

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

RGBD kamery v mobilnom robotickom mapovaní

Dizertačná práca

Bratislava 2018

Ing. Peter Beňo

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

RGBD kamery v mobilnom robotickom mapovaní

Dizertačná práca

Študijný program: Robotika a kybernetika

Školateľ: doc. Ing. František Duchoň, PhD.

Školiace pracovisko: Ústav robotiky a kybernetiky

Evidenčné číslo: FEI-104364-50733

Bratislava 2018

Ing. Peter Beňo

Abstrakt

FAKULTA ELEKTROTECHNIKY A INFORMATIKY, Slovenská Technická Univerzita v Bratislave

Názov práce: RGBD kamery v mobilnom robotickom mapovaní

Študijný program: Robotika a kybernetika

Autor: Ing. Peter Beňo

Školiteľ: doc. Ing. František Duchoň, PhD.

Školiace pracovisko: Ústav robotiky a kybernetiky

Kľúčové slová: mapovanie, lokalizácia, vizuálna odometria, SLAM, hĺbkové kamery, Kinect

Dôležitým prvkom naozaj autonómneho mobilného robotického systému je jeho schopnosť vnímať prostredie, v ktorom sa nachádza a pohybovať sa v ňom. Aby bol robot schopný autonómne prechádzať prostredím a plniť v ňom svoje úlohy, je výhodné, aby disponoval mapou prostredia. Mapu prostredia je možné vytvárať na základe dát zo širokého spektra snímačov. V posledných rokoch výrazne získavajú na popularite vizuálne mapovacie algoritmy používajúce hĺbkové kamery. Hlavným dôvodom je nízka cena a dobrá dostupnosť takýchto zariadení.

Dostupné algoritmy sa zameriavanú najmä na kvalitu výslednej mapy, bez ohľadu na náročnosť spracovania. To výrazne limituje možnosti nasadenia aktuálnych metód na platformách s obmedzeným výpočtovým výkonom.

Práca čitateľovi predstavuje témy hĺbkových kamier, vizuálnej odometrie a robotického mapovania v širšom kontexte. Z početného spektra dostupných postupov identifikuje algoritmy, ktoré je možné optimalizovať pre beh na malom mobilnom robote. Na ich základe navrhuje riešenie vizuálnej odometrie a mapovania pre túto platformu. Pre zvýšenie robustnosti voči zmenám osvetlenia je použitý filter obrazu CLAHE. Za účelom umožnenia behu s frekvenciou 30 Hz je predstavená schéma paraleлизácie na výpočtovej jednotke CPU. Na zníženie tempa rastu absolútnej chyby je predstavený mechanizmus extrakcie lokálneho modelu mapy. Ostatné časti riešenia sú dôsledne analyzované s ohľadom na výpočtovú zložitosť a kvalitu výstupu.

Navrhnuté riešenie bolo implementované a jeho vlastnosti overené na reálnom robotickom systéme.

Abstract

FACULTY OF ELECTRICAL ENGINEERING AND INFORMATION TECHNOLOGY,
Slovak University of Technology in Bratislava

Name of thesis: RGBD maps in mobile robot mapping

Study programme: Robotics and cybernetics

Author: Ing. Peter Beňo

Supervisor: doc. Ing. František Duchoň, PhD.

Department: Institute of Robotics and Cybernetics

Keywords: mapping, localization, visual odometry, SLAM, depth cameras, Kinect

One of important aspects of truly autonomous mobile robot is its ability to sense the environment around. For tasks such as navigation, exploration and disaster response, it is significantly beneficial for the robot to know the map of the environment. Maps of environments can be created with data from various sensors. In recent years, mapping algorithms utilizing depth cameras have gained significant popularity, mainly because of their low price and good availability on the market.

Available implementations focus on quality of produced maps, sacrificing the processing speed. This fact limits applications on systems with limited processing power.

This thesis introduces topics of depth cameras, visual odometry and robotic mapping in wider context. From numerous range of available approaches, algorithms suitable for small mobile robot are identified and used for design of proposed solution. To raise the robustness to varying lightning conditions, usage of CLAHE filter is proposed. Schema of parallelization on CPU is introduced which enables the solution to run with frame-rate of 30 Hz. Method of local map extraction is then introduced. This method limits the rate of growth of absolute error and enables camera position recovery after extreme errors. The proposed solution was implemented and its properties were evaluated on real robotic system.

Podčakovanie

Za odborné vedenie, podporu a cenné rady by som rád podčakoval svojmu školiteľovi, doc. Ing. Františkovi Duchoňovi, PhD., vedúcemu oddelenia prof. Ing. Petrovi Hubinskému, PhD., kolegom Ing. Martinovi Dekanovi, PhD. a Ing. Michalovi Tolgyessymu, PhD., ako aj ostatným členom kolektívu Ústavu robotiky a kybernetiky.

Rád by som podčakoval mojim rodičom, blízkej rodine a mojej partnerke Lýdii za ich vytrvalú podporu, ich osobné obety pre zabezpečenie efektívnosti mojej práce, nekonečnú trpežlivosť a hlavne dôveru v moje schopnosti.



ZADANIE DIZERTAČNEJ PRÁCE

Autor práce:

Ing. Peter Beňo

Študijný program:

robotika a kybernetika

Študijný odbor:

9.2.7. kybernetika

Evidenčné číslo:

FEI-104364-50733

ID študenta:

50733

Vedúci práce:

doc. Ing. František Duchoň, PhD.

Názov práce:

3D mapy v mobilnej robotike

slovenský jazyk

Jazyk, v ktorom sa práca vypracuje:

Špecifikácia zadania:

Kľúčovou otázkou pri riešení akejkoľvek úlohy v mobilnej robotike zohráva vnímanie prostredia robotom. Aby robot bol schopný vnímať svoje prostredie, musí používať snímače. Posledným trendom v mobilnej robotike je využívanie snímača Kinect. Tento snímač poskytuje informácie z vizuálneho systému, ale aj vzdialenosť k odpovedajúcim vizuálnym informáciám. Cieľom práce je navrhnuť vlastnú metodológiu spracovania údajov z tohto snímača tak, aby bolo dosiahnuté kvalitatívne a kvantitatívne lepšie riešenie štandardných úloh mobilnej robotiky ako je lokalizácia, navigácia alebo mapovanie. Táto problematika je vysoko aktuálna, čo dokazujú nové sekcie na vedeckých konferenciách nesúce názov "RGB-D mapping".

Dátum zadania:

27. 08. 2013

Dátum odovzdania:

31. 05. 2016

Ing. Peter Beňo
riešiteľ

prof. Ing. Ján Murgaš, PhD.
vedúci pracoviska

prof. Ing. Ján Murgaš, PhD.
garant študijného programu

Tézy práce

1. Navrhnúť nový vizuálny SLAM algoritmus využívajúci RGBD dátá pre malý mobilný robot s čo možno najnižšími nárokmi na hardvérovú konfiguráciu robota.
2. Navrhnúť mapové reprezentácie a spôsob prenosu údajov medzi viacerými robotmi, ktoré by umožňovali ľahký prenos údajov z mapy cez sieť a jej vhodnú distribúciu medzi viaceré roboty.
3. Vytvoriť stratégie multiagentového mapovania priestoru viacerými robotmi a spájania týchto čiastkových máp z viacerých robotov do jednej spoločnej globálnej mapy.
4. Všetky navrhnuté postupy a algoritmy implementovať do reálnych zariadení a verifikovať ich z hľadiska požadovaných vlastností.

1 Obsah

1	OBSAH	7
2	ZOZNAM POUŽITÝCH SKRATIEK	10
3	ZOZNAM OBRÁZKOV	11
4	ZOZNAM TABULIEK	14
5	ÚVOD	15
6	OD KAMIER KU HĽBKOVÝM KAMERÁM	17
6.1	HĽBKOVÝ OBRAZ, STEREOVÍZIA	18
6.2	HĽBKOVÉ KAMERY	21
6.2.1	<i>Hľbkové kamery pracujúce na princípe Time-of-Flight (ToF)</i>	22
6.2.2	<i>Hľbkové kamery pracujúce na princípe štruktúrovaného svetla</i>	23
7	MAPOVANIE A LOKALIZÁCIA, NAVIGÁCIA A PRESKÚMAVANIE PROSTREDIA	24
7.1	SLAM	25
7.2	LOKALIZÁCIA, ODOMETRIA	26
7.2.1	<i>Robot v priestore</i>	26
7.2.2	<i>Globálny pozičný systém</i>	27
7.2.3	<i>Dead-reckoning</i>	27
7.2.4	<i>Orientačné body</i>	28
7.2.5	<i>Problémy lokalizácie</i>	29
8	VIZUÁLNA ODOMETRIA	32
8.1	DEFINÍCIA PROBLÉMU VIZUÁLNEJ ODOMETRIE, KLASIFIKÁCIA	34
8.2	PRÍZNAKY	38
8.2.1	<i>Vlastnosti príznakov</i>	39
8.2.2	<i>Typy príznakov v obraze</i>	40
8.2.3	<i>Príznaky vo vizuálnej odometrii</i>	41
8.2.4	<i>Detektory príznakov</i>	43
8.2.5	<i>Deskriptory príznakov</i>	44
8.2.6	<i>Párovanie príznakov</i>	45
8.3	RANSAC	46

8.3.1	<i>Príklad použitia metódy RANSAC</i>	49
8.4	REGISTRÁCIA MNOŽÍN BODOV V 3D PRIESTORE	51
8.5	ICP - ITERATIVE CLOSEST POINT	52
8.6	OPTIMALIZÁCIA GRAFU	53
9	MAPOVANIE	56
9.1	KATEGÓRIE ROBOTICKÉHO MAPOVANIA	56
9.1.1	<i>Priestorové a topologické mapy</i>	56
9.1.2	<i>Mapovanie zamerané na svet a na robota</i>	58
9.1.3	<i>Úplný SLAM a online SLAM</i>	59
9.1.4	<i>Iteratívne a neiteratívne metódy</i>	60
9.1.5	<i>Aktívny a pasívny SLAM</i>	60
9.1.6	<i>Jeden robot a viaceré robyty</i>	61
9.1.7	<i>Priestorový SLAM a SLAM pomocou orientačných bodov</i>	61
9.1.8	<i>Známa a neznáma korešpondencia</i>	62
9.1.9	<i>Statické a dynamické prostredia</i>	62
9.1.10	<i>Veľká a malá neistota, problém uzavretia slučky</i>	63
9.2	PRIESTOROVÉ MAPOVÉ REPREZENTÁCIE	64
9.2.1	<i>Mriežka obsadenia</i>	66
9.2.2	<i>Stromové reprezentácie, oktálový strom</i>	67
9.2.3	<i>Mračná bodov</i>	69
9.2.4	<i>Povrchové reprezentácie</i>	71
9.2.5	<i>Objektové 3D mapy</i>	72
9.3	PROBLÉMY ROBOTICKÉHO MAPOVANIA	73
9.3.1	<i>Šum</i>	73
9.3.2	<i>Viacrozmernosť</i>	73
9.3.3	<i>Problém korešpondencie</i>	74
9.3.4	<i>Dynamické prostredia</i>	74
9.3.5	<i>Problém skúmania prostredia</i>	75
9.4	ZHRNUTIE	75
10	EXPERIMENTÁLNA PLATFORMA	77
10.1	ROBOTICKÝ PODVOZOK KOBUKI	78
10.2	RIADENIE POHYBU ROBOTA	78
10.3	SNÍMAČ KINECT	79
10.4	ZÁZNAM DÁT NA EXPERIMENTÁLNEJ PLATFORME	82

11	VIZUÁLNA ODOMETRIA A MAPOVANIE PRE MALÝ MOBILNÝ ROBOT	84
11.1	PREHĽAD PREDCHÁDZAJÚCEJ PRÁCE, DOSTUPNÉ IMPLEMENTÁCIE.....	86
11.2	EXPERIMENTÁLNE DATASETY.....	88
11.3	METODIKA VYHODNOTENIA.....	89
11.3.1	<i>Absolútна chyba translácie a rotácie</i>	90
11.3.2	<i>Iteratívna chyba translácie a rotácie.....</i>	92
11.3.3	<i>Kvalita rekonštrukcie pohybu kamery</i>	93
11.4	IDENTIFIKOVANÉ PROBLÉMY DOSTUPNÝCH RIEŠENÍ	96
11.5	NAVRHNUTÉ RIEŠENIE	101
11.6	MASKA OREZANIA RGB SNÍMKU PODĽA HĽBKY	103
11.7	FILTER CLAHE	104
11.8	PARALELNÉ SPRACOVANIE	107
11.9	DETEKCIA, EXTRAKCIA A POROVNÁVANIE PRÍZNAKOV, FILTER POROVNANÝCH PRÍZNAKOV PODĽA VZDIALENOSTI	112
11.10	IMPLEMENTOVANÁ METÓDA RANSAC	117
11.10.1	<i>Koeficient ukončenia a minimálny počet iterácií.....</i>	117
11.10.2	<i>Maximálny počet iterácií.....</i>	119
11.10.3	<i>Prahová hodnota vzdialenosť pre identifikáciu inlier</i>	120
11.10.4	<i>Veľkosť bázy</i>	122
11.11	MODEL MAPY	122
11.12	EXTRAKCIA LOKÁLNEHO MODELU	126
12	ZÁVER.....	130
12.1	PRÍNOSY PRÁCE	131
12.2	PLÁNY DO BUDÚCNA	132
12.3	DISKUSIA KU TÉZAM PRÁCE.....	133
13	POUŽITÁ LITERATÚRA	134
14	PRÍLOHA 1 – MAPY PROSTREDÍ VYTVORENÉ POMOCOU NAVRHOVANÉHO RIEŠENIA	
	145	
15	PRÍLOHA 2 – PSEUDOKÓD IMPLEMENTOVANEJ METÓDY RANSAC	148
16	PRÍLOHA 3 – PUBLIKOVANÉ PRÁCE AUTORA	149

2 Zoznam použitých skratiek

SLAM – Simultaneous Localization and Mapping

RANSAC – Random Rample Consensus

AHE – Adaptive Histogram Equalization

CLAHE – Contrast Limited Adaptive Histogram Equalization

ICP – Iterative Closest Point

SIFT – Scale Invariant Feature Transform

SURF – Speeded Up Robust Features

NARF – Normal Aligned Radial Feature

BRIEF - Binary Robust Independent Elementary Features

FAST - Features from accelerated segment test

ORB - Oriented FAST and Rotated BRIEF

BRISK - Binary Robust Invariant Scalable Keypoints

AKAZE – Accelerated KAZE

PCL – Point Cloud Library

ROS – Robotic Operating System

CPU – Central Processing Unit

GPU – Graphics Processing Unit

4PCS - 4-points Congruent Sets

CMOS - Complementary Metal Oxide Semiconductor

FLANN - Fast Library for Approximate Nearest Neighbors

OpenCV - Open Source Computer Vision Library

HOG-Man - Hierarchical Optimization on Manifolds

g^2o - General Graph Optimization

TORO - Tree-based netwORK Optimizer

iSAM - incremental smoothing and mapping

3 Zoznam obrázkov

Obrázok 1 - Schéma stereovízie	18
Obrázok 2 - Obrazy z kamier stereosystému.....	20
Obrázok 3 - Princíp ToF hĺbkovej kamery	22
Obrázok 4 - Snímače Swiss Ranger SR4000 a Kinect	22
Obrázok 5 - infračervený vzor, ktorý projektuje snímač Kinect.....	23
Obrázok 6 - Obrázok 6 - Robot v 2D priestore.....	26
Obrázok 7 - Globálny pozičný systém vytvorený pomocou majákov	27
Obrázok 8 - Pravdepodobnostné zovšeobecnenie kinematiky mobilného robota	29
Obrázok 9 - Ilustrácia problému vizuálnej odometrie.	35
Obrázok 10 - Schéma priamej vizuálnej odometrie	35
Obrázok 11 - Schéma vizuálnej odometrie postavenej na príznakoch.....	36
Obrázok 12 - Vývojový diagram vizuálnej odometrie využívajúcej príznaky	37
Obrázok 13 - Vývojový diagram priamej vizuálnej odometrie	37
Obrázok 14 - 2D ku 2D výpočet posunu kamery na základe 2D príznakov.....	37
Obrázok 15 - 3D ku 3D výpočet posunu kamery na základe 3D príznakov.....	38
Obrázok 16 - Ukážka rozdielnych typov príznakov v 2D obraze.....	41
Obrázok 17 - Korešpondencie v priestore	42
Obrázok 18 - Potlačenie ne-maxím	43
Obrázok 19 - SIFT deskriptor.. ..	44
Obrázok 20 - Grafické znázornenie korešpondencií určených pomocou detektora ORB.....	48
Obrázok 21 - Porovnanie výsledkov lineárnej regresie bez a s použitím metódy RANSAC ..	50
Obrázok 22 - 4 iterácie algoritmu RANSAC pre lineárnu regresiu	50
Obrázok 23 – Globálna registrácia mračien bodov	51
Obrázok 24 - Ukážka registrácie mračien bodov pomocou metódy ICP.	53
Obrázok 25 - Inkrementálna vizuálna odometria, drift	54
Obrázok 26 - Princíp metód optimalizácie grafu.	55
Obrázok 27 - Topologická mapa prostredia	57
Obrázok 28 - Výsledok mapovania Victoria park 2D datasetu	62
Obrázok 29 - Loop closure, problém uzavretia slučky.....	63

Obrázok 30 - Obrázok mapy pomocou mriežky obsadenia.....	67
Obrázok 31 - Jednoduchý objekt uložený v štruktúre oktálového stromu.....	68
Obrázok 32 -Mapa vo formáte oktostromu pri rôznych úrovniach kompresie.....	69
Obrázok 33 - Mesh mapa vytvorená pomocou techniky KinectFusion	72
Obrázok 34 - Objektová mapa.....	72
Obrázok 35 - Vytvorený mobilný robot TurtleBot 2	77
Obrázok 36 - Rozloženie hnaných a oporných kolies na robotickom podvozku Kobuki.....	78
Obrázok 37 - Porovnanie polohovania.	79
Obrázok 38 - Závislosť hĺbkového rozlíšenia snímača Microsoft Kinect od vzdialnosti.....	80
Obrázok 39 - Vľavo hĺkový obraz, vpravo RGB obraz zo snímača Kinect.....	81
Obrázok 40 - Snímač Microsoft Kinect po odstránení plastového krytu.	81
Obrázok 41 - Robotická platforma Turtlebot 2.	83
Obrázok 42 - Absolútна chyba posunu kamery počas piatich rotácií.....	90
Obrázok 43 - Priebeh absolútnej chyby určenia orientácie.....	91
Obrázok 44 - Iteratívna chyba určenia polohy a orientácie robota pri rotácii	93
Obrázok 45 - Histogramy chyby určenia polohy a orientácie pre rotačný pohyb.	94
Obrázok 46 - Rozloženie chyby určenia polohy robota po filtrácii extrémnych hodnôt.	95
Obrázok 47 - Rozloženie chyby orientácie po aplikácii metódy 3 sigma.	95
Obrázok 48 – Problematické oblasti na snímku zo snímača Kinect, prvá časť	97
Obrázok 49 – Problematické oblasti na snímku zo snímača Kinect, druhá časť	98
Obrázok 50 - oblasti výskytu extrémnej chyby určenia globálnej polohy kamery	99
Obrázok 51 - drift polohy kamery pri statickom teste.....	100
Obrázok 52 - Bloková schéma vytvoreného riešenia	102
Obrázok 53 - Maska orezania podľa hĺbky.	103
Obrázok 54 - Porovnanie RGB obrazu v presvetenej oblasti s filtrom CLAHE	104
Obrázok 55 - Priemerný počet detegovaných príznakov na snímkach.....	105
Obrázok 56 - Chyby polohy a orientácie s filtrom CLAHE	107
Obrázok 57 - Schéma distribúcie výpočtu vizuálnej odometrie	110
Obrázok 58 – Odvodenie koeficientu orezania vzdialosti navrhnuté D. G. Lowe.....	114
Obrázok 59 - Závislosť počtu iterácií, času spracovania a podielu inliers v RANSAC.....	121
Obrázok 60 - Tri zložky mapovej reprezentácie používanej v našom riešení	125

Obrázok 61 - Priebeh absolútnej chyby určenia polohy v závislosti od veľkosti lokálnej mapy.	128
Obrázok 62 - Priebeh absolútnej chyby orientácie v závislosti od veľkosti lokálneho modelu mapy.....	128
Obrázok 63 - Porovnanie zrekonštruovaných trajektórii pohybu kamery pri rôznych veľkostiah lokálneho modelu mapy.....	129
Obrázok 64 - Mapa obývacej izby vytvorená pomocou skenovania z voľnej ruky	145
Obrázok 65 - Mapa bytu vytvorená pomocou nášho riešenia	145
Obrázok 66 - Mapa priestorov Národného centra robotiky.....	146
Obrázok 67 - Mapa chodby pri Národnom centre robotiky (NCR), pohľad zhora	146
Obrázok 68 - Mapa pri NCR, pohľad zvnútra.....	147

4 Zoznam tabuliek

Tabuľka 1 - Porovnanie priamych a nepriamych metód určenia polohy kamery z obrazu ...	36
Tabuľka 2 - Redukcia počtu pixelov RGB obrazu po orezaní podľa hĺbky.....	103
Tabuľka 3 - Celkový výpočtový čas paralelného spracovania snímku	111
Tabuľka 4 - Prehľad detektorov ORB, AKAZE, SURF a BRISK.....	112
Tabuľka 5 - Prah vzdialenosťí príznakov	115
Tabuľka 6 - Histogram vzdialenosťí príznakov pre CLAHE	116
Tabuľka 7 - Vplyv koeficientu ukončenia na čas behu algoritmu RANSAC.	118
Tabuľka 8 - Vplyv prahu určenia inlier na dĺžku spracovania metódy RANSAC	120
Tabuľka 9 - Vplyv veľkosti bázy RANSAC na kvalitu riešenia	122

5 Úvod

V posledných rokoch sme boli svedkami nástupu mobilných robotických systémov do našich životov. Vo výrobe automatické vozíky pomáhajú s presunom tovarov a výrobkov. Čistenie podlahy v domácnostiach vykonávajú robotické vysávače, deti sa hrajú s robotickými hračkami, povrch Marsu brázdia ľuďmi vytvorené vozidlá. Pri praktických aplikáciach je veľmi výhodné, keď mobilný robot disponuje určitým stupňom autonómie. Automatický vysávač musí byť schopný samostatne pokrýť podlahu celej upratovanej miestnosti, Mars rover Opportunity musí byť schopný pracovať na svojich úlohách aj v čase, keď nie je k dispozícii rádiové spojenie.

Dôležitým prvkom naozaj autonómneho mobilného robotického systému je jeho schopnosť vnímať prostredie, v ktorom sa nachádza a pohybovať sa v ňom. Aby bol robot schopný autonómne prechádzať prostredím a plniť v ňom svoje úlohy, je veľmi výhodné, aby disponoval mapou prostredia. Mapu prostredia je možné vytvárať na základe dát zo širokého spektra snímačov. V posledných rokoch výrazne získavajú na popularite vizuálne mapovacie algoritmy používajúce hĺbkové kamery. Hlavným dôvodom je nízka cena a dobrá dostupnosť takýchto zariadení. Od uvedenia lacných zariadení ako napríklad snímača Microsoft Kinect boli predstavené viaceré techniky, ktoré dokážu vytvárať modely priestoru pomocou tohto snímača. To láka robotickú komunitu ku využívaniu týchto techník pre vytvorenie mapovacích algoritmov. Problém robotického mapovania nie je však novým problémom a pri návrhu naozaj úspešných nových riešení je potrebné zohľadniť aj historický vývoj tejto problematiky.

Práca čitateľovi predstavuje témy hĺbkových kamier, vizuálnej odometrie a robotického mapovania v širšom kontexte. Z početného spektra dostupných postupov identifikuje algoritmy, ktoré je možné optimalizovať pre beh na malom mobilnom robote. Na ich základe navrhuje riešenie vizuálnej odometrie a mapovania pre túto platformu. Pre zvýšenie robustnosti voči zmenám osvetlenia je použitý filter obrazu CLAHE. Za účelom umožnenia behu s frekvenciou 30 Hz je predstavená schéma paraleлизácie na výpočtovej jednotke CPU. Na zníženie tempa rastu absolútnej chyby je predstavený mechanizmus extrakcie lokálneho modelu mapy. Ostatné časti riešenia sú dôsledne analyzované a optimalizované s ohľadom na výpočtovú zložitosť a kvalitu výstupu. Navrhnuté riešenie bolo implementované a jeho vlastnosti overené na reálnom robotickom systéme.

Štruktúra práce je nasledovná: Kapitola *Od kamier ku hĺbkovým kamerám* predstavuje technické koncepty pre získanie hĺbkového obrazu a základné typy hĺbkových kamier, ktoré sú k dispozícii na trhu. Kapitola, *Mapovanie a lokalizácia, navigácia a preskúmavanie prostredia*, predstavuje problém robotického mapovania v širšom kontexte, vysvetľuje motiváciu pre simultánnu lokalizáciu a mapovanie, poskytuje prehľad základných metód lokalizácie a problémov, s ktorými sa tieto metódy musia implementačne vysporiadať. Kapitola *Vizuálna odometria* prestavuje čitateľovi problém vizuálnej odometrie a najčastejšie používané prístupy ku rekonštrukcii pohybu kamery na základe sekvencie snímok. Špecifická pozornosť je venovaná príznakovým metódam, pre ktoré sú vysvetlené všetky základné stavebné prvky nutné pre vytvorenie funkčného riešenia. V kapitole *Mapovanie* je predstavený široký teoretický prehľad techník robotického mapovania a ich podrobná klasifikácia, najčastejšie používané priestorové mapové reprezentácie a problémy robotického mapovania. V časti *Experimentálna platforma* predstavujeme robotickú platformu, na ktorej bolo implementované riešenie vizuálnej odometrie a mapovania. Kapitola *Vizuálna odometria a mapovanie pre malý mobilný robot* popisuje návrh, implementáciu, metodiku vyhodnotenia a overenie riešenia, ktoré je prezentované v tejto práci. Záver práce zhŕňa dosiahnuté výsledky a plány do budúcnosti.

6 Od kamier ku hĺbkovým kamerám

Vizuálne systémy sa uplatňujú v širokom spektri vedných disciplín a priemyselných oblastí čo vo výraznej miere stimuluje ich rýchly vývoj. V ostatných rokoch sme boli svedkami veľkého množstva technologických inovácií, ktoré priniesli dramatické zlepšenie technických parametrov kamier, alebo ich obohatili o nové vlastnosti. Podrobny prehľad technologických inovácií kamier od objavenia digitálneho snímača obrazu nájdeme v [1].

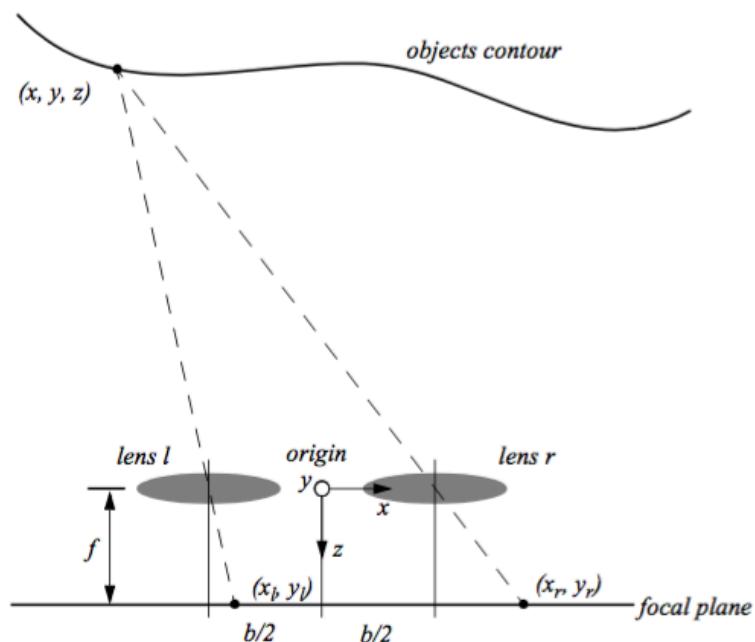
V súčasnej dobe existujú na trhu riešenia s minimálnou veľkosťou, vysokou frekvenciou snímania, vysokým rozlíšením a nízkou spotrebou za veľmi priateľné ceny. Špecializované kamerové zariadenia sú schopné dosahovať naozaj extrémne parametre snímacej frekvencie a rozlíšenia, čo prináša výrazné navýšenie dátového toku na výstupe z kamery a výrazne mení pohľad na problematiku spracovania obrazu. Výzvou už nezostáva ako dátá zosnímať, ale ako ich spracovať a získať z nich vyššie množstvo informácií. Veľkou výhodou kamier je ich prirodzená analógia s ľudským videním – dátá, ktoré snímajú sú usporiadane tak, že ich človek dokáže prirodzene interpretovať.

Výrazný posun sme zaznamenali aj v schopnosti kamier zachytávať stále širšie spektrum svetla. **Monochromatické alebo čierno-biele kamery** zachytávajú iba intenzitu odrazeného svetla, čiže individuálna hodnota farby jednotlivého pixelu je u monochromatickej kamery skalár. **Farebné kamery** poskytujú pre každý pixel hodnoty v súradničiach preddefinovaného farebného priestoru, čo býva tradične RGB. Ku každému pixelu teda merajú hodnotu pixela ako vektor. **Multispektrálne kamery** merajú viacero rozsahov spektra snímaného svetla a výstupom ich pixela je vektor pozostávajúci z výsledkov individuálnych meraní rôznych intervalov vlnovej dĺžky svetla. Multispektrálne snímky nezaznamenávajú celé spektrum scény, iba jeho zvolené časti spektra [2]. Naopak **hyperspektrálne kamery** merajú pre každý pixel celé snímateľné spektrum [3]. To umožňuje spracovanie naozaj všetkých svetelných informácií o objekte, čím sa zaoberá vedná disciplína spektroskopia [4]. Multi a hyperspektrálne zobrazovanie má svoj pôvod vo vesmírnom výskume a medicíne, ale nasadzuje sa aj v poľnohospodárstve, lesohospodárstve, ťažobnom priemysle, baníctve, potravinárstve a stavebnom priemysle. Širokému nasadeniu týchto metód snímania zatiaľ však zabráňujú vysoké ceny kamier, extrémne nároky na úložisko zosnímaných dát a vysoký výpočtový výkon pre ich užitočné spracovanie [5]. Štandardná

farebná kamera predstavuje základný vizuálny snímač, ktorý sa využíva aj v mobilnej robotike. Snímky prichádzajúce z kamery sú informačne veľmi bohaté a pomocou ich ďalšieho spracovania vieme úspešne riešiť široké spektrum úloh mobilného robota. Algoritmy, ktoré spracovávajú dátu z kamier nám poskytujú riešenia pri teleriadení, identifikácii prekážok, identifikácii pohybu kamery, pri rozpoznávaní objektov, ale aj pri mapovaní, lokalizácii a navigácii a mnohých ďalších úlohách. Ak robot disponuje nejakou formou digitálneho spracovania a interpretácie obrazu, hovoríme, že implementuje vizuálny systém. Pre lepšie pochopenie, ako dôležitý je vizuálny systém pre naozaj autonómny mobilný robot uvedieme, že pri činnosti, akou je šoférovanie, človek prijíma 83%-96% informácií práve zrakom [6].

6.1 Híbkový obraz, stereovízia

Ľudské videnie disponuje oproti štandardnej kamere ešte dôležitým aspektom – sníma aj híbku. Vďaka tomu je ľudský vizuálny systém schopný určovať vzdialenosť ku objektom a odhadovať ich rozmer. Túto schopnosť nazývame aj u človeka aj u robota **stereovízia**. Pre jej vysvetlenie uvedieme zjednodušený príklad, ktorý používa 2 rovnaké kamery, ktoré majú optické osi umiestnené paralelne.



Obrázok 1 - schéma stereovízie [7]

Vzdialenosť medzi kamerami je označená b , optické stredy kamier l a r , a f je vzdialenosť medzi stredmi šošoviek a rovinou ostrenia (focal plane). Súradnice zmeraného bodu (x, y, z) Môžeme potom určiť pomocou nasledujúceho vzťahu:

$$x = b \frac{(x_l + x_r)/2}{x_l - x_r}; \quad y = b \frac{(y_l + y_r)/2}{x_l - x_r}; \quad z = b \frac{f}{x_l - x_r}$$

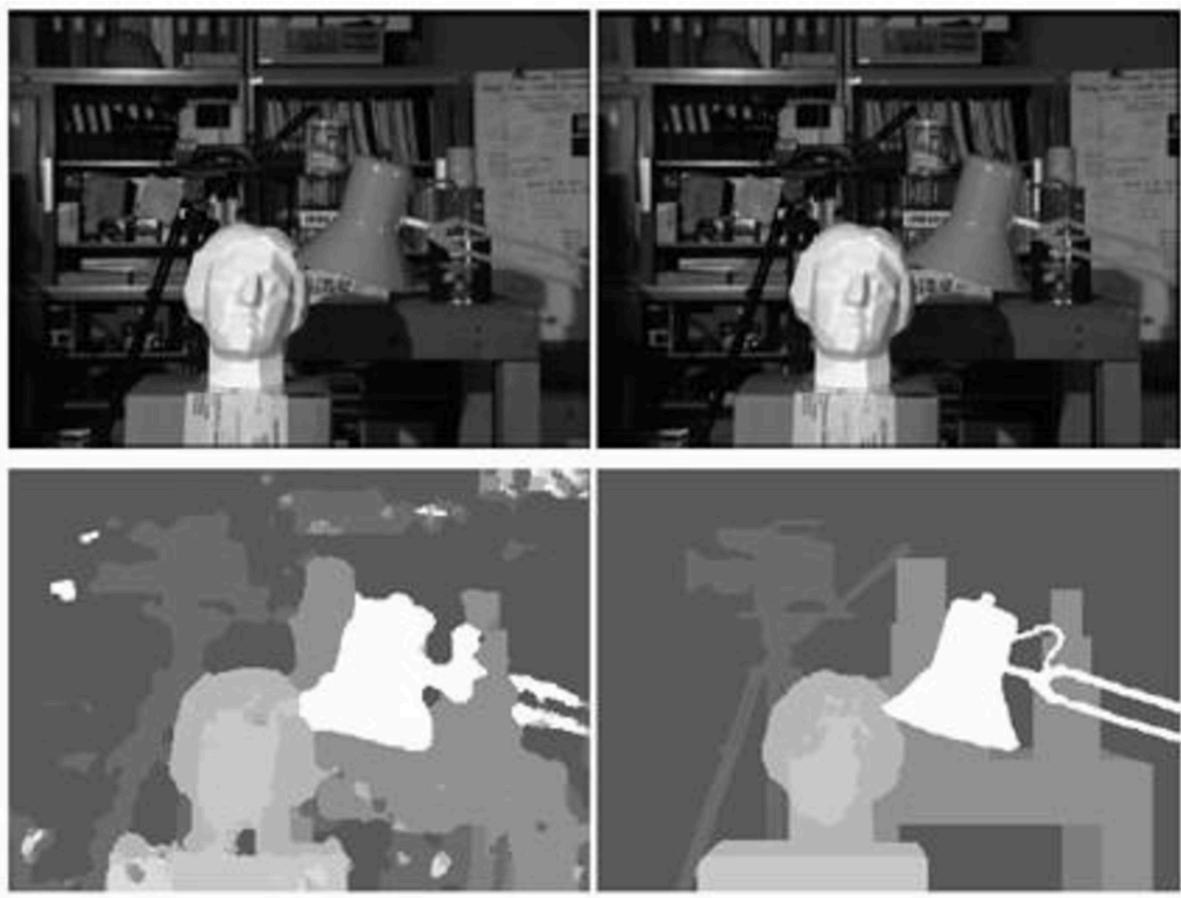
Rozdiel $x_l - x_r$ v obrazových súradniciach sa nazýva **disparita**. Z disparity sme schopní určiť hĺbku v snímanom bode. Z týchto rovníc zároveň vyplýva :

- Vzdialenosť je inverzne proporcionálna ku disparite. Objekty, ktoré sa nachádzajú bližšie ku kamerám je možné zmerať presnejšie ako objekty vo väčšej vzdialosti.
- Disparita je proporcionálna ku b . S rastúcou vzdialenosťou b rastie aj presnosť určovania snímanej hĺbky.
- Keď sa b zvyšuje, niektoré objekty môžu zmiznúť z obrazu jednej z kamier. Vzdialenosť od týchto objektov nebude možné pomocou stereovízie zmerať.

V praxi je veľmi ťažké vytvoriť perfektne zarovnaný páár kamier, preto je kamery nutné kalibrovať. Pre úspešnú kalibráciu potrebujeme zmerať pozíciu jedného konkrétneho bodu v oboch obrazoch. Tento fundamentálny problém, teda identifikácia párov bodov v oboch obrazoch ktoré predstavujú jeden bod v reálnom priestore sa v tomto kontexte nazýva **problémom korešpondencie**, ktorý je bližšie popísaný v príslušných častiach tejto práce.

Po zosnímaní obrazov z oboch kamier sa pokračuje softvérovým spracovaním obrazu, ktoré môže využívať rôzne metódy výpočtu vzdialnosti z dvoch obrazov. V najrozšírenejších, **príznakových metódach** sa problém korešpondencie rieši pomocou extrakcie príznakov a ich vzájomného párovania. Kvalita vytvorenej disparitnej mapy závisí od podmienok snímania a počtu detegovaných príznakov a preto je vo väčšine prípadov dátu nutné interpolovať [8]. Alternatívne metódy využívajú priame porovnávanie oblastí obrazu alebo sa snažia priamo dosadiť 3D povrch do disparitnej mapy [9].

Výsledný obraz nazývame **hĺbkový obraz**, keďže pre každý pixel predstavuje vzdialenosť fyzickej oblasti v snímanej scéne od roviny kamery.



Obrázok 2 - Hore obraz z ľavej a pravej kamery stereosystému, vľavo dole disparitná mapa, vpravo dole výsledný hĺbkový obraz po interpolácii dát [8]

Hĺbkovú mapu je zároveň možné v stereovízii obohatiť o farbu, keďže hĺkové pixely priestorovo korešpondujú s pixelmi, ktoré zosnímali kamery stereopáru. V tomto prípade hovoríme, že výstupom kamery sú **RGBD dátá**. Pre každý pixel v týchto dátach dostávame 4 hodnoty, **farebné zložky - r,g,b** a **vzdialenosť - d**. Koncept stereovízie je možné aplikovať aj na systémy, ktoré obsahujú viac ako 2 kamery. Motiváciou pre pridanie ďalšej kamery môže byť ďalšie spresnenie určenia hĺbky na určitom rozsahu alebo pokrytie väčšej časti scény. Tento koncept využíva napríklad priemyselná stereovízna kamera Bumblebee XB3 1394b [10], ktorá používa 2 pri sebe blízko umiestnené kamery pre presný výpočet hĺbkového obrazu v blízkosti kamery a 1 prídavnú kamery s vysokým rozlíšením pre presné určovanie hĺbky obrazu vo vyššej vzdialenosťi od kamery. V súčasnej dobe existuje na trhu veľké množstvo stereovíznych kamier [11] [12] [13], ktoré poskytujú nízku spotrebu, malé fyzické rozmery a relatívne vysoké presnosti určovania vzdialenosťí, čo umožňuje ich široké využitie v mobilnej robotike. Pomocou viacerých vysokorýchlosných kamier sa implementujú lokalizačné systémy pre

vnútorné použitie, ktoré sú schopné vysokých presnosťí [14]. Úspešnosť určovania hĺbky v stereovíznom systéme je však limitovaná viacerými faktormi. Na nedostatočne alebo nevhodne osvetlenej alebo veľmi málo štruktúrovanej scéne nebude možné určiť dostatočné množstvo príznakov a určená disparita bude nekompletná. Tento problém je do značnej miery možné riešiť interpoláciou dát, ale v tom prípade hodnoty vystupujúce z vizuálneho systému nereprezentujú presné merania. Navyše, keďže stereovízia používa klasický obraz, veľkým problémom budú aj odrazivé povrhy ako sú napríklad zrkadlá. Výhodou stereovízie je, že sa jedná o **pasívne snímanie (passive sensing)** [7] hĺbky. To znamená, že pre získanie merania sa využíva iba energia prostredia, ktorá sa už v prostredí nachádza (v tomto prípade svetlo) a snímač s prostredím neinteraguje.

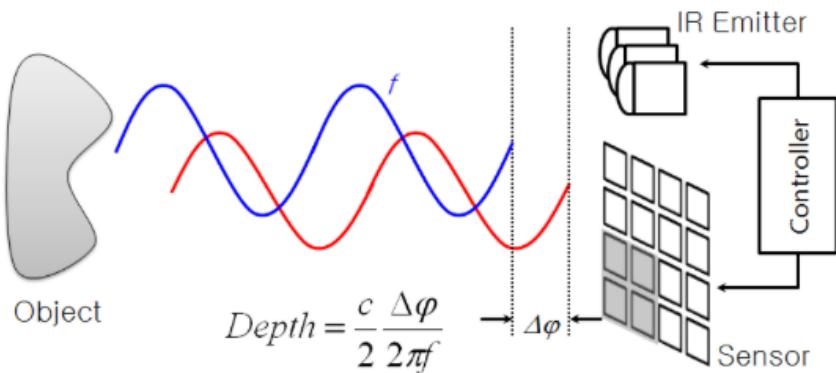
6.2 Hĺbkové kamery

Pokroky v snímaní vzdialenosť umožnili vznik nového typu snímača – hĺbkovej kamery. Hĺbkové kamery sú snímače schopné merať vzdialosť. Ich výstupom je sekvencia snímok, ktoré sú organizované v pixelovej mriežke. Na rozdiel od klasických kamier však pre každý pixel merajú vzdialosť od roviny snímania, hĺbku – depth. Hĺbkové kamery pracujú na viacerých princípoch, ktoré predstavíme v nasledujúcom teste, ale ich spoločným prvkom je implementácia **aktívneho snímania (active sensing)** [7]. Znamená to, že pre svoju činnosť vysielajú energiu (svetlo) do prostredia, ktoré snímajú. Vo svetovej literatúre sa stretнемe aj s označením RGB-D kamery. Táto skratka je akronymom pre Red, Green, Blue and Depth, čiže pre kanály, ktoré kamera poskytuje pre každý zosnímaný pixel obrazu.

Odlišnosťou hĺbkovej kamery od iných typov snímačov vzdialnosti je spôsob, akým sú výstupné vzdialosti usporiadané vo výstupe. Hĺbková kamera vracia podobne ako RGB kamera sekvenciu snímok, v ktorých sú jednotlivé pixely usporiadané do pixelovej mriežky a súradnica Z označuje vzdialosť od roviny snímania. Naopak iné 3D snímače vzdialnosti, rotačné laserové skenery, ako napríklad Velodyne VLP-16, organizujú dátá z jedného snímku – jednej otáčky podľa aktuálnej hodnoty natočenia rotačnej časti v čase snímania. Nameraná vzdialosť rotačného laserového skenera je teda v polárnych súradničiach.

6.2.1 Híbkové kamery pracujúce na princípe Time-of-Flight (ToF)

Híbkové kamery Time-of-Flight pracujú na princípe merania fázového oneskorenia odrazeného infračerveného (IR) svetla. Princíp fungovania zobrazuje Obrázok 3. Infračervená vlna, zobrazená červenou, je vyslaná na objekt. Snímač následne zmeria odrazenú infračervenú vlnu, zobrazenú modrou farbou.



Obrázok 3 - Princíp ToF híbkovej kamery. Fázové oneskorenie medzi vyslaným a prijatým svetlom umožní vypočítanie hĺbky pre každý bod obrazu [15]

Pomocou merania fázového posunu medzi vyslanou a prijatou svetelnou vlnou je možné určiť vzdialenosť k meranému objektu. Chyby, ktorými takýto spôsob snímania trpí, sú najmä systematické chyby – chyba infračervenej demodulácie, chyba integrácie času, chyba zapríčinená teplotou, chyba zapríčinená farebnosťou scény a pohybovým rozmazaním (motion blur). Podrobné predstavenie ToF kamier a analýzu problémov tohto princípu snímania a ich riešení nájdeme v [16].

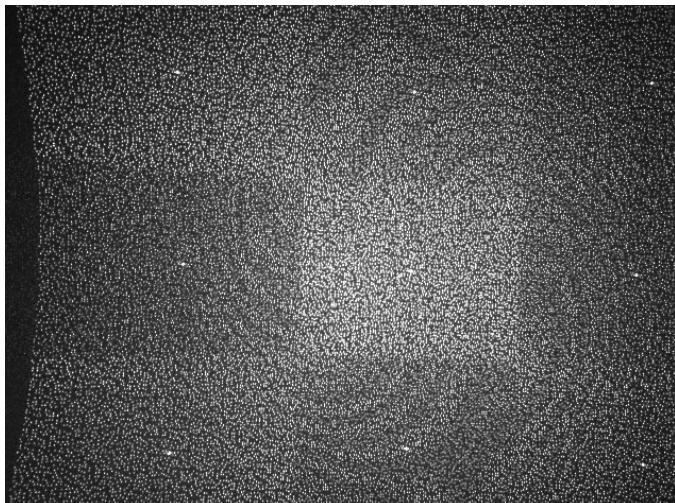


Obrázok 4 - Vľavo snímač Swiss Ranger SR4000, jedna z prvých na trhu dostupných híbkových priemyselných kamier pracujúca na princípe ToF [16], vpravo snímač Kinect v druhej verzii, tiež ToF kamera [17]

6.2.2 Hĺbkové kamery pracujúce na princípe štruktúrovaného svetla

Alternatívny princíp, na ktorom pracujú hĺbkové kamery je princíp projekcie a spätného snímania tzv. štruktúrovaného svetla. Na tomto princípe pracuje aj hĺbková kamera Microsoft Kinect vo svojej prvej verzii a práve na tejto kamere bude tento princíp podrobne vysvetlený. Kamera Kinect sa skladá z infračerveného projektora, ktorý na snímanú scénu projektuje štruktúrovaný infračervený vzor. Súčasne, klasická CMOS kamera vybavená infračerveným filtrom túto scénu sníma. Obrazový procesor snímača Kinect je následne schopný na základe deformácií zosnímaného vzoru oproti projektovanému a známej vzdialosti medzi IR projektorom a kamerou určiť hĺbku snímanej scény. Dôležitým prvkom tejto technológie je, že snímaná hĺbka je určená od roviny snímania. Na rozdiel od laserových skenerov, ktorých výstup sa vyjadruje tradične v polárnych súradniach. Snímač Kinect je vybavený aj RGB kamerou, čiže výstupom nie sú iba hĺbkové informácie, ale kompletne RGBD dátá. Pre zvýšenie rýchlosť výpočtu hĺbkového obrazu používa tento snímač viaceré postupy, ktoré je možné nájsť v prislúchajúcich patentoch a výstupoch reverzného inžinierstva [18] [19] [20].

Na rozdiel od technológie Time-of-Flight, ktorá vyžaduje dizajn špeciálneho snímacieho čipu a jeho spracovávajúcej elektroniky, snímanie hĺbky pomocou



štruktúrovaného svetla je možné ľahko reprodukovať aj v laboratórnych podmienkach. Ak nepotrebuje plne integrované riešenie, akým je napríklad snímač Kinect, môžeme hĺbkový snímač pracujúci na tomto princípe vytvoriť aj v laboratórnych podmienkach, ako v [21].

Obrázok 5 - infračervený vzor, ktorý projektuje snímač Kinect na snímanú scénu. Vzor je tvorený bodkami organizovanými vo viacerých segmentoch. Spracovanie začína identifikáciou 9 segmentov pomocou najjasnejších bodiek, ktoré sú umiestnené v strede každého segmentu. Takýto spôsob spracovanie prináša ďalšie zrýchlenie pri výpočte hĺbky obrazu [22].

7 Mapovanie a lokalizácia, navigácia a preskúmavanie prostredia

Robotické mapovanie je jedným z najintenzívnejšie skúmaných problémov v rámci modernej robotiky. Mapovanie rieši problém získavania modelov prostredí pomocou robotov a je základným prvkom, ktorý musí obsahovať autonómny mobilný robot. Aby mohol **autonómny mobilný robot** splniť zdanlivo jednoduchú úlohu, preniesť predmet z bodu A do bodu B, musí byť schopný zodpovedať 3 základné otázky:

Aké je prostredie v ktorom sa nachádzam?

Kde sa v tomto prostredí nachádzam?

Ako sa dostanem na iné miesto v tomto prostredí?

Tieto 3 otázky priamo korešpondujú s troma hlavnými problémami robotického mapovania. **Problém mapovania** predstavuje spôsob, ako získavať dostatočne presné reprezentácie prostredia. Pre určenie svojej polohy v tomto prostredí musí robot vedieť vyriešiť **problém lokalizácie**. Aby bol robot schopný presunúť sa v tomto prostredí, musí vedieť navrhnúť svoju trasu do cieľa a vygenerovať a vykonať sériu pohybových príkazov, ktoré mu umožnia sa do tohto cieľa presunúť. Tento problém označujeme ako **problém navigácie**. Pridruženým problémom je **problém preskúmavania prostredia**. Tento problém rieši, ako efektívne prechádzať nové prostredia za účelom získania ich mapy. Predchádzajúci vývoj riešení pre mapovanie ukázal, že problémy lokalizácie a mapovania nie je možné v kontexte mapovania rozdeliť a musíme ich riešiť spoločne. Povedzme, že sa mobilný robot nachádza v bode A. Zosníma prostredie vo svojom okolí a následne sa presunie do bodu B, kde tak isto zosníma svoje okolie. Pre spojenie týchto dvoch snímok dohromady a vytvorenie mapy, ktorá bude obsahovať aj bod A aj bod B, musíme vedieť určiť, aká je vzájomná poloha týchto dvoch snímok – čiže musíme vyriešiť problém lokalizácie.

7.1 SLAM

Súčasné vedecké kolektívy sa sústredujú na vyriešenie problému **simultánnej lokalizácie a mapovania** (Simultaneous Localisation and Mapping - SLAM). Termín SLAM bol pôvodne určený pravdepodobnostným technikám riešenia lokalizácie a mapovania, avšak v súčasnej dobe sa používa aj pre metódy, ktoré pravdepodobnosti využívajú málo alebo vôbec. Pre zjednodušenie terminológie budeme ďalej v texte označovať ako SLAM riešenia aj riešenia, ktoré nepoužívajú pravdepodobostné techniky. Ak chceme problém SLAM formálne definovať, potom nech v každom jednotlivom časovom kroku procesu mapovania $t \in \{0,1,2 \dots \infty\}$ je:

x_t - stavový vektor obsahujúci informácie o polohe robota

u_t - vektor polohových (kontrolných, ovládacích) príkazov, ktoré robot prijal v časovom kroku $t - 1$ a vykonal ich, aby prešiel do stavu x_t

z_t - vektor senzorických vnemov robota

m_t - podoba vytvorennej mapy

Potom je problém SLAM možné formálne vyjadriť ako:

$$(x_t m | z_t, u_t)$$

Tento zápis môžeme slovne vyjadriť ako: „Ako určovať polohu robota a mapu priestoru na základe senzorických meraní a príkazov, ktoré robot prijal pre ovládanie“. Tento formálny zápis problému SLAM je základným východiskom pre pochopenie princípov pravdepodobnostnej robotiky [23] [24].

