



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

**DETEKCE POHYBLIVÝCH OBJEKTŮ V PROSTŘEDÍ
MOBILNÍHO ROBOTA**

MOVING OBJECT DETECTION IN THE ENVIRONMENT OF MOBILE ROBOT

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE

AUTHOR

Bc. VIKTOR DOROTOVIČ

VEDOUCÍ PRÁCE

SUPERVISOR

Ing. MARTIN VEĽAS

BRNO 2017

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií

Akademický rok 2016/2017

Zadání diplomové práce

Řešitel: **Dorotovič Viktor, Bc.**

Obor: Počítačová grafika a multimédia

Téma: **Detekce pohyblivých objektů v prostředí mobilního robota**
Moving Object Detection in the Environment of Mobile Robot

Kategorie: Zpracování obrazu

Pokyny:

1. Prostudujte základní metody zpracování obrazu a hloubkových dat. Zaměřte se na problematiku detekce pohyblivých objektů v prostředí, v kterém se pohybuje mobilní robot, autonomní automobil a pod.
2. Získejte bližší pohled na techniky a existující řešení, které se v současnosti používají v oblasti detekce pohybu.
3. Navrhněte jednoduchý systém, který je schopný detekovat pohyblivé části okolí robota na základe sensorických 2D a 3D dat.
4. Implementujte navržený systém s vhodným využitím existujících řešení
5. Získejte testovací datovou sadu pro testování.
6. Navrhněte a realizujte experimenty s Vaší implementací. Zhodnoťte dosažené výsledky a možnosti budoucího vývoje.
7. Vytvořte video nebo stručný plakát, který prezentuje obsah vaší práce a dosažené výsledky.

Literatura:

- Dle pokynů vedoucího.

Při obhajobě semestrální části projektu je požadováno:

- Bez požadavků.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese <http://www.fit.vutbr.cz/info/szz/>

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci dřívějších projektů (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Veřás Martin, Ing.**, UPGM FIT VUT

Datum zadání: 1. listopadu 2016

Datum odevzdání: 24. května 2017

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 56 Brno, Božetěchova 2



doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Táto práca rieši problém detekcie pohybujúcich sa objektov v okolí robota. Prostredie je reprezentované dvojrozmernou okupačnou mriežkou, ktorá obsahuje aktuálne viditeľné prostredie, bez filtrovania v čase. Ako samotný detektor pohybu slúži časticový filter založený na systéme v článku Grid-based Mapping and Tracking in Dynamic Environments using a Uniform Evidential Environment Representation, ktorý uviedol Tanzmeister a kolektív. Implementácia s využitím Robotického operačného systému poskytuje možnosť pre znovupoužitie modulov, z ktorých riešenie pozostáva. Ako zdroj LiDARových dát pre experimenty bola zvolená databáza KITTI Visual Odometry, ktorá obsahuje aj pózy vozidla. Mračná bodov boli predspracované vynechaním bodov ležiacich na zemi metódou Loopy Belief Propagation. Vytvorený detektor dokáže na sekvenciách databázy rozlišovať pohybujúce sa vozidlá. Pri testoch na simulovanom prostredí sa ukázali nedostatky detekcie v prípade pohybu veľkých súvislých objektov.

Abstract

This work's aim is movement detection in the environment of a robot, that may move itself. A 2D occupancy grid representation is used, containing only the currently visible environment, without filtering in time. Motion detection is based on a grid-based particle filter introduced by Tanzmeister et al. in Grid-based Mapping and Tracking in Dynamic Environments using a Uniform Evidential Environment Representation. The system was implemented in the Robot Operating System, which allows for re-use of modules which the solution is composed of. The KITTI Visual Odometry dataset was chosen as a source of LiDAR data for experiments, along with ground-truth pose information. Ground segmentation based on Loopy Belief Propagation was used to filter the point clouds. The implemented motion detector is able to distinguish between static and dynamic vehicles in this dataset. Further tests in a simulated environment have shown some shortcomings in the detection of large continuous moving objects.

Kľúčové slová

robotika, mračno bodov, okupačná mriežka, detekcia pohybu, časticový filter, ROS

Keywords

robotics, point cloud, occupancy grid, motion detection, particle filter, ROS

Citácia

DOROTOVIČ, Viktor. *Detekce pohyblivých objektů v prostředí mobilního robota*. Brno, 2017. Diplomová práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce Velas Martin.

Detekce pohyblivých objektů v prostředí mobilního robota

Prehlásenie

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne pod vedením pána Ing. Martina Velasa. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

.....

Viktor Dorotovič

24. mája 2017

Podakovanie

Rád by som poďakoval vedúcemu práce, Ing. Martinovi Velasovi, za odbornú diskusiu, cenné rady a nasmerovanie na užitočné nástroje. Tiež ďakujem Ing. Michalovi Hradišovi, Ph.D. za pripomienky k obhajobe semestrálneho projektu.

Obsah

1	Vnímanie v robotike	4
1.1	Senzory	4
1.1.1	Pasívne senzory	4
1.1.2	Aktívne senzory	4
1.2	Pravdepodobnostná robotika	5
1.2.1	Základné koncepty pravdepodobnosti	6
1.2.2	Dempster-Shaferova teória	7
1.2.3	Kumulatívny zlučovací operátor	10
1.3	Reprezentácia okolia	11
1.3.1	Stavový priestor a viera	13
2	Detekcia pohybujúcich sa objektov	19
2.1	KITTI Visual Odometry	19
2.2	Segmentácia zeme	20
2.3	Lokálna mapa	20
2.4	Časticový filter na okupačnej mriežke	21
2.4.1	Predikcia	22
2.4.2	Váhy častíc	22
2.4.3	Prevzorkovanie	24
2.4.4	Inicializácia	24
2.5	Model prostredia	25
2.5.1	Pohyb	25
2.5.2	Fúzia okupačnej mriežky a pohybu	27
2.5.3	Filtrovanie v čase	28
3	Implementácia	29
3.1	Robotický operačný systém	29
3.2	Uzly	29
3.3	Správy	30
3.4	roslaunch	30
3.5	Geometrické transformácie	30
3.6	Štruktúra	31
3.6.1	loader	31
3.6.2	pose_odometry	33
3.6.3	local_grid	33
3.6.4	particle_filter	34
3.6.5	requester	36
3.6.6	mapper	36

3.6.7	pcl_accumulator	36
3.6.8	local_grid_simulator	36
3.7	Vizualizácia	37
4	Experimentálne výsledky	38
4.0.1	Simulácia okupačnej mriežky	38
4.1	Testovanie na meraniach zo senzoru LIDAR	40
4.1.1	Akumulovaná mapa	42
4.2	Zrnutie	42
	Literatúra	45
	Prílohy	47
	A Obsah CD	48
	B Plagát	49

Úvod

Lokalizácia a mapovanie sú jedným z najdôležitejších problémov robotiky. Medzi oblasti záujmu v robotike autonómnych vozidiel patrí prevažne určovanie stavu robota a okolitého prostredia. Základnými problémami sú lokalizácia robota na základe existujúcej mapy a senzorov, ktoré snímajú bezprostredné okolie alebo v opačnom prípade, zoskupenie informácií z okolia do podoby mapy ak sú dostupné spoľahlivé údaje o polohe. SLAM (Simultaneous Localization And Mapping) spája tieto dve úlohy do jednej a lokalizáciu a mapovanie vykonáva súbežne. Bez ohľadu na to, za akým účelom sa pozoruje okolie robota, je žiaduce, aby model prostredia poskytoval informácie aj o dynamike, teda o častiach prostredia, ktoré sú voči iným v pohybe.

Cielom tejto práce je navrhnúť a zostaviť systém, ktorý zvládne rozlíšiť medzi statickými a pohybujúcimi sa časťami okolia robota, čo by umožnilo odfiltrovanie nežiadúcich meraní pre účely mapovania alebo lokalizácie. Prehľad o pohybujúcich sa objektoch v bezprostrednej blízkosti je rovnako možné použiť pre proaktívne vyhýbanie sa prekážkam.

Výsledný systém by mal mať malý počet vstupných parametrov a požadovanou vlastnosťou je aj schopnosť kompromisu medzi presnosťou a výpočtovým výkonom.

V niektorých špecifických prípadoch, ako sú štruktúrované interiéry alebo diaľničná premávka, je prijateľné prostredie modelovať geometrickými útvarmi, napríklad kvádrami[16], reprezentovanými pozíciou, veľkosťou a vektorom pohybu. Najčastejšie sa ale v bežnom prostredí nachádzajú objekty, ktoré nie je možné jednoducho parametricky popísať. Výsledné riešenie by malo teda vyžadovať čo najmenší počet a priori informácií v podobe parametrov, alebo modelov objektov. Výhodnou vlastnosťou by tiež bola organizácia algoritmu do takej podoby, aby bolo možné operácie čo najviac paralelizovať a tým umožniť napríklad spracovanie na grafických procesoroch (GPU).

Tato práca je rozdelená do 4 kapitol. **Prvá kapitola** poskytuje teoretické základy, ktoré boli v riešení použité a odkazujú na ne ostatné kapitoly. **Druhá kapitola** popisuje spôsob použitia časticového filtra ako detektora pohybu. Podrobnosti implementovaného systému v robotickom operačnom systéme ROS sú prebrané v kapitole **Implementácia**. **Posledná kapitola** obsahuje vyhodnotenie systému z hľadiska požadovaných vlastností a výkonu.

Kapitola 1

Vnímanie v robotike

Táto kapitola sa zaoberá teoretickými poznatkami snímania a reprezentácie prostredia. Popisuje tiež použitie pravdepodobnostných metód pre sledovanie objektov a určenie pohybových vektorov.

1.1 Senzory

Senzory umožňujú robotom vnímať fyzikálne veličiny prevedením na použiteľné signály pre spracovanie počítačom. Surový výstup sensorov je následne transformovaný technikami pre spracovanie signálu, výsledok je použitý pre získanie modelu prostredia.

1.1.1 Pasívne senzory

Pasívne senzory merajú javy, ktoré sa v prostredí bežne vyskytujú bez toho, aby do prostredia nejakým spôsobom zasahovali. Medzi pasívne senzory je možné zaradiť snímače intenzity svetla (fotočlánky, fotoodpory), mikrofóny, senzory teploty, tlaku vzduchu, mechanického tlaku, RGB kamery a pod.

V dôsledku pasivity sú náchylné na rôznorodosť prostredia, ktoré je potrebné rôzne kompenzovať. Napríklad pri použití RGB kamery v tme je nutné pre zachovanie dynamického rozsahu zvyšovať dobu expozície, čo má ale za následok zníženie rozlišovacej schopnosti v čase – rozmazanie pohybujúcich sa objektov.

1.1.2 Aktívne senzory

Aktívne senzory pracujú na princípe zásahu do prostredia (vyvolanie zmeny) a meraní odozvy tejto zmeny. Možnými technikami sú priame meranie doby letu (*Time Of Flight*), meranie pomocou fázového posuvu (*Phase-shift measurement*) a frekvenčná modulácia radarového signálu [13].

Najčastejšie používané Time-of-flight senzory pracujú na princípe merania času, za aký častica alebo vlna precestuje určitú vzdialenosť. Umožňujú získať informácie o prostredí periodickým vysielaním vln a detekciou odrazenej vlny. Predpokladom správneho merania je to, že materiál prekážok, ktorých pozícia sa má určiť, odrazí aspoň časť vlny späť smerom k senzoru. Doba medzi vyslaním impulzu a detekciou je potom priamo umerná vzdialenosti senzoru od prekážky.

Rozšírené Time-of-flight senzory podľa druhu vlny:

- **Sonar:**
Signálom je mechanické vlnenie (zvuk).
- **RADAR:**
Vysiela a prijíma rádiové spektrum elektromagnetického žiarenia.
- **LIDAR:**
Médiom je elektromagnetické žiarenie v pásme svetla, prevažne infračerveného.
- **Time-of-flight kamera:**
Používa vetlo rovnako ako LIDAR, ale na rozdiel od LIDAR-u dokáže nasnímať celú scénu jedným impulzom, výstup je ale v podobe rastra.

Výstup Time-of-flight senzoru môže byť ovplyvnený nasledujúcimi faktormi[13]:

- **Premenlivá rýchlosť šírenia pulzu:**
Problém prevažne v systémoch založených na vysielaní zvukových vln (sonary), kde rýchlosť môže byť ovplyvnená lokálnymi zmenami tlaku v dôsledku zmeny teploty.
- **Chyba detekcie odrazeného signálu:**
Môže ovplyvniť výpočet presnej doby letu. Chyby môžu byť spôsobené širokým dynamickým rozsahom odrazeného signálu spôsobeným rôznorodou odrazivosťou cieľového objektu. Výsledkom je, že niektoré málo odrazivé objekty nemusia byť viditeľné vôbec, alebo je detekovaná nesprávna vzdialenosť.
- **Chyba merania času:**
Prejavuje sa hlavne pri použití laserových senzorov. Rýchlosť svetla je vysoká, v dôsledku čoho je interval medzi vyslaným a detekovaným impulzom veľmi malý. Pre meranie vzdialeností v rozlíšení 30cm je potrebné čas merať v nanosekundách a pri milimetrovej presnosti dokonca v pikosekundách. Sonary na tento problém náchylné nie sú, pretože šírenie zvuku je mnohórádovo pomalšie.
- **Vlastnosti povrchu:**
Sú problémom všetkých systémov. Pri dopade vlny na povrch dochádza súčasne k čiastočnému pohlteniu signálu a jeho odrazeniu. Problematickými sú materiály, ktoré vlnu úplne pohltia, alebo povrchy, ktoré veľkú časť impulzu odrazia mimo detektor.

1.2 Pravdepodobnostná robotika

Oblasť robotiky venujúca sa použitiu a reprezentácii pravdepodobnosti v robotike, pravdepodobnostná robotika, rieši problémy vyplývajúce z nepresnosti vnímania a riadenia v modernej robotike. Využíva štatistické metódy pre reprezentáciu informácií a následné rozhodovanie [19, kap. 2.2].

Robotické systémy sú súčasťou fyzického sveta a dokážu ním manipulovať pomocou efektorov. Aby mohli vykonávať rôzne úlohy, musia byť schopné vysporiadať sa s veľkou neurčitou reálnym svetom. Existuje mnoho javov, ktoré prispievajú k neurčitosti robota.

V prvom rade, okolie robota je zo svojej podstaty nepredvídateľné. Kým miera neistoty v štruktúrovaných priestoroch typu montážnych hál a vopred navrhnutých priemyselných budov je malá, lokality ako cesty, diaľnice, interiéry obývacích priestorov sú dynamické a

v mnohých ohľadoch nepredvídateľné. Neurčitost' je vysoká najmä pri robotoch, ktoré sa vyskytujú v okolí ľudí, alebo s nimi aj interagujú.

Senzory popísané v kapitole 1.1 sú ďalším zdrojom nepresností v dôsledku fyzických limitov. Majú obmedzený dosah, nevidia za prekážky a ich rozlíšenie je konečné. V signále senzoru je prítomný šum, ktorý náhodne ovplyvňuje merania a tým obmedzuje informácie, ktoré je možné vyťažiť. Detekovanie poruchy senzoru je obzvlášť náročné.

Ďalším zdrojom nepresnosti sú aktuátory robotov, nielen týkajúce sa motorov, ktoré pohybujú robotom v priestore, ale aj manipulačných ramien a pod. Fyzické systémy sú tiež vystavené opotrebeniu.

Softvér robota tiež vplýva na neurčitost'. Všetky interné modely sveta sú aproximáciami. Často použité reprezentácie sú pomerne hrubé a nie sú schopné obsiahnuť všetky aspekty reálneho sveta, modelujú preto len časť procesov prebiehajúcich v robote v prostredí. Algoritmické spracovanie je tiež zdrojom nepresnosti. Roboty pracujú v reálnom čase s obmedzenými výpočtovými prostriedkami. Množstvo aproximačných metód je kompromisom medzi trvaním výpočtu a rozlíšením.

1.2.1 Základné koncepty pravdepodobnosti

Nech X značí náhodnú premennú a x presnú hodnotu, ktorú môže X nadobudnúť. Ak je X diskretnou veličinou,

$$p(X = x) \tag{1.1}$$

značí pravdepodobnosť, že náhodná premenná X nadobudne hodnotu x . Funkcia p sa nazýva pravdepodobnostnou funkciou. Pravdepodobnosť je vyjadrená kladným číslom, pričom súčet diskretných pravdepodobností je vždy 1:

$$\sum_x p(X = x) = 1 \tag{1.2}$$

Pre zjednodšenie môže byť pravdepodobnosť $p(X = x)$ zapísaná len ako $p(x)$.

Ak je náhodná veličina spojitá, funkcia sa nazýva hustotou rozdelenia pravdepodobnosti. V tomto prípade platí:

$$\int p(x) dx = 1 \tag{1.3}$$

Často používanou funkciou rozdelenia pravdepodobnosti je normálne rozdelenie definované strednou hodnotou μ a rozptylom σ^2 . Toto rozdelenie popisuje Gaussova funkcia:

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{1.4}$$

Spojená pravdepodobnosť dvoch náhodných premenných X a Y je označená ako

$$p(x, y) = p(X = x \wedge Y = y) \tag{1.5}$$

Pokiaľ sú premenné X a Y nezávislé, platí $p(x, y) = p(x)p(y)$

Náhodné premenné môžu byť nosičmi informácií o iných náhodných premenných. Ak je známa hodnota y náhodnej premennej Y , je možné vyjadriť pravdepodobnosť určitého javu x premennej X s prihliadnutím na tento fakt.

$$p(x|y) = p(X = x|Y = y) \quad (1.6)$$

Zápis $p(x|y)$ značí podmienenú pravdepodobnosť. Pre $p(y) > 0$ platí:

$$p(x|y) = \frac{p(x, y)}{p(y)} \quad (1.7)$$

Ak sú náhodné premenné X a Y nezávislé, Y neposkytuje žiadnu informáciu o pravdepodobnosti hodnoty X :

$$p(x|y) = \frac{p(x, y)}{p(y)} = p(x) \quad (1.8)$$

Bayesov vzťah

Bayesova veta v teórii pravdepodobnosti udáva, ako podmienená pravdepodobnosť javu súvisí s opačnou podmienenou pravdepodobnosťou.

