Robust and light-weight simultaneous localization and mapping for autonomous vehicles

Wang, Han

2021

Wang, H. (2021). Robust and light-weight simultaneous localization and mapping for autonomous vehicles. Doctoral thesis, Nanyang Technological University, Singapore.
<https://hdl.handle.net/10356/151717>

<https://hdl.handle.net/10356/151717>

<https://doi.org/10.32657/10356/151717>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 19:28:34 SGT

**Robust and Light-weight Simultaneous
Localization and Mapping for Autonomous
Vehicles**

Wang Han

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

2021

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

17th January 2021

.....

Date

wang Han

.....

Wang Han

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

17th January 2021

.....

Date

.....

Prof. Xie Lihua

Authorship Attribution Statement

This thesis contains materials from 6 papers published in the following peer-reviewed journals/conferences in which I am listed as a co-author.

Chapter 4 is submitted as [Han Wang, Chen Wang, Chun-Lin Chen and Lihua Xie, “F-LOAM: Fast Lidar Odometry and Mapping in Large Scale Environment”, 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems \(IROS\), Prague, Czech Republic, 2021.](#)

The contributions of the co-authors are as follows:

- I proposed the idea, formulated the problem, designed the experiments, and prepared the manuscript.
- Dr. Chen Wang assisted in the theoretical analysis.
- Chun-Lin Chen and I conducted the experiment.
- The manuscript was revised by Dr. Chen Wang and Prof. Lihua Xie.

Chapter 5 is published as [Han Wang, Juncheng Li, Maopeng Ran and Lihua Xie, “Fast Loop Closure Detection via Binary Content”, 2019 IEEE International Conference on Control and Automation \(ICCA\), Edinburgh, United Kingdom, 2019, pp. 1563-1568.](#)

The contributions of the co-authors are as follows:

- I proposed the idea, formulated the problem, designed the experiments, and prepared the manuscript.
- Maopeng Ran assisted in the theoretical analysis.
- Juncheng Li and I conducted the experiment.
- The manuscript was revised by Prof. Lihua Xie.

Chapter 6 is published as [Han Wang, Chen Wang and Lihua Xie, “Online Visual Place Recognition via Saliency Re-identification”, 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems \(IROS\), Las Vegas, NV, USA, 2020, pp.5030-5036.](#)

The contributions of the co-authors are as follows:

- I proposed the idea, formulated the problem, designed and conducted the experiments, and prepared the manuscript.
- Dr. Chen Wang assisted in the theoretical analysis.
- The manuscript was revised by Dr. Chen Wang and Prof. Lihua Xie.

Chapter 7 is published as [Han Wang, Chen Wang and Lihua Xie, “Intensity Scan Context: Coding Intensity and Geometry Relations for Loop Closure Detection”, 2020 IEEE International Conference on Robotics and Automation \(ICRA\), Paris, France, 2020, pp. 2095-2101.](#)

The contributions of the co-authors are as follows:

- I proposed the idea, formulated the problem, designed and conducted the experiments, and prepared the manuscript.
- Dr. Chen Wang assisted in the theoretical analysis.
- The manuscript was revised by Dr. Chen Wang and Prof. Lihua Xie.

Chapter 8 is published as [Han Wang, Chen Wang and Lihua Xie, “Intensity-SLAM: Intensity Assisted Localization and Mapping for Large Scale Environment”, IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 1715-1721, 2021 \(the contents of this paper are also selected by ICRA’21 Program Committee for presentation at the conference\).](#)

The contributions of the co-authors are as follows:

- I proposed the idea, formulated the problem, designed and conducted the experiments, and prepared the manuscript.
- Dr. Chen Wang assisted in the theoretical analysis.
- The manuscript was revised by Dr. Chen Wang and Prof. Lihua Xie.

Chapter 9 is published as [Han Wang, Chen Wang and Lihua Xie, “Lightweight 3-D Localization and Mapping for Solid-State LiDAR”, IEEE Robotics and Automation Letters, vol. 6, no. 2, pp. 1801-1807, 2021 \(the contents of this paper are also selected by ICRA’21 Program Committee for presentation at the conference\).](#)

The contributions of the co-authors are as follows:

- I proposed the idea, formulated the problem, designed and conducted the experiments, and prepared the manuscript.
- Dr. Chen Wang assisted in the algorithm analysis.
- The manuscript was revised by Dr. Chen Wang and Prof. Lihua Xie.

17th January 2021
.....

Date

wang Han
.....

Wang Han

Acknowledgements

I wish to express my heartfelt gratitude to my supervisor, Professor Xie Lihua, for offering me this great opportunity to pursue my Ph.D degree at Nanyang Technological University. Professor Xie is undoubtedly one of the best experts one can look for and I am much inspired by his professionalism during my Ph.D period. His dedication and the meticulous attitude towards research have deeply influenced my research career. This thesis would not be possible without his tremendous support.