Aj napriek dramatickým pokrokom v tejto oblasti v posledných rokoch má robotické mapovanie pred sebou ešte veľa výziev, ktoré musia byť vyriešené a implementované na to, aby sme dokázali napodobiť to, čo je v mnohých prípadoch ľuďom prirodzené. Dynamické prostredia, príliš štruktúrované prostredia, neštruktúrované prostredia a veľké prostredia stále predstavujú významné prekážky v implementácii dostatočne presného, efektívneho a lacného riešenia, ktoré by bolo možné univerzálne aplikovať na rôzne kategórie robotov.

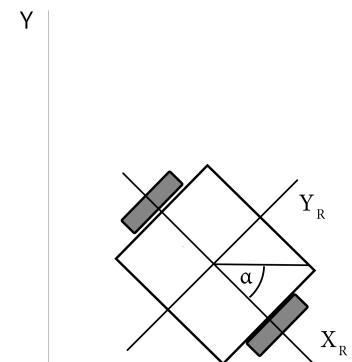
7.2 Lokalizácia, odometria

Ako sme uviedli v predchádzajúcim texte, problém robotického mapovania je úzko spätý s problémom lokalizácie. Aby bol mobilný robot schopný určiť cestu do svojho cieľa, musí vedieť, kde sa nachádza. Ak chce robot vytvoriť mapu priestoru, v ktorom sa pohybuje, musí vedieť, ako priestorovo usporiadať svoje merania zo snímačov, čiže musí vedieť svoju polohu. Proces určovania trajektórie mobilného robota nazývame aj **odometriou**. Odometria však popisuje len jeden čiastkový problém lokalizácie, t.j.: ako určovať polohu robota na základe predchádzajúcej polohy robota a dostupných senzorických meraní. V tejto kapitole si predstavíme najčastejšie používané spôsoby lokalizácie mobilného robota.

7.2.1 Robot v priestore

Pre určenie polohy robota v mape musíme poznáť súradnicový systém jej geometrickej reprezentácie. V praxi sa používa *Kartézsky súradnicový systém*, a to v dvojdimenziuálnej aj trojdimenziuálnej podobe. Geometrický vzťah robota a prostredia je v ňom daný polohou a orientáciou robota. V **2D priestore** je poloha pre mobilný robot definovaná pomocou dvoch súradníc **x**, **y** a jednej hodnoty pre orientáciu α . V **3D priestore** pomocou troch súradníc **x**, **y**, **z** a troch hodnôt φ , ϑ , ψ pre určenie rotácie robota okolo každej z 3 osí v priestore. Polohu robota je možné vyjadriť aj pomocou kvaterniónov, ktoré sú jednoduchšie na spracovanie alebo pomocou matíc homogénnych transformácií, ktoré boli použité aj v tejto práci. Oproti uhlom natočenia okolo osí totiž netrpia problémom nazývaným gimbal lock.

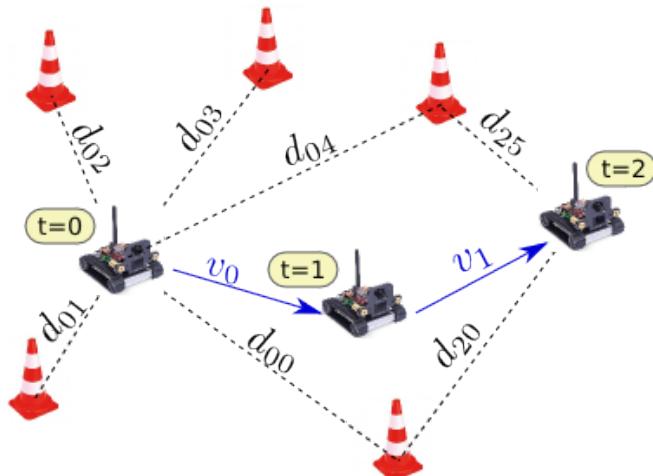
Vzájomnú polohu robota a prostredia vieme vyjadriť vzťahom robot-prostredie, ale aj naopak. Preto rozlišujeme globálny a lokálny súradnicový systém. Globálny súradnicový systém používa ako základ prostredie, lokálny robot. Na obrázku 6 môžeme vidieť **globálny súradnicový systém X a Y**, aj **lokálny súradnicový systém robota X_R, Y_R**. Počiatok globálneho súradnicového systému bol v našich experimentoch umiestnený vždy do bodu [0,0,0].



Obrázok 6 - Robot v 2D priestore. X a Y označujú globálny súradný systém, X_R a Y_R systém robota.

7.2.2 Globálny pozičný systém

V mnohých scenároch mobilnej robotiky potrebujeme vedieť pozíciu robota presne, a to v každom čase. V prostredí bez snímačov globálnej polohy sa však jedná o netriviálny problém. Typickým reprezentantom globálneho pozičného systému je GPS, avšak ten funguje spoľahlivo iba vo vonkajšom prostredí. Vo vnútornom prostredí môžeme použiť globálnu senzorickú sieť s použitím infračervených, sonarových, laserových, vizuálnych alebo rádiových vysielačov - majákov (beacons). Ako výborné riešenie sa ukázalo použitie skupiny kamier. Ak poznáme polohu majákov, takéto vnútorné riešenie nám umožní priamo pristupovať ku súradniciam robota po vyriešení jednoduchej matematickej úlohy. Obrovskou výhodou globálneho pozičného systému je, že jednotlivé určenia polohy robota sú od seba nezávislé – každý nový pokus o určenie polohy robota je izolovaný od predchádzajúcich určení a jeho chyba je závislá hlavne na viditeľnosti majákov, kvalite prijatého signálu a presnosti merania vzdialenosť medzi majákmi a presnosti výpočtu polohy.



Obrázok 7 - Globálny pozičný systém vytvorený pomocou majákov a pohyb robota v ňom [25]

7.2.3 Dead-reckoning

Ďalším riešením lokalizačného problému je **dead-reckoning**, alebo takzvaná geometria korytnačky. Na začiatku je nutné poznať počiatočnú polohu a orientáciu korytnačky (robota). Pre určenie každej nasledujúcej polohy je potrebné poznať predchádzajúcu polohu a pohybové príkazy, ktoré robot prijal. Pre všetky ďalšie presuny sa poloha robota aktualizuje až po prijatí spätej odozvy od jeho pohybového systému. Pojem dead-reckoning je námornícky pojem zo 17. storočia, ktorý popisuje, že lode, ktoré nemali moderné námornícke

vybavenie, sa museli spoliehať na vektorové sčítavanie častí ich kurzu pre určenie ich aktuálnej polohy. Táto metóda lokalizácie sa stáva po dlhšom čase nepresnou a to najmä z dôvodov:

1. presnosť aktuálneho určenia pozície robota je zaľažená zdedenou chybou z predchádzajúcich meraní - robot vyrazil z iného miesta, ako si myslel
2. chybou vykonania pohybového príkazu – napríklad: pohybový systém robota mal prejsť 1m, ale prešiel iba 0,995 metra
3. chybou určenia pozície v aktuálnej iterácii - počas pohybu robota došlo ku prešmyku kolesa a robot sa pri tom otočil o pol stupňa

Najvýraznejšie chyby pri dead-reckoningu sú chyby orientácie, pretože tie majú najvýznamnejší podiel na presnosti určenia pozície robota.

Dead-reckoning používame napríklad pre implementáciu **kolesovej odometrie mobilného robota (wheel odometry)**. Robot v každom časovom kroku prijme pohybový príkaz a vykoná ho, ale pre výpočet finálnej polohy robota nepoužijeme pohybový príkaz, ale meranie z inkrementálnych snímačov, ktorými je vybavený podvozok robota. Výsledná poloha robota bude vypočítaná na základe integrácie predchádzajúcich meraní v predchádzajúcich časových intervaloch.

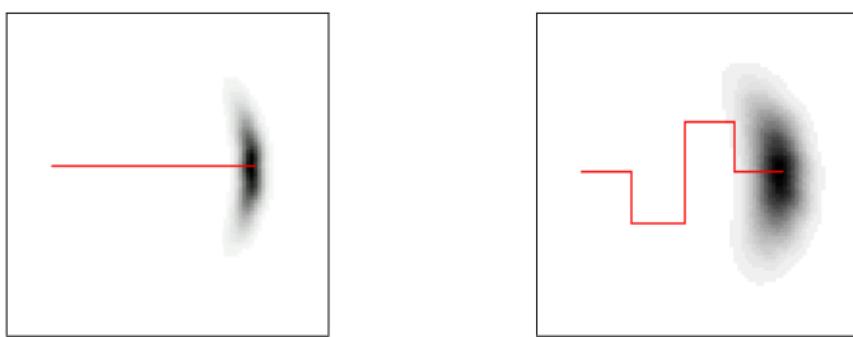
7.2.4 Orientačné body

Naozaj autonómny robot musí byť schopný vykonávať úlohu mapovania aj v prostredí, ktoré predtým nenaštívil a v ktorom nie je možné vytvoriť globálny pozičný systém - napríklad sonda Curiosity na Marse. Mapovacie algoritmy v týchto situáciách nemajú k dispozícii majáky, ktorých polohu v priestore vopred poznajú, dead reckoning môže byť vo voľnom teréne extrémne nepresný z dôvodu prešmyku kolies a preto musia mobilné roboty nájsť iné riešenie, ktoré im umožní určovať polohu robota.

Vzorom pri riešení tohto problému môže byť model, ktorý používa človek. Predstavme si, že nás náš priateľ prevádzza po meste, v ktorom sme ešte neboli, pričom vieme, že sa budeme musieť samostatne vrátiť späť. Pravdepodobne sa budeme snažiť zapamätať si unikátné budovy, stromy a iné objekty - orientačné body, ktoré nám pomôžu sa pri ceste naspať zorientovať. V robotickom mapovaní sa tieto body nazývajú **orientačné body (landmarks)**. Sú to teda miesta (body, priamky, objekty), ktoré sa dajú v senzorickom meraní

opakovane alebo dokonca unikátne detegovať a ktoré korešpondujú s určitou vlastnosťou mapovaného priestoru v určitej polohe v súradnicovej sústavy mapy.

Mapovacie algoritmy, ktoré riešia problém mapovania a problém lokalizácie naraz teda využívajú orientačné body pre určenie polohy. Rozdielom oproti použitiu globálneho súradnicového systému je *zdedená neistota* v polohe každého nasledujúceho orientačného bodu v procese mapovania. Pre určenie pozície nového orientačného bodu totiž používame informáciu o polohe, ktorú aktuálne máme a ktorá je taktiež zaťažená chybou.



Obrázok 8 - Pravdepodobostné zovšeobecnenie kinematiky mobilného robota. Každá červená čiara reprezentuje dráhu, po ktorej bol robot navigovaný. Čím tmavšia plocha na mape, tým väčšia je pravdepodobnosť, že sa v nej robot nachádza. [25]

7.2.5 Problémy lokalizácie

Problém lokalizácie nie je iba problémom určenia presných absolútnych súradníc robota vzhľadom na prostredie (pri systéme GPS vzhľadom na planétu), ale aj problém určenia relatívnej pozície voči orientačnému bodu.

Senzorický a motorický systém robota hrá veľkú úlohu pri všetkých spôsoboch lokalizácie. Z dôvodov nepresnosti zozbieraných dát a ich nekompletnosti sa problém lokalizácie robota musí vysporiadať s rôznymi problémami vo forme neistôt. Najuspokojivejšie výsledky v tejto oblasti dosahujú algoritmy, ktoré sa snažia modelovať zdedenú neistotu cez pravdepodobnostné metódy. V nasledujúcich podkapitolách si predstavíme hlavné zdroje neistôt v lokalizácii robota.

7.2.5.1 Senzorický šum

Senzory sú základnými stavebnými prvkami vnímania robota, čo im dáva kritickú dôležitosť. **Senzorický šum** reprezentuje limitáciu konzistentnosti senzorických meraní v rovnakých

podmienkach prostredia a zároveň počet užitočných bitov v bitoch zozbieraných informácií. Častým zdrojom senzorického šumu sú prvky prostredia, ktoré nie sú zachytené v reprezentácii robota.

Napríklad - vizuálny navigačný systém pre použitie v budovách môže používať farebný systém detegovaný farebnou CCD kamerou. Ak je slnko za mrakom, osvetlenie interiéru je iné ako keď jasne svieti slnko. Výsledok tohto javu je ten, že namerané hodnoty odtieňa farby nie sú konštantné. Farebný CCD snímač dodáva z pohľadu robota zašumený obraz z neznámych dôvodov. Hodnoty odtieňov získané z CCD snímača budú nepoužiteľné, pokiaľ robot nedokáže vo svojej reprezentácii prostredia zahrnúť aj intenzitu a farbu osvetlenia.

Závislosť na osvetlení je iba jeden z príkladov zjavného šumu vizuálneho senzorického systému robota. Ďalšími zdrojmi šumu sú napr. jitter, signal gain, blooming a blurring [7].

7.2.5.2 Senzorické vyhľadzovanie

V ľudskom vnímaní je bežné, že jednotlivé výstupy nášho zmyslového systému sú unikátne. Inak povedané, každé miesto, ktoré človek navštívi zvykne vyzeráť odlišne a preto sa ľudia vedia veľmi efektívne v prostrediach zorientovať a orientovať. Naopak, v robotike sú bežné aj navzájom veľmi podobné senzorické merania. Ak je robot napríklad vybavený jednoduchým laserovým skenerom, je možné, že jeho pozorovania budú takmer rovnaké vo viacerých miestach mapy a robot nie je schopný ich odlišiť bez informácií o kontexte, v ktorom ich zmeral. Z tohto dôvodu pracujú tradičné lokalizačné postupy nad bázou dát pochádzajúcich z viacerých meraní. Túto neodlísiteľnosť jednotlivých meraní nazývame **senzorické vyhľadzovanie (sensoric aliasing)**. Meno pochádza z teórie spracovania signálov. V teórii spracovania signálov je **aliasing** jav, ktorý spôsobuje, že dva signály sa stanú po vzorkovaní neodlísiteľnými. Môže to znamenať aj to, že zrekonštruovaný signál z diskrétnej podoby bude odlišný ako pôvodný, vzorkovaný signál.

7.2.5.3 Šum efektorov

Kolesová odometria je proces transformácie informácií o pohybe kolies robota na informácie o relatívnej zmene polohy robota počas posledného pohybu. Je to najjednoduchšia implementácia lokalizácie pomocou dead reckoningu. Pre svoje fungovanie používa snímače pohybu kolies robota, ktorých výstupy sú spracované v systéme určenia pozície robota. Kolesové senzory však trpia určitým množstvom šumu. Pridanie dodatočným senzorov nám

možno pomôže spresniť informácie o polohe robota a zvýšiť presnosť nášho merania, ale kľúčový problém - šum, sa bude v určitej miere objavovať stále.

Medzi hlavné zdroje problémov so šumom vstupných lokalizačných dát môžeme považovať napríklad limity rozlíšenia meraní alebo rozlíšenia kroku času, vady mechanickej konštrukcie robota alebo obmedzenia kinematického modelu pohybu robota pre dané prostredie [26].

8 Vizuálna odometria

Pod vizuálnou odometriou (visual odometry, VO) rozumieme proces získavania trajektórie mobilného agenta (roberta, vozidla, človeka) pomocou jednej alebo viacerých kamier v 3D priestore. Jej uplatnenie nájdeme v širokom spektri oblastí od robotického mapovania, 3D rekonštrukcie, rozšírenej a virtuálnej reality až po automobilový priemysel. Pojem bol prvýkrát použitý v roku 2004 v článku o orientačných bodoch (landmarks). Vybraný bol podľa svojej podobnosti s kolesovou odometriou (wheel odometry), ktorá určuje polohu robota na základe integrácií počtu otočení kolies podvozku a kinematického modelu podvozku. Podobne, vizuálna odometria sleduje zmeny obrazu vyvolané pohybom medzi jednotlivými snímkami a používa ich pre výpočet trajektórie. Základnými predpokladmi pre použitie vizuálnej odometrie sú:

- **primerané osvetlenie scény** – potrebné pre získanie užitočných dát z kamery
- **statická scéna** – určenie zmeny pohybu kamery medzi jednotlivými snímkami musí prebiehať voči statickému objektu. Ak sú na scéne umiestnené aj dynamické objekty, je potrebné ich odfiltrovať.
- **snímky prichádzajú s dostatočne vysokou frekvenciou.** Kedže určenie polohy prebieha prostredníctvom asociácie užitočných dát medzi dvoma za sebou nasledujúcimi snímkami, je potrebné, aby sa tieto snímky v dostatočnej mieri prekrývali. Na snímkach by sa nemalo prejavovať pohybové rozmazanie (motion blur).
- **snímky obsahujú dostatok textúry alebo štruktúry hĺbkového obrazu.** Je spravidla nemožné určiť posun kamery pri zábere na scénu s nedostatkom užitočných informácií, napríklad na bielu stenu. Dva za sebou nasledujúce snímky v tomto prípade jednoducho nenesú dostatok informácií.

Moderné riešenia vizuálnej odometrie prinášajú vyššiu presnosť ako kolesové odometrie, a to najmä vďaka nezávislosti od prešmykov kolies, mechanických výrobných nedostatkov robotického podvozku a nedokonalosti riadenia. Moderné mobilné roboty implementujú systémy vizuálnej odometrie ako doplnok ku GPS alebo gyro-odometrii. V prípade výpadku GPS, napríklad pri pohybe vo vnútornom prostredí, je vizuálna odometria totiž jedným z najlepších spôsobov určenia polohy robota.

Problém rekonštrukcie pohybu kamery a získavania trojrozmernej reprezentácie priestoru je v oblasti počítačového videnia (computer vision) známy aj ako problém získavania **štruktúry z pohybu (structure-from-motion)**. Oproti vizuálnej odometrii je tento problém viac všeobecný a zahŕňa aj získavanie modelu 3D reprezentácie priestoru zo sekvenie zoradených alebo nezoradených obrázkov. **Vizuálna odometria** sa zameriava iba na výpočet transformácie pohybu kamery medzi dvoma za sebou idúcimi snímkami. **Vizuálny SLAM** sa na rozdiel oproti vizuálnej odometrii zameriava na kompletnú rekonštrukciu trajektórie robota a tvorí 3D mapu prostredia. Oproti vizuálnej odometrii prináša schopnosť uzavretia slučky – ak robot navštívi oblasť sveta, ktorú už predtým zosnímal, je schopný tento fakt zistiť a pomocou neho späťne upraviť mapu sveta tak, aby čo najviac zodpovedala realite. Spravidla tak učiní pomocou použitia informácií o pravdepodobnostných rozloženiacach jednotlivých meraní pozície alebo dát, ktoré ku týmto meraniam viedli. Vizuálna odometria sa sústredí na čo najpresnejšie získanie modelu lokálnej mapy a lokálnej trajektórie. Vizuálny SLAM sa snaží zrekonštruovať čo najpresnejšiu mapu prostredia, cez ktoré sa robot pohyboval. Keďže algoritmy vizuálneho SLAM zahŕňajú aj výpočet aktuálnej pozície robota, môžeme ich použiť aj ako vizuálnu odometriu. Vizuálny SLAM je spravidla výpočtovo náročnejší, ale prináša väčšiu mieru globálnej konzistentnosti určenia pozície robota. Vizuálny SLAM rieši problém rekonštrukcie aj z dvoch nie po sebe idúcich snímkov, vizuálna odometria susednosť snímkov naopak vyžaduje.

Na najvyšom stupni môžeme rozdeliť postupy vizuálnej odometrie podľa použitej konfigurácie kamier do dvoch hlavných skupín:

- **Vizuálna odometria pomocou jednej kamery (monocular visual odometry, monocular SLAM)** - 3D štruktúra scény aj pohyb kamery sú počítané súčasne, v každej iterácii algoritmu sú spracované nové 2D dátá.
- **Vizuálna odometria pomocou viacerých kamier (stereo visual odometry) alebo hĺbkovej kamery (RGBD alebo depth camera odometry)** – pri tomto prístupe sa budú priamo zosnímajú 3D dátá alebo je obraz z viacerých kamier prepočítaný z 2D do 3D. Určenie pohybu kamery následne spravidla prebieha pomocou porovnania 3D štruktúry scény v dvoch za sebou idúcich iteráciách.

Problém určenia trajektórie robota pomocou kamier bol formulovaný a prvýkrát úspešne vyriešený v [27]. Táto práca využívala pre určenie 3D scény pohybujúcu sa kameru, ktorá vedela zastavovať na deviatich predefinovaných pozíciah. Pohyb robota prebiehal v sekvencií: pohyb, zastavenie, snímanie scény, výpočet scény. Pri každom zastavení robot urobil 9 snímok, v ktorých následne detegoval rohy (táto práca popisovala aj jeden z prvých predstavených detektorov rohov v obrazu), medzi ktorými pomocou známych vlastností kamerového systému, epipolárnej geometrie a filtrácie nekonzistentných meraní medzi kamerami určoval 3D súradnice týchto rohov. Finálny posun kamery bol určený na základe korešpondencií v najpravdepodobnejších rohoch nájdených medzi dvoma pozíciami robota. Výpočet bol váhovaný podľa vzdialenosťí od robota. Dôvodom bolo, že systém posuvnej kamery používa princípy stereovízie pre ktorú platí, že výpočet polohy význačných bodov scény (bodov, ktoré budú použité pre výpočet polohy) je tým presnejší, čím bližšie sa nachádza určená korešpondencia ku snímaciemu systému.

Historicky bol začiatok výskumu vizuálnej odometrie motivovaný najmä NASA, pre robustné určenie pohybu mobilných robotov vo výskume iných kozmických telies – roverov. V roku 2004 bola vizuálna odometria prvýkrát úspešne použitá na roveri na povrchu Marsu.

8.1 Definícia problému vizuálnej odometrie, klasifikácia

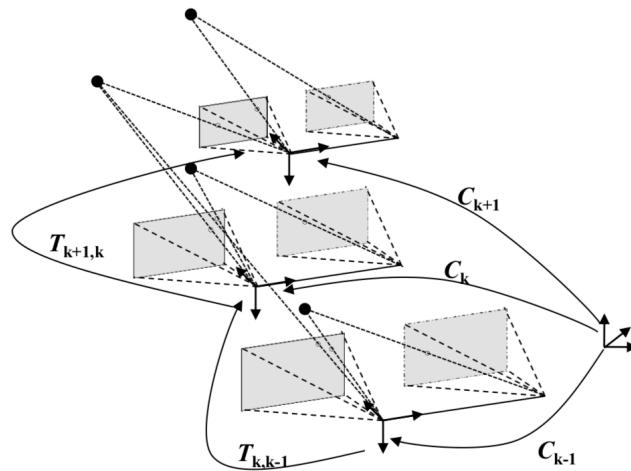
Problém vizuálnej odometrie môžeme formálne definovať nasledovne:

Nech pohybujúci sa agent vybavený kamerou, hĺbkou kamerou alebo sústavou kamier získava obraz z týchto kamier v diskrétnych časových intervaloch k . Vstupný vektor snímok budeme označovať ako $I_{0:n} = \{I_0, I_1, \dots, I_n\}$. Dve susedné pozície kamery $k, k - 1$ sú voči sebe orientované transformáciou $T_{k,k-1} \in \mathbb{R}^{4 \times 4}$ v nasledovnej podobe:

$$T_{k,k-1} = \begin{bmatrix} R_{k,k-1} & t_{k,k-1} \\ 0 & 1 \end{bmatrix}$$

kde $R_{k,k-1} \in SO^3$ je matica rotácie a $t_{k,k-1}$ je translačný vektor. Množina $T_{k,k-1} = \{T_{1,0}, \dots, T_{n,n-1}\}$ potom obsahuje všetky transformácie kamery v časovom slede. Potom, množina pozícii kamery $C_{0:n} = \{C_0, \dots, C_n\}$ obsahuje všetky pozície kamery voči súradnicovému priestoru v prvom snímku $k = 0$. Aktuálna poloha kamery môže byť určená vynásobením všetkých po sebe idúcich matíc $T_k (k = 1 \dots n)$. Súradnicový systém prvého snímku sa často na začiatku merania inicializuje jednotkovou maticou. Úlohou vizuálnej

odometrie teda je určiť relatívne transformácie T_k z obrazov $I_{0:n} = \{I_0, I_1, \dots, I_n\}$ a získať úplný pohyb kamery $C_{0:n}$.



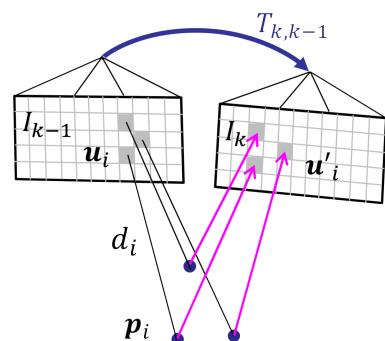
Obrázok 9 - Ilustrácia problému vizuálnej odometrie. Relatívne transformácie dvoch kamerových pozícii $T_{k,k-1}$, sú určené z príznakov na jednotlivých obrazoch a spojené do pozícií kamery C_k vzhľadom na úvodný súradnicový priestor $k = 0$.

Existujú dva hlavné spôsoby výpočtu transformácie medzi dvoma snímkami používané vo vizuálnej odometrii. **Priame algoritmy** používajú všetky nasnímané dátá, t.j.: všetky pixely oboch obrazov a **algoritmy postavené na príznakoch (features, feature-based algoritmy)**, ktoré najprv z obrazu extrahujú významné oblasti – príznaky a tie potom používajú pre určenie pohybu kamery. Priame algoritmy minimalizujú fotometrickú (photometric) chybu:

$$T_{k,k-1} = \arg \min_T \sum_i \|I_k(u'_i - I_{k-1}(u_i))\|_\sigma^2$$

kde:

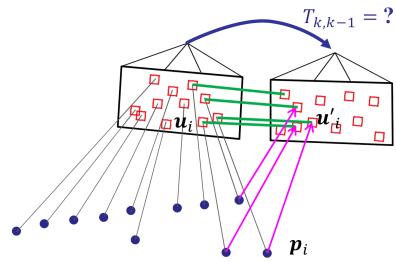
$$u'_i = \pi(T \cdot (\pi^{-1}(u_i) \cdot d))$$



Obrázok 10 - schéma priamej vizuálnej odometrie

Algoritmy využívajúce príznaky naopak minimalizujú chybu reprojekcie:

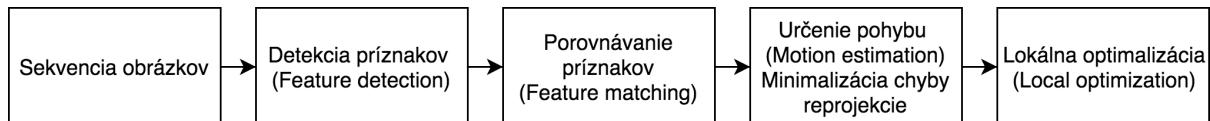
$$T_{k,k-1} = \arg \min_T \sum_i \|u'_i - \pi(p_i)\|_{\Sigma}^2$$



Obrázok 11 - schéma vizuálnej odometrie postavenej na príznakoch

	Priame metódy	Algoritmy využívajúce príznaky
Princíp	<ul style="list-style-type: none"> Minimalizujú fotometrickú chybu (rozdiel intenzít) 	<ul style="list-style-type: none"> Detekcia, extrakcia a porovnávanie príznakov. Minimalizujú chybu reprojekcie (euklidovské vzdialenosť)
Výhody	<ul style="list-style-type: none"> Zvýšenie snímkovacej frekvencie kamery znižuje výpočtovú zložitosť medzi snímkami (menší posun kamery medzi snímkami je možné priamou metódou určiť rýchlejšie) Efektívne využívajú všetky informácie z obrazu 	<ul style="list-style-type: none"> Dokážu spracovať aj veľké posuny kamery Presnosť: Vďaka menšiemu počtu vzoriek dokážu efektívne optimalizovať aj 3D mapu, aj pohyb kamery (bundle adjustment)
Nevýhody	<ul style="list-style-type: none"> Obmedzené pre malý maximálny posun kamery medzi snímkami Súčasná optimalizácia trajektórie kamery aj 3D modelu je výpočtovo zložitá 	<ul style="list-style-type: none"> Pomalé kvôli zdĺhavému procesu detekcie a porovnávania príznakov Z dôvodu výskytu nesprávnych korešpondencií medzi príznakmi je nutné filtrovať tzv. outliers

Tabuľka 1 - Porovnanie priamych a nepriamych metód určenia polohy kamery z obrazu



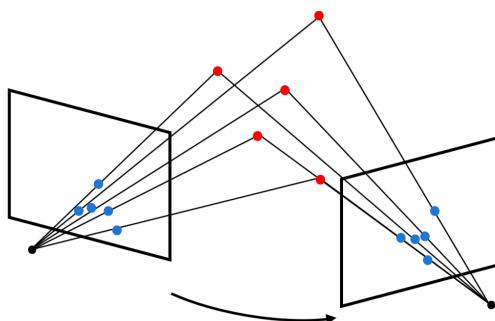
Obrázok 12 - vývojový diagram vizuálnej odometrie využívajúcej príznaky



Obrázok 13 - Vývojový diagram priamej vizuálnej odometrie

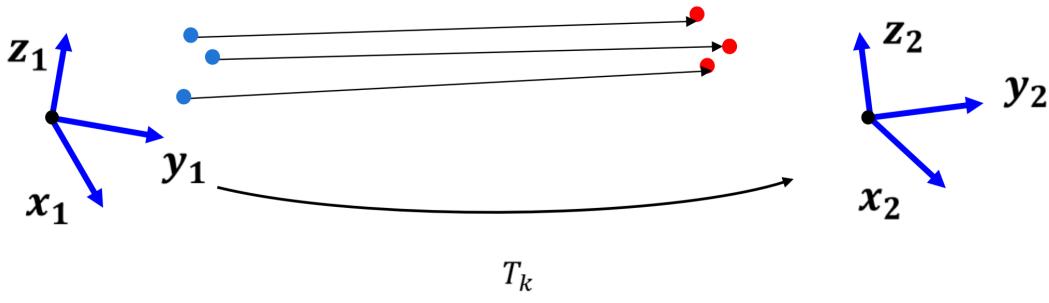
Algoritmy vizuálnej odometrie môžeme rozdeliť aj podľa toho, akým spôsobom určujú transformáciu medzi jednotlivými snímkami a to na:

- **2D ku 2D algoritmy** – pri týchto algoritmoch sa najprv určia príznaky v oboch snímkach, potom sa príznaky porovnajú a určia sa korešpondencie. Riešenie je vypočítané na základe minimalizácie chyby reprojekcie. Štruktúra scény aj pohyb kamery sú určované súčasne. Minimálny počet korešpondencií pre určenie transformácie pohybu kamery je 5. Populárne riešenia používajú 5 až 8 korešpondencií.



Obrázok 14 - 2D ku 2D výpočet posunu kamery na základe 2D príznakov

- **3D ku 2D** – 3D poloha príznakov z predchádzajúceho snímku je známa, spolu s 3D štruktúrou scény, aktuálne spracovávaný snímok má príznaky určené v 2D.
- **3D ku 3D** – každý pár korešpondujúcich príznakov má polohu určenú v 3D. Výpočet prebieha pomocou minimalizácie euklidovskej vzdialenosť medzi korešpondenciami. Minimálny počet korešpondencií pre výpočet riešenia je 3. [29] [30]



Obrázok 15 - 3D ku 3D výpočet posunu kamery na základe 3D príznakov

Algoritmy vizuálnej odometrie je možné rozdeliť ešte podľa spôsobu výberu referenčného modelu, voči ktorému bude určovaná transformácia aktuálneho snímku na:

- **Prístupy porovnávajúce snímku so snímkou (frame-to-frame matching)**, ktoré vždy porovnávajú aktuálny snímok s inou snímkou nasnímaným v minulosti. Pre zvýšenie robustnosti sa často pre porovnanie ako referencia nepoužíva iba predchádzajúci snímok, ale sekvencia snímok o určitej veľkosti, ktoré sú porovávané po jednom. Príkladom tohto riešenia je metóda vizuálnej odometrie RGBD-6D-SLAM [30], kde je aktuálne spracovávaný snímok porovnávaný so sekveniou 15 predchádzajúcich snímok.
- **Prístupy porovnávajúce snímok s modelom (frame-to-model matching)**. Tieto prístupy využívajú priamo mapový model alebo jeho lokálnu časť ako referenčný model pre určenie transformácie aktuálneho snímku. Príkladom je algoritmus Kinect Fusion [31], kde je pomocou techniky raycasting [32] vytvorený model povrchu lokálnej časti mapy a ten je následne porovnaný s aktuálnou snímkou. Výhoda tohto riešenia spočíva najmä v tom, že model mapy predstavuje integrovanú reprezentáciu predchádzajúcich meraní, čo výrazne redukuje šum.

8.2 Príznaky

Príznak (Feature) je časťou obrazu, ktorá sa významne odlišuje od svojho okolia. Zvyčajne súvisí so zmenou určitej vlastnosti alebo viacerých vlastností obrazu vo vybranom mieste obrazu, aj keď nemusí byť umiestnený presne v mieste takejto zmeny, t.j.: zmena sa môže nachádzať v okolí miesta príznaku. Vlastnosti obrazu, ktoré najčastejšie zvažujeme pri vyhľadávaní takýchto zmien sú jas, farba a textúra obrazu. Príznaky sú podmnožinami obrazu, v ktorom sme ich detegovali a môžu mať rôzne podoby – môžu to byť jednotlivé body ale aj rohy, hrany alebo rôzne regióny v obraze, ktoré charakterizuje určitá spoločná vlastnosť.

Príznak je teda špecifickým regiónom v obraze a pri jeho určovaní zvažujeme aj jeho priestorové okolie v rámci obrazu. Priestorové okolie príznaku na obraze však nemusí korešpondať s vizuálnou interpretáciou obrazu. Okolia príznakov sa môžu navzájom prekrývať, niektoré nezaujímavé časti obrazu naopak nemusia patriť do žiadneho okolia príznaku. V ideálnom prípade by príznaky korešpondovali so sémanticky významnými časťami obrazu – objektami alebo ich časťami. Tento stav je však v praxi zatial prakticky neuskutočiteľný, keďže vyžaduje vysokú úroveň rozpoznávania a interpretácie snímanej scény, čo je výpočtovo veľmi náročné. Aktuálne riešenia preto často vyberajú príznaky priamo na základe intenzít vybraných vlastností obrazu.

8.2.1 Vlastnosti príznakov

Opakovateľnosť: Z dvoch obrazov tej istej scény získaných za iných podmienok snímania (napríklad z inej pozície kamery) je potrebné detegovať vysoké percento rovnakých príznakov v oboch obrazoch.

Odlíšiteľnosť/informatívnosť: Príznaky by mali byť vyberané z veľkého množstva možných variácií, takže by malo byť možné ich jednoznačne identifikovať, navzájom odlišovať a porovnavať.

Početnosť: Počet detegovaných príznakov v obraze by mal byť dostatočne vysoký tak, aby aj na malých objektoch v scéne bol detegovaný dostatočný počet príznakov pre ich identifikáciu. Optimálny počet príznakov však závisí na konkrétnej aplikácii. V ideálnom prípade by mal byť počet príznakov variabilný v širokom rozsahu. Hustota príznakov v obraze by mala odrážať informačnú bohatosť obrazu a príznaky by mali teda vytvárať kompaktnú, unikátnu reprezentáciu obrazu.

Presnosť: Príznaky by malo byť možné dostatočne presne lokalizovať v priestore nezávisle od ich orientácie alebo mierky.

Efektívnosť: Rýchlosť detekcie príznakov by mala dovoľovať použitie príznakov pre časovo kritické aplikácie.

Najdôležitejšou vlastnosťou príznakov z hľadiska vizuálnej odometrie je opakovateľnosť. Vysokú opakovateľnosť je možné dosiahnuť pomocou dvoch vlastností:

Invariantnosť: Ak očakávame veľké deformácie, preferovaným postupom je určenie matematických transformácií pre tieto deformácie a vytvorenie detektorov príznakov, ktoré sú na takýchto transformáciách nezávislé. Príkladom takejto situácie môže byť detekcia príznakov v dvoch obrazoch, ktoré snímajú tú istú scénu, ale každý z nich bol zosnímaný z inej pozície kamery (napríklad došlo ku posunu kamery).

Robustnosť: V prípade malých deformácií často postačuje vytvorenie takého detektora príznakov, ktorý je menej citlivý – to znamená, že ak sa v obrazu nachádzajú deformácie, presnosť detekcie bude klesať, ale nie významne. Deformácie, ktoré sa pokúšame kompenzovať týmto spôsobom sú napríklad šum, artefakty vzniknuté kompresiou obrazu, drobné skreslenia snímacieho systému alebo fotometrické odchýlky.

Viaceré z týchto vlastností nie je možné zlepšovať nezávisle od iných. Opakovateľnosť je napríklad veľmi závislá od invariantnosti, robustnosti a početnosti. Naopak vyššia odlíšiteľnosť zväčša znamená nižšiu robustnosť a naopak. Pre nami vybranú aplikáciu je preto veľmi dôležité zvoliť správny mechanizmus detekcie príznakov [33].

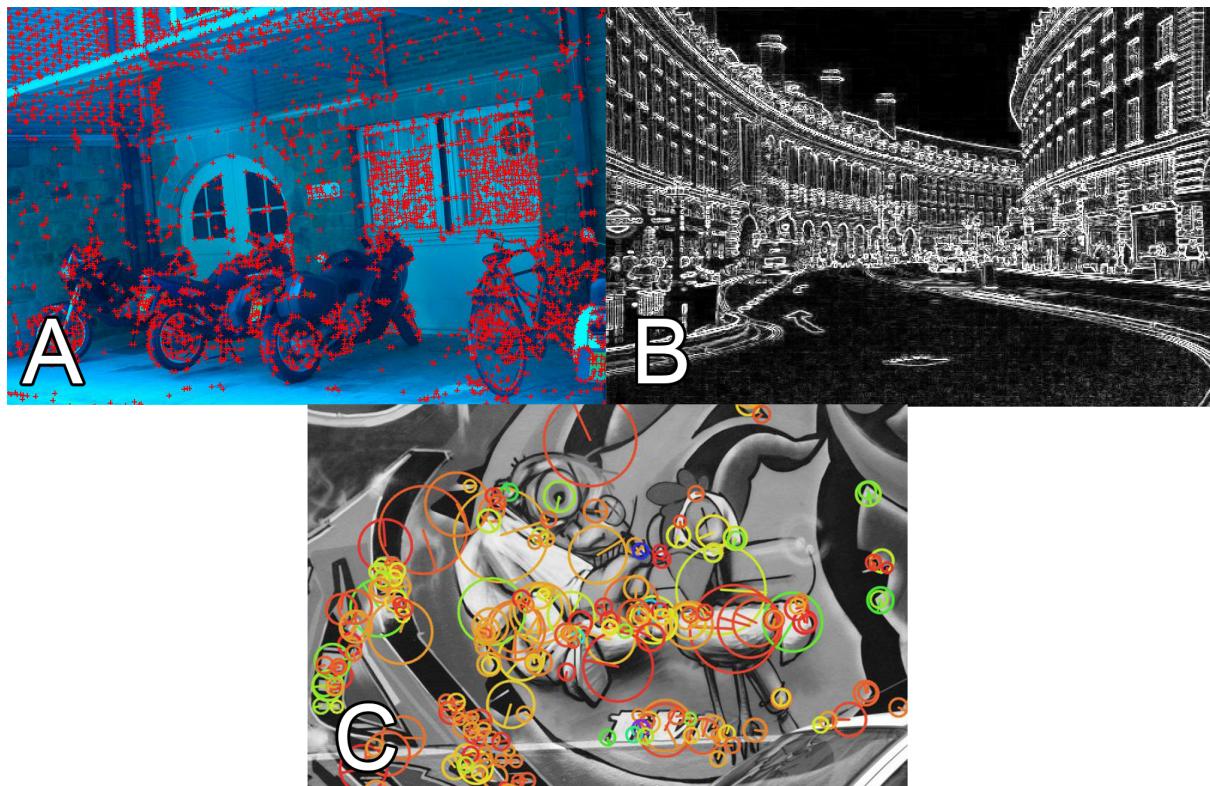
8.2.2 Typy príznakov v obraze

Ako sme uviedli vyššie, príznaky môžu korešpondovať s viacerými typmi podmnožín bodov obrazu, v praxi sa najviac využívajú najmä:

Hrany (Edges): Hrany sú miesta obrazu, ktoré tvoria hranice medzi dvoma regiónmi v obrazu. Môžu mať rôzne tvary a môžu vytvárať prepojenia. V praxi sa zvyčajne definujú ako množiny bodov obrazu, ktoré majú výraznú odchýlku gradientu obrazu. Okrem hrany algoritmy zvažujú aj iné vlastnosti hrany ako tvar, vyhľadenosť (smoothness) a hodnotu gradientu. Hrany majú jednorozmernú štruktúru. Pre pretiahnuté (t.j.: objekty ktorých šírka je násobne vyššia ako výška alebo naopak) objekty používame spoločné označenie hrebene. Ich detekcia býva ľahšia ako u iných typov príznakov, avšak sú veľmi výhodné v aplikáciach ako vyhľadávanie ciest v leteckých snímkach a vyhľadávanie ciev v medicínskych snímkach.

Rohy/body záujmu (corners/interest points): Tieto pojmy označujú bodové príznaky obrazu, ktoré majú lokálne dvojrozmernú štruktúru. Názov "rohy" priniesli prvé algoritmy pre detekciu takýchto príznakov, keďže tieto analyzovali hrany tak, aby našli náhle zmeny v ich smere - rohy. Tieto body nemusia zodpovedať vizuálne identifikovateľným rohom v obrazu, označenie roh sa používa z historických dôvodov.

Škvrny, regióny záujmu (Blobs, regions of interest): Škvrny popisujú regióny na obraze v kontraste s hranami, ktoré reprezentujú hranice medzi nimi. Regióny bývajú zvyčajne určené jedným bodom - stredom, alebo maximom operátora, ktorý použijeme pre vyhodnotenie obrazu pre získanie týchto regiónov (tzv. blob operátor). Výhodou tohto typu príznakov je to, že dokážu lepšie detegovať oblasti obrazu, ktoré sú príliš vyhľadené na to, aby boli identifikované ako rohy.



Obrázok 16 - Ukážka rozdielnych typov príznakov v 2D obraze. Obrázok A zobrazuje rohové príznaky, obrázok B výstup hranového detektora a obrázok C príznaky detektora SIFT

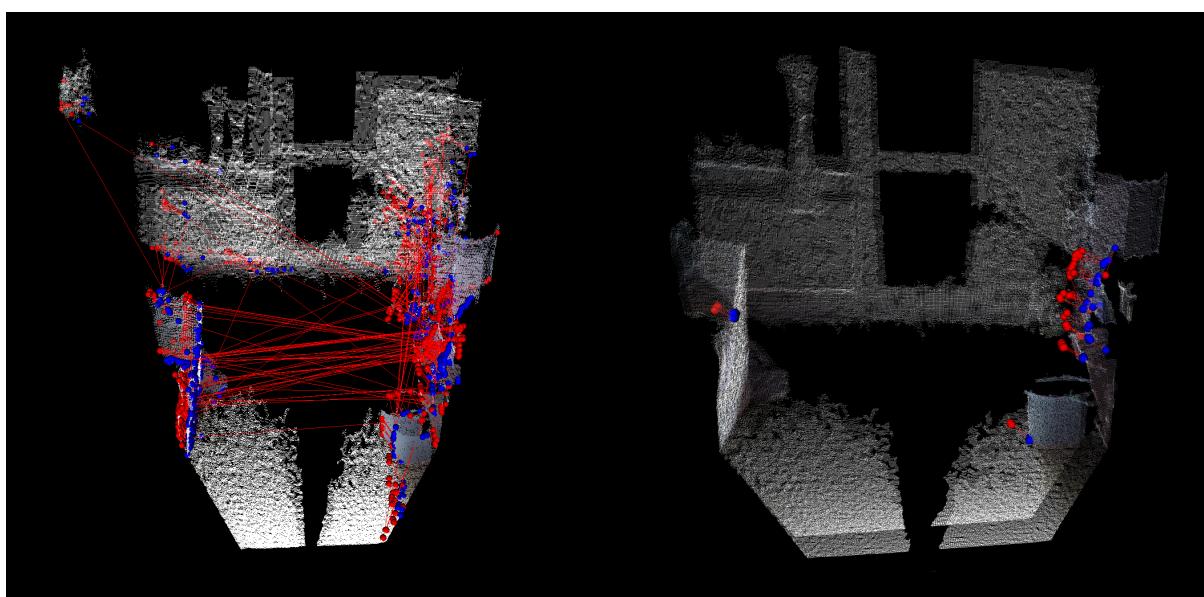
8.2.3 Príznaky vo vizuálnej odometrii

Vo vizuálnej odometrii sa príznaky využívajú pomocou dvoch hlavných prístupov. Prvý prístup funguje nasledovne:

1. V oboch snímkach, medzi ktorými je potrebné určiť transformáciu pohybu kamery sa detegujú príznaky pomocou **detektora príznakov (feature detector)**.
2. Príznaky sa opíšu **deskriptorom príznakov (feature descriptor)**.
3. Vypočítané deskriptory príznakov z oboch obrazov sa navzájom porovnávajú a určujú sa medzi nimi zhody pomocou **porovnávača príznakov (feature mather)**.

V terminológii vizuálnej odometrie sa týmto spôsobom rieši **problém korešpondencie**, teda asociácie dát oboch snímok nutnej pre výpočet transformácie pohybu kamery.

Detektor, deskriptor aj porovnávač príznakov spracovávajú pri vizuálnej odometrií merania zatažené šumom a preto poskytujú iba určitú mieru správnosti výstupu. To znamená, že medzi asociáciami, ktoré boli určené z dvoch po sebou nasledujúcich snímok, budú aj asociácie nesprávne a v niektorých prípadoch budú dokonca nesprávne asociácie prevažovať. Preto ich nebude možné priamo použiť pre výpočet transformácie pohybu kamery medzi týmito snímkami. Metódou, ktorá tento problém rieši je napríklad metóda RANSAC, predstavená v ďalších častiach kapitoly vizuálnej odometrie.



Obrázok 17 - Vľavo korešpondencie určené pomocou detektora FAST, opísané deskriptorom ORB a porovnané pomocou BF porovnávača. Červené body predstavujú príznaky z prvej snímky, modré body príznaky z druhej snímky. Červené čiary medzi nimi graficky znázorňujú korešpondenciu. V 3D priestore vytvorenom pomocou hĺbkových snímok je možné dedukovať, že výpočet korektnej transformácie pohybu kamery za použitia aj chybných korešpondencií nebude možný. Vpravo sú zobrazené korešpondencie, ktoré boli použité pre výpočet korektnej transformácie kamery. V tomto prípade je ich 5% z celkového počtu.

Popísaný prístup je výhodný pri malých posunoch kamery a malých scénach. Jeho nevýhodou je, že postupne akumuluje **drift**, teda polohovú a orientačnú odchýlku určenia pozície v čase.

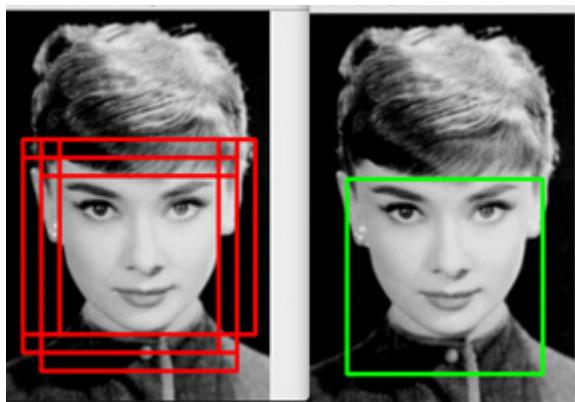
Alternatívny postup nezávisle deteguje príznaky vo všetkých dostupných snímkach a porovnáva ich na základe určitej metriky medzi výslednou množinou deskriptorov. Tento prístup je viac populárny v ostatných rokoch, v ktorých sa čo raz viac presadzuje požiadavka

pre mapovanie väčších prostredí. Posuny kamery v tomto prístupe sú najväčšie možné, čo výrazne redukuje drift.

8.2.4 Detektory príznakov

V histórii počítačového spracovania obrazu bolo predstavených množstvo detektorov príznakov, ktoré využívajú príznaky rôznych typov. Rohové detektory ako **Harris** [34] a **FAST corner detector** [35] sú rýchle a menej rozlíšiteľné, naopak detektory škvŕn ako **SIFT** [36] a **SURF** [37] sú pomalšie a rozlíšiteľné vo väčšej miere. Rohy sú v obraze detegované presnejšie, ale naopak škvŕny sú viac odolné na zmenu mierky (**scale**). To znamená, že rohy nebudú tak dobre párovateľné ako škvŕny pri veľkých posunoch kamery alebo zmenách mierky. Výber detektora teda závisí najmä od, dostupného výpočtového výkonu, typu prostredia a maximálnej vzdialenosťi posunu medzi snímkami

Detekcia príznakov pomocou každého detektora sa skladá z dvoch krokov. V prvom kroku sa na obraz aplikuje **operácia príznakovej odozvy (feature response function)**. Úlohou



Obrázok 18 - Vľavo úspešné detekcie tváre na obraze, vpravo vybrané maximum po potlačení ostatných detekcií. [127]

tohto kroku je transformovať obraz do priestoru, v ktorom je možné detegovať lokálne minimá alebo maximá tohto priestoru. Ďalej prebehne **potlačenie nemaxím (non-maxima suppression)**. Podstatou tejto operácie je identifikovať na obraze viacnásobné detekcie tej istej oblasti v obraze a vybrať z nich tú, ktorá predstavuje lokálne maximum.

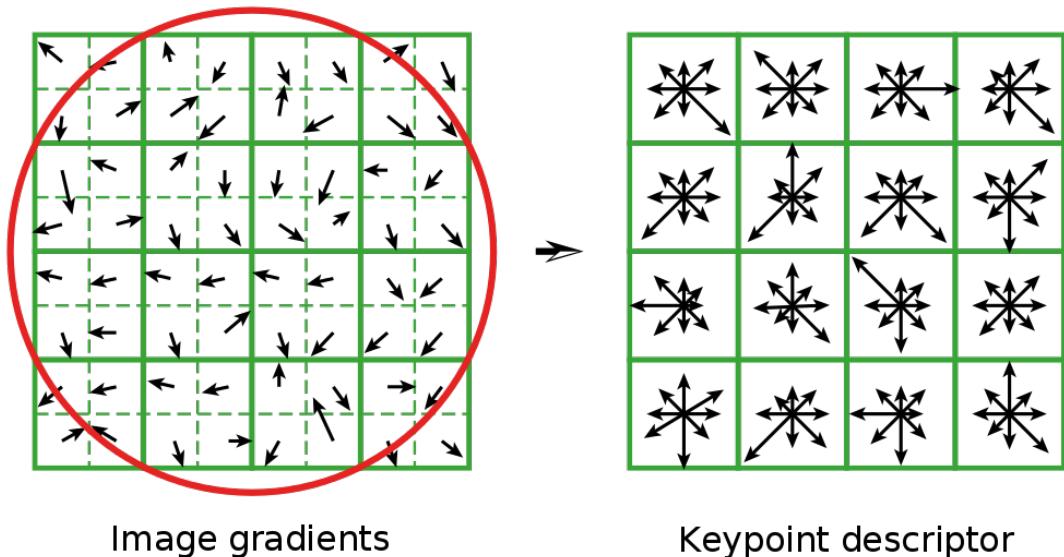
Aby bol detektor nezávislý od mierky obrazu, detekcia prebehne na aj na kópiach obrazu s menšou a väčšou mierkou.

