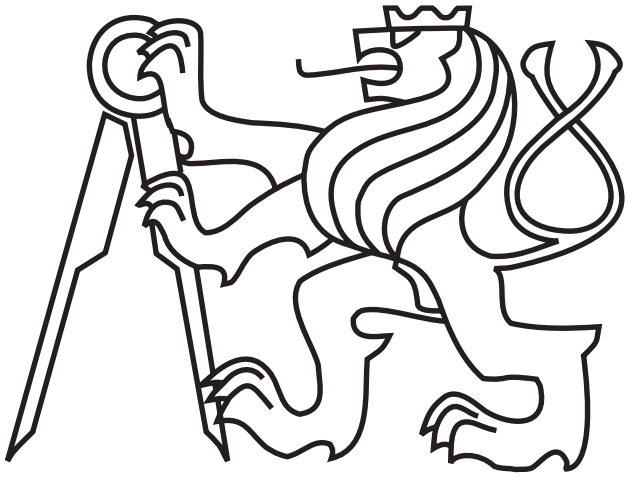
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA ELEKTROTECHNICKÁ

BAKALÁŘSKÁ PRÁCE



Josef Štrunc

Multirobotická explorace prostředí

Katedra kybernetiky
Vedoucí bakalářské práce: **Ing. Jan Faigl**

Praha, 2009

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra kybernetiky

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Student: Josef Štrunc

Studijní program: Elektrotechnika a informatika (bakalářský), strukturovaný

Obor: Kybernetika a měření

Název tématu: Multi-robotická explorace prostředí

Pokyny pro vypracování:

1. Seznamte s úlohou robotické explorace prostředí [1] a souvisejícím problémem lokalizace robotu [2].
2. Seznamte se s robotickou platformou MORBOT a s příslušejícím základním softwarovým vybavením [3].
3. Seznamte se s robotickým prostředím Player/Stage [4,5].
4. Seznamte s úlohou multi-robotické explorace prostředí[6].
5. Navrhněte a implementujte algoritmus řešící úlohu robotické explorace.
6. Úlohu rozšiřte o uvažování více mobilních robotů.
7. Algoritmus experimentálně ověřte v simulátoru a na reálném robotu/robotech MORBOT.

Seznam odborné literatury:

[1] Yamauchi, B.: Frontier-Based Exploration Using Multiple Robots. In AGENTS '98: Proceedings of the Second International Conference on Autonomous Agents, pages 47-53, Minneapolis, Minnesota, United States, 1998. ACM Press.
[2] Thrun, S.; Burgard, W. and Fox, D.: Probabilistic Robotics (Intelligent Robotics and Autonomous Agents). TheMIT Press, September 2005.
[3] Grimmer, V.: Bakalářská práce - Platforma pro výuku mobilní robotiky. FEL ČVUT v Praze, 2008.
[4] http://playerstage.sf.net
[5] Szücsová, H.: Bakalářská práce - Podpůrný software pro výuku mobilní robotiky. FEL ČVUT v Praze, 2008.
[6] Burgard, W.; Moors, M.; Fox, D.; Simmons, R. and Thrun, SS.: Collaborative Multi-Robot Exploration. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 2000.

Vedoucí bakalářské práce: Ing. Jan Faigl

Platnost zadání: do konce zimního semestru 2009/2010

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE Fakulta elektrotechnická L.S.

prof. Ing. Vladimír Mařík, DrSc.
vedoucí katedry

doc. Ing. Boris Šimák, CSc.
děkan

V Praze dne 10.12.2008

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, SW, projekty atd.) uvedené v přiloženém seznamu.

V Praze dne ....

podpis

Poděkování

Především bych rád poděkoval vedoucímu mé bakalářské práce Ing. Janu Faiglovi za jeho trpělivost a čas, který věnoval kontrolám této práce. Dále bych chtěl poděkovat členům skupiny Inteligentní a Mobilní Robotiky za rady a za pomoc s přípravou experimentu s reálnými roboty.

Abstrakt

Tato bakalářská práce se zabývá problémem explorace prostředí jedním a skupinou mobilních robotů a tvorbou 2D mapy prohledávaného prostředí. Práce navazuje na systém ARIEL popsaný v článcích [10] a [9] a rozšiřuje jej o aktualizaci mapy v reálném čase a častější přeplánování cílových frontierů a cesty. Multirobotická explorace rozšiřuje původní systém o aspekty koordinace a kooperace více robotů, které nebyly původními autory zohledňovány. V práci je řešena tvorba dvojrozměrné mapy na mřížce, integrace senzorických dat do mapy, plánování cesty na mřížce, určování míst pro exploraci a také problémy související s použitím více robotů, jako je vzájemné vyhýbání a jejich kooperace při exploraci. Součástí práce jsou výsledky experimentů v simulovaném prostředí, které ukazují vliv počtu robotů a vliv kooperace na rychlost explorace. Součástí práce jsou také výsledky experimentu se dvěma reálnými roboty, který ukazuje, že je možné systém použít pro exploraci reálných statických prostředí uvnitř budov.

Abstract

The thesis deals with problems of single robot and multirobotic exploration and simultaneous map building of an unknown environment. The work is inspired by the ARIEL system, which is described in the articles [10] and [9]. In addition, it is extended to cope with real-time map updating and more frequent replanning of target frontiers and paths. The multirobotic exploration deals with coordination and cooperation of robots, especially aspect which have not been addressed by authors of the ARIEL. The thesis solves following problems: building of 2D grid map, integration of sensorial data into the map, path planing on the grid, determination of places for exploration and also problems related to multirobotic exploration such as mutual path avoidance and cooperation. Experimental results of the simulated environment are presented and discussed. The results show the effect of number of robots and the effect of cooperation to performance of the exploration task. Experimental result of the experiment with two real robots are presented as well. It shows, that developed system can be used for mapping of real static indoor environments.

Obsah

Seznam tabulek

Seznam obrázků

Úvod

Cílem této práce je implementovat exploraci a tvorbu 2D mapy prostředí skupinou mobilních robotů. Explorací se rozumí proces, při kterém je prohledáváno neznámé prostředí a je vytvářen jeho model. Model prostředí je možné použít pro dosažení určitého stupně autonomnosti mobilního robotu, který tak se tak může pohybovat a vykonávat různé akce v proměnném prostředí. Model prostředí je také jedním ze základních prostředků jak automatizovat proces uvažovaní o prostředí a následné adaptaci, která vede na takové chování mobilního robotu, které by lidé označili za inteligentní. V úloze explorace lze také využít několik spolupracujících robotů a zvýšit tak rychlost a robustnost řešení této úlohy.

Součástí řešení úlohy explorace je nezbytné vyřešit několik základních podúloh - způsob reprezentace prostředí (mapy), lokalizaci robotu v prostředí, plánování cesty a vlastní řízení pohybu mobilního robotu. Tato práce se zabývá exploraci prostředí autonomním mobilním robotem a skupinou robotů. Při řešení úlohy více mobilními roboty je studován vliv počtu robotů a vliv mechanismu kooperace na celkový průběh a rychlost explorace celého prostředí.

Práce je strukturována do následujících kapitol. V první kapitole je nejdříve popsán úvod do problematiky robotického mapování a tvorby modelu prostředí spolu s uvedením motivace řešení těchto problémů - příklady možných aplikací. Dále zde jsou uvedeny příklady existujících řešení robotické explorace jedním i více roboty, kterými byla práce inspirována. Hlavní část bakalářské práce je věnována popisu vlastního řešení úlohy robotické explorace založené na navigaci k takzvaným frontierům (hraničním oblastem mezi známým a dosud neprozkoumaným prostore) a řešením dílčích úloh potřebných pro robotickou exploraci. V této části je popsána tvorba mapy, model senzoru a způsob integrace senzorických dat do mapy. Součásti popisu jsou i nezbytné algoritmy, mezi které patří nafukování překážek, plánování cesty na vytvořené mapě, vyhledávání míst vhodných pro prohledání a navigace k nim. Poté je popsáno rozšíření explorace jediným robotem na multirobotickou exploraci a řešení souvisejících problémů jako vzájemné vyhýbání robotů, zabránění vzájemnému mapování robotů jako překážek a kooperace robotů. Algoritmy řešící celý studovaný problém jsou implementovány v jazyce C++ s využitím knihoven systému Player [1]. V práci je uvedena základní architektura implementované řídicí aplikace. Experimentální ověření bylo provedeno v robotickém simulačním prostředí Stage a následně ověřeno v reálném prostředí. Výsledky experimentů jsou uvedeny v kapitole 3. V závěru jsou shrnuty dosažené cíle a možnosti rozšíření.

Kapitola 1

Přehled problematiky

Tvorba map nebo modelů prostředí je jedním ze základních problémů mobilní robotiky. Má-li se mobilní robot autonomně a bezpečně pohybovat v prostředí, je vhodné, aby se dokázal vyhýbat překážkám. K efektivnímu vyhýbání se překážkám je nutná znalost rozmístění překážek v jeho okolí. Jednou z možností je tuto znalost předat robotu formou apriorně známé mapy prostředí. Například lze změřit mapu prostředí manuálně, což však bývá při požadované přesnosti velmi obtížné a časově náročné. Pro pohyb v budovách se nabízí využití například architektonických plánů budov nebo jiných předem vytvořených map. Ty však nemusí být vždy k dispozici nebo jsou nepoužitelné kvůli jejich nespolehlivosti a nepřesnosti, nemusí v nich být například zaznamenané různé přemístitelné objekty, například nábytek. Dalším problémem je, že se roboty mohou pohybovat v prostředí, které podléhá různě rychlým změnám, a jeho mapu je třeba často aktualizovat. Proto je nejlepší možností tento proces automatizovat a nechat robot samotný tvořit a opravovat model prostředí s využitím jeho senzorů.

V této kapitole jsou uvedeny základní přístupy při řešení robotické explorace a několik motivační příklad využití mobilních robotů v praktických aplikací. V oddíle 1.1 jsou popsány modely prostředí používané v mobilní robotice, oddíl 1.2 uvádí několik používaných metod mapování a problémy, které s mapováním souvisí. V oddílech 1.3 a 1.4 je popsána úloha explorace prostředí jedním respektive skupinou mobilních robotů a uvedeny příklady existujících implementací těchto úloh, kterými je inspirováno vlastní řešení této práce popisované v následující kapitole 2. V závěru kapitoly je jako motivace k řešení robotické explorace uvedeno několik oblastí jejího možného využití.