I would also like to express my sincere gratitude to Professor Wang Han, for his guidance and supervision. Professor Wang has very deep understanding of state-of-the-art works in robotics and computer vision. The discussion with him is always very inspiring and encouraging.

I would also like to express my sincere gratitude to Dr. Wang Chen, who has patiently guided me step by step on research. I would like to thank Dr. Qiu Zhirong, who has helped me improve my capability in theoretical formulation. Special thanks to Dr. Chen Chun-Lin, Dr. Zou Han and Dr. Thien Minh Nyuan Hoang, for guiding me on experiment. Their suggestions on research are very useful in my future study. I would also like to thanks to Ji Tete, Li Juncheng, Cao Muqing and Nguyen Hoang Thien for helping me on experiments. I would like to thank my colleagues, Dr. Yuan Shenghai, Dr. Guo Kexin, Dr. Lyu Yang, Dr. Abdul Hanif bin Zaini, Cao Kun, Yang Jianfei, Fang Xu, Liu Hanxiao, and Wang Dazhuo for their support and encouragement over the years.

Last but not least, I would like to express my deepest gratitude to my family. Without their support and encouragement, this work would not be possible.

Abstract

Simultaneous Localization And Mapping (SLAM) is one of the most fundamental and essential topics in robotics research. SLAM is a task for a robot to perceive the environment and localize itself based on inputs from its on-board sensors. The robot is also supposed to construct a map of the surrounding environment for subsequent task planning and collision avoidance. As the robotic industry develops over the past decades, there are more applications depending on the performance of the SLAM system. However, a good SLAM system needs to meet the following requirements. First of all, many advanced robotic applications require localization accuracy at sub-meter level or even centimeter level, e.g., precision landing for Unmanned Aerial Vehicles (UAVs) and auto charging for Automated Guided Vehicles (AGVs). Secondly, the implementation of SLAM extends from indoor AGVs to outdoor autonomous driving cars or UAVs. As the robots move faster, the localization needs to be real-time as well. Any delay in localization result may lead to serious safety issues such as object collision or car accident. Lastly, the robotic applications are expanding from static to dynamic environments, from simple to complex environments, from short term to long term operation, etc. The SLAM framework is supposed to provide reliable localization under different scenarios and be robust to environmental changes. However, mobile robots often have limited computational resources to achieve a good SLAM performance. Motivated by this challenge, this thesis presents a unified SLAM framework with high flexibility, practicality and stability for autonomous vehicles. Specifically, we explore improvement opportunities in front-end and back-end SLAM systems separately, and then examine their integration in warehouse robots and autonomous driving cars.

In the first part of this thesis, we establish a LiDAR-based odometry to provide real-time localization for warehouse robots. Light Detection And Ranging (LiDAR) is an important sensor used for autonomous vehicles due to its high accuracy. Note that lines and planes are often distinct, we adopt an efficient feature extraction via

local smoothness analysis to search for edge and planar features respectively. The extracted features are associated with global lines and planes and the robot pose is achieved by minimizing point-to-edge and point-to-plane distances. Moreover, we adopt non-iterative sensor motion estimation and distortion correction to reduce the computational cost. As a result, the framework achieves competitive localization accuracy with a processing rate of more than 10 Hz in the public dataset evaluations. It provides a good trade-off between performance and computational cost for practical applications.

A LiDAR odometry solves the localization problem for short travel distance before the measurement noise causes localization drifts in the long run. The drifting problem needs to be resolved to enable a mobile robot to run for long hours in scenarios such as continuous warehouse operation. Hence, in the second part of this thesis, we investigate the mitigation of the drifting problem with loop closure detection at the back-end. Loop closure detection is the task to identify the repetitive places from a database and re-localize the robot to eliminate the drifting error. We first explore a vision-based system for loop closure detection. Saliency analysis is introduced to identify the distinctive landmarks from the image stream. New landmarks are compared with existing landmarks to retrieve the repetitive scenes. Since the database is incremental when more places are explored, a Kernel-Cross-Correlator-based landmark retrieval is introduced to reduce the computational complexity. The experiment results show that it is faster than existing feature-based approaches while achieving similar recall rates.

The vision-based loop closure detection only explores the visual characteristic of a place, which can be easily affected by environmental changes such as illumination difference. Compared to visual information, the geometry feature is less sensitive to such variation. We next explore the LiDAR based loop closure detection. Specifically, a global novel descriptor named Intensity Scan Context (ISC) is introduced with both geometry and intensity properties. To improve the efficiency for place retrieval, an efficient two-stage hierarchical re-identification process is proposed, including binary-operation based fast geometric relation retrieval and intensity structure re-identification. The experimental results show that the LiDAR based loop closure detection achieves similar performance with vision-based approach in terms of recall rate, but has higher robustness in some difficult scenarios such as reverse visiting.