Okrem detektorov príznakov pre 2D obraz existuje aj veľké množstvo detektorov pracujúcich priamo nad 3D dátami prichádzajúcimi z hĺbkovej kamery a to či už na hĺbkovom obraze (napr. NARF detektor [38]), alebo priamo nad mračnom bodov. Vyčerpávajúce prehľady a porovnania jednotlivých metód detekcie príznakov pre 3D dátá nájdeme v [39] [40].

8.2.5 Deskriptory príznakov

Pri výpočte deskriptorov sa z obrazu vyextrahuje okolie každého príznaku a určí sa preň kompaktný opis, ktorý je možné porovnávať s inými deskriptormi pomocou porovnávača.

Najjednoduchšie deskriptory zvažujú iba vzhľad príznaku, teda intenzitu pixelov v jeho okolí. Tieto deskriptory však nie sú invariantné voči rotácii alebo zmene mierky. Tento problém prekonal napríklad deskriptor SIFT, ktorý používa histogram lokálnych gradientov obrazu.



Obrázok 19 - SIFT deskriptor. Okolie príznaku je rozdelené na 4 regióny. Pre každý kvadrant je určený histogram ôsmich orientácií gradientu. Tieto histogramy sú spojené za seba a tak vytvoria 128 elementový vektor deskriptora. Pre redukciu vplyvu osvetlenia je descriptor normalizovaný [36].

Deskriptor SIFT [36] sa ukázal ako stabilný voči zmenám osvetlenia, rotácie aj mierky. Tento deskriptor umožňuje správnu asociáciu aj pri veľkých pohyboch kamery a preto sa s ním už tradične stretávame v oblasti vizuálnej odometrie. Jednou z nevýhod deskriptora SIFT je jeho vysoká výpočtová zložitosť a teda pomalosť. Tento problém čiastočne prekonal detektor a deskriptor SURF [41], ktorý je v podstate vysoko optimalizovanou verziou detektora a deskriptora SIFT. V nedávnej minulosti boli však predstavené deskriptory, ktoré používajú jednoduché binárne reťazce a je teda rádovo jednoduchšie ich vypočítátať. Prvým z týchto deskriptorov bol BRIEF [42] [43], ktorý ale neboli invariantný voči rotácii a zmene mierky a na ktorého základe bol vytvorený jeden z najúspešnejších detektorov a deskriptorov súčasnej doby, ORB [44]. V rovnakom čase detektor bol predstavený aj detektor BRISK [45], ktorý

podobným spôsobom rozširuje možnosti rohového detektora FAST [35]. V literatúre sa stretнемe aj s označením deskriptora ako **extraktor**.

8.2.6 Párovanie príznakov

Párovanie alebo porovnávanie príznakov porovnáva jednotlivé deskriptory za účelom nájdenia korešpondencií. Vo vizuálnej odometrii sa tým rieši problém vzájomnej asociácie dát z jednotlivých snímkov. Pre určenie korešpondencií je možné jednoducho porovnať všetky vzorky navzájom, vypočítať pre ne mieru odlišnosti a pre každú vzorku vybrať práve tú vzorku, ktorá je najmenej odlišná. Tento prístup nazývame **porovnávanie hrubou silou (brute force matching)**. Túto mieru odlišnosti medzi príznakmi nazývame aj **vzdialenosťou príznakov (feature distance)**.

Riešenie hrubou silou je výpočtovo veľmi náročné pre väčšie množstvá vzoriek. Preto existujú implementácie riešenia problému **hľadania najbližšieho suseda (nearest neighbor search)**, ktoré dokážu aj vo veľkých datasetoch vyhľadávať veľmi efektívne, najmä vďaka ich implementácii pomocou **kd-stromov (kd-tree)**. Knižnica OpenCV napríklad pre túto úlohu poskytuje modul **FLANN, Fast Library for Approximate Nearest Neighbors**. Pre výpočet vzdialenosťi porovnávače používajú okrem kd-stromov najmä **Hammingove vzdialosti** a **hash tabuľky**. Ďalšou možnosťou, ako urýchliť alebo spresniť párovanie je predikovať, kde sa majú korešpondencie nachádzať v ďalšom snímku. Pre túto predikciu vieme použiť napríklad IMU jednotku, kolesovú odometriu alebo môžeme modelovať pohyb robota.

Vo väčšine implementácií sa stretнемe s filtráciou nájdených párov vzoriek na základe ich vzájomných vzdialostí a to napríklad [36]:

- Vylúčením zhôd so vzdialenosťou väčšou ako preddefinovaná prahová hodnota,
- vylúčením vzoriek so vzdialenosťou väčšou ako medián alebo aritmetický priemer vzdialostí,
- vylúčením vzoriek na základe maximálnej a minimálnej vzdialosti vzoriek.

Ako bolo spomenuté v úvode kapitoly, výstup z porovnávača príznakov spravidla stále obsahuje nesprávne identifikované páry príznakov pre vizuálnu odometriu a je potrebné ich pre určenie transformácie pohybu kamery filtrovať. Jednou z metód, ktoré toto robia je RANSAC.

8.3 RANSAC

Random Sample Consensus algoritmus (zhoda náhodných vzoriek) je iteratívna metóda určená pre určenie parametrov vybraného modelu z množiny vzoriek meraní, ktoré sú zaťažené šumom. Metóda bola publikovaná už v roku 1981 pre riešenie kartografického problému určenia polohy – location determination problem (LDP), ktorý je veľmi blízky problému robotického mapovania. [46]

Metódu je možné aplikovať na široké spektrum problémov počítačového videnia a robotického mapovania, kde je potrebné snímanú scénu interpretovať v kontexte definovaných modelov zo zašumených dát prichádzajúcich zo snímačov.

Interpretácia scény rieši súčasne 2 problémy: a.) problém klasifikácie, teda problém určenia, ktoré vzorky dát patria do jedného z určených modelov, b.) problém výpočtu najlepších parametrov vybraného modelu na základe dostupných vzoriek. V praxi sú oba tieto problémy prepojené – pre úspešnú klasifikáciu musíme vybrať vhodné parametre.

Riešenia predchádzajúce metóde RANSAC, ako napríklad metóda najmenších štvorcov, sa snažili odhadnúť parametre modelu na základe všetkých vzoriek dát. Tieto algoritmy neobsahovali mechanizmus pre detekciu a vyradenie chýb - vzoriek, ktoré boli významne odlišné od zvyšku datasetu. Využívali predpoklad, že maximálna odchýlka jednotlivej vzorky od zvoleného modelu je priamou funkciou veľkosti datasetu, takže nezávisle od veľkosti datasetu bude vždy k dispozícii dostatok dobrých vzoriek na to, aby v priemere vyhľadili všetky hrubé chyby. V praktických aplikáciách však tento predpoklad často zlyháva – dáta obsahujú veľké percentuálne množstvo výrazne chybných meraní a parametre výsledného riešenia tieto chyby zohľadňujú. Prvé implementácie mechanizmov pre identifikáciu a vylúčenie týchto chybných meraní využívali heuristické postupy. Určili parametre modelu pre celý dataset, následne vyhodnotili, ktoré vzorky sa od modelu určeného týmito parametrami významne odlišujú, teda hodnota ich chyby voči vybranému modelu prekročila určitý definovaný prah, vylúčili ich a znova určili parametre celého datasetu. Proces bol ukončený vtedy, keď dataset neobsahoval vzorky, ktorých chyba voči vybranému modelu prekračovala prahovú hodnotu. Autori metódy RANSAC ukázali, že tento postup je pre identifikáciu chybných vzoriek často nevhodný a navýše je výpočtovo náročný, keďže využíva viacero iterácií cez celý dataset [46].

Metóda RANSAC je presným opakom priemerovacích techník. Namiesto vyhodnocovania všetkých dostupných vzoriek pre identifikáciu parametrov a späťej identifikácie vzoriek, ktoré spôsobujú jeho chybu, RANSAC využíva malú vzorku dát pre určenie parametrov modelu a túto vzorku následne rozširuje o vzorky konzistentné s týmito parametrami. Model, ktorý obsahuje najviac konzistentných vzoriek, je vybraný ako víťaz a finálne parametre modelu sú určené z týchto konzistentných vzoriek. Pre vzorky, ktoré patria do modelu používa metóda výraz **inlier** a pre chybné vzorky výraz **outlier**.

Všeobecná schéma RANSAC algoritmu je nasledovná:

Nech p je množina všetkých vzoriek. Nech n je minimálny počet vzoriek potrebných pre určenie parametrov modelu tak, že n je menšie ako celkový počet vzoriek $n \leq \#(p)$. Nech t predstavuje práhovú hodnotu minimálneho počtu identifikovaných správnych vzoriek a k predstavuje maximálny počet iterácií, ktoré môže algoritmus vykonať. Potom môžeme popísť všeobecnú schému algoritmu RANSAC ako:

1. Náhodne vyber n vzoriek z p . Urči parametre modelu pre tento model.
2. Urči, koľko iných vzoriek modelu vyhovuje týmto parametrom. Označ ich ako *inliers*.
3. Ak je počet *inliers* menší ako t , pokračuj opakováním náhodného výberu v bode 1.
4. Ak je počet *inliers* väčší ako t , vypočítaj parametre výsledného modelu pre tieto *inliers* a ukonči algoritmus.
5. Ak v žiadnej iterácii neboli identifikované dostatočné počet *inliers* pre úspešné ukončenie algoritmu a počet iterácií prekročil k , ukonči algoritmus a ako výstup použí parametre modelu, ktoré boli vypočítané v iterácii s najvyšším dostupným množstvom *inliers*.

[46]

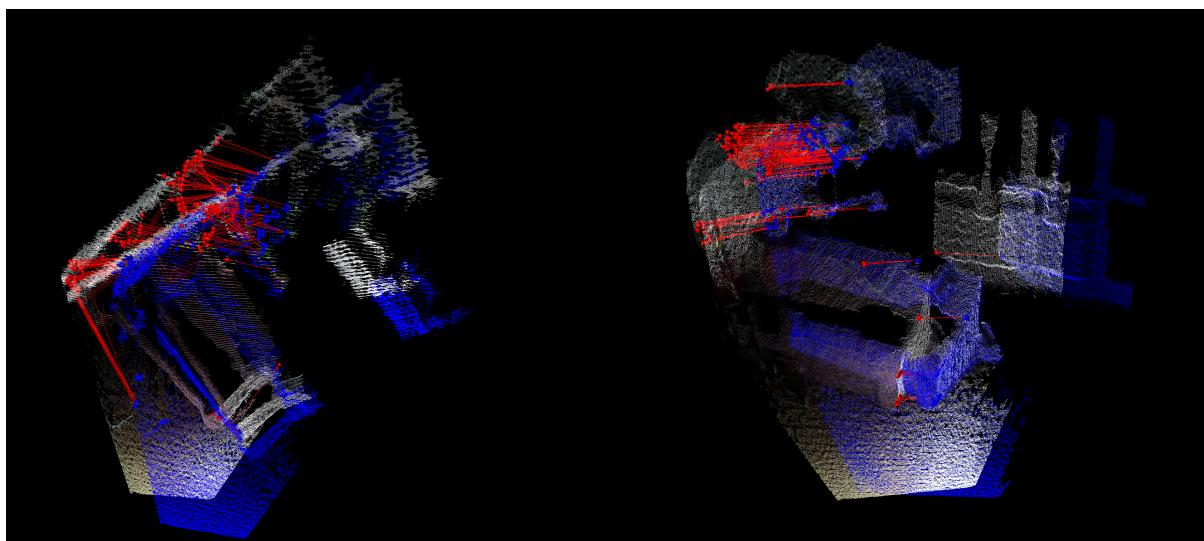
Algoritmus RANSAC je možné ukončiť aj po prejdení preddefinovaného počtu iterácií. V tomto prípade bude vybrané riešenie riešením s najväčším počtom *inliers* zo všetkých riešení. Minimálny počet iterácií algoritmu RANSAC, ktorý garantuje, že vypočítané riešenie je správne je možné vypočítať ako:

$$N = \frac{\log(1 - p)}{\log(1 - (1 - \epsilon)^s)}$$

Kde s je počet bodov z ktorých je možné vypočítať riešenie, ϵ je precentuálny podiel inliers v prehľadávaných dátach a p je požadovaná pravdepodobnosť správnosti riešenia. Z dôvodu robustnosti sa často N násobí desiatimi. Pokročilejšie implementácie metódy RANSAC určujú podiel inliers iteratívne počas svojho behu.

Metóda RANSAC je náhodná iteratívna metóda. To znamená, že jej výstup nie je deterministický a vypočítané riešenie je rozdielne pri samostatných spracovaniach jedného datasetu. Napriek tomu je však metóda stabilná s väčším počtom iterácií.

Vo vizuálnej odometrii je modelom hľadaným metódou RANSAC transformácia medzi dvoma množinami bodov v priestore, ktorá má **minimálnu chybu reprojekcie**. Chyba reprojekcie je geometrická odchýlka medzi polohou príznakov z jedného snímku a ich projekciami do druhého snímku pomocou testovanej transformácie.



Obrázok 20 - Vľavo: Grafické znázornenie korešpondencií určených pomocou detektora ORB. Jedna snímka zo snímača Kinect je zobrazená s textúrou, druhá snímka je z dôvodu viditeľnosti renderovaná s modrou textúrou. Červené a modré body spojené červenou čiarou zobrazujú korešpondencie medzi príznakmi z oboch snímok. Ako vidíme, medzi párimi korešpondencií sa nachádzajú aj nevyhovujúce vzorky. Vpravo: Iba vyhovujúce vzorky – **inliers** identifikované pomocou metódy RANSAC.

Pre určenie transformácie medzi snímkami z kalibrovanej kamery pomocou korešpondujúcich párov bodov bez ďalších informácií je potrebné disponovať aspoň **piatimi** vzorkami, tj. piatimi korešpondujúcimi párimi príznakov z oboch snímok. Riešenia umožňujúce výpočet z menšieho počtu vzoriek vyžadujú dodatočné informácie ako napríklad informácie z IMU alebo iné fakty o snímanej scéne. Vyšší minimálny počet bodov pre výpočet transformácie pohybu kamery

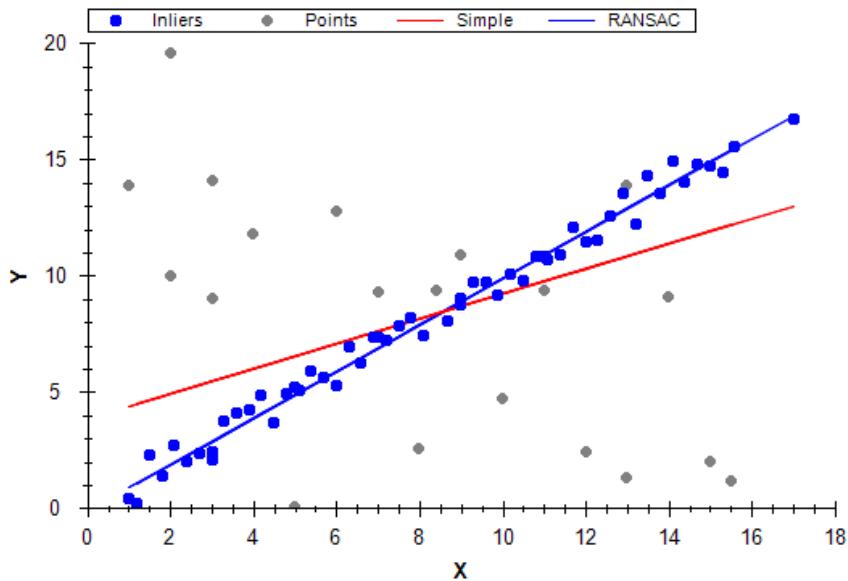
bude znamenať zvýšenie výpočtovej zložitosti a rast počtu iterácií, ale riešenie bude odolnejšie na šum. V praxi sa najčastejšie stretávame s použitím 5,6,7 a 8 bodov. Takto veľké bázy pre výpočet zaručujú ešte dostatočne vysokú rýchlosť pre bežné datasety.

Pôvodná metóda RANSAC bežne vyžaduje pre niektoré datasety aj tisícky iterácií, čo nemusí byť vyhovujúce najmä pre spracovanie v reálnom čase. Z tohto dôvodu boli predstavené viaceré zlepšenia metódy, ktoré efektívne znižujú počet iterácií algoritmu. Riešenia zväčša generujú hypotézy o pohybe kamery na základe dodaných dát, alebo modelujú kinematický model kamery alebo využívajú pravdepodobnostné modelovanie správnosti korešpondencií [28].

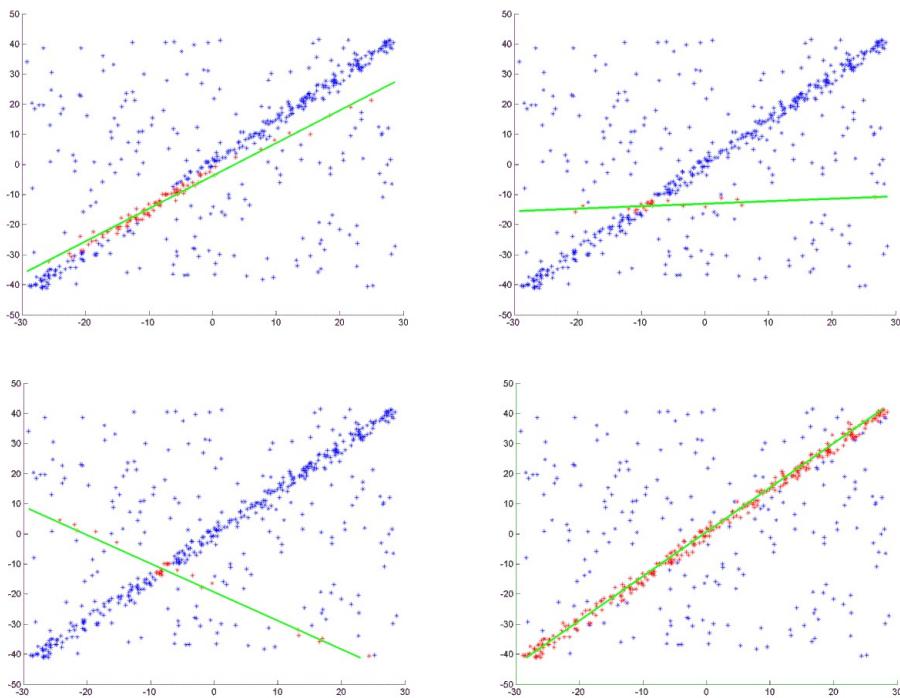
8.3.1 Príklad použitia metódy RANSAC

Pre jednoduché pochopenie implementácie metódy RANSAC čitateľom bude uvedený príklad, v ktorom aplikujeme túto metódu na problém detekcie čiary v 2D obraze. Hľadaným modelom bude 2D priamka. Algoritmus RANSAC bude postupovať nasledovne:

1. *Vyberieme 2 body, keďže práve toľko bodov potrebujeme pre určenie priamky v priestore. Určíme parametre priamky pomocou týchto bodov.*
2. *Overíme, kolko bodov vyhovuje určenej rovnici priamky*
3. *Ak je množstvo správne určených bodov menšie ako preddefinovaná prahová hodnota, parametre modelu zahodíme a pokračujeme ďalšou iteráciou.*
4. *Ak je množstvo správne určených bodov väčšie ako prahová hodnota, vyberieme všetky tieto body a určíme výsledné parametre priamky pomocou všetkých týchto bodov.*



Obrázok 21 - porovnanie výsledkov lineárnej regresie bez a s použitím metódy RANSAC. Pri použití jednoduchej lineárnej regresie (červená priamka) výsledné riešenie zohľadňuje aj chybné merania. Metóda RANSAC úspešne klasifikuje inliers (modré body) a outliers (šedé body). Výsledné riešenie (modrá priamka) vznikne tak isto pomocou použitia lineárnej regresie, ale iba pomocou správnych vzoriek – inliers [47].



Obrázok 22 - 4 iterácie algoritmu RANSAC pre lineárnu regresiu. Modré body predstavujú jednotlivé vzorky datasetu, červené body predstavujú identifikované inliers. Zelená čiara predstavuje priamku identifikovanú pomocou linerárnej regresie v danej iterácii na základe náhodných vzoriek. Iterácie 1,2 a 3 neobsahovali dostatočný počet inliers pre ukončenie algoritmu. Vo štvrtej iterácii bol tento počet prekročený, algoritmus sa ukončil a výslednú priamku určil zo všetkých identifikovaných inliers [48].

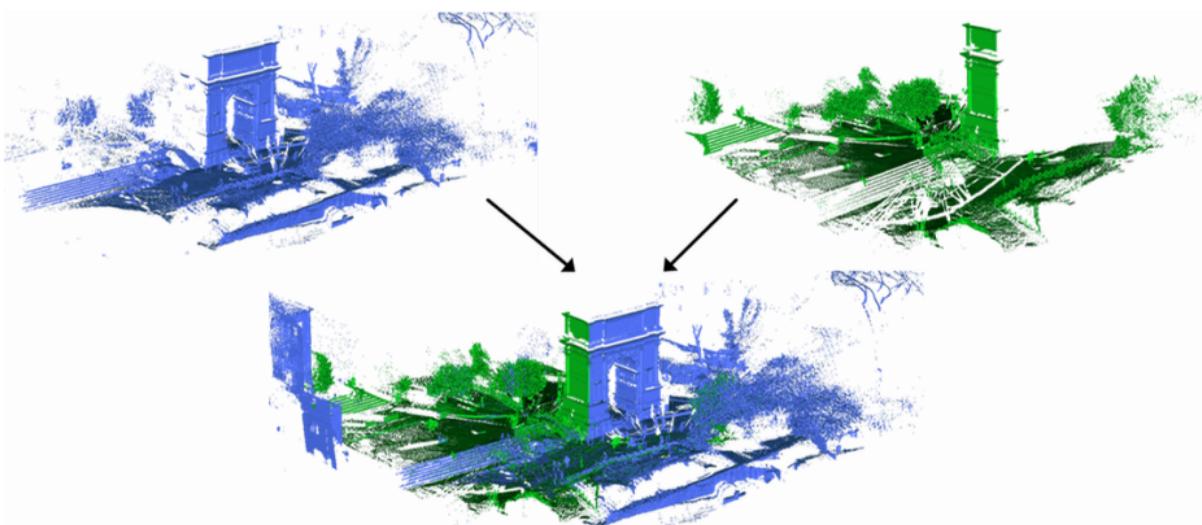
8.4 Registrácia množín bodov v 3D priestore

Typ vizuálnej odometrie 3D ku 3D, ktorý rieši úlohu asociácie dvoch množín bodov v priestore nazývame aj problémom **registrácie mračien bodov (point cloud registration)**. Formálne:

$${}^B_A T = \operatorname{argmin}_T (\operatorname{error}(T(P_A), Q_B))$$

Kde P je 3D množina bodov reprezentujúca aktuálnu snímku (reading) v jeho vlastnom súradnicovom priestore, Q je 3D množina bodov reprezentujúca predchádzajúcu snímku (reference), tiež v jeho vlastnom súradnicovom priestore. Úlohou registrácie mračien bodov je potom určiť transformáciu T medzi súradnicovými systémami A a B pomocou minimalizácie chybovej funkcie error , kde $T(S)$ vyjadruje aplikáciu transformácie na súradnicový systém S . Základnou požiadavkou všetkých algoritmov je, že sa jednotlivé mračná bodov musia čiastočne prekrývať. Metódy registrácie mračien bodov môžeme rozdeliť na:

- **globálne**, kde je poloha mračna Q výrazne odlišná od súradného systému množiny P . Príkladom môže byť registrácia skenu z hĺbkovej kamery do mračna bodov celej skenu alej ulice. V tomto prípade je nutné prehľadať celé P tak, aby sa našlo miesto, na ktorom bude chyba globálne minimálna. Metódami tohto typu sú napríklad 4PCS (four points congruent sets) [49] alebo Point par feature [50].



Obrázok 23 – globálna registrácia mračien bodov. Žiadne predpoklady o vzájomnej polohe súradných systémov nie sú k dispozícii. [51]

- **lokálne**, kde je úvodná transformácia medzi oboma množinami bodov viac-menej správna. Populárnu metódou tohto typu je ICP.

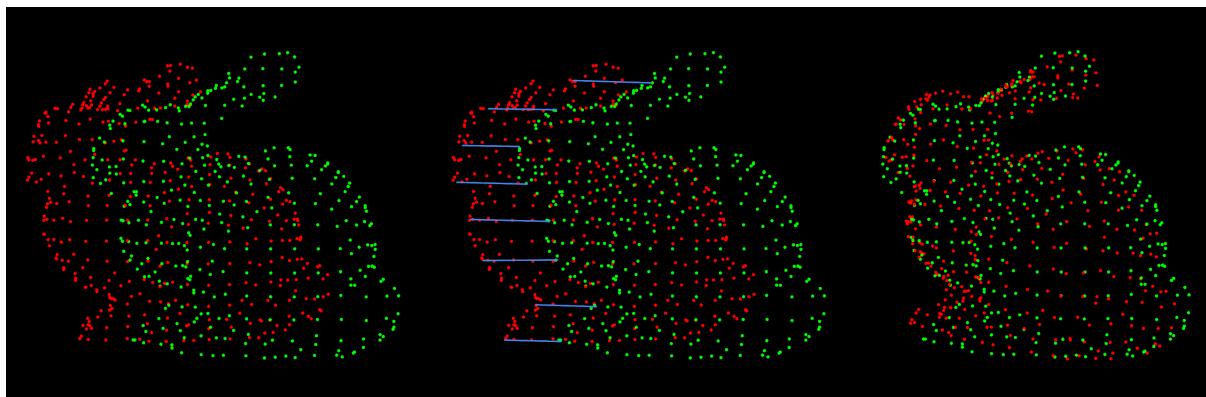
8.5 ICP - Iterative closest point

Iterative closest point je regisračná metóda pre výpočet transformácie medzi dvoma datasetmi v 3D aj 2D priestore. Vstupom metódy sú dva datasety, medzi ktorými je potrebné transformáciu určiť. Prvý dataset budeme označovať ako referenčný, druhý ako modelový. Algoritmus následne určí, ako je potrebné transformovať modelový dataset tak, aby došlo k jeho zarovnaniu s referenčným datasetom. Na rozdiel od väčšiny algoritmov pre registráciu 3D datasetov, ICP nepoužíva pre svoj beh príznaky, ale korešpondencie určuje triviálnym výberom priestorovo najbližších bodov. Myšlienkom ICP je teda, že aj napriek vadným korešpondenciám je minimalizácia chybovej funkcie užitočná. Po aplikovaní minimalizácie sú korešpondencie určené opäťovne a algoritmus pokračuje v ďalšej iterácii.

Všeobecná schéma algoritmu ICP je nasledovná:

1. Pre každý bod z modelového datasetu vyber najbližší bod z referenčného datasetu. Nájdené páry označ ako korešpondencie.
2. Urči najlepšiu kombináciu translácie a rotácie pomocou techniky minimalizácie strednej kvadratickej chyby medzi korešpondenciami. Použi váhovanie pre vylúčenie chybných korešpondencií.
3. Transformuj modelový dataset podľa transformácie vypočítanej v predchádzajúcim kroku.
4. Iteruj od začiatku, ukonči ak nastala ukončovacia podmienka.

Algoritmus sa môže ukončiť splnením viacerých podmienok. Tradične používaná podmienka je počet iterácií, ktorá nám umožní zabezpečiť, že algoritmus bude pracovať pre rôzne, ale zhruba rovnako veľké datasety zhruba rovnaký čas. Druhým často používaným kritériom je veľkosť strednej kvadratickej chyby medzi datasetmi (RMS - Root mean square error). Tretím používaným spôsobom je ukončenie, keď chyba začne klesať veľmi pomaly. Pri nevhodnom nastavení ukončovacej podmienky môže metódou ICP pracovať veľmi dlho. Obrázok 24 zobrazuje prípad registrácie dvoch mračien bodov pomocou tejto metódy.

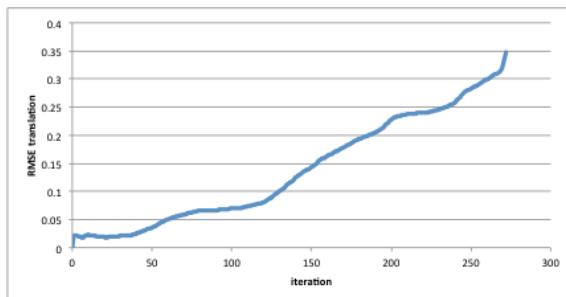


Obrázok 24 - Ukážka registrácie mračien bodov pomocou metódy ICP. Prvý obrázok zľava ukazuje 2 datasety v úvodnej polohe. Obrázok 2 zľava zobrazuje korešpondencie, v tomto prípade najbližších susedov. Obrázok vpravo zobrazuje výsledok po aplikácii ICP [53].

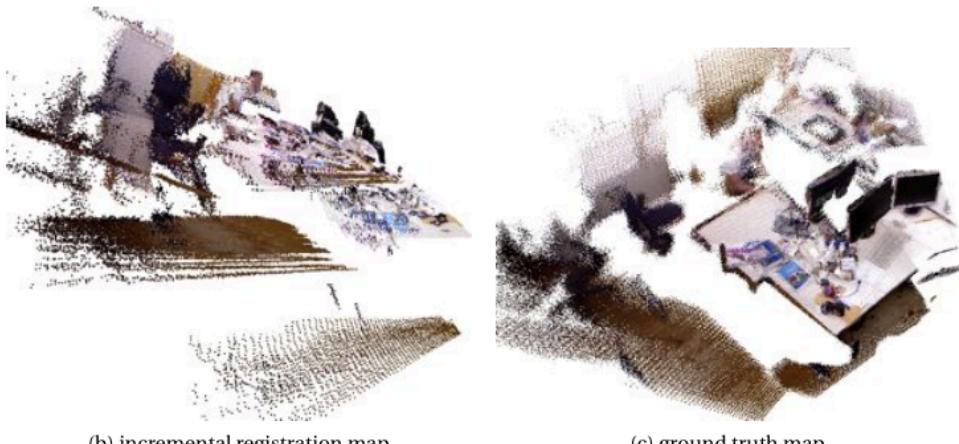
ICP je jedna z najpopulárnejších metód pre registráciu 3D modelov, dokáže pracovať takmer s ľubovoľnou 3D reprezentáciou od mračien bodov, cez krivky, trojuholníkové modely, až po parametricky vyjadrené povrhy. Výpočtová zložitosť algoritmu je $O(N * M * S)$, kde N je počet bodov v cieľovom datasete, M je počet bodov, ktoré zvažujeme pri hľadaní najbližších bodov a S je počet iterácií. Metóda nie je schopná určiť transformáciu v prípade hrubých posunov medzi datasetmi. V tom prípade dôjde ku chybnej asociácií korešpondencií a algoritmus nebude pracovať správne. V praxi sa preto metóda ICP využíva iba pre jemné doladenie hrubo odhadnutej transformácie pomocou iných metód. Metóda bola prvýkrát predstavená na začiatku deväťdesiatych rokov v [52] a [53]. Prepracovaný prehľad metódy ICP, jej zlepšení a aplikácií je k dispozícii v [54].

8.6 Optimalizácia grafu

Vizuálna odometria však prináša rovnaké nežiadúce javy ako klasická odometria, najvýraznejším je drift. Drift vo vizuálnej odometrii vyjadruje, že ak predpokladaná poloha závislá na predpokladanej polohe v minulej iterácii, pravdepodobnosť, že správne určujeme pozíciu s časom klesá. Absolútна poloha robota sa teda s časom stále viac odlišuje od polohy reálnej. Drift reprezentuje mieru rastu tejto chyby. Hovoríme, že rôzne riešenia vizuálnej odometrie akumulujú drift.

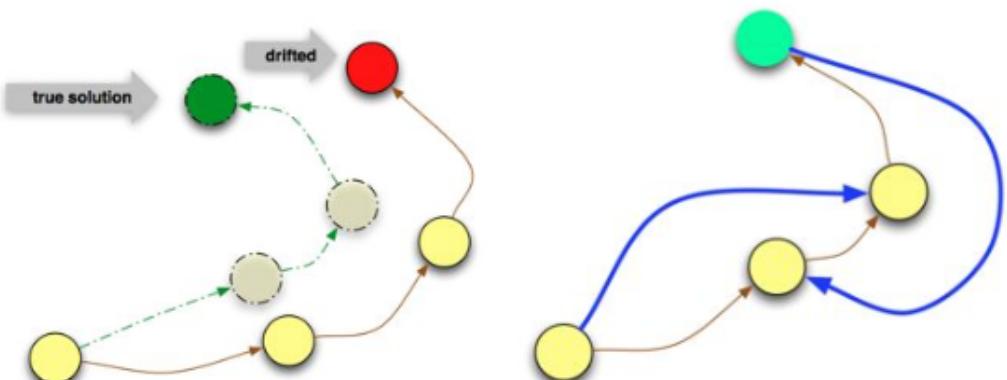


(a) translational error accumulation



Obrázok 25 - Inkrementálna vizuálna odometria vytvára drift. Graf (a) zobrazuje vývoj translačnej chyby od počtu iterácií. Obrázky (b, c) zachytávajú drift pri snímaní zo statickej polohy kamery.

Problém driftu môžeme čiastočne ovplyvniť tak, že transformáciu vo vizuálnej odometrii nebudeme určovať iba zo snímku predchádzajúceho, ale zo súrady predchádzajúcich snímok. Okrem toho však môžeme použiť **metódy pre optimalizáciu grafu (graph optimisation methods, GO methods)**. Graf, na ktorom tieto metódy pracujú je grafom, v ktorom jednotlivé polohy robota reprezentujú vrcholy a transformácie pohybu kamery medzi nimi sú reprezentované hranami. Následne tieto metódy na grafe určia dodatočné hrany medzi snímkami a použijú ich pre upravenie polôh vrcholov.



(a) incremental registration drift in graph (b) with additional edges to correct for the drift

Obrázok 26 - princíp metód optimalizácie grafu. Obrázok vľavo zobrazuje drift v grafe inkrementálnej registrácie, obrázok vpravo zobrazuje pridané dodatočné hrany, ktoré korigujú drift

Medzi najznámejšie algoritmy tohto typu patria HOG-Man [56], TORO [57], iSAM [58] a G²O [59]. Metódy rozdeľujeme na 2 skupiny:

- **Globálne metódy optimalizácie grafu (global graph optimisation)** – tieto metódy pracujú na množine všetkých senzorických meraní. Často vedia identifikovať v grafoch slučky a drift redukujú veľmi výrazne.
- **Lokálne metódy optimalizácie grafu (local graph optimisation)** – ak potrebujeme vykonávať optimalizáciu grafu počas behu algoritmu, nie je možné algoritmu dodať v každom časovom kroku všetky merania. Čas spracovania totiž závisí od množstva spracúvaných vzoriek. Z tohto dôvodu existujú aj lokálne metódy, ktoré pracujú iba na množine predchádzajúcich snímok o preddefinovanej veľkosti. Tieto metódy neimplementujú riešenie problému uzavretia slučky, ale je ich možné spúštať v reálnom čase počas mapovania, ak to dovoľuje výpočtový výkon mobilného robota.

9 Mapovanie

Problematika robotického mapovania zahŕňa veľa rôznych čiastkových problémov, ktoré je potrebné vyriešiť a preto sa aj postupy, ktoré boli pre získanie mapy vytvorené a implementované, v mnohom odlišujú. Táto kapitola predstavuje a porovnáva najčastejšie používané prístupy ku riešeniu problému mapovania pomocou mobilných robotov a formuluje problémy s ktorými sa táto oblasť mobilnej robotiky snaží vysporiadať.

9.1 Kategórie robotického mapovania

V tejto kapitole predstavujeme rôzne prístupy ku riešeniu vizuálneho SLAM s ktorými je možné sa stretnúť v aktuálnej literatúre. V zahraničnej literatúre sa stretнемe v tomto kontexte aj s pojmom **taxonómia robotického mapovania (Taxonomy of robotic mapping)** [60] [61]. Pri každom rozdelení definujeme, na základe čoho riešenia odlišuje a krátko definujeme každú z jednotlivých taxonomických skupín.

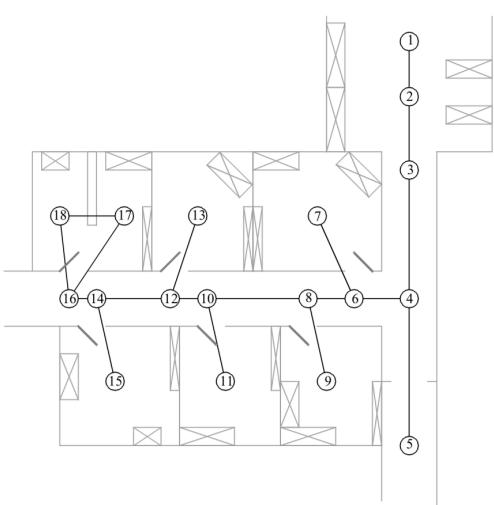
9.1.1 Priestorové a topologické mapy

Historicky základným rozdelením, ktoré vzniklo v tejto oblasti je rozdelenie podľa druhu mapy, ktorú sa metódy snažia vytvoriť na **priestorové** (metrické, objemové) a **topologické** algoritmy [62] [7].

Priestorové mapy prostredia sa snažia presne modelovať geometrické vlastnosti prostredia. Ak je mobilný robot schopný pohybu iba v dvojrozmernom priestore, ako napríklad robotický vysávač, ktorý nie je schopný prekonávať schody, môže byť mapa priestoru, ktorú potrebuje robot pre svoju činnosť reprezentovaná formou analogickou s pôdorysom podlahy tohto priestoru. Ak je robot schopný pohybu v trojrozmernom priestore, ako napríklad lietajúci robot, bude pre plnenie svojich úloh potrebovať presný 3D model tohto priestoru.

Topologické mapy prostredia sa nesnažia priamo popisovať geometrické vlastnosti prostredia, ale miesto sa sústredia na miesta, ktoré sú najvýznamnejšie pre lokalizáciu robota. Topologické mapy môžu byť chápané aj ako určitý stupeň abstrakcie nad geometrickou reprezentáciou. Formálne je topologická mapová reprezentácia zložená

z dvoch prvkov - vrcholov (nodes) a hrán (edges), ktoré reprezentujú konektivitu medzi vrcholmi. Vrcholmi v tejto mape sú miesta, kam sa robot môže dostať a hrany reprezentujú konektivitu medzi týmito vrcholmi. V topologických mapách nemusia mať vrcholy predpísanú veľkosť, ani priestorovú špecifikáciu, predstavujú iba miesto v priestore, ktoré je robot schopný opakovane identifikovať.



Obrázok 27 - topologická mapa prostredia [7]

Pre jednoduchšie pochopenie tohto rozdelenia mapových reprezentácií uvedieme príklad. Predstavme si, že potrebujeme popísť priateľovi, ako sa dostať v meste, ktoré nepozná z bodu A do bodu B. Musíme mu teda poskytnúť mapu priestoru a vyriešiť za neho problém navigácie. Náš priateľ nie je vybavený výdobytkami modernej techniky, nemá k dispozícii smartfón, ani GPS navigátor. Ak sa rozhodneme mu poskytnúť priestorovú mapu, vytlačíme mu snímku príslušnej časti

mesta zo služby Google Maps a na nej vyznačíme cestu, ktorou sa má priateľ vydať. Táto mapa poskytne priateľovi dostatok informácií na presun a navyše ho oboznámi aj s prostredím mimo vymedzenej trasy. Mapa teda obsahuje nie len informácie nevyhnutné k presunu, ale aj iné, dodatočné informácie. Ak však musíme priateľovi vysvetliť cestu telefonicky, kde sme obmedzení množstvom informácií, ktoré mu môžeme predať, môžeme mu poskytnúť topologickú mapu v nasledujúcej forme: Z bodu A prejdi po hlavnej ceste na tretiu križovatku. Na tejto križovatke odboč vpravo. Pokračuj až po veľký kostol. Pri kostole odboč do uličky s trafikou...atď.

Z uvedeného príkladu je zrejmé, že priestorové mapy sú informačne bohatšie. Navigačné úlohy v týchto mapách sa riešia globálne v kontexte daného segmentu mapy a vyžadujú menšiu mieru autonómie robota pre úspešnú navigáciu. Umožňujú efektívne plánovanie trajektórii, po ktorých sa má robot pohybovať a v prípade, že vopred vybraný navigačný scenár zlyhá, je možné plánovanú trasu prepočítať a použiť alternatívne riešenie. Jedinou úlohou, ktorú musí robot vedieť úspešne vyriešiť je problém lokalizácie v tejto mape

– musí vedieť určiť, kde sa nachádza. Topologické mapy sú veľmi výhodné z hľadiska objemu dát, ktoré využívajú, keďže sa sústredujú len na opis naozaj užitočných informácií o prostredí, ale disponujú dramaticky menšou mierou konkrétnosti opisu prostredia. Vďaka ich kompaktnosti je ich navyše možné spracovať s nižšími nárokmi na výkon, ako je to u priestorových máp. Pre navigáciu v topologickej mape však musí robot vedieť úspešne identifikovať jednotlivé vrcholy mapy a navyše byť schopný samostatnej navigácie medzi týmito vrcholmi.

Topologické a priestorové mapy však nie je možné úplne oddeliť. Pre efektívne plnenie úloh v priestorovej mape musíme poznáť miesta, ktoré sú pre vykonávanie týchto úloh podstatné, čo je analogické ku vrcholom topologických máp. Naopak aj tá najabstraktnejšia forma topologickej mapy nebude užitočná, ak nebude obsahovať aspoň elementárne informácie o geometrii priestoru. Komplexné moderné riešenia pre reálne nasadenie v oblasti mapovania preto často kombinujú tieto 2 reprezentácie do **máp hybridných**.

V súčasnej dobe sú priestorové mapy ďaleko najpoužívanejšou formou reprezentácie máp pre mobilné roboty. V prípade potreby totiž dokážeme z dostatočne presnej priestorovej mapy abstrahovať informácie do formy topologickej mapy vždy, keď to potrebujeme.

9.1.2 Mapovanie zamerané na svet a na robota

Druhé rozdelenie, s ktorým sa v literatúre stretávame je rozdelenie podľa priestoru, v ktorom je mapa reprezentovaná. Ak sa rozhodneme mapu reprezentovať v globálnom súradnicovom priestore (2D, 3D) a všetky senzorické merania robota budeme zlučovať do jednotnej mapy v tomto priestore, hovoríme o **mapovaní zameranom na svet** (world-centric). V tejto reprezentácii jednotlivé elementy mapy nedisponujú informáciami, ktoré viedli k ich objaveniu. **Algoritmy mapovania zamerané na robota** (robot-centric) popisujú mapy v priestore senzorických meraní robota, čo predstavuje dramaticky odlišné výzvy pri implementácii riešení lokalizácie a navigácie. Na prvý pohľad by sa mohlo zdáť, že je jednoduchšie vytvárať mapy zamerané na robot, pretože nie je potrebné prekladať senzorické merania do globálneho súradnicového priestoru, avšak v tomto type máp je veľmi náročné odhadovať z jednotlivých meraní informáciu o ich susedstve, prípadne o nezmapovaných miestach v okolí, čo je v globálnych súradničiach typicky jednoduché. Inak povedané, pri týchto mapových reprezentáciách nie je zvyčajne k dispozícii žiadna zjavná geometria

prostredia. Navyše, ak sa dve miesta podobajú, robot má väčší problém ich odlišiť. Z dôvodov vymenovaných v predchádzajúcom texte sa robotický výskum zameriava hlavne na mapy orientované na svet.

9.1.3 Úplný SLAM a online SLAM

Tretím rozdelením je rozdelenie SLAM riešení podľa úlohy, ktorú sa snažia vyriešiť. Takmer všetky úspešné riešenia SLAM problému v posledných rokoch v sebe integrujú princípy **pravdepodobnosnej robotiky** a práve pravdepodobnostné riešenia sa delia na 2 hlavné postupy. Ak sa **pravdepodobostné SLAM** riešenie snaží určiť mapu a zároveň celú trasu robota pri procese mapovania, hovoríme o riešení **úplného SLAM problému** (full SLAM) a môžeme ho definovať ako:

$$p(x_{0:t}m|z_{1:t}, u_{1:t})$$

Ak sa riešenie sústredí na určenie mapy a poslednej polohy robota v tejto mape, hovoríme o **online SLAM probléme**, ktorý je definovaný nasledovne:

$$p(x_t m | z_{1:t}, u_{1:t})$$

Kde:

x_t - stavový vektor obsahujúci informácie o polohe robota

u_t - vektor polohových (kontrolných, ovládacích) príkazov, ktoré robot prijal v časovom kroku $t - 1$ a vykonal ich, aby prešiel do stavu x_t

z_t - vektor senzorických vnemov robota

m_t - podoba vytvorennej mapy

p – pravdepodobostné rozloženie

Pravdepodobostná robotika (probabilistic robotics) je spoločné označenie princípov spoločných pre riešenia z prostredia robotiky, ktoré zohľadňujú **neistotu** (uncertainty) vo **vnímaní** (perception) a **konaní** (action) robota. Kľúčovou myšlienkovou je reprezentovať neistotu explicitne, pomocou matematiky a teórie pravdepodobnosti. Inak povedané, namiesto spoliehania sa na jeden „najlepší odhad“, ako môže byť bežným prípadom v reálnom svete, pravdepodobostné algoritmy reprezentujú informácie pomocou pravdepodobostných rozložení cez celý priestor možných hypotéz. Týmto spôsobom môžu

reprezentovať neurčitosť a stupeň dôveryhodnosti v matematickej prívestivej forme, čo im umožňuje pokryť všetky možné zdroje neistoty a efektívne **potláčať šum** v častiach SLAM systému, kde sa vyskytuje. [63] [62]

9.1.4 Iteratívne a neiteratívne metódy

Štvrté rozdelenie riešení SLAM je podľa rozsahu dát ku ktorým pristupujú pri tvorbe mapy prostredia. Metódy, ktoré **používajú sériu posledných senzorických vnemov** pre určenie aktuálnej pozície v mape nazývame **iteratívne metódy**. Výhodou týchto metód je, že bežia nad určeným rozsahom dát, od ktorého sa odvíja aj výpočtová zložitosť týchto algoritmov. Vizuálna technika RGBD-6D-SLAM napríklad používa pre určenie aktuálnej pozície aktuálnu snímku a sekveniu desiatich predchádzajúcich snímok [64]. Kedže pre určenie pozície nie je potrebné poznať všetky merania, tieto metódy dokážu bežať v reálnom čase. Významné riešenia čiastkových problémov SLAM sa taktiež často vyznačujú iteratívnym charakterom a viaceré z takýchto iteratívnych algoritmov budú predstavené v ďalších častiach práce. Neiteratívne riešenia SLAM, ktoré potrebujú pre svoju činnosť všetky dostupné dátá vznikali v počiatkoch robotického mapovania a v súčasnej dobe sa už prakticky nevyvíjajú.

9.1.5 Aktívny a pasívny SLAM

Piatym rozdelením je rozdelenie na **aktívny a pasívny SLAM**. Toto rozdelenie úzko súvisí s problémom preskúmavania prostredia, ktorý rieši, ako efektívne získavať mapy prostredia v zmysle plánovania trajektórie pohybu robota. Pri **pasívnom prístupe** je robot riadený externým riadiacim systémom, ktorý plánuje jeho pohyb. Ak by ste si chceli takéto riešenie predstaviť, predstavme si robota riadeného operátorom pomocou teleriadenia pri prieskume miesta nešťastia. Ak robot navyše disponuje pasívnym SLAM systémom, pri svojom pohybe zároveň vytvára aj mapu tohto prostredia, čo môže vzdialenému operátorovi významne uľahčiť orientáciu v priestore, v ktorom sa robot pohybuje. Operátor tak môže naplánovať efektívne trajektórie presunov na vopred známe miesta, prípadne sa vyhnúť mestam, v ktorých robotu hrozí poškodenie podľa predchádzajúcich meraní, aj keď sa k nim blíži z inej strany. **Aktívny prístup ku riešeniu SLAM** v sebe integruje aj systém riadenia trajektórie tak, aby bolo mapovanie čo najefektívnejšie. Robot je teda schopný samostatne preskúmavať priestor, plniť v ňom úlohy a rozhodovať sa, kam sa vydá v budúcnosti. Problém, ktorý popisuje, ako nájsť najlepšie riešenie pre preskúmanie neznámeho prostredia nazývame

problémom preskúmavania prostredia (map exploration problem). Takýto prístup musíme zvoliť napríklad v prípade robota umiestneného na Marse. Oneskorenie riadiaceho rádio-signálu medzi Zemou a Marsom môže byť v rozsahu 4-20 minút. Ak by sme mali robota riadiť zo Zeme, po každom pohybovom príkaze by sme museli čakať takto dlhú dobu. Preto je nevyhnutné, aby prieskumný robot na planéte Mars v sebe integroval aj aktívny prístup ku riešeniu problému SLAM. Inak povedané, aby aktívne preskúmaval prostredie okolo seba [61].