1.1 Modely prostředí

Tvorba mapy vyžaduje řešení několika podproblémů - reprezentace modelu prostředí, integrace senzorických dat, lokalizace robotu v prostoru, plánování trajektorie pohybu. Způsob reprezentace modelu prostředí závisí na typu prostředí a na úloze, kterou má robot vykonávat. Například pro pohyb robotu ve venkovním terénu mimo cesty není vhodné používat dvojrozměrnou planární mapu. Model prostředí by také měl respektovat nepřes-

nost senzorických dat i nepřesnost určení polohy robotu. Podle stupně abstrakce lze mapy prostředí rozdělit na (uvedeno v [4]):

- **senzorické** - obsahují pouze data ze senzorů,
- **geometrické** - mapu tvoří 2D nebo 3D geometrické objekty vypočtené ze senzorických dat,
- **mapy obsahující lokální vztahy** - obsahují i informace o funkčních, strukturálních nebo sémantických relacích mezi geometrickými objekty, které spolu sousedí,
- **topologické** - obsahují informace o vzájemných vztazích mezi objekty a lokacemi v měřítku celé mapy,
- **sémantické** - obsahují funkční označení prvků mapy.

Jedním z prvních řešených problémů v mobilní robotice je mapování prostředí uvnitř budov. K tomu lze využít dvojrozměrné mřížky obsazenosti a pravděpodobnostní mřížky, ve kterých je dvojrozměrný prostor rozdělen na jednotlivé buňky, které obsahují informaci o obsazenosti, respektive o míře pravděpodobnosti obsazení buňky překážkou. Vzhledem k tomu, že prostředí v budovách je vysoce strukturované a skládá se převážně ze vzájemně kolmých rovin, lze jej s výhodou reprezentovat soustavou úseček. To sice vyžaduje více výpočtů nad senzorickými daty, ale snižuje paměťové nároky pro uložení výsledné mapy. K práci s neurčitostí jsou používány Kalmanovy filtry. S přesunem k mapování vnějších přirozených prostředí se začaly používat 2,5-D mřížky, kde jednotlivé buňky obsahují navíc informaci o vertikální poloze povrchu. K mapování trojrozměrného světa se dále používají shluky bodů, 3-D mřížky a 3-D sítě. Ve 3-D mapách je třeba zpracovávat mnohem větší objemy dat, proto je zde potřeba upravovat senzorická data do jednodušších objektů. Většina používaných modelů předpokládá statické prostředí, pro uvažování zahrnující pohybující se objekty je třeba předchozí modely rozšířit o časový rozměr.

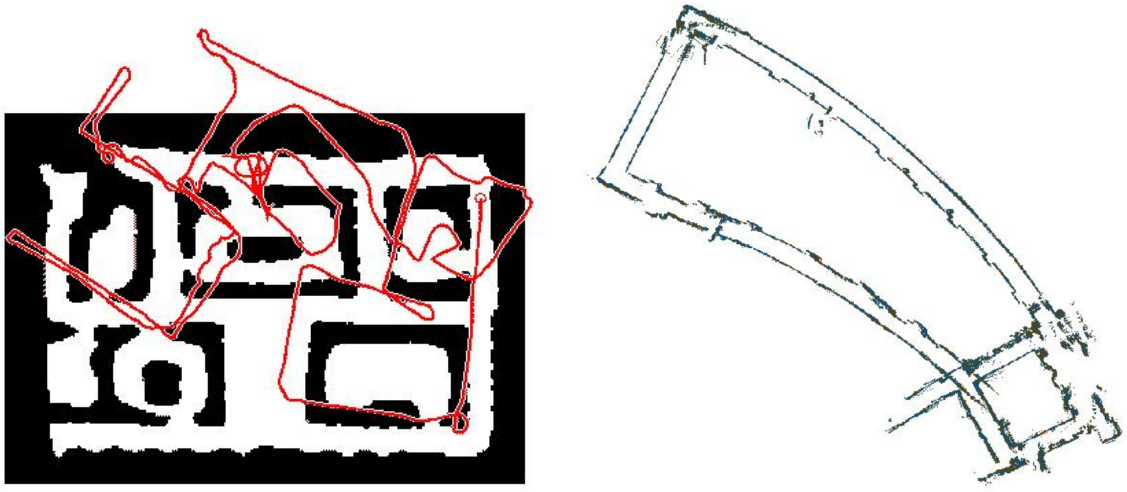
1.2 Mapování prostředí

V této oblasti probíhal v posledních třiceti letech velmi aktivní výzkum. Jak je uvedeno v [6] existují již robustní metody pro mapování prostředí, která jsou statická (neměnná v čase), strukturovaná a nejsou příliš rozlehlá. Jedním z prvních řešení byl Elfesův a Moravcův algoritmus mapování využívající mřížku obsazenosti. Tento přístup byl poté využit v mnoha dalších robotických systémech. Metrické mapy byly uvedeny Chatilou a Laumondem, kteří využívali mnohoúhelníky k popisu geometrie prostředí. Topologické mapy využívali při mapování Mataric, Kuipers a další. Mezi metrickými mapami a topologickými mapami není jasná hranice, protože téměř všechny přístupy stavějí na geometrických informacích. V praxi bývají metrické mapy podrobnější než topologické, což sice vyžaduje více výpočetních prostředků, ale umožňuje řešit některé další problémy.

Od devadesátých let začaly v mapování převažovat pravděpodobnostní techniky. Smith, Self a Cheesman uvedli ve svých článcích mocnou statistickou metodu pro současné řešení mapování a s ním souvisejícího problému lokalizace robotu ve vytvářené mapě. Od té doby je mapování označováno jako SLAM (Simultaneous Localization And Mapping) nebo CLM (Concurrent Mapping and Localization). Pravděpodobnostní přístupy se dají rozdělit na tři skupiny. První z nich využívá Kalmanovy filtry k odhadu mapy a pozice robotu. Druhá, tzv. Dempster expectation maximization algorithm, řeší zda data nasnímaná senzory na různých pozicích odpovídají stejnému fyzikálnímu objektu v reálném světě. Třetí skupina pravděpodobnostních technik hledá objekty, které mohou odpovídat například zdem, dveřím, nábytku nebo pohybujícím se objektům.

1.2.1 Problémy při tvorbě mapy

Při tvorbě mapy bývají používány kamery, laserové nebo infračervené dálkoměry, sonary, radary, doteková čidla, kompasy a systém GPS. Všechny tyto senzory jsou však zdrojem chyb a šumu. Další společnou vlastností je omezený dosah senzorů a jejich neschopnost „vidět" za překážky, proto se musí robot při tvorbě mapy prostředím pohybovat. V okamžicích snímání dat je třeba znát polohu senzorů. Tu můžeme zjistit z polohy robotu a odometrie, která je však také zatížena chybami, protože je například špatně změřen obvod nebo vzdálenost kol robotu nebo mohou kola prokluzovat atd. Na rozdíl od nepřesnosti senzorů, která je stále stejná, chyba v určení polohy se postupem času zvětšuje, jak je vidět na obrázku 1.1(a), což je hlavním problémem při mapování. Mnoho existujících mapovacích algoritmů je proto velmi složitých, jak po matematické, tak po implementační stránce.



Obrázek 1.1: Příklady chyb při mapování, a) špatně změřená trajektorie způsobená chybou odometrie, která se při pohybu postupně akumulovala b) příklad Correspondence Problem: nesprávně vytvořená mapa kruhového prostředí. převzato z [6].

Další komplikací je vysoký počet stupňů volnosti prostředí, neboli počet čísel potřebných pro popis tohoto prostředí. Možná nejtěžší problém, spočívající v určení toho, jak odpovídají senzorická měření provedená v různých časech reálným fyzikálním objektům, je nazýván Correspondence Problem. Při špatném řešení může mapování uzavřené kruhové chodby dopadnout jako na obrázku 1.1(b), kde je kvůli chybě odometrie chodba „narovnána" a výchozí bod je po uzavření cyklu namapován jako dvě různá místa.

Obtížným úkolem při mapování je volba cesty robotu. Problémy jsou způsobeny neúplností známého prostředí v okamžiku plánování pohybu. Navíc, aby se využilo maximální množství senzorických dat, musí algoritmus tvorby mapy běžet v reálném čase, což je důležité omezení pro mnoho existujících přístupů.

1.3 Robotická explorace

Robotická explorace je úloha, při které je hlavním úkolem prohledat zadanou oblast prostředí a vytvořit jeho mapu. Rozšiřuje úlohu mapování prostředí o navigaci robotu tak, aby vytvořil mapu prostředí co nejrychleji. Případnými dalšími požadavky může být co nejkratší ujetá vzdálenost, volba cesty co nejdále od překážek, vyhýbání se určitým objektům a podobně. Podstatou problému je rozhodování kam poslat robot, aby se minimalizoval čas potřebný pro úplné prohledání prostředí.

1.3.1 Frontier-Based Exploration

Příkladem implementace robotické explorace je systém ARIEL (Autonomous Robot for Integrated Exploration and Localization) popsaný v článku [10]. Systém řeší problém tvorby mapy, která je reprezentována dvojrozměrnou mřížkou, a současnou lokalizaci robotu v této mapě pomocí přístupu nazývaného Frontier-Based Exploration. Hlavní myšlenka tohoto přístupu je následující: *Pro získání nejvíce nových informací o světě pošli robot na hranici mezi prázdným a neznámým prostorem.* Tyto oblasti jsou označovány jako tzv. frontiery a jsou definovány jako skupiny buněk, které se nacházejí na rozhraní prohledaného prázdného prostoru a neznámého prostoru, toto označení bude používáno dále v textu.

Hlavními vlastnostmi frontieru jsou jeho poloha a velikost. Vliv velikosti frontieru na rychlost explorace není zřejmý a je třeba jej určit experimentálně. Pokud se robot pohybuje k frontierům, může vidět dříve neznámé prostředí a přidat novou informaci do mapy. Jako výsledek se hranice mezi známým a neznámým prostorem posunuje a zmapovaná plocha se zvětšuje. Postupným odkrýváním neznámého prostoru frontiery zanikají. Pokud v mapě neexistuje žádný frontier, prostředí je celé zmapované.

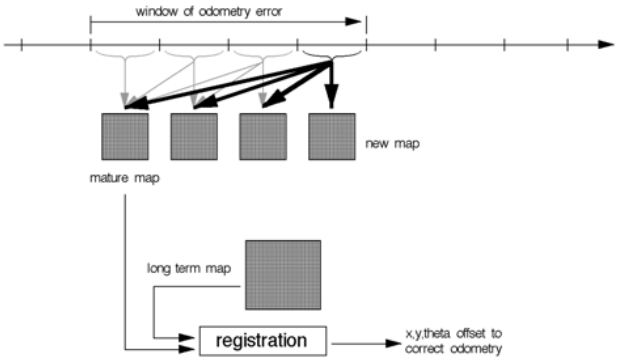
Pro reprezentaci mapy je použita dvojrozměrná pravděpodobnostní mřížka, která rozděluje prostor na kartézský systém čtvercových oblastí. Čtverce, označované jako buňky, odpovídají částem plochy v rovině podlahy s danými rozměry. Každá buňka je charakterizována reálným číslem, které představuje pravděpodobnost, že obsahuje překážku. Výhodou pravděpodobností mřížky je možnost integrovat do ní data z různých senzorů a také to,

že neklade žádné požadavky na strukturu prostředí - překážky mohou mít libovolný tvar a orientaci.