After exploring the front-end SLAM and back-end SLAM independently, we integrate the two components together to form a full SLAM solution. An intensity-assisted front-end odometry estimation and an ISC-based back-end optimization are running in parallel to provide real-time localization and global optimization. Thorough experiments are performed on both outdoor autonomous driving and indoor warehouse robot manipulation. Compared to state-of-the-art works, this work achieves the fastest processing speed while maintaining satisfactory accuracy in various environments. It is a cost-effective solution for mobile robots with limited computational power.

Contents

Acknowledgements	ix
Abstract	xi
List of Figures	xxi
List of Tables	xxv
Symbols and Acronyms	xxvii
1 Introduction	1
1.1 Background and Motivation	1
1.2 Objectives	4
1.3 Main Contributions	4
1.4 Overview of the Thesis	9
2 Literature Review	13
2.1 Localization Techniques for Robotics	13
2.1.1 Global Navigation Satellite System	14
2.1.2 Motion Capture System	15
2.1.3 Ultra-Wideband Localization	15
2.1.4 Wireless Local Area Networks	17
2.1.5 Other Landmark-based Approaches	17
2.1.6 Simultaneous Localization And Mapping	18
2.1.7 Feasibility Analysis of Different Approaches	21
2.2 SLAM in Unknown Environments	21
2.2.1 Visual Odometry	22
2.2.1.1 Feature-based Approach	22
2.2.1.2 Appearance-based Approach	24
2.2.2 LiDAR Odometry	25
2.2.2.1 Iterative Closest Point	25
2.2.2.2 Normal Distribution Transformation	26
2.2.2.3 Line and Plane Features Tracking	28
2.2.3 Limitations of Existing Localization Methods in Applications	30

2.3	Loop Closure Detection and Re-Localization	30
2.3.1	Graph-based Optimization	31
2.3.2	Loop Closure Detection via Visual Features	33
2.3.2.1	Appearance-based Matching	33
2.3.2.2	Deep-learning-based Approach	35
2.3.2.3	Saliency-based Approach	37
2.3.3	Loop Closure Detection via Geometric Features	38
2.3.4	Limitations of Loop Closure Detection in Robotic Applications	40
2.4	Mathematical Preliminaries	41
2.4.1	Lie Algebra and Lie Group	41
2.4.2	Kernel Cross Correlator	42
2.5	Conclusion	44
3	Lightweight LiDAR Odometry for Autonomous Vehicles	45
3.1	Introduction	45
3.2	Related Works	46
3.3	System Overview	48
3.3.1	Sensor Model and Feature Extraction	48
3.3.2	Motion Estimation and Distortion Compensation	49
3.3.3	Feature Correspondence and Pose Estimation	51
3.3.4	Map Building	54
3.4	Experiment Results	56
3.4.1	Evaluation on Public Dataset	56
3.4.2	Ablation Study	61
3.4.3	Experiment on Warehouse Logistics	61
3.4.3.1	Simulation	62
3.4.3.2	Experiment	62
3.4.3.3	Performance Evaluation	62
3.5	Conclusion	64
4	Fast Loop Closure Detection via Binarized Saliency Map	67
4.1	Introduction	67
4.2	Related Works	69
4.3	System Overview	70
4.3.1	Binarized Saliency Map Construction	70
4.3.2	Fast Image Retrieval	73
4.3.3	Precise Loop Closure Detection	73
4.4	Experiment Results	75
4.4.1	Evaluation on Public Datasets	75
4.4.2	Comparison with Other Methods	77
4.5	Conclusion	79
5	Online Visual Place Recognition via Saliency Re-identification	81
5.1	Introduction	81

5.2	Related Works	82
5.2.1	Feature-based Methods via Online Training	82
5.2.2	Deep Learning-based Methods	83
5.2.3	Saliency Re-identification-based Methods	84
5.3	System Overview	84
5.3.1	Saliency Detection	84
5.3.1.1	Definition	85
5.3.1.2	Property	87
5.3.2	Saliency Retrieval	88
5.3.2.1	VSR Training	88
5.3.2.2	VSR Retrieval	90
5.3.3	Consistency Verification	91
5.4	Experiment Results	91
5.4.1	Setup and Metrics	91
5.4.2	Performance Evaluation	92
5.4.2.1	Results	93
5.4.2.2	Precision-Recall Curve	93
5.4.3	Comparison	94
5.4.3.1	Efficiency Evaluation	95
5.4.3.2	Overall Performance	96
5.5	Conclusion	98
6	Intensity Scan Context: Coding Intensity and Geometry Relations for Loop Closure Detection	99
6.1	Introduction	99
6.2	Related Works	101
6.3	System Overview	102
6.3.1	Intensity Calibration and Pre-processing	102
6.3.2	Intensity Scan Context	104
6.3.3	Place Re-identification	105
6.3.3.1	Fast Geometry Re-identification	105
6.3.3.2	Intensity Structure Matching	107
6.3.4	Consistency Verification	107
6.3.4.1	Temporal Consistency Check	108
6.3.4.2	Geometrical Consistency Check	108
6.4	Experiment Results	108
6.4.1	Experiment on Autonomous Guided Vehicle	110
6.4.2	Evaluation on Public Dataset	110
6.5	Conclusion	113
7	Intensity-SLAM: Intensity Assisted Localization and Mapping for Large Scale Environment	115
7.1	Introduction	115
7.2	Related Works	116