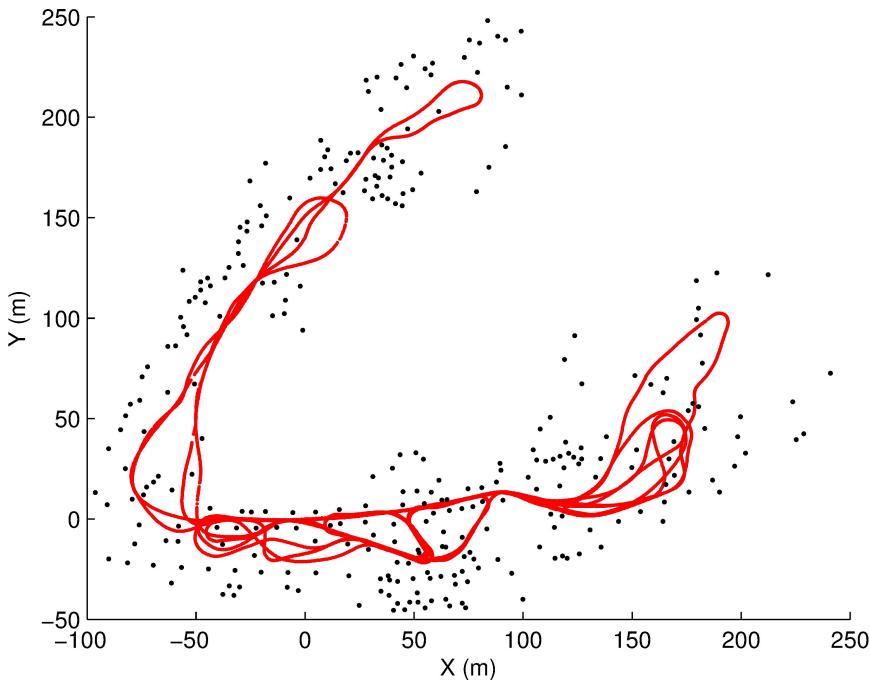
9.1.6 Jeden robot a viaceré roboty

Iné rozdelenie SLAM algoritmov rozdeľuje riešenia podľa počtu robotov, ktoré používajú. **Riešenia pre jeden robot** (single-robot SLAM) predstavujú väčšinu publikovaných riešení. S väčšou dostupnosťou robotických platform s dostatočným výkonom sa však dostávajú do popredia aj **riešenia pre viaceré roboty** (multi-robot SLAM). Tieto riešenia predstavujú široké spektrum rôznych postupov. V niektorých sa musia roboty priamo vidieť, resp. byť schopné sa navzájom identifikovať v priestore. V niektorých sú roboty vopred oboznámené so svojimi relatívnymi pozíciami. Roboty môžu medzi sebou komunikovať prostredníctvom veľkého množstva komunikačných technológií, ako je napríklad rádio, bluetooth alebo wifi. Pri reálnych aplikáciach je potrebné brať do úvahy aj oneskorenie (latenciu) a šírku komunikačného pásma (bandwidth) danej technológie [61] [7].

9.1.7 Priestorový SLAM a SLAM pomocou orientačných bodov

Ďalším rozdelením, s ktorým sa v literatúre stretávame je rozdelenie na **priestorové SLAM riešenia** (volumetric SLAM) a **SLAM riešenia postavené na orientačných bodoch** (landmarks). Priestorové SLAM riešenia sa sústredujú na modelovanie priestoru prostredia. Tieto mapy sa prirodzene vyznačujú viac-rozmernosťou a výpočtový výkon potrebný na ich spracovanie je značný. V prípade pravdepodobnostných riešení modelujú tieto prístupy aj pravdepodobnostné rozdelenia častí vytvorennej mapy. SLAM riešenia postavené na orientačných bodoch určujú priestorové pozície unikátne identifikovateľných orientačných bodov v senzorických vnemoch. Pre predstavenie si riešenia SLAM pomocou techník orientačných bodov pre nás je model, ktorý používa človek. Pri prechode cez nové mesto sa pravdepodobne budeme snažiť zapamätať si unikátne budovy, stromy a iné objekty - orientačné body, ktoré nám pomôžu sa zorientovať neskôr. Nevýhodou tohto prístupu je zdedená neistota. Ak určujeme polohu nového orientačného bodu, jeho poloha bude určená

so zdedenou chybou určenia pozície z predchádzajúcej iterácie. Posledná nevýhoda postupu postaveného na orientačných bodoch je zjavná – určujeme iba pozície orientačných bodov a ostatné senzorické údaje nevyužívame. Presnosť týchto metód je rádovo horšia ako presnosť priestorových SLAM riešení, ale v minulosti boli hojne využívané hlavne z dôvodu limitovaného dostupného výpočtového výkonu na mobilných robotoch [7].



Obrázok 28 - Výsledok mapovania Victoria park 2D datasetu. Červenou je vyznačená dráha robota, čierne bodky predstavujú orientačné body.

9.1.8 Známa a neznáma korešpondencia

Problém korešpondencie predstavuje problém asociácie identity vnímaných objektov ku iným objektom. Ak v aktuálnom senzorickom vneme robot zaregistruje objekty, ktoré by mohli byť kandidátmi na orientačné body, musí ich úspešne asociovať s orientačnými bodmi z predchádzajúceho merania, aby vedel určiť, ktoré z nich vníma opakovane a ktoré nie. Problém korešpondencie je podrobne rozobratý [61].

9.1.9 Statické a dynamické prostredia

Jedno z posledných rozdelení SLAM algoritmov berie do úvahy charakter sveta, ktorého model sa robot prostredníctvom mapovania snaží vytvoriť. Drvivá väčšina algoritmov predpokladá, že prostredie je **statické**, čiže sa v čase nemení a modelujú dynamické elementy prostredia, napríklad pohybujúce sa osoby iba na úrovni senzorického šumu. **Dynamické**

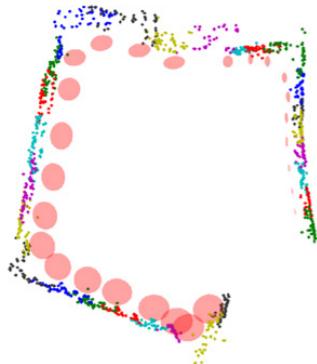
SLAM riešenia sú schopné pracovať v premenlivých prostrediach. Na jednej strane môžu v mape lokalizovať dynamické objekty a filtrovať ich, ale môžu ich aj v týchto mapách zahŕňať a využiť znalosť o ich existencií napríklad pri obchádzaní dynamických prekážok.

9.1.10 Veľká a malá neistota, problém uzavretia slučky

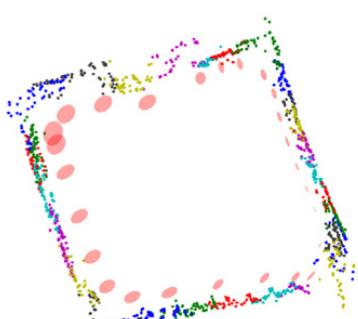
SLAM riešenia môžeme charakterizovať aj podľa stupňa neistoty určenia pozície, ktorú dokážu ešte spracovať. Najjednoduchšie algoritmy povoľujú iba malé neistoty určenia pozície pre svoj beh. Môžu byť výhodné pri trajektóriách robota, ktoré sa so sebou nepretínajú, čiže neobsahujú tzv. slučky. Vo viacerých prostrediach je ale možné dosahovať určité pozície z viacerých smerov. V týchto miestach potom robot často akumuluje neistotu, čo vedie ku zlyhaniu konzistenie mapy. Tento problém je známy ako **problém uzavretia slučky (loop closure problem)**. Schopnosť riešiť problém uzavretia slučky je jednou z kľúčových charakteristik úspešného SLAM riešenia vhodného pre veľké prostredia. Neistota tohto typu sa dá úspešne eliminovať použitím globálneho pozičného systému v rámci absolútneho súradnicového systému, napríklad použitím GPS (Global Positioning System) v ktorom absolútna chyba určenia polohy s časom nerastie.



(a) Aerial view of the courtyard.



(b) Before loop closure.



(c) After loop closure.

Obrázok 29 - Loop closure, problém uzavretia slučky. Obrázok A zobrazuje leteckú snímku oblasti. Obrázok B ukazuje príznaky rozpoznané počas mapovania ako farebné body. Kružnice zobrazujú polohu robota a neistotu, s akou je jeho poloha určená. Obrázok C zobrazuje mapu po uzavretí slučky. Mapa sa uzavrela a rozloženie neistoty polohy robota v jednotlivých pozíciách bolo prepočítané. Mapa bola znova vygenerovaná, avšak s nižšími neistotami, čo dramaticky zlepšilo korešpondenciu mapy s reálne snímaným priestorom. [65]

9.2 Priestorové mapové reprezentácie

Priestorové mapové reprezentácie modelujú priestor vo forme najbližšej ľudskému vnímaniu. Používajú reálny 3D priestor v reálnych mierkach a tak sú prirodzené ľudskému vnímaniu.

Mnoho najlepších metód, ktoré používa robotické mapovanie má svoj pôvod v 3D počítačovej grafike. Najvýznamnejšia oblasť, v ktorej sa tieto 2 témy prelínajú je hľadanie odpovedi na otázku, ako efektívne reprezentovať objekty v 3D priestore. Pre pochopenie konceptov priestorových mapových reprezentácií je preto vhodné poznať niektoré pojmy 3D počítačovej grafiky.

Z hľadiska 3D počítačovej grafiky môžeme 3D reprezentácie rozdeliť na:

- **Priestorové dekompozície (solid , space-partitioning)**, ktoré a snažia definovať priestorovú obsadenosť objektov v nich uložených,
- **povrchové (shell/boundary/surface)**, ktoré modelujú povrch objektov a
- **systémy častíc**. Pomocou nich sa modelujú komplexné objekty ako priesok, vodný sprej, oblaky... Vo veľkej miere sa využívajú pri simuláciách týchto materiálov vo fyzike. Individuálna častica môže byť tvorená polygónom alebo bodom.

Priestorové dekompozície rozdeľujú priestor do voxelov. **Voxel** je jednotka priestoru vyjadrená pomocou súradnice na pravidelnej mriežke vytvorenej v trojrozmernom priestore a je to elementárna jednotka priestorovej dekompozície priestoru. Je to ekvivalent pixelu z 2D obrazu. Vo vzťahu ku reálnemu svetu a použitia voxelu pre 3D mapovanie v mobilnej robotike, musí každý voxel definovať svoj fyzický rozmer. V tomto prípade hovoríme, že sa jedná o **približnú dekompozíciu priestoru mapy (approximate cell decomposition)**. Pri vytváraní mapy môžeme využiť aj **presnú dekompozíciu priestoru (exact decomposition)** – v tom prípade budú mať voxely navzájom odlišný tvar, budú uložené aj informácie o hranách a mapa bude modelovať prostredie bezstratovo [7].

Povrchové reprezentácie v 3D grafike sa môžu skladať z viacerých typov objektov, ktorých množiny popisujú povrch uloženého objektu. Polygónové modely sú zložené z polygónov. **Polygón** je planárne teleso definované konečnou množinou úsečiek tak, že formujú uzavretý objekt. Tieto úsečky nazývame hranami alebo stranami polygónu a miesta, kde sa 2 úsečky stretávajú nazývame vrcholmi polygónu (vertices). Polygóny klasifikujeme

podľa mnohých geometrických atribútov. V robotickom mapovaní sa často stretávame s typom trojuholníkových polygónov. Pre reprezentáciu povrchu sa používajú okrem polygónov aj krvky. V tomto prípade sa mapové reprezentácie skladajú z krviek a bodov, ktoré ich zaťažujú váhou a tým definujú ich tvar. Tento formát robotickej mapy je však veľmi zriedkavý. Systémy častíc popisujú 3D objekty ako množiny bodov v priestore, kde každý bod je definovaný pomocou jeho súradníc v tomto priestore. Elementárnu jednotkou systému častíc je **bod**, analogický s geometrickým bodom. V prípade 3D mapovania môže so sebou niesť aj iné informácie, ako napríklad farbu, intenzitu odrazeného lúča, teplotu atď.. V prípade hĺbkovej kamery odpovedá hodnota nameraná pre jeden pixel vzdialenosť roviny snímača a prieniku priestorovej projekcie tohto pixela a snímaného povrchu. V systémoch častíc bod tradične nemá veľkosť, ale musí mu byť priradená pri procese renderingu a zobrazovania.

S problémom robotického mapovania súvisia aj 2 pridružené problémy 3D grafiky. Prvým problémom je **problém 3D skenovania (3D scanning)**, ktorý popisuje spôsob získavania 3D modelov reálnych fyzických predmetov. Druhým pridruženým problémom je problém **3D rekonštrukcie (3D reconstruction)**, ktorý popisuje ako získavať 3D modely prostredí. V oboch týchto aplikáciách sme v posledných rokoch boli svedkami významného pokroku [66] [67] [31]. Problém 3D rekonštrukcie sa môže zdať identickým s problémom robotického mapovania, avšak tieto problémy sú oddelené. Problém 3D rekonštrukcie sa snaží získať modely prostredí s najvyššou možnou presnosťou, často bez ohľadu na výpočtový výkon potrebný pre ich spracovania a proces mapovania samotný. Problém robotického mapovania rieši širší kontext, zohľadňuje výpočtové možnosti robota a presnosť, ktoré musí robot pri svojej práci dosahovať. Z hľadiska robotického mapovania je problém výberu mapovej reprezentácie problém analogický s reprezentáciou samotnej polohy robota. Výber reprezentácie mapy totiž môže ovplyvňovať aj možnosti definície polohy robota ako takej. Pre výber vhodnej mapovej reprezentácie si preto musíme uvedomiť najmä že:

1. Presnosť mapy musí vhodne korešpondovať s presnosťou, ktorú robot potrebuje dosiahnuť pre úspešné vykonávanie svojich úloh. Robot zároveň musí disponovať senzorickým systémom s dostatočnou presnosťou.
2. Komplexnosť mapy má priamy dopad na výpočtovú zložitosť riešenia úloh lokalizácie, mapovania a navigácie

Za priestorovú mapovú reprezentáciu môžeme považovať aj mapu orientačných bodov (viz. 9.1.7) [62] [68].

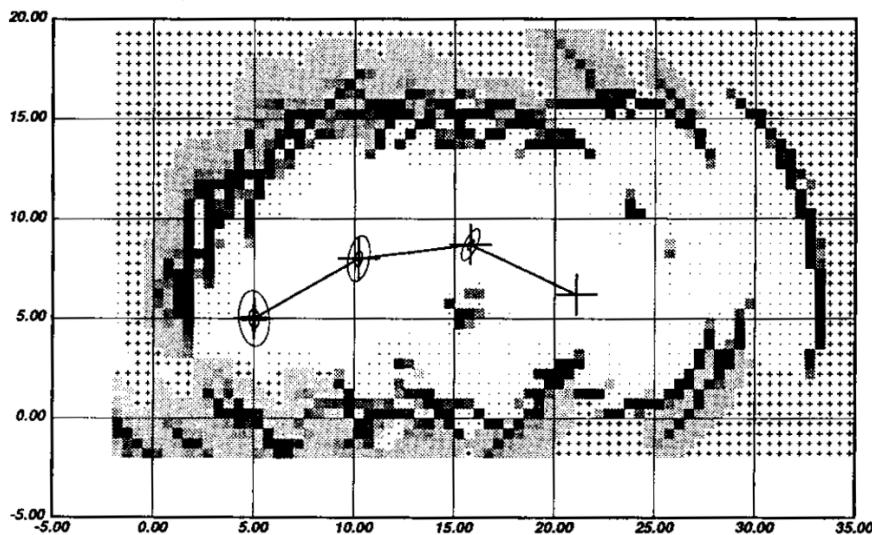
9.2.1 Mriežka obsadenia

Skorým predstaviteľom priestorových mapových reprezentácií je **mriežka obsadenia** (occupancy grid). Mriežka obsadenia je viacozmerné pole, ktoré udržuje stochastické hodnoty stavu obsadenosti v bunkách priestoru. Bunky tohto priestoru sú rovnako veľké a rozmer jednotlivej bunky môžeme označiť aj ako rozlíšenie tejto mriežky obsadenia. Mapa je vytvorená tak, že na senzorické vnemy mobilného robota sú interpretované do hodnôt obsadenosti časti priestoru, ktorá zodpovedá snímanej časti prostredia. Táto interpretácia je implementovaná pomocou pravdepodobnostných metód, čo umožňuje inkrementálne aktualizácie mriežky obsadenia pomocou dát z viacerých snímačov a z viacerých uhlov pohľadu.

Mriežka obsadenia umožňuje riešenie viacerých problémov v oblasti robotiky – napríklad vzdialenosné mapovanie, problém integrácie viacerých snímačov, plánovanie trajektórie a problém navigácie. Algoritmus má 2 hlavné nevýhody. Po prve, veľkosť mapy prostredia je priamo úmerná fyzickým rozmerom zmapovaného prostredia. Mriežka neposkytuje žiadnu úroveň kompresie, pre každý diskrétny blok priestoru je alokovaná pamäť a je v ňom uložená hodnota jeho obsadenosti. Hovoríme o fixnej dekompozícii priestoru. Ak týmto postupom mapujeme veľké prostredia, alebo použijeme veľmi vysoké rozlíšenie (veľkosť bunky bude príliš malá), veľkosť mapy bude v čase narastať do extrémnych hodnôt. Pri takejto forme reprezentácie mapy zároveň predpisujeme formu priestorovej dekompozície bez akýchkoľvek znalostí o reálnej geometrii priestoru, čo sa môže v niektorých prípadoch ukázať ako nevhodné.

Mriežka obsadenosti funguje aj v 2D aj v 3D a vo väčšine implementácií umožňuje ukladanie dodatočných dát ku každej bunke priestoru. V týchto dátach sa môžu nachádzať napríklad informácie o farbe alebo teplote danej časti priestoru.

Z hľadiska 3D počítačovej grafiky chápeme mriežku obsadenia ako množinu voxelov.



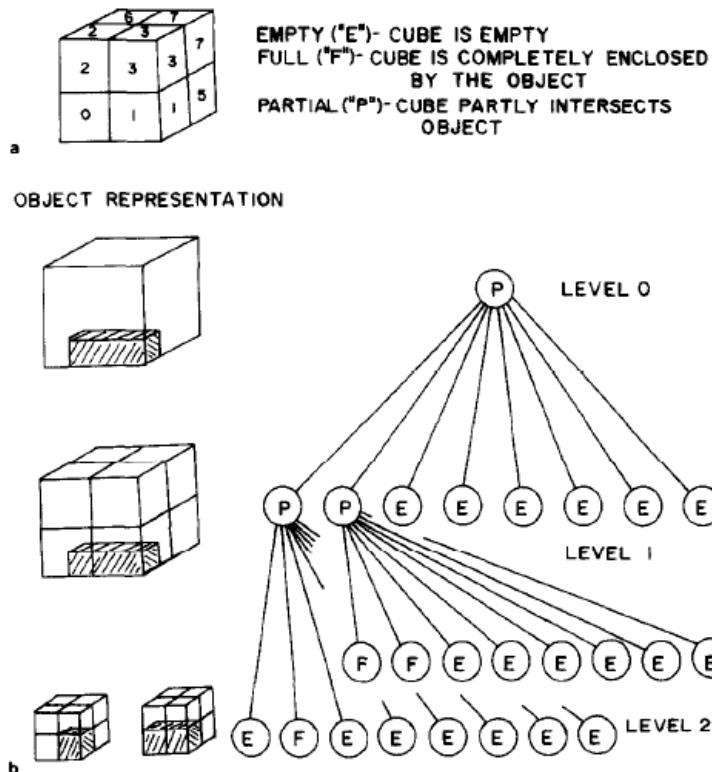
Obrázok 30 - Obrázok mapy pomocou mriežky obsadenia. Odtiene šedej reprezentujú úroveň pravdepodobnosti obsadenia príslušného segmentu priestoru. V strede mapy je znázornená trajektória robota, ktorá viedla ku vytvoreniu tejto mapy. [69]

9.2.2 Stromové reprezentácie, oktálsový strom

Väčšina nevýhod reprezentácie mapy pomocou mriežky obsadenosti bola skoro vyriešená v pomocou stromových reprezentácií. V prípade 2D stromových máp najčastejšie hovoríme o štruktúre **quadtree**, v prípade 3D štruktúry hovoríme o **oktálsovom strome (octree, octree encoding)** alebo o **oktostrome**.

V priestorovom vyjadrení sú tieto typy tátu štruktúr ekvivalentné so štruktúrou mriežky obsadenia. Taktiež sa skladajú z buniek priestoru, voxelov. Rozdiel týchto reprezentácií spočíva v štruktúre ukladania údajov o obsadenosti priestoru. V mriežke obsadenia sa ukladá hodnota obsadenosti pre každý jeden voxel, bez ohľadu na to, či je tento voxel naozaj obsadený. **Oktálsový strom** však ukladá dátá v **hierarchickej stromovej štruktúre**, v ktorej **uzly** (nodes) stromu predstavujú nesúvisle usporiadane kocky exponenciálne klesajúceho objemu. Každý uzol stromu korešponduje s jemu prislúchajúcou časťou súradnicového priestoru. Každý uzol stromu, ktorý kompletne popisuje jemu prislúchajúci priestor, je **listom**. Ak uzol nie je list, tak má 8 potomkov, ktoré priestorovo prislúchajú ôsmim regiónom – oktantom v jeho priestorovom rámci 3D priestoru. V prípade quadtree, čiže analogickej štruktúry v 2D priestore má každý uzol, ktorý nie je listom, práve 4 potomkov.

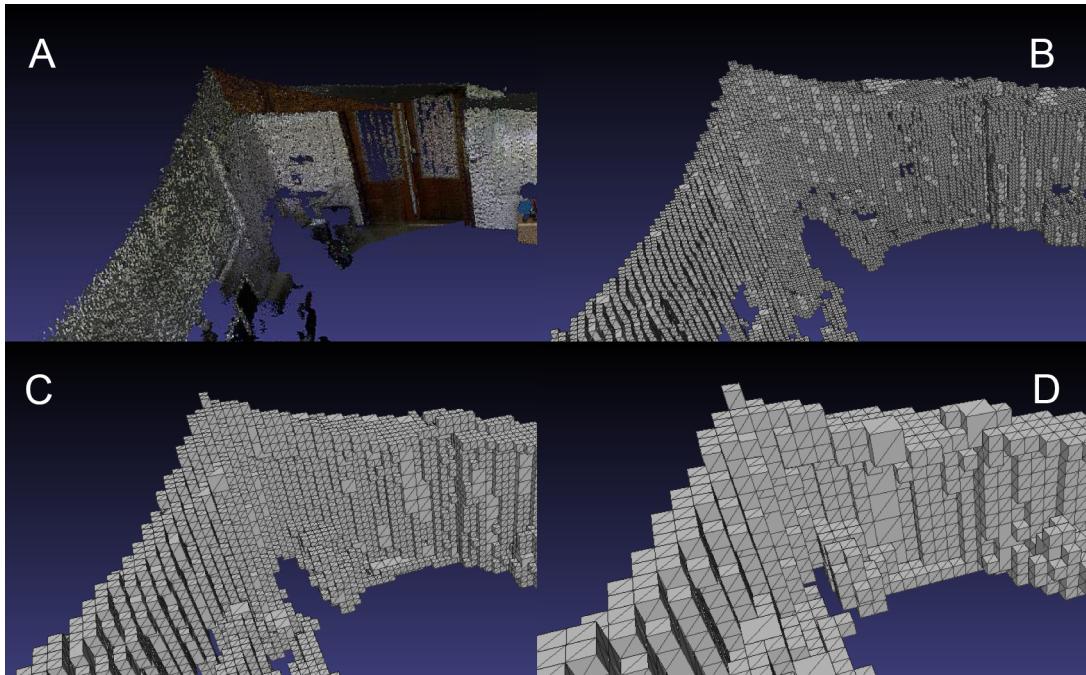
OCTREE ENCODING



Obrázok 31 - Jednoduchý objekt uložený v štruktúre oktálového stromu. A.) Sekvencia číšlovania a definície označení, B.) Troj-úrovňová reprezentácia zakódovaného objektu. Vpravo zobrazenie stromovej štruktúry, vľavo priestorové zobrazenie

Oktálový strom disponuje radou výhod. Poprvé, akýkoľvek objekt dokážeme reprezentovať s rozlišením prislúchajúcim najmenšiemu rozmeru kocky. Podruhé, na štruktúre oktálového stromu je implementované veľké množstvo operácií, či už analytických alebo manipulačných, ktoré je možné využívať nezávisle od objektu v nej reprezentovaného. Väčšina týchto operácií je navyše implementovaných veľmi efektívne z hľadiska požadovaného výpočtového výkonu. Po tretie, všetky objekty sa ukladajú v priestorovo predriedenej forme. Ak prechádzame oktálový strom správnou sekvenciou, vieme sa napríklad presúvať medzi voxelmi vo vybranom smere. Po štvrté, je jednoduché vykonávať binárne operácie nad dvoma alebo viacerými oktálovými stromami. Algoritmus v tomto prípade len musí prejsť oba stromy v správnom poradí, aby vedel vytvárať výstupný strom. Navyše, uzly na určitej úrovni (level) oktálového stromu reprezentujú rôzne úrovne rozlíšenia, ktoré nám oktálový strom dovoľuje prehľadávať, a tak je možné vybrať úroveň presnosti potrebnú pre konkrétnu aplikáciu. Táto vlastnosť zároveň umožňuje efektívny prenos dát mapy cez sieť. Mapu je

možné prenášať od menej početných úrovní stromu a tak mapu najprv prenieť v nižšom rozlíšení a potom ju priebežne aktualizovať presnejšími dátami a to aj po častiach. Pri opakovanom snímaní už jestvujúceho segmentu mapy zároveň nezvyšujeme množstvo zaznamenaných vzoriek [70]. Myšlienka použitia oktostromov v 3D mapovaní sa stala okamžite populárnoch a dnes tvorí de facto štandard v robotickom mapovaní. Jeden z väčších projektov, ktorý túto myšlienku rozpracoval a úspešne implementoval je OctoMap [71] .



Obrázok 32 - A) Mapa vo formáte mračna bodov, B) Mapa vo formáte oktostromu s hĺbkou 8, C) S hĺbkou 7, D) s hĺbkou 6 [72]

9.2.3 Mračná bodov

Z hľadiska robotického mapovania je pomerne novým formátom reprezentácia pomocou mračna bodov (point cloud representation). Ak chceme mapu reprezentovať v tomto formáte, jednoducho zlúčime všetky vzdialenosné merania, ktoré robot nameradal, do globálneho súradnicového priestoru. Mračno bodov je teda množina bodov reprezentovaných priestorovými súradnicami x , y a z . Jednotlivé body reprezentujú povrch snímaného prostredia, zmeraný z príslušnej pozície robota. V závislosti od použitého snímača môžeme tieto body dopĺňať dodatočnými informáciami, napríklad farbou. Farebné mračná bodov sú štandardným výstupným formátom RGBD kamier. Ich výstupom je sekvencia snímok v čase, pričom každá snímka je tvorená množinou bodov, nameraných v konkrétnom časovom

okamihu, kde každý bod obsahuje informácie o svojej priestorovej polohe, ale aj informácie o farbe. Ak poznáme pohyb RGBD kamery v čase, mapu vo formáte mračien bodov vytvoríme triviálnou integráciou jednotlivých senzorických meraní. Ďalším snímačom, ktorého tradičným výstupom sú mračná bodov, sú 3D laserové skenery. V tomto prípade body nenesú informáciu o farbe, ale zväčša disponujú informáciou o intenzite odrazeného signálu, čo môže byť užitočné pri identifikácii charakteru povrchu snímaných objektov alebo pri identifikácii nesprávnych meraní.

V posledných rokoch sa výskum intenzívne zameriava v oblasti spracovania mračien bodov pre riešenia problémov 3D skenovania a 3D rekonštrukcie. Najvýznamnejšou iniciatívou v tomto smere je **Point Cloud Library, PCL** [73]. PCL je knižnica s otvoreným zdrojovým kódom, ktorá rieši široké spektrum problémov spracovania 3D obrazu a mračien bodov. PCL obsahuje veľké množstvo najmodernejších algoritmov vrátane filtrácie, detekcie význačných bodov, rekonštrukcie povrchu, registrácie jednotlivých snímok, segmentácie objektov a mnohých ďalších. PCL umožňuje napríklad implementáciu riešenia, ktoré odfiltruje zašumené mračná na vstupe, spojí ich do jedného modelu, identifikuje význačné body a vypočíta ich deskriptory a na ich základe úspešne identifikuje objekty v snímanom prostredí. PCL obsahuje aj nástroje pre vizualizáciu a vizuálnu inšpekciu týchto mračien bodov.

Nevýhody reprezentácie pomocou mračien bodov sú najmä:

- a.) **Obmedzenia veľkosti** – veľkosť vytvorenej mapy rastie lineárne s počtom integrovaných snímok do celkovej mapy.
- b.) **Vysoké požiadavky na výpočtový výkon pri spracovaní** – pre spracovanie veľkých vzoriek dát je potrebný veľký výpočtový výkon. Mračno bodov predstavuje nepredspracované, surové dáta zo snímača a pre vykonávanie elementárnych geometrických operácií je často potrebná iterácia cez všetky body celej spracovávanej vzorky. Tento problém je do určitej miery možné riešiť paralelným spracovaním viacerých bodov naraz, ale to si vyžaduje podporu u spúšťaného algoritmu a hardvér, ktorý je schopný paralelného spracovania.
- c.) **Redundancia dát** – ak chceme spojiť 2 snímky z RGBD kamery, ktoré boli zosnímané tak, že sa ich zorné polia prekrývali, budú mračná bodov v mieste prekrytia redundantné – budú obsahovať pre daný segment priestoru vyššiu hustotu bodov, ako

je hustota na výstupe snímača. Pri malých posunoch kamery medzi snímkami môže byť redundancia veľmi výrazným problémom.

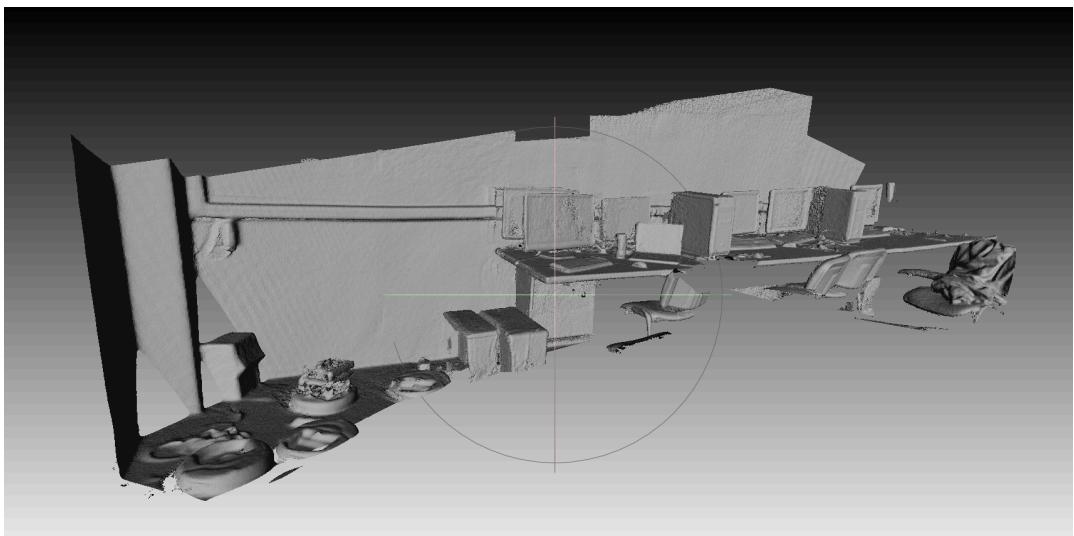
- d.) **Zlá využiteľnosť pre riešenie štandardných úloh robotiky.** Aj keď je možné pokúsiť sa plánovať trajektóriu aj v mračne bodov, táto úloha je neporovnatelne výpočtovo zložitejšia, ako pri použití niekorektnej diskrétnej dekompozície priestoru, ako napríklad mriežky obsadenia.

[73] [74] [75]

9.2.4 Povrchové reprezentácie

Najčastejšie využívaným spôsobom 3D reprezentácie povrchu v 3D počítačovej grafike je **sietová mapa - mesh**. Mesh je tvorený súborom polygónov, ktoré popisujú povrhy zosnímaných objektov. Najčastejšie používané sú jednoduché polygóny ako trojuholníky a štvoruholníky, ale sietové mapy môžu používať aj iné tvary polygónov. Existujú vo veľkom množstve formátov, je na nich efektívne implementovaných veľa operácií počítačovej grafiky ako booleanovské operácie, vyhľadzovanie hrán, rendering a detekcia kolízii. V robotickom mapovaní sa sieťové modely často využívajú pre zobrazovanie máp. Mesh mapy sú taktiež tradičným formátom využívaným v 3D skenovaní, keďže často dokážu aproximovať zložité geometriu pomocou malých polygónov efektívnejšie, ako napríklad reprezentácie postavené na voxeloch. Mesh mapy často predstavujú presnejší spôsob vyjadrenia výstupných dát z RGBD kamery ako systémy častíc. Rovinná veľkosť pixela snímaného kamerou totiž závisí od vzdialenosť od kamery a tak je presnejšie, keď vieme definovať rozmer snímaného polygónu. Naopak mračná bodov používajú jednotnú veľkosť pre každý bod, čo je do značnej miery nepresné. Ak sa mesh model generuje z mračna bodov, zväčša sa skupiny bodov approximujú priemerom. Efektívne sa tak znižuje množstvo vzoriek, ktoré je potrebné pre pokrytie prislúchajúceho segmentu mapy. To znamená, že ak sa aktualizuje už existujúci segment mapy, nezvyšuje sa počet vzoriek, podobne ako pri octree reprezentácii.

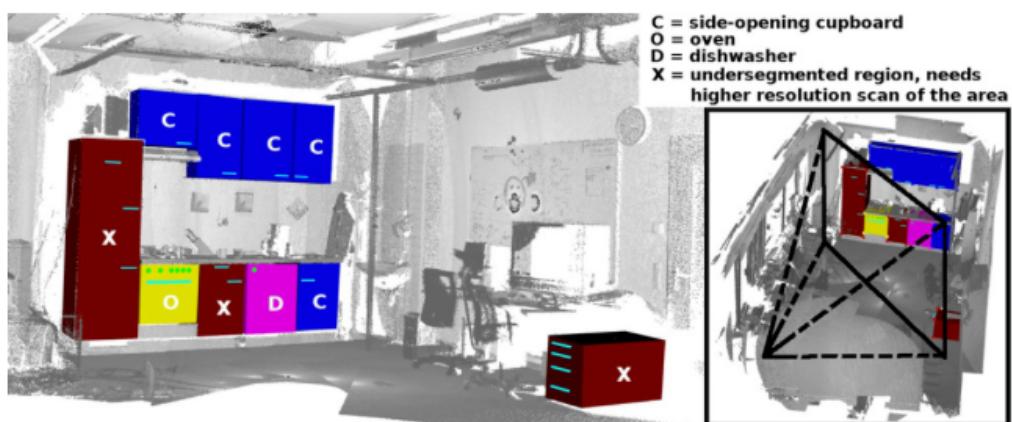
Nevýhodou povrchovej reprezentácie vo forme meshu je výpočtová zložitosť jej určenia z mračna bodov, ktoré prichádza z hĺbkovej kamery. Pre mesh zodpovedajúci povrhom realite je nutné korektne spočítať všetky polygóny a ich normály. Tento postup nazývame aj **meshing** mračna bodov. Na CPU je to výpočtovo náročná operácia, na GPU sa už sú však dostupné aj implementácie dosahujúce vysokú kvalitu aj rýchlosť spracovania.



Obrázok 33 - Mesh mapa vytvorená pomocou techniky Kinect Fusion, ktorá bude predstavená v ďalších častiach práce

9.2.5 Objektové 3D mapy

V súčasnosti rozšíreným postupom pri získavaní informačne bohatých máp a zároveň postupom, ktorý je snáď najbližšie vnímaniu človeka, sú **objektové mapy** (object maps). Ich podstata spočíva v identifikácii každodenných objektov v scéne, ktoré potom využívajú ako orientačné body. Algoritmy tvoriace takéto mapy využívajú poznatky z viacerých vedných disciplín ako počítačová grafika, počítačové videnie, robotika a strojové učenie. Takéto mapy nesú vysokú informačnú hodnotu. Ich orientačné body sú tvorené reálnymi objektmi a bývajú správne lokalizované v priestore. Algoritmy sú schopné zdokonaľovať sa v rozpoznávaní a klasifikácii objektov. Ich dramatickou nevýhodou je však komplexnosť a výpočtová zložitosť vytvorenia takejto mapy [76] [77] [78].



Obrázok 34 - Objektová mapa vytvorená pomocou algoritmu popísaného v [77]

9.3 Problémy robotického mapovania

V tejto kapitole práca poskytuje prehľad štandardne formulovaných problémov robotického mapovania, s ktorými sa stretneme v literatúre.

9.3.1 Šum

Aby dokázal robot získať mapu prostredia, v ktorom sa nachádza a pohybuje, musí disponovať exteroceptínymi senzormi, ktoré mu umožnia vnímať svet naokolo. Senzory, ktoré sa na toto zvyčajne používajú, sú napríklad kamery, laserové skenery, sonary, infračervené snímače, radar, snímače dotyku, ultrazvuky, kompasy a GPS. Pre spresnenie však môžeme integrovať aj snímače proprioceptívne ako IMU alebo odometriu. Senzory majú svoj **šum** a majú svoje priestorové obmedzenia. Práve tieto obmedzenia znamenajú, že robot potrebuje pre svoj úspešný pohyb riešiť aj navigačnú a lokalizačnú úlohu. **Pohybové príkazy (motion commands)**, ktoré boli robotu počas preskúmavania mapy dodané, taktiež nesú veľmi dôležité informácie o tom, kde sa robot pri danom senzorickom meraní nachádzal. Pohyb robota je však tiež zaľažený chybou, takže pohybové príkazy časom prestanú byť dostatočné pre určenie pozície robota (miesto a rotácia) relatívne voči prostrediu. Kľúčový problém v robotickom mapovaní je povaha šumu týchto meraní. Problémy modelovania - ako problém robotického mapovania - sú zvyčajne relativne jednoducho riešiteľné v prípade, keď sú jednotlivé merania *štatisticky nezávislé*. V tomto prípade by stačilo robiť stále viac meraní, a tým by bolo možné eliminovať efekty šumu. Nanešťastie, v robotickom mapovaní sú merania *štatisticky závislé*, pretože chyby v pohybe robota sa časom akumulujú a navyše aj ovplyvňujú spôsob, akým bude nasledujúce senzorické meranie interpretované. Ovládnutie šumu je preto kľúčovým prvkom pri úlohe úspešného vytvárania máp, ale zároveň aj jej kľúčovým komplikujúcim faktorom. Z týchto dôvodov je veľa mapovaných algoritmov až prekvapujúco komplexných z implementačného hľadiska.

9.3.2 Viacozmernosť

Druhým faktorom, ktorý významne komplikuje vyriešenie problému robotického mapovania je problém jeho viacozmernosti. Pre pochopenie si môžeme predstaviť, koľko čísel by si vyžadovalo popísanie prostredia nášho domova. Ak by sme toto prostredie popísali pomocou topologických entít ako chodieb, dverí a miestností, postačilo by zopár čísel. Detailný

dvojrozmerný plán podlahy, ktorý je taktiež častou formou reprezentácie máp v mobilnej robotike, často vyžaduje popis pomocou tisícok čísel. Avšak detailná 3D vizuálna mapa budovy (alebo dna oceánu) môže požadovať až milióny čísel. Zo štatistického hľadiska každé takéto číslo, či už sa jedná o polohu orientačného bodu v mape alebo polohu robota v mape samotnej, je potrebné určovať s určitou pravdepodobnosťou, problém teda môže byť extrémne viacrozmerný.

9.3.3 Problém korešpondencie

Je známy tiež ako *data association problem*. Problém korešpondencie je problém určenia, či senzorické merania uskutočnené v iných časoch korešpondujú s rovnakými fyzickými objektmi v prostredí. Vhodným príkladom je **problém uzavretia slučky** (closing loop problem). Keď robot prejde cez cyklické prostredie a vráti sa do miesta, v ktorom štartoval, môže byť nahromadená chyba jeho odometrie tak významná, že nebude možné získať informáciu, že prostredie korešponduje s predtým navštíveným prostredím. Tento problém je rozobratý aj v kapitole Vizuálna odometria, kde je nutné v procese mapovania nielen párovať susedné senzorické merania na základe korešpondencie príznakov, ale zároveň je potrebné aj určiť geometrickú transformáciu medzi týmito meraniami, a to práve na základe zmeny polohy príznakov v obraze.

9.3.4 Dynamické prostredia

Prostredia sa v čase menia. Niektoré zmeny môžu byť relatívne pomalé, ako napríklad zmena vzhľadu stromov v parku. Niektoré zmeny sú rýchlejšie, ako napríklad zmena stavu dverí alebo poloha nábytku. Najrýchlejšie zmeny v prostredí môžu byť napríklad ľudia alebo autá. Dynamika týchto prostredí vytvára veľké výzvy. Predstavme si, že robot stojí pred zavretými dverami, ktoré sú v mape mapované ako otvorené. Mohol sa zmeniť stav dverí, ale je možné, že zlyhala lokalizácia a robot stojí pred inými dverami, ako si myslí. Takmer žiadne algoritmy nevedia vytvárať zrozumiteľné mapy dynamických prostredí. Výskum v oblasti robotického mapovania sa sústreduje na statické prostredia, v ktorých iba robot je závislý na čase, čiže všetko, čo sa pohybuje, je len šum. Zároveň každé senzorické meranie trvá iba zlomok času.

9.3.5 Problém skúmania prostredia

Počas mapovania si robot musí vyberať cestu, ktorou sa bude pohybovať. Tento problém sa nazýva *robotic exploration* problem. Pohyb robota je ľahko optimalizovateľný z viacerých hľadísk za predpokladu, že máme k dispozícii mapu prostredia. Prieskumný robot však disponuje iba čiastočným modelom, takže je vystaveným novým problémom a rizikám. Tento problém je veľmi dôležitý, ale často sa rieši sub-optimálne pomocou jednoduchej heuristiky. Pri výbere, kam sa pohnúť, záleží na veľa faktoroch - koľko času a energie bude pohyb stáť, aký je predpokladaný informačný nárast po takomto pohybe, aké je nebezpečenstvo ovplyvnenia lokalizačného systému a ďalšie.

9.4 Zhrnutie

Po tridsiatich rokoch vývoja mobilného mapovania je situácia povzbudzujúca. Roboty sú schopné vytvárať mapy prostredí rôznej štruktúry, rôzneho rozsahu a s rôznou informačnou hodnotou zo senzorických meraní robotov a to najmä vo vnútornom prostredí. Napriek tomu však robotické mapovanie čelí veľkému množstvu iných problémov. Veľkou výzvou zostáva mapovanie veľkých a dynamických prostredí. Veľké prostredia a ich spracovanie so sebou prinášajú extrémne veľké nároky na úložisko, operačnú pamäť aj výpočtový výkon robota. V dynamických prostrediach je dôležité napríklad nielen pozorovať štrukturálne zmeny prostredia, ale aj chápať jeho dynamiku, napríklad predikovať pohyb jednotlivých dynamických prekážok.

Veľkou výzvou je aj implementácia chápania prostredia tak, ako ho chápu ľudia. Ľudia disponujú veľkým množstvom informácií o objektoch v prostredí, kde sa nachádzajú - dokážu ich identifikovať, klasifikovať, analyzovať. Podobné postupy boli zatiaľ v mobilnej robotike aplikované najčastejšie na plochy v geometrickom zobrazení merania - či už za účelom detektie príznakov, alebo za účelom zredukovania geometrickej zložitosti mapy. Navyše, človek má niekedy k dispozícii aj iné informácie - satelitné snímky prostredia, GPS súradnice, architektonické návrhy budov. Je dôležité navrhnúť koncepty, ako tieto informácie integrovať do mapovacích algoritmov.

Pre mapovanie vo vonkajšom prostredí zostáva veľa otázok stále otvorených. Problémom môže byť štruktúrovanosť prostredia - vnútorné prostredia majú vysokú mieru geometrickej štruktúrovanosti, avšak vo vonkajšom to tak nemusí byť. Dno oceánu alebo

povrch Marsu má tiež štruktúru, ale v omnoho vyššej typovej rozmanitosti. Opačným extrémom môže byť miesto zemetrasenia s kopami trosiek naokolo - prostredie je extrémne geometricky štruktúrované, ale možnosť rozpoznania štruktúry rozmiestnenia konkrétnych objektov je veľmi nízka.

Metódy rozobraté v tomto texte neriešia vzťah mapovania a ovládania robota. V užšom kontexte môžeme tento vzťah chápať tak, že je pri aktívnom mapovaní je nutné implementovať algoritmus preskúmavania prostredia. V širšom kontexte to môže znamenať problém určenia - čo treba v mape modelovať na to, aby mohol robot vykonávať svoje úlohy.

[60] [79] [26]

10 Experimentálna platforma

Vytvorené riešenie vizuálnej odometrie a mapovania bolo vyvíjané a experimentálne overované na robote TurtleBot 2. **TurtleBot** je svetovo najpopulárnejšia ROS platforma pre vzdelávanie a výskum využívajúca softvér s otvoreným zdrojovým kódom. Vznikla v roku 2010 v spoločnosti Willow Garage a vďaka podpore ROS sa rýchlo rozšírila do celého sveta. Na trh boli doposiaľ uvedené 3 generácie robota TurtleBot a ich viaceré modifikácie [80].



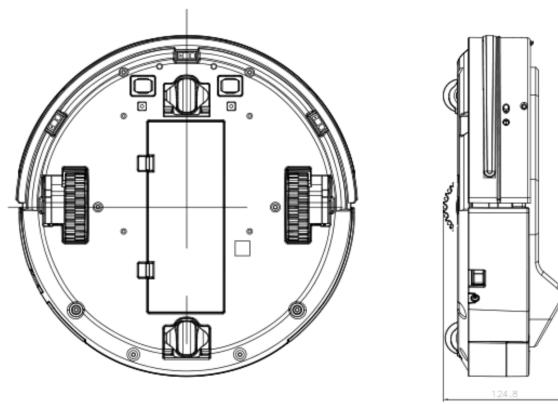
Obrázok 35 - Vytvorený mobilný robot TurtleBot 2. Vľavo robot v skupine robotov rovnakého typu, na ktorý je pre použitie treba umiestniť notebook. Vpravo robot so zvýšeným umiestnením snímača Kinect a osadeným pc v Mini-ITX formáte. Napájanie osadeného PC je zabezpečené samostatnou batériou.

Použitý robot TurtleBot 2 sa skladá z robotického podvozku Kobuki, snímača Kinect for Windows a bežného notebooku alebo inej výpočtovej jednotky. Na podvozku je upevnená nosná konštrukcia a hĺbková kamera, na nosnej konštrukcii je umiestnený notebook. Snímač Kinect aj podvozok Kobuki sa ku notebooku pripájajú pomocou zbernice USB 2.0. Potrebná kabeláž bola vyrobená podľa dokumentácie z bežne dostupných elektronických

komponentov. Snímač Kinect je umiestnený v zadnej časti robota na zvýšenej platforme, čo mu umožňuje pokryť svojim zorným poľom väčšiu časť prostredia.

10.1 Robotický podvozok Kobuki

Robotický podvozok Kobuki je diferenciálny podvozok s dvoma hnanými kolesami na stranách a dvoma opornými kolesami umiestnenými v prednej a zadnej časti robota. Hnané kolesá sú vybavené magnetickými enkodérmi s rozlíšením 52 impulzov na otáčku. Podvozok je vybavený aj gyroskopom, ktorý sme využili pri implementácii gyro odometrie. Gyro odometria sa ukázala ako výrazne presnejšia, než výpočet natočenia pomocou údajov z enkodérov. Robot Kobuki je zamýšľaný ako vývojová platforma, preto priamo poskytuje pre ostatné zariadenia napäťový výstup 12V 2A, pomocou ktorého sa napája snímač Kinect.

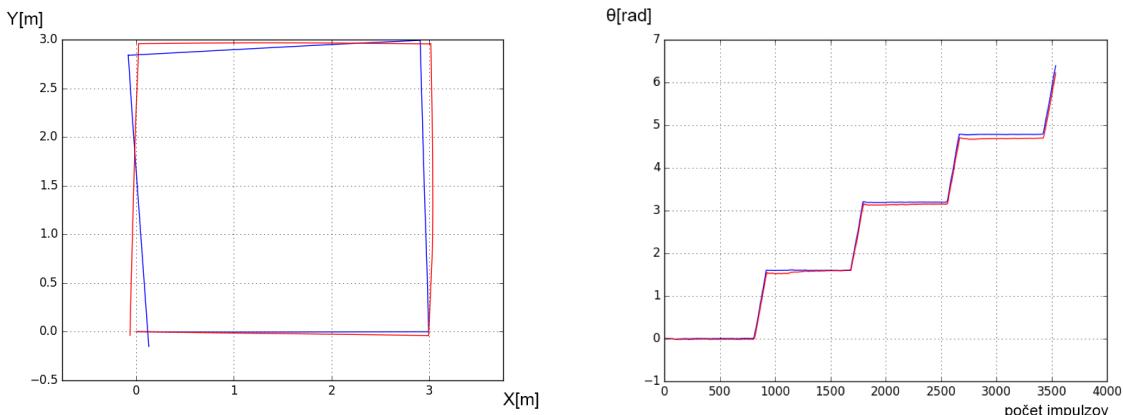


Obrázok 36 - rozloženie hnaných a oporných kolies na robotickom podvozku Kobuki [81].

10.2 Riadenie pohybu robota

Pre riadenie pohybu robota bol implementovaný vlastný ovládač v programovacom jazyku C++. Implementované boli pohybové príkazy pre pohyb po priamke dopredu, dozadu a rotáciu v smere a proti smeru hodinových ručičiek. Pre riadenie oboch pohybov sa využíva **P** regulátor, riadenou veličinou je poloha robota. Pre plynulejší rozbeh je priebeh rýchlosťi do momentu dosiahnutia maxima nahradený rampou. Implementované riadenie bolo experimentálne overené a poskytuje mierne presnejšie polohovanie, ako softvér voľne dostupný k tomuto robotickému podvozku. Parametre ovplyvňujúce priebeh rýchlosťi robota

pri translácii alebo rotácií sú závislé na zaťažení robota, rozložení hmotnosti neseného bremena a druhu povrchu, po ktorom sa robot pohybuje. Pre dosiahnutie optimálnych výsledkov je preto vhodné často parametre regulácie upraviť manuálne. Určenie natočenia podvozku je možné určiť z odometrie alebo je možné ho odčítať z gyroskopu osadeného v podvozku Kobuki. Pri experimentálnom overovaní sa potvrdilo, že použitie gyroskopu poskytuje výrazne vyššiu presnosť, a preto všetky implementované pohybové príkazy používajú práve gyroskop. Robotický podvozok umožňuje translačný pohyb rýchlosťou až 700 mm/s, avšak už od rýchlosťi 400 mm/s sa riadenie robota stáva veľmi nepresným z dôvodu prešmyku kolies. To isté platí aj v prípade rotačného pohybu. Z tohto dôvodu bola maximálna rýchlosť mobilnej platformy obmedzená na 400 [mm/s] pre transláciu a 0.5 [π/s] pre rotáciu.