Senzory použité v systému ARIEL jsou ultrazvukové dálkoměry a laserový dálkoměr. Autoři vyvinuli techniku nazývanou laser-limited sonar, díky níž jsou potlačovány spekulární (zrcadlové) odrazy ultrazvuku. V této technice se vezme jako správná menší vzdálenost naměřená oběma senzory. Pokud vrací sonar větší vzdálenost než laserový dálkoměr, pravděpodobně došlo k odrazu ultrazvukového laloku a jako správná se vezme hodnota z laserového dálkoměru. Výhodou sonaru je, že jeho zorné pole není omezeno pouze na rovinu jako u laserového dálkoměru, který používá laserový paprsek rozmítaný pouze v jedné rovině.

Algoritmus pro detekci frontierů nejprve podle pravděpodobnosti obsazenosti označí každou buňku mapy jako prázdnou, neznámou nebo obsazenou. Poté jsou neznámé buňky, které sousedí s prázdnou buňkou označeny jako část frontieru. Sousední buňky splňující vlastnosti frontieru jsou poté slučovány do větších skupin a pokud skupina obsahuje více buněk než je určité nastavené minimum, je označena jako frontier.

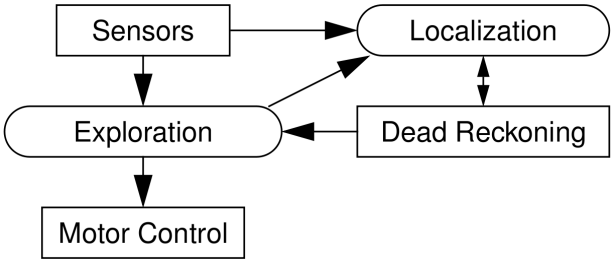
Robot je poté vždy navigován do středu nejbližšího z nalezených frontierů. Plánovač cesty využívá algoritmus *depth-first search* pro nalezení nejkratší bezkolizní cesty z pozice robotu k cílovému frontieru. Při pohybu robotu zabraňuje reaktivní systém kolizím s překážkami, které nebyly zaznamenány do mapy. Pokud robot dorazí na určené místo, provede senzorické měření, aktualizuje mapu, vyhledá v ní frontiery a naplánuje si cestu k dalšímu nejbližšímu frontieru. Tvorba mapy tedy neprobíhá v reálném čase, ale vždy až po dojezdu k cílovému frontieru.



Obrázek 1.2: Princip kontinuální lokalizace používaný systémem ARIEL, převzato z [10].

Systém používá kontinuální lokalizaci robotu, korekce polohy robotu jsou prováděny cyklicky s krátkou periodou. Díky tomu je chyba polohy malá a lze použít rychlé korekční techniky. Na obrázku 1.2 je diagram lokalizační techniky, kterou používá systém ARIEL. Lokalizační algoritmus generuje v krátkém čase ze senzorických dat malé lokální mapy a několik jich udržuje v paměti. Lokální mapa, která je nejdéle v paměti a je v ní tedy integrováno nejvíce senzorických měření, je použita k procesu označovanému jako registrace. Proces registrace zahrnuje prohledávání v prostoru translací a rotací lokální mapy,

které minimalizuje chybu v porovnání krátkodobé mapy s dlouhodobou. Očekávaná chyba odometrie za krátký čas je malá, proto se translace omezuje na ± 15 cm a rotace na ± 2 stupně. Proto je možné proces registrace provádět rychle. Při procesu registrace je zjištěna délka posunutí a úhel otočení, při kterých krátkodobá mapa nejlépe odpovídá dlouhodobé a o tyto hodnoty je opravena poloha robotu. Tento systém lokalizace selhává v případě, že se robot pohybuje v rozlehlém prázdném prostoru, kde nelze zachytit žádný vzor v prostředí, který by šel porovnat s dlouhodobou mapou. Obrázek 1.3 shrnuje architekturu systému ARIEL.



Obrázek 1.3: Architektura systému ARIEL, převzato z [10].

1.4 Multirobotická explorace

V některých aplikacích využívajících exploraci je kritickým parametrem čas, za který je třeba prohledat dané prostředí. Kromě možnosti vylepšování plánování pohybu jednoho robotu, popřípadě vývoje nového robotu s lepšími fyzickými vlastnostmi je výhodné rozdělit tuto úlohu mezi více robotů, čímž je možné dosáhnout podstatného urychlení. Další velkou výhodou je vyšší robustnost provádění úkolu. V případě, že se některý z robotů porouchá, je vyšší pravděpodobnost, že ostatní roboty úlohu dokončí. Při použití více robotů dochází často k překrývání senzorických dat, což přispívá ke snižování neurčitosti ve vytvářené mapě. Motivací pro vývoj systému s více roboty může být také vylepšení metody lokalizace. Větší přesnosti při lokalizaci lze dosáhnout například tak, že zatímco se jeden robot pohybuje, jiný stojí na místě a slouží jako referenční bod, podle kterého si pohybující se robot opravuje polohu získanou z odometrie.

Řešení explorace více roboty však přináší některé komplikace. Je nutné zajistit, aby nedocházelo k vzájemným kolizím robotů, což zvyšuje nároky na plánování cesty. Při spoléhání pouze na jednoduchý algoritmus zabránění kolizi (*collision avoidance*), kdy se robot zastaví a čeká, jakmile naměří překážku příliš blízko, se mohou roboty vzájemně zablokovat. Další možná komplikace nastává, pokud roboty používají stejné senzory, může docházet k přeslechům mezi senzory dvou různých robotů.

Dále je třeba rozhodnout jakou použít architekturu řídicího programu. Jednou z možností je udržovat jednu globální mapu a do ní integrovat data od jednotlivých robotů

a roboty navigovat po mapě centrálním plánovacím algoritmem. Jiným přístupem je distribuovaná architektura, ve které každý robot plánuje svůj pohyb samostatně a vytváří si lokální mapu, která se poté slučuje do společné globální. Důležité je také řešení komunikace robotů, to znamená, jak si roboty budou při exploraci předávat naměřené informace. Výhodné je používání bezdrátového přenosu dat, který však může být omezen svým dosahem, což se někdy řeší například plánováním setkání robotů v určitých místech, kde si mohou předat naměřené informace.

Hlavním problémem multirobotické explorace je navrhnout plánovací algoritmus, který bude distribuovat roboty na různá místa mapy. Aby úloha měla smysl, je potřeba, aby plánovací algoritmus urychlil prohledání mapy v porovnání s nasazením stejného počtu nekooperujících robotů. V některých případech nemusí použití více robotů přinést očekávané zlepšení výkonu. V důsledku vzájemného objíždění může například dojít k prodloužení času explorace.

1.4.1 Frontier-Based Exploration Using Multiple Robots

Příkladem implementace multirobotické explorace je systém rozšiřující systém ARIEL, který je popsaný v předchozí části. Metoda byla publikována v článku [9].

Systém funguje stejně jako při exploraci jedním robotem, avšak lokální mapy vytvořené po příjezdu k frontieru jsou v okamžiku jejich vzniku jednak integrovány do globální mapy robotu a jednak posílány ostatním robotům. Každý robot si ukládá lokální mapy od ostatních robotů do okamžiku, kdy přijede k frontieru a pak je všechny integruje, spolu se svojí právě naměřenou lokální mapou do své globální mapy. Po integraci všech lokálních map je vybrán nový frontier, ke kterému je robot navigován. Výhodou tohoto přístupu je, že je kooperativní a decentralizovaný. Informace získaná jedním robotem je k dispozici všem ostatním robotům a ty se na jejím základě mohou rozhodovat kam se navigovat. Zatímco informace je sdílená, řízení každého robotu je nezávislé, a proto celý systém není tolik závislý na selháních jednotlivých robotů.

Tento systém však neřeší problém navigace několika robotů ke stejnému frontieru ani vyhýbání se robotů. Proto se může stát, že se roboty vzájemně zablokují a označí své cílové frontiery jako nedostupné, ostatní nezablokované roboty k nim však stále mohou dojet. V tomto přístupu také není řešeno vzájemné mapování robotů jako překážek.

1.5 Aplikace

V tomto oddílu je uvedeno několik příkladů aplikací, při kterých je využíván model prostředí získaný mapováním, nebo ve kterých jde o mapování samotné, a ve kterých musí robot dosahovat určitého stupně autonomnosti. Mobilní roboty mají využití především v aplikacích, při nichž se pracuje v nebezpečných prostředích nebo prostředích příliš vzdálených, kam je obtížné vyslat člověka. Dále je lze využít v úlohách, které jsou pro člověka nepříjemné nebo které mají dlouhý a náročný pracovní cyklus. Příklady jsou čerpány zejména z knihy [4].

Transport materiálu

Mnoho průmyslových, komerčních a lékařských provozů vyžaduje převoz předmětů a materiálu. Využití mobilních robotů, kteří jsou schopni autonomního pohybu, je vhodné v případech, kdy je prostředí proměnné a obsahuje mnoho překážek a cestu nelze vyznačit napevno dopředu. Například v nemocniční praxi už je nasazeno několik mobilních robotů, které pomáhají rozvážet jídlo, léky a dokumenty. Dalším příkladem je evropský projekt MARTHA (Multiple Autonomous Robots for Transport and Handling Applications), jehož cílem je vyvinout skupinu autonomních robotů schopných koordinovaného transportu předmětů a materiálu. Tento systém by našel uplatnění v přístavech, na letištích a ve skladech.

Zemědělství a lesnictví

Autonomní mobilní roboty mohou nalézt uplatnění také při údržbě lesa (prořezávání vysázených stromů) i při kácení stromů a svážení dřeva. Tento úkol je velmi složitý kvůli komplexnímu terénu, který je obtížné namapovat a vytvořit jeho model, a v němž je pro robot těžké se pohybovat. Za tímto účelem jsou vyvíjeny kráčející roboty, se kterými experimentuje například Kanadský národní lesnický institut. V zemědělství mohou být roboty využívány na polích při orbě, sklízení úrody a aplikaci postřiků nebezpečných pro člověka. Sklízení zeleniny nebo ovoce v sadech je velmi náročné na lidskou práci a jsou při něm kladeny požadavky na rychlost sklízení a opatrnou manipulaci s plodinami. Obzvláště náročné je například česání ovoce z vysokých stromů. Proto je tato oblast vhodná pro použití mobilních robotů, ale zároveň klade vysoké nároky na jejich schopnosti.