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} = \frac{p(y|x)p(x)}{\sum_{x'} p(y|x')p(x')dx} \quad (1.9)$$

Ak x je jav, ktorého rozdelenie pravdepodobnosti by sme chceli odvodiť z javu y , rozdelenie $p(x)$ bude označené ako a-priori rozdelenie. Rozdelenie $p(x)$ predstavuje doterajšie vedomosti o náhodnej premennej X , bez zahrnutia nových. Rozdelenie $p(y)$ predstavuje naopak nové informácie, napríklad dáta z merania senzorom.

Bayesov vzťah teda udáva spôsob výpočtu posteriórnej distribúcie $p(y|x)$ s použitím opačnej podmienenej pravdepodobnosti $p(x|y)$ s prihliadnutím na a-priori rozdelenie $p(x)$.

1.2.2 Dempster-Shaferova teória

Teória pravdepodobnosti Dempstera a Shafera[14] predstavuje generalizáciu klasickej Bayesovskej pravdepodobnosti. Na rozdiel od klasickej pravdepodobnosti umožňuje vyjadriť a pracovať s prípadom, keď je na diskretnom priestore pravdepodobnosť priradená nielen jednotlivým hodnotám ale aj množinám – rôznym kombináciám hodnôt, ktoré môže veličina nadobudnúť. Umožňuje vyjadriť spoločnú pravdepodobnosť, že veličina nadobudne niektorú z množiny hodnôt, bez toho, aby musela byť funkcia pravdepodobnosti definovaná pre všetky hodnoty.

Ako príklad je možné použiť problém klasifikácie jednoduchých objektov \triangle , \square , a \blacksquare , ktoré majú spoločné vlastnosti definované v tabuľke 1.1.

	Objekt	Výplň	Tvar	Veľkosť
A	\triangle	0	0	0
B	\square	0	1	1
C	\blacksquare	1	1	0

Tabuľka 1.1: Klasifikované objekty a ich vlastnosti

Sú dostupné senzory na meranie jednotlivých veličín, ale nie plne spoľahlivé. Porovnaním s referenčnými údajmi sa zistili nasledovné pravdepodobnosti správneho merania.

	Výplň	Tvar	Veľkosť
0	0.95	0.7	0.9
1	0.8	0.75	0.6

Tabuľka 1.2: Pravdepodobnosť správnosti nameranej hodnoty

Senzor výplne teda v 95% prípadov identifikuje správne vyplnený objekt (true positive) a v 5% prípadov pokladá vyplnený objekt za prázdny. Na základe nasledujúcich meraní je potrebné vyhodnotiť o ktorý objekt sa jedná:

Výplň	Tvar	Veľkosť
1	0	1

Tabuľka 1.3: Vlastnosti neznámeho objektu

V Dempster-Shaferovej teórii je možné tento problém vyjadriť nasledovne definovaním objektu ako náhodnej premennej $X \in \Theta$. $\Theta = A, B, C$ je rámec rozlíšenia (frame of discernment)– úplná množina vzájomne disjunktných hodnôt (hypotéz), ktoré môže táto náhodná premenná nadobudnúť. Potenčná množina 2^Θ obsahuje všetky podmnožiny Θ , vrátane Θ a prázdnej množiny \emptyset . Lubovoľná podmnožina 2^Θ môže byť jednotlivou hypotézou A , alebo množinou hypotéz, napríklad A, C .

Základné priradenie dôvery (basic belief assignment, BBA) je definované ako zobrazenie $m : 2^\Theta \mapsto \langle 0, 1 \rangle$, pričom musí platiť nasledovné:

$$\sum_{X \subseteq \Theta} m(X) = 1 \tag{1.10}$$

$$m(\emptyset) = 0$$

Toto znamená, že všetky tvrdenia z jednotlivých senzorov musia byť normalizované, aby sa zaistilo, že všetky zdroje majú rovnakú váhu. Platí teda, že základné priradenie dôvery (BBA) reprezentované funkciou m je priraduje každému prvku z 2^Θ mieru dôvery pomocou reálneho čísla z intervalu $\langle 0, 1 \rangle$ tak, že súčet týchto čísel je rovný 1. Podľa týchto podmienok nemusí byť miera dôvery priradená len atomickým hypotézam h_i ale aj

množinám hypotéz $A = h_1, \dots, h_n$ kde $A \in 2^\Theta$. Hodnota $m(A)$ je teda miera dôvery v to, že platí celá hypotéza A, pričom nevypovedá o jednotlivých zložkách množiny A. Nemusí platiť $m(B) \leq m(A)$ pre $B \subseteq A$

Merania senzorov z tabuľky 1.3 potom na základe pravdepodobnosti správnosti z tabuľky 1.2 definujú nasledovné funkcie základného priradenia dôvery:

$$\begin{aligned}
m_{vypln}(\{C\}) &= 0.8 \\
m_{vypln}(\{A, B\}) = 1 - m_{vypln}(\{C\}) &= 0.2 \\
m_{vypln}(\emptyset) &= 0 \\
m_{vypln}(\{A\}) &= 0 \\
m_{vypln}(\{B\}) &= 0 \\
m_{vypln}(\{A, C\}) &= 0 \\
m_{vypln}(\{B, C\}) &= 0 \\
m_{vypln}(\{A, B, C\}) &= 0
\end{aligned} \tag{1.11}$$

$$\begin{aligned}
m_{tvar}(\{A\}) &= 0.7 \\
m_{tvar}(\{B, C\}) = 1 - m_{tvar}(\{A\}) &= 0.3 \\
\forall x \in 2^\Theta \setminus \{\{A\}, \{B, C\}\} : m(x) &= 0
\end{aligned}$$

$$\begin{aligned}
m_{velkost}(\{B\}) &= 0.6 \\
m_{velkost}(\{A, C\}) = 1 - m_{velkost}(\{B\}) &= 0.4 \\
\forall x \in 2^\Theta \setminus \{\{B\}, \{A, C\}\} : m(x) &= 0
\end{aligned}$$

Tieto tvrdenia je možné skombinovať do jedného priradenia dôvery pomocou zlučovacieho operátora (Dempster's rule) \oplus , ktorý je definovaný nad priradeniami pravdepodobnosti m_1 a m_2 nasledovne:

$$\begin{aligned}
m_1 \oplus m_2(c) &= m_{12}(c) = \frac{1}{K} \sum_{A, B: A \cap B = C \neq \emptyset} m_1(A) \cdot m_2(B) \\
m_{12}(\emptyset) &= 0 \\
K &= 1 - \sum_{A, B: A \cap B = \emptyset} m_1(A) \cdot m_2(B) = \sum_{A, B: A \cap B \neq \emptyset} m_1(A) m_2(B)
\end{aligned} \tag{1.12}$$

hodnota K (konflikt) je indikátorom toho, do akej miery si m_1 a m_2 odporujú. V prípade, keď $K = 0$ sú zdroje úplne nezlúčiteľné. Operátor Θ je komutatívny a asociatívny[5] a teda umožňuje zlučovanie informácií z ľubovoľného počtu zdrojov.

	m_{vypln}	m_{tvar}	$m_{velkost}$	$m_{vypln} \oplus m_{tvar}$	$m_{vypln} \oplus m_{tvar}$
A		0,7		0.32	0.128
B			0.6	0.14	0.084
C	0.8			0.54	0.216
A,B	0.2				
B,C		0.3			
A,C			0.4		
K (konflikt)				0.44	0.428

Tabuľka 1.4: Dempsterov zlučovací operátor aplikovaný na dáta zo senzorov za účelom klasifikácie objektov. Nulové hodnoty základného priradenia pravdepodobnosti boli vynechané.

Tabuľka 1.4 obsahuje výsledky po aplikovaní operátora zlúčenia na priradenie dôvery definované v rov. 1.11. Vzhľadom na miery (ne)spoľahlivosti senzorov je možné usúdiť, že sa s najväčšou pravdepodobnosťou jedná o objekt C (■).

1.2.3 Kumulatívny zlučovací operátor

Predpokladajme senzor robota, ktorý rozpoznáva presne vyhradený priestor ako voľný alebo obsadený. Môže sa jednať napríklad o bunku rastrovej mapy – v našom prípade okupačnej mriežky. Tento binárny senzor vykonáva veľké množstvo meraní v pravidelných časových intervaloch. Cieľom je vyjadriť pravdepodobnosť obsadenia ako dlhodobého trendu odvodeného z množstva meraní.

Nech výstup senzoru je vyjadrený podobne ako v predchádzajúcom prípade pomocou rámca rozlíšenia $\theta = \{1, 0\}$. Senzor vráti v 99% prípadov správnu hodnotu, ale má 1% šancu na zlyhanie, keď je jeho výstup nepoužiteľný. V tomto prípade sa môže jednať o voľný aj obsadený priestor a preto je pravdepodobnosť priradená množine $\{1, 0\}$. Rovnica 1.13 definuje 2 základné priradenia dôvery pre obsadenú bunku m_o a voľnú bunku m_v :

$$\begin{aligned}
m_O(\{0\}) &= 0 \\
m_O(\{1\}) &= 0.99 \\
m_O(\{1, 0\}) &= 0.01 \\
m_V(\{0\}) &= 0.99 \\
m_V(\{1\}) &= 0 \\
m_V(\{1, 0\}) &= 0.01
\end{aligned} \tag{1.13}$$

V tomto prípade sa jedná o hromadenie informácií z meraní rovnakej veličiny v rôznych, nezávislých stavoch, pričom medzi meraniami môže vzniknúť konflikt. Pokiaľ by základné priradenie dôvery bolo dogmatické, pre ľubovoľné (a jediné) $x \in \Theta$ by platilo $m(x) = 1$ a zároveň by merania boli opačné, hondota konfliktu K v Dempsterovom operátore (rov. 1.12) by dosiahla hondotu 0, pri ktorej tento operátor nie je definovaný.

Zhlukovanie meraní z nezávislých stavov rieši Jøsangov kumulatívny operátor[14], označený ako \diamond . Pre priradenia pravdepodobnosti m_A a m_B je definovaný v dvoch prípadoch podľa priradenia pravdepodobnosti celého rámca rozlíšenia Θ .

Ak platí $m_A(\Theta) \neq 0 \vee m_B(\theta) \neq 0$:

$$\begin{aligned} m_{A \diamond B}(x_i) &= \frac{m_A(x_i)m_B(\Theta) + m_B(x_i)m_A(\Theta)}{m_A(\Theta) + m_B(\Theta) - m_A(\Theta)m_B(\Theta)} \\ m_{A \diamond B}(\Theta) &= \frac{m_A(\Theta)m_B(\Theta)}{m_A(\Theta) + m_B(\Theta) - m_A(\Theta)m_B(\Theta)} \end{aligned} \quad (1.14)$$

V opačnom prípade, keď $m_A(\Theta) = 0 \wedge m_B(\theta) = 0$:

$$\begin{aligned} m_{A \diamond B}(x_i) &= \gamma_A m_A(x_i) + \gamma_B m_B(x_i) \\ m_{A \diamond B}(\Theta) &= 0 \end{aligned} \quad (1.15)$$

Váhy γ_A a γ_B sú relatívnymi váhami, pričom platí $\gamma_A + \gamma_B = 1$. Bez predchádzajúcich informácií je vhodné hodnoty nastaviť ako $\gamma_A = \gamma_B = 0.5$ [14].

Množina hypotéz	Meranie 1 m_1	Meranie 2 m_2	Dempsterov operátor $m_1 \oplus m_2$	Kumulatívny operátor $m_1 \diamond m_2$
{0}	0.99	0.99	0.9999	0.995
{1}	0.00	0.00	0.0000	0.000
{1, 0}	0.01	0.01	0.0001	0.005

Tabuľka 1.5: Porovnanie Dempsterovho zlučovacieho operátora a Jøsangovho kumulatívneho operátora [14].

Tabuľka 1.5 porovnáva výsledok aplikovania Dempsterovho a kumulatívneho operátora v prípade rovnakých meraní. Dôsledkom je zníženie neurčitosti a konvergencia k hypotéze s najvyšším priradením pravdepodobnosti. V prípade Dempsterovho operátora je neurčitosť znížená na $m_1(\{1, 0\})m_2(\{1, 0\}) = 0.0001$. Pri zlúčení 2 meraní je zníženie o dva rády na 1/100 príliš rýchle na kumulatívnu fúziu. Očakávalo by sa zníženie o 1/2, čo je prípadom kumulatívneho operátora [14].

1.3 Reprezentácia okolia

Spôsob reprezentácie nie je ustálený, ale je predmetom výskumu a vhodnosť závisí od dostupnosti požadovaných senzorov a účelu.

Pre potreby pohybujúceho sa robota je vhodné základné rozdelenie priestoru na voľný a obsadený. Tvar celistvých objektov v exteriéri je príliš komplexný na to, aby bol vyjadrený jednoduchými geometrickými primitívami ako sú priamky, kvádre alebo gule [11]. Alternatívou je rasterizácia – vyjadrenie priestoru v podobe mriežky s rovnomerne rozloženými bunkami, kde každá bunka vyjadruje pravdepodobnosť obsadenia patričnej pozície v priestore. Tento prístup sa nazýva okupačná mriežka (occupancy grid).

Trojrozmerný priestor je možné reprezentovať v podobe 3D mriežky, napríklad v datovej štruktúre octree – OctoMap [12]. S prihliadnutím na stratu informácií o vertikálnej polohe prekážok je možné okupačnú mriežku zjednodušiť na dvojrozmernú, na rovine rovnobežnej s povrchom, na ktorom sa vozidlo pohybuje.

Okupačné mriežky boli s úspechom aplikované na rôzne účely ako plánovanie trasy [10], súbežná lokalizácia a mapovanie (SLAM) [9] a sledovanie cieľov [6]. Najčastejšie používanými senzormi sú LIDAR-y, okupačné mriežky sú ale vhodné aj pre fúziu výstupu rôznych senzorov [17].

Nech m je okupačná mriežka. Bunka mriežky je značená ako $m_{x,y}$. Nech $z_{1:t}$ je množina všetkých meraní po čas t a $x_{1:t}$ trasa prejdená robotom, vyjadrená ako sekvencia póz. Každá hodnota $m_{x,y}$ je spojená s binárnou hodnotou obsadenosti – 1 pre obsadenú bunku, 0 pre voľnú. Zápis $p(m_{x,y} = 1)$ alebo zjednodušene $p(m_{x,y})$ vyjadruje pravdepodobnosť, že bunka je obsadená. Mapovanie znamená získanie posteriórnej pravdepodobnosti

$$p(m|z_1, \dots, z_t, x_1, \dots, x_t) \quad (1.16)$$

na základe meraní a trasy. Okupačné mriežky sú ale definované na príliš veľkom obore hodnôt. Pre mriežku obsahujúcu $100x100$ buniek je počet reprezentovateľných máp 2^{10000} . Výpočet pravdepodobnosti pre každú možnosť je preto neriešiteľným problémom. Štandardný prístup rozdeľuje problém na množstvo menších problémov – určenie $p(m_{x,y}|z_{1:t}, x_{1:t})$ pre každú bunku $m_{x,y}$ [19, kap. 9.2]. Predpokladom je, že bunky sú navzájom nezávislé, posteriórna pravdepodobnosť je potom určená nasledovne:

$$p(m|z_{1:t}, x_{1:t}) = \prod_{x,y} p(m_{x,y}|z_{1:t}, x_{1:t}) \quad (1.17)$$

Algorimus 1 aplikuje Bayesov vzťah popísaný v kapitole 1.2.1 pre riešenie problému mapovania. Pre zjednodušenie bola vzťahov vynechaná trasa x , predpokladá sa, že je známa. Výsledkom je sekvenčný Bayesovský filter, ktorého aktualizácia je vyjadrený vzťahom:

$$p(m_{x,y}|z_{1:t}) = \frac{p(z_t|m_{x,y})p(m_{x,y}|z_{1:t-1})}{p(z_t|z_{1:t-1})} \quad (1.18)$$

Po aplikovaní Bayesovho vzťahu na model merania dostaneme:

$$p(z_t|m_{x,y}) = \frac{p(m_{x,y}|z_t)p(z_t)}{p(m_{x,y})} \quad (1.19)$$

$$p(m_{x,y}|z_{1:t}) = \frac{p(m_{x,y}|z_t)p(z_t)p(m_{x,y}|z_{1:t-1})}{p(m_{x,y})p(z_t|z_{1:t-1})}$$

Vyjadrením pravdepodobnosti vo forme šance (odds) – pomeru pravdepodobnosti, že binárna udalosť nastane a pravdepodobnosti, že nenastane, je možné sa delením zbaviť obtiažne vyčísliteľných výrazov $p(z_t)$ a $p(z_t|z_{1:t-1})$:

$$p(\neg m_{x,y}|z_{1:t}) = \frac{p(\neg m_{x,y}|z_t)p(z_t)p(\neg m_{x,y}|z_{1:t-1})}{p(\neg m_{x,y})p(z_t|z_{1:t-1})} \quad (1.20)$$

$$\frac{p(m_{x,y}|z_{1:t})}{p(\neg m_{x,y}|z_{1:t})} = \frac{p(z|z_t)}{1-p(z|z_t)} \frac{p(m_{x,y}|z_{1:t-1})}{1-p(m_{x,y}|z_{1:t-1})} \frac{1-p(m_{x,y})}{p(m_{x,y})}$$

Súčiny vo vzťahu je možné následne nahradiť súčtom logaritmov, čím vznikne vyjadrenie pravdepodobnosti v tvare logaritmickej šance (log odds).

$$l_{t,x,y} = \log \frac{p(m_{x,y}|z_{1:t},x_{1:t})}{1-p(m_{x,y}|z_{1:t},x_{1:t})} \quad (1.21)$$

Táto reprezentácia obmedzuje problémy s numerickou presnosťou pre pravdepodobnosti blízko k hraničným hodnotám 0 a 1. Spätný prepočet na pravdepodobnosť vyjadruje vzťah:

$$p(m_i|z_{1:t},x_{1:t}) = 1 - \frac{1}{1+e^{l_{t,x,y}}} \quad (1.22)$$

Premenná l_0 v algoritme obsahuje a-priori pravdepodobnosť, rovnako vo forme logaritmickej šance:

$$l_0 = \log \frac{1-p(m_{x,y})}{p(m_{x,y})} \quad (1.23)$$

Celý postup odvodenia sekvenčného Bayesovho filtra popísal Thrun a kolektív v kapitole 4.1.4 knihy Probabilistic Robotics[19].