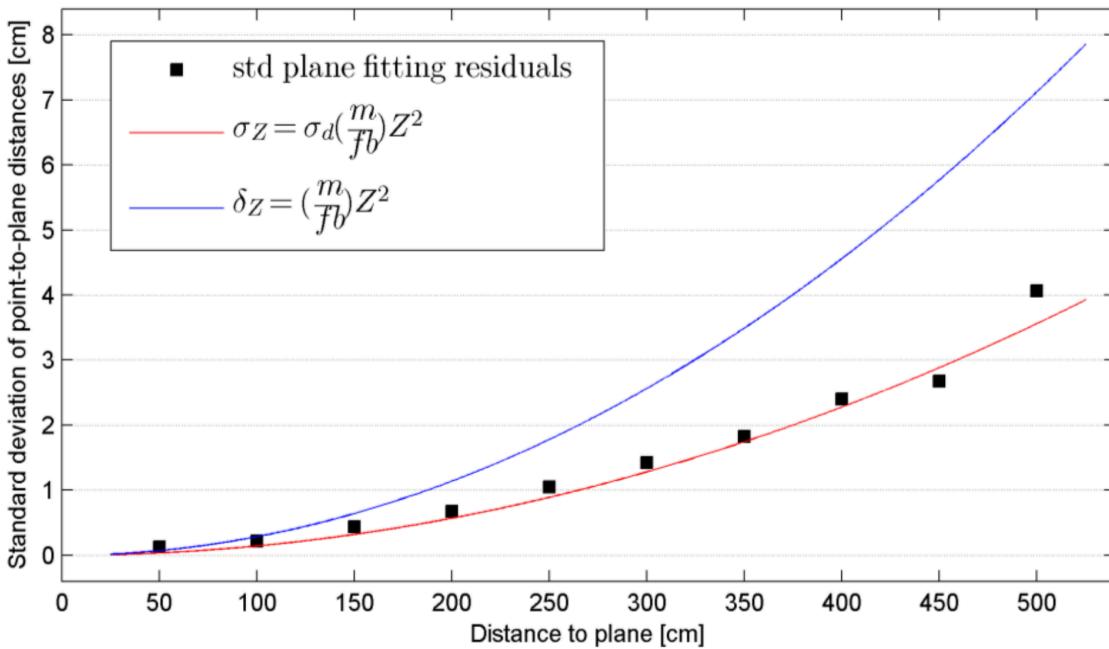


Obrázok 37 - Porovnanie polohovania pri výpočte natočenia z enkodérov (modrá farba) a z gyroskopu (červená farba). Robot bol vyslaný z počiatku súradnicovej sústavy po štvorcovej trajektórii s dĺžkou hrany 3m. Výsledná polohová odchýlka po vykonaní pohybu bola v prípade enkodérov 150mm, v prípade riadenia pomocou gyroskopu 37mm. Kumulovaná chyba natočenia bola v prípade enkodérov dvojnásobná oproti gyrokopu.

10.3 Snímač Kinect

Snímač Kinect bol prvou masovo dostupnou hĺbkovou kamerou, ktorú priniesla na trh spoločnosť Microsoft v novembri roku 2010 ako uzavretú perifériu pre hračku konzolu Xbox 360. Jej predaj bol ukončený v októbri 2017. Vývojárska komunita v krátkom čase po uvedení na trh vyvinula voľne dostupné ovládače pre platformu Windows, Mac OS X a Linux, a tak sprístupnila toto zariadenie vedcom, umelcom a nadšencom po celom svete. V súčasnej dobe je na trhu dostupné veľké množstvo zariadení, ktoré poskytujú kvalitnejší výstup, nižšiu spotrebu energie a dobrú softvérovú vybavenosť ako napríklad snímač Structure [82],

Microsoft Kinect vo verzii 2 a zariadenia postavené na platforme Intel® RealSense™ Technology [83]. Snímač Kinect pracuje na princípe projekcie štruktúrovaného svetla. Presnosť hĺbkových dát prichádzajúcich z tohto typu RGBD kamier klesá so vzdialovaním objektov od kamery. Závislosť štandardnej odchýlky hĺbkových dát od vzdialenosť pre snímač Kinect je zobrazená na Obrázok 38.

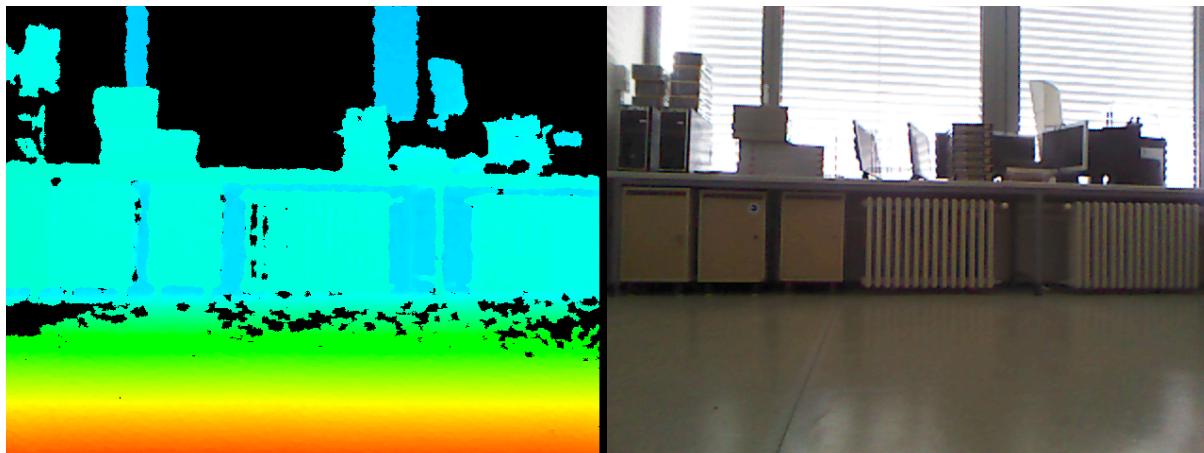


Obrázok 38 - Závislosť hĺbkového rozlíšenia snímača Microsoft Kinect od vzdialenosť. Teoretická náhodná chyba je zobrazená červenou, teoretické rozlíšenie modrou [84]

Snímač Kinect ponúka snímkovaci frekvenciu približne 30Hz, rozlíšenie 640x480 pixelov. Vzdialenosť, na ktorú snímač spoľahlivo funguje je 0,7 – 6m, avšak po použití neoficiálnych ovládačov je táto vzdialenosť väčšia (maximálna nameraná hodnota bola 9757mm). Ovládače pre snímač sú dostupné pod licenciou open source v projektoch OpenNI (zaniknutý) [85] a libfreenect [86]. Pre platformu Windows je k dispozícii oficiálny ovládač aj SDK (Software Development Kit). RGB kamera snímača má relatívne nízku kvalitu a hĺbkový obraz trpí viacerými charaktermi šumu. Pre získanie snímkov v čo najvyššej kvalite je zároveň potrebné snímač presne kalibrovať. Sú známe postupy pre kalibráciu: radiálneho a tangenciálneho kreslenia, rotačnej chyby uloženia snímača na obrazovú rovinu, vnútorné aj vonkajšie parametre oboch kamier a vzájomnú registráciu pixelových mriežok oboch kamier. Zároveň je známych aj viacero metód potlačenia a filtrácie šumu. Snímač Kinect trpí aj viacerými

zdrojmi chýb zapríčinených samotnou technológiou hĺbkového snímania scény, ako napríklad zmiešané pixely, hĺbkový šum na hranách a šum závislý na orientácii snímaného povrchu.

Obrazový výstup zo snímača Kinect v jednom cykle vyzerá nasledovne:



Obrázok 39 - Vľavo hĺbkový obraz, vpravo RGB obraz zo snímača Kinect

Snímač poskytuje hĺbkové dátá aj v úplnom zatienení, keďže sa jedná o aktívnu technológiu snímania. V tomto prípade však nebude k dispozícii obraz z RGB kamery. Použité zariadenie prešlo iba základnou kalibráciou a registráciou snímok. Je dôležité tak isto poznamenať, že RGB kamera snímača Kinect má veľmi malý dynamický rozsah a snímky z neho prichádzajúce často obsahujú presvietené alebo naopak veľmi tmavé oblasti.



Obrázok 40 - Snímač Microsoft Kinect po odstránení plastového krytu.

10.4 Záznam dát na experimentálnej platforme

Experimentálna platforma je schopná záznamu informácií z podvozku robota do textového súboru. Informácie prichádzajúce z kolesovej odometrie sú spoločne s informáciami prichádzajúcimi z gyroskopu využité pre globálne určenie polohy a orientácie robota. Poloha robota P v aktuálnom časovom kroku t je určená pomocou stavového vektora:

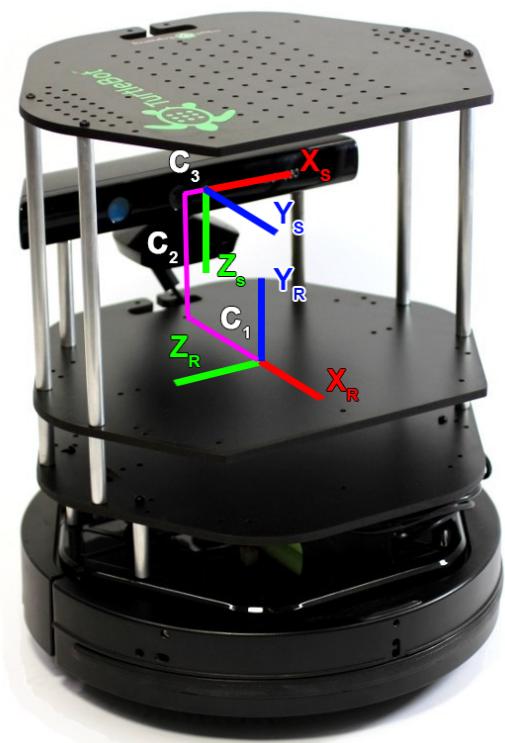
$$P_t = [x, y, \theta]$$

Kde x je súradnica x-ovej osi robota v globálnom súradnicovom priestore, y je súradnica v osi Y a θ predstavuje uhol natočenia robota.

Farebné aj hĺbkové snímky zo snímača Kinect sa získavajú pomocou ovládača libfreenect a zapisujú sa počas záznamu na disk v komprimovanom formáte Portable Network Graphics (PNG). Zápis do tohto formátu sice zaťažuje výpočtovú jednotku procesora pre kompresiu dát, ale zároveň dramaticky znižuje nároky na úložný priestor.

Uložené sekvencie dát je možné prehrávať a tak porovnávať viaceré metódy spracovania za rovnakých podmienok. Pre zrýchlenie zápisu snímkov na pevný disk bol počas experimentov využitý zápis na úložisko umiestnené v operačnej pamäti počítača, tzv. RAMdisk. Sekvencie je možné vo vytvorenej aplikácii prehrávať so simulovaním rôznej snímkovacej frekvencie RGBD kamery.

Kedže snímač Kinect má počiatok súradnicovej sústavy svojich snímkov orientovaný inak, ako je orientovaný súradnicový priestor robotického podvozku Kobuki, je pre porovnávanie presnosti polohovania nutné tieto dva systémy transformovať do jedného systému. Zároveň, počiatok súradnicového systému snímača Kinect je umiestnený odlišne, ako počiatok súradného systému podvozku Kobuki a tak je nutné určiť aj posuny v jednotlivých osiach. Rotačná chyba uloženia snímača bola zanedbaná.



Obrázok 41 - Robotická platforma Turtlebot 2. Súradnicový systém snímača Kinect je anotovaný ako X_s, Y_s, Z_s . Súradnicový systém podvozku Kobuki je anotovaný pomocou X_R, Y_R, Z_R . Translácie v jednotlivých osiach sú označené ako C_1, C_2, C_3 ružovou farbou. Ich hodnoty sú dostupné v dokumentácii ROS balíka ku robotu Turtlebot 2. Ich výpočet sa realizuje pomocou riešenia kalibrácie Ruka-Oko (Hand-Eye calibration). Matematicky je problém popisovaný a riešený ako problém $AX = XB$ [87].

11 Vizuálna odometria a mapovanie pre malý mobilný robot

Hlavným cieľom tejto dizertačnej práce je implementovať a experimentálne overiť systém vizuálnej odometrie a mapovania pre malý mobilný robot. Pod týmto označením rozumieme v kontexte práce robot, ktorý disponuje obmedzeným výpočtovým výkonom. V súčasnej dobe môže túto platformu dostupným výpočtovým výkonom predstavovať aj bežný smartfón. Kedže predpokladáme, že vytvorený systém bude v budúcnosti slúžiť na účely ďalšieho výskumu a výuky robotického mapovania na pracovisku, vytvorené riešenie by malo byť ľahko replikovateľné, rozšíriteľné a malo by byť možné ho využívať na bežne dostupných konfiguráciách osobných počítačov. V tejto kapitole sú bližšie špecifikované požiadavky na tento systém a diskutované dizajnové rozhodnutia, osvojené pri návrhu návrhu nášho riešenia.

Výkon procesora – v súčasnej dobe obsahujú výpočtové jednotky počítačov spravidla viaceru jadier, teda disponujú hardvérovými možnosťami pre paralelné spracovanie inštrukcií a simultánny beh viacerých aplikácií. Utilizácia viacerých výpočtových jednotiek pomocou paraleлизácie spúštaných procesov je všeobecne implementovaná za účelom zrýchlenia výpočtu. Paraleлизáciu je možné implementovať na hlavnej výpočtovej jednotke-procesore, alebo na grafickej karte (napríklad pomocou Nvidia CUDA, ktoré využíva techniku Kinect Fusion). Predpokladáme, že aj malý mobilný robot bude schopný paraleлизácie na hlavnej výpočtovej jednotke, ktorá bude implementovaná aj v našom riešení.

Grafická karta – v súčasnej dobe disponuje veľké množstvo mobilných platform aj integrovanými grafickými kartami, ktoré však poskytujú len obmedzený výkon a veľmi špecifickú aplikačnú podporu. Z tohto dôvodu nebude grafická karta využitá v našom riešení.

Pamäťová náročnosť – vytvorené riešenie by malo využívať dostupnú operačnú pamäť úsporne. Zároveň nemôže predpokladať, že jej bude k dispozícii viac ako niekoľko GB.

Dostupné úložisko – nie je možné očakávať, že malý systém bude disponovať veľkým úložiskom. Vytvorená mapa prostredia by mala byť preto po svojom zápisе na disk čo najmenšia.

Operačný systém – Pre znovupoužiteľnosť aplikácie je dôležité, aby zvolený operačný systém bol široko používaný v aktuálnych robotických aplikáciách. Z tohto dôvodu bola ako podporovaná platforma vybraná platforma Linux, podobne ako podporuje Robotický operačný systém, ROS [88]. Ak to však bude možné, riešenie by malo byť schopné spustenia aj na počítačoch s operačným systémom Windows a OS X. Robotický operačný systém sme sa rozhodli pri začatí prác na implementácii nevyužiť, aplikácia by mala byť schopná samostatného behu len pomocou operačného systému a dostupného hardvéru. Robotický operačný systém zároveň podporuje iba operačný systém Linux.

Programovací jazyk – Tradične využívanými programovacími jazykmi v počítačovom spracovaní obrazu sú jazyky C++ a Python. Python, spolu s prostredím Matlab predstavujú tradičnú programovú podporu pre vývoj prototypov riešení, avšak pre vysokorýchlosné spracovanie často neposkytujú dostatočný výkon. Preto sú spravidla všetky najpopulárnejšie riešenia vizuálnej odometrie a vizuálneho SLAM implementované v C++. Tento kompilovaný a typový programovací jazyk poskytuje vysoký výkon, možnosti objektového, funkcionálneho aj generického programovania a širokú programovú podporu pre vytváranie všetkých typov aplikácií. Zároveň, je dostatočne populárny v komunite výskumníkov a takmer každé vysoko výkonné riešenie je v ňom primárne implementované. Pre C++ sú dostupné knižnice výborné knižnice pre počítačové spracovanie obrazu Point Cloud Library (PCL) a OpenCV. Tieto knižnice podporujú všetky 3 vyššie zmieňované operačné systémy.

Frekvencia spracovania – algoritmus by mal byť schopný, pokiaľ je to možné, spracovať prichádzajúce obrazové dátá v reálnom čase. Naše riešenie je dimenzované na snímač MS Kinect, ktorý má snímkovaci frekvenciu približne 30Hz. Snímač s nižšou snímacou frekvenciou sa pre zabezpečenie dostatočného prekryvu prichádzajúcich snímok bude musieť pohybovať iba pomalšie.

RGBD kamera – riešenie by malo byť schopné pracovať s RGBD kamerou bežne dostupnou na trhu, ktorú je možné ľahko integrovať na mobilnú robotickú platformu. Zariadenie MS Kinect reprezentuje práve takýto snímač a v mnohých projektoch bol úspešne umiestnený na mobilnú robotickú platformu za účelom implementácie robotického mapovania.

Snímané prostredie – väčšina na trhu dostupných RGBD kamier nie je určená pre vonkajšie použitie. Preto budeme riešenie navrhovať iba pre vnútorné prostredie.

Osvetlenie snímaného prostredia – algoritmus by mal fungovať za podmienok umožňujúcich plnohodnotné snímanie použitým snímačom. Použitie snímača Kinect vyžaduje dostatočné osvetlenie scény pre snímanie farebného obrazu. Zároveň je snímač nevhodný pre použitie vonku, preto bude navrhované riešenie určené pre vnútorné použitie.

Vlastnosti vytvorenej mapy – chyba určenia pozície medzi dvoma snímkami sa bude akumulovať v nepresnosti uloženej mapy. Tento efekt by malo byť možné dodatočne kompenzovať a mapu korigovať, mala by teda byť ľahko modifikovateľná v každej svojej časti. Mapa by mala umožňovať dodatočnú implementáciu lokálnej aj globálnej optimalizácie grafu a problému uzavretia slučky. Vo vytvorenej mape by malo byť možné implementovať a experimentálne overiť možnosti navigácie a plánovania pohybu. Predpokladáme, že mapa bude použitá na plánovanie pohybu a absolútну lokalizáciu a mobilný robot bude využívať pre krátke pohyby reaktívnu navigáciu. V absolútном vyjadrení by nepresnosť určenia pozície robota v lokálnej mape bola maximálne **10 cm**. V priestorovom vyjadrení môžeme teda použiť voxel s hranou 10 cm.

11.1 Prehľad predchádzajúcej práce, dostupné implementácie

V našej predchádzajúcej práci boli identifikované [89] a analyzované [90] dostupné riešenia pre tvorbu 3D mapy prostredia vo formáte meshu pomocou snímača Kinect. Analyzované boli dve implementácie – **RGBD-6D-SLAM** [30] a **KinectFusion** [31]. V tejto kapitole budú obe implementácie predstavené a zhodnotené z hľadiska možností použitia pre robotické mapovanie.

Riešenie **RGBD-6D-SLAM** implementuje systém vizuálnej odometrie postavenej na príznakoch z 2D obrazu (feature-based SLAM). Typ použitého detektora je SURF [41]. Pre zabezpečenie robustnosti je použitá metóda RANSAC. Transformácia medzi snímkami je ďalej korigovaná pomocou metódy ICP [52] [53]. Poloha kamery je lokálne optimalizovaná pomocou algoritmu HOG-man [56]. Výstupný formát mapy je mračno bodov. **Riešenie poskytuje veľmi nízku rýchlosť spracovania 2 s na jeden snímok a vytvorená mapa vo formáte mračna bodov nie je vhodná pre riešenie úloh mobilnej robotiky.**

Kinect fusion algoritmus je metóda 3D povrchovej rekonštrukcie na základe RGBD dát z kamery Microsoft Kinect. Umožňuje konzistentné mapovanie vo vysokej rýchlosťi iba na základe hĺbkových dát, takže funguje aj bez viditeľného svetla. Systém v reálnom čase sleduje

polohu kamery v šiestich stupňoch voľnosti a integruje nové merania do jedného globálneho 3D modelu pomocou kízavého priemeru. Rekonštrukcia sa stáva detailnejšou s vyšším počtom spracovaných snímok danej oblasti. Základným stavebným prvkom Kinect Fusion algoritmu je **Truncated signed distance function (TSFD)** [91]. TSDF je reprezentácia priestoru v ktorej každý element uchováva informáciu o jeho vzdialosti ku najbližšiemu povrchu. Integrácia dát z jednotlivých snímok prebieha nasledovne:

1. Vypočítame povrchovú a normálovú mapu híbkových dát. Pre redukciu šumu algoritmus aplikuje na híbkové snímky **bilaterálny filter (bilateral filter)**.
2. Z aktuálneho modelu mapy vytvoríme pomocou techniky raycasting [32] referenčný snímok, voči ktorému určíme polohu aktuálneho snímku pomocou ICP [53].
3. Za použitia vypočítanej polohy aktuálneho snímku sa aktuálne híbkové dáta integrujú do TSFD pomocou priemerovania (running average).

TSFD je maticová priestorová reprezentácia, ktorá dokáže uchovávať všetky híbkové merania a integrovať ich do jednej povrchovej reprezentácie. Zároveň v nich dokáže uzatvárať diery a dokáže sa aktualizovať inkrementačne, bez ohľadu na poradie [91]. Výsledky algoritmu Kinect Fusion boli veľmi presvedčivé už v jeho prvej implementácii. Pre účely robotického mapovania sa však táto metóda nepoužíva najmä z dôvodov:

- Metóda dosahuje spracovanie v reálnom čase najmä vďaka využitiu masívneho paralelizmu pomocou GPU. Prakticky každý krok algoritmu vykonáva operácie nad množinou dát paralelne.
- TSFD je limitovaná veľkosťou. Algoritmus Kinect Fusion na začiatku mapovania vyhradí miesto v pamäti pre TSFD, kde potom integruje senzorické merania. V prvej implementácii mapovanie začína vždy uprostred priestoru tvaru kocky o hrane veľkosti cca 3,5 metra. Veľkosť spracovávanej mapy prostredia je teda limitovaná veľkosťou voľnej pamäte robotického systému. Aj keď boli predstavené metódy, ktoré riešia tento problém [92] [93], malé mobilné robotické systémy si zväčša nemôžu dovoliť spracovávať takto pamäťovo náročné operácie.

Po overení dostupných riešení vizuálnej odometrie a získaní máp vo formáte meshu bola navrhnutá vlastná implementácia mapy vo formáte oktálového stromu. Mapy vo formáte oktálového stromu boli potom využité pre riešenie problému navigácie mobilného robota pomocou metódy A* [72]. V práci bolo preukázané, že mapa vo formáte oktálového stromu predstavuje optimálny mapový formát pre návrh mapovacieho algoritmu pre malý mobilný robot. Výhodné sú najmä nízke nároky na úložisko a vysoká rýchlosť pridania nového snímku do už existujúcej mapy. Pri rozlíšení 9 cm bola veľkosť súboru mapy zredukovaná o 99,99%. Na tejto mape bolo vykonané aj experimentálne overenie rýchlosťi plánovania globálnej trajektórie pomocou algoritmu A*, ktoré potvrdilo, že v tejto mape je možné časovo efektívne plánovať pohyb mobilného robota [72].

11.2 Experimentálne datasety

Pre efektívny návrh riešenia vizuálnej lokalizácie a mapovania je výhodné začať zberom datasetov z prostredí, kde má byť vytvorené riešenie využívané. Takto nazbierané dátá umožnia priebežné ladenie algoritmov na reálnych vstupoch. Všetky sekvencie boli zozbierané vo vnútornom prostredí. Svetelné podmienky boli premenlivé, v niektorých miestach prevládalo vonkajšie osvetlenie prichádzajúce z okien, niektoré oblasti boli zosnímané len za použitia umelého osvetlenia. Snímané prostredie predstavovalo bežné vnútorné prostredie na pracovisku – kanceláriu, chodby, laboratóriá. Trajektórie, po ktorých sa robot pohyboval mali tvar kriviek aj úsečiek a obsahovali aj oblasti, na ktorých pohyb spočíval len v rotácii robotického podvozku.

Ako kontrolné datasety boli vybrané 3 datasety korešpondujúce s elementárnymi zložkami pohybu v každej trajektórii:

- **Statický dataset** – v tomto datasete sa robot nepohyboval, snímač Kinect snímal statickú, dobre štruktúrovanú scénu po dobu dvoch minút. Táto doba sa ukázala ako dostatočná pre zber reprezentatívnej vzorky dát po tom, čo boli otestované datasety o dĺžke 5, 10 aj 20 minút.
- **Rotačný dataset** – pohyb robota spočíval v rotácii v smere hodinových ručičiek 5x okolo jeho vertikálnej osi.

- **Translačný dataset** – v tomto datasete sa robot pohyboval 10x dopredu a 10x dozadu o vzdialenosť 30 cm.

Všetky kontrolné datasety boli zozbierané v primerane vizuálne štruktúrovanej scéne (obývacia izba), kde sa hĺbkové merania jednotlivých pixelov nachádzali v celom meracom rozsahu.

Dataset bol uložený ako sekvencia RGB a hĺbkových snímok a textový záznam z odometrie robotického podvozku Kobuki.

11.3 Metodika vyhodnotenia

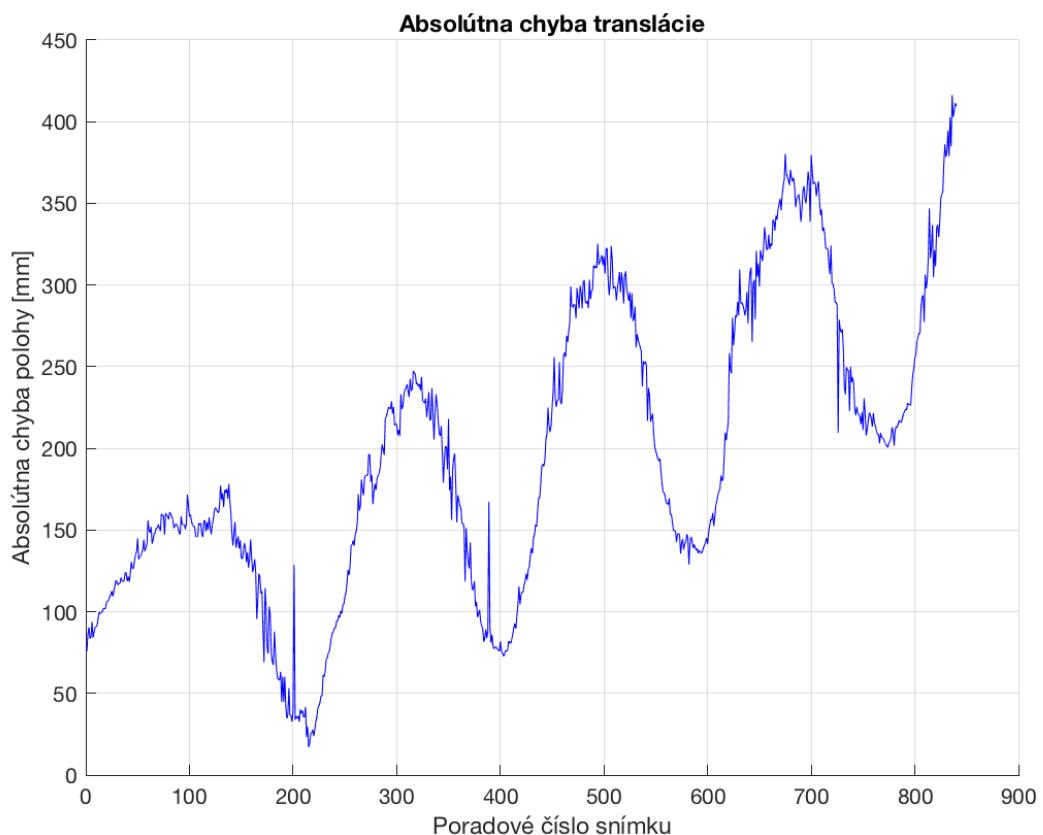
Experimenty prebiehali v offline režime, kde snímky z datasetu boli prehrávané ako snímky prichádzajúce z reálneho zariadenia Kinect. Ku páru snímok prislúchal jeden riadok v textovom súbore odometrie. Každý experiment bol vykonaný na rovnakej hardvérovej konfigurácii a to MacBook Pro Mid 2014, Intel Core i5 2,8 GHz, 8 GB RAM za súčasného používania operačného systému inými úlohami. Výrazné zrýchlenie prinášalo spúšťanie algoritmu hned po štarte systému a nespúšťanie žiadnych iných aplikácií. Tento postup by bolo možné použiť ako referenčný v prípade, že by robot počas svojej prevádzky exkluzívne riešil iba vizuálnu odometriu a mapovanie. V praxi však musí mobilný robot vykonávať aj množstvo iných úloh – navigáciu, detekciu dynamických prekážok atď. Preto bol uprednostnený reálny model, kde v operačnom systéme beží paralelne aj viacero iných úloh – aplikácií.

Počas experimentu sa zbierali a ukladali rôzne štatistické dátá o behu algoritmu, napríklad priemerné časy jednotlivých krokov alebo počty detegovaných príznakov, ktoré budú ďalej prezentované v prislúchajúcich častiach práce. Počas behu experimentu bola vykonávaná aj vizuálna inšpekcia kvality tvorby mapy. Pre kvalitatívnu analýzu kvality rekonštrukcie pohybu kamery bola ako referencia použitá informácia z odometrie robota. Po behu algoritmu vznikli 2 nové textové súbory, ktoré obsahovali sekvenciu matíc homogénnych transformácií, prislúchajúcich polohám robota a kamery v globálnych súradničiach. Tieto súbory boli ďalej spracované a vyhodnotené v programovom prostredí MATLAB.

Pri každom experimente prebehlo viacero vyhodnotení nazbieraných dát. **Cieľom bolo vybrať** metriku, ktorá **bude dostatočne dobre vyjadrovať úspešnosť nášho algoritmu určenia polohy voči gyro odometrii z robotického podvozku Kobuki.**

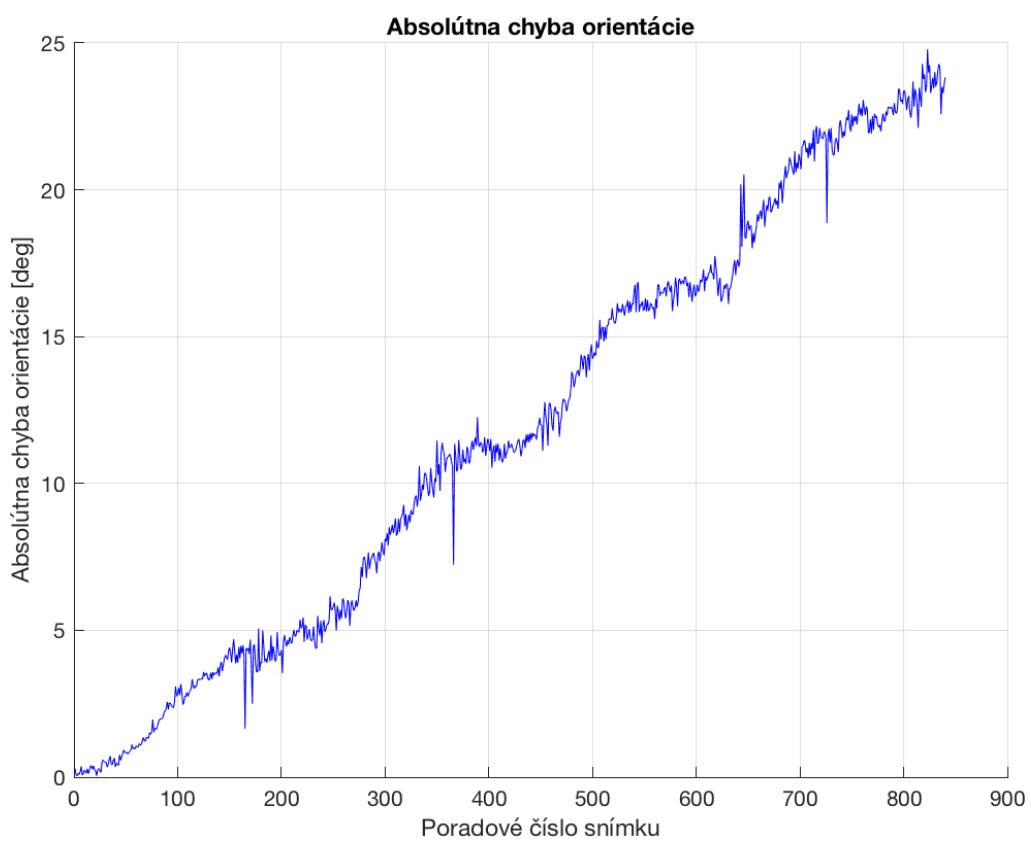
11.3.1 Absolútна chyba translácie a rotácie

Absolútna chyba určenia polohy je v našom prípade priestorový rozdiel - dĺžka vektora medzi počiatkom súradnicového systému robota a kamery v globálnych súradničiach. Kedže absolútna poloha robota aj kamery je zatažená kumulatívou chybou z predchádzajúcich iterácií algoritmu, jedná sa o metriku, ktorej relevantnosť s časom klesá. Aj tak sa však jedná o dobrý prostriedok pre sledovanie charakteru odlišnosti týchto dvoch pohybov a porovnanie nárastu globálnej chyby u viacerých variantov algoritmu. Napríklad, Obrázok 42 - absolútna chyba posunu kamery počas piatich rotácií robota pri rotačnom teste. zobrazuje absolútnu chybu posunu kamery pri rotačnom teste. Ako vidíme, jednotlivé iterácie majú približne rovnaký priebeh, ale celková chyba časom rastie. Výrazné skoky korešpondujú s miestami výskytu výraznej chyby určenia polohy.



Obrázok 42 - absolútna chyba posunu kamery počas piatich rotácií robota pri rotačnom teste.

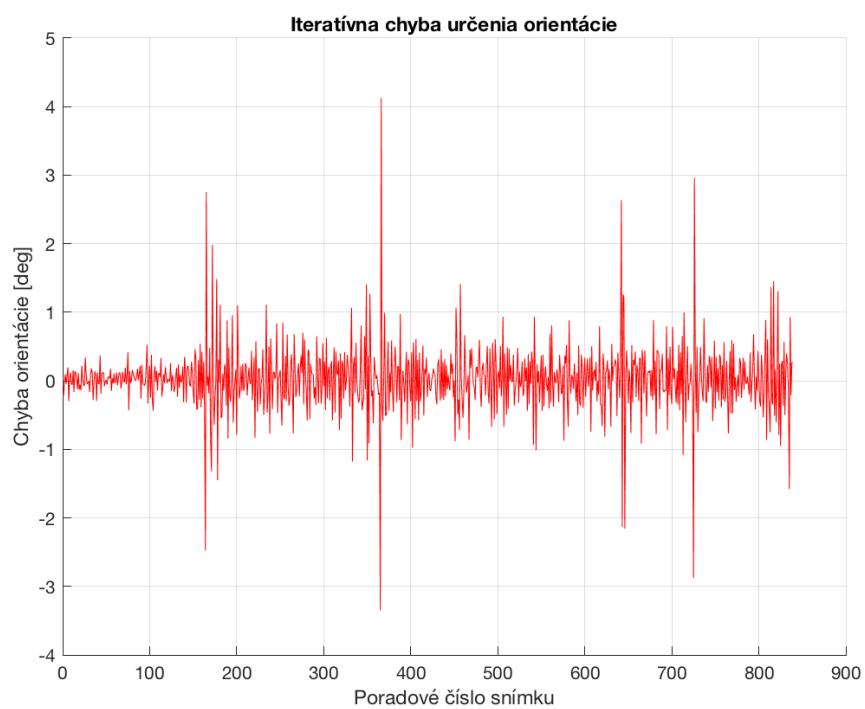
Analogicky ku translačnej chybe sme analyzovali aj chybu rotácie, teda určenia orientácie súradnicového systému kamery a robota. V princípe sa jedná o určenie miery odlišnosti dvoch rotačných matíc 3×3 , častí matíc homogénnych transformácií vyjadrujúcich pohyb. Pre analýzu priebehu takého pohybu pomocou grafu je nevhodné použiť eulerovské uhly, pretože sa v nich prejavuje gimbal lock, t.j. problém, ktorý spôsobí stotožnenie dvoch alebo všetkých troch rotačných osí a tým pádom aj stratu stupňov voľnosti. Preto sme zvolili metriku, pri ktorej sú obe matice aplikované na jednotkový vektor [1,1,1] a medzi výslednými vektormi je určený uhol v rovine preloženej týmito vektormi. Tento spôsob určenia rozdielu orientácie sa využíva 3D grafickej a robotickej komunity a je možné ho považovať za vhodný spôsob vyjadrenia danej metriky. Navyše, je primerane názorný na 2D grafe. Priebeh absolútnej chyby orientácie pri rotačnom teste zobrazuje Obrázok 43. Skoky na tomto grafe korešpondujú s miestami výskytu výraznej chyby.

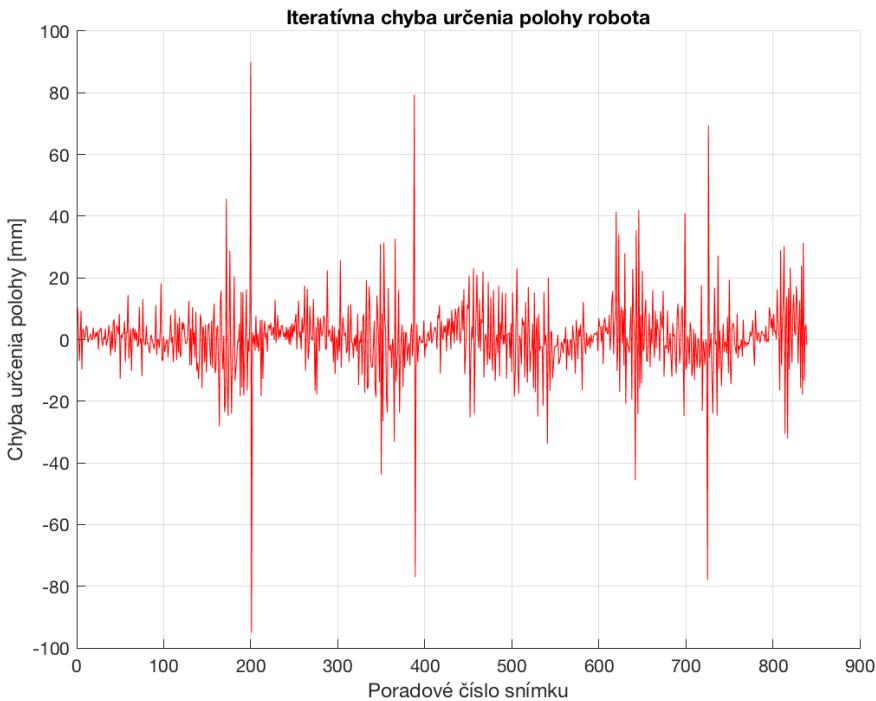


Obrázok 43 - Priebeh absolútnej chyby určenia orientácie.

11.3.2 Iteratívna chyba translácie a rotácie

Pre odstránenie vplyvu akumulovanej chyby z predchádzajúcich meraní je vhodnejšou metrikou vyhodnocovanie chyby iba v jednej iterácii algoritmu. Toto vyhodnotenie je možné implementovať pomocou porovnávania aktuálnej transformačnej matice v súradniach robota a aktuálnej transformačnej matice v súradniach kamery. Alternatívne, ak je k dispozícii polohu v globálnych súradniach, môžeme jednoducho od aktuálnej chyby určenia polohy aj orientácie odrátať jej predchádzajúcu hodnotu. Obrázok 44 zobrazuje priebeh iteratívnej chyby pri rotačnom teste. Sledujeme odstránenie vplyvu akumulovanej chyby predchádzajúcich meraní. Zároveň je jednoduchšie pozorovať výrazné skoky v hodnote chyby ako na grafe absolútnej chyby.





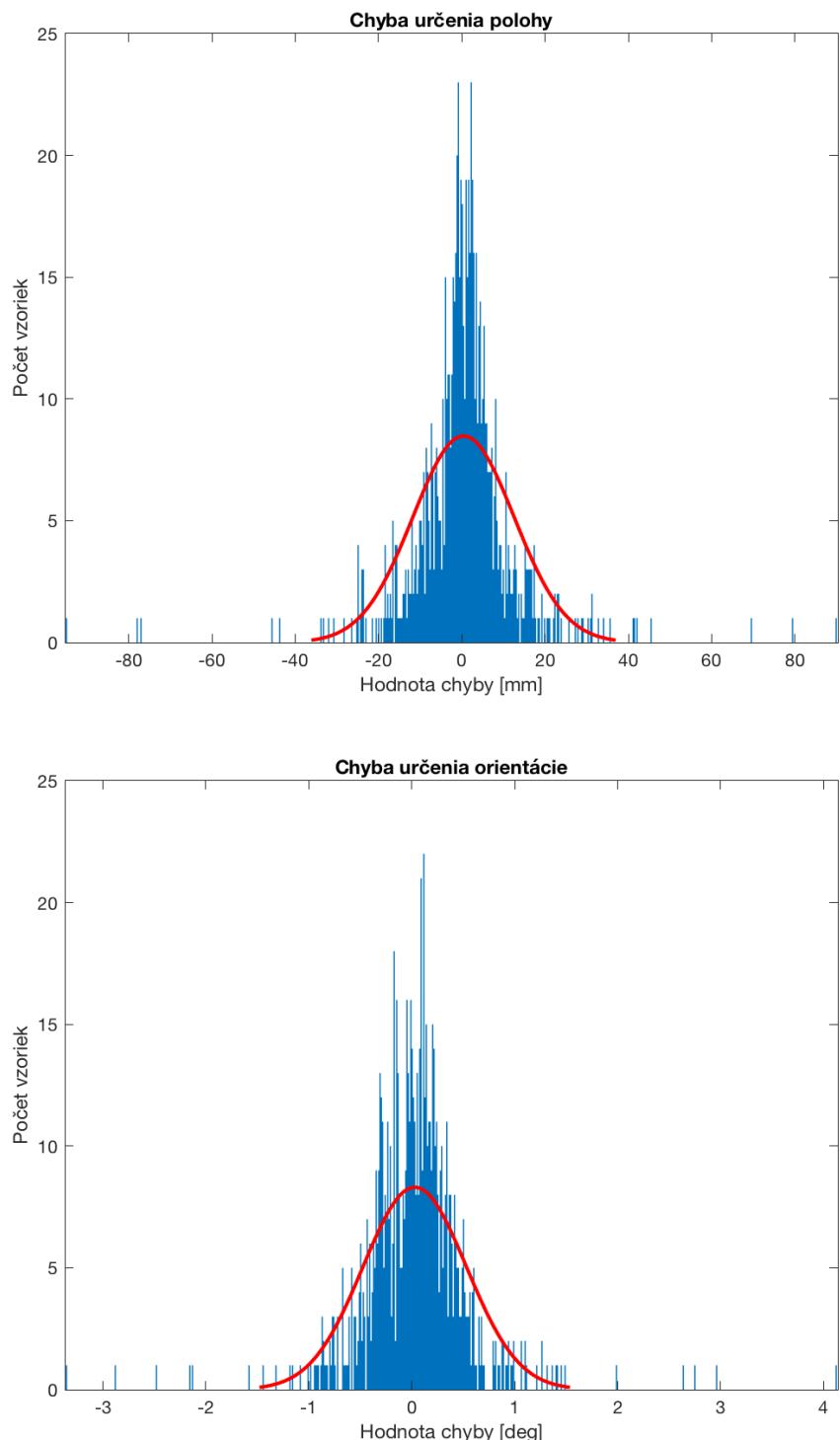
Obrázok 44 - Iteratívna chyba určenia polohy a orientácie robota pri rotácii 5x okolo svojej osi.

11.3.3 Kvalita rekonštrukcie pohybu kamery

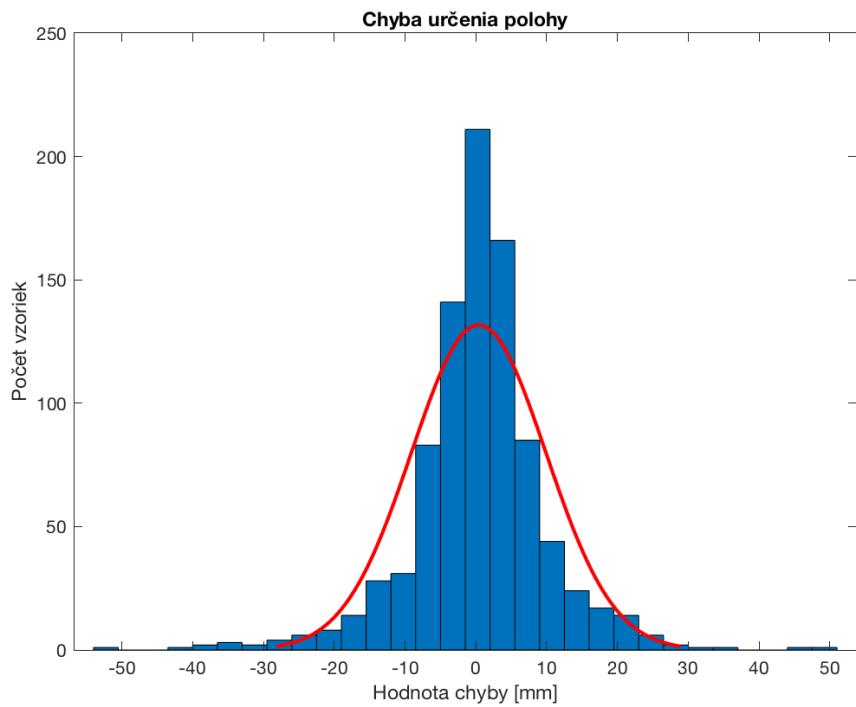
Iteratívna chyba predstavuje v každej iterácii štatisticky nezávislú vzorku. Pravdepodobnostné rozloženie týchto chýb bolo zobrazené pomocou histogramu a porovnané s normálnym rozložením. Ako je vidieť Obrázok 45, chybu je možné považovať za zhruba normálne rozloženú a je ju teda možné charakterizovať pomocou strednej hodnoty a štandardnej odchýlky. Práve tieto dva údaje boli použité pre vyhodnotenie kvality rekonštrukcie pohybu kamery pomocou nášho algoritmu.

Kvalita rekonštrukcie pohybu kamery však závisí hlavne od kvality snímaných príznakov. Scény, kde príznakov nie je dostatok, alebo sú príznaky detegované nespoľahlivo (napríklad snímky obsahujú výrazný motion blur alebo snímky obsahujú opakujúce sa oblasti) budú spracované s výraznou chybou určenia polohy aj orientácie. Z tohto dôvodu bola použitá metóda 3-sigma. To znamená, že z rozloženia chyby orientácie aj posunu kamery budú odstránené hodnoty väčšie ako trojnásobok štandardnej odchýlky, pretože sú považované za extrémne náhodné hodnoty. Vizuálnou kontrolou datasetov bolo potvrdené, že výrazné chyby sa nachádzajú práve v oblastiach s problematickými scénami s nedostatom príznakov alebo vznikajú pri príliš rýchлом pohybe robota. Po odstránení extrémnych hodnôt je znova určená štandardná odchýlka aj stredná hodnota. Tieto týmto spôsobom lepšie

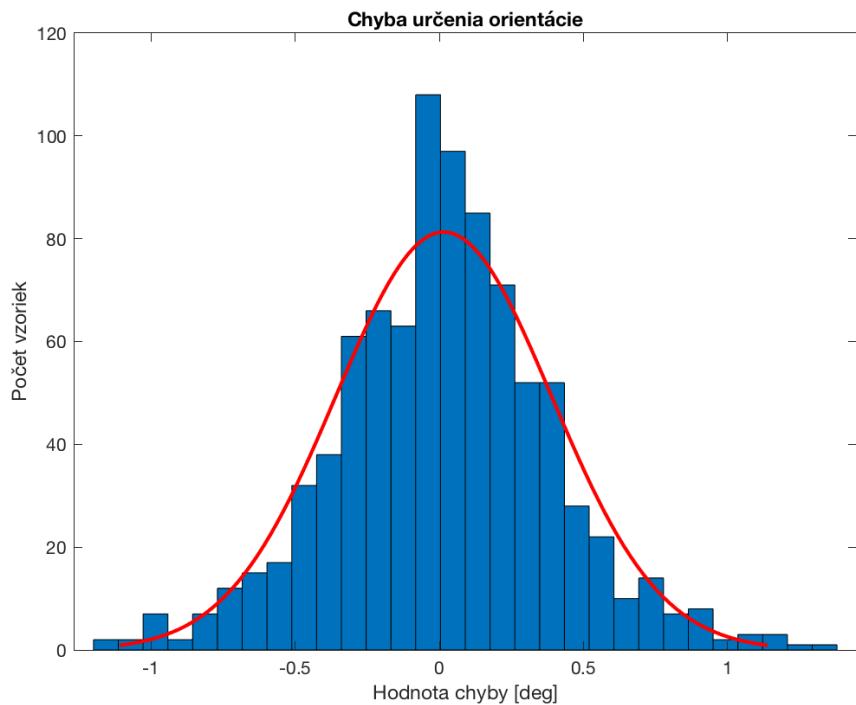
odrážajú štandardné správanie algoritmu. Pre extrémne hodnoty bude potrebné vytvoriť inú stratégiu spracovania. Nové rozloženia chýb polohy a orientácie po odstránení extrémnych hodnôt zobrazuje Obrázok 46.



Obrázok 45 - Histogramy chyby určenia polohy a orientácie pre rotačný pohyb 5x okolo osi robota. Červená krivka predstavuje normálne rozloženie.



Obrázok 46 - Rozloženie chyby určenia polohy robota po filtrácii extrémnych hodnôt pomocou pravidla 3-sigma.



Obrázok 47 - Rozloženie chyby orientácie po aplikácii metódy 3 sigma.