Úklidové roboty

Rutinní prací, pro kterou se již dnes využívají autonomní mobilní roboty, je úklid, především umývání nebo vysávání podlah. Toho se využívá v rozlehlých továrnách nebo ve veřejných prostorách v budovách (například v supermarketech), a také v domácnostech. Na současném trhu jsou dnes k dostání automatické robotické vysavače, například iRobot Roomba nebo Electrolux Trilobite.

Inteligentní vozidla

V této oblasti existuje několik přístupů, jak využívat techniky mobilní robotiky:

- Asistenty pro řízení automobilu - tyto systémy se používají již v současnosti, slouží k zobrazování dodatečných informací řidiči vozidla. Používají se například pro upozorňování na dopravní značky, na udržování bezpečné vzdálenosti mezi vozidly, doporučování optimální okamžité rychlosti nebo parkovací asistenty, které jsou už dnes schopny díky senzorickému mapování okolí úplně řídit směr při parkování (rychlost řídí člověk).

- Konvojové systémy - jsou použitelné při silniční nákladní dopravě, první vozidlo je řízené člověkem a ostatní automaticky řízená vozidla jej sledují.
- Autonomní vozidla - vozidla schopná samostatného pohybu v terénu.

Roboty pro průzkum

Příkladem aplikace, kdy je třeba rychle prohledat neznámé a nebezpečné prostředí, je úloha nazývaná Search and Rescue, ve které jde o rychlé prohledání například domu při požáru nebo po zemětřesení mobilními roboty a nalezení zraněných osob. Dalším využitím může být mapování nebezpečného prostředí při policejních zásazích nebo vojenských akcích.

Vesmírná robotika

Vesmírné prostředí je ideální pro uplatnění mobilních robotů. Na cizí vesmírná tělesa je velmi drahé a nebezpečné vyslat člověka. Motivací pro využití autonomních robotů je skutečnost, že robot nelze teleoperovat v reálném čase kvůli dlouhé době letu signálu. Mobilní roboty lze využít při průzkumu planet nebo jiných těles, případně pro složitější úkony jako je stavba a udržování trvalých stanic.

Kapitola 2

Explorace

V této kapitole je popsáno řešení úlohy explorace prostředí. Nejprve je popsáno řešení explorace jedním robotem, které následně rozšířeno na úlohu explorace týmem více robotů. Řešení je inspirováno systémem popsaným v části 1.3.1, který je založen na reprezentaci prostředí pravděpodobnostní mřížkou obsazenosti. Systém je navíc vylepšen o aktualizaci mapy v reálném čase a o častější vyhledávání frontierů a přeplánování cesty. Tím je docíleno toho, že robot/y mohou rychleji reagovat na změny v postupně vytvářené mapě.

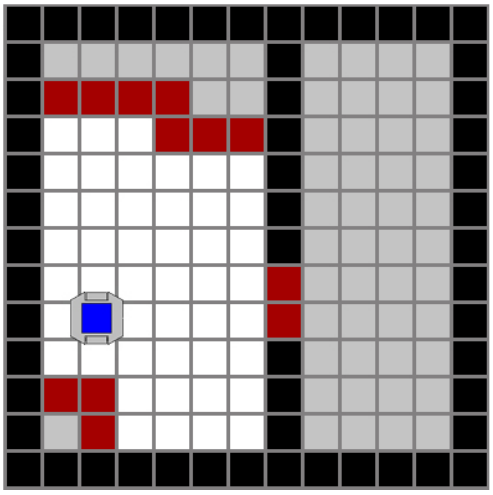
Nedílnou součástí problému mapování je lokalizace robotů v prostředí. Řešení tohoto problému je však mimo rámec této práce. Z tohoto důvodu jsou v práci uvažovány lokalizované roboty.

Dále je v algoritmu explorace prostředí uvažován diskový robot s diferenciálním podvozkem a laserový dálkoměr jako hlavní zdroj informací o okolním prostředí robotu.

2.1 Explorace prostředí jedním robotem

Hlavním úkolem při exploraci je navigovat robot tak, aby postupně svými senzory prozkoumal celý prostor zadaného prostředí. Jednou z možností jak toho dosáhnout je navigace robotu k nejbližším frontierům (jejich definice byla uvedena v oddíle 1.3.1), která je popsána v článku [9]. Tím, že robot k těmto místům dojede je zajištěno změření dosud nezmapovaného prostoru. Určit optimální posloupnost frontierů, ke kterým má být robot postupně navigován, aby bylo prostředí prohledáno co nejrychleji, je velmi obtížné, a to především z následujících důvodů. Například není vhodné řídit se podle velikosti frontieru, protože malý frontier může představovat průchod do rozlehlé místnosti a za velkým frontierem může být zeď. Příklad takové situace je ilustrován na obrázku 2.1, kde černé buňky představující překážky v prostředí nejsou ještě všechny robotem objeveny (jsou zobrazeny pouze pro představu o celém prostředí).

Na druhé straně, malé frontiery bývají například také často v rozích místností. Při průjezdu robotu kolem nich by k jejich úplnému prozkoumání stačilo ujet pouze malou vzdálenost navíc. Pokud explorační strategie preferuje velké frontiery musí se robot k takovýmto místům zdlouhavě vracet.



Obrázek 2.1: Výběr frontieru - černé buňky představují překážky, šedé neznámý prostor a červené frontiery.

Postup explorace prostředí jedním robotem shrnuje algoritmus 1.

Algoritmus 1: Explorace prostředí jedním robotem

```
1  begin
2      otočení robotu o 180° při současné integraci laserových dat do mapy
3      vytvoření navigační mřížky a „nafouknutí" překážek
4      nalezení frontierů while seznam frontierů není prázdný do
6          //přeplánování na aktuální mapě
7          vytvoření navigační mřížky a „nafouknutí" překážek
8          nalezení frontierů
9          naplánování cesty k nejbližšímu frontieru a odebrání frontieru ze seznamu
10         výběr nejbližšího bodu cesty a navigace robotu k němu
12         while robot není v cílovém bodě nebo neujel vzdálenost větší než nastavený
           práh do
13             výpočet a nastavení dopředné a úhlové rychlosti
14             integrace aktuálních dat z laseru do mřížky s mapou
15         end
16     end
17 end
```

Uvedené základní schema explorace se skládá z následujících dílčích úloh:

- způsob reprezentace mapy a integrace nových dat,
- plánování cesty v mapě,
- vyhledávání a úprava frontierů a

- navigace robotu k frontieru.

Navržená řešení těchto dílčích problémů jsou popsána v následujících oddílech.

2.2 Tvorba Mapy

Mapa je reprezentována dvojrozměrnou pravděpodobnostní mřížkou obsazenosti, která má tři parametry - velikost buněk a počet buněk v horizontálním a vertikálním směru. Tyto parametry je třeba volit s ohledem na předpokládanou velikost prohledávaného prostoru. Na mřížce je uvažováno osmi okolí. Při prohledávání jedním robotem a při žádných počátečních znalostech prostředí je počáteční poloha robotu nastavena do středu mřížky. Každá buňka je charakterizována reálným číslem v intervalu $\langle 0, 1\rangle$, které představuje pravděpodobnost obsazení buňky překážkou. Při spuštění úlohy je pravděpodobnost obsazení každé buňky neznámá, proto je nastavena na výchozí hodnotu 0,5. Přidání nového měření do mapy je provedeno dle Bayesova vzorce

$$P(a \mid r) = \frac{p(r \mid a)P(a)}{p(r \mid a)P(a) + \neg p(r \mid a)\neg P(a)},$$

kde $P(a \mid r)$ je pravděpodobnost obsazení buňky a po novém měření, $P(a)$ je předchozí pravděpodobnost obsazení buňky a, $p(r \mid a) = 1 - \neg p(r \mid a)$ je pravděpodobnost obsazení buňky a zjištěná senzorem a je rovna

$$p(r \mid a) = \frac{1 + \mathrm{model}_o^r(\alpha, d) - \mathrm{model}_v^r(\alpha, d)}{2},$$

kde model odpovídá použitému senzoru.

Model laserového dálkoměru

Laserový dálkoměr funguje na principu měření doby letu úzkého laserového paprsku, který je rozmítán rotujícím zrcátkem v rozsahu $\langle -90, 90\rangle$ stupňů. Důležitý parametr laserového senzoru je počet vzorků, tedy počet měřených paprsků na jeden průchod zorným polem, který je nepřímo úměrný úhlovému rozlišení laseru. Další potřebný parametr je maximální dosah laseru, což je vzdálenost, do které měří laser s dostatečnou přesností. Laserový dálkoměr je velmi přesný [1], problém ovšem nastává při spekulárních (zrcadlových) odrazech, kdy se vyslaný paprsek neodrazí zpět k senzoru, což může způsobit naměření větší vzdálenosti překážky než jaká je ve skutečnosti. Při řešení úlohy explorace je uvažován model laseru, kde pro každý paprsek platí:

$$r < d, p(r \mid a) = 0,9,$$

[1] Současné používané dálkoměry dosahují udávané přesnosti v řádu jednotek milimetrů, což s ohledem na volbu velikosti mřížky v jednotkách centimetrů je více než dostačující.

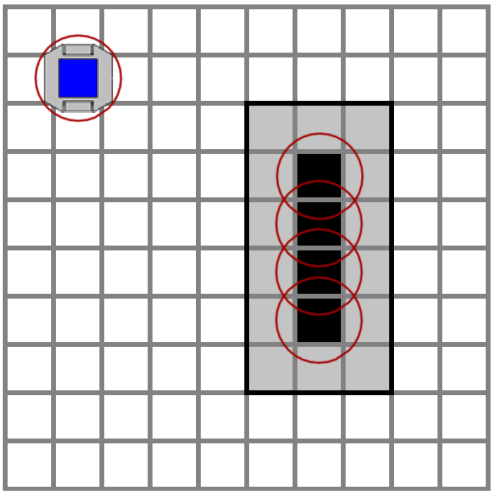
$$x < r - \text{velikost buňky}, p(x \mid a) = 0,3,$$

r je vzdálenost naměřená laserem, d je dosah laseru a $p(r \mid a)$ je pravděpodobnost obsazení buňky a ve vzdálenosti r.

Při pohybu robotu jsou uvedeným způsobem průběžně aktualizovány hodnoty obsazenosti buněk mapy. Při dostatečně velkém rozlišení senzoru se vytváří souvislé volné plochy nebo obsazené linie. Jistota obsazenosti resp. neobsazenosti buňky je úměrná počtu měření a integrací buňky do mapy. Frekvence integrace je téměř stejná jako frekvence příchodu nových dat (10 Hz), proto je jistota obsazenosti resp. neobsazenosti po krátké době poměrně vysoká.