Algoritmus 1: Výpočet okupačnej mriežky, na základe rekurzívneho Bayesovho odhadu. Prevzaté z [19, kap. 9.2].

```

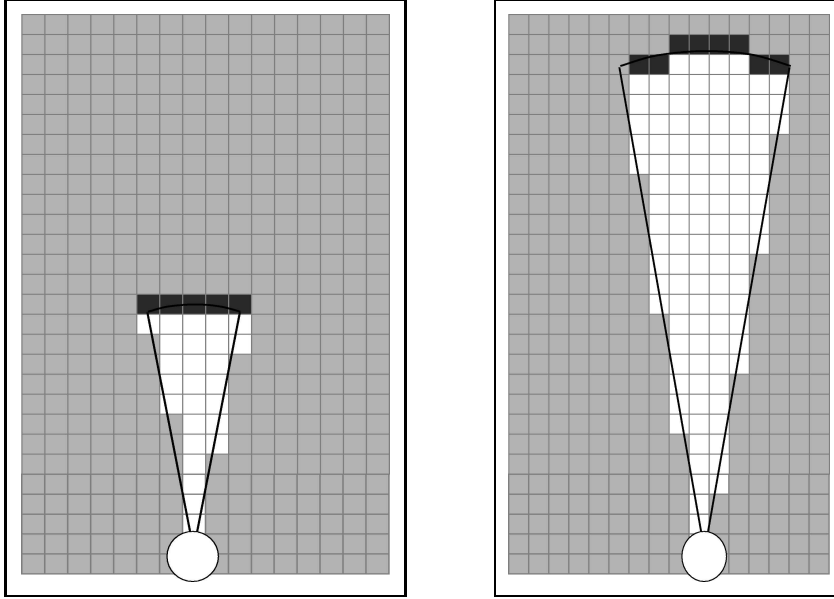
1 Function occupancy_grid_mapping( $\{l_{t-1,x,y}\}, z_t$ )
2   for all cells  $m_{x,y}$  do do
3     if  $m_{x,y}$  in perceptual field of  $z_t$  then
4        $l_{t,x,y} = l_{t-1,x,y} + \textit{inverse\_sensor\_model}(m_{x,y}, z_t) - l_0$ 
5     end
6     else
7        $l_{t,x,y} = l_{t-1,x,y}$ 
8     end
9   end
10  return  $\{l_{t,x,y}\}$ ;

```

V bunkách, ktoré spadajú do výseče merania senzoru algoritmus 1 upraví šancu obsadenosti pomocou funkcie *inverse_sensor_model*. Funkcia *inverse_sensor_model* implementuje výpočet inverznej pravdepodobnosti merania $p(m_{x,y}|z_t, x_t)$ v tvare logaritmickej šance, pričom množina $\{l_{t,x,y}\}$ reprezentuje pravdepodobnosti $p(m_{x,y}|z_t, x_t)$ pre všetky bunky s pozíciou x, y v čase t . Inverzná pravdepodobnosť je viazaná na konkrétny použitý senzor. Pri použití diaľkometerov ako sonar alebo LIDAR je reprezentovateľný guľovým výsekom, ktorý vyjadruje nepresnosť jedného merania. Obrázok 1.1 znázorňuje inverznú pravdepodobnosť meraní rôzne vzdialených prekážok.

1.3.1 Stavový priestor a viera

Dôležitým konceptom v robotike je subjektívna viera (belief). Viera odpovedá interným poznatkom robota o prostredí, ktoré nie je možné priamo zmerať. Rozlišuje sa preto viera, subjektívna pravdepodobnosť, od reálnej pravdepodobnosti stavu prostredia.



Obr. 1.1: Ukážky inverzného modelu merania pre dve rôzne vzdialené prekážky. Intenzita značí pravdepodobnosť obsadenia bunky. Prevzaté z [19, kap. 9.3].

Pravdepodobnostná robotika reprezentuje vieru pomocou podmienenej pravdepodobnosti. Viera priraduje každej hypotéze hodnotu s ohľadom na reálny stav, je to ale posteriórna pravdepodobnosť stavových premenných podmienených dostupnými dátami o prostredí.

Viera o stave premennej x_t je označená ako $bel(x_t)$:

$$bel(x_t) = p(x_t | z_{1:t}, u_{1:t}). \quad (1.24)$$

Premenná $z_{1:t}$ predstavuje všetky minulé merania od začiatku do času t a $u_{1:t}$ všetky minulé riadiace signály, ak sa robot pohybuje.

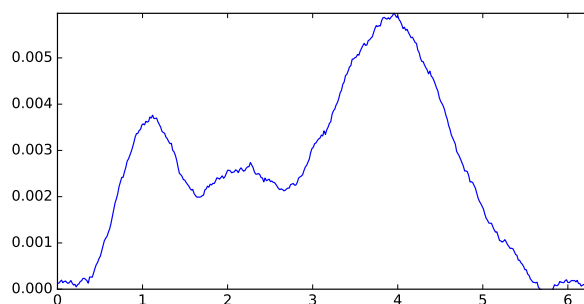
Distribúcia $bel(x_t)$ je viazaná na posledné meranie z_t . Pri filtrovaní je ale potrebné definovať aj rozdelenie $\overline{bel}(x_t)$ nasledovne:

$$\overline{bel}(x_t) = p(x_t | z_{1:t-1}, u_{1:t}). \quad (1.25)$$

Toto rozdelenie pravdepodobnosti predstavuje predpovedaný stav prostredia v čase t z predchádzajúcich meraní $z_{1:t-1}$ a riadiacich signálov $u_{1:t}$ vrátane aktuálneho, bez viazanosti na súčasné meranie z_t . Výpočet $bel(x_t)$ z $\overline{bel}(x_t)$ je korekciou – aktualizáciou subjektívnej viery na základe merania.

Časticový filter

Časticový filter patrí medzi neparametrické filtre, ktoré nie sú závislé na pevnej reprezentácii rozdelenia pravdepodobnosti napríklad v podobe Gaussovej funkcie[19]. Posteriórne rozdelenie aproximuje konečným počtom hodnôt, kde každá reprezentuje hypotézu v stavovom



Obr. 1.2: Multimodálne rozdelenie pravdepodobnosti. Módom sa označuje najčastejšie sa vyskytujúca hodnota. V tomto prípade má funkcia rozdelenia pravdepodobnosti viacero lokálnych maxim, teda je multimodálna.

priestore. Posteriórne rozdelenie nie je obmedzené žiadnym predpokladom, môže byť teda aj multimodálne, ako na obr. 1.2. Použitie časticového filtra je preto v robotike vhodné v prípadoch, keď je potrebné udržiavať množstvo hypotéz. Príkladom je lokalizácia pomocou existujúcej mapy, kde je potrebné v jednom momente reprezentovať množstvo hypotéz o aktuálnej polohe, ktorých pravdepodobnosti sa v čase upravujú podľa novo získaných meraní z okolia. Následkom tejto schopnosti sú ale pochopiteľne zvýšené požiadavky na výpočetnú zložitosť.

Časticový filter modeluje posteriórnu distribúciu v podobe množiny častíc

$$X_t = \{x_t^{[1]}, x_t^{[2]}, \dots, x_t^{[M]}\}, \quad (1.26)$$

pričom každá častica $x_t^{[m]}$ pre $1 \leq m \leq M$ je konkrétnou hypotézou o stave v čase t . Počet častíc M množiny X_t je často vysoký (tisíciky) a môže byť v čase konštantný – dopredu určený parametrom, alebo sa adaptívne mení podľa modelovanej distribúcie. Tento parameter umožňuje presne stanoviť hornú hranicu výpočetného času, čo je výhodné pre systémy ktoré bežia v reálnom čase. Vždy je potrebné urobiť kompromis medzi náročnosťou a presnosťou, ale je výhodné, že to časticové filtre priamočiaro umožňujú.

Cieľom je aproximovať posteriórne rozdelenie stavu x_t v podobe viery (belief) $bel(x_t)$, pričom pravdepodobnosť, že sa táto častica nachádza v množine častíc by mala byť priamo úmerná posteriórnej distribúcii stavu:

$$x_t^{[m]} \sim p(x_t, z_{1:t}, u_{1:t}) \quad (1.27)$$

Dôsledkom je priama úmernosť medzi hustotou častíc a hodnotou $bel(x_t)$ v ľubovoľnom regióne stavového priestoru. Vzťah 1.27 platí asymptoticky, keď sa počet častíc M blíži ∞ . V praxi sa pri konečnom počte častíc budú tieto rozdelenia líšiť, čo je možné kompenzovať počtom častíc, v závislosti od dimenzionality modelovaného priestoru.

Rovnako ako ostatné algoritmy Bayesovského filtrovania, časticový filter vytvára posteriórne rozloženie rekurzívne z predchádzajúceho kroku v čase. Keďže rozdelenie je reprezentované množinou častíc, znamená to rekurzívne vytváranie množiny X_t z množiny X_{t-1} .

Sequential importance sampling

Sekvenčné vzorkovanie podľa váhy predstavuje spôsob vytvorenia množiny častíc X_t z množiny X_{t-1} v predchádzajúcom časovom kroku. Vstupom je teda X_{t-1} spolu s aktuálnym meraním z_t . Častice $x_{t-1}^m \in x_{t-1}$ sa jednotlivito spracujú podľa algoritmu 2 zloženého z 3 krokov:

– Predikcia

Z častice $x_{t-1}^{[m]}$ pochádzajúcej z množiny X_{t-1} sa vytvorí hypotetický stav $x_t^{[m]}$. Pri tomto kroku sa vzorkuje z rozdelenia $p(x_t|u_t, x_{1:t})$, pre ktoré musí existovať algoritmus generovania vzorkov. Výsledkom je množina, ktorá časticami reprezentuje $\overline{bel}(x_t)$.

– Váhovanie

Pre každú časticu $x_t^{[m]}$ je určená váha $w_t^{[m]}$. Tento faktor slúži na začlenenie merania do množiny častíc. Váha reprezentuje pravdepodobnosť merania z_t podmienenú časticou $x_t^{[m]}$, tj. $w_t^{[m]} = p(z_t|x_t^{[m]})$. Toto rozdelenie sa nazýva dopredný model senzoru¹. Množina váhovaných častíc reprezentuje posteriórne rozdelenie $bel(x_t)$.

– Prevzorkovanie

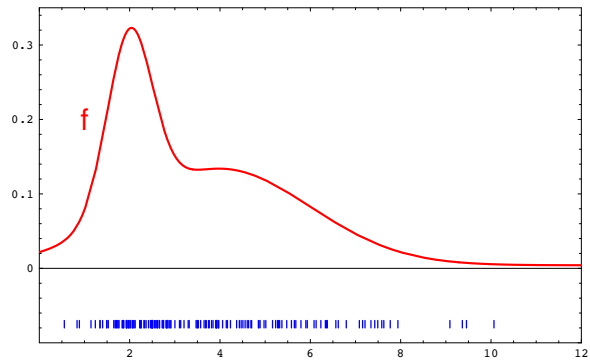
Jadro časticového filtra tvorí krok prevzorkovania. Algoritmus náhodne vyberá (s opakovaním) M častíc z aktuálnej množiny častíc \overline{X}_t v čase t . Pravdepodobnosť výberu tejto častice je úmerná priradenej váhe. Prevzorkovanie nahradí množinu častíc novou množinou, o rovnakej veľkosti. Vďaka váhovaniu sa rozdelenie zmení, pred prevzorkovaním sú častice rozdelené približne podľa $\overline{bel}(x_t)$, po prevzorkovaní podľa $bel(x_t) = \eta p(z_t|x^{[m]_t})\overline{bel}(x_t)$. Častice s malými priradenými váhami sa v novej množine nemusia vyskytnúť vôbec, pričom častice s vysokými váhami môžu byť v novej množine obsiahnuté mnoho krát.

Algoritmus 2: Časticový filter. Prevzaté z [19, kap. 4.2].

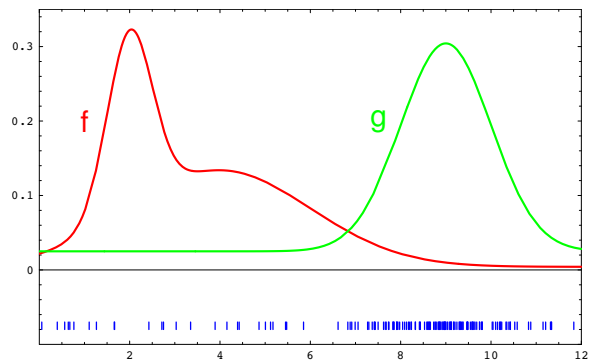
```
1 Function particle_filter( $X_{t-1}, u_t, z_t$ )
2    $\overline{X}_t = X_t = \emptyset$ 
3   for  $m = 1$  to  $M$  do do
4     | sample  $x_t^{[m]} \sim p(x_t|u_t, x_{t-1}^{[m]})$ 
5     |  $w_t^{[m]} = p(z_t|x_t^{[m]})$ 
6     |  $\overline{X}_t = \overline{X}_t + [x_t^{[m]}, w_t^{[m]}$ 
7   end
8   for  $m = 1$  to  $M$  do do
9     | draw  $i$  with probability  $w_t^{[m]}$ 
10    |  $X_t = X_t \cup \{x_t^{[i]}\}$ 
11  end
12  return  $X_t$ ;
```

Obrázok 1.3 znázorňuje krok prevzorkovania.

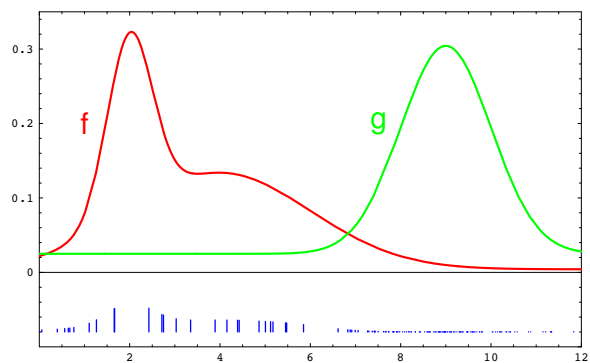
¹Opakom je inverzný model, ktorý udáva pravdepodobnosť stavu prostredia, na základe merania



(a)



(b)



(c)

Obr. 1.3: (a) Cieľové rozdelenie pravdepodobnosti f . (b) Častice sú generované podľa rozdelenia g . (c) Rozdelenie f reprezentované časticami s priradenými váhami $f(x)/g(x)$. Prevzaté z [19, kap. 4.2].

Cieľom je získať funkciu rozdelenia pravdepodobnosti f . Pokiaľ nie je možné priamo získať vzorku z tejto cieľovej distribúcie, častice sa navzorkujú z iného, predpokladaného rozdelenia g . V jednom časovom kroku (cykle) časticového filtra rozdelenie f odpovedá viere $bel(x_t)$ a g predpovedanému rozdeleniu $\overline{bel}(x_t)$. Pre funkciu g musí platiť, že pravdepodobnosť vygenerovania vzorku x z predpokladaného rozdelenia $g(x)$ musí byť nenulová pre ľubovoľnú vzorku, ktorá spadá do rozdelenia f :

$$f(x) > 0 \implies g(x) > 0 \quad (1.28)$$

Pre pokrytie rozdielu medzi medzi f a g sú častice $x^{[m]}$ váhované podielom týchto rozdelení:

$$w^{[m]} = \frac{f(x^{[m]})}{g(x^{[m]})} \quad (1.29)$$

Kapitola 2

Detekcia pohybujúcich sa objektov

Táto kapitola sa venuje návrhu detektora pohybu na základe čiastového filtra na mriežke, vychádzajúc zo systému pre mapovanie a sledovanie pohybu na mriežke[18].

2.1 KITTI Visual Odometry

Pre testovanie bola zvolená databáza KITTI Visual Odometry[8], ktorá obsahuje dáta z rôznych senzorov na špeciálne upravenom osobnom aute. Pre detekciu pohybu je atraktívny predovšetkým LASER-vý diaľkomer (LiDAR) Velodyne HDL-64E a kamery Point Grey Flea 2 v stereo usporiadaní. Okupačnú mriežku, ktorá slúži ako základná reprezentácia okolia robota v navrhovanom systéme, je možné získať aj použitím stereovízie[20]. V tejto práci však bol ako zdroj sensorických dát zvolený Velodyne. Výhodou je, že pri umiestení na streche vozidla poskytuje 360° pohľad na okolie, na rozdiel od kamier s omedzeným zorným uhlom.

Velodyne HDL-64E je laserovým diaľkomerom, ktorý dokáže v jednom skene (rotácii o 360°) snímať scénu v 64 uhloch. Oproti planárnym LiDAR-om, ktoré snímajú len v jednej rovine umožňuje nasnímať trojrozmerný priestor. Tabuľka 2.1 obsahuje základné parametre diaľkomeru HDL-64E.

Voľba testovacej databázy KITTI bola zohľadnená pri návrhu systému. Táto databáza obsahuje 11 sekvencií s odtupnými pózami vozidla (ground truth), ktoré boli v riešení využité. Zároveň je dôležité, že sken LiDAR-om neprebíha v jednom okamihu, ako je to prípade kamier, ale postupným meraním v uhlových výsekoch. Pri pohybe vozidla je následkom tohto faktu skreslenie, ktoré ale bolo v tejto databáze vykompenzované.