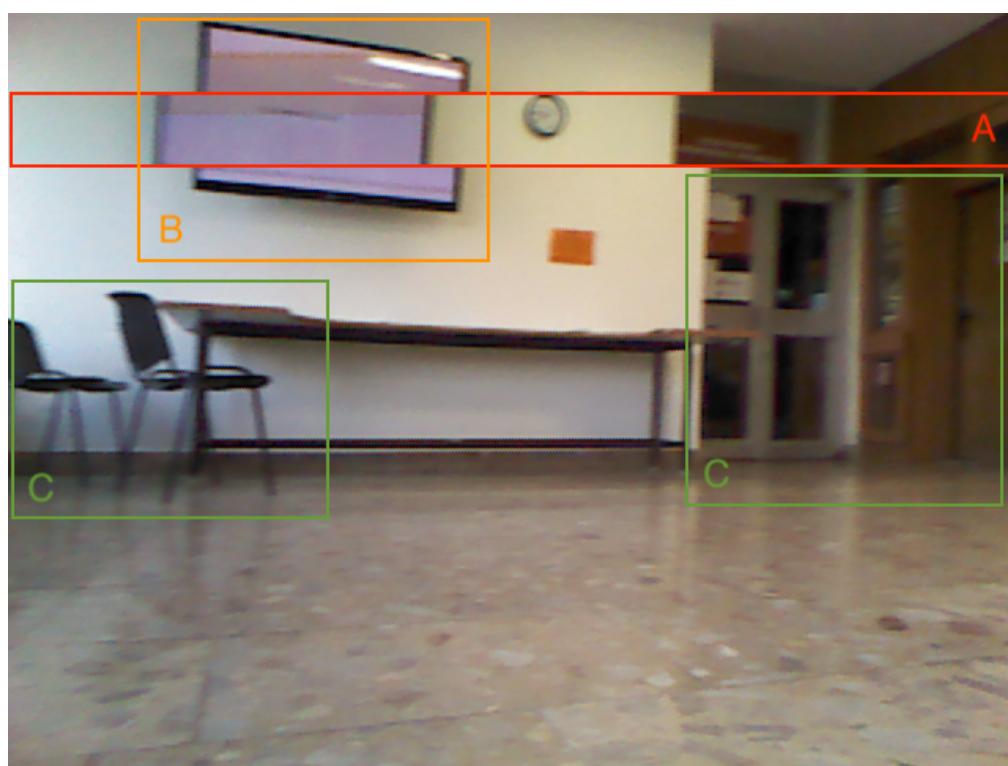
11.4 Identifikované problémy dostupných riešení

V datasetoch boli identifikované problémy snímania nasledovne:

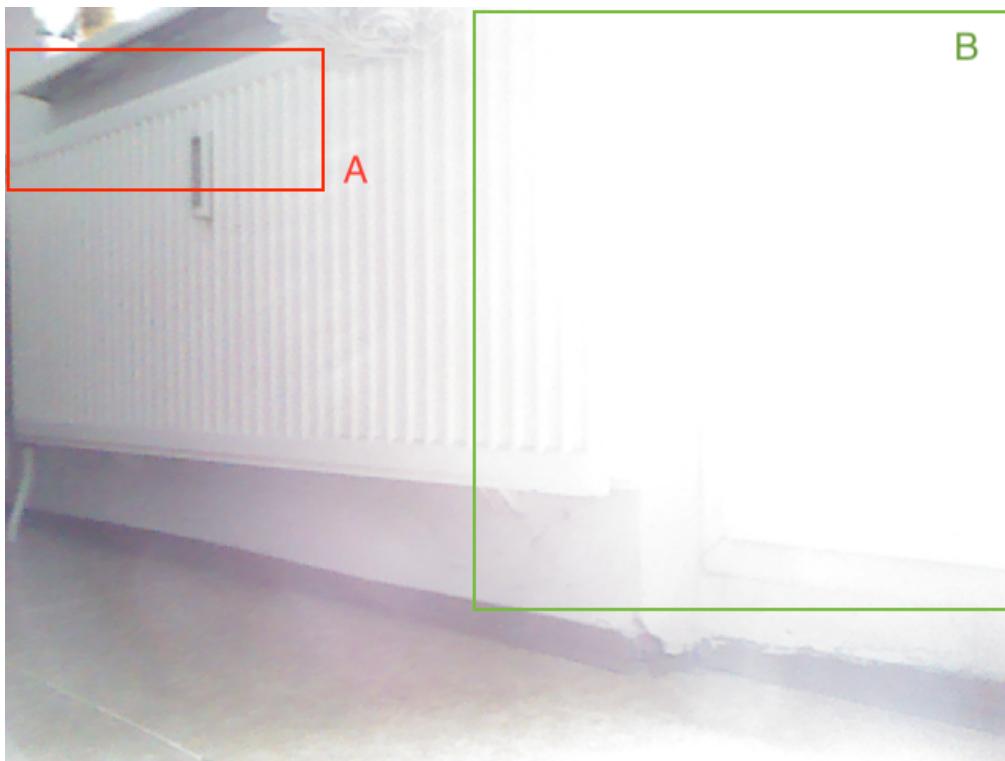
1. **Snímky s nevhodnou expozičiou** – snímky s presvetlenými alebo naopak slabovo osvetlenými oblasťami predstavujú problém pre spracovanie 2D obrazu, keďže obsahujú výrazne menej informácií ako bežné snímky. Na kvalite rekonštrukcie pohybu kamery sa tieto snímky prejavia zvýšením lokálneho šumu určenia polohy a orientácie, najmä vďaka nedostatku príznakov a korešpondencií. V extrémnych prípadoch tieto snímky nebudú obsahovať dostatok informácií pre ďalší výpočet a sledovanie pohybu bude ukončené.
2. **Dynamické objekty na scéne** – Ak scéna obsahuje dynamické objekty ako napríklad pohybujúce sa osoby alebo televízia zobrazujúca v čase meniaci sa obraz, môže dôjsť k chybnému výberu korešpondencií a ich asociácií. Chybné korešpondencie potom zapríčinia zlyhanie alebo veľmi chybné určenia transformácie polohy kamery.
3. **Snímky obsahujúce pohybové rozmazanie (motion blur)** – motion blur alebo pohybové rozmazanie predstavuje problém pri detekcii príznakov. Aj keď je väčšina detektorov navrhovaná tak, aby poskytovali určitú úroveň robustnosti voči tomuto javu, v praxi je zväčša potrebné výraznejšie rozmazania korigovať.
4. **Prostredia obsahujúce objekty odrážajúce všetko alebo časť svetla** – skenovanie povrchov z priečinného materiálu je problémom pre všetky kamery pracujúce na princípe projekcie štruktúrovaného svetla. Hĺbkové súradnice môžu úplne chýbať, alebo môžu byť nesprávne. Snímač Kinect trpí týmto problémom, ale prekvapivo dokáže snímať relatívne dobre cez sklo, napríklad sklenené dvere. Hĺbkové súradnice budú v tomto prípade iba výraznejšie šumieť.
5. **Poškodené snímky** – na našom kuse snímača Kinect sa prejavovala chyba, kde hĺbkový obraz neobsahoval celú informáciu, ale iba niekoľko riadkov z hĺbkovej kamery. Ostatné pixely boli nulové. Takýto snímok neobsahuje žiadne informácie pre prepočet polohy RGB korešpondencií do 3D. Podobne, na RGB obrazoch sa niekedy objavili chybné oblasti.
6. **Snímky obsahujúce periodicky opakujúcu sa štruktúru** – ak sa na scéne nachádza objekt, ktorý bude v 2D obraze obsahovať periodicky sa opakujúcu štruktúru je možné, že na každej opakujúcej sa vzorke budú síce príznaky detegované, ale kvôli ich výraznej vizuálnej

podobnosti budú nevhodne asociované. Tento problém je spoločný pre všetky riešenia vizuálnej odometrie a vizuálneho SLAM.

Niektoré z popísaných problémov zobrazujú obrázky Obrázok 48 – Problematické oblasti na snímku zo snímača Kinecta Obrázok 49. Na obrázku Obrázok 48 – Problematické oblasti na snímku zo snímača Kinect oblasť A predstavuje poškodený snímok zo snímača Kinect, dáta v nej sú úplne nesprávne. Oblasť B vyznačuje dynamický objekt. Ak pre výpočet transformácie pohybu kamery použijeme príznaky z tejto oblasti a na TV sa zmenil obraz, transformácia bude určená nesprávne. Oblasti C obsahujú motion blur. Na obrázku Obrázok 49 oblasť A predstavuje opakujúcu sa štruktúru – radiátor. Horné rohy jednotlivých rebier budú detegované ako príznaky, avšak vďaka ich vzájomnej vizuálnej podobnosti je možné, že tieto príznaky budú nesprávne asociované pri porovnávaní. Oblasť B je presvetiená vonkajším svetlom prichádzajúcim z okna. V tejto oblasti príznaky nebudú detegované.



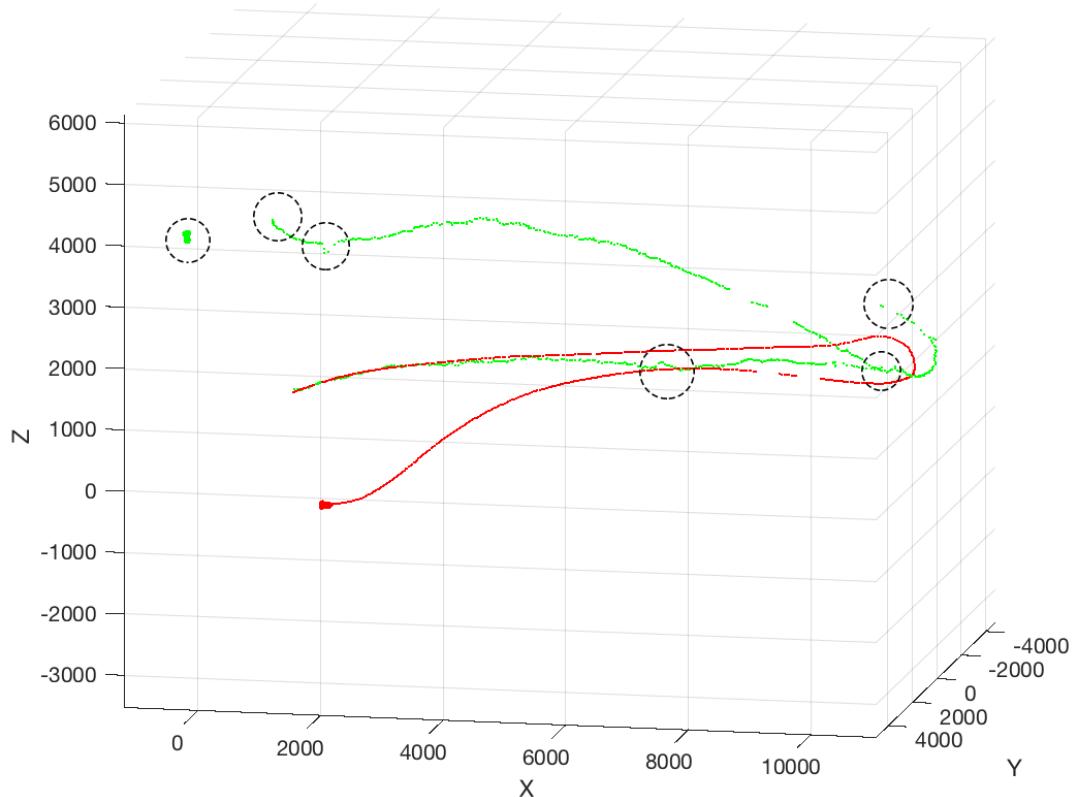
Obrázok 48 – Problematické oblasti na snímku zo snímača Kinect, prvá časť



Obrázok 49 – Problematické oblasti na snímku zo snímača Kinect, druhá časť

Na trajektóriach určených pomocou vizuálnej odometrie sa prejavuje kumulatívny charakter chyby určenia globálnej polohy. Ak je transformácia v aktuálnom snímku určená extrémne chybne, určenie všetkých nasledujúcich polôh bude taktiež zaťažené touto chybou. Obrázok 50 - oblasti výskytu extrémnej chyby určenia globálnej polohy kamery zobrazuje červenou referenčnú trajektóriu z gyro odometrie podvozku Kobuki, zelená farba reprezentuje trajektóriu zrekonštruovanú pomocou systému RGBD-6D-SLAM. Oblasti, v ktorých vyššie popisovaný problém nastal, sú vyznačené prerušovanými kružnicami. Takéto hrubé chyby majú za následok drastickú degradáciu kvality globálnej mapy a je nutné ich výskyt potláčať. Jednou z možností sú algoritmy optimalizácie grafu (kapitola 8.6).

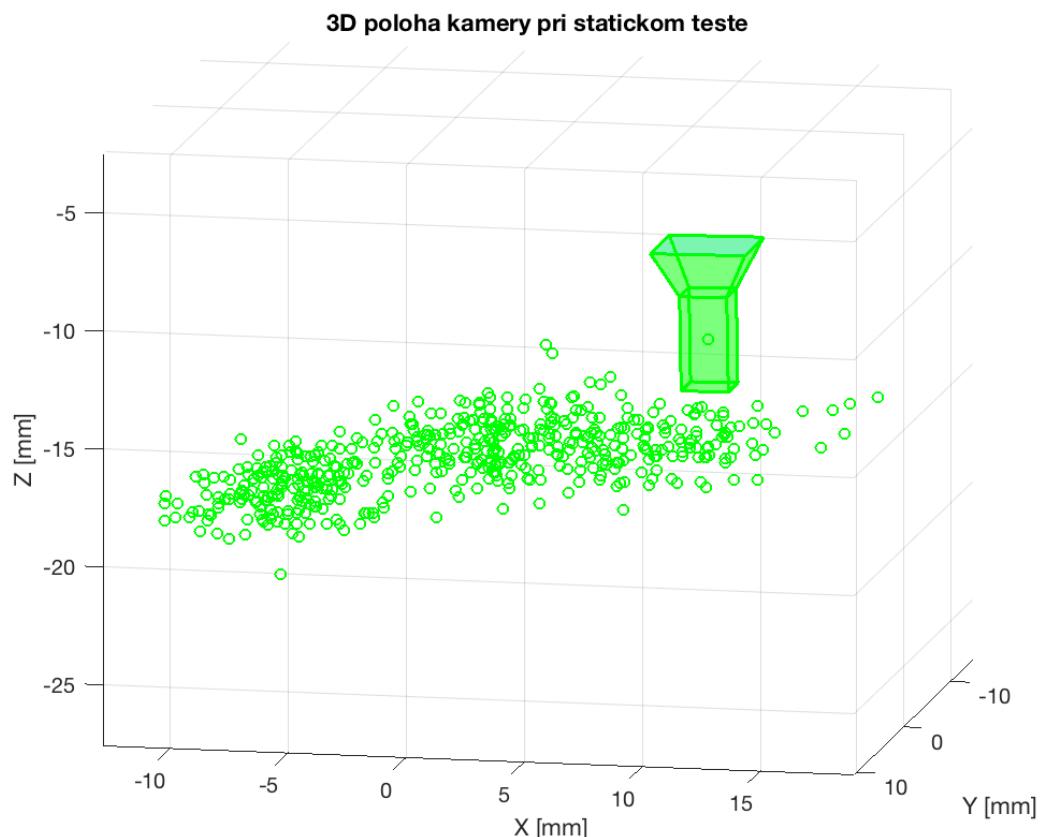
Poloha kamery v 3D



Obrázok 50 - oblasti výskytu extrémnej chyby určenia globálnej polohy kamery

Posledným identifikovaným problémom, ktorý bude riešený v tejto práci je problém **lokálnej stability určenia pozície kamery**. Ak kamera stojí a sníma statickú scénu, riešenie RGBD-6D-SLAM začne akumulovať drift a pozícia kamery sa v po čase presunie na úplne iné miesto. Vygenerovaná mapa sa tak často stane úplne nepoužiteľnou. Problém je možné riešiť pomocou detektie určitej metriky zmeny v obraze a definície prahovej hodnoty, ktorá bude spúštať určenie polohy. Všetky snímky s hodnotou pod týmto prahom budú jednoducho vynechané a poloha kamery bude považovaná za statickú. Tým efektívne potlačíme tempo rastu absolútnej chyby určenia pozície kamery. S týmto prístupom je možné sa stretnúť **riešení SLAM postavených na kľúčových snímkach (keyframe based SLAM)**. Ako metriku pre určenie pohybu je možné využiť štandardné prístupy detektie dynamických objektov v obraze alebo napríklad strednú hodnotu *vzdialenosť* porovnaných príznakov. Obrázok 51 - drift polohy kamery pri statickom teste zobrazuje, ako sa poloha kamery určená pomocou riešenia RGBD-6D-SLAM mení v čase, aj keď kamera fyzicky stojí. Kamera deteguje pohyb, aj keď ku žiadному pohybu nedochádza. Tento jav je zapríčinený šumovým

charakterom dát zo snímača Kinect, chybami výpočtov a neistotou, ktorú so sebou prináša detekcia príznakov a nedeterministická metóda RANSAC. Úspešné riešenie vizuálnej odometrie by malo tento jav potláčať.



Obrázok 51 - drift polohy kamery pri statickom teste

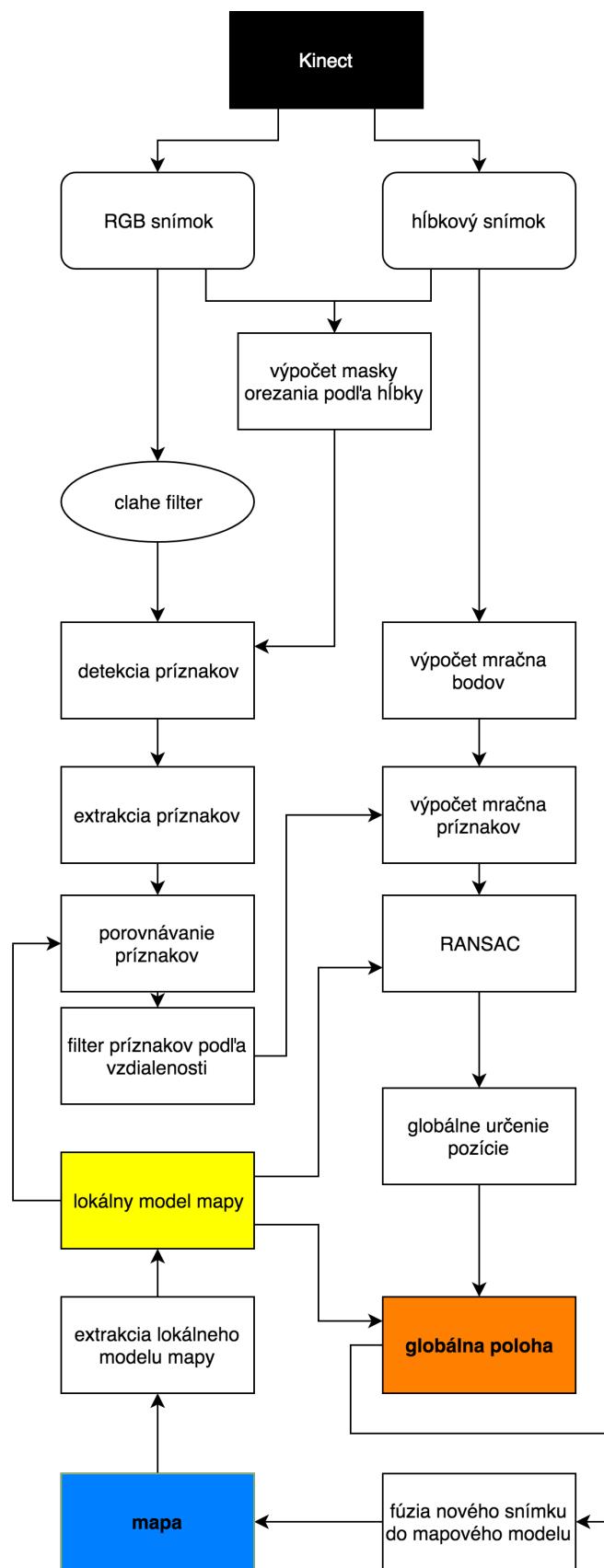
11.5 Navrhnuté riešenie

Základnou otázkou, ktorú je potrebné vyriešiť pri návrhu riešenie vizuálnej odometrie a mapovania je výber typu vizuálnej odometrie vhodného pre cieľovú platformu. Na platformách s vysokým výkonom alebo disponujúcimi grafickou kartou je výhodné využiť náročnejšie metódy. To sú napríklad metódy priamej vizuálnej odometrie alebo prístupy postavené na technike Kinect Fusion. Tieto prístupy umožňujú vytváranie máp vo veľmi vysokej kvalite. Situácia je však odlišná na malých mobilných platformách. Pre tieto platformy a teda aj pre naše riešenie, je najvhodnejšie použiť typ **vizuálnej odometrie postavenej na príznakoch (feature-based VO)**, ktoré:

- Redukujú počet spracovaných dát na minimálnu relevantnú vzorku, čím dosahujú výraznú úsporu času spracovania.
- Sú odolné na väčšie posuny kamery, čo umožňuje konzistentné mapovanie aj pri rýchlejších posunoch kamery a naopak možnosť vynechávania snímok pri pomalom pohybe za účelom zníženia výpočtovej náročnosti riešenia.

Nevýhodou tohto typu vizuálnej odometrie sú kroky detekcie, extrakcie a porovnávania príznakov, ktoré sú výpočtovo náročné a je potrebné ich vykonávať sekvenčne za sebou. Pre prekonanie tohto problému bola navrhnutá **schéma paralelného spracovania na CPU**. Zároveň bol vybraný najvhodnejší detektor príznakov z hľadiska výpočtového času. Nevýhodou príznakových metód je aj fakt, že porovnané príznaky obsahujú tzv. outliers, čiže ich nie je možné priamo použiť pre určenie transformácie pohybu kamery. Tento problém bol riešený pomocou **filtra príznakov a metódy RANSAC**. Pre zvýšenie odolnosti na zlé svetelné podmienky a nedostatočnú štruktúru scény bolo navrhnuté využitie filtra **CLAHE**.

Za účelom zníženia tempa rastu globálnej chyby boli využité princípy **porovnávania snímku a lokálneho modelu (frame-to-model matching)**. Tento prístup zároveň umožňuje algoritmu obnovu správnej pozície po výskytte extrémnej chyby. Blokovú schému celého navrhovaného riešenia zobrazuje Obrázok 52. Kľúčové časti riešenia sú popísané a vyhodnotené v nasledujúcich kapitolách. Oproti riešeniu RGBD-6D-SLAM [30] nebola pre doladenie transformácie pohybu kamery použitá metóda ICP, keďže ju nebolo možné implementovať ani parametrizovať pri zachovaní požadovanej frekvencie spracovania.



Obrázok 52 - Bloková schéma navrhnutého riešenia

11.6 Maska orezania RGB snímku podľa híbkky

Veľkosť spracovávanej dátovej vzorky určuje u algoritmov prezentovaných v tejto práci časovú náročnosť pre výpočet algoritmu. V každej časti algoritmu by mali byť spracovávané iba relevantné dátá a procesorový čas by nemal byť miňaný na zbytočné výpočty. V práci bude predstavených viacero konceptov, ktoré týmto spôsobom pracujú a umožňujú beh riešenia v reálnom čase. Jedným zo základných konceptov redukcie dátovej vzorky pri híbkových kamerách je orezanie RGB snímku podľa híbkového obrazu. Pixely, kde nebolo možné určiť híbkovú súradnicu Z nebudú zahrnuté ani do výpočtu detektora príznakov, teda budú ignorované všetkými nasledujúcimi časťami algoritmu. To sa týka oblastí okrajov obrazu, kde híbkové merania nie sú k dispozícii priamo zo snímača Kinect, a oblastí, kde sa híbka nachádza za meracím rozsahom. Odstránenie zbytočne spracovávaných bodov prináša výrazné zrýchlenie spracovania v ostatných častiach algoritmu najmä na scénach, kde sa tieto body vyskytujú vo vysokej mieri.



Obrázok 53 - Vľavo RGB obraz, vpravo maska orezania podľa híbkky. Detektor príznakov bude spracovávať pixely ľavého obrazu iba tam, kde je pixel s rovnakými súradnicami $[x, y]$ v pravom obraze biely.

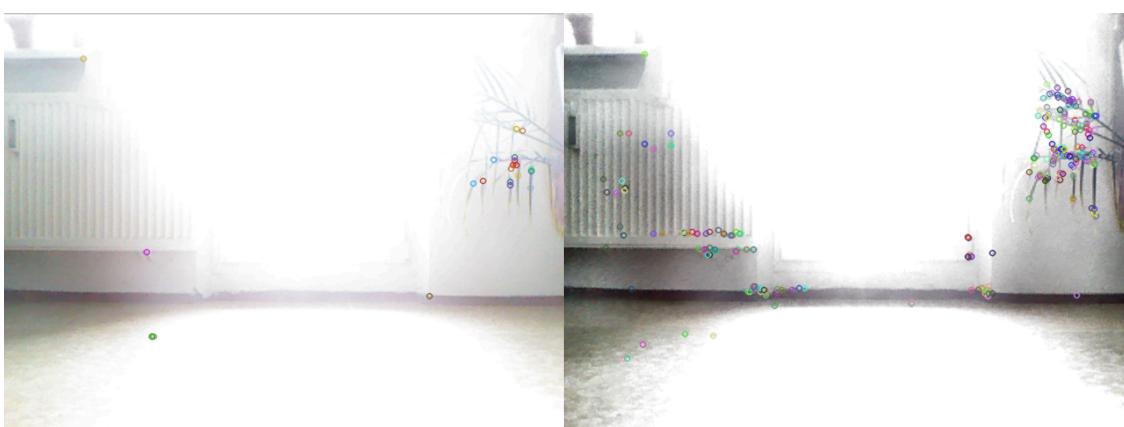
p.č.	% redukcia množstva bodov
1	70,5348
2	62,1471
3	68,8212
4	68,6435
5	68,8212
6	64,6113
7	68,0501

Tabuľka 2 - Redukcia počtu pixelov RGB obrazu po orezanií podľa híbkky na viacerých testovacích datasetoch. Prvý stĺpec zobrazuje poradové číslo spracovaného datasetu. Percentuálna hodnota uvádzá percento ponechaných pixelov z 307 200, čiže z plného rozlíšenia 640x480 snímača Kinect.

11.7 Filter CLAHE

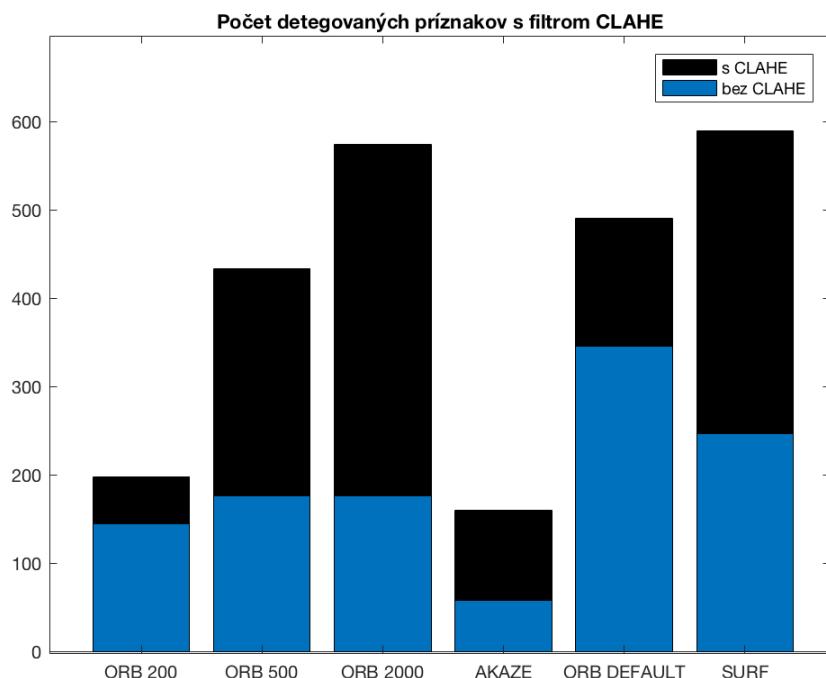
AHE je skratka pre **adaptive histogram equalization** teda **adaptívnu ekvalizáciu histogramu** [94]. Jedná sa o techniku využívanú v počítačovom spracovaní obrazu pre zlepšenie kontrastu v obraze. Odlišuje sa od klasickej ekvalizácie histogramu [95] tak, že pre výpočet používa viacero histogramov, ktoré korešpondujú s lokálnymi časťami obrazu a používa ich pre zlepšenie lokálneho kontrastu v tejto oblasti a teda zvýraznenie rohov a hrán v obraze. Jednou z nevýhod AHE je, že zvýrazňuje šum v relatívne homogénnych častiach obrazu. Metóda CLAHE (contrast limited AHE) [96] oproti AHE navyše obmedzuje silu zvýraznenia v lokálnych častiach a tak úspešne vznik šumu potláča.

Filter CLAHE je v navrhnutom riešení použitý pre kompenzáciu nevhodného osvetlenia scény. Snahou bolo nájsť filter, ktorý je možné použiť pre spracovanie v reálnom čase, teda s čo najmenšími nárokmi na výpočtovú zložitosť algoritmu. Filter CLAHE má výpočtovú zložitosť $O(N^2)$ kde N je šírka okna použitého pre výpočet v pixeloch. Rýchlosť výpočtu CLAHE filtra pre jeden snímok z RGB kamery snímača Kinect, teda pre rozlíšenie 640x480 pixelov bola priemerne 7 ms pre všetky overované datasety. Kedže cieľom bola úprava svetelnosti scény, obraz bol pred aplikáciou filtra prevedený do farebného modelu CIELAB [97] a filter bol aplikovaný iba na zložku obrazu L , ktorá korešponduje so svetelnosťou obrazu. Vplyv aplikácie filtra CLAHE na RGB snímok zo snímača Kinect obsahujúcim presvetlenú oblasť zobrazuje Obrázok 54.



Obrázok 54 - porovnanie RGB obrazu v presvetenej oblasti. Vľavo obrázok z kamery s detegovanými príznakmi, vpravo obraz po aplikácii filtra CLAHE. Na obraze vidíme vyššie množstvo detegovaných príznakov, výraznejšie hrany a rohy.

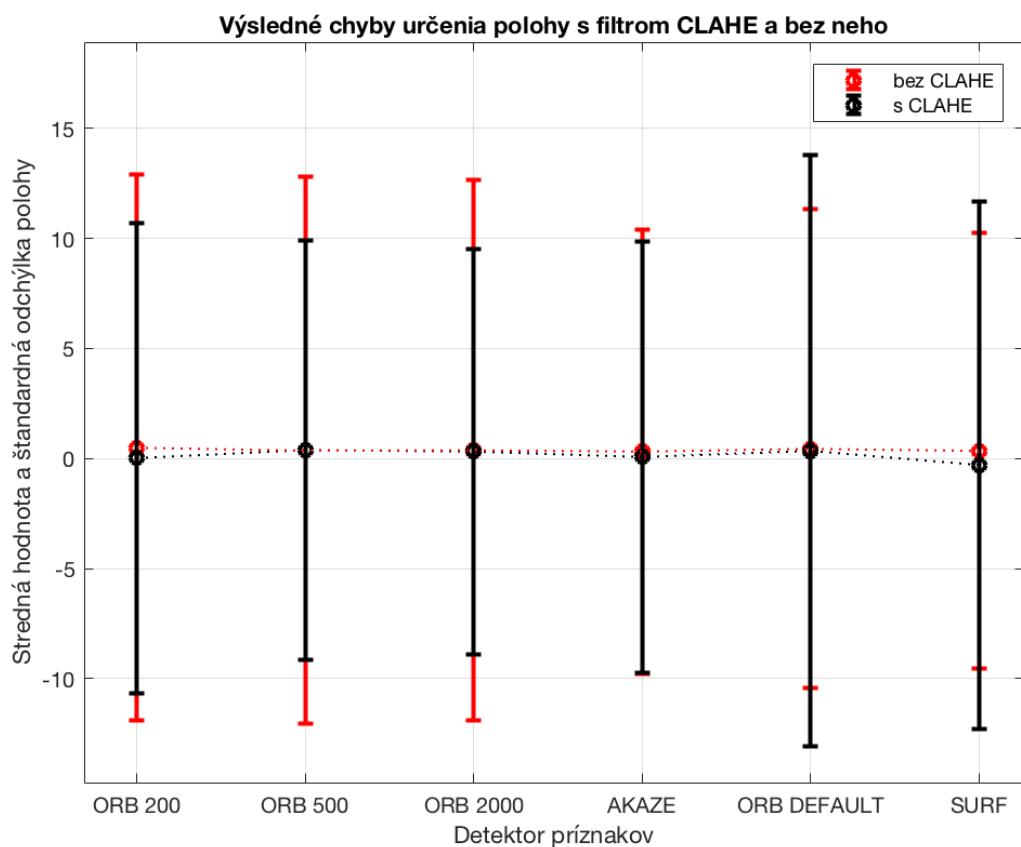
Experimentálne overenie ukázalo, že filter CLAHE zvyšuje kvalitu výpočtu transformácie pohybu kamery v oblastiach trajektórie, kde RGB kamera sníma presvietené, podsvietené, alebo príliš svetelne kontrastné scény. U všetkých testovaných detektorov príznakov dôjde ku zvýšeniu práhu pozorovateľnosti príznaku ako následku lokálnej korekcie kontrastu. Tým pádom sa zvýsi počet detegovaných príznakov. Experimentálne bol overený aj vplyv filtra CLAHE na kvalitu rekonštrukcie pohybu. Vyhodnotených bolo 6 konfigurácií detektorov príznakov. Prvé 4 konfigurácie – počtovo obmedzený ORB detektor a detektor AKAZE predstavovali rýchle detektory, ktoré boli nakonfigurované pre rýchlosť s relatívne malou robustnosťou na zmenu mierky. Detektor ORB Default a SURF predstavovali detektory v konfigurácii priamo z knižnice OpenCV, ktoré majú túto robustnosť značne vyššiu. Dôvody, prečo boli použité práve tieto detektory príznakov sú diskutované v kapitole 11.9.

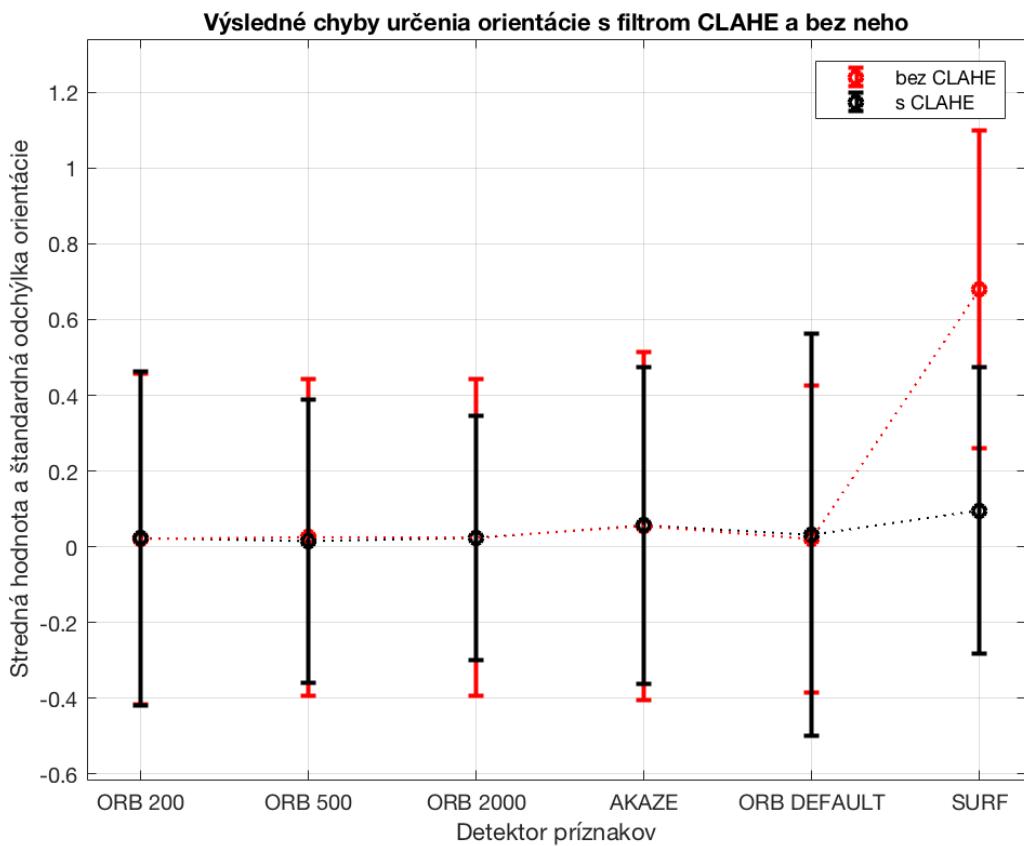


Obrázok 55 - priemerný počet detegovaných príznakov na snímkach.

Počas experimentov nebolo pozorované, že by použitie filtra CLAHE s detektormi príznakov ORB a AKAZE so zníženou robustnosťou na zmenu mierky prinášalo nevýhody, ak sa jednalo o spracovanie v reálnom čase, t.j. algoritmus bol schopný spracovať všetky dodané snímky s frekvenciou ~ 30 Hz aj s využitím filtra. Ak však použitie filtra prináša spomalenie celkového výpočtu vizuálnej odometrie a tým pádom aj pokles frekvencie spracovania snímok, je potrebné jeho použitie zvážiť. Do úvahy pripadá aj jeho cielená aktivácia a deaktivácia vo

zvolených úsekokch scén, keď si to osvetlenie scény vyžaduje alebo na surových snímkach z RGB kamery nie je dostatočné množstvo detegovaných príznakov. Zvýšenie počtu spracovávaných príznakov totiž so sebou prináša aj pridanú výpočtovú náročnosť v ďalších častiach algoritmu, napríklad v metóde RANSAC. Negatívny vplyv na kvalitu rekonštrukcie sme zaznamenali u detektorov výrazne robustných na zmenu mierky. Tento jav je zapríčinený zvýšením počtu falošných detekcií v jednotlivých úrovniach mierky. Preto pri týchto detektoroch použitie filtra limitujeme na scény, kde sa príznaky nachádzajú vo veľmi malom počte a je nevyhnutné ich počet zvyšovať.





Obrázok 56 - Chyby polohy a orientácie s filtrom CLAHE a bez neho pre rôzne typy a konfigurácie detektorov príznakov

11.8 Paralelné spracovanie

Využitie viacerých výpočtových jadier pri implementácii operácií v počítačovom spracovaní obrazu je tradičnou technikou, ako zvýšiť rýchlosť spracovania dátovej vzorky. V praxi sa je možné sa stretnúť s dvoma druhmi paraleлизmu – paraleлизmus na grafickej karte počítača (GPU) a paraleлизmus na výpočtovej jednotke CPU. Oba postupy využívajú odlišnú hardvérovú jednotku a ku ich návrhu je potrebné pristupovať odlišne. Pri paraleлизme na GPU sú implementované algoritmy s vysokým prietokom dát, kde je vysoké množstvo dátových vzoriek spracovávaných súčasne jedným algoritmom na veľkom počte jadier. Pri paraleлизme na CPU odlišujeme úlohy, ktoré môžu prebiehať súčasne od tých, ktoré musia prebiehať sekvenčne, organizujeme ich simultánny beh a tým je dosiahnutá úspora času spracovania.

Kedže neuvažujeme, že malý mobilný robot bude vybavený GPU, vytvorený systém vizuálnej odometrie bol paraleлизovaný na CPU. Táto kapitola opisuje návrh distribúcie výpočtu vizuálnej odometrie na viaceru jadier, ktorý bol implementovaný v našom riešení.

Po úspešnej implementácii algoritmu schopného spracovať všetky zozbierané datasety na jednom vlákne boli zmerané priemerné časy behu všetkých jeho kľúčových častí na všetkých zozbieraných datasetoch (16 000 snímok). Ako výpočtovo najzložitejšie sa ukázali – výpočet filtra CLAHE (9 ms), výpočet mračna bodov (8 ms) detekcia príznakov a extrakcia príznakov (15 – 200 ms), porovnávanie príznakov (15-40 ms) a samotná metóda RANSAC (2 – 200 ms). Kedže stredná hodnota spracovania jedného snímku na jednom výpočtovom jadre je vždy viac ako 30 ms, nebolo možné dosiahnuť spracovanie v reálnom čase na jednom jadre.

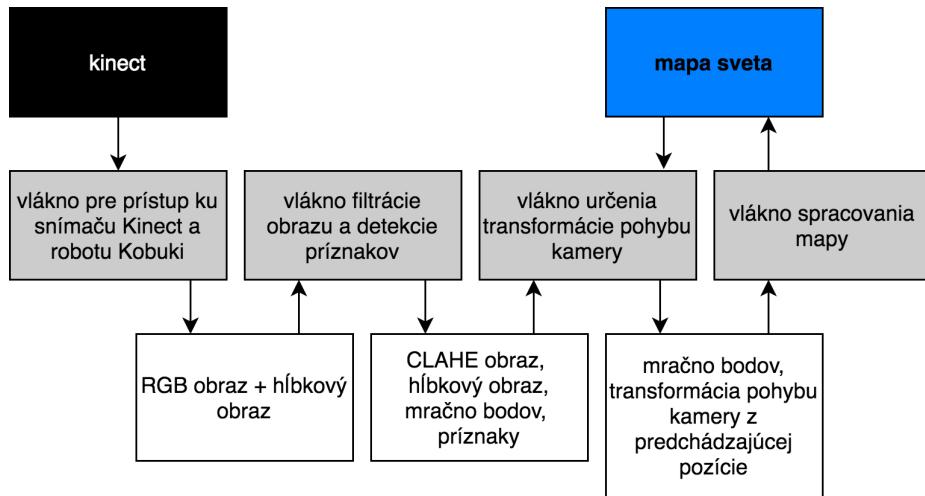
Pri našej implementácii boli využité 4 simultánne bežiace vlákna, ktoré spolu zdieľajú 3 zdieľané prostriedky:

1. **Vlákno pre prístup ku snímaču Kinect** – vlákno potrebuje v priemere 1 ms pre získanie snímku zo snímača Kinect a 5 ms pre prečítanie uloženého snímku z disku pri behu zo záznamu. Vlákno aktuálne získaný snímok uloží do zdieľanej premennej pre ďalšie vlákno a spí. Čas spania je nastavený tak, aby vlákno generovalo nový snímok približne každých 30 ms, ako snímač Kinect. Pomocou zmeny hodnoty spania je možné simulovať iné snímkovacie frekvencie snímača.
2. **Vlákno filtrácie a detektie príznakov** – toto vlákno si preberie snímok zo zdieľanej premennej z vlákna 1 a vykoná nasledovné operácie:
 - a. **Aplikuje CLAHE filter**
 - b. **Vypočíta masku orezania podľa hĺbky**
 - c. **Vypočíta mračno bodov pre aktuálne spracovávaný snímok**
 - d. **Deteguje príznaky**
 - e. **Extrahuje príznaky**

Vypočítané dátá o snímku vlákno uloží spolu so vstupnými dátami do zdieľanej premennej pre vlákno 3 a spí, pokiaľ sa v premennej zdieľanej s vláknom 1 neobjaví nový, nespracovaný snímok. Cieľom bolo implementovať všetky potrebné operácie v tomto vlákne pod 30 ms, keďže snímkovacia frekvencia snímača Kinect je ~30 Hz a snímky teda prichádzajú každých 30 ms.

3. **Vlákno určenia transformácie pohybu kamery** – vlákno porovná deskriptory z predchádzajúcej snímky so snímkou v zdieľanej premennej s vláknom 2, ak už ho nespracovalo v predchádzajúcej iterácii. Deskriptory sú následne filtrované podľa vzdialenosťi z porovnávača príznakov. Potom sa skonštruujú mračná bodov z úspešne porovnaných príznakov a tie spracuje metóda RANSAC. Po robustnom určení transformácie sa transformácia uloží do zdieľaného prostriedku s vláknom 4 a vlákno spí, pokiaľ sa na jeho vstupe neobjaví nový snímok. Ak určenie transformácie zlyhá (napríklad v aktuálnom snímku nie je k dispozícii aspoň 5 príznakov a teda nie je možné určiť transformáciu pohybu kamery), vlákno na svoj výstup nastaví jednotkovú maticu 4×4 , predstavujúcu nulový pohyb.
4. **Vlákno spracovania mapy** – v tomto vlákne sa aktuálny posun kamery integruje s predchádzajúcimi transformáciami pre získanie globálnej polohy. Mračno bodov z aktuálneho snímku je následne transformované pomocou tejto polohy do globálnych súradníc a zlúčené s mapou prostredia.

Implementácia takéhoto systému predpokladá použitie korektného synchronizačného mechanizmu prístupu ku zdieľaným prostriedkom. V teórii softvérového inžinierstva je možné implementovanú schému klasifikovať ako konkurentnú (concurrent) bežiacu paralelne [98]. Viaceré odlišné úlohy navrhnutej schéme totiž prebiehajú konkurentne (súčasne) a zároveň na viacerých jadrách procesora, disponujú zdieľanými prostriedkami a organizované ich využívajú. V ideálnom prípade by malo byť možné pri takomto modeli umiestniť do každého vlákna algoritmus trvajúci maximálne 30 ms a aplikácia by mala byť schopná spracovať snímky v reálnom čase. V praxi však algoritmy nie je možné zrýchliť lineárne pridávaním výpočtových CPU jadier, keďže obsahujú aj časti, ktoré musia bežať sekvenčne za sebou. V tomto prípade sú takýmito časťami najmä časti prístupu ku zdieľaným prostriedkom. Každý spracovávaný snímok je totiž nutné spracovať štyrmi vláknami v poradí presne od prvého po posledné a vlákna teda k tejto úlohe pristupujú sekvenčne. Zároveň, pri spracovaní na bežnom PC je možné, že operačný systém sa rozhodne niektorú z úloh pozastaviť na väčší ako požadovaný čas, keďže bežné implementácie paraleлизmu na CPU negarantujú presný dostupný procesorový čas. Schéma distribúcie výpočtu vizuálnej odometrie na viaceré vlákna je zobrazená na Obrázok 57.



Obrázok 57 - Schéma distribúcie výpočtu vizuálnej odometrie vo vytvorenom riešení medzi viacerom výpočtových vlákien. Šedé bloky predstavujú vlákna, biele bloky predstavujú zdieľané prostriedky.

Experimentálne overenie bolo realizované pomocou spúšťania algoritmu s inou konfiguráciou detektora príznakov ORB tak, aby vždy vykonal detekciu za rozdielny čas. Tento detektor je pre tento experiment vhodný najmä z dôvodu, že umožňuje definíciu požadovaného množstva detegovaných príznakov. V experimente bol zvyšovaný počet požadovaných vzoriek aj parametre ovplyvňujúce úspešnosť detekcie pri zmene mierky, ktoré sa u množstva detektorov ukázali ako jednoduchý spôsob zrýchlenia behu detekcie za cenu straty istej miery robustnosti. Pre ďalšie predĺženie spracovania bol využitý aj filter CLAHE. **V praxi bolo možné využívať 2 jadrá procesora s dobou spracovania max 25 ms pre jednu iteráciu a zvyšné 2 jadrá s dobou spracovania max 10 ms a dosahovať spracovanie snímok frekvenciou 30 Hz.**

Ak sa algoritmus dostane do situácie, kde nestíha snímky spracovať s frekvenciou 30 Hz, začne snímky tzv. odhadzovať - nebude ich akumulovať vo vstupnej fronte, ale odhadí ich. **Odhadzovanie snímok (frame dropping)** je v oblasti vizuálnej odometrie a vizuálneho SLAM štandardnou technikou, slúžiacou pre optimalizáciu času spracovania. Nebezpečím tohto prístupu je zníženie maximálnej možnej rýchlosťi pohybu kamery a zníženie celkového množstva extrahovaných informácií o pohybe kamery zo vstupných dát. Výhodou naopak môže byť znížené tempo rastu globálnej chyby, keďže pri nižšej frekvencii spracovania kumulatívna chyba narastá pomalšie. Prístupy vizuálneho SLAM, ktoré využívajú tento jav implementujú koncept kľúčových snímok (keyframe SLAM). Pri tomto type SLAM sa snažíme spracovať čo najmenej snímok, teda sa zameriavame na rekonštrukciu z maximálnych možných pohybov kamery. Takto dosahujeme zníženie tempa rastu globálnej chyby vizuálnej

odometrie, pretože tá je závislá od počtu spracovaných snímok, ako vidíme na Obrázok 42 - absolútnej chybe posunu kamery počas piatich rotácií robota pri rotačnom teste.. Ako vidíme v Tabuľke 3, algoritmus je schopný spracovať snímky s frekvenciou 30 Hz aj pri akumulovanom čase spracovania cca 45 ms.

názov datasetu	vlákno 2 [ms]	vlákno 3 [ms]	total [ms]
ORB 200 C	18,5881	4,06569	22,65379
ORB 200	11,532	3,20626	14,73826
ORB 500 C	20,501	13,502	34,003
ORB 500	10,6227	3,34736	13,97006
ORB 2000 C	20,9755	24,3108	45,2863
ORB 2000	10,5005	3,30039	13,80089
ORB D C	29,9256	15,7048	45,6304
ORB D	19,9903	7,58725	27,57755

Tabuľka 3 - Celkový výpočtový čas paralelného spracovania snímku pri výstupnej frekvencii 30Hz. Prvý stĺpec označuje názov datasetu. Číslo označuje horný limit počtu detegovaných príznakov detektora ORB, písmeno C označuje, že sa jedná o prednastavenú konfiguráciu detektora z knižnice OpenCV. Písmeno D označuje použitie filtra CLAHE. Stĺpec 2. a 3. zobrazujú priemerné časy behu výpočtu dvoch najdôležitejších vláken programu – vlákna spracovania a vlákna výpočtu transformácie pohybu kamery. Posledný stĺpec pre prehľadnosť zobrazuje sumu stĺpcov 2 a 3.

Pri analýze správania implementovaného modelu bolo zároveň pozorované, že na začiatku nábehu aplikácie sa snímky spracovávajú výrazne pomalšie. Rýchlosť spracovania sa stabilizuje po približne 2 s, teda cca 60 snímkach. Dôvodom bude pravdepodobne rézia spojená so spustením viacerých vláken programu a inicializovaním potrebných prostriedkov operačným systémom.

11.9 Detekcia, extrakcia a porovnávanie príznakov, filter porovnaných príznakov podľa vzdialenosťi

Pre implementáciu detekcie a extrakcie príznakov z 2D obrazu boli využité niektoré algoritmy dostupné v knižnici OpenCV. V literatúre nájdeme veľké množstvo rôznych komparatívnych analýz úspešnosti, rýchlosťi, opakovateľnosti aj spoľahlivosti týchto detektorov [99] [100] [101] [102] [103] [104]. Zvýšenú pozornosť v ostatných rokoch dostávajú moderne a rýchle detektory ORB [44] a AKAZE [105] používajúce robustné binárne deskriptory, ktoré poskytujú vysokú rýchlosť detekcie, extrakcie, aj porovnávania príznakov. V úvode práce boli spustené nasnímané sekvencie a analyzovaná rýchlosť behu a počet detegovaných príznakov. Prehľad priemerného počtu detegovaných príznakov v jednom snímku a priemerný čas behu detekcie a extrakcie príznakov pre jednotlivé testované detektory zobrazuje Tabuľka 4.

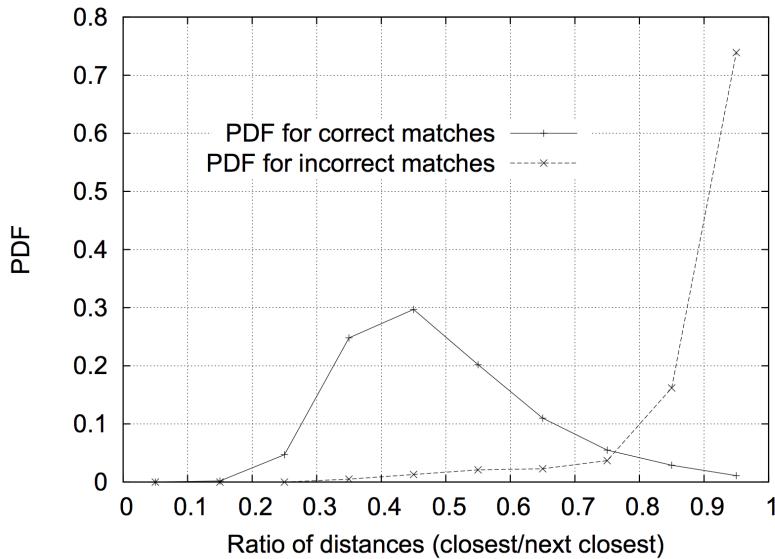
názov datasetu	priemerný počet príznakov v jednom snímku	priemerný čas detekcie a extrakcie [ms]
ORB 200 + clahe	197.993	3.48101
ORB 200	144.229	3.33818
ORB 500 + clahe	433.462	4.39181
ORB 500	175.714	2.98351
ORB 2000 + clahe	574.047	4.54848
ORB 2000	175.679	2.92823
ORB CV default + clahe	489.995	14.6869
ORB CV default	345.78	12.3531
AKAZE FAST + clahe	159.826	47.3733
AKAZE FAST	57.4758	42.7495
SURF 400	246.734	96.0456
SURF 400 + CLAHE	589.352	158.517
BRISK	1407.23	53.7613
BRISK + CLAHE	7628.71	428.354

Tabuľka 4 - prehľad priemerného času behu a počtu identifikovaných príznakov pomocou detektorov ORB, AKAZE, SURF a BRISK. Vidíme, že použitie filtra CLAHE významne zvyšuje počet identifikovaných príznakov. Vyznačený riadok reprezentuje name zvolenú konfiguráciu detektora pre výsledné riešenie.