7.3	System Overview	118
7.3.1	Intensity Calibration	118
7.3.2	Salient Point Selection and Feature Extraction	120
7.3.3	Intensity Map Building	120
7.3.4	Scan to Map Matching	121
7.3.4.1	Geometry Residual	121
7.3.4.2	Intensity Residual	122
7.3.4.3	Pose Estimation	123
7.3.5	Loop Closure Detection and Global Optimization	123
7.4	Experiment Results	125
7.4.1	Experiment Setup	125
7.4.2	Evaluation on Public Dataset	125
7.4.3	Ablation Study of Intensity Information	125
7.4.4	Comparison with Existing Methods	126
7.4.5	Validation on Practical Robots	129
7.5	Conclusion	130
8	Lightweight 3D Localization and Mapping for Solid-State LiDAR	131
8.1	Introduction	131
8.2	Related Works	134
8.3	System Overview	134
8.3.1	Feature Extraction	134
8.3.2	Odometry Estimation	136
8.3.3	Probability Map Construction	138
8.4	Experiment Results	140
8.4.1	Experiment Setup	140
8.4.2	Performance Evaluation and Comparison	140
8.4.3	Performance on Warehouse Robot	142
8.4.4	Performance on Hand-held Device	143
8.4.4.1	3D Mapping	143
8.4.4.2	Rotation Test	144
8.5	Conclusion	144
9	Conclusion and Future Work	147
9.1	Conclusion	147
9.2	Limitations of Current Research	150
9.2.1	SLAM in Degraded Environments	150
9.2.2	Loop Closure Detection under Illumination Changes	150
9.2.3	SLAM in Dynamic Environments	151
9.3	Future Work	151
9.3.1	Multi-sensor Fusion	151
9.3.1.1	Integration with Inertial Measurement Unit Sensor	152
9.3.1.2	Combination of Both LiDAR and Visual SLAM	152

9.3.1.3	Integration with Other Localization Techniques . . .	152
9.3.2	Machine Learning	153
9.3.2.1	Loop Closure Detection via Incremental Learning .	153
9.3.2.2	Semantic SLAM	153
9.3.3	Applications	154
9.3.3.1	Virtual Reality	154
9.3.3.2	Drone Inspection	154
 List of Author's Publications		 155
Bibliography		157

List of Figures

1.1	Overview of this thesis.	10
2.1	Operational principle of Differential-GPS.	14
2.2	MCS used in different research scenarios.	16
2.3	Operational principle of UWB-based localization.	17
2.4	QR code localization for humanoid robot.	18
2.5	AGV localization via magnetic stripes.	19
2.6	Illustration of the BA problem. The solid line represents the projection procedure, while the dotted line represents the reprojection procedure.	23
2.7	Matching of two point cloud by ICP.	26
2.8	Example of 3D-NDT representation for a tunnel section. Brighter and denser parts represent higher probabilities. (a) original point cloud; (b) NDT result.	27
2.9	Illustration of global edge and plane estimation. (a) global edge estimated from two edge features; (b) global surface estimated from three planar features.	29
2.10	Illustration of graph-based optimization.	31
2.11	Global optimization with constraints from loop closure detection. (a) mapping result without loop closure; (b) mapping result after global optimization.	32
2.12	BoW model for LCD.	34
2.13	Training a CNN framework with a massive (2.5M image) place recognition-specific dataset.	36
2.14	CNN architecture with the NetVLAD layer.	36
2.15	The formulation of the emission matrix using the combination of both traditional methods and deep-learning based methods.	37
2.16	The influence region diagram for a FPFH feature.	39
3.1	Relation between sweeping angle and time.	50
3.2	Setup of the autonomous driving car on the KITTI dataset.	57
3.3	Result of the proposed FLOAM on the KITTI dataset. The estimated trajectory and ground truth are plotted in green and red respectively. (a)-(k) KITTI sequence 00-10.	58