Nafukování překážek

Pokud má být vytvářena mapa použita pro navigaci robotu, je potřeba ji nejprve upravit tak, aby navigační algoritmy respektovaly skutečné rozměry robotu. Dále popsané algoritmy počítají s nulovými rozměry robotu, u kterého je poloha robotu v prostředí určena pouze jediným bodem. Vzhledem k tomu, že robot má nenulové rozměry, je třeba mapu upravit tak, aby se i při tomto zjednodušení nemohlo stát, že by robot narazil do překážky. Toho je dosaženo použitím techniky „nafukování překážek", kdy jsou okraje překážek rozšířeny v každém směru o požadovanou vzdálenost. V tomto případě je kolem každé buňky představující překážku vykreslen kruh o poloměru opsaná kružnice robotu, neboť je uvažován diskový robot s diferenciálním typem podvozku. Každá buňka, do které kruh zasahuje je označena jako překážka. Tento postup je zobrazen na obrázku 2.2.



Obrázek 2.2: Princip nafukování překážek.

Po uvedené úpravě a při použití plánovacích algoritmů uvedených dále je docíleno toho, že se robot nikdy nedostane k překážce tak blízko, aby do ní narazil. Nejbližší body k překážce, na kterých se může vyskytovat střed robotu leží na černé hranici kolem nafouknuté překážky.

2.3 Plánování cesty

V této práci je zvolena explorační strategie, kdy je robot navigován vždy k nejbližšímu frontieru. Při plánování pohybu robotu k nim je třeba brát v úvahu překážky nalezené při mapování a plánovat trajektorii tak, aby se jim vyhnul a neuvízl ani na nekonvexních překážkách. To vše je potřeba provádět současně s vytvářením mapy. Tento problém je řešen tak, že nová data jsou do mapy integrována v okamžiku, kdy jsou k dispozici a po ujetí určité vzdálenosti robotem je přeplánována trajektorie budoucího pohybu pro případ, že se v dříve naplánované trajektorii objevily nové překážky.

Podle frekvence opakování rozdělují autoři [5] algoritmy řídicí robot do následujících vrstev:

1. strategické rozhodování - vyhledávání a určování cílových frontierů,
2. taktické rozhodování - plánování cesty na mřížce,
3. quasi real-time - integrace aktuálních dat z laseru do mřížky s mapou, collision avoidance,
4. hard real-time (výpočet skončí dříve než jsou k dispozici nová data) - regulace rychlosti.

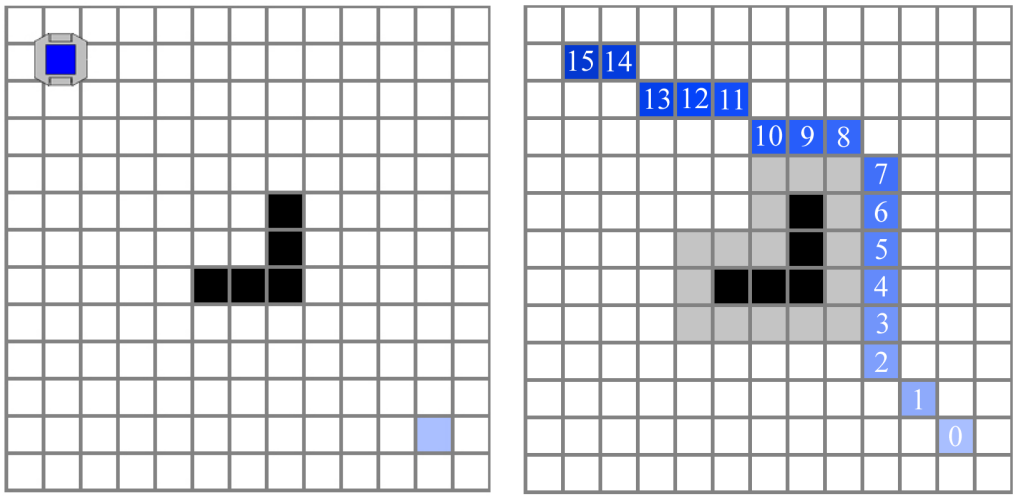
Algoritmus nalezení cesty je založen na ohodnocení buněk volného prostoru gradientním algoritmem „flood fill" [8]. Ten jednotlivé buňky označí číslem, které vyjadřuje kolik buněk je k nim třeba projít na cestě z cílové buňky. Příklad výstupu algoritmu je znázorněn na obrázku 2.3, kde na modré buňce se nachází robot a červená buňka je cíl.

9	8	7	6	5	4	3	2	1	0
9	8	7	6	5	4	3	2	1	1
9	8	7	6	5	4	3	2	2	2
9	8	7	■	■	■	■	3	3	3
9	8	8	8	9	10	■	4	4	4
9	9	9	9	9	9	■	5	5	5
10	10	10	10	9	8	■	6	6	6
11	11	11	10	9	8	7	7	7	7
	12	11	10	9	8	8	8	8	8
		11	10	9	9	9	9	9	9

Obrázek 2.3: Flood fill - ohodnocení buněk podle počtu buněk, které je k nim třeba projít na cestě od cíle (buňka s hodnotou 0).

Mřížka s mapou obsahuje buňky, které jsou ohodnoceny pravděpodobností obsazenosti, jelikož algoritmus „flood fill" uvažuje pouze buňky volného prostoru, je pro plánování

vytvořena nová mřížka. Nejdříve je mřížka obsazenosti převedena na pomocnou mřížku, do které jsou převedeny pouze překážky, které jsou uvažovány jako buňky a mající $P(a) > 0,5$. Následně je takto získaná mřížka upravena algoritmem nafukování překážek a použita pro plánování cesty. Příklad pomocné mřížky a nalezené cesty je zobrazen na obrázku 2.4(a), resp. 2.4(b), kde je robot umístěn v levém horním rohu a cílová buňka v pravém dolním rohu.



(a) Robot v mapě s překážkou.

(b) Naplánovaná cesta kolem překážky.

Obrázek 2.4: Plánování cesty na mřížce. Černé buňky jsou překážky, šedé představují nafouknutí překážek.

Použitý algoritmus je navíc doplněn o heuristiku udávající eukleidovskou vzdálenost jednotlivých buněk od buňky s robotem, která při číslování buněk vybírá ty, které jsou nejblíže robotu. S použitím této heuristiky není třeba procházet všechny prázdné buňky v mřížce. To urychlí nalezení cesty především v mřížkách s velkým počtem buněk a v případech, kdy je třeba nalézt cestu do blízkého cílového bodu. Tímto způsobem je nalezena nejkratší bezpečná cesta, která se vyhne dosud objeveným překážkám. Na obrázku 2.4(b) je vidět jak algoritmus postupuje. Směrem od cílové buňky jsou buňky ohodnocovány rostoucím číslem, dokud se nedojde k buňce s robotem. Pokud se použije uvedená heuristika, jsou ohodnoceny pouze buňky ležící na cestě od robotu k cíli. Všem neoznačeným buňkám je přiřazeno vyšší číslo, než se vyskytuje kdekoli v ohodnocené cestě. Robot se pak pohybuje vždy na sousední buňku s nejnižší hodnotou, dokud nedorazí do cíle.

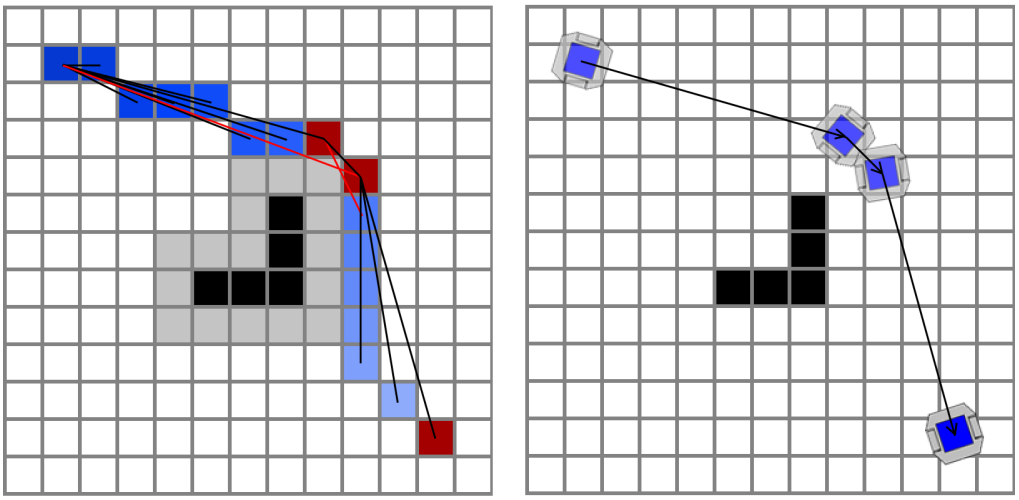
Předchozí algoritmus ohodnocuje vždy sousední prázdné buňky směrem od cílové buňky. Pokud už jsou všechny prázdné buňky v souvislé oblasti, ve které se nachází cílová buňka ohodnocené a stále se nedošlo k cílové buňce, je cílová buňka nedostupná a cesta k ní neexistuje.

Pokud je cesta nalezena, skládá z posloupnosti sousedících buněk. Taková cesta však může být zbytečně komplikovaná, proto je nalezená cesta dále upravena algoritmem vyhlazení.

Vyhlazení cesty

Aby byl pohyb robotu plynulejší a aby se nepohyboval stále jen po sousedních buňkách, což by bylo při malé velikosti buněk velmi pomalé, je vhodné cestu zjednodušit, neboli vyhladit - redukovat počet bodů, ve kterých musí robot měnit směr. Nalezená posloupnost buněk se převede na co nejmenší nutný počet buněk, kterými musí robot projet, aby se vyhnul překážkám. K tomu je využit Bresenhamův algoritmus pro kreslení úsečky na mřížce [3].

Princip činnosti algoritmu pro zjednodušení cesty je zobrazen na obrázku 2.5(a). V zobrazené situaci se robot při cestě do pravého dolního rohu naviguje tak, aby postupně projel třemi červenými buňkami. Jak je vidět na obrázku 2.5(b), robot při cestě změní směr pouze na dvou místech a pohybuje se v dostatečné vzdálenosti od překážek.



(a) Zjednodušování cesty. (b) Pohyb robotu.

Obrázek 2.5: Princip zjednodušení naplánované cesty na mřížce. Černé buňky jsou překážky, šedé představují nafouknutí překážek, na červené buňky reprezentují robot navigovaný do cíle.

Postup pro vyhlazení cesty uvádí algoritmus 2.