Iba detektor príznakov ORB bol schopný behu pod 30 ms pre jeden snímok zo snímača Kinect. Preto bol detektor ORB zvolený ako primárny detektor pre naše riešenie. Detektory ktoré neboli schopné v riešení konzistentne rekonštruovať pohyb kamery, alebo ich beh trval ešte dlhšie ako 400 ms, nie sú v tabuľke uvádzané. Jedná sa o detektory a deskriptory BRIEF, FAST, FREAK, MSER, STAR a Harris. Uvedený nie je ani detektor KAZE, ktorý je iba výpočtovo náročnejšou verziou detektora AKAZE, iba s mierne lepšími výsledkami. Pre porovnanie však uvádzame detektor SURF, ktorý bol využívaný v riešení RGBD-6D-SLAM [30]. Čas spracovania závisí čiastočne od počtu detegovaných príznakov. Preto by mal byť počet detegovaných príznakov v každej iterácii najnižší možný pre úspešný výpočet dostatočne presnej transformácie pohybu kamery.

Pre implementáciu porovnávania príznakov bola využitá knižnica OpenCV, ktorá ponúka dve stratégie určenia korešpondujúceho príznaku – výpočet hrubou silou (Brute-force matching) a rýchle hľadanie približného najbližšieho suseda (Fast Approximate Nearest Neighbor Search). **Výstupom porovnávania príznakov z predchádzajúceho a aktuálneho snímku je tabuľka popisujúca zoznam najbližších susedov a hodnotou vzdialenosťi. Hodnota vzdialenosťi v tomto prípade nepopisuje euklidovskú vzdialenosť príznakov v priestore, ale vyjadruje mieru odlišnosti asociovaných príznakov.** Preto bude v nasledujúcom texte vyznačená kurzívou. **Príznaky s najnižšou možnou vzdialenosťou predstavujú identické časti obrazu. Naopak príznaky s najvyššou možnou vzdialenosťou reprezentujú úplne odlišné časti obrazu. Vzdialenosť nemá definovanú jednotku.**

Každý z testovaných detektorov má iné štatistické rozloženie hodnôt vzdialenosťi. Medzi hodnotami sa totiž nachádzajú všetky vypočítané asociácie – aj tie správne, aj tie nesprávne. Preto už spolu s detektorom SIFT [36] jeho autor David G. Lowe v experimente určil štatistické rozloženie správnych a nesprávnych asociácií a z nich navrhol orezanie množiny asociácií podľa určeného koeficientu. Efektom bola výrazná redukcia počtu chybných korešpondencií pri malej redukcii počtu správnych korešpondencií, ako zobrazuje Obrázok 58 – Neprávne vzorky sú vyznačené prerušovanou čiarou, správne plnou. Na základe tohto grafu navrhol David G. Lowe orezanie korešpondencií so vzdialenosťou nad 0.8, čo eliminuje 90% nesprávnych vzoriek, ale len 5% správnych vzoriek. [36]



Obrázok 58 – Odvodenie koeficientu orezania vzdialenosť navrhnuté David G. Lowe v [36]

Inšpirovať sa týmto kritériom je užitočné aj pri deskriptoroch SURF, ORB a AKAZE. V niektorých implementáciách vizuálnej odometrie využívajúcich detektor SURF sa najprv spočíta aritmetický priemer všetkých *vzdialenosťí* a potom sa odstránia vzorky, ktoré sú *ďalej* ako táto priemerná hodnota. Tento postup sa ukazuje ako účinný spôsob redukcie nesprávnych vzoriek v porovnaných príznakoch pri tomto detektore. Experimentálne bola určená optimálna hodnota tohto parametra aj pre detektory ORB a AKAZE. Pre každý detektor bola určená stredná hodnota *vzdialenosťí* porovnávaných príznakov. Následne bola táto hodnota použitá ako parameter prahu pre redukciu počtu korešpondencií. Experimentálne boli overené aj hodnoty blízko strednej hodnoty a vybraná bola tá, kde sa nachádzalo najbližšie lokálne minimum chyby určenia transformácie pohybu kamery.

Experimentálne určená hodnota prahu sa javila u každého detektora ako najlepšia bez ohľadu na skúmaný dataset. Konkrétnie hodnoty zobrazuje Tabuľka 5. Experimenty preukázali, že je výhodné hodnotu prahu oproti strednej hodnote všetkých určení mierne zvýšiť na 110-150% strednej hodnoty. To zvýši aj robustnosť celého algoritmu odometrie. Na scénach, kde došlo ku veľkým pohybom kamery medzi snímkami alebo zmenám osvetlenia sa často mení aj vizuálna podoba príznaku. Ak je táto dynamika príznakov zanedbaná, spôsobí to pokles robustnosti. Zvýšená *vzdialosť* korešpondencie teda nemusí nevyhnutne znamenať, že korešpondencia je chybná.

Zvýšenie prahu na viac ako 150% strednej hodnoty spôsobuje zvýšenie strednej hodnoty a štandardnej odchýlky chyby určenia polohy aj orientácie. Dôvodom je výber príliš veľkého množstva chybných korešpondencií do vzorky spracúvanej metódou RANSAC.

Parameter prahovej hodnoty *vzdialenosťi* príznakov sa ukázal ako kľúčový parameter pre úspešnosť vytvoreného riešenia. Príliš nízka hodnota aj príliš vysoká hodnota spôsobia úplné zlyhanie algoritmu.

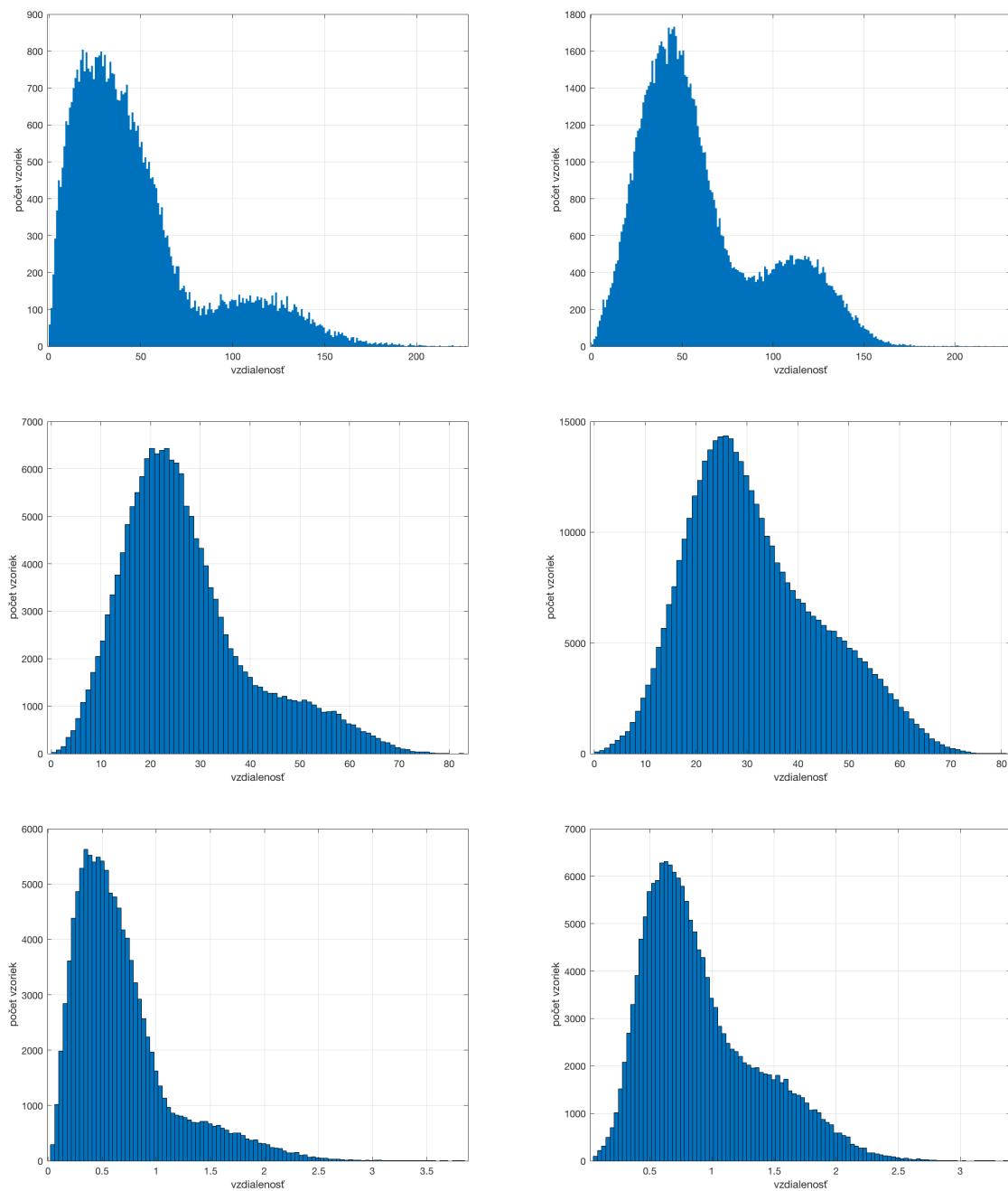
Názov datasetu	Priemerná <i>vzdialenosť</i> príznakov	Priemerná <i>vzdialenosť</i> inliers	Experimentálne určená najlepšia hodnota prahovej hodnoty
ORB 200	27,1566	21,0416	35
ORB 500	26,996	21,1615	35
ORB 2000	26,9939	21,1645	35
ORB CV default	24,6803	19,7824	35
AKAZE FAST	47,9007	32,7435	70
SURF 400	0,679846	0,50187	1
BRISK	75,4793	53,4294	80

Tabuľka 5 - Hodnota prahu vzdialenosťi príznakov pre redukciu chybných korešpondencií určených porovnávačom príznakov. Prvý stĺpec popisuje detektor príznakov, druhý stĺpec zobrazuje priemernú vzdialenosť všetkých určených korešpondencií vo všetkých snímkach testovaného datasetu. Tretí stĺpec zobrazuje priemernú hodnotu vzdialenosťí príznakov, ktoré boli určené ako správne korešpondencie metódou RANSAC. Posledný stĺpec zobrazuje experimentálne určenú hodnotu prahu orezania.

Výsledky testov vplyvu filtra CLAHE na rozloženie *vzdialenosťí* príznakov zobrazuje Tabuľka 6. Ako vidíme, filter viac-menej zvyšuje iba početnosti príznakov, nie ich štatistické rozloženie.

Histogram vzdialenosí korešpondencí príznakov detektorov AKAZE, ORB a SURF

bez a s filtrom CLAHE



Tabuľka 6 - Histogram vzdialenosí príznakov pre rôzne detektory s filtrom CLAHE a bez neho. Prvý riadok je detektor AKAZE, druhý detektor ORB a tretí detektor SIFT. Ľavý stĺpec zobrazuje histogram vzdialenosí príznakov samotného detektora, pravý stĺpec s použitím filtra CLAHE. Počet stĺpcov v histograme je rovnaký.

11.10 Implementovaná metóda RANSAC

Metóda RANSAC bola použitá pre zvýšenie robustnosti a zrýchlenie výpočtu transformácie medzi dvoma množinami 3D korešpondencií. Táto kapitola popisuje odlišnosti našej implementácie od referenčnej implementácie RANSAC. Vyhodnocuje vplyv všetkých parametrov na efektívnosť riešenia a pre každý parameter zároveň navrhнемe metodiku stanovenia jeho hodnoty.

Implementácia metódy RANSAC v prezentovanom riešení definuje nasledovné vstupné parametre:

- **Koeficient ukončenia/minimálny počet iterácií** – je prahová hodnota pomeru počtu inliers v aktuálnej iterácii ku počtu všetkých vzoriek. Ak je pomer inliers v riešení väčší ako táto hodnota, algoritmus sa ukončí. Hodnotu je možné určiť aj na základe spracovávanej vzorky dát. Tento parameter nahrádza parameter prahovej hodnoty počtu inliers (viz kapitola 8.3 RANSAC) z referenčnej implementácie metódy RANSAC. Koeficient ukončenia je možné určovať aj automaticky z určitého minimálneho počtu iterácií.
- **Maximálny počet iterácií** – je maximálny možný počet iterácií, ktoré môže algoritmus vykonať.
- **Veľkosť bázy** – je veľkosť množiny bodov, ktoré sú v každej iterácii vybrané zo všetkých vzoriek a určíme z nej hľadané parametre modelu. Minimálna hodnota je 5, maximálna je rovná počtu všetkých vzoriek. V praxi sa používajú hodnoty 5,6,7,8.
- **Prahová hodnota vzdialenosť pre inlier** – je maximálna hodnota chyby reprojekcie jednej 3D korešpondencie, pri ktorej bude ešte korešpondencia označená ako inlier. Je vyjadrená euklidovskou vzdialenosťou v priestore.

11.10.1 Koeficient ukončenia a minimálny počet iterácií

Všeobecná schéma metódy RANSAC definuje podmienku predčasného ukončenia algoritmu na základe počtu inliers. Ak je počet inliers v aktuálnej iterácii väčší ako vopred definovaná hodnota, algoritmus sa ukončí. Tento prístup je vhodný v implementáciách, kde je veľkosť vstupného datasetu konštantná. V našom prípade sa však počet detegovaných vzoriek na

obraze v čase mení. Z tohto dôvodu bola pre implementáciu ukončovacej podmienky použitá miesto absolútneho počtu inliers pomer počtu inliers ku počtu všetkých dostupných vzoriek. Parameter tak reprezentuje požadovanú dôveryhodnosť vypočítaného riešenia nezávisle od počtu všetkých vzoriek. Vplyv tohto parametra na rýchlosť behu algoritmu RANSAC bol overený experimentom. Ostatné parametre boli počas experimentov nastavené nasledovne: veľkosť bázy RANSAC – 6, vzdialenosť inliers – 100, detektor – ORB s 500 príznakmi a CLAHE filtrom.

stop %	počet iterácií	% inliers	priemerný čas behu [ms]
15	1,22184	49,2416	0,00232
25	9,14435	52,9732	0,04074
35	21,5802	57,0709	0,15232
45	58,1301	60,3717	0,45296
55	59,5012	60,2791	0,39883
60	371,531	64,0361	3,49127
63,487	131,722	65,2169	0,93155
65	544,203	64,8231	5,34109
75	800,225	65,5870	7,54408
85	881,196	65,8462	8,44172
95	931,758	65,8641	8,49593
	1000	65,8978	10,1278

Tabuľka 7 - Vplyv koeficientu ukončenia na čas behu algoritmu RANSAC. Prvý stĺpec zobrazuje testovanú hodnotu parametra koeficientu ukončenia v percentách, druhý stĺpec zobrazuje dosiahnutý počet iterácií metódy RANSAC predtým, ako sa algoritmus ukončil. Tretí stĺpec obsahuje priemerný percentuálny podiel inliers v konečných riešeniach vypočítaných metódou RANSAC v celom datasete. Posledný stĺpec obsahuje čas behu algoritmu. Posledný riadok zobrazuje merané hodnoty pri behu bez možnosti predčasného ukončenia, s konštantným počtom iterácií definovaným ako 1000. Vyznačený riadok reprezentuje automaticky určenú hodnotu koeficientu z prvých 60 iterácií.

Výsledky experimentov zobrazuje Tabuľka 7. Sledovaný pomer inliers ku všetkým vzorkám narastá s počtom iterácií až po maximálnu hodnotu, ktorá je vlastná datasetu a nie je možné ju ďalej zvýšiť. Táto hodnota v prípade uvedeného experimentu predstavuje cca 65%, pre rôzne datasety sa však hodnota líši (experimenty na rôznych datasetoch vykazovali hodnoty od 55-90%). Zároveň, ak je požadovaný percentuálny podiel inliers vyšší ako priemerná

hodnota podielu inliers vo všetkých vzorkách, potrebný počet iterácií prudko stúpa, a s ním aj čas výpočtu.

Hodnotu tohto koeficientu preto určujeme automaticky počas behu tak, že algoritmus vykoná určitý minimálny počet iterácií, z ktorých je určená priemerná hodnota sledovaného koeficientu a tá je následne použitá ako ukončovacia podmienka. Parameter koeficient ukončenia je potom nahradený parametrom reprezentujúcim minimálny počet iterácií, ktorý musí metóda RANSAC vykonať, aby koeficient určila automaticky.

Hodnota minimálneho počtu iterácií bola stanovená experimentálne na 60 iterácií tak, že bol určený minimálny počet iterácií RANSAC, pri ktorom určenie pohybu kamery ešte nevykazovalo hrubé chyby. Experimenty potvrdili, že stanovená hodnota úspešne skracuje dĺžku behu algoritmu RANSAC nezávisle od datasetu.

11.10.2 Maximálny počet iterácií

Maximálny počet iterácií predstavuje v iteratívnych metódach tradičný spôsob obmedzenia dĺžky behu algoritmu. Pri metóde RANSAC je dôležité vykonať dostatočný počet náhodných výberov zo vstupných dát na to, aby bola vybraná reprezentatívna vzorka datasetu. Zároveň, počet iterácií určuje aj celkovú výpočtovú zložitosť a tým aj čas behu algoritmu.

Pre stanovenie optimálnej hodnoty tohto parametra bola využitá

Tabuľka 7 - Vplyv koeficientu ukončenia na čas behu algoritmu RANSAC. Prvý stĺpec zobrazuje testovanú hodnotu parametra koeficientu ukončenia v percentách, druhý stĺpec zobrazuje dosiahnutý počet iterácií metódy RANSAC predtým, ako sa algoritmus ukončil. Tretí stĺpec obsahuje priemerný percentuálny podiel inliers v konečných riešeniach vypočítaných metódou RANSAC v celom datasete. Posledný stĺpec obsahuje čas behu algoritmu. Posledný riadok zobrazuje merané veličiny pri behu bez možnosti predčasného ukončenia, s konštantným počtom iterácií definovaným ako 1000. Vyznačený riadok reprezentuje automaticky určenú hodnotu koeficientu z prvých 60 iterácií. Pre počet iterácií nad 800 sa už priemerný pomer inliers vo vybraných správnych riešeniach nezvyšuje. Z dôvodu zavedenia určitej miery robustnosti bol teda parameter nastavený na 1000, čo sa ukázalo ako postačujúca hodnota pre všetky spracované datasety.

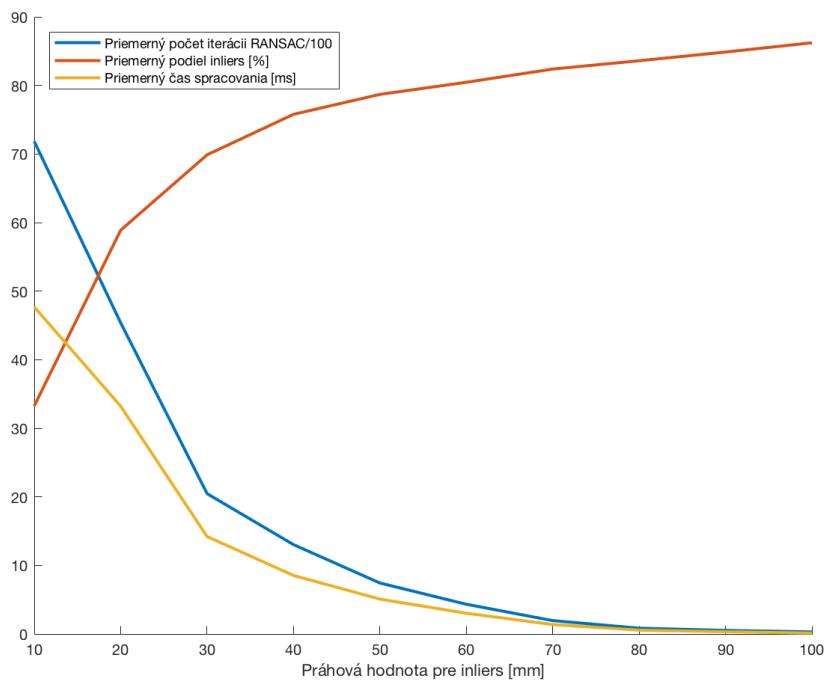
11.10.3 Prahová hodnota vzdialenosť pre identifikáciu inlier

Určenie správneho riešenia prebieha v metóde RANSAC pomocou minimalizácie určenej chyby reprojekcie. Overované sú hypotézy o pohybe kamery určené zo vzoriek vybraných náhodným výberom. Existuje viacero možností výpočtu chyby reprojekcie. Najčastejšie sa používa hodnota euklidovskej vzdialenosť alebo jej druhej mocniny, ktorá znižuje nároky na výpočet riešenia pomocou odstránenia operácie odmocnenia. V praxi je však takto formulovaná chyba nedostatočne robustná voči šumu. Preto metóda RANSAC určuje pre každú vzorku, ktorú verifikuje voči práve spracúvanej hypotéze, zavádzajúc prah chyby pre identifikáciu inlier. Ak je chyba reprojekcie pre bod vyššia ako prahová hodnota, bod bude klasifikovaný ako outlier. Ak je chyba menšia ako prah, bod je považovaný za inlier. Pre každú hypotézu sa následne neporovnáva konkrétna hodnota chyby reprojekcie, ale iba počet identifikovaných inliers. Vybrané je to riešenie, ktoré má najväčší počet inliers. Pomocou nich sú potom určené parametre výsledného modelu.

Veľkosť hodnoty prahu pre určenie inlier má významný vplyv na celkový počet iterácií pre ukončenie algoritmu, aj pre celkový čas behu metódy RANSAC, čo potvrdzujú výsledky experimentu v Tabuľka 8. Všetky merania boli vykonané s detektorm ORB s limitom 500 príznakov, filtrom CLAHE a ukončovacím koeficientom 65%. Maximálny počet iterácií bol nastavený na 7500.

Vzdialosť [mm]	počet iterácií	% inliers	priemerný čas behu
10	7183.65	33.2559	47.6953
20	4538.95	58.8801	33.2149
30	2046.65	69.8751	14.2097
40	1303.02	75.7817	8.53252
50	743.12	78.6906	5.08958
60	433.501	80.4573	3.01931
70	194.866	82.3849	1.37599
80	84.2116	83.5974	0.552165
90	52.0402	84.8657	0.321569
100	29.9472	86.2182	0.12818

Tabuľka 8 - Vplyv prahu určenia inlier na dĺžku spracovania a počet iterácií metódy RANSAC. Prvý stĺpec označuje prah určenia inlier, v druhom stĺpci je počet vykonaných iterácií, v treťom stĺpci priemerný pomer inliers vo všetkých vypočítaných riešeniach pre celý dataset a v poslednom stĺpci priemerný čas behu algoritmu.



Obrázok 59 - Závislosť počtu iterácií, času spracovania a priemerného podielu inliers v metóde RANSAC od prahovej hodnoty pre určenie inlier.

Ako zobrazuje Tabuľka 8, určenie vhodnej hodnoty prahu môže vo významnej mieri ovplyvniť rýchlosť spracovania metódy RANSAC. Príliš nízka hodnota bude pôsobiť príliš reštriktívne a nebude v dostatočnej mieri kompenzovať hĺbkový šum snímača Kinect, ktorý už pri vzdialosti 5 m dosahuje 40 mm (Obrázok 38 - Závislosť hĺbkového rozlíšenia snímača Microsoft Kinect od vzdialosti. Teoretická náhodná chyba je zobrazená červenou, teoretické rozlíšenie modrou.). V takomto prípade bude metóda RANSAC pre určenie polohy využívať iba príznaky veľmi blízko ku kamere, čo veľmi negatívne ovplyvní robustnosť riešenia. **Je preto nevhodné určiť túto prahovú hodnotu na menšiu hodnotu, ako je maximálna chyba určenia polohy snímaného bodu v sledovanom meracom rozsahu.** Príliš vysoká hodnota tohto prahu spôsobí, že budú vyberané aj mnohé riešenia obsahujúce výraznejšie chyby a výsledná kvalita rekonštrukcie bude nižšia. Ako pri ostatných parametroch, je výhodné identifikovanú hodnotu zvýšiť pre zvýšenie robustnosti. Napríklad, v našom experimente bola hodnota stanovená na 70, pretože snímač Kinect na konci svojho rozsahu dosahuje až takúto chybu určenia hĺbky, ale výslednú hodnotu bola určená na 100 pre zvýšenie robustnosti a až desaťnásobné zrýchlenie výpočtu. Ako bolo uvedené, hodnota závisí od presnosti merania hĺbky hĺbkovou kamerou. Závislosť hodnoty od spracúvaného datasetu nebola potvrdená.

11.10.4 Veľkosť bázy

Parameter veľkosť bázy definuje počet vzoriek, ktoré budú náhodne vybrané v každej iterácii a bude podľa nich určená hypotéza o pohybe kamery. Tá bude následne testovaná voči celému datasetu. V literatúre sa nachádzajú zmienky o zvýšenej odolnosti na šum pri použití väčšej veľkosti bázy [28]. Minimálna hodnota pre úspešný výpočet riešenia transformácie medzi dvoma množinami 3D bodov bez iných informácií je 5. Rozhodli sme sa otestovať aj iné najčastejšie používané hodnoty – 6,7 a 8. Pri bežnom spracovaní však prináša vyšší počet bodov bázy iba malé rozdiely v celkovej kvalite riešenia, ako zobrazuje Tabuľka 9.

báza	počet iterácií	% inliers	outliers poloha	outliers orientácia	std poloha	mean poloha	std orientácia	mean orientácia
5	30.335	85.76	11	11	12.2825	0.4258	0.4867	0.0265
6	30.009	86.13	4	13	10.2605	0.4821	0.3728	0.0213
7	30.352	85.96	4	13	9.9516	0.4709	0.3894	0.0137
8	31.957	86.10	4	12	10.0457	0.4407	0.3933	0.0148

Tabuľka 9 - Vplyv veľkosti bázy RANSAC na kvalitu riešenia.

Ako najlepšiu hodnotu pre spracovanie našich datasetov sme určili hodnotu 7, keďže dosiahla najvyššiu kvalitu rekonštrukcie pohybu kamery.

11.11 Model mapy

Výber vhodného spôsobu uloženia mapy je dôležitým prvkom pri návrhu a tvorbe akéhokoľvek mapovacieho systému pre mobilný robot. Presnosť vytvorenej mapy musí korešpondovať s presnosťou, ktorú robot musí dosahovať pri plnení svojich úloh a s presnosťou jeho senzorického systému. Zároveň, komplexnosť a veľkosť mapy majú priamy dopad na výpočtovú zložitosť riešenia úloh lokalizácie, mapovania a navigácie.

Z hľadiska požadovaného výpočtového výkonu pre aktualizáciu a spracovanie mapy a nárokov na úložisko je najvhodnejšou priestorovou reprezentáciou mapy pre navrhnutú aplikáciu je oktálový strom (9.2.2.). Oktálový strom umožňuje extrémne rýchle integrácie nových meraní do existujúcej mapy, vďaka nízkym nárokom na úložisko je vhodný aj pre rozľahlé mapy a umožňuje implementáciu pravdepodobnostného modelovania obsadenosti jednotlivých voxelov. V našej predchádzajúcej práci [72] sme experimentálne preukázali, že tento formát mapy je vhodným formátom pre implementáciu algoritmov navigácie mobilných

robotov pomocou implementácie a overenia navaigačnej metódy A-Star (A*). Navyše, v robotickej komunite sú dostupné vysoko optimalizované implementácie oktálových stromov pod licenciou open-source [106] [73]. Priestorová globálna mapa v tomto formáte však disponuje aj viacerými nevýhodami:

- **Nemožnosť dodatočnej korekcie vytvorenej mapy** – ak mapovací algoritmus zozbiera viacero informácií o svojej trajektórii a dokáže ju späť korigovať, na globálnej mape táto korekcia nebude možná. K tejto situácii dôjde napríklad pri uzavretí slučky.
- **V globálnej mape vo formáte oktálového stromu je komplikované implementovať algoritmus globálnej lokalizácie a algoritmus uzavretia slučky** – po integrácii všetkých zozbieraných meraní do jednej globálnej mapy sa zmení aj charakter priestorovej reprezentácie. Viaceré body zosnímaných mračien budú integrované do spoločných voxelov oktálového stromu a množstvo informácií z mračna bodov sa stratí. To komplikuje implementáciu spoľahlivého algoritmu globálnej lokalizácie.

Pre vyriešenie týchto problémov bola navrhnutá sekundárna mapa vo formáte mračna bodov, organizovaná podľa polohy kamery. Táto mapa umožňuje zhromažďovať spracované príznaky zo všetkých iterácií s najmenšími možnými nárokmi na úložisko pri zachovaní vysokej informačnej bohatosti.

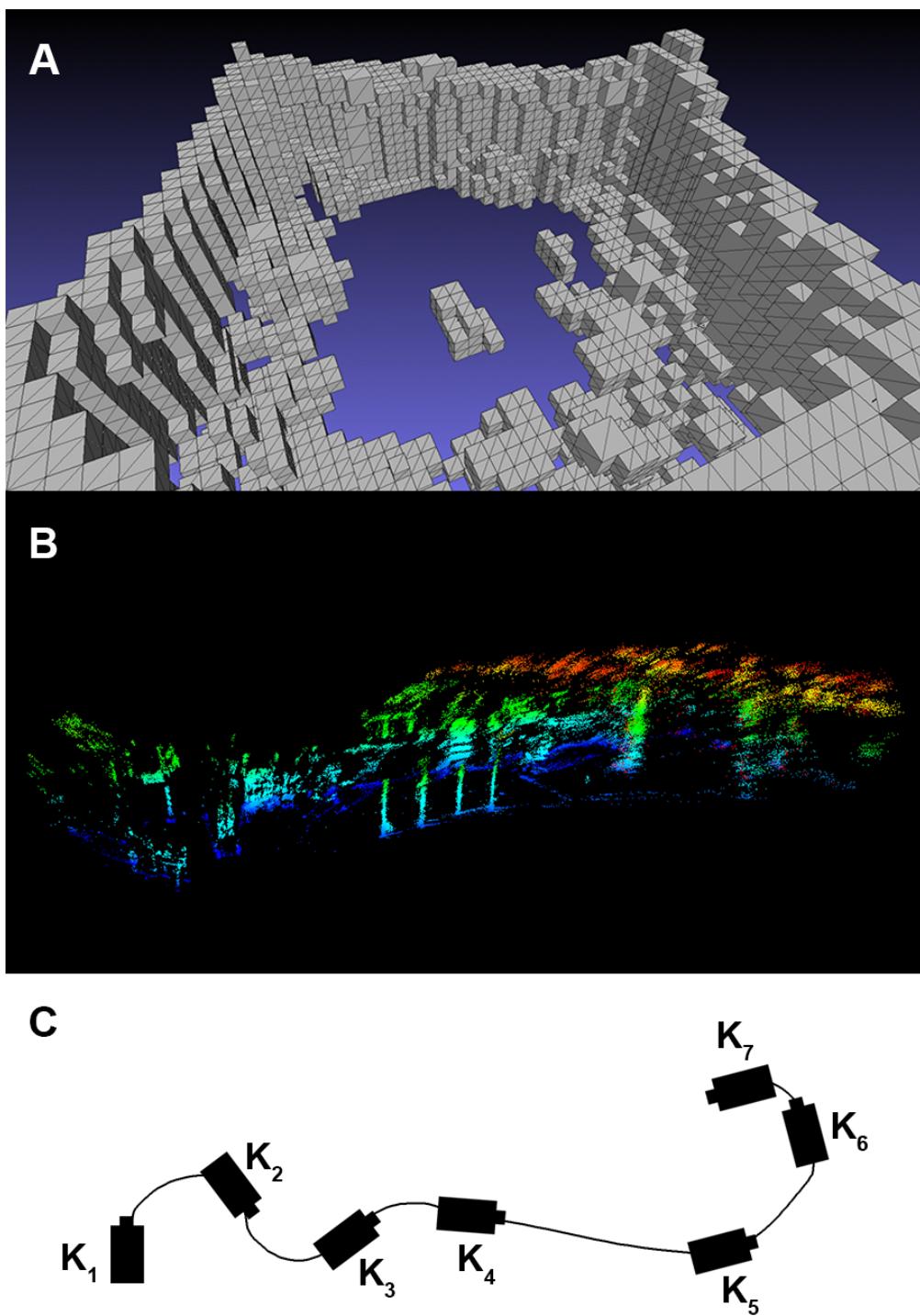
Výsledná mapa používaná v navrhnutom riešení sa tak skladá z troch častí:

1. **Globálna mapa vo formáte oktálového stromu** – pre umožnenie okamžitého spustenia navaigačného algoritmu v ľubovoľnom časovom okamihu je použitá globálna mapa, do ktorej sú priebežne zlučované všetky merania počas mapovania. Mapa je začažená narastajúcou globálnou chybou, ale je možné ju veľmi rýchlo vytvárať aj aktualizovať. Pre čiastočnú kompenzáciu dopadov akumulovanej chyby je použitý oktálový strom, ktorý bude uchovávať pravdepodobnostné rozloženie pre každý voxel. Rozlíšenie oktálového stromu bude nastavené podľa požadovanej presnosti mapy. Požadovaný úložný priestor je oproti reprezentácii vo formáte mračna bodov minimálny, ako sme dokázali v [72].
2. **Globálna mapa orientačných bodov vo formáte mračna bodov, vytvoreného z príznakov z 2D obrazu.** Počas mapovania je vytváraná aj sekundárna mapa, ktorá akumuluje 3D body, korešpondujúce s 2D príznakmi. Tieto body sa ukladajú vo formáte mračna bodov, čo v budúcnosti umožní použitie štandardných algoritmov pre

spracovanie mračien bodov pre riešenie problémov robotického mapovania. Príkladom takého použitia môže byť použitie metódy 4PCS [49] pre globálnu identifikáciu predtým navštíveného segmentu mapy a teda **detekciu slučky (loop detection)** alebo implementáciu **globálnej lokalizácie**. Ku každému bodu z mračna bodov je zároveň uložený aj jeho posledný detegovaný deskriptor, čo umožňuje implementáciu algoritmov uzavretia slučky využívajúcich príznaky [107] [108].

3. **Vektor transformácie pohybu kamery** – tento vektor v čase akumuluje polohy kamery v globálnych súradničach. Globálne súradnice sú sice zaťažené akumulovanou chybou, ale táto môže byť jednoducho odstránená odpočítaním predchádzajúcej globálnej polohy, čím určíme relatívnu transformáciu medzi týmito dvoma polohami. Výhodou oproti použitiu relatívnych transformácií je, že pri nich je potrebné pre získanie globálnej polohy vykonať integráciu predchádzajúcich vzoriek, čo môže byť, najmä pri veľkom počte snímok, výpočtovo náročné. Pre každú polohu kamery tento vektor zároveň uchováva informácie o tom, ktoré body z globálneho mračna bodov boli z tejto polohy zosnímané, čo umožňuje spätnú aktualizáciu alebo korekciu tejto globálnej mapy pomocou algoritmov lokálnej alebo globálnej optimalizácie grafu [53].

Z hľadiska dátových štruktúr sú všetky 3 časti mapy iba jednoduchými akumulátormi. Reprezentáciu oktálového stromu je možné jednoducho priamo aktualizovať pomocou 3D mračien bodov. Mračno bodov príznakov sa aktualizuje pridaním mračna príznakov z aktuálne spracovaného snímku. Podobne jednoduchý je aj spôsob pridania novej pozície kamery do mapy. Mapu je tak možné generovať vysokou rýchlosťou, kde jedinou nutnou operáciou pre pridanie snímku do celej mapy je transformácia všetkých jeho lokálnych 3D súradníc do globálneho súradnicového priestoru. Experimentálne nameraný čas pridania nového snímku do takto navrhutej existujúcej mapy bol v priemere 1 ms. Jednotlivé časti globálnej mapy zobrazuje Obrázok 60 - 3 zložky mapovej reprezentácie používanej v našom riešení. Obrázok A zobrazuje globálnu mapu vo formáte oktálového stromu, obrázok B zobrazuje globálnu mapu príznakov, obrázok C reprezentuje vektor polôh kamery v globálnych súradničach. Pre každú polohu kamery K_x z obrázku C mapa uchováva prislúchajúce 2D príznaky, ich 3D súradnice a deskriptory.



Obrázok 60 - 3 zložky mapovej reprezentácie používanej v našom riešení. Obrázok A zobrazuje globálnu mapu vo formáte oktálového stromu, obrázok B zobrazuje globálnu mapu príznakov, obrázok C reprezentuje vektor polôh kamery v globálnych súradničiach. Pre každú polohu kamery K_x z obrázku C mapa uchováva prislúchajúce 2D príznaky, ich 3D súradnice a deskriptory.

11.12 Extraktia lokálneho modelu

V riešení RGBD-6D-SLAM [30], je poloha aktuálne určovaného snímku určovaná voči podmožine predchádzajúcich snímok. Následne sú vypočítané transformácie lokálne optimalizované [53], čo výrazne redukuje drift, teda nárast globálnej chyby určenia pohybu kamery. Zároveň, tento postup výrazne zvyšuje lokálnu stabilitu určenia polohy. Ak kamera sleduje segment scény bez pohybu, chyba rastie veľmi pomaly. Porovnávanie príznakov a určovanie transformácie pohybu kamery je však výpočtovo náročná operácia, ktorej výpočtové nároky rastú s počtom spracovávaných snímok, a na malom mobilnom robote nemusí byť možné ju implementovať v reálnom čase, ak aktuálnu snímku porovnávame viacnásobne.

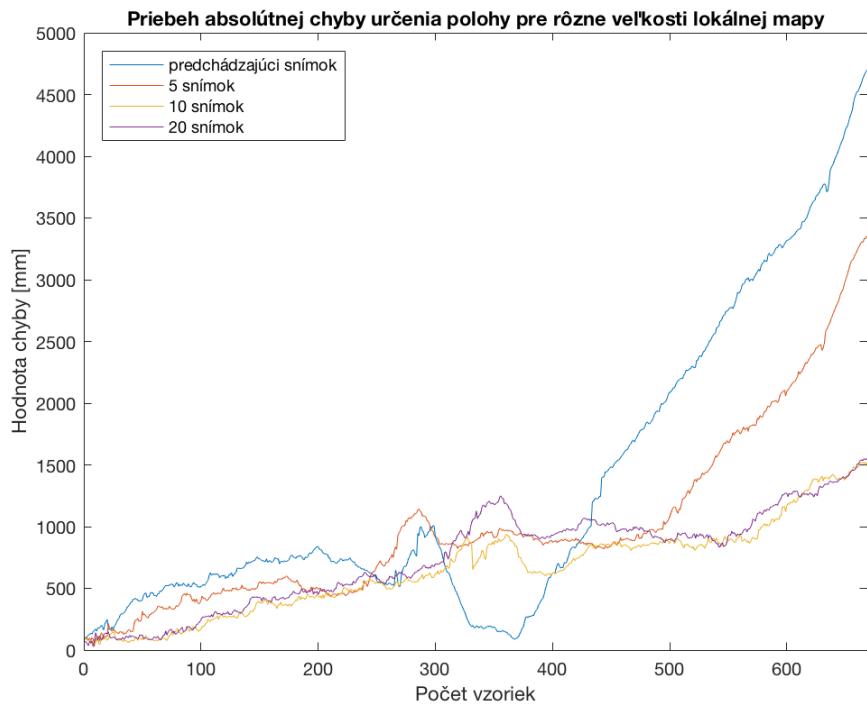
Ak je k dispozícii mapa, z ktorej je možné späť získať príznaky a využiť ich pre ďalšie určenie polohy, je výhodnou stratégiou vyberať príznaky, ktoré majú dobrú opakovateľnosť v rámci aktuálne spracovávaného prostredia. Preto je výhodné evidovať, koľkokrát bol jednotlivý príznak detegovaný a úspešne porovnaný v minulosti.

V riešení publikovanom v knihe Hacking the Kinect [109] sa preto pre určenie aktuálnej polohy používa miesto množiny príznakov z predchádzajúceho snímku množina príznakov a ich globálnych 3D súradníc, ktoré boli aktualizované počas posledných piatich iterácií algoritmu. Algoritmus tak vytvára lokálnu mapu príznakov, vďaka ktorej zvyšuje robustnosť zvyšovaním počtu referenčných príznakov oproti samotnému predchádzajúcemu snímku. Týmto postup bol vzorom aj pri návrhu nášho riešenia.

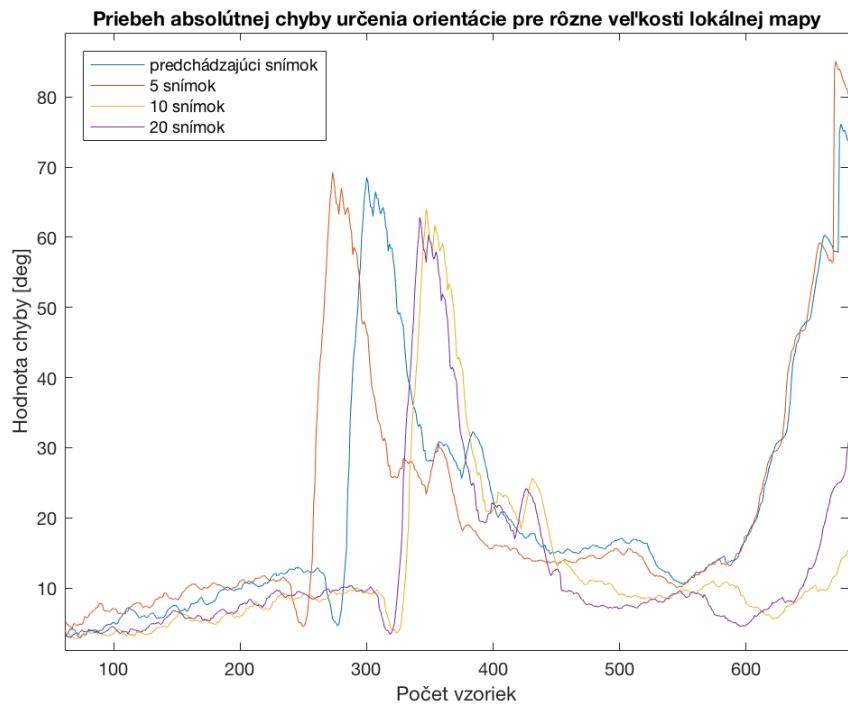
V navrhnutom riešení pre určenie referenčnej množiny príznakov, ktorá bude porovnávaná s aktuálne spracovávanou snímkou, sa použije **lokálny model mapy**. Lokálny, pretože je generovaný z globálnej mapy zo sledu n snímok, ktoré boli nasnímané bezprostredne pred aktuálnym snímkom. Tento proces nazývame **extrakciou lokálneho modelu mapy**. Týmto spôsobom je možné kompenzovať situácie, v ktorých je aktuálne spracúvaný snímok poškodený alebo neobsahuje dostatočné množstvo príznakov, ale predchádzajúce pozície sú pre určenie pohybu kamery vhodné. Nevýhodou tohto prístupu je, že jednoduchým akumulovaním všetkých spracovaných príznakov z predchádzajúcich snímok sa zvyšuje počet vzoriek, ktoré vstupujú do porovnávania a metódy RANSAC, a tým sa predlžuje výpočtový čas. Ďalšou nevýhodou je, že sa zanedbáva fakt, že príznaky z viacerých

snímok boli detegované viacnásobne. To znamená, že v lokálnom modeli mapy sa môže každý najlepší príznak nachádzať viackrát, len s inou priestorovou polohou. Preto je v našom riešení priestorová poloha viacnásobne detegovaných príznakov nahradená strednou hodnotou ich polohy, čo je realizované pomocou určenia každej súradnice ako strednej hodnoty polohy príznaku v danej osi. **Tento postup výrazne redukuje počet spracúvaných príznakov, v priemere o 45%, čo výrazne znižuje čas potrebný pre porovnanie príznakov aktuálne spracovávaného snímku a referenčnej lokálnej mapy.**

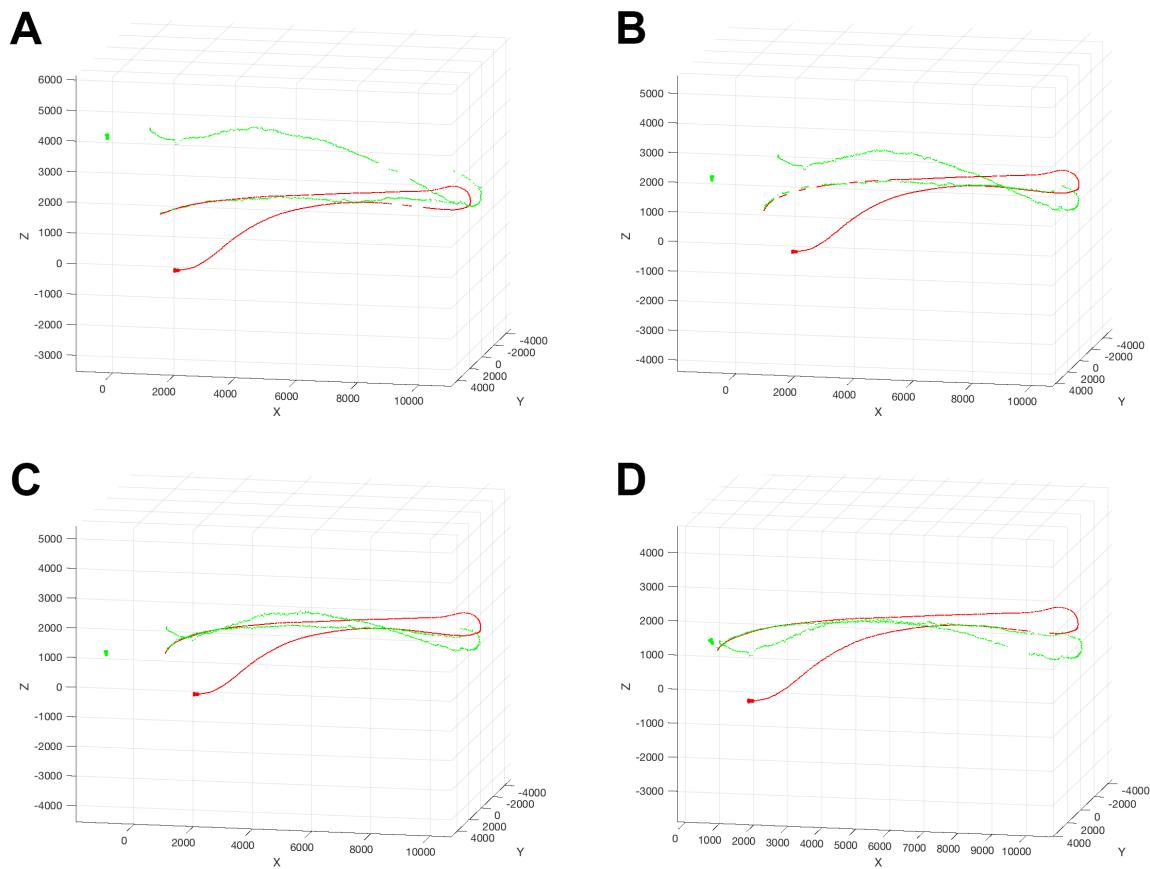
Experimentálne overenie prinieslo predpokladané výsledky. Pomocou použitia viacerých predchádzajúcich snímok pre výpočet správnej transformácie pohybu kamery dochádza k úspešnému potlačeniu driftu a teda zníženiu tempa rastu absolútnej chyby polohy. Navyše, algoritmus je schopný sa lepšie vysporiadať so skenovanými oblasťami, kde je krátkodobo snímaná nevhodná scéna, čo zvyšuje celkovú konzistentnosť mapy a limituje výskyt extrémnych hodnôt chyby. Závislosť absolútnej chyby určenia polohy a orientácie od veľkosti lokálneho mapového modelu zobrazujú obrázky Obrázok 61 a Obrázok 62. Obrázok 63 zobrazuje vplyv veľkosti mapového modelu na výslednú zrekonštruovanú trajektóriu v jednom z našich experimentov. Ako vidíme, použitie už 5 snímok výrazne znižuje tempo rastu globálnej chyby. Použitie 10 snímok redukuje tempo rastu chyby ešte výraznejšie, ale zvýšenie počtu na 20 snímok neprinieslo už žiadne zlepšenie. Chyby orientácie tento prístup ovplyvňuje v menšej miere ako chyby polohy, z dôvodu nízkej miery vzájomného prekrytia lokálneho mapového modelu a aktuálne spracovávaného snímku. V rámci prezentovaného riešenia je prípustné, pre dodržanie výstupnej frekvencie 30 Hz, použiť sekvenciu maximálne posledných 35 snímok, z ktorých sa vyberie každá druhá, teda 17 snímok.



Obrázok 61 - Priebeh absolútnej chyby určenia polohy v závislosti od veľkosti lokálnej mapy.



Obrázok 62 - Priebeh absolútnej chyby orientácie v závislosti od veľkosti lokálneho modelu mapy



Obrázok 63 - Porovnanie zrekonštruovaných trajektórií pohybu kamery pri rôznych veľkostiach lokálneho modelu mapy. Červená trajektória reprezentuje gyro odometriu podvozku Kobuki, zelená trajektória predstavuje trajektóriu určenú pomocou vizuálnej odometrie. A.) Iba predchádzajúci snímok, B.) predchádzajúcich 5 snímok, C.) predchádzajúcich 10 snímok, D.) predchádzajúcich 20 snímok.

12 Záver

Táto dizertačná práca čitateľa uviedla do problematiky robotického mapovania pomocou relatívne nového typu snímača – hĺbkovej kamery. Práca poskytuje komplexný taxonomický prehľad metód robotického mapovania a lokalizácie a ich pridružených problémov. Boli predstavené priestorové mapové reprezentácie a zhodnotená ich vhodnosť pre riešenie štandardných úloh robotiky. V práci boli predstavené a vzájomne porovnané hlavné postupy, ktoré môžu byť použité riešenie problému vizuálnej odometrie pomocou hĺbkovej kamery. Dôraz bol kladený na príznakové metódy, ktoré boli použité aj v prezentovanom riešení.