3.4	Example of the proposed method on the KITTI dataset. (a) the mapping result on sequence 05; (b) the reconstructed 3D road scenery by integrating camera view; (c) result comparison of FLOAM and ground truth.	59
3.5	Comparison of the different localization approaches on the KITTI dataset.	61
3.6	Simulated warehouse environment with both static and dynamic objects. (a) simulated environment in Gazebo; (b) simulated Pioneer robot and Velodyne LiDAR for evaluation; (c) trajectory comparison of FLOAM and ground truth.	63
3.7	Advanced factory environment built for AGV manipulation, including operating machine, auto charging station and storage shelves. .	63
3.8	Evaluation of the proposed method in a warehouse environment. (a) AGV used for the experiment; (b) FLOAM result of warehouse localization and mapping.	64
3.9	Comparison with ground truth. (a) Pioneer robot used for performance evaluation; (b) testing area.	65
3.10	Comparison of the proposed method and ground truth. Our method can precisely track robot's pose with an average localization error of 2 cm	65
4.1	Framework of the proposed method.	71
4.2	An example of Log spectral residual approach.	72
4.3	Examples of binarized saliency map extraction.	74
4.4	An example of SURF feature points matching.	74
4.5	Loop closure detection result of the proposed method.	76
4.6	Comparison of binarized saliency map extraction with existing methods.	78
5.1	System overview of the proposed method.	85
5.2	An example of saliency detection from the KITTI dataset. (a) mask of the detected saliency map; (b) the detected salient regions enclosed by red bounding boxes.	87
5.3	The procedure of saliency retrieval. The same traffic sign is identified when a car re-visits the same place.	91
5.4	Examples of place recognition on the KITTI dataset. (a, b) and (c, d) are the results of sequence 00 and sequence 05, respectively. The trajectory color changes from light to dark with respect to time. The matches from ground truth are marked with black, while the detected matches are highlighted with red. It can be seen that most of the re-visited places are recognized.	94

5.5	Examples of place recognition on public datasets. The trajectory from GPS is plotted from light grey to dark grey with respect to time. The loop closure reported by the proposed method is marked with a red dot. (a, b, c) are collected from KITTI sequence 02, 07 and 09 respectively; (d, e, f) are collected from Oxford RobotCar sequence 2014_05_14_13_50_20, 2014_08_11_10_22_21 and 2014_12_02_15_30_08 respectively. Our method achieves 80% recall rate with no false positive.	95
5.6	Precision-recall curve of the proposed method on different datasets.	96
5.7	Performance evaluation. (a) comparison of saliency extraction with feature extraction methods on KITTI sequence 00; (b) time cost comparison of different approaches on KITTI sequence 00. Our approach achieves much faster speed than existing methods.	97
5.8	Overall performance of different approaches.	98
6.1	An example of intensity reading from the KITTI dataset. The relationships of intensity information and landmarks are highlighted with red rectangles.	103
6.2	A visual illustration of the proposed intensity scan context. Top figure: original point cloud is decomposed into subspace based on geometry characteristics. Bottom figure: derived intensity scan context by intensity projection on each subspace.	106
6.3	Autonomous warehouse robot platform used for our experiment.	109
6.4	Local experiment result of the proposed method.	111
6.5	Loop closure detection result. The left images are camera view, the right images are loop closure detection result. (a) KITTI sequence 00; (b) KITTI sequence 02; (c) KITTI sequence 05.	112
6.6	Performance comparison between the different methods.	114
7.1	System overview of the proposed Intensity-SLAM. In the front-end, we analyze both geometry and intensity features to estimate odometry. In the back-end, we use intensity scan context for loop closure detection and global pose graph optimization.	118
7.2	Trilinear interpolation.	122
7.3	Result of the proposed method on the KITTI dataset. In the first row from left to right are sequences 00, 02, in the second row are sequences 05, 07, and 09. The result of intensity-SLAM is plotted in green color and the ground truth is plotted in red color.	126
7.4	Comparison of FLOAM and intensity-SLAM. The ground truth is plotted in red and the estimated trajectory is plotted in green. (a) and (b) are the results of FLOAM on KITTI Sequences 00 and 05. (c) and (d) are the results of Intensity-SLAM on KITTI Sequences 00 and 05.	127
7.5	Performance comparison on KITTI sequence 05.	128

7.6	Localization and mapping result in a warehouse environment. (a) AGV platform for warehouse manipulation; (b) localization and mapping result of the proposed method. The robot's trajectory is plotted in red color.	129
8.1	Comparison of mechanical LiDAR and Solid-state LiDAR.	132
8.2	System overview of the proposed method. It consists of three main modules: feature extraction, odometry estimation and probability map construction, which are highlighted in yellow, blue and red color respectively.	135
8.3	Performance evaluation with ground truth. (a) Pioneer robot equipped with a solid-state LiDAR; (b) testing field with VICON system. . .	141
8.4	Comparison of the proposed method, LOAM and ground truth. LOAM loses track during large rotation while the proposed method is able to track accurately. And our approach achieves 5 <i>cm</i> localization accuracy.	141
8.5	An example of indoor localization and mapping in a warehouse environment. (a) the AGV platform used for warehouse manipulation with a solid-state LiDAR mounted in the front; (b) reconstructed map.	142
8.6	Map quality evaluation. The reconstructed map is shown in the middle. We randomly picked two places for illustration. (a) and (b) are the original camera view; (c) and (d) are reconstructed scenes based on the proposed approach.	143
8.7	Integration of the proposed method on a hand-held scanner. (a) lightweight hand-held scanner using solid-state LiDAR as perception system; (b) localization and mapping result.	145

