://wiki.ros.org/pylon_camera, 2016-07-21. cit. 2017-05-22.

[26] "image_transport - ros wiki. documentation - ros wiki [online]. dostupné z:." http://wiki.ros.org/image_transport, 2015-05-06. cit. 2017-05-22.

[27] "Github - lukx19/dynamic_slam. [online]. dostupné z:." https://github.com/Lukx19/dynamic_slam, 2017-01-26. cit. 2017-05-22.

[28] R. B. RUSU and S. COURSINS, "3d is here: Point cloud library (pcl)," 2011. IEEE International Conference on Robotics and Automation (ICRA).

[29] R. KUEMMERLE, G. GRISETTI, H. STRASDAT, K. KONOLIGE, and W. BURGARD, "g^2o: A general framework for graph optimization," 2011. IEEE International Conference on Robotics and Automation (ICRA).

[30] B. JACOB and G. GUENNEBAUND, "Eigen 3.3." http://eigen.tuxfamily.org. cit. 2017-05-22.

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: David N o v á č e k

Studijní program: Kybernetika a robotika (bakalářský)

Obor: Robotika

Název tématu: Celoživotní určování polohy mobilního robotu

Pokyny pro vypracování:

1. Seznamte se s metodami lokalizace a mapování mobilního robotu při trvalém provozu (lifelong SLAM).
2. Implementujte metodu založenou na reprezentaci za pomoci normálního rozdělení (Normal Distribution Transform).
3. Metodu vyzkoušejte na mobilním robotu vybaveném lidarem.
4. Výsledky vyhodnoťte, učiňte závěry a navrhněte zlepšení.

Seznam odborné literatury:

[1] Gian Diego Tipaldi, Daniel Meyer-Delius, Wolfram Burgard: Lifelong localization in changing environments, IJRR 2013
[2] Einhorn, E.; Gross, H.-M.: "Generic 2D/3D SLAM with NDT maps for lifelong application," in Mobile Robots (ECMR), 2013 European Conference on , vol., no., pp.240-247, 25-27 Sept. 2013
[3] Giorgio Grisetti, Cyrill Stachniss, and Wolfram Burgard: Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters, IEEE Transactions on Robotics, vol. 23, no. 1, February 2007

Vedoucí bakalářské práce: Ing. Vladimír Smutný, Ph.D.

Platnost zadání: do konce letního semestru 2017/2018

L.S.

prof. Dr. Ing. Jan Kybic
vedoucí katedry

prof. Ing. Pavel Ripka, CSc.
děkan

V Praze dne 24. 11. 2016

Příloha A

Struktura přiloženého CD

- bagfiles
  - sklenena_kostka_budova_AB_KN.bag - data z vestibulu ČVUT na Karlově náměstí
  - telocvicna_budova_C_KN.bag - data z tělocvičny na Karlově náměstí
- dataSheets
  - datovy_kabel.pdf - technický popis datového kabelu
  - Jackal_brochure_2015.pdf - informační soubor o robotu Jackal
  - Jackal_DataSheet_2014.pdf - technický popis robotu Jackal
  - kamera_dsl215.pdf - technický popis kamery Pylon
  - lidar_sick_TiM361.pdf - technický popis lidaru Sick TiM361
  - napajeci_kabel.pdf - technický popis napájecího kabelu
- src
  - dynamic_slam - balík řešící SLAM pomocí transofrmace normálního rozdělení
  - jackal_simulation-kinetic-devel - balíky pro robot Jackal
  - pylon_camera - balík pro kameru Pylon
  - sick_tim - balík pro lidar Sick TiM361
  - remote-jackal.sh - skript zajišťující spojení s robotem Jackal
  - ...
- BP_novacda1_2017.pdf - elektronická verze této bakalářské práce