Algoritmus 2: Vyhlazení cesty

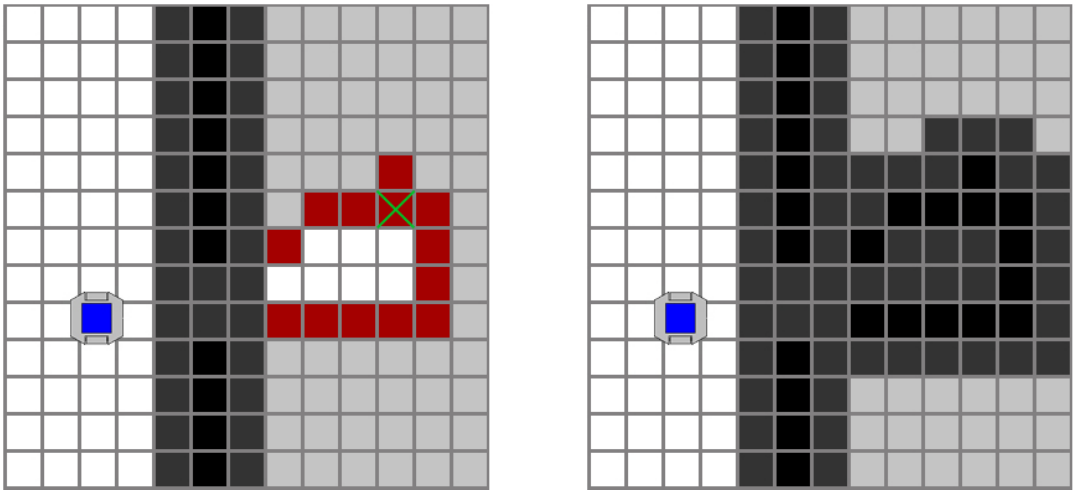
```
   Data: vstupní seznam buněk
   Result: zjednodušená cesta (posloupnost buněk)
 1 begin
 2     zjednodušená cesta := prázdný seznam buněk
 3     i = 1 // i je index buňky ve vstupním seznamu
 4     while i ≤ velikost vstupního seznamu do
 5         j = i
 6         while j ≤ velikost vstupního seznamu do
 7             j++
 8             if úsečka z buňky vstupní seznam(i) do buňky vstupní seznam(j)
               neobsahuje překážku then
 9                 přejdi na začátek cyklu
10             end
11             else
12                 Vlož do zjednodušené cesty buňku na pozici j − 1 ve vstupním
                   seznamu
13                 konec cyklu
14             end
15             if j = velikost vstupního seznamu then
16                 i = j
17                 vlož buňku vstupní seznam(j) do zjednodušené cesty break
18             end
19         end
20         i++
21     end
22 end
```

2.4 Vyhledávání frontierů

Při exploraci je robot navigován k frontierům, které tvoří rozhraní mezi známým a neznámým prostorem. Nalezení těchto hraničních buněk je založena na postupném vytvoření nové mřížky, která obsahuje pouze frontiery. Podle mřížky s mapou je nejdříve vytvořena mřížka, kde jsou jednotlivé buňky označeny podle pravděpodobnosti obsazení p jako:

- prázdné pro $p < 0,5$,
- neznámé pro $p = 0,5$ nebo
- obsazené pro $p > 0,5$.

Všechny prázdné buňky, které mají ve svém okolí (osm nejbližších buněk) alespoň jednu neznámou buňku, jsou označeny jako buňky tvořící frontier. Tyto buňky jsou poté seskupovány do souvislých oblastí určité velikosti. Tyto oblasti jsou hledané frontiery, které jsou uloženy do fronty podle vzdálenosti od robotu. Robot je poté navigován k nejbližšímu frontieru ve frontě. To znamená do středu buňky, která je součástí frontieru a je nejblíže středu frontieru, který je vypočtený jako aritmetický průměr středů všech buněk, které jej tvoří.



(a) Nedostupný frontier.

(b) Odstranění nedostupného frontieru.

Obrázek 2.6: Problém nedostupnosti frontieru po nafouknutí překážek. Černé buňky jsou překážky, tmavě šedé překážky po nafouknutí, světle šedé neznámý prostor a červené buňky jsou součástí frontieru.

Pokud algoritmus plánující cestu nenalezne volnou cestu k frontieru, je tento frontier vyřazen ze seznamu, a buňky, které jej tvoří jsou v mřížce s mapou označené jako obsazené, aby tento nedostupný frontier nebyl znovu nalezen při přeplánování. Taková situace může nastat například při průjezdu kolem úzkých dveří, kdy za nimi robot vidí rozhraní mezi

neznámým a prázdným prostorem, čili frontier. K takovému frontieru se však robot nemůže bezpečně dostat, neboť po nafouknutí překážek není příslušná cesta nalezena. Příklad takové situace je zobrazen na obrázku 2.6, kde černé buňky představují překážky, bílé volný prostor, světle šedé neznámý prostor, tmavě šedé nafouknuté překážky a červené buňky jsou součástí frontieru, zelený křížek označuje bod, ke kterému je robot navigován.

Použité algoritmy při exploraci prostředí

V následujícím výčtu jsou shrnuty dílčí algoritmy používané v úloze robotické explorace využívající navigaci k frontierům:

- integrace dat ze senzorů do mapy,
- vytvoření pomocné mřížky podle pravděpodobnosti obsazení buněk a prahové hodnoty pravděpodobnosti 0,5,
- nalezení frontierových buněk,
- seskupení buněk do frontierů a výběr frontieru,
- nafouknutí překážek,
- plánování cesty na mřížce,
- vyhlazení cesty,
- navigace robotu na nalezené cestě.

2.5 Explorace prostředí skupinou robotů

Tato části se věnuje popisu rozšíření problému explorace jedním robotem na exploraci skupinou robotů. Pro zjednodušení je použit přístup, při které je udržována pouze jediná mřížka reprezentující aktuální mapu prohledávaného prostředí a jediný seznam frontierů, které jsou sdíleny všemi roboty a cílové frontiery jsou robotům přidělovány centrálním plánovačem.

Aktualizace mapy, plánování cesty a vyhledávání frontierů je řešeno identickými postupy jako v předchozí části. Řešení multirobotické explorace rozšiřuje předchozí úlohu o tyto algoritmy:

- koordinace - vzájemné vyhýbání robotů,
- zabránění mapování robotů jako překážek,
- kooperace - plánování přidělování frontierů jednotlivým robotům.

Koordinace robotů

Koordinací robotů je především myšleno jejich vzájemné vyhýbání. Proto je již při samotné spuštění úlohy nutné definovat vzájemnou polohu robotů, která je explicitně zadána v počáteční mřížce s mapou. Aby se robot při exploraci vyhnul srážce s jiným robotem, neustále při svém pohybu kontroluje, jestli není jiný robot příliš blízko. Pokud jsou dva roboty blízko u sebe, jeden z nich nastaví svůj status na „čekající" a čeká než se druhý robot bezpečně vzdálí. Robot, který nečeká, zaznamená čekající robot do své navigační mapy jako překážku, přeplánuje svou cestu, aby se s čekajícím robotem nesrazil a pokračuje v jízdě. Tuto situaci je třeba ošetřit tak, aby oba roboty nezačaly provádět identickou akci, například čekat. Pokud přijede robot do blízkosti již čekajícího robotu, zaznamená ho jako překážku, přeplánuje svou cestu a pokračuje v jízdě. Robot je zaznamenán jako překážka právě do navigační mřížky proto, že při nejbližším přeplánování tato překážka zmizí (navigační mřížka je znovu vytvořena z mřížky s mapou).

Řešení vzájemného mapování robotů

Při multirobotické exploraci také může nastat situace, kdy je dočasně se nepohybující robot namapován jiným robotem jako překážka a vzhledem k tomu, že mapa je společná, byl by v takovém případ uvězněn uvnitř překážky a jeho další pohyb by byl znemožněn (nelze najít cestu skrze překážku). Pokud se robot nebo jiný objekt v prostředí pohybuje, z dlouhodobého pohledu se jeho přítomnost v mapě neprojeví, protože je překryta dalšími měřeními již prázdného prostoru.

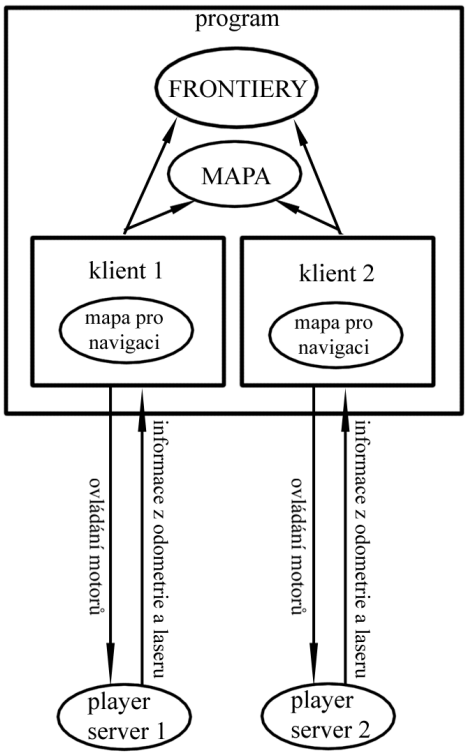
Namapování jiného robotu jako překážky je řešeno tak, že vždy po integraci nové série vzorků z laserového dálkoměru do mapy, jsou buňky, na kterých se nachází ostatní roboty, označeny jako prázdné. Tímto se překryje nesprávně interpretovaná informace z laserového dálkoměru v mřížce s mapou.

Kooperace robotů

Kooperace robotů je rozdělení úkolu mapování tak, aby prohledaly prostředí pokud možno efektivně, například rychleji než jediný nebo nižší počet robotů. Základním přístupem je, aby roboty neplánovaly ve stejný čas jízdu ke stejnému frontieru. Tím se také sníží pravděpodobnost, že se trajektorie robotů bude křížit a roboty se budou muset vzájemně objíždět, což způsobuje výrazné zpomalení explorace. Přiřazování různých frontierů robotům je řešeno společným seznam frontierů, a jednotlivé roboty z něj vybírají vždy nejbližší frontier, který je poté ze seznamu vymazán. Proto se nemůže stát, že by si dva roboty naplánovaly cestu ke stejnému frontieru. Ověření, jak tento přístup urychluje dobu explorace je součástí experimentálního ověření, které je součástí kapitoly 3 věnované popisu experimentů.

Základní architektura řídicí aplikace

Schéma znázorňující komunikaci s roboty a základní architekturu řídicí aplikace realizující multirobotickou exploraci je uvedeno na obrázku 2.7. Jednotlivé klienty jsou části, které ovládají robot a získávají informace rozhraním systému Player, a běží paralelně v samostatných vláknech. Rozhraní jsou poskytována serverem, který představuje reálné či simulované roboty.