List of Tables

2.1	Comparison of different localization approaches.	22
3.1	Information of KITTI sequence 00-10.	58
3.2	Performance evaluation of FLOAM on the KITTI dataset.	60
3.3	Ablation study of localization accuracy and computational cost. . .	61
4.1	Information of different datasets.	75
4.2	Loop detection results of our approach.	77
4.3	Performance comparison of FABMAP, DBoW2 and the proposed method on different datasets.	78
5.1	Parameter list.	92
5.2	Details of datasets used for evaluation.	93
6.1	Parameter list.	110
6.2	Comparison with existing methods.	113
7.1	Result analysis of intensity-aided method and non-intensity-aided method on the KITTI dataset.	128
7.2	Performance comparison of different approaches on the KITTI dataset.	128
8.1	Difference of mechanical and solid-state LiDAR.	133
8.2	Results of rotation test.	144

Symbols and Acronyms

Symbols

$\mathbb{R}^n$	the set of n -dimensional real column vectors
$\mathbb{R}^{m \times n}$	the set of $m \times n$ -dimensional real matrices
$\mathbf{A}^T$	the transpose of matrix $\mathbf{A}$
$\mathbf{A}^{-1}$	the inverse of matrix $\mathbf{A}$
$\mathbf{I}_{m \times m}$	an m -dimensional identity matrix
$\mathbf{0}_{m \times n}$	an $m \times n$ matrix with all entries being zero
$\hat{\mathbf{x}}$	the DFT of a vector or matrix $\mathbf{x}$
$\mathbf{x}^*$	the conjugate of a vector or matrix $\mathbf{x}$
$\odot$	the Hadamard (element-wise) product
$[\cdot, \cdot]$	the set of all integers between two values
$\ \cdot\ $	the 2-norm of a vector or matrix in Euclidean space
$ \cdot $	the absolute value of a real number or the cardinality of a set
$[\cdot]_{\times}$	the skew matrix of a 3D vector
$\mathbf{a} \cdot \mathbf{b}$	the dot product of two vectors $\mathbf{a}$ and $\mathbf{b}$
$\mathbf{a} \times \mathbf{b}$	the cross product of two vectors $\mathbf{a}$ and $\mathbf{b}$
$\angle(\mathbf{a}, \mathbf{b})$	the angle between two vectors $\mathbf{a}$ and $\mathbf{b}$
$XOR(\mathbf{a}, \mathbf{b})$	element-wise Exclusive Or of two matrices $\mathbf{a}$ and $\mathbf{b}$
$\mathcal{F}^{-1}(\cdot)$	the inverse DFT of a vector or matrix
$\mathcal{O}(\cdot)$	the compleixty of an operation
$\min_{x \in \mathcal{S}} f(x)$	minimum of $f(x)$ where x takes value from the set $\mathcal{S}$
$\max_{x \in \mathcal{S}} f(x)$	maximum of $f(x)$ where x takes value from the set $\mathcal{S}$
$\arg \min_{x \in \mathcal{S}} f(x)$	argument of the minimum of $f(x)$ over the set $\mathcal{S}$
$\arg \max_{x \in \mathcal{S}} f(x)$	argument of the maximum of $f(x)$ over the set $\mathcal{S}$

Acronyms

ADS	Autonomous Driving System
UAV	Unmanned Aerial Vehicles
AGV	Automated Guided Vehicles
UAV	Unmanned Aerial Vehicles
MCS	Motion Capture System
UWB	Ultra-Wide Band
SLAM	Simultaneous Localization and Mapping
LiDAR	Light Detection And Ranging
GNSS	Global Navigation Satellite System
LCD	Loop Clousre Detection
VPR	Visual Place Recognition
GNSS	Global Navigation Satellite System
MLE	Maximum Likelihood Estimation
MAP	Maximum A Posterior
BA	Bundle Adjustment
VSR	Visual Salient Region
DoF	Degree of Freedom
LO	LiDAR Odometry
VO	Visual Odometry
VSLAM	Visual Simultaneous Localization and Mapping
LSLAM	LiDAR Simultaneous Localization and Mapping
NLS	Nonlinear Least Squares
IMU	Inertial Measurement Unit
ROS	Robotics Operating System
ICP	Iterative Closest Point
NDT	Normal Distribution Transform
FoV	Field of View
DFT	Discrete Fourier Transform
ISC	Intensity Scan Context
KCC	Kernel Cross-Correlator
FFT	Fast Fourier Transform
BoW	Bag of Words
KNN	K-Nearest Neighbour

DoF	Degree of Freedom
EKF	Extended Kalman Filter
CNN	Convolutional Neural Network

Chapter 1

Introduction

1.1 Background and Motivation

Over the past decades, Unmanned Grounded Vehicles (UGVs) applications have profound impacts in many aspects of human society, e.g., Autonomous Driving System (ADS) [1, 2], smart manufacturing [3, 4], domestic cleaning [5, 6] and coal mine rescue [7, 8]. Modern intelligent robots have drawn great research attention due to their superior advantages over humans in the scope of high-intensity and high-risk tasks. For example, it has been witnessed that ADS is able to reduce the chance of car accidents compared to human drivers, which could potentially save human lives [9]. Besides, in the field of industry 4.0, Automated Guided Vehicles (AGVs) have taken over human labour to improve the durability, efficiency and productivity for smart manufacturing [4].