Počet kanálov	64
Dosah	120m
Horizontálne zorné pole	360°
Vertikálne zorné pole	26.9°
Horizontálne uhlové rozlíšenie	0.08°
Vertikálne uhlové rozlíšenie	0.4°
Presnosť	2cm
Frekvencia	5 Hz – 20 Hz

Tabuľka 2.1: Parametre senzoru Velodyne HDL-64E[4]



Obr. 2.1: Polárna okupačná mriežka. Prevzaté z [11].

2.2 Segmentácia zeme

Sekvencie databázy KITTI obsahujú všetky body zo senzoru Velodyne. Za účelom vytvorenia okupačnej mriežky popísanej v nasledujúcej kapitole je ale nutné najprv rozlíšiť body, ktoré patria objektom (prekážkam) a body, ktoré ležia na zemi a nie sú z hľadiska detekcie pohybu zaujímavé.

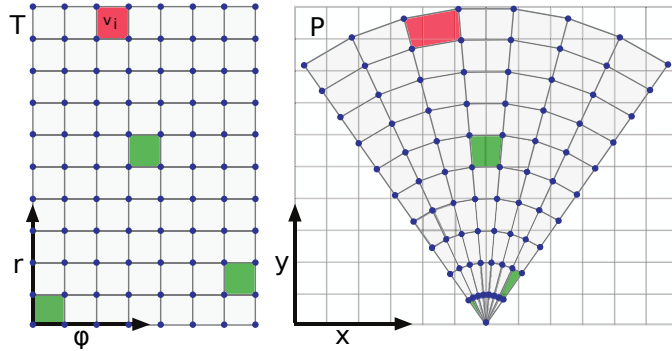
Mračná bodov zo sekvencií boli preto pred použitím anotované pomocou implementácie detektora zeme založeného na metóde Loopy Belief Propagation[21]. Detekcia pohybu sa potom vykonáva len na mračne, ktoré obsahuje body prekážok – body nad úrovňou zeme.

2.3 Lokálna mapa

Aby bolo možné sledovať pohyb, pre vytvorenie aktuálneho prostredia robota sa nevyužíva metóda rekurzívnej aktualizácie okupačnej mriežky v čase popísaná v kapitole 1.3. Namiesto toho je využitý fakt, že LiDAR Velodyne poskytuje v jednom skene informácie o okolí v okruhu 360°, na rozdiel od sonaru alebo radaru využívajúceho len 1 lúč na meranie. Ako podklad pre detekciu pohybu je vo výsledku použitý inverzný model senzoru Velodyne, ktorý obsahuje merania počas jedného otočenia – celý 360°sken okolia. Tento model bude ďalej označovaný ako lokálna okupačná mriežka (mapa).

Väčšina dostupných riešení pre tvorbu inverzneho modelu senzoru predpokladá ako vstup planárny sken dialkometerom, ktorý obsahuje len jednu hodnotu pre každý uhlový výsek, ktorý obsahuje meranie. Sensory Velodyne HDL-64E (kap. 2.1) majú ale viacero kanálov, ktoré pokrývajú aj vertikálne zorné pole. Do jedného horizontálneho uhlového výseku teda môže spadať viacero meraní (bodov). Pre zlúčenie týchto informácií v rámci uhlového výseku by bolo možné použiť Bayesov vzťah z kapitoly 1.2.1, body boli ale získané meraním pod rôznymi vertikálnymi uhlami a tak by nemali byť považované za konfliktné medzi sebou. Do inverzneho modelu lokalnej mapy su teda integrované nezávisle.

Jedným zo spôsobov vytvorenia inverzneho modelu je priame zakreslenie do karteziánskeho priestoru pomocou algoritmov ktoré pre jeden bod merania určia ovplyvnenú oblasť. Alternatívnym prístupom je prvotná reprezentácia inverzneho modelu v podobe mriežky v polárnych súradniciach [11] ako na obrázku 2.1). Riadky matice reprezentujú uhlové výseky korešpondujúce s meraniami v skene, stĺpce vzdialenosť od senzoru. Každé meranie potom spadá len do jedného riadku matice a ie je nutné riešiť rozptyl so vzdialenosťou od senzoru. voľný priestor je značený hodnotou 0, obsadený hodnotou 1. Merania z rôznych kanálov (v rôznych uhloch od horizontálnej roviny) môžu spadať do rovnakého riadku



Obr. 2.2: Transformácia z polárneho priestoru na karteziánsky. Prevzaté z [11].

matice, pričom viacnásobné zásahy do rovnakej bunky už nastavenú hodnotu nemenia. Posledná nenulová hodnota v riadku, najvzdialenejší nameraný bod, sa považuje za hraničné meranie. Bunky za touto hodnotou sa považujú za nezmeraný priestor, s rovnakou pravdepodobnosťou obsadenosti a voľnosti.

Obr. 2.2 znázorňuje následnú transformáciu polarnej reprezentácie okolia na karteziánsku pomocou algoritmu pre transformáciu obrazu s využitím interpolácie.

Tento model je značne zjednodušený, ale umožňuje využiť trojrozmerné mračno bodov. Zlepšením by mohla byť reprezentácia nameraného bodu Gaussovou funkciou s rozptylom narastajúcim so vzdialenosťou. Pri predpoklade celistvosti objektov od zeme do výšky merania by sa tiež mohli zahodiť ďalšie merania, ktoré sú v zákryte.

2.4 Časticový filter na okupačnej mriežke

Danescu a kolektív. popisuje použitie multimodality časticového filtra pre sledovanie objektov na lokálnej mriežke [7]. Častice sú reprezentované množinou

$$S = \{p_i \mid p_i = (x_i, y_i, v_{x_i}, v_{y_i}, a_i), i = 1, \dots, N_s\} \quad (2.1)$$

kde každá častica p_i je reprezentovaná spojitou pozíciou v priestore, ktorá je transformovateľná na pozíciu v mriežke. Bunku, do ktorej častica spadá je možné nájsť diskretizáciou v rozsahu riadkov a stĺpcov lokálnej mriežky. Vektor (v_{x_i}, v_{y_i}) určuje aktuálnu rýchlosť častice a parameter a_i vek častice od jej vytvorenia.

Častica v bunke vyjadruje hypotézu toho, že bunka je obsadená a dochádza na nej k pohybu s vektorom rýchlosti rovným vektoru častice. Veľké množstvo častíc v bunke značí veľkú podporu pre hypotézu obsadenosti. Málo častíc značí podporu pre hypotézu voľnej bunky.

Nech C značí bunku mriežky ako množinu $C \subset \{(x, y) \mid x, y \in \mathbb{R}\}$ definujúcu ohraničený priestor. Celkový počet častíc bunky nie je konštantný, ale závisí od zaplnenia scény – od množstva obsadených buniek mriežky. Jedným z parametrov systému je ale maximálny povolený počet častíc v jednej bunke N_C . Pomer medzi počtom častíc, ktoré sa nachádzajú v bunke C a limitom udáva pravdepodobnosť hypotézy obsadenosti o_C . Rozdiel medzi

limitom N_C a reálnym počtom častíc v bunke naopak značí pravdepodobnosť hypotézy v_C , hypotézy že bunka je voľná.

$$p(o_C) = \frac{|\{p_i \in S | (x_i, y_i) \in C\}|}{N_C} \quad (2.2)$$

Nastavenie limitu N_C je kompromisom medzi presnosťou reprezentácie a výpočtovou náročnosťou.

Počet dimenzií Častice by bolo možné zväčšiť a modelovať napríklad pohyb po kružnici, táto informácia ale pre detekciu pohybu bez sledovania objektov nie je potrebná. Vytváraním nových častíc je časticový filter schopný prispôbiť sa zmene smeru pohybujúceho sa objektu. Zvyšovanie počtu dimenzií by viedlo k nutnosti použiť väčší počet častíc, pričom pri zachovaní presnosti potrebný počet častíc stúpa s každým rozmerom exponenciálne.

Ak predpokladáme, že bunka C pokrýva jediný objekt, rýchlosť tohoto objektu je aproximovateľná ako priemerná rýchlosť častíc patriacich bunke.

$$(v_{x_C}, v_{y_C}) = \frac{\sum_{p_i \in S} (v_{x_i}, v_{y_i})}{|\{p_i \in S | (x_i, y_i) \in C\}|} \quad (2.3)$$

Populácia častíc je ale schopná popísať aj stav, v ktorom sa v bunke nachádzajú dva rôzne sa pohybujúce objekty.

Časticový filter je teda dostatočne reprezentatívny pre rozdelenie pravdepodobnosti obsadenia a rýchlosti na jednotlivých bunkách lokálnej mriežky. Pre jedinou bunku je možné popísať viacero hypotéz súčasne a neurčitost obsadenia bunky je určená počtom častíc, ktoré sa v nej nachádzajú. V nasledujúcich kapitolách je rozvinutý celý algoritmus vytvárania, aktualizácie a zahadzovania častíc tak, aby efektívne reprezentovali pohyb v okupačnej mriežke.

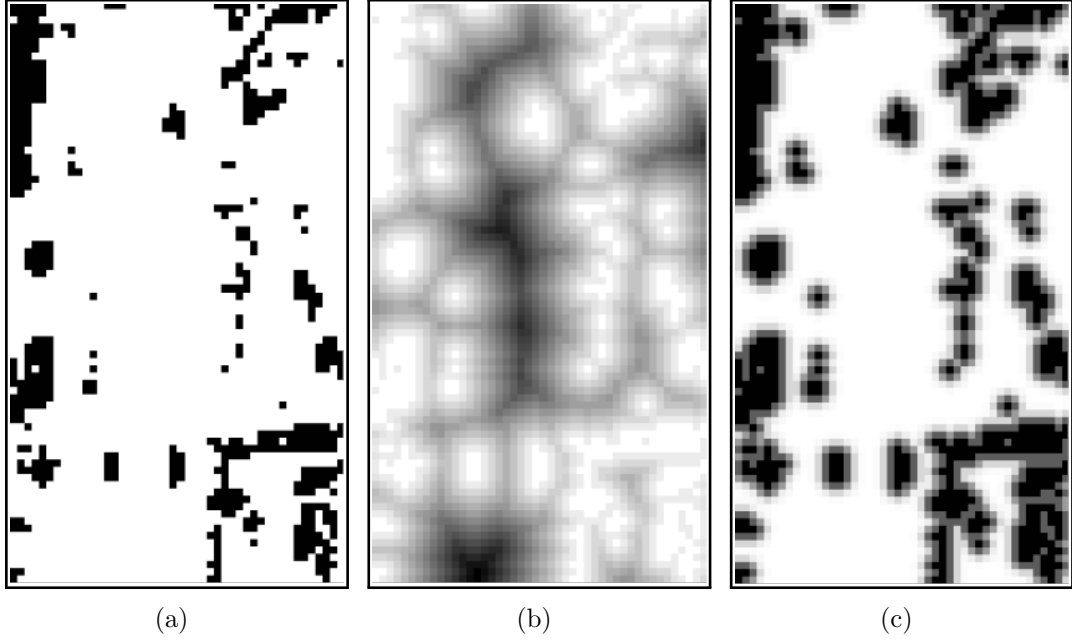
2.4.1 Predikcia

Predikcia predstavuje úpravu aktuálnych pozícií častíc na základe vektoru rýchlosti. V každom kroku sa podľa aktuálnej pózy robota voči počiatku jazdy tiež upraví transformačná matica udávajúca počiatok súradnicovej sústavy lokálnej mriežky. Všetky častice, ktoré sa dostali mimo rozsah posunutej lokálnej mriežky sú okamžite zahodené.

2.4.2 Váhy častíc

Častice v tomto prípade nie je možné váhovať priamo pomocou pravdepodobnosti nameňovania danej pozície alebo rýchlosti. Jedinou dostupnou informáciou sú pravdepodobnosti obsadenia buniek okupačnej mriežky z inverzného modelu senzoru.

Ako metriku, či častica vyhovuje meraniam, je možné použiť vzdialenosť od najbližšej obsadenej bunky. Aby nebolo nutné pre každú časticu zvlášť prehľadávať okupačnú mriežku, použije sa predspracovanie v podobe vzdialenostnej transformácie. Následne sa vzdialenosť od najbližšej obsadenej bunky použije ako vstup pre Gaussovu funkciu (bez normalizácie). Rozptyl je parametrom systému a reprezentuje toleranciu voči nepresnosti merania a chaotickému pohybu objektov. Častice ktoré sa dostanú mimo obsadenú bunku sa takto



Obr. 2.3: (a) prahovaná okupačná mriežka, (b) vzdialenosť k najbližšej obsadenej bunke, (c) Gaussova funkcia aplikovaná na vzdialenosť

okamžite nezahodia, ale majú určitú šancu prežiť vhladom na vzdialenosť k najbližšej obsadenej bunke. Obr. 2.3 zobrazuje mriežku po aplikovaní týchto operácií.

Nech funkcia `distance_to_occupied` vracia vzdialenosť k najbližšej obsadenej bunke, `distance_to_free` vzdialenosť k najbližšej voľnej bunke a funkcia `gauss` je spomenutá Gaussova funkcia bez normalizácie, centrovaná na hodnote 0 so štandardnou odchýlkou μ_{dist} . o_C značí hypotézu obsadenosti bunky C .

Výsledná podmienená pravdepodobnosť

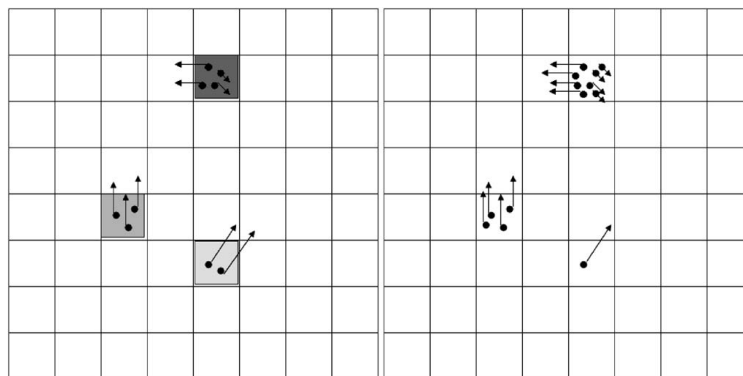
$$\begin{aligned}
 p(z_C|o_C) &= \text{gauss}(\text{distance_to_occupied}(C), \mu_{dist}) \\
 p(z_C|v_C) &= \text{gauss}(\text{distance_to_free}(C), \mu_{dist})
 \end{aligned}
 \tag{2.4}$$

je použitá ako váha, na základe ktorej je určená posteriórna pravdepodobnosť hypotézy obsadenosti[7]:

$$p(o_C|z_C) = \frac{p(z_C|o_C)p(o_C)}{p(z_C)}
 \tag{2.5}$$

Normalizačný člen Bayesovho vzťahu je daný ako:

$$\begin{aligned}
 p(z_C) &= p(z_C|o_C)p(o_C) + p(z_C|v_C)p(v_C) \\
 &= p(z_C|o_C)p(o_C) + p(z_C|v_C)(1 - p(o_C))
 \end{aligned}
 \tag{2.6}$$



Obr. 2.4: Častice pred a po prevzorkovaní. Tmavšie bunky majú vyššiu pravdepodobnosť obsadenosti. Prevzaté z [7].

2.4.3 Prevzorkovanie

Štandardný algoritmus prevzorkovania by vybral konečný počet častíc z predchádzajúcej populácie s pravdepodobnosťou výberu úmernou váhe častice. V navrhovanom systéme ale počet častíc nie je konštantný (kap. 2.4). Riešením je spracovanie každej častice nezávisle, pričom váha priradená častici určuje koľkokrát (štatisticky) bude častica replikovaná. Hodnota väčšia ako 1 značí rozmnoženie častice a hodnota menšia ako 1 šancu, že sa zahodí.

Keďže nie je možné váhu určiť na základe pohybového vektoru, všetky častice spadajúce do jednej bunky C budú váhované rovanko. Faktor replikácie častíc bunky je určený na základe posteriórnej pravdepodobnosti z predchádzajúcej kapitoly, aktuálneho počtu častíc v bunke a parametru N_C , ktorý limituje maximálny počet častíc v bunke.

$$f_C = \frac{p(o_C|z_C) N_C}{|\{p_i \in S | (x_i, y_i) \in C\}|} \quad (2.7)$$

Prevzorkovanie ilustruje obrázok 2.4, zobrazené sú častice v mriežke pred a po prevzorkovaní.

2.4.4 Inicializácia

Krok prevzorkovania síce nové častice vytvára, ale len na základe už existujúcich. Ak sa na okupačnej mriežke objaví nová zaplnená bunka, do ktorej sa v kroku predikcie nepresunuli častice, alebo je ich málo. Množina častíc prislúchajúca tejto bunke musí byť inicializovaná. Z pravdepodobnostného hľadiska je stav každej bunky okamžite po začatí sledovania neznámy. Znamená to rovnakú pravdepodobnosť pre to, že je bunka voľná aj obsadená. Vytvorenie náhodných častíc pre dosiahnutie polovice limitu na bunku by ale znamenalo neustále zahlcovanie mriežky časticami aj miestach, kde nie sú žiadne objekty pre sledovanie. Nové častice sú teda vždy generované na bunkách mriežky, kde je pravdepodobnosť obsadenia bunky vysoká.

Nové častice sú generované v strede bunky, pričom vektor rýchlosti uvedený v polárnych súradniciach sa vyberá náhodne s orientáciou v rozmedzí 0° - 360° . Veľkosť vektoru rýchlosti má hornú hranicu, ktorá je jedným z parametrov systému. Určitá časť generovaných častíc je inicializovaná s veľkosťou rýchlosti rovnej 0, čo zrychľuje konvergenciu pre statické bunky[18]. Pomer častíc vytvorených ako statických je parametrom systému.

Inicializáciou je zaručený stály prísun nových hypotéz, pričom častice, ktoré nereprezentujú reálne pohyby sa zahodia buď v procese prevzorkovania, keď sa dostanú mimo mriežku, alebo pri prekročení maximálneho počtu na bunku.