Vybrané dostupné riešenia boli analyzované a boli identifikované problémy, ktorými tieto implementácie trpia. Bola skonštruovaná robotická platforma po vzore robota TurtleBot 2, ktorá bola použitá pre záznam referenčných trajektórii. Následne bola navrhnutá metodika vyhodnotenia kvality rekonštrukcie pohybu kamery (algoritmu vizuálnej odometrie) voči takto získanej referencii. Táto metodika bola použitá pre vyhodnotenie vlastností navrhovaného riešenia.

Navrhnuté riešenie umožňuje beh algoritmu vizuálnej odometrie v reálnom čase, teda s výstupnou frekvenciou rovnou frekvencii snímača, čiže ~ 30 Hz. Toto bolo umožnené pomocou implementácie paralelizmu na výpočtovej jednotke CPU a viacerým mechanizmom, ktoré redukujú množstvo spracovávaných dát v jednotlivých krokoch algoritmu na minimálnu relevantnú vzorku. Pre dosahnutie maximálnej rýchlosťi spracovania bol na základe experimentov vybraný detektor a deskriptor ORB. Pre zvýšenie robustnosti voči zmenám osvetlenia a nedostatočnej štruktúre scény bol využitý filter CLAHE a jeho dopad na ovplyvnené časti riešenia bol dôkladne analyzovaný. V rámci riešenia bola navrhnutá modifikovaná schéma metódy RANSAC, pre ktorú boli popísané postupy stanovenia všetkých parametrov na optimálne hodnoty z hľadiska rýchlosťi spracovania. Navrhnutá mapová reprezentácia sa skladá z viacerých individuálnych máp, ktoré umožňujú implementáciu ďalších metód pre vytvorenie globálne konzistentného SLAM riešenia v budúcnosti. Pre zníženie tempa rastu globálnej chyby bola implementovaná metóda extrakcie lokálnej mapy, ktorá je použitá ako referencia pre určenie transformácie voči aktuálne spracúvanému snímku. Riešenie je tak schopné zotaviť sa po extrémnych náhodných chybách určenia polohy.

Riešenie funguje vo všetkých šiestich stupňoch voľnosti (6 DoF), čo umožňuje jeho nasadenie ako na pozemných, tak aj na lietajúcich robotoch. Riešenie je implementované v jazyku C++ s využitím multiplatformných knižníc, čo umožňuje jeho beh pod operačnými systémami Windows aj platformách na architektúre Unix. Vytvorený systém je tak možné použiť ako riešenie lokalizácie a mapovania, alebo doplnok existujúcich metód na širokom spektri mobilných robotických platforiem. Uvádzané postupy a teoretický prehľad je možné využiť aj pri riešení problémov 3D rekonštrukcie a 3D skenovania. Zdrojové kódy implementácie sú voľne dostupné v [110]. Vytvorené riešenie bolo implementované a jeho vlastnosti overené na reálnej robotickej platforme.

12.1 Prínosy práce

Za prínosy práce je možné považovať:

1. Návrh, implementácia a overenie algoritmu vizuálnej odometrie a mapovania s nízkou výpočtovou náročnosťou vhodného pre malý mobilný robot.
2. Popis a experimentálne vyhodnotenie schémy paraleлизácie výpočtu príznakovej vizuálnej odometrie pomocou viacerých výpočtových jadier CPU.
3. Návrh použitia a vyhodnotenie dopadu filtra CLAHE pre kompenzáciu nevhodného osvetlenia scény. S použitím tohto filtra v kontexte vizuálnej odometrie sa autor doposiaľ nestretol.
4. Návrh a implementácia mechanizmu tvorby a extrakcie lokálneho modelu mapy za účelom zníženia tempa rastu globálnej chyby.
5. Overenie efektívnosti viacerých detektorov príznakov na reálnych datasetoch. Návrh filtrácie porovnaných príznakov za účelom zvýšenia kvality riešenia pre detektory a deskriptory ORB a AKAZE.
6. Návrh metodiky vyhodnotenia kvality riešenia vizuálnej odometrie voči referenčnej trajektórii.
7. Široký teoretický prehľad komplexne spracúvajúci problematiku vizuálnej odometrie a jej aplikácií pre robotické mapovanie. Podľa vedomostí autora sa jednu z prvých publikáciu tohto typu v SR.

12.2 Plány do budúcna

Vzhľadom na aktuálnosť problematiky vizuálneho SLAM pomocou hĺbkovej kamery je vytvorená implementácia predurčená pre ďalší vývoj v tejto oblasti. V čase dokončenia práce je na pracovisku k dispozícii päť robotických podvozkov Kobuki, ktoré autor prostredníctvom **Programu pre podporu mladých výskumníkov STU** v rámci projektu **Platforma pre výskum multiagentových systémov mobilných robotov** dovybavil nadstavbami TurtleBot 2 a ktoré je možné použiť pre ďalší výskum. Navyše, momentálne najpokročilejšia mobilná robotická platforma na pracovisku, robot MRVK je už aktuálne vybavený hĺbkovou kamerou - snímačom Kinect 2. Z tohto dôvodu autor plánuje vyčistenie a zdokumentovanie zdrojového kódu vytvoreného riešenia a jeho integráciu do platformy Robotického Operačného Systému, ROS. Situácia ohľadom ROS sa totiž počas práce zmenila. Na začiatku prác bol ROS jednou z možných alternatív robotických softvérových frameworkov s limitovaným počtom podporovaných algoritmov. Na konci prác je však situácia odlišná – ROS sa stal de facto štandardom pre publikáciu nových robotických riešení.

Ďalším plánom je implementácia prístupov pravdepodobnostnej robotiky za účelom zvýšenia kvality výslednej mapy a rozšírení a spresnení metód lokalizácie. Na tento výskum sa v súčasnej dobe na pracovisku snažia získať prostriedky. Vytvorené riešenie predstavuje hotový základ, na ktorom je možné tento výskum realizovať. Predpokladaná je implementácia algoritmov typu particle filter.

Autor by prácu rád ďalej kolaboratívne rozširoval formou diplomových a bakalárskych projektov. Úlohy, ktoré navrhuje riešiť sú napríklad:

- Implementácia a komparatívna analýza metód lokálnej optimalizácie grafu.
- Riešenie problému uzavretia slučky pomocou globálnych metód pre registráciu mračien bodov.
- Implementácia globálnej optimalizácie grafu a pospracovanie (postprocessing) vytvorennej mapy.
- Využitie mapy vo formáte oktálového stromu pre vytvorenie rozhrania pre teleriadenie.
- Vytvorenie nového typu detektora príznakov postaveného na hĺbkových dátach.
- Implementácia globálnej lokalizácie v mape príznakov a v oktálovom strome.

12.3 Diskusia ku tézam práce

Tézy schválené komisiou na obhajobe minimovej práce boli nasledovné:

1. Navrhnuť nový vizuálny SLAM algoritmus využívajúci RGBD dátá pre malý mobilný robot s čo možno najnižšími nárokmi na hardvérovú konfiguráciu robota.
2. Navrhnuť mapové reprezentácie a spôsob prenosu údajov medzi viacerými robotmi, ktoré by umožňovali ľahký prenos údajov z mapy cez sieť a jej vhodnú distribúciu medzi viaceré roboty.
3. Vytvoriť stratégie multiagentového mapovania priestoru viacerými robotmi a spájania týchto čiastkových máp z viacerých robotov do jednej spoločnej globálnej mapy.
4. Všetky navrhnuté postupy a algoritmy implementovať do reálnych zariadení a verifikovať ich z hľadiska požadovaných vlastností.

V našej práci sme sa plne sústredili na tézu číslo 1, ktorú sme overili na reálnom systéme podľa tézy 4. Odpoveď na otázku načrtnutú v téze 2 zodpovedá kapitola práce Priestorové mapové reprezentácie. Najvhodnejšou reprezentáciou mapy pre jej presun cez sieť je podľa autora oktálový strom. Ten predstavuje formát mapy, ktorý je možné ľahko prenášať cez sieť a to aj progresívne, teda od nízkeho po vysoké rozlíšenie. V druhom rade, jedná sa o najkompaktniešiu reprezentáciu z hľadiska nárokov na úložisko. Navyše, elementárne operácie nad touto mapou sú vysoko efektívne a vo voxeloch je možné ukladať aj dodatočné informácie ako farbu alebo pravdepodobnostné rozloženie merania.

Problematika načrtnutá v téze 3 nebola v čase začiatku práce robotickou komunitou aktívne riešená, snaha o prehľad literatúry priniesla zanedbateľné výsledky. Autor preto identifikoval analogickú problematiku, ktorá v minulosti riešená byť musela. Jedná sa o prieskum územia z hľadiska vojenského výskumu. Na osobných stretnutiach s reprezentantami Akadémie ozbrojených síl gen. M. R. Štefánika bolo autorovi práce potvrdené, že danú problematiku vojenský výskum rieši, avšak žiadne dostupné materiály mu nemôžu byť sprístupnené. Z tohto dôvodu sa autor tejto téze už ďalej nevenoval. Terajšia situácia je však odlišná, téma kolaboratívneho preskúmavania prostredia viacerými robotmi sa objavuje v aktuálnych vedeckých konferenciách, čo predpovedá jej aktívne riešenie.

13 Použitá literatúra

- [1] „Unbounded innovation with digitalization: A case of digital camera,” rev. *Annual Meeting of the Academy of Management*, 2010.
- [2] R. a. H. J. Y. Shrestha, „Multispectral imaging using LED illumination and an RGB camera,” rev. *Color and Imaging Conference*, Society for Imaging Science and Technology, 2013, pp. 8-13.
- [3] G. a. F. B. Lu, „Medical hyperspectral imaging: a review,” *Journal of Biomedical Optics*, zv. 19, %1. vyd.1, pp. 1083-3668, 2014.
- [4] D. a. S. G. Manolakis, „Detection algorithms for hyperspectral imaging applications,” *IEEE signal processing magazine*, zv. 19, %1. vyd.1, pp. 29-43, 2003.
- [5] C.-I. Chang, Hyperspectral imaging: techniques for spectral detection and classification, zv. 1, Springer Science & Business Media, 2003.
- [6] S. M, „The information that drivers use: is it indeed 90% visual?,” *Perception*, zv. 25, %1. vyd.í, pp. 1081-1089, 1996.
- [7] R. a. N. I. R. a. S. D. Siegwart, Introduction to autonomous mobile robots, MIT press, 2011.
- [8] M. a. J.-F. A. a. P.-V. R. a. L.-B. A. a. J.-M. G. Dominguez-Morales, „Stereo matching: From the basis to neuromorphic engineering,” rev. *Current Advancements in Stereo Vision*, InTech, 2012.
- [9] P. Fua, „A parallel stereo algorithm that produces dense depth maps and preserves image features,” *Machine vision and applications*, zv. 6, %1. vyd.1, pp. 35--49, 1993.
- [10] „Bumblebee XB3 1394b,” FLIR, [Online]. Available: <https://www.ptgrey.com/bumblebee-xb3-1394b-stereo-vision-camera-systems-2>.
- [11] „ZED Depth Sensor 2K Stereo Camera,” Stereolabs, [Online]. Available: <https://www.stereolabs.com/>. [Cit. 2017].
- [12] „DUO3D stereovision cameras,” DUO3D, 2017. [Online]. Available: <https://duo3d.com/>.

- [13] „Ensenso 3D cameras for 3D vision and robot vision applications!,” Imaging Development Systems GmbH, [Online]. Available: <https://en.ids-imaging.com/ensenso-stereo-3d-camera.html>. [Cit. 2017].
- [14] R. a. S. T. Mautz, „Survey of optical indoor positioning systems.,“ rev. *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*, 2011.
- [15] S. L. O. C. R. H. Miles Hansard, Time of Flight Cameras: Principles, Methods, and Applications, Springer, 2012.
- [16] [Online]. Available: <http://www.mesa-imaging.ch/home/>.
- [17] „Kinect 2 sensor review,“ [Online]. Available: <http://www.gottabemobile.com/2014/11/22/should-you-buy-a-kinect-2-sensor-for-your-xbox-one/>.
- [18] S. A. M. M. A. Y. Freedman Barak, „DEPTH MAPPING USING PROJECTED PATTERNS“. United States Patent 20100118123, 13 5 2010.
- [19] A. (. M. U. Shpunt, „DEPTH MAPPING USING MULTI-BEAM ILLUMINATION“. United States Patent 20100020078, 28 1 2010.
- [20] A. Reichinger, „Kinect Pattern Uncovered,“ 3 4 2011. [Online]. Available: <https://azttm.wordpress.com/2011/04/03/kinect-pattern-uncovered/>. [Cit. 16 4 2017].
- [21] R. S. Daniel Scharstein, „High-Accuracy Stereo Depth Maps Using Structured Light,“ rev. *In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2003)*, Madison, WI, June 2003.
- [22] T. J. P. L. A. M. M. H. T. G. a. P. A. M.R. Andersen, „Kinect Depth Sensor Evaluation for Computer Vision Applications,“ Aarhus University, 2012.
- [23] H. a. B. T. Durrant-Whyte, „Simultaneous Localisation and Mapping (SLAM): Part I The Essential Algorithms,“ *IEEE robotics & automation magazine*, zv. 13, %1. vyd.2, pp. 99-110, 2006.
- [24] S. Thrun, W. Burgard a D. Fox, PROBABILISTIC ROBOTICS, 2000.
- [25] S. Thrun, „Probabilistic Algorithms in Robotics,“ Pittsburgh, 2000.

- [26] I. R. N. Roland Siegwart, *Introduction to Autonomous Mobile Robots*, MIT Press, 2004.
- [27] H. Moravec, „Obstacle avoidance and navigation in the real world by a seeing robot rover,” Stanford Univ., Stanford, CA, 1980..
- [28] D. a. F. F. Scaramuzza, „Visual odometry [tutorial],“ *IEEE robotics & automation magazine*, zv. 18, %1. vyd.4, pp. 80-92, 2011.
- [29] F. a. S. D. Fraundorfer, „Visual odometry: Part ii: Matching, robustness, optimization, and applications,“ *IEEE Robotics & Automation Magazine*, zv. 19, %1. vyd.2, pp. 78-90, 2012.
- [30] N. E. a. F. E. a. J. H. a. J. S. a. W. Burgard, „Real-time 3D visual SLAM with a hand-held RGB-D camera,“ rev. *Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum*, Vasteras, Sweden, 2011.
- [31] D. K. O. H. D. M. R. N. P. K. J. S. S. H. D. F. A. D. A. F. Shahram Izadi, „KinectFusion: Real- time 3D Reconstruction and Interaction Using a Moving Depth Camera“.
- [32] S. D. Roth, „Ray casting for modeling solids.,“ *Computer graphics and image processing*, zv. 18, %1. vyd.2, pp. 109-144, 1982.
- [33] T. a. K. M. Tuytelaars, „Local invariant feature detectors: a survey.,“ *Foundations and Trends® in Computer Graphics and Vision* 3.3, pp. 177-280, 2008.
- [34] C. a. M. S. Harris, „A combined corner and edge detector.,“ rev. *Alvey vision conference*, 1988.
- [35] E. a. T. D. Rosten, „Machine learning for high-speed corner detection.,“ Berlin Heidelberg, 2006.
- [36] D. G. Lowe, „Distinctive Image Features from Scale Invariant Keypoints,“ *International Journal of Computer Vision*, no. 60(2), pp. 91-100, November 2004.
- [37] H. E. A. T. T. V. G. L. Bay, „Speeded-up robust features (SURF),“ rev. *Computer vision and image understanding*, 110, 2008.
- [38] B. R. R. B. K. K. B. W. Steder, „NARF: 3D range image features for object recognition,“ rev. *Workshop on Defining and Solving Realistic Perception Problems in Personal Robotics at the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

- [39] S. F. T. a. L. D. S. Salti, „A performance evaluation of 3d keypoint detectors,” rev. *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on. IEEE*, 2011.
- [40] S. a. L. A. A. Filipe, „A comparative evaluation of 3D keypoint detectors in a RGB-D object dataset,” rev. *A comparative evaluation of 3D keypoint detectors in a RGB-D object dataset*, Lisbon, Portugal, 2014.
- [41] T. T. L. V. G. H. B. Andreas Ess, „Speeded- Up Robust Features (SURF),” *Comput. Vis. Image Underst.*, no. 110 , pp. 346-359 , June 2008.
- [42] M. e. a. Calonder, „Brief: Binary robust independent elementary features.,“ rev. *Computer Vision–ECCV 2010*, Berlin Heidelberg, 2010.
- [43] M. Calonder, „Robust, High-Speed Interest Point Matching for Real-Time Applications,” 2010 .
- [44] E. e. a. Rublee, „ORB: an efficient alternative to SIFT or SURF.,“ rev. *Computer Vision (ICCV), 2011 IEEE International Conference on. IEEE*, 2011.
- [45] S. M. C. a. R. Y. S. Leutenegger, „BRISK: Binary robust invariant scalable keypoints.,“ rev. *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011.
- [46] M. A. a. B. R. C. Fischler, „Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, zv. 24, %1. vyd.6, pp. 381-395, 1981.
- [47] „RANdom Sample Consensus (RANSAC) in C#,“ 2 6 2010. [Online]. Available: <http://accordvision.blogspot.sk/2010/06/random-sample-consensus-ransac-in-c.html>. [Cit. 9 4 2017].
- [48] „RANSAC-Algorithmus,“ 4 8 2007. [Online]. Available: <https://de.wikipedia.org/wiki/RANSAC-Algorithmus>. [Cit. 9 4 2017].
- [49] D. N. J. M. a. D. C.-O. Aiger, „4-points congruent sets for robust pairwise surface registration.,“ *ACM Transactions on Graphics (TOG)*, zv. 27, %1. vyd.3, 2008.
- [50] B. U. M. N. N. &. I. S. Drost, „Model globally, match locally: Efficient and robust 3D object recognition.,“ rev. *In Computer Vision and Pattern Recognition (CVPR)*, 2010.

- [51] K. S. P. W. Theiler, „AUTOMATIC REGISTRATION OF TERRESTRIAL LASER SCANNER POINT CLOUDS USING NATURAL PLANAR SURFACES,” *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, zv. 3, pp. 173–178, 2012.
- [52] P. J. & M. N. D. Besl, „A method for registration of 3-D shapes,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, %1. vyd.12(2), p. 239–256, 1992.
- [53] Y. & M. G. Chen, „Object modelling by registration of multiple range images.,“ *Image and Vision Computing*, zv. 3, %1. vyd.10, p. 145–155, 1992.
- [54] F. C. R. S. Francois Pomerleau, „A Review of Point Cloud Registration Algorithms for Mobile Robotics,” *Foundations and Trends in Robotics*, %1. vyd.4, p. 1–104, 2015.
- [55] A. E. I. T. B. R. B. Dirk Holz, „Registration with the Point Cloud Library: A Modular Framework for Aligning in 3-D,” *IEEE Robotics & Automation Magazine*, pp. 110 - 124, 12 2015.
- [56] K. R. C. F. U. H. C. Grisetti G., „Hierarchical optimization on manifolds for online 3d and 3d mapping.“
- [57] S. C. G. S. B. W. Grisetti G., „A tree parametrization for efficiently computing maximum likelihood mapd using gradient descent.,“ rev. *In. Proc. of Robotics: Science and Systems RSS*, 2007.
- [58] R. A. D. F. Kaess M., „iSAM: Incremental smoothing and mapping,” *IEEE Trans. on Robotics (TRO)*, pp. 1365-1378, Dec 2008.
- [59] G. G. S. H. K. K. B. W. Kummerle R., „g2o: A general framework for graph optimization,” rev. *In proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011.
- [60] S. Thrun, „Robotic Mapping: A Survey,” Pittsburgh, 2002.
- [61] B. a. K. O. Siciliano, Springer handbook of robotics, Springer, 2016.
- [62] S. a. o. Thrun, „Robotic mapping: A survey,” *Exploring artificial intelligence in the new millennium*, zv. 1, pp. 1-35, 2002.
- [63] S. a. B. W. a. F. D. Thrun, Probabilistic robotics, MIT press, 2005.

- [64] N. a. E. F. a. H. J. a. S. J. a. B. W. Engelhard, „Real-time 3D visual SLAM with a hand-held RGB-D camera,” rev. *Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum, Västerås, Sweden*, 2011.
- [65] B. P. W. a. M. J. C. a. J. N. a. P. N. a. I. D. R. a. J. D. Tardos, „A comparison of loop closing techniques in monocular SLAM,” *Robotics and Autonomous Systems*, zv. 57, pp. 1188-1197, 2009.
- [66] M. A. J. G. L. M. Nicolas Burrus, „Object reconstruction and recognition leveraging and RGB-D camera,” rev. *MVA2011 IAPR Conference on Machine Vision Applications*, Nara, JAPAN, 2011.
- [67] M. K. E. H. X. R. D. F. Peter Henry, „RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments,” *The International Journal of Robotics Research*.
- [68] F. D. M. T. P. H. a. K. M. P. Beno, „3d map reconstruction with sensor kinect: Searching for solution applicable to small mobile robots,” in 23rd International Conference on Robotics in Alpe-Adria-Danube Region,“ Smolenice, 2014.
- [69] A. Elfes, „Occupancy Grids: A Probabilistic Framework for Robot Perception and Navigation,” 1989.
- [70] D. Meagher, „Geometric Modeling Using Octree Encoding,” *Computer Graphics and Image Processing*, zv. 19, pp. 129-147, June 1981.
- [71] K. M. e. a. Wurm, „OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems.,” rev. *roc. of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation*, 2010.
- [72] B. Peter, P. Vladimír, D. Frantiek a D. Martin, „Using Octree Maps and RGBD Cameras to Perform Mapping and A* Navigation,” rev. *2016 International Conference on Intelligent Networking and Collaborative Systems (INCoS)*, Ostrava, Czech Republic, 2016.
- [73] R. B. R. a. S. Cousins, „3D is here: Point Cloud Library (PCL),” rev. *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011.

- [74] R. B. R. H. I. C. A. a. J. B. T. S. Gedikli, „Efficient Organized Point Cloud Segmentation with Connected Components,” rev. *Workshop on Semantic Perception Mapping and Exploration (SPME)*, Karlsruhe, Germany,, 2013.
- [75] A. a. K. B. Dargazany, „Terrain traversability analysis using organized point cloud, superpixel surface normals-based segmentation and PCA-based classification.,“ rev. *Workshop on Field and Assistive Robotics (WFAR)*, Workshop on Field and Assistive Robotics (WFAR), 2014.
- [76] N. B. Z. M. A. S. M. B. Radu Bogdan Rusu, „Towards 3D Object Maps for Autonomous Household Robots,“ *Robotics and Autonomous Systems*, %1. vyd.56.11, pp. 927-941, 2008.
- [77] Z. C. M. N. B. M. D. M. B. Radu Bogdan Rusu, „Towards 3D Point cloud based object maps for household environments,“ *Robotics and Autonomous Systems*, %1. vyd.56, p. 927–941, 2008.
- [78] R. A. N. H. S. P. H. J. K. A. J. D. Renato F. Salas-Moreno, „SLAM++: Simultaneous Localisation and Mapping at the Level of Objects,“ rev. *Computer Vision and Pattern Recognition (CVPR)*, 2013.
- [79] T. Bräunl, Embedded robotics, Mobile Robot Design and Applications with Embedded Systems, Second Edition ed., Springer, 2006.
- [80] W. Garage, „Turtlebot,“ 2011. [Online]. Available: <http://turtlebot.com>. [Cit. 23 Jún 2018].
- [81] Y. Robot, „Kobuki,“ [Online]. Available: <http://kobuki.yujinrobot.com/>. [Cit. 23 Jún 2018].
- [82] Occipital, „3D scanning, augmented reality, and more for mobile devices.,“ [Online]. Available: <https://structure.io>. [Cit. 23 Jún 2018].
- [83] „Intel® RealSense™ Technology - Observe the World in 3D,“ 23 Jún 2018. [Online]. Available: <https://www.intel.com/content/www/us/en/architecture-and-technology/realsense-overview.html>.
- [84] K. Khoshelham a Elberink, „Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications,“ *Sensors*, pp. 1437-1454, 2012.

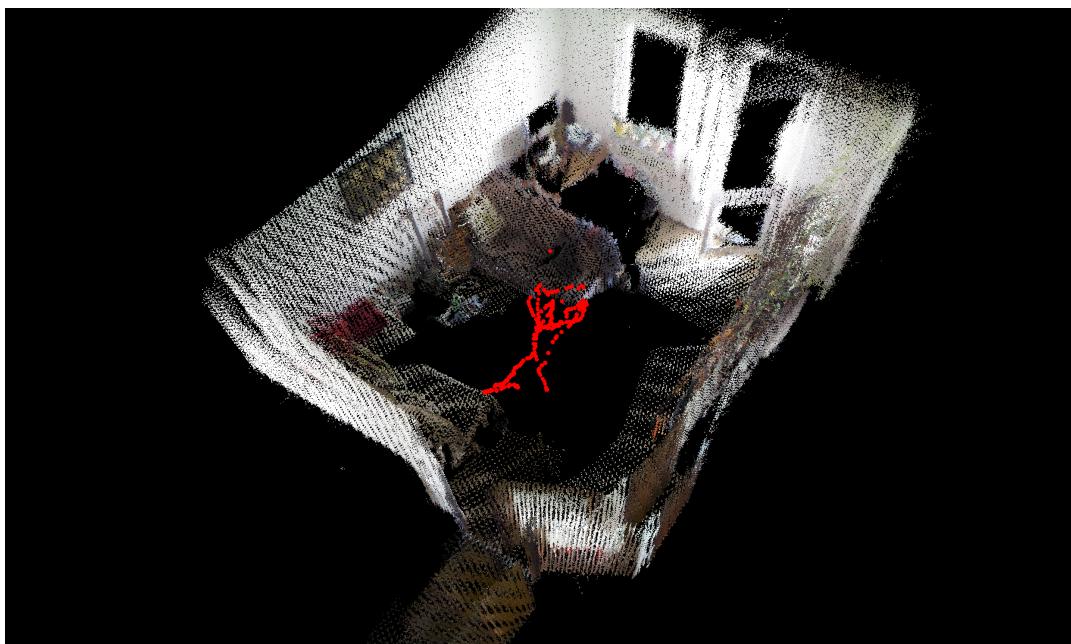
- [85] „OpenNI2,“ [Online]. Available: <https://structure.io/openni>. [Cit. 29 May 2018].
- [86] „OpenKinect,“ [Online]. Available: <https://openkinect.org>. [Cit. 29 May 2018].
- [87] F. J. Andrew Hundt, „handeye_calib_camodocal,“ [Online]. Available: https://github.com/jhu-lcsr/handeye_calib_camodocal. [Cit. 7 Jún 2018].
- [88] „The Robot Operating System (ROS),“ [Online]. Available: <http://www.ros.org/>. [Cit. 28 May 2018].
- [89] B. Peter, „SPRACOVANIE DÁT ZO SNÍMAČA KINECT ZA ÚČELOM TVORBY 3D MAPY PROSTREDIA,“ Bratislava, 2013.
- [90] D. F. T. M. H. P. M. Beňo Peter, „3d map reconstruction with sensor kinect: Searching for solution applicable to small mobile robots,“ rev. In *Robotics in Alpe-Adria-Danube Region (RAAD), 23rd International Conference on IEEE*, Smolenice, Slovakia, 2014.
- [91] B. CURLESS a M. LEVOY, „A volumetric method for building complex models from range images.,“ rev. *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* , 1996.
- [92] M. Pirovano, „Kinfu - an open source implementation of Kinect Fusion + case study: implementing a #D scanner with PCL,“ 2012.
- [93] M. V. Henry Roth, „Moving Volume KInectFusion,“ *BMVC*, pp. 1-11, 2012.
- [94] D. J. R. W. L. a. J. W. W. Ketcham, „Image enhancement techniques for cockpit displays.,“ CULVER CITY CA , 1974.
- [95] M. a. C. P. Petrou, *Image processing: the fundamentals*, John Wiley & Sons, 2010.
- [96] K. Zuiderveld, „Contrast limited adaptive histogram equalization.,“ rev. *Graphics gems*, 1994, pp. 474-485.
- [97] I. C. Consortium, *Image technology colour management - Architecture, profile format, and data structure, Specification ICC.1:2004-10*, 2004.
- [98] G. R. Andrews, *Concurrent programming: principles and practice*, Benjamin/Cummings Publishing Company, 1991.
- [99] T. & M. K. Tuytelaars, „Local invariant feature detectors: a survey,“ *Foundations and Trends® in Computer Graphics and Vision*, zv. 3, %1. vyd.3, p. 177–280, 2007.

- [100] S. A. K. a. Z. S. Tareen, „A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK,” rev. *Computing, Mathematics and Engineering Technologies (iCoMET)*, 2018.
- [101] Ş. İşik, „A comparative evaluation of well-known feature detectors and descriptors,” *International Journal of Applied Mathematics, Electronics and Computers*, zv. 3, %1. vyd.1, pp. 1-6.
- [102] A. e. a. Pieropan, „Feature Descriptors for Tracking by Detection: a Benchmark.,“ 2016.
- [103] E. H. S. a. A. C. L. Daniel R. Roosab, „Comparing ORB and AKAZE for visual odometry of unmanned aerial vehicles,” 2015.
- [104] A. Kostusiak, „The Comparison of Keypoint Detectors and Descriptors for Registration of RGB-D Data,” rev. *Challenges in Automation, Robotics and Measurement Techniques*, Springer International Publishing, 2016, pp. 609-622.
- [105] J. N. A. B. Pablo F. Alcantarilla, „Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces,” rev. *In British Machine Vision Conference (BMVC)*, Bristol, UK, 2013.
- [106] K. M. W. B. S. B. Armin Hornung, „OctoMap: an efficient probabilistic 3D mapping framework based on octrees,” *Autonomous Robots*, zv. 34, %1. vyd.3, p. 189–206, April 2013.
- [107] E. a. T. D. Eade, „Unified Loop Closing and Recovery for Real Time Monocular SLAM,” rev. *BRITISH MACHINE VISION CONFERENCE (BMVC)*, 2008.
- [108] M. a. F. M. Labbé, „Online global loop closure detection for large-scale multi-session graph-based slam,” rev. *Intelligent Robots and Systems (IROS 2014), IEEE/RSJ International Conference on*. IEEE, 2014.
- [109] J. B. N. E. F. D. H. C. & P. M. Kramer, Hacking the kinect, zv. 268, New York, NY: Apress, 2012.
- [110] I. B. Peter, „Kinect visual odometry mapping,” Slovak University of Technology, 30 6 2018. [Online]. Available: <https://github.com/najlepsiwebdesigner/kinect-vo-mapping>.

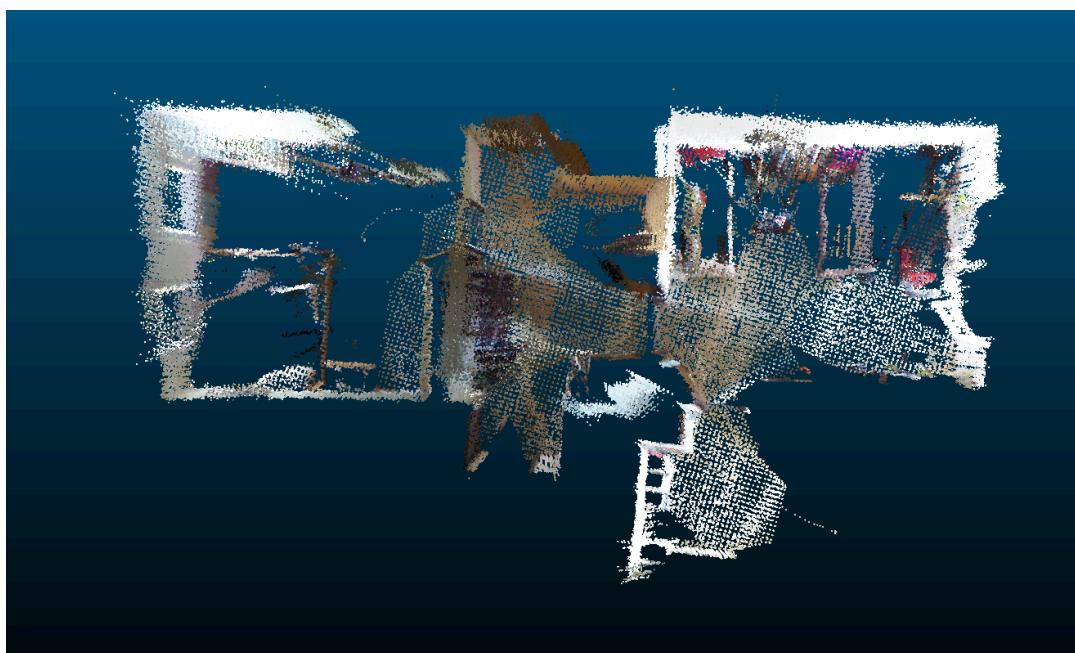
- [111] „Handbook of Research on Computational Simulation and Modeling in Engineering,” rev. *Human Motion Analysis and Simulation Tools: A Survey*, IGI Global, 2016, pp. 359-388.
- [112] S. L. O. C. R. H. Miles Hansard, „Time of Flight Cameras: Principles, Methods, and Applications,” rev. *SpringerBriefs in Computer Science*, 2012, p. 95.
- [113] Q. BREFORT, „Mobile Localization and Interval Analysis,” Angers, 2014.
- [114] L. K. H. S. K. G. B. W. K.-. R. L. T. S. CHOSET H., Principles of Robot Motion: Theory, Algorithms, and Implementations, Cambrige MA: MIT Press, 2005.
- [115] T. Braunl, Embedded Robotics, Springer-Verlag Berlin Heidelberg , 2003.
- [116] C. Evans, „Open SURF implementation in Accord.NET framework,” [Online]. Available: https://code.google.com/p/accord/wiki/SampleApp_SURF.
- [117] T. D. Z. F. A. Bódis-Sz., „A Lane Detection Algorithm based on Wide-Baseline Stereovision for Advanced Driver Assistance,” rev. *7th Conference of the Hungarian Association for Image Processing and Pattern Recognition (KÉPAF'2009)*, Budapest, Hungary, 28-30 Jan. 2009.
- [118] E. L. a. M. S. Konstantinos Derpanis. [Online]. Available: http://www.cse.yorku.ca/~kosta/Generalized_Integral_Image/gii_main.html.
- [119] C. H. a. M. Stephens, „A Combined Corner and Edge Detector,” rev. *In proceedings of the 4th Alvey Vision Conference* , 1988.
- [120] J. m. F. M. P. a. Y. G. S. N. Sinha, „Gpu-based video feature tracking and matching.,“ 2006.
- [121] T. Lindeberg, Scale-space Theory in Computer Vision, Norwell, MA: Kluwer Academic Publishers, 1994.
- [122] M. L. W. a. Y. H. Li, „Image matching based on SIFT features and kd-tree.,“ rev. *2nd International Conference on Computer Engineering and Technology. Vol. 4.*, 2010.
- [123] W. H. J. a. T. Y. Wei, „Image matching for geomorphic measurement based on SIFT and RANSAC methods.,“ rev. *Computer Science and Software Engineering, 2008 International Conference on. Vol. 2.*, 2008.
- [124] V. HÖGMAN, „Building a 3D map from RGB-D sensors,” Stockholm, Sweden, 2012.

- [125] 2015. [Online]. Available: http://docs.opencv.org/trunk/doc/py_tutorials/py_feature2d/py_surf_intro/py_srf_intro.html.
- [126] N. S. P. P. Johannes Bauer, „COMPARING SEVERAL IMPLEMENTATIONS OF TWO RECENTLY PUBLISHED FEATURE DETECTORS,” rev. In *Proc. of the International Conference on Intelligent and Autonomous Systems*, Toulouse, France, 2007.
- [127] L. Hulstaert, „A Beginner's Guide to Object Detection,” DataCamp, 19 April 2018. [Online]. Available: <https://www.datacamp.com/community/tutorials/object-detection-guide>. [Cit. 27 May 2018].
- [128] F. D. M. T. P. H. M. K. Peter Beňo, „3D map reconstruction with sensor Kinect,” rev. *International Conference on Robotics in Alpe-Adria-Danube Region (RAAD), 2014 23rd*, Smolenice, Slovak republic, 2014.

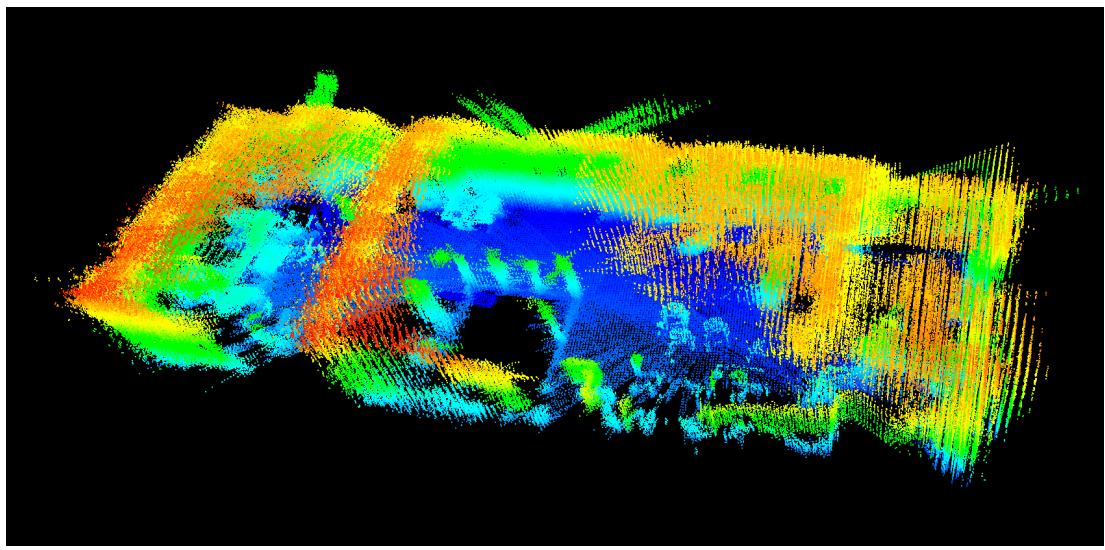
14 Príloha 1 – Mapy prostredí vytvorené pomocou navrhovaného riešenia



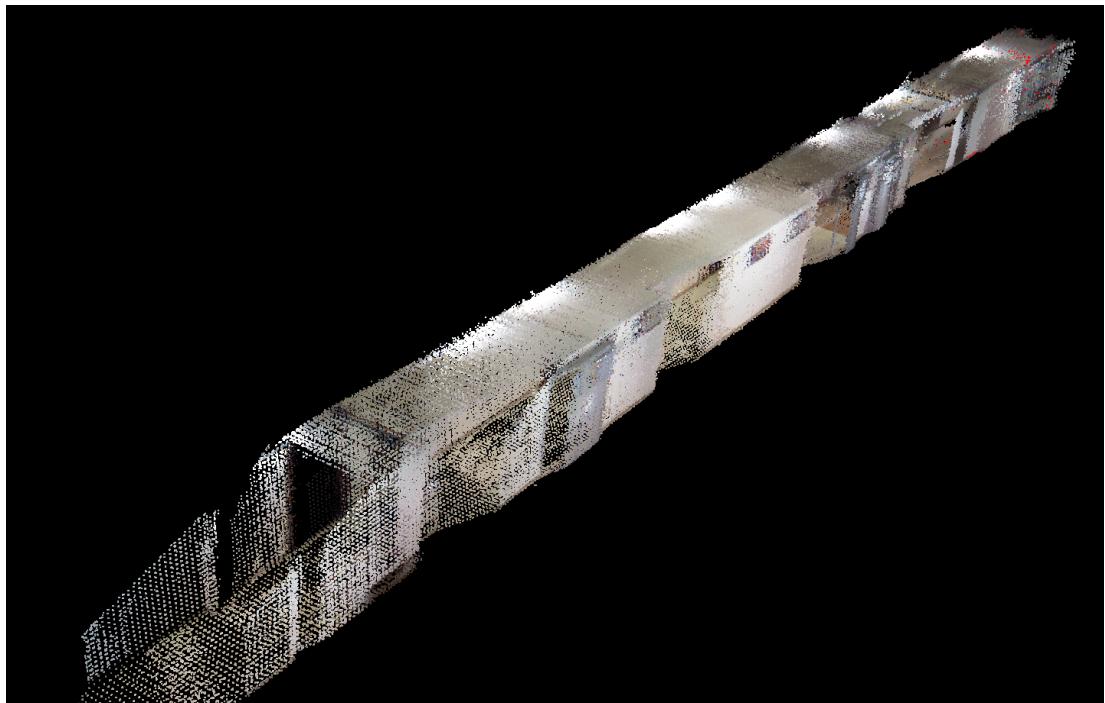
Obrázok 64 - Mapa obývacej izby vytvorená pomocou skenovania z voľnej ruky. Červenou sú vyznačené jednotlivé pozície kamery a v trajektórii pohybu.



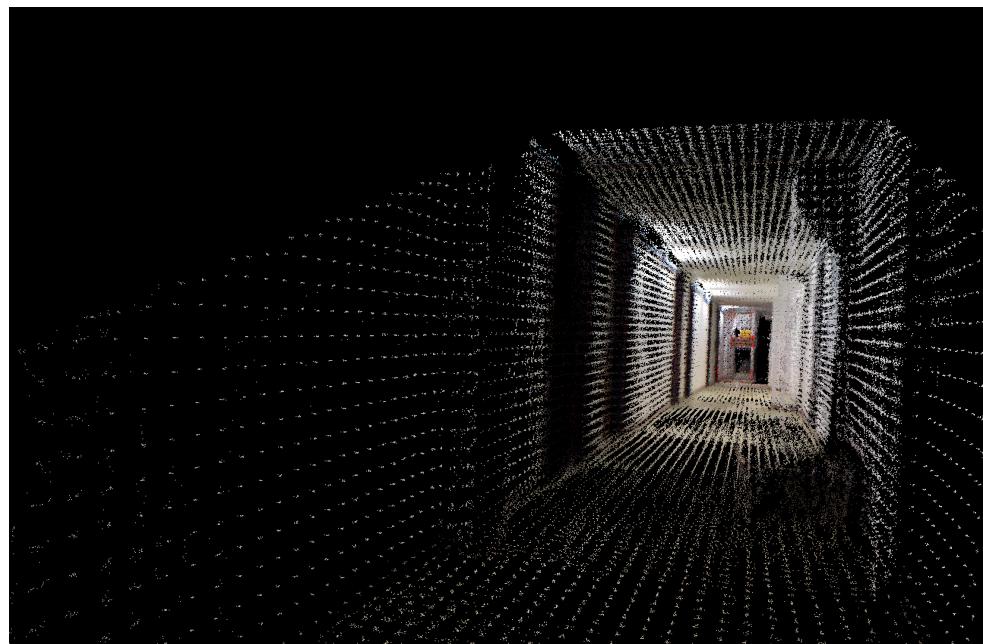
Obrázok 65 - Mapa bytu vytvorená pomocou nášho riešenia jedným rýchlym prechodom cez byt panelákového domu



Obrázok 66 - Mapa priestorov Národného centra robotiky. Miesto textúry bola mapa zafarbená podľa výškovej súradnice.



Obrázok 67 - Mapa chodby pri Národnom centre robotiky (NCR), pohľad zhora



Obrázok 68 - Mapa pri NCR, pohľad zvnútra

15 Príloha 2 – Pseudokód implementovanej metódy RANSAC

```
// inicializácia a definícia parametrov
max_iteracie = 1000
min_iteracie = 60
velkost_bazy = 7
prah_chyby_inliers = 100
pocet_korespondencii = velkost(korespondencie)
stredna_hodnota_chyby_inliers = 0
najlepsi_inliers // inliers s najmenšou hodnotou chybovej funkcie

// iteruj po maximálny počet iterácií
for (k = 0; k < max_iteracie; k++) {
    // vyber náhodné korešpondencie o veľkosti bázy
    nahodne_korespondencie = vyber_nahodne(korespondencie, velkost_bazy)
    // pomocou metódy Singular Value Decomposition určí
    // transformáciu medzi týmito dvoma množinami
    transformacia = urci_transformaciu_svd(nahodne_korespondencie)

    // ##### Chybová funkcia metódy RANSAC #####
    inliers // zásobník práve spracovávaných inliers v chybovej funkcií
    for (korespondencia z korespondencie) {
        chyba = euklidovska_vzdialenosť(korespondencia, transformacia);
        // ak je euklidovská vzdialenosť menšia ako prahová hodnota,
        // ulož index korešpondencie medzi inliers
        if (chyba < prah_chyby_inliers) {
            pridaj_inlier(inliers, korespondencia)
        }
    }

    // ak mala táto transformácia viac inliers ako riešenia spracované
    // doteraz, ulož indexy týchto korešpondencií ako najlepšie
    if (velkost(inliers) > velkost(najlepsi_inliers)) {
        najlepsi_inliers = inliers
    }

    // ak sa našlo sa riešenie s pomerom inliers väčším ako
    // priemerná hodnota krát 1.5 a už prebehlo 50 iterácií, skončí
    if (k > min_iteracie && ((velkost(inliers) * 100) / pocet_korespondencii)
> stredna_hodnota_chyby_inliers)
    {
        ukonci_ransac()
    }

    // priebežne aktualizuj strednú hodnotu v každej iterácii
    stredna_hodnota_chyby_inliers.aktualizuj(najlepsi_inliers)
}

// po ukončení iterácií vypočítame finálnu transformáciu pomocou najlepších
// inliers zo všetkých iterácií
urcena_transformacia_ransac = urci_transformaciu_svd(najlepsi_inliers)
```

16 Príloha 3 – Publikované práce autora

ADC Vedecké práce v zahraničných karentovaných časopisoch

- ADC01 KONIAR, Dušan - HARGAŠ, Libor - LONCOVÁ, Zuzana - DUCHOŇ, František - BEŇO, Peter. Machine vision application in animal trajectory tracking. In *Computer Methods and Programs in Biomedicine*. Vol. 127, (2016), s. 258-272. ISSN 0169-2607. V databáze: CC: 000372521500022.
- ADC02 KONIAR, Dušan - HARGAŠ, Libor - LONCOVÁ, Zuzana - SIMONOVÁ, Anna - DUCHOŇ, František - BEŇO, Peter. Visual system-based object tracking using image segmentation for biomedical applications. In *Electrical Engineering*. Vol. 99, Iss. 4 (2017), s. 1349-1366. ISSN 0948-7921. V databáze: CC: 000415972700022 ; SCOPUS: 2-s2.0-85026909750.

ADE Vedecké práce v ostatných zahraničných časopisoch

- ADE01 DUCHOŇ, František - BUČKA, Peter - SZABOVÁ, Martina - DEKAN, Martin - BEŇO, Peter - TÖLGYESSY, Michal. Image processing of motion for security applications. In *European Scientific Journal*. Vol. 13, No. 27 (2017), s. 44-58. ISSN 1857-7881.
- ADE02 KONIAR, Dušan - HARGAŠ, Libor - LONCOVÁ, Zuzana - DUCHOŇ, František - BEŇO, Peter. Usage of video analysis for tracking of laboratory animals. In *Acta Technica Corviniensis - Bulletin of Engineering [elektronický zdroj]*. Tom. 10, fasc. 2 (2017), online, s. 147-152. ISSN 2067-3809.

ADF Vedecké práce v ostatných domácich časopisoch

- ADF01 DUCHOŇ, František - BEŇO, Peter - BABINEC, Andrej - DEKAN, Martin - DÚBRAVSKÝ, Jozef - JURNICKÝ, Tomáš - MUSIČ, Josip. Use of children's games in robotics. In *Acta Mechatronica [elektronický zdroj]*. Vol. 1, Iss. 4 (2016), online s. 7-11. ISSN 2453-7306.

AFC Publikované príspevky na zahraničných vedeckých konferenciách

- AFC01 BEŇO, Peter - PAVELKA, Vladimír - DUCHOŇ, František - DEKAN, Martin. Using octree maps and RGBD cameras to perform mapping and A* navigation. In *IEEE INCoS 2016 : International conference on intelligent networking and collaborative systems. Ostrava, Czech Republic. 7-9 September 2016*. Danvers : IEEE, 2016, S. 66-72. ISBN 978-1-5090-4123-7. V databáze: IEEE ; WOS: 000386596100013.

AFD Publikované príspevky na domácich vedeckých konferenciách

- AFD01 BEŇO, Peter - DUCHOŇ, František - TÖLGYESSY, Michal - HUBINSKÝ, Peter - KAJAN, Martin. 3D Map Reconstruction with Sensor Kinect. In *RAAD 2014 [elektronický zdroj] : 23rd International Conference on Robotics in Alpe-Adria-Danube Region. Conference Proceedings. September 3-5, 2014, Smolenice Castle, Slovakia*. 1. vyd. Bratislava : Publishing House of Slovak University of Technology, 2014, CD-ROM, [6] p. ISBN 978-80-227-4219-1.

- AFD02 DUCHOŇ, František - BABINEC, Andrej - KAJAN, Martin - BEŇO, Peter - FLOREK, Martin - FICO, Tomáš - JURIŠICA, Ladislav. Path planning with modified A star algorithm for a mobile robot. In *Procedia Engineering [elektronický zdroj] : The 6th International Conference on Modelling of Mechanical and Mechatronic Systems MMaMS 2014, 25-27 November 2014, Vysoké Tatry, Slovakia*. Vol. 96, (2014), p. 59-69. ISSN 1877-7058.
- AFD03 DUCHOŇ, František - VONDRAČEK, Martin - DEKAN, Martin - BABINEC, Andrej - SPIELMANN, Róbert - SZABOVÁ, Martina - MIKULOVÁ, Zuzana - BEŇO, Peter - DÚBRAVSKÝ, Jozef. Homogenous multi-robot system for mapping of unknown environment. In *SAMI 2016 : IEEE 14th international conference Symposium on Applied Machine, Intelligence and Informatics. Herľany, Slovakia, January 21-23, 2016*. 1. vyd. [S. l.] : IEEE, 2016, S. 17-22. ISBN 978-1-4673-8739-2. V databáze: IEEE ; WOS: 000381795100002 ; SCOPUS: 2-s2.0-84964607617.
- AFD04 KAJAN, Martin - ŠOVČÍK, Ján - DUCHOŇ, František - MRAFKO, Leo - FLOREK, Martin - BEŇO, Peter. Sensoric subsystem of Automated Guided Vehicle. In *RAAD 2014 [elektronický zdroj] : 23rd International Conference on Robotics in Alpe-Adria-Danube Region. Conference Proceedings. September 3-5, 2014, Smolenice Castle, Slovakia*. 1. vyd. Bratislava : Publishing House of Slovak University of Technology, 2014, CD-ROM, [6] p. ISBN 978-80-227-4219-1.
- AFD05 KONIAR, Dušan - HARGAŠ, Libor - LONCOVÁ, Zuzana - DUCHOŇ, František - BEŇO, Peter. Laboratory animals tracking in videosequences. In *ELEKTRO 2016 : 11th International conference. Štrbské Pleso, Slovak Republic. May 16-18, 2016*. Žilina : University of Žilina, 2016, S. 537-542. ISBN 978-1-4673-8698-2. V databáze: IEEE ; SCOPUS: 2-s2.0-84981343555 ; WOS: 000386959800108.