To achieve various high level intelligent tasks, precise localization is a common and fundamental challenge shared by all kinds of robots. Global Positioning System (GPS) is the most popular localization method, which provides the absolute location based on satellite signals. It often has a localization error of 3-5 meters and is not applicable under GPS-denied environments such as indoor areas. In recent years, it has been demonstrated that the sub-meter level localization can be achieved by setting external infrastructures in the context of known environment, e.g., Motion Capture System (MCS) [10], Ultra-Wide Band (UWB) Localization [11], and Wi-Fi Localization [12]. The localization can be solved by geometry analysis based on the distance between the robot and fixed landmarks or anchors.

Unfortunately, setting up of fixed anchors is not possible in many practical scenarios where the environment is large or inaccessible.

Recent researches focus on Simultaneous Localization And Mapping (SLAM) and target to solve localization without external setup. To localize robots in an unknown environment, a SLAM system is required to automatically explore, identify and track landmarks/features. In practice, a good SLAM system needs to tackle several challenges. First of all, many advanced robotic applications require localization accuracy at sub-meter level or even centimeter level, e.g., precision landing for Unmanned Aerial Vehicles (UAVs) and auto charging for AGVs. Different from infrastructure-based methods, SLAM has to define and identify landmarks automatically. Therefore, robust landmark identification and tracking are of great importance towards a high-accuracy SLAM. Secondly, the implementation of SLAM extends from indoor AGVs to outdoor autonomous driving cars or UAVs. To guarantee successful completion of tasks, real-time localization is essential for both control and navigation of fast moving robots. Any delay in localization results may lead to serious safety issues such as object collision or car accident. Lastly, the robotic applications are expanding from static to dynamic environments, from simple to complex environments, from short term to long term operation, etc. The SLAM framework is supposed to provide reliable localization under different scenarios, regardless of environmental changes. However, mobile robots often have limited computational resources so that the computational complexity must be low in order to achieve real-time performance. In general, SLAM is divided into two independent topics: odometry estimation and re-localization, which involves feature recognition, extraction, retrieval and pose optimization to run simultaneously. The dilemma between accuracy and computational efficiency has become a bottleneck for practical applications.

Odometry estimation is a task to estimate the robot state and its relative location towards multiple landmarks observed. However, there is not yet a unified solution with a good trade-off between performance and computational efficiency. In the scope of Visual SLAM (VSLAM), the odometry estimation is often known as Bundle Adjustment (BA) which is a Maximum Likelihood Estimation (MLE) problem. BA can be achieved by minimizing the reprojection error between features in the image plane and the predicted landmark positions. This can be subsequently interpreted as a high-dimensional nonlinear least-squares problem, which is of high

computational complexity. Another approach to solve odometry estimation is to use Iterative Closest Point (ICP) in LiDAR SLAM (LSLAM), where all observations are considered without identifying specific landmarks. The current observations are simply matched to historical observations by minimizing the distance between point clouds. However, it is rather computationally inefficient to match dense point clouds recursively in practice.

In the presence of accumulative localization drifts, odometry estimation does not perform well in the long run. The drifting problem can be caused by false detection, false matching, dynamic landmarks, measurement noise, etc. This problem can be eliminated by loop closure detection and re-localization, which is a task to identify the repetitive landmarks or scenes from historical observations. Existing works have demonstrated the great importance of re-localization in improving localization accuracy [13]. However, realization of Loop Closure Detection (LCD) or Place Recognition (PR) can be rather complicated. One limitation is that the re-observed landmarks can look different from different viewing angles, lighting conditions, etc. On the other hand, computational cost is also significant in practice. With more places being visited, the number of observed landmarks in the database increases. Hence it is also a great challenge to efficiently retrieve a matched landmark from an incremental database in practice.

In recent years, the introduction of solid-state LiDAR provides a cost-effective solution for robot perception systems. A solid-state LiDAR integrates all measurement sensors into a single chip, which is more compact and stable compared to traditional mechanical LiDAR. It is considered as one of the most promising perception systems in robotic applications. However, its small Field of View (FoV) limits the pose estimation capability and there are very limited works to tackle this problem for solid-state LiDAR.

In summary, motivated by the practical localization challenges, our research focuses on building a robust and light-weight SLAM system for autonomous vehicles. Specifically, we explore improvement opportunities for odometry estimation and re-localization separately, and then examine their integration in warehouse robots and autonomous driving cars.