2.5 Model prostredia

Prostredie robota je charakterizované pomocou Dempster-Shaferovej teórie popísanej v kapitole 1.2.2. K rozlíšeniu statických a dynamických buniek, rovnako ako prázdneho priestoru je použitý rámec rozlíšenia $\Theta = \{F, S, D\}$, kde F je hypotéza neobsadenej bunky, S značí statický a D dynamický objekt. Tieto hypotézy sú priradené každej bunke na súradniciach i, j v čase t . Základné priradenie dôvery bude teda ďalej označované ako napr. $p^{i,j,t}(\{S, D\})$.

Lokálna mriežka a časticový filter sú chápané ako dva rozdielne zdroje informácií. Napriek tomu, že vstupom časticového filtra je okupačná mriežka, tieto dva kanály vypovedajú o odlišných hypotézach. Na základe okupačnej mriežky je určené základné priradenie pravdepodobnosti pre hypotézy $\{F\}$ a $\{S, D\}$, tj. nerozlišuje sa medzi statickou a dynamickou bunkou. Časticový filter naopak priraduje pravdepodobnosť hypotézam $\{S\}$ a $\{D\}$.

Základné priradenie pravdepodobnosti rozlišujúce voľné a obsadené bunky je dané vzťahom:

$$\begin{aligned} m_o^{x,y,t}(\{F\}) &= 1 - g_{x,y} \\ m_o^{x,y,t}(\{S, D\}) &= g_{x,y} \\ \forall x \in 2^\Theta \setminus \{\{F\}, \{S, D\}\} : m_o^{x,y,t}(x) &= 0 \end{aligned} \tag{2.8}$$

pričom $g_{x,y}$ reprezentuje bunku okupačnej mriežky na súradniciach x, y (pravdepodobnosť obsadenia).

2.5.1 Pohyb

Nech $P^{x,y,t}$ je množina častíc časticového filtra patriaca do bunky okupačnej mriežky x, y .

Táto množina je rozdelená na statickú a dynamickú množinu P_s a P_d podľa prahu rýchlosti v_{static} .

$$\begin{aligned} P_s^{x,y,t} &= \{p_i \mid p_i \in P^{x,y,t}, vel(p_i) < v_{static}\} \\ P_d^{x,y,t} &= \{p_i \mid p_i \in P^{x,y,t}, vel(p_i) \geq v_{static}\} \end{aligned} \tag{2.9}$$

Veľkosť množiny $P^{x,y,t}$ je limitovaná maximálnym počtom častíc na bunku N_{max} . Tento fakt je možné využiť na normalizáciu priradenia dôvery, ktoré rozlišuje medzi statickými a

dynamickými bunkami:

$$\begin{aligned}
 m_d^{x,y,t}(\{S\}) &= \frac{|P_s^{x,y,t}|}{N_{max}} \\
 m_d^{x,y,t}(\{D\}) &= c^{x,y,t} \frac{|P_d^{x,y,t}|}{N_{max}} \\
 m_d^{x,y,t}(\{S, D\}) &= 1 - m_d^{x,y,t}(\{S\}) - m_d^{x,y,t}(\{D\})
 \end{aligned} \tag{2.10}$$

$$\forall x \in 2^\Theta \setminus \{\{S\}, \{D\}, \{S, D\}\} : m_o^{x,y,t}(x) = 0$$

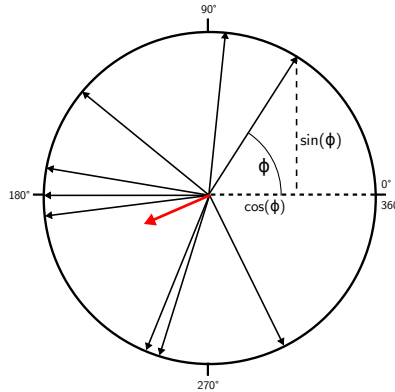
Množina dynamických častíc $P_d^{x,y,t}$ reprezentuje funkciu prevdepodobnosti $p(\phi, R)$ s argumentami odpovedajucimi vektoru rýchlosti častice prevedeného do polárnych súradníc – uhlu a veľkosti rýchlosti. Predpokladá sa, že ak sa v bunke práve nachádza pohybujúci sa objekt, vektory rýchlosti častíc budú menej rozptýlené ako v prípade statickej bunky. Ako miera konvergencie časticového filtra na pohybujúci sa objekt v danej bunke ($c^{x,y,t}$) je použitá štandardná odchýlka orientácie[18]:

$$c^{x,y,t} = 1 - \frac{\sigma^{x,y,t}}{\sigma_{max}^t} \tag{2.11}$$

Štandardná odchýlka cyklickej veličiny

Orientácia je ale cyklická veličina, a teda výpočet štandardnej odchýlky podľa vzorca $\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (\phi_i - \bar{\phi})^2}$ nie je vhodný.

Určenie štandardnej odchýlky je závislé na výpočte strednej hodnoty. V prípade dvoch uhlov, 358° a 30° je stredná hodnota vypočítaná bez prihľadnutia na cyklickosť ako $E(\phi) = \frac{1}{N} \sum_{i=1}^N \phi_i = 194^\circ$. Táto hodnota je očividne nesprávna.



Obr. 2.5: Stredná hodnota uhlov pomocou jednotkovej kružnice

Riešením je vyjadrenie uhlu ako bodu na jednotkovej kružnici v euklidovských súradniciach, ktoré cyklické nie sú, znázornené na obr. 2.5.

$$\begin{aligned}\bar{\phi} &= \text{atan2} \left(\frac{1}{N} \sum_{i=1}^N \sin \phi_i, \frac{1}{N} \sum_{i=1}^N \cos \phi_i \right) = \text{atan2} \left(\sum_{i=1}^N \sin \phi_i, \sum_{i=1}^N \cos \phi_i \right) \\ R &= \left\| \left[\frac{1}{N} \sum_{i=1}^N \sin \phi_i, \frac{1}{N} \sum_{i=1}^N \cos \phi_i \right] \right\|\end{aligned}\quad (2.12)$$

Funkcia atan2 je na rozdiel od arctan binárna a na základe znamienok argumentov zaraďuje uhol do patričného kvadrantu. Veľkosť priemerného vektora R v tomto prípade vyjadruje koncentráciu rozloženia. Polomer R je možné previesť na cyklickú štandardnú odchýlku[15] vzťahom:

$$\sigma_c = \sqrt{-2 \ln(R)} \quad (2.13)$$

Štandardná odchýlka ale nie je zhora ohraničená a preto je nemožné ustanoviť konštantnú maximálnu hodnotu σ_{max} pre výpočet konvergenčného faktoru $c_{x,y}$ vo vzťahu 2.11. Jedným z riešení by bolo určenie σ_{max} ako maximálnej hodnoty $\max_{i,j} \{\sigma^{i,j,t}\}$ na aktuálnej okupačnej mriežke v čase t . To by ale znamenalo, že priradenie dôvery pre hypotézu pohybu by závislo od ostatných buniek, takže by sa táto hodnota mohla meniť aj v prípade stabilne sa pohybujúceho objektu.

Lepším riešením je priame použitie vzdialenosti od stredu jednotkovej kružnice R , ktoré sa pohybuje v intervale $< 0, 1 >$. Hodnota σ_{max}^t je teda v rovnici 2.11 stanovená ako konštanta $\sigma_{max}^t = 1$ a aproximácia štandardnej odchýlky je jednoducho určená ako:

$$\tilde{\sigma}^{x,y,t} = 1 - R \approx \sigma_c \quad (2.14)$$

2.5.2 Fúzia okupačnej mriežky a pohybu

Fúzia okupačnej mriežky a výstupu z časticového filtra v podobe základných priradení pravdepodobnosti $m_o^{x,y,t}$ a $m_d^{x,y,t}$ je zlúčená do jednej domnienky $m^{x,y,t}$ pomocou Dempsterovho zlučovacieho operátora (kapitola 1.2.2):

$$m^{x,y,t}(A) = m_o^{x,y,t}(A) \oplus m_d^{x,y,t}(A) \quad (2.15)$$

pre $\forall A \subseteq \Theta, A \neq \emptyset$

Medzi pármí hypotéz $F - S$ a $F - D$ môže dôjsť ku konfliktu, v tomto prípade ale výsledok nie je normalizovaný. Konflikt je namiesto toho priradený hypotéze prázdneho priestoru, pretože údaje okupačnej mriežky sú „bližšie” surovým meraniam zo senzoru[18].

$$m^{x,y,t}(F) = K = m_o^{x,y,t}(F) m_d^{x,y,t}(S) + m_o^{x,y,t}(F) m_d^{x,y,t}(D) \quad (2.16)$$

2.5.3 Filtrovanie v čase

Aktuálna reprezentácia v podobe $m^{x,y,t}$ je postupne filtrovaná pre vytvorenie pohľadu na prostredie, ktorý zlučuje informácie z viacerých skenov okolia počas jazdy. Výsledná reprezentácia je v podobe mriežky na absolútnych súradniciach, ktorá je zarovnaná na osi globálnej suradnicovej sústavy (world coordinates). Na rozdiel od predchádzajúcej sekcie, kde boli v rámci Dempster-Shaferovej teórie zlučované nezávislé hypotézy za účelom vytvorenia logickej kombinácie, na kumulovanie základného priradenia pravdepodobnosti v čase je použitý Jøsangov operátor (Kap. 1.2.3).

Vykonávať fúziu hypotézy dynamickej bunky $m^{x,y,t}(\{D\})$ v čase nemá zmysel, takže toto priradenie dôvery kumulované nie je. V každom kroku je ale aktualizované.

Pri transformácii aktuálnej reprezentácie do globálnej mriežky je použité 2D prevzorkovanie pre kompenzáciu rotácie. Pri zlúčení máp sa predpokladá, že lokálna mriežka bude je rovnobežná s osami X,Y rovnako ako globálna. V prípade stúpania alebo klesania by došlo ku skresleniu.

Kapitola 3

Implementácia

Systém bol implementovaný s využitím *Robotického Operačného Systému (ROS)*, z dôvodu širokej škály dostupných nástrojov a implementovaných algoritmov z oblasti robotiky. Dôležitým je komunikačný systém popísaný v časti 3.3, vďaka čomu bolo možné rozdeliť riešenie na oddelené uzly (moduly). Toto oddelenie uľahčilo a zrýchlilo vývoj a testovanie prototypov, pretože nebolo nutné opakovane zostavovať celý spustiteľný súbor pri malých zmenách. V prípade samostaných uzlov je nutné prekladať len ten, ktorý sa práve vyvíjal. Zároveň to umožnilo implementovať uzly v rôznych jazykoch.

Ďalšou praktickou súčasťou ROS-u je prostriedok *tf* pre reprezentáciu súradnicových sústav, transformácií a ich zdieľanie medzi uzlami, popísaný v časti 3.5.

Kapitola 3.7 vysvetľuje vizualizačný nástroj *rviz*, ktorý bol veľmi užitočný pre základný prehľad o prostredí robota, senzorických dátach zasielaných medzi uzlami a overenie funkčnosti systému.

3.1 Robotický operačný systém

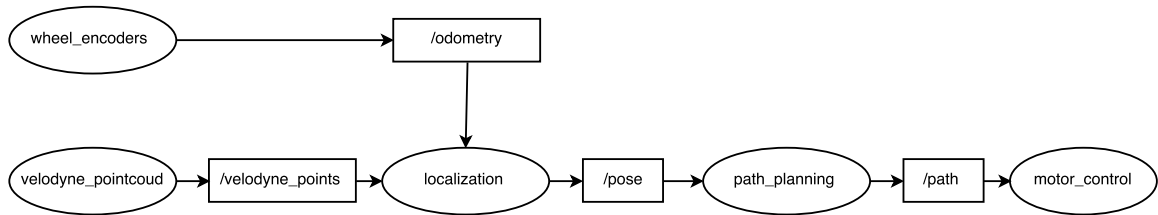
„ROS je open-source, meta-operačný systém pre roboty. Poskytuje služby, ktoré je možné očakávať od operačného systému vrátane abstrakcie hardvéru, nízkoúrovňovej obsluhy zariadení, implementácie bežne používanej funkcionality, zasielanie správ medzi procesmi a správu balíčkov. Tiež poskytuje nástroje a knižnice pre získanie, preklad, vývoj a spustenie kódu na viacerých počítačoch.[2]“

ROS nenahradzuje tradičný operačný systém ale predstavuje vrstvu medzi operačným systémom a aplikáciami – middleware. V dobe písania tejto práce bol ROS spustiteľný na Unixových platformách, primárne Ubuntu a Mac Os X.

3.2 Uzly

Uzol je procesom, ktorý vykonáva nejaký výpočet. Uzly sú usporiadané do grafu a komunikujú spolu použitím správ. Cieľom je vytvárať malé uzly, ktoré plnia jednu úlohu. Robotický systém je spravidla zložený z množstva uzlov. Jeden uzol môže napríklad obsuhovať LASER-ový diaľkomer a publikovať merania v rámci systému ROS, ďalší uzol vykonávať lokalizáciu, tretí plánovanie trasy, atď. Obrázok 3.1 znázorňuje takýto systém.

Použitie uzlov poskytuje niekoľko výhod ako zníženie zložitosti kódu abstrakciou a zpúzdrením v porovnaní s monolitickým systémom. Implementačné detaily sú skryté, uzol je



Obr. 3.1: Príklad hypotetického grafu v Robotickom operačnom systéme. Oválne bloky reprezentujú uzly, hranaté bloky predstavujú kanály (témy) správ.

popísaný aplikačným rozhraním s presne definovanými parametrami, čo umožňuje nahradenie uzlu alternatívnou implementáciou s rovnakým rozhraním.

Všetky uzly sú v grafe unikátne identifikované názvom a typom. Vytvárajú sa použitím klientskej knižnice, napr. *roscpp* alebo *rospy*.

3.3 Správy

Uzly medzi sebou komunikujú publikovaním správ v pomenovaných témach (angl. topics) [3]. Témy slúžia na kategorizáciu správ podľa zamerania a typu. Každý uzol sa môže prihlásiť do niekoľkých tém ako odberateľ, čo znamená, že mu počas jeho behu bude zaslaná každá správa v danej téme. Zároveň môže uzol pôsobiť ako zdroj správ, kde informácie zdieľa publikovaním do niektorej/ých tém. Jedná sa o návrhový vzor *publish-subscribe*¹.

Správy majú vopred určený formát, pričom je možné definovať vlastné typy pomocou doménovo špecifického jazyka. ROS obsahuje nástroje pre automatické generovanie serializačných a deserializačných rutín pre niekoľko jazykov ako C++, Python a Lisp, a teda ich nie je nutné implementovať manuálne pre každý typ. V mnohých prípadoch sa dajú využiť štandardné typy správ definované v niektorom z balíkov. Výsledkom je jednoduchá znovupoužiteľnosť už implementovaných uzlov a možnosť zostaviť rozsiahly systém z existujúcich komponent. Snahou bolo teda čo najväčšie využitie existujúcich typov správ, čo sa aj podarilo dosiahnuť, s výnimkou správy obsahujúcej lokálne prostredie, ktorá obsahuje základné priradenie dôvery v rámci Dempster-Shaferovej teórie (kap. 2.5.2).

3.4 roslaunch

Roslaunch je nástrojom pre automatizáciu spustenia a organizácie uzlov v systéme ROS. Vstupom je jeden, alebo viacero konfiguračných XML súborov, ktoré určujú parametre a uzly, ktoré sa majú spustiť. Má tiež podporu pre spustenie systému rozdeleného na niekoľko strojov (Správy môžu byť prenášané aj cez sieť).

V tejto práci roslaunch umožnil rýchle spustenie nakonfigurovaného systému bez nutnosti spúšťať každý uzol zvlášť, testovanie rôznych parametrov zmenou konfigurácie a rýchlu iteráciu vývoja jednotlivých uzlov.

3.5 Geometrické transformácie

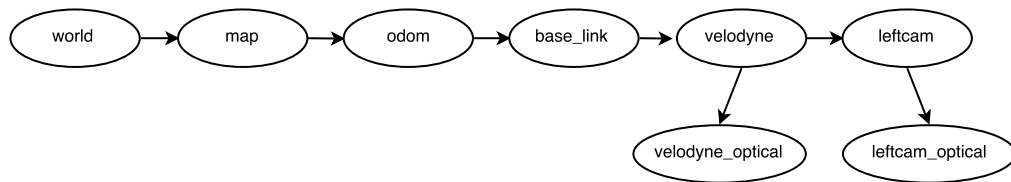
Robotický systém typicky používa množstvo trojrozmerných súradnicových sústav, ktoré sa v čase menia. Balíkmi nástrojov a knižníc, ktoré túto funkcionality implementujú sú

¹<http://wiki.ros.org/roscpp/Overview/Publishers%20and%20Subscribers>

tf a následník *tf2*. Súradnicové systavy sú zaradené do stromu, ktorý vyjadruje vzťahy – transformácie medzi sústavami, ktoré sa v čase môžu meniť. Transformácie sú medzi uzlami prenášané v téme */tf* správami typu *geometry_msgs/Transform*, ktoré obsahujú aj časové razítko.

Triedy implementované v knižnici balíka *tf* pre uzly túto tému dokážu sledovať a zaznamenávať, takže sú zmeny transformácií dostupné aj spätne v určitom (nastaviteľnom) intervale. Trieda potom poskytuje metódy pre výpočet transformácie medzi dvoma ľubovoľnými súradnicovými sústavami (na základe ich názvu) v požadovanom časovom okamihu. V prípade, že neexistuje záznam o transformácii z časovým razítkom rovným požadovanému času, transformácia sa interpoluje z dvoch najbližších záznamov. Knižnica mimo získania samotnej transformácie podporuje aj transformáciu bodov, vektorov, póz, atd.

V tomto projekte bol použitý strom súradnicových sústav na obr. 3.2 odvodený od doporučeného², používaného pre pohybujúce sa roboty. Systavy *velodyne* a *leftcam* boli pridané pre reprezentáciu usporiadania senzorov na vozidle v databázi Kitti³ (kap. 2.1) a odpovedajú umiestneniu LIDAR-u a ľavej kamery. Snímky z RGB kamery pre detekciu pohybu síce použité neboli, ale pózy vozidla priradené záznamom v testovacej sade sú uložené v jej súradnicovej sústave. Táto póza je určená transformáciou medzi sústavami *world* a *odom*. Ostatné transformácie sa v čase nemenia, ale sú periodicky publikované uzlami typu *static_transform_publisher* z balíka *tf*. Spustenie týchto uzlov vykonáva *roslaunch*, pričom hodnoty transformácií sú zapísané v konfiguračnom súbore.