Obrázek 2.7: Multirobotická explorace - struktura programu a komunikace s roboty.

Kapitola 3

Experimenty

Úloha multirobotické explorace popsaná v předešlé kapitole byla implementována v jazyce C++ s využitím knihoven systému Player verze 2.0.5, pro ladění bylo použito simulační prostředí Stage verze 2.0.4 [1]. Při reálných experimentech byly použity mobilní robotické platformy G^2Bot, které mají dostatečně přesnou odometrii pro lokalizaci robotu při exploraci prostředí. Pro získávání informací o okolním prostředí jsou roboty vybaveny laserovými dálkoměrnými senzory. Komunikace je řešena v rámci řídicího programu, který ovládá roboty prostřednictvím rozhraní serveru Player. Program byl použit pro simulované i reálné experimenty. Při reálném experimentu byl na každém robotu spuštěn server Player poskytující příslušná rozhraní, k těmto serverům byla připojena řídicí aplikace bezdrátovou sítí WiFi.

Klientský program byl přeložen kompilátorem gcc verze 4.1.3 a spouštěn na počítači s operačním systémem Ubuntu 7.10 a s jednojádrovým procesorem na frekvenci 1,8 GHz.

Experimenty byly zaměřeny na vliv počtu robotu na rychlost explorace a také ověření významu kooperace více robotů při exploraci. Nejdříve byly algoritmy ověřeny v prostředí simulátoru a následně byla aplikace spuštěna s reálnými roboty. Experimentální výsledky jsou uvedeny v následujících oddílech.

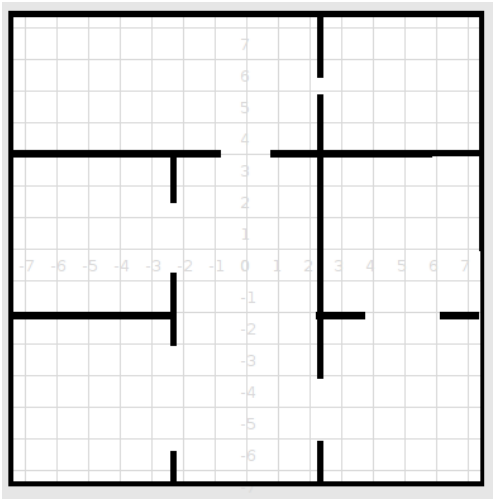
3.1 Experimenty v simulátoru

V této sekci jsou popsány experimenty provedené v simulátoru. K experimentům bylo použito prostředí o rozměrech 15m $\times$ 15m, které je zobrazeno na obrázku 3.1(a). Toto prostředí bylo simulováno v robotickém simulátoru Stage, který byl spuštěn jako zásuvný modul serveru Player. Vytvářena mapa byla reprezentována mřížkou o rozměrech 150×150 buněk, tj. s rozměry buňky 0,1m$\times$ 0,1m. Dosah laserových dálkoměrů byl omezen na 5m.

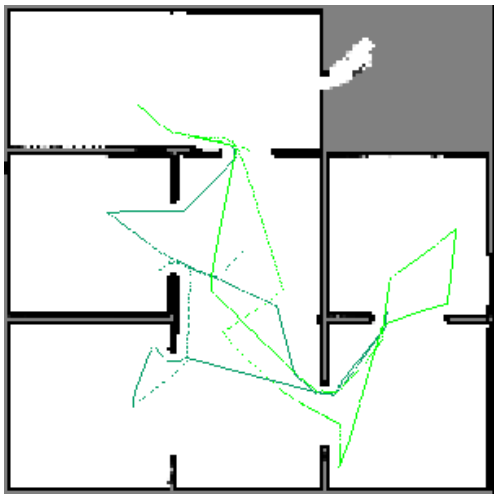
Výkon použitého výpočetního systému dostačoval pro řízení tří nekooperujících robotů. Při vyšším počtu robotů již docházelo k častým kolizím mezi roboty v důsledku nedostatečného výpočetního výkonu použitého výpočetního systému. V případě použití algoritmu kooperace v řídicí aplikaci, byly pozorováno výrazně nižší nároky na výpočetní prostředky. A to je zejména z toho důvodu, že je při kooperativním přístupu je vyčíslován pouze jediný

seznam frontierů na rozdíl od nekooperovaného přístupu, při kterém je výpočet seznamu frontierů realizován pro každý robot zvlášť.

Experimentálně byl ověřen vliv počtu robotů na rychlost explorace a dále význam kooperace. Obrázek 3.1(b) znázorňuje mapu prostředí vytvořenou při exploraci dvěma kooperujícími roboty.



(a) Prostředí použité pro experimenty v simulátoru.



(b) Mapa vytvořená při exploraci dvěma kooperovanými roboty. Zelené křivky jsou trajektorie robotů.

Obrázek 3.1: Mapa prostředí pro experimenty v simulátoru.

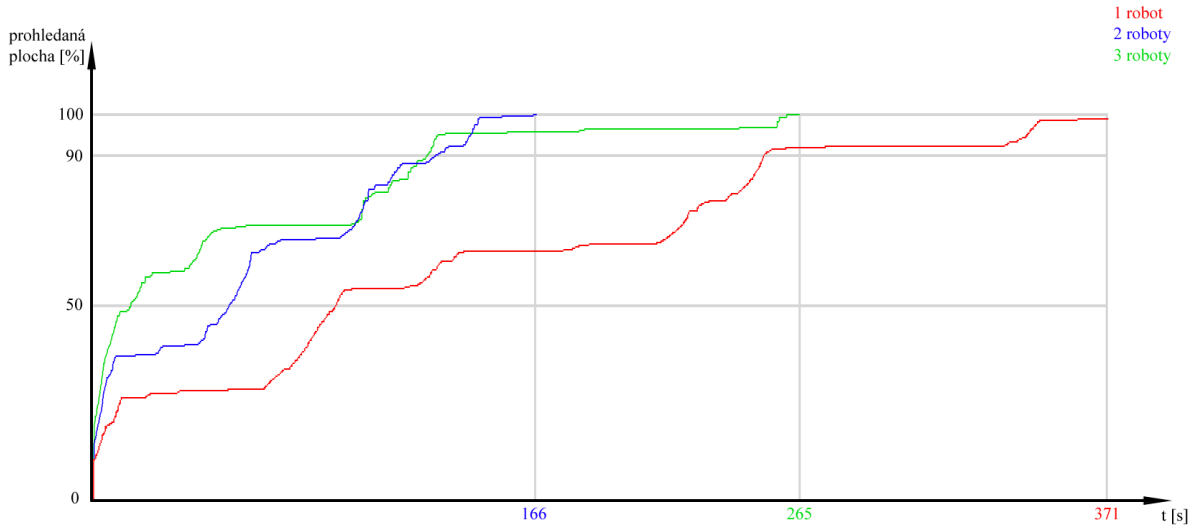
Vliv počtu robotů na rychlost explorace

Na obrázku 3.2 je uveden graf závislosti procent prohledaného prostoru na čase při exploraci jedním, dvěma a třemi roboty. Podle očekávání, nejdéle trvalo zmapovat prostředí jedinému robotu. Explorace dvěma roboty v tomto experimentu však skončila dříve než explorace třemi roboty. Při exploraci třemi roboty si roboty v malém prostředí navzájem překáželi a objíždění robotů způsobilo ke konci explorace zpomalení, jak si lze všimnout na zeleném průběhu na obrázku 3.2. Přitom zpočátku experimentu byla explorace třemi roboty nejrychlejší.

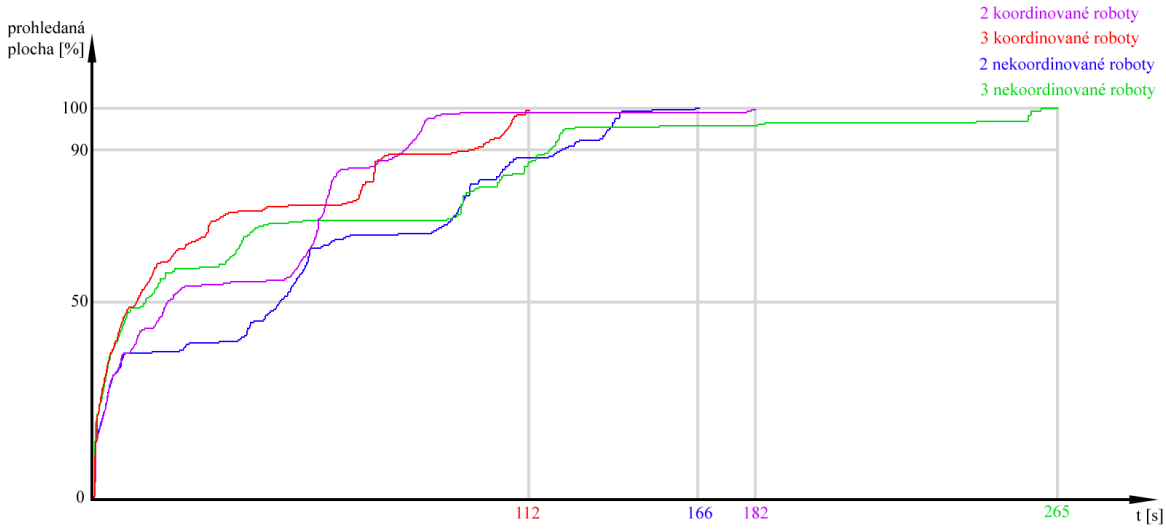
Vliv kooperace na rychlosti explorace

Na obrázku 3.3 je uveden graf závislosti procent prohledaného prostoru na čase při exploraci dvěma a třemi nekooperovanými a kooperovanými roboty. Z průběhů je vidět, že v případě aktivované kooperace, skončila explorace výrazně rychleji. Také si lze všimnout, že při kooperované exploraci skončí dříve tři roboty, u nekooperované explorace je tomu naopak. Kooperace tedy pomohla vyhnout se situacím, kdy se roboty musí vzájemně objíždět, což v důsledku vede na celkové zpomalení explorace. Jsou-li uvažovány časy dosažení 90%

prohledaného prostoru z grafů na obrázku 3.2 a 3.3, tak rozdíl rychlosti explorace dvěma a třemi roboty není tolik významný. Pomalé přírůstky prohledaného prostoru u konce většiny průběhů jsou způsobené tím, že se roboty vracejí už prohledaným prostorem k malým neprohledaným oblastem.



Obrázek 3.2: Graf prohledané plochy v závislosti na čase - porovnání explorace jedním, dvěma a třemi roboty v simulátoru.