Obr. 3.2: Strom súradnicových sústav použitý v tejto práci.

3.6 Štruktúra

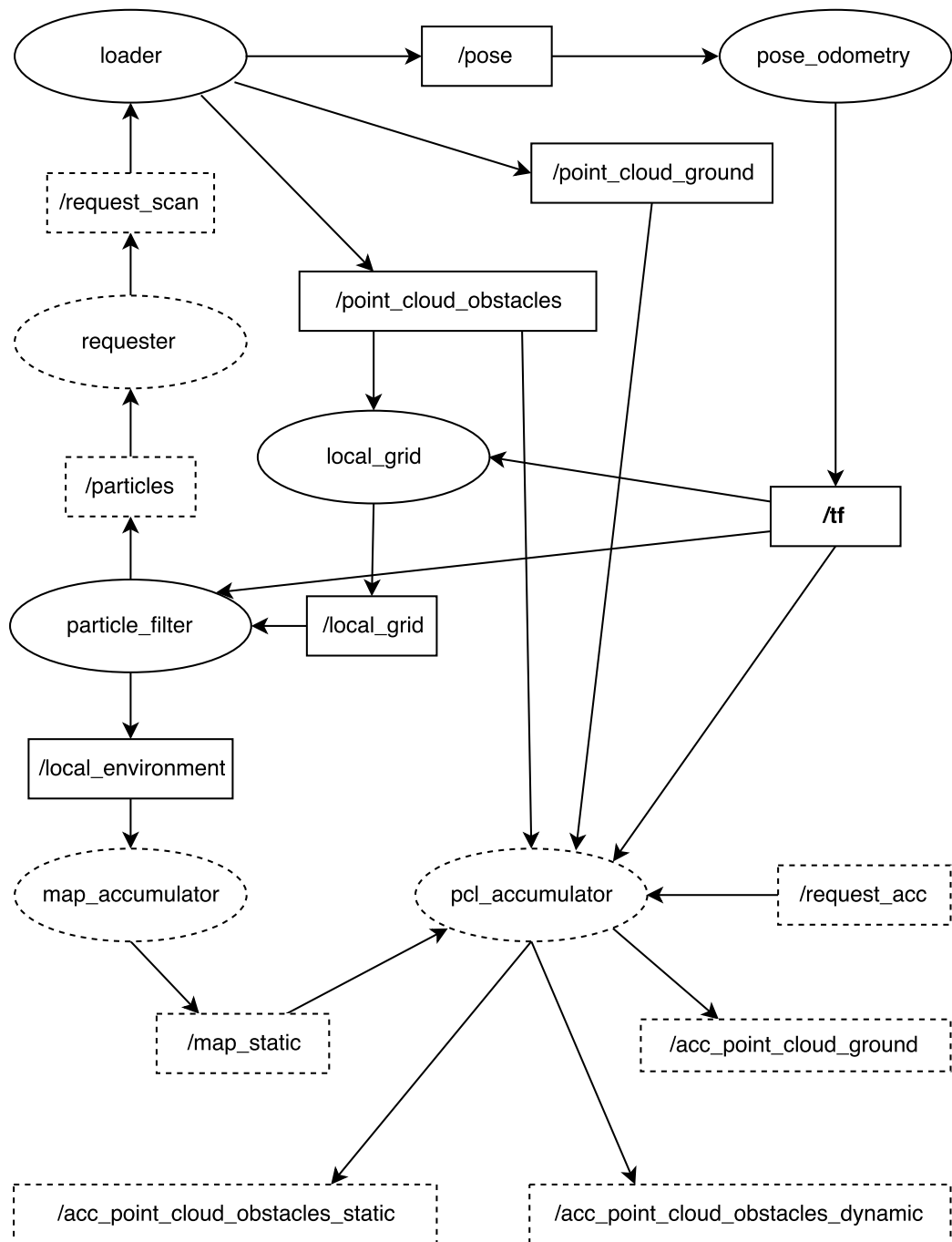
Táto kapitola je venovaná štruktúre systému pre detekciu pohybujúcich sa objektov zobrazenej na obrázku 3.3. Systém bol rozdelený na niekoľko samostatných uzlov tak, aby každý uzol plnil len jednu úlohu. V grafe sú zahrnuté len uzly implementované pre túto prácu, neobsahuje využité pomocné uzly dostupné v balíčkoch systému ROS.

3.6.1 loader

Z dôvodu použitia databázy Kitti ako zdroja dát namiesto reálne vozidla so zenzormi bol vytvorený uzol *loader*, ktorý zastáva úlohu laserového diaľkomeru – LIDAR-u a zároveň spoľahlivého lokalizátora na základe ground-truth póz z datovej sady.

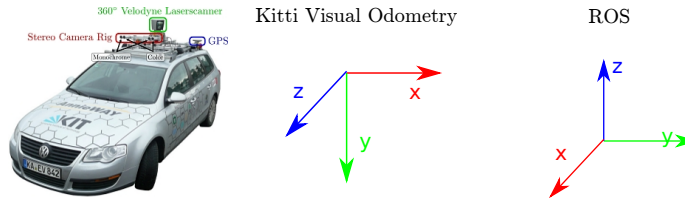
²<http://www.ros.org/reps/rep-0105.html#coordinate-frames>

³<http://www.cvlibs.net/datasets/kitti/setup.php>



Obr. 3.3: Graf uzlov riešenia v Robotickom operačnom systéme. Oválne bloky reprezentujú uzly, hranaté bloky predstavujú kanály (témy) správ. Uzly značené prerušovane nie sú nevyhnutné pre detekciu pohybu, slúžia na testovanie a demonstráciu systému.

Implementovaný bol v jazyku C++ využívajúc knižnicu *but_velodyne_lib*⁴ pre načítanie mračien bodov z Kitti sekvencií. Mračná bodov sú rozdelené na body ležiace na zemi a body prekážok pomocou uložených anotácií, ktoré boli vytvorené použitím metódy Loopy Belief Propagation [21] pre segmentáciu zeme (kap. 2.2).



Obr. 3.4: Porovnanie súradnicovej sústavy databázy Kitti Visual Odometry a sústavy použitej v systéme ROS.

Mračná sú následne transformované do súradnicovej sústavy používanej v systéme ROS, ako je znázornené na obrázku 3.4. Výstupom sú správy obsahujúce spomenuté mračná bodov a prislúchajúce správy obsahujúce pózu vozidla. Z každého snímku *loader* vygeneruje 3 správy – mračno zeme, mračno prekážok a pózu, ktoré majú rovnaké časové razítko.

Uzol má 2 módy behu. V jednom prípade používa interný časovač pre inkrementáciu aktuálnej pozície a následný výstup správ. Táto perióda je nastaviteľná. Počas vývoja systému ale nie je možné dopredu určiť trvanie spracovania, preto tak je pevná perióda nepraktická. Bol teda vytvorený druhý mód behu, ktorý sleduje tému *request_scan* a ďalší snímok zašle až po prijatí správy. Téma *request_scan* je typu *std_msgs/Empty* a teda neobsahuje žiadne dáta – dôležitá je len udalosť prijatia. Tento mód tiež podporuje nastavenie minimálneho časového odstuhu.

3.6.2 pose_odometry

Uzol *pose_odometry* publikuje pózu vozidla v Kitti sekvencii vo forme transformácie pomocou balíka *tf* z kapitoly 3.5. Vstupom je teda póza */pose* a výstupom je transformácia medzi súradnicovým systémom *map* a *odom*. Táto transformácia mohla byť publikovaná priamo z uzlu *loader*, ale s ohľadom na oddelenie zodpovedností bol vytvorený separátny uzol.

3.6.3 local_grid

Tento uzol implementuje generovanie lokálnej mriežky – okupačnej mriežky aktuálneho okolia pomocou postupu popísaného v kapitole 2.3. Vstupom je mračno bodov prekážok (*/point_cloud_obstacles*), výstupom okupačná mriežka v počiatku súradnicovej sústavy *base_link* (podvozok robota). V tejto sústave je okupačná mriežka rovnobežná s rovinou určenou osami X, Y, v globálnej sústave *world* to ale nemusí platiť v závislosti od pózy robota, ktorá nepriamo určuje transformáciu medzi sústavami *world* a *base_link*. Pre transformáciu mračna je použitá funkcionálnosť z balíka *tf2*.

Výstupné správy typu *nav_msgs/OccupancyGrid* majú časové razítko totožné so vstupnou správou obsahujúcou mračno bodov.

⁴https://github.com/robofit/but_velodyne_lib

Okrem mračna bodov *sensor_msgs/PointCloud2*, dokáže *local_grid* spracovať aj pláňárne skeny typu *sensor_msgs/LaserScan*.

3.6.4 `particle_filter`

Uzol *particle_filter* implementuje algoritmus časticového filtra na okupačnej mriežke z kapitoly 2.4. Implementácia bola napísaná v jazyku Python. Toto rozhodnutie vyplynulo zo (subjektívne) širšej ponuky hotových nástrojov a jednoduchosti použitia. Použitie interpretovaného jazyka má výhody aj nevýhody. Program nie je nutné pred spustením prekladať, čo zrýchľuje spätnú väzbu – kontrolu výsledku po úprave nejakej časti. Zároveň je možné skúšať krátke experimenty v interaktívnom príkazovom riadku. Nevýhodou pri vysokej výpočetnej náročnosti je pomalý beh čistého kódu v jazyku Python, bez použitia prekladových nástrojov ako *Cython*⁵ alebo *Numba*⁶. Riešením je použiť Python ako „lepídlo“, kde kód popisuje základnú logiku algoritmu a všetky náročné operácie sú prenechané optimalizovaným, predkompilovaným knižniciam. Cieľom bolo teda vyhnúť sa kódu v jazyku Python, ktorý by iteroval na veľkom zozname ako sú častice alebo bunky okupačnej mriežky a vykonával malý výpočet.

Použitá knižnica *Numpy*, ktorá implementuje datový typ N-rozmernej matice a rôzne maticové operácie toto splňuje. Zároveň je kompatibilná s väzbou na knižnicu pre spracovanie obrazu, *OpenCV*[1], bez nutnosti prekladať matice z jedného datového typu na druhý.

Okupačná mriežka je vyjadrená 2-rozmernou maticou (typ *Numpy.ndarray*). Pre výpočet vzdialenostnej transformácie potrebnej k váhovaniu častíc časticového filtra poslúžila funkcia *cv2.distanceTransform()* zo spomenutej knižnice *OpenCV*.

Množina častíc je rovnako uložená vo forme matice o 2 rozmeroch. Riadky predstavujú častice, stĺpce atribúty častíc ako poloha, vektor rýchlosti, váha, atď. Vždy sa nakladá s maticou ako s celkom – používajú sa funkcie knižnice *Numpy*, ktorých argumentami a hodnotami sú matice. Matice je možné indexovať pre vytvorenie pohľadu na jej časť, napr. `particles[:, [0,1,2]]` pre získanie novej matice z matice `particles` ktorá obsahuje len stĺpce s indexmi 0, 1 a 2.

Ukážka kódu 1 obsahuje algoritmus pre získanie počtu častíc na jednu bunku s použitím funkcie *numpy.histogram2d*. Jedným z nepovinných argumentov funkcie *numpy.histogram2d* je aj matica váh – *weights*. Znamená to, že dokáže vypočítať súčet ľubovoľnej veličiny pre jednotlivé koše histogramu. Toto váhovanie je využité pri výpočte (približnej) cyklickej štandardnej odchýlky smeru pohybu v ukážke kódu 2.

Prevzorkovanie častíc na základe váhy a obsadenosti bunke mriežky je založené na určení počtu opakovaní pre každú časticu. Počet opakovaní je udaný ako reálne číslo. Celá časť udáva istotný počet opakovaní, desatinná časť šancu pre jedno opakovanie navyše. Konečný počet sa potom pre desatinnú časť určí porovnaním s náhodným číslom z intervalu $(0, 1)$. Samotné prevzorkovanie je vykonané jedným volaním *numpy.histogram2d*⁷, pričom návratovou hodnotou je nová matica prevzorkovaných častíc.

Pre overenie funkcie počas vývoja alebo testovania na simulovaných vstupoch bola v tomto uzle tiež implementovaná grafická vizualizácia stavu časticového filtra, pomocou knižnice *matplotlib*⁸. Tieto vizualizácie boli použité vo vyhodnotení v kapitole 4.

⁵<http://cython.org/>

⁶<http://numba.pydata.org/>

⁷<https://docs.scipy.org/doc/numpy-1.12.0/reference/generated/numpy.repeat.html>

⁸<https://matplotlib.org/>

```

def particle_counts(grid, particles):

    cols = grid.cols() # počet stĺpcov okupačnej mriežky
    rows = grid.rows() # počet riadkov

    # Vypočet pozície častice v okupačnej mriežke
    # P_{X|Y|Z} značí index stĺpca v matici častíc
    p_rows = grid.get_row_i(particles[:, [P_X, P_Y, P_Z]])
    p_cols = grid.get_col_i(particles[:, [P_X, P_Y, P_Z]])

    # Výpočet počtu častíc na bunku je ekvivalentný 2-rozmernému histogramu
    particle_counts = numpy.histogram2d(p_rows, p_cols, bins=[rows, cols],
    ↪ range=[[0, rows - 1], [0, cols - 1]])
    return particle_counts # Výsledkom je matica o rozmeroch okupačnej mriežky

```

Ukážka kódu 1: Hustota častíc v bunkách okupačnej mriežky pomocou histogramu

```

def grid_angle_stdev(grid, particles, particle_counts):

    cols = grid.cols() # počet stĺpcov okupačnej mriežky
    rows = grid.rows() # počet riadkov

    p_rows = grid.get_row_i(particles[:, [P_X, P_Y, P_Z]])
    p_cols = grid.get_col_i(particles[:, [P_X, P_Y, P_Z]])

    # P_DX a P_DY sú indexy stĺpcov obsahujúcich vektora rýchlosti častice
    particles_theta_sin = np.sin(np.arctan2(particles[:, P_DY],
    ↪ particles[:, P_DX]))
    sum_theta_sin = np.histogram2d(p_rows, p_cols, bins=[rows, cols],
    ↪ range=[[0, rows - 1], [0, cols - 1]], normed=False,
    ↪ weights=particles_theta_sin)

    particles_theta_cos = np.cos(np.arctan2(particles[:, P_DY],
    ↪ particles[:, P_DX]))
    sum_theta_cos = np.histogram2d(p_rows, p_cols, bins=[rows, cols],
    ↪ range=[[0, rows - 1], [0, cols - 1]], normed=False,
    ↪ weights=particles_theta_cos)

    # delenie korešpondujúcich buniek matíc rovnakých rozmerov
    avg_theta_sin = sum_theta_sin / particle_counts
    avg_theta_cos = sum_theta_cos / particle_counts

    R = np.hypot(avg_theta_sin, avg_theta_cos) # Euklidovská vzdialenosť

    return 1.0 - R # Výsledkom je matica o rozmeroch okupačnej mriežky

```

Ukážka kódu 2: Priemerná hodnota pomocou váhovaného histogramu pre výpočet cyklickej štandardnej odchýlky

Výstupom uzlu je lokálna mapa vyjadrená pomocou Dempster-Shaferovej teórie (kap. 2.5.2) v téme *local_environment*. Pre tieto správy sa nepodarilo nájsť vhodný existujúci typ a tak bol definovaný nový – *GridDST*. Uzol zároveň pre diagnostické účely publikuje množinu častíc v téme *particles*, ktorá s vynechanými vektormi rýchlosti tvorí mračno bodov. Použitý je teda typ *sensor_msgs/PointCloud2*.

3.6.5 requester

Tento uzol slúži na generovanie požiadaviek na ďalší snímok pre uzol *loader* počas vývoja. Akonáhle skončí spracovanie snímku časticovým filtrom a v téme */particles* sa objaví nový stav častíc, odošle požiadavku do témy */request_scan*.

Čakanie na požiadavku ďalšieho snímku by mohlo byť v module *loader* implementované poskytnutím služby v systéme ROS (service), vo výsledku by ale návratová hodnota bola rovnako prázdna. Uzol *loader* je tiež zodpovedný len za načítanie dát a pre oddelenia zodpovedností by nemal vedieť o existencii uzlu časticového filtra, na ktorý sa čaká.

Oddelenie uzlu *requester* od časticového filtra umožnilo počas vývoja sekvenciu pozastaviť ukončením tohoto uzlu, a znovu načítať nejaký uzol so zmenenými parametrami alebo kódom bez nutnosti reštartu KITTI sekvencie alebo celého systému *roslaunch*.

3.6.6 mapper

Tento uzol slúži na vytvorenie globálnej mapy, ktorá obsahuje informácie zo všetkých lokálnych reprezentácií prostredia získaných počas jazdy (téma *local_environment*). Táto mapa uchováva len základné priradenie dôvery pre statickú bunku ($m^{x,y,t}(\{S\})$), pričom pre fúziu lokálnej mapy je využitý postup z kapitoly 2.5.3.

Globálna mapa je implementovaná ako dvojrozmerná matica, ktorá pokrýva najmenšiu obdĺžnikovú plochu pokrývajúcu všetky doterajšie vstupné lokálne mapy. V prípade, že pri aktualizácii lokálna mapa zasahuje mimo tento priestor, je vytvorená nová matica, do ktorej je skopírovaný obsah pôvodnej. Lepšou alternatívou by bolo použitie riedkej matice, ale mapovanie bolo implementované len pre demonštráciu a nie je cieľom tejto práce.

3.6.7 pcl_accumulator

Pre zobrazenie mračien bodov filtrovaných na body prislúchajúce dynamickým a statickým objektom bol vytvorený uzol *pcl_accumulator*, ktorý uchováva skeny z LIDAR-u v témach *point_cloud_ground* a */mboxpoint_cloud_obstacles*. Zároveň má pamäť pre poslednú globálnu mapu v téme *map_static*.

Na požiadanie v téme *request_acc* vráti doposiaľ naakumulované mračno zeme *acc_point_cloud_ground*, a mračná *acc_point_cloud_obstacles_static* a *acc_point_cloud_obstacles_static*, ktoré sú na statické a dynamické rozdelené na základe poslednej mapy s nastaviteľnou hraničnou hodnotou.

3.6.8 local_grid_simulator

Tento modul, ktorý nie je zahrnutý v grafe bol vytvorený za účelom otestovania detekcie pohybu časticovým filtrom na presných, syntetických vstupných dátach. Nahradzuje moduly *loader* a *local_grid*

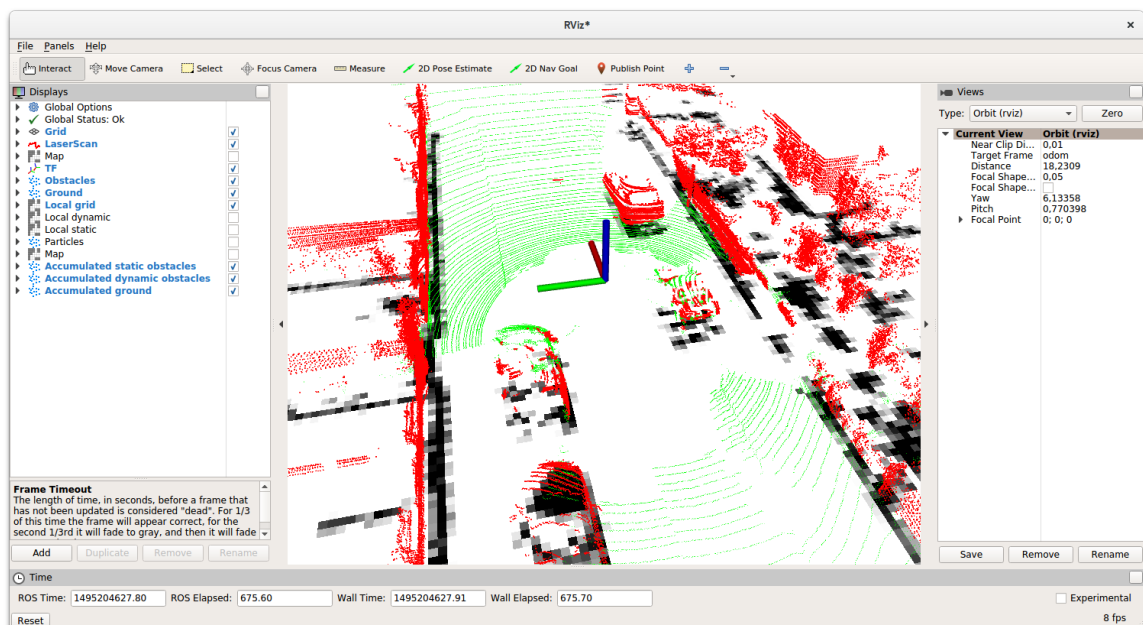
Uzol podporuje aj vmiešanie náhodného šumu z normálnej distribúcie do výstupu.

3.7 Vizualizácia

Ako hlavný nástroj pre vizualizáciu sensorických dát a stavu detektora pohybu bol použitý štandardný nástroj rviz na obr. 3.5, ktorý je súčasťou Robotického operačného systému. Rviz je univerzálny 3D vizualizér s podporou pre množstvo bežne používaných typov správ. Témy, ktorých správy má rviz sledovať a zobrazovať sú užívateľsky nastaviteľné, vrátane parametrov.

V tejto práci bol *rviz* využitý na zobrazenie sensorických dát v podobe mračen bodov v okolí robota, znázornenie stromu transformácií a zobrazenie rôznych planárnych máp ako lokálna okupačná mriežka a základné priradenia pravdepodobnosti pre hypotézu statickej a hypotézu dynamickej bunky v lokálnom prostredí.

Pri zobrazovaní akumulovaných mračen bodov sú už počty bodov príliš vysoké na efektívnu vizualizáciu, v tomto prípade boli mračná podvzorkované pomocou vstavaných uzlov typu *pcl/VoxelGrid*, ktoré využívajú knižnicu PCL⁹. VoxelGrid redukuje mračno pomocou 3D mriežky, výstupom je jeden bod na každú bunku mriežky – centroid všetkých bodov, ktoré do nej spadajú.



Obr. 3.5: Vizualizačný nástroj rviz, zobrazené je nefiltrované mračno bodov **prekážok**, z neho získaná lokálna okupačná mriežka a mračno **zeme**.

⁹http://pointclouds.org/documentation/tutorials/voxel_grid.php

Kapitola 4

Experimentálne výsledky

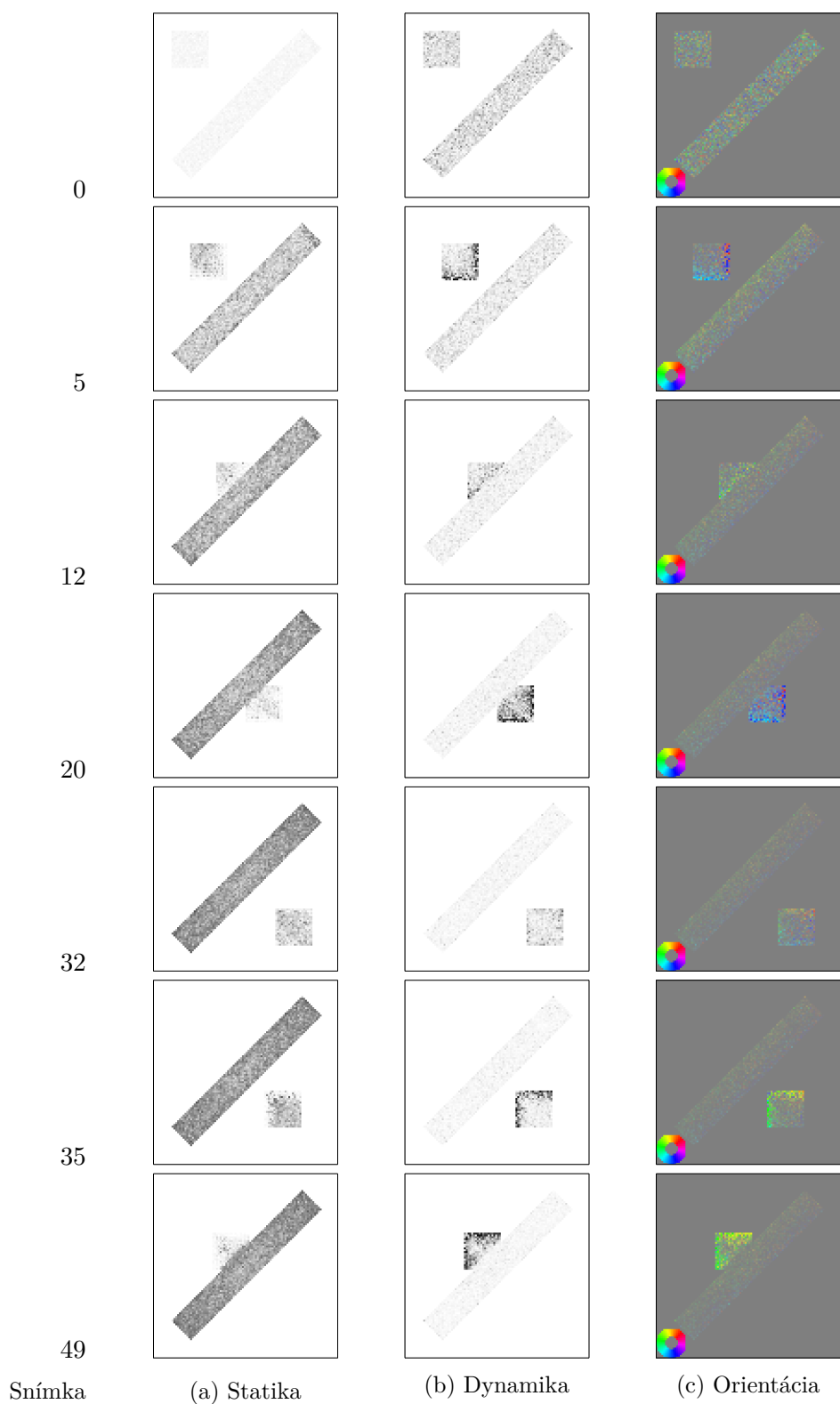
Táto kapitola sa venuje vyhodnoteniu funkčnosti detektora pohybu. Napriek cieľu minimalizovať počet parametrov, konfigurácia detektora pohybu na základe časticového filtra ich obsahuje 8. Pri testovaní boli zvolené nasledujúce hodnoty:

Parameter	Hodnota	Vysvetlenie
grid_occupied_threshold	0.7	Hraničná hodnota pre obsadenú bunku na okupačnej mriežke (binarizácia)
dist_stdev	1.6	Parameter Gaussovej funkcie aplikovanej po vzdialenostnej transformácii μ_{dist} , kapitola 2.4.2
max_cell_aprticles	50	Maximálny počet častíc v 1 bunke N_c , kapitola 2.4
new_particle_ratio	0.2	Zlomok voľného počtu častíc v bunke, ktorý sa zaplní v jednom kroku
max_particle_velocity	25 m/s	Horná hranica náhodnej rýchlosti pri vytváraní častice
static_particle_ratio	0.3	Zlomok počtu novo vytváraných častíc, ktorým sa nastaví nulová rýchlosť
min_survival_probability	0.5	Minimálna šanca pre prežitie častice, aj mimo zaplnenej bunky
static_threshold	0.01 m/s	Minimálna rýchlosť, pri ktorej sa častica považuje za dynamickú

4.0.1 Simulácia okupačnej mriežky

Pre vylúčenie účinkov šumu bol časticový filter otestovaný na syntetickom vstupe obsahujúcom jeden dynamický objekt a jeden statický, pričom po určitý čas sa na okupačnej mriežke prekrývajú.

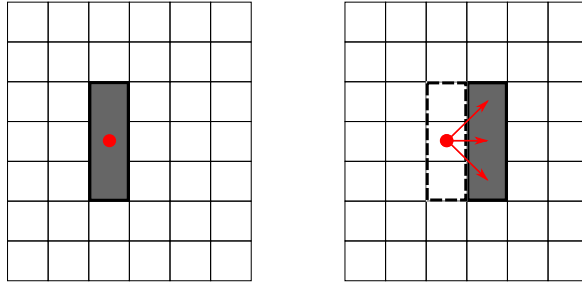
Správanie implementovaného detektora je zobrazené ako séria lokálnych máp na obrázku 4.1. Simulovaný bol statický objekt (diagonálny obdĺžnik) a objekt tvaru štvorca, ktorý sa periodicky presúva tam a späť po druhej diagonálnej osi. Prvé dva stĺpce znázor-



Obr. 4.1: Lokálne prostredie simulovaného statického a dynamického sa objektu. Štvorec sa pohybuje diagonálne a v snímku 30 mení smer na opačný. Orientácia častíc konverguje prevažne na napredujúcej kontúre pohybujúceho sa objektu, ktorý je v smere posuvu dostatočne dlhý na to, aby sa častice postupne rozptýlili. Mimo kontúry sa zvyšuje viera v hypotézu statickej bunky. Zobrazená orientácia je priemerom smeru častíc v bunke.

ňujú základné priradenie dôvery pre hypotézu statickej bunky $m(S)$ a hypotézu dynamickej bunky $m(D)$ pre univerzum $\Theta = \{F, S, D\}$ v rámci Dempster-Shaferovej teórie. Tretí stĺpec zobrazuje priemernú orientáciu častíc v danej bunke mriežky. Z dôvodu lepšej viditeľnosti orientácie bola použitá malá okupačná mriežka o rozmeroch 100×100 .

Časticový filter na mriežke je schopný úspešne rozlišovať medzi pohybujúcimi sa a statickými objektmi. Smer častíc na bunkách obsadených dynamickým objektom konverguje a aj pri zmene smeru pohybu o 180° nedochádza k prílišnej zotrvačnosti.



Obr. 4.2: Nejednoznačnosť pohybu pre individuálnu bunku mriežky

Orientácia bunky – priemerná orientácia častíc v bunke ale závisí aj od orientácie hrany, nielen od pohybu. Aj prípade pevného objektu, ktorý nemení tvar a pohybuje sa priamočiaro, pohyb v individuálnej bunke nie je jednoznačný. Tento problém je zobrazený na obr. 4.2. V bunkách existujú častice vo všetkých smeroch, pravdepodobnosť výskytu ale klesá s uhlom medzi smerom častice a orientáciou hrany. Priemerný smer častíc v bunke teda bude aproximovať kolmicu na hranu. Pre detekciu pohybu takýto filter síce postačuje, nie je ale možné pre jednotlivé bunky určiť reálny smer pohybu objektu.

Z experimentu tiež vyplýva, že smer pohybu častíc konverguje len na doprednej kontúre objektu. Pokiaľ je bunka dlhodobo obsadená, dochádza na nej k rozptylu častíc, pretože bez striedania stavu obsadenosti nie je možné rozlíšiť, ktorým smerom sa pohybuje objekt na tejto bunke. V bunkách mimo tejto doprednej kontúry sa teda zvyšuje priradenie dôvery pre statickosť ($m(S)$).

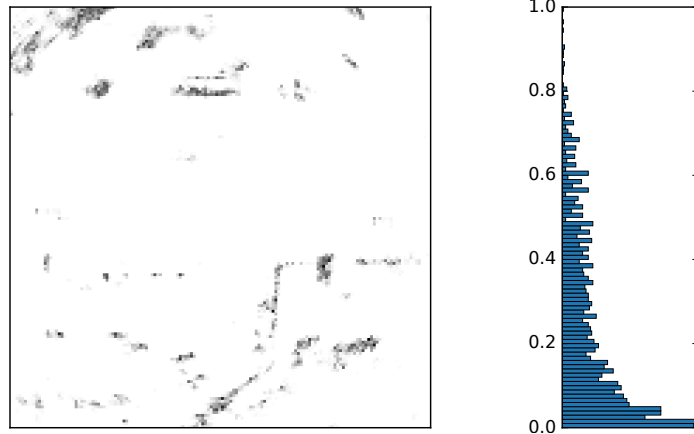
Pre generovanie simulačných dát bol použitý uzol *local_grid_simulator* z kapitoly 3.6.8.

4.1 Testovanie na meraniach zo senzoru LIDAR

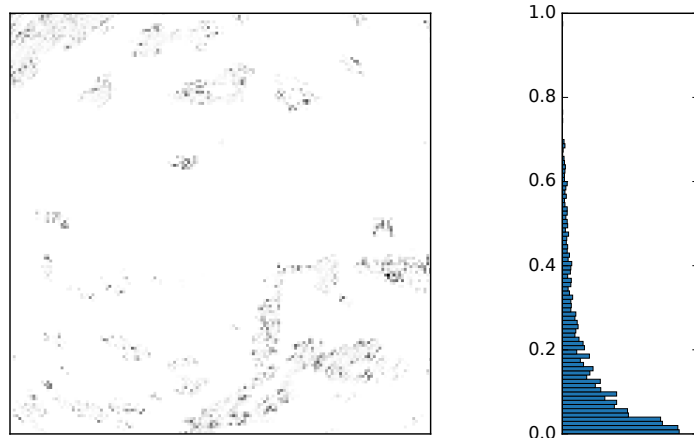
Systém bol otestovaný aj na lokálnej okupačnej mriežke, ktorá vznikla z meraní Laserovým dialkometerom. Použitá databáza KITTI Visual Odometry neobsahuje anotácie o pohybe v scéne a tak bol test vykonaný len vizuálne podľa vykresleného lokálneho prostredia na obr. 4.3.

V grafe základného priradenia pravdepodobnosti pre hypotézu dynamickej bunky (dynamickej mapa na obrázku) je zreteľné veľké množstvo šumu, dôležitý je ale fakt, že v statickej mape sú pohybujúce sa objekty utlmené. Zdá sa, že v tomto prípade nedochádza k tak výraznému efektu doprednej kontúry z predchádzajúcej kapitoly. Vplýva na to prítomnosť šumu a nízke rozlíšenie okupačnej mriežky v porovnaní s veľkosťou pohybujúcich sa vozidiel.

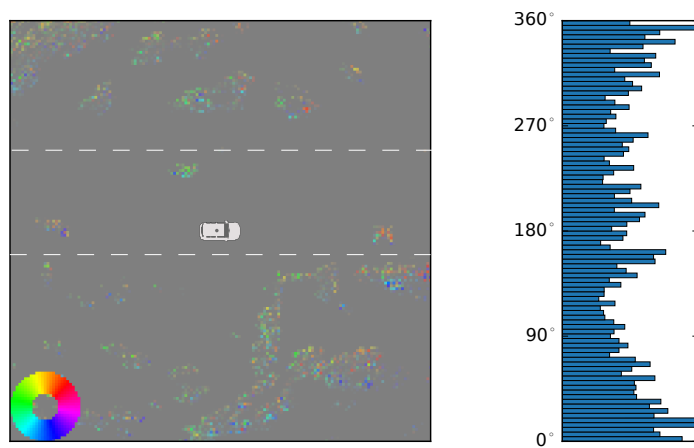
Statická mapa je teda použiteľná pre ofiltrovanie senzorických dát prislúchajúcich statickým častiam prostredia. Účelom môže byť napríklad mapovanie alebo vizuálna odometria – stanovenie pózy vozidla na základe obrazových informácií o prostredí.



(a) Statická mapa



(b) Dynamická mapa



(c) Priemerná orientácia častíc v bunke znázornená farbou

Obr. 4.3: Lokálne prostredie robota v snímku 11 sekvencie 04 databázy KITTI. Jedná sa o rovnú ulicu s riedkou premávkou. Dve autá idúce v rovnakom pruhu a jedno v protismere sú na statickej mape potlačené. Časticový filter bol spustený na začiatku sekvencie (od snímku 0).