Obrázek 3.3: Graf prohledané plochy v závislosti na čase - porovnání explorace nekooperovanými a kooperovanými roboty v simulátoru.

3.2 Experiment s reálným robotem

Reálný experiment byl proveden se dvěma mobilními roboty G^{2}Bot. Tato robotická platforma je používána v řešitelském pracovišti pro experimenty ve vnitřních prostorách, platforma je zobrazena na obrázku 3.5(a). Obrázek 3.2 zobrazuje požitý laserový dálkoměr SICK LMS200, kterým byly roboty vybaveny. Parametry laseru jsou uvedené v tabulce 3.1, dosah byl však z důvodu velikosti prohledávaného prostoru omezen na 4 m. K řízení robotů byl při experimentu v reálném prostředí použit nekooperující přístup.

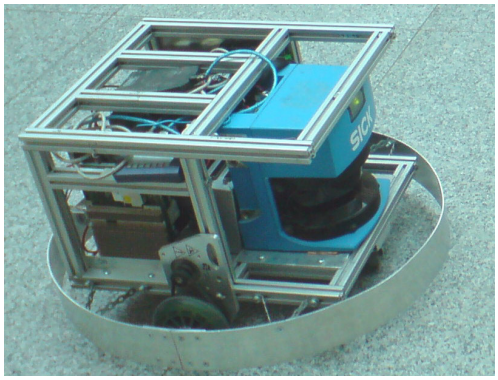
Prohledávaným prostředím byla uzavřená oblast o rozměrech cca 10 m $\times$ 10 m, která je zobrazena na obrázku 3.5(b). Mapa byla vytvářena na mřížce s velikostí buňky 0,1m $\times$ 0,1m, mřížka s rozměry 150 $\times$ 150 buněk. Výsledná vytvořená mapa tohoto prostředí je na obrázku 3.5(c). Z přesnosti vytvořené mapy lze usuzovat, že lokalizace z odometrie byla v tomto případě dostačující.

Obrázku 3.6 obsahuje graf průběhu prohledávání. Již na první pohled je patrné, že má podobný průběh jako při experimentech v simulátoru. Nejvíce prostoru je prohledáno na začátku, ke konci se rychlost přibývání prohledaného prostoru zmenšuje, jak se roboty pohybují již částečně prohledaným prostorem.

Obrázek 3.4: Použitý laserový dálkoměr - SICK LMS200, převzato z [2].

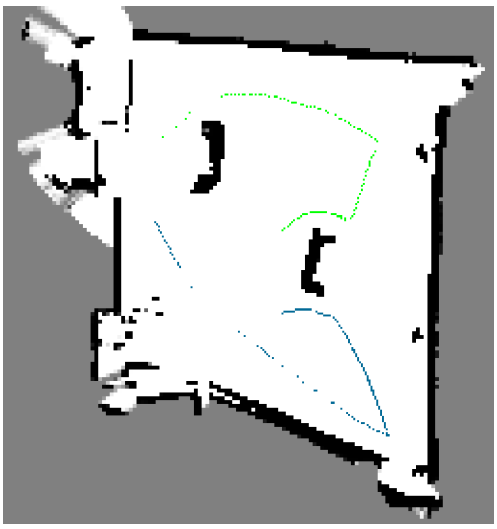
Zorné pole	180°
Úhlové rozlišení	0, 5°
Dosah	8m
Délkové rozlišení	1mm
Vzorkovací frekvence	5 Hz

Tabulka 3.1: Parametry použitého laserového dálkoměru SICK LMS200

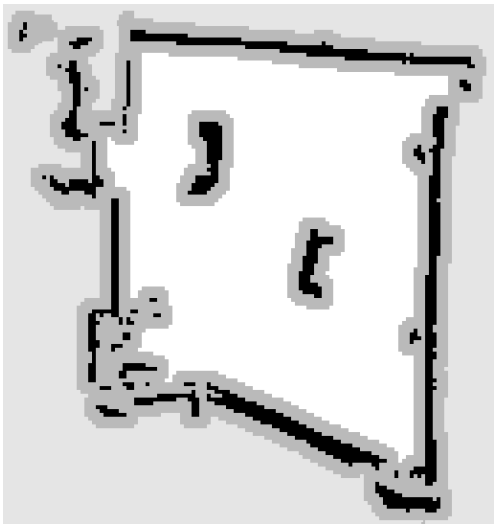


(a) Mobilní platforma G^{2}Bot použitá pro reálné experimenty.

(b) Prostředí prohledávané při reálném experimentu.

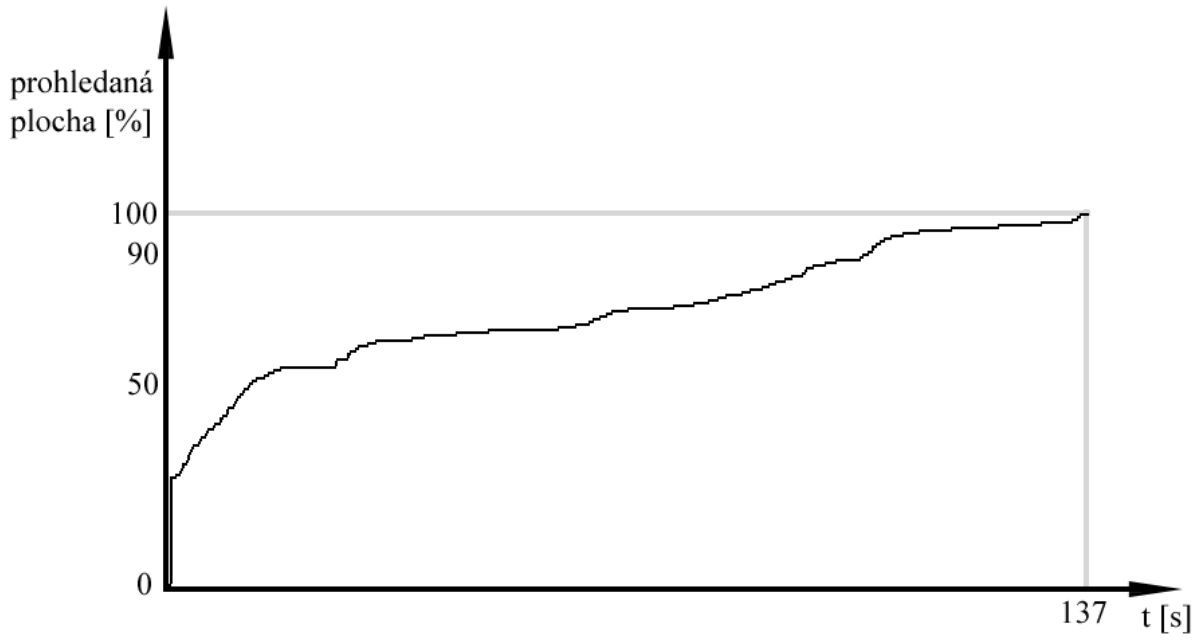


(c) Mapa vytvořená dvěma roboty. Zelená a modrá křivka jsou trajektorie robotů.



(d) Mapa vytvořená dvěma roboty s nafouknutými překážkami.

Obrázek 3.5: Experiment s reálnými roboty.



Obrázek 3.6: Graf prohledané plochy v závislosti na čase z experimentu s reálnými roboty.

Kapitola 4

Závěr

V práci bylo navrženo a popsáno řešení úlohy robotické explorace založené na navigaci k frontierům, které bylo dále rozšířeno na exploraci více roboty. Řešení bylo inspirováno systémem ARIEL, popsaným v článcích [10] a [9] a bylo navíc rozšířeno o aktualizaci mapy v reálném čase, častější přeplánovávání cílových frontierů a cesty a v případě více robotů také o jejich koordinaci a kooperaci.

Navržené řešení bylo experimentálně ověřeno. Multirobotická explorace byla implementována v jazyce C++ jako vícevláknová aplikace. Tato aplikace byla použita pro experimenty v simulátoru, při kterých byl ověřen vliv počtu robotů a vliv kooperace na rychlost explorace. Dále byly provedeny experimenty s reálnými roboty, které prokázaly, že lze vytvořený systém použít pro exploraci reálného statického prostředí uvnitř budov. Průběh experimentu v reálném prostředí se příliš nelišil od průběhu experimentu v simulátoru a to především v důsledku dostačující přesnosti lokalizace reálných robotů, při které byla uvažována pouze odometrie.

Přirozeným rozšířením navrženého algoritmu je řešení úlohy lokalizace tak, aby bylo možné použít metodu i v případě, kdy nedosahuje odometrie dostačující přesnosti. Metodu lokalizace lze založit na některé z metod uvedených v [10], [7]. V případě multirobotické explorace je také výhodné použít distribuovaný přístup, kdy jsou jednotlivé roboty více autonomní a každý robot rozhoduje o tom kam se navigovat. Distribuovaným přístupem lze dosáhnout vyšší robustnosti systému při selhání jednotlivých robotů.‘

Literatura

[1] The Player Project [online]. 2008.
URL `http://playerstage.sourceforge.net/`

[2] Sick: webové stránky výrobce senzorů a katalog [online]. 2009.
URL `www.mysick.com`

[3] Bresenham, J. E.: Bresenhams line algorithm. 1962.
URL `www.en.wikipedia.org/wiki/Bresenham%27s_line_algorithm`

[4] Dudek, G.; Jenkin, M.: *Computational Principles of Mobile Robotics*. Cambridge, U.K.: Cambridge University Press, 2000.

[5] Siegwart, R.; Nourbakhsh, I. R.: *Introduction to Autonomous Mobile Robots*. Cambridge, Massachusetts: The MIT Press, 2004.

[6] Thrun, S.: Robotic Mapping: A Survey. 2002.

[7] Thrun, S.; Burgard, W.; Fox, D.: *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. Leuven, Belgium: The MIT Press, 2005.

[8] Winkler, Z.: Plánování na mřížce [online]. 2003.
URL `http://robotika.cz/guide/gridplan/cs`

[9] Yamauchi, B.: Frontier-based exploration using multiple robots. *AGENTS '98: Proceedings of the second international conference on Autonomous agents*, 1998: s. 47–53.

[10] Yamauchi, B.; Schultz, A.; Adams, W.: Mobile robot exploration and mapbuilding with continuous localization. *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, 1998.

Obsah CD

Přiložené CD obsahuje zdrojové kódy pro multirobotickou exploraci, text diplomové práce ve formátu PDF a zdrojové kódy celého textu pro systém LaTeX. V následující tabulce je popsána struktura CD.

Adresář	Popis
`src`	zdrojové kódy knihovny
`doc`	zdrojové kódy textu diplomové práce
`thesis.pdf`	text diplomové práce

Tabulka 1: Adresářová struktura na CD