Rýchlosť spracovania

Pomocou nástroja cProfile bola preskumaná výpočetná náročnosť implementácie časticového filtra. Na 15 snímkoch sekvencie 04 bola zistená priemerná doba výpočtu 129ms. Táto hodnota stúpa s počtom zaplnených buniek, keďže je ale počet častíc na jednu bunku limitovaný, doba výpočtu je tiež zhora ohraničená. Nastavením maximálneho počtu častíc na bunku je možné urobiť kompromis medzi presnosťou a dobou výpočtu.

Databáza obsahuje merania z LIDAR-u, ktorého frekvencia skenov je 10Hz. Maximálna povolená doba výpočtu, v prípade behu v reálnom čase je teda 100ms. Nie je vylúčené, že by sa k tomuto výsledku nepodarilo dostať po optimalizácii najnáročnejších rutín, ktorými sú vzorkovanie nových častíc, alebo portovanie na GPU.

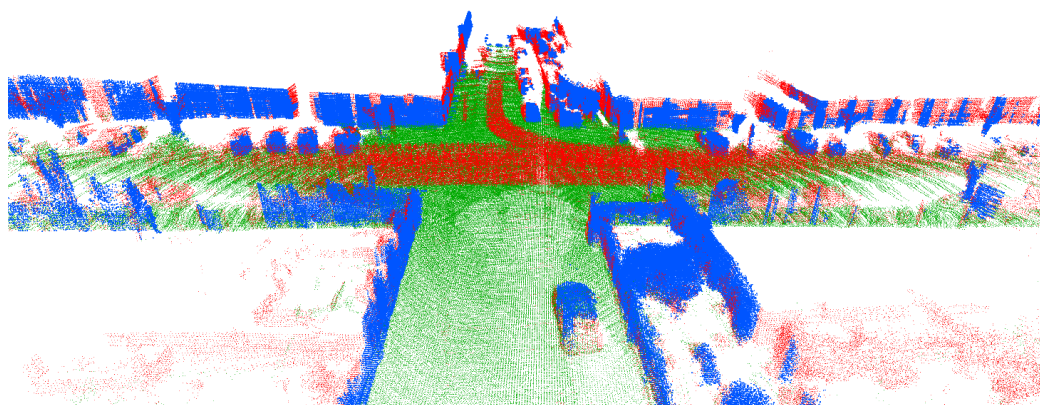
Nameraná hodnota je ale čistá doba výpočtu pre aktualizáciu stavu časticového filtra, bez prihliadnutia na réžiu načítania a konverzie dát, zasielania správ v Robotickom operačnom systéme a diagnostické výstupy. Pri behu celého systému je správa lokálneho prostredia publikovaná každých 1058,2ms so štandardnou odchýlkou 99.36ms (meranie trvalo 60snímkov). Testy prebiehali na počítači s procesorom Intel(R) Core(TM) i5-6200U (2.30GHz), pričom knižnica NumPy bola nalinkovaná s balíkom pre rutiny lineárnej algebry OpenBLAS.

4.1.1 Akumulovaná mapa

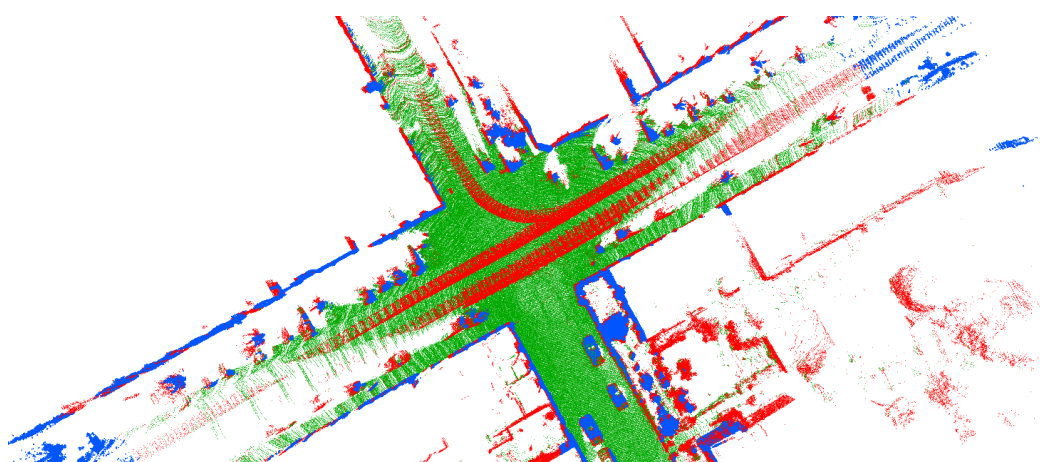
Po túto kapitolu sa vyhodnotenie týkalo len lokálnej reprezentácie prostredia – v jednom časovom okamihu. Pri filtrovaní v čase sa vytvára globálna mapa, podľa kapitoly 2.5.3, ktorá je rekurzívnou fúziou lokálneho prostredia v každom kroku. Ak sa počas tej istej jazdy akumulujú mračná bodov z LIDAR-u, je možné ich následne rozdeliť na statické a dynamické, ako na obrázku 4.4. Dynamické objekty, ktoré v tejto situácii zanechávajú stopy, sú jasne odlíšené.

4.2 Zrnutie

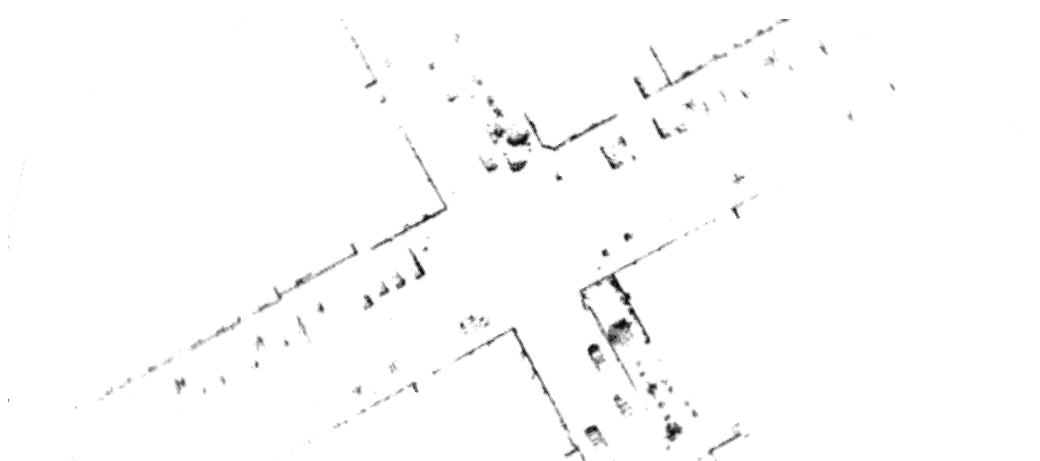
Ukázalo sa, že použitie časticového filtra na základe [18] je vhodné na detekciu pohybu. Na vlastnej implementácii systému bolo ale zistené, že takýto systém nie je plne robustný a pokrýva len určitú podmnožinu prípadov, v závislosti od konfigurácie. Na bežných dátach z prostredia ulice systém dokáže rozlíšiť pohybujúce sa vozidlá, pretože nastavenie rozlíšenia okupačnej mriežky na 0.2m je dostatočne hrubé na to, aby pohybujúcimi sa objektmi nebola pokrytá príliš veľká plocha. Zároveň k tomu prispieva použitie LASER-ového diaľkomeru, ktorého výstupné mračná bodov sa vyskytujú prevažne na priklonených hranách objektov. Okupačná mriežka potom podobne obsahuje len kontúru objektu v smere detektoru. Pri použití iného senzoru pre vytvorenie okupačnej mriežky, alebo fúzii dát zo sensorov z viacerých ulov by sa predpokladalo zvýšenie falošných poplachov (false positives) na mriežke hypotéz statickosti bunky.



(a) Pohľad v smere jazdy



(b) Pohľad zvrchu



(c) Globálna mapa statickej viery

Obr. 4.4: Akumulované mračná bodov. **Zelenou** sú značené body ležiace na zemi, **modrou** body spadajúce do statických buniek globálnej mapy a **červenou** ostatné, body dynamických objektov. Použité boli snímky 600 – 747 zo sekvencie 07.

Záver

Cieľom práce bolo preskúmať metódy pre spracovanie obrazu a hĺbkových dát za účelom detekovania pohybujúcich sa objektov v okolí robota, ktorý sa zároveň môže pohybovať ako pozorovateľ. V riešení boli použité metódy pravdepodobnostnej robotiky, vysvetlené v kapitole 1, ktoré dokážu reprezentovať a pracovať s neurčitostou. Navrhnuté riešenie je postavené na vytvorení dvojrozmiernej okupačnej mriežky bez filtrovania v čase, v práci nazývanej ako lokálna mapa. Okupačná mriežka je dostatočne jednoduchá na to, aby mohla byť vytvorená na základe dát z rôznych typov senzorov, a zároveň nevnucuje parametrické modelovanie objektov prostredia napríklad vo forme geometrických útvarov. Ako detektor pohybu je použitý časticový filter popísaný v kapitole 2.4, ktorý modeluje obsadenosť a pohyb na jednotlivých bunkách mriežky, pričom sa ale častice môžu medzi bunkami presúvať. Informácie o obsadenosti, statickosti a dynamickosti buniek su potom vyjadrené v rámci Dempster-Shaferovej teórie.

Systém bol implementovaný v Robotickom operačnom systéme, ktorý umožnil využitie existujúcich nástrojov pre modulárny návrh, diagnostiku a vizualizáciu výsledkov merania. Rozdelenie na uzly s presne vymedzenými zodpovednosťami umožnilo výmenu dát s nástrojmi systému ROS a učinilo súčasti riešenia použiteľnými aj pri vývoji budúcich systémov.

Zrealizovaný detektor bol otestovaný na prispôsobenom simulovanom prostredí, kde sa ukázalo, že je schopný detekovať pohyb a rozlíšiť statické a dynamické objekty, do určitej miery aj ich smer. Problémom sú ale veľké celistvé pohybujúce sa objekty, ktoré bunku mriežky prekrývajú dlhý čas. Dochádza vtedy k rozplytu častíc časticového filtra rovnako, ako na statických častiach scény.

Počas testov na reálnom prostredí mestskej premávky v databázi KITTI Visual Odometry bol tento problém ale málo výrazný v dôsledku šumu, vhodne nastaveného rozlíšenia mriežky a vlastnostiam senzoru typu LiDAR. Kapitola 4 obsahuje podrobnejšie vyhodnotenie systému a ukážku použitia pre odfiltrovanie pohybujúcich sa objektov z mapy tvorenej počas jazdy.

Primárnym smerom ďalšieho vývoja by mohlo byť lepšie preskúmanie metód a možných úprav, ktoré by zmiernili spomenutý problém s veľkými objektmi. Riešenie tiež nie je dostatočne výkonné, aby dokázalo spracovať senzorické dáta v reálnom čase. Beh v reálnom čase síce nebol nevyhnutným cieľom, implementácia ale neobsahuje žiadne kritické algoritmy, ktoré nebolo možné paralelizovať. Priestor pre ďalšiu optimalizáciu algoritmu, zníženie dopadu réžie systému ROS, a prípadnej akcelerácie na grafických procesoroch je teda veľký.

Literatúra

- [1] The OpenCV Reference Manual. April 2014.
- [2] ROS/Introduction - ROS Wiki. [online], Navštívené 14. 5. 2017.
- [3] Topics - ROS Wiki. [online], Navštívené 14. 5. 2017.
- [4] Velodyne HDL-64E. [online], Navštívené 14. 5. 2017.
- [5] Beránek, L.: Základy Dempster-Shaferovy teorie a její aplikace pro modelování bezpečnosti a spolehlivosti. *Chemmagazín*, ročník 2010, č. 6, 2010: s. 28–30, ISSN 1210-7409.
- [6] Chen, C.; Tay, C.; Laugier, C.; aj.: Dynamic Environment Modeling with Gridmap: A Multiple-Object Tracking Application. In *2006 9th International Conference on Control, Automation, Robotics and Vision*, Dec 2006, s. 1–6, doi:10.1109/ICARCV.2006.345399.
- [7] Danescu, R.; Oniga, F.; Nedevschi, S.: Modeling and Tracking the Driving Environment With a Particle-Based Occupancy Grid. *IEEE Transactions on Intelligent Transportation Systems*, ročník 12, č. 4, Dec 2011: s. 1331–1342, ISSN 1524-9050, doi:10.1109/TITS.2011.2158097.
- [8] Geiger, A.; Lenz, P.; Urtasun, R.: Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [9] Grisetti, G.; Stachniss, C.; Burgard, W.: Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, April 2005, ISSN 1050-4729, s. 2432–2437, doi:10.1109/ROBOT.2005.1570477.
- [10] Himmelsbach, M.; von Hundelshausen, F.; Wuensche, H.-J.: LIDAR-Based Perception for Offroad Navigation. In *Proceedings of 5. Workshop Fahrerassistenzsysteme (FAS)*, editace C. Stiller; M. Maurer, Walting, Germany, 2008.
- [11] Homm, F.; Kaempchen, N.; Ota, J.; aj.: Efficient occupancy grid computation on the GPU with lidar and radar for road boundary detection. In *2010 IEEE Intelligent Vehicles Symposium*, June 2010, ISSN 1931-0587, s. 1006–1013, doi:10.1109/IVS.2010.5548091.

- [12] Hornung, A.; Wurm, K. M.; Bennewitz, M.; aj.: OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Autonomous Robots*, 2013, doi:10.1007/s10514-012-9321-0, software available at <http://octomap.github.com>. URL <http://octomap.github.com>
- [13] Ing. Filip Orság, P.: *Robotika*. Brno, 2006 vydání, 2006, ISBN Filip Orság.
- [14] Jøsang, A.; Pope, S.: Dempster's Rule As Seen by Little Colored Balls. *Computational Intelligence*, ročník 28, č. 4, may 2012: s. 453–474, doi:10.1111/j.1467-8640.2012.00421.x. URL <https://doi.org/10.1111%2Fj.1467-8640.2012.00421.x>
- [15] Mardia, K. V.: *Directional statistics*. Chichester New York: J. Wiley, 2000, ISBN 9780470317815.
- [16] Petrovskaya, A.; Thrun, S.: Model based vehicle detection and tracking for autonomous urban driving. *Autonomous Robots*, ročník 26, č. 2, 2009: s. 123–139, ISSN 1573-7527, doi:10.1007/s10514-009-9115-1. URL <http://dx.doi.org/10.1007/s10514-009-9115-1>
- [17] Stepan, P.; Kulich, M.; Preucil, L.: Robust data fusion with occupancy grid. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, ročník 35, č. 1, Feb 2005: s. 106–115, ISSN 1094-6977, doi:10.1109/TSMCC.2004.840048.
- [18] Tanzmeister, G.; Thomas, J.; Wollherr, D.; aj.: Grid-based mapping and tracking in dynamic environments using a uniform evidential environment representation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, ISSN 1050-4729, s. 6090–6095, doi:10.1109/ICRA.2014.6907756.
- [19] Thrun, S.; Burgard, W.; Fox, D.: *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series)*. The MIT Press, 2005, ISBN 9780262332750. URL <https://mitpress.mit.edu/books/probabilistic-robotics>
- [20] Yu, C.; Cherfaoui, V.; Bonnifait, P.: Evidential occupancy grid mapping with stereo-vision. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, June 2015, ISSN 1931-0587, s. 712–717, doi:10.1109/IVS.2015.7225768.
- [21] Zhang, M.; Morris, D. D.; Fu, R.: Ground Segmentation Based on Loopy Belief Propagation for Sparse 3D Point Clouds. In *2015 International Conference on 3D Vision*, Oct 2015, s. 615–622, doi:10.1109/3DV.2015.76.

Prílohy

Príloha A

Obsah CD

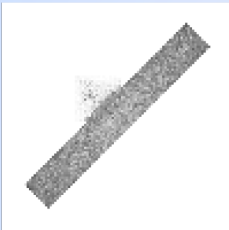
- `/doc/dp.pdf` Tento dokument
- `/doc/tex` Zdrojové súbory tejto správy
- `/src/mdet` Zdrojový kód detektoru pohybu
- `/src/Readme.txt` Popis a pokyny k použitiu
- `/kitti_example/` Zlomok databázy KITTI Visual Odometry pre demonštráciu, anotačné súbory zeme

Príloha B

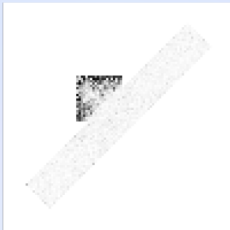
Plagát

**Moving Object Detection in
the Environment of a Mobile Robot**

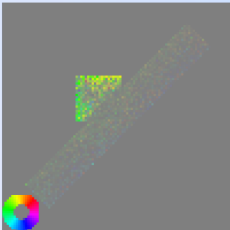
- Occupancy grid from LiDAR point cloud
- Motion detection by modeling cell occupancy and motion using a particle filter



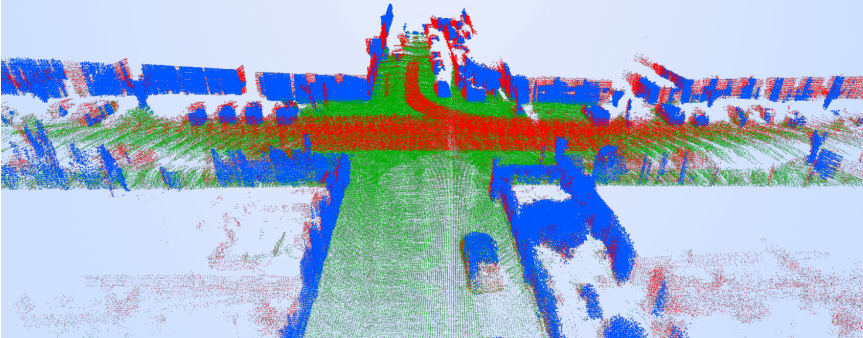
Static cells




Dynamic cells



Orientation





FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ

Author
Bc. Viktor Dorotovič
Supervisor
Ing. Martin Veľas