1.2 Objectives

Simultaneous Localization And Mapping (SLAM) is one of the most fundamental and essential topics in robotics research. As the robotic industry develops over the past decades, there are more applications depending on the performance of the SLAM system. However, a good SLAM system needs to meet the following requirements. First of all, many advanced robotic applications require localization accuracy at sub-meter level or even centimeter level, e.g., precision landing for Unmanned Aerial Vehicles (UAVs) and auto charging for Automated Guided Vehicles (AGVs). Secondly, the implementation of SLAM extends from indoor AGVs to outdoor autonomous driving cars or UAVs. As the robots move faster, the localization needs to be real-time as well. Any delay in localization result may lead to serious safety issues such as object collision or car accident. Lastly, the robotic applications are expanding from static to dynamic environments, from simple to complex environments, from short term to long term operation, etc. The SLAM framework is supposed to provide reliable localization under different scenarios and be robust to environmental changes. However, mobile robots often have limited computational resources to achieve a good SLAM performance. Motivated by this challenge, this thesis presents a unified SLAM framework with high flexibility, practicality and stability for autonomous vehicles. Specifically, we explore improvement opportunities in front-end and back-end SLAM systems separately, and then examine their integration in warehouse robots and autonomous driving cars.

1.3 Main Contributions

The contribution of this thesis is summarized as follows: Simultaneous Localization And Mapping (SLAM) is one of the most fundamental and essential topics in robotics research. SLAM is a task for a robot to perceive the environment and localize itself based on inputs from its on-board sensors. The robot is also supposed to construct a map of the surrounding environment for subsequent task planning and collision avoidance. As the robotic industry develops over the past decades, there are more applications depending on the performance of the SLAM system. However, a good SLAM system needs to meet the following requirements. First of

all, many advanced robotic applications require localization accuracy at sub-meter level or even centimeter level, e.g., precision landing for Unmanned Aerial Vehicles (UAVs) and auto charging for Automated Guided Vehicles (AGVs). Secondly, the implementation of SLAM extends from indoor AGVs to outdoor autonomous driving cars or UAVs. As the robots move faster, the localization needs to be real-time as well. Any delay in localization result may lead to serious safety issues such as object collision or car accident. Lastly, the robotic applications are expanding from static to dynamic environments, from simple to complex environments, from short term to long term operation, etc. The SLAM framework is supposed to provide reliable localization under different scenarios and be robust to environmental changes. However, mobile robots often have limited computational resources to achieve a good SLAM performance. Motivated by this challenge, this thesis presents a unified SLAM framework with high flexibility, practicality and stability for autonomous vehicles. Specifically, we explore improvement opportunities in front-end and back-end SLAM systems separately, and then examine their integration in warehouse robots and autonomous driving cars.

In the first part of this thesis, we establish a LiDAR-based odometry to provide real-time localization for warehouse robots. Light Detection And Ranging (LiDAR) is an important sensor used for autonomous vehicles due to its high accuracy. Note that lines and planes are often distinct, we adopt an efficient feature extraction via local smoothness analysis to search for edge and planar features respectively. The extracted features are associated with global lines and planes and the robot pose is achieved by minimizing point-to-edge and point-to-plane distances. Moreover, we adopt non-iterative sensor motion estimation and distortion correction to reduce the computational cost. As a result, the framework achieves competitive localization accuracy with a processing rate of more than 10 Hz in the public dataset evaluations. It provides a good trade-off between performance and computational cost for practical applications.

A LiDAR odometry solves the localization problem for short travel distance before the measurement noise causes localization drifts in the long run. The drifting problem needs to be resolved to enable a mobile robot to run for long hours in scenarios such as continuous warehouse operation. Hence, in the second part of this thesis, we investigate the mitigation of the drifting problem with loop closure detection at the back-end. Loop closure detection is the task to identify the repetitive places

from a database and re-localize the robot to eliminate the drifting error. We first explore a vision-based system for loop closure detection. Saliency analysis is introduced to identify the distinctive landmarks from the image stream. New landmarks are compared with existing landmarks to retrieve the repetitive scenes. Since the database is incremental when more places are explored, a Kernel-Cross-Correlator-based landmark retrieval is introduced to reduce the computational complexity. The experiment results show that it is faster than existing feature-based approaches while achieving similar recall rates.

The vision-based loop closure detection only explores the visual characteristic of a place, which can be easily affected by environmental changes such as illumination difference. Compared to visual information, the geometry feature is less sensitive to such variation. We next explore the LiDAR based loop closure detection. Specifically, a global novel descriptor named Intensity Scan Context (ISC) is introduced with both geometry and intensity properties. To improve the efficiency for place retrieval, an efficient two-stage hierarchical re-identification process is proposed, including binary-operation based fast geometric relation retrieval and intensity structure re-identification. The experimental results show that the LiDAR based loop closure detection achieves similar performance with vision-based approach in terms of recall rate, but has higher robustness in some difficult scenarios such as reverse visiting.

After exploring the front-end SLAM and back-end SLAM independently, we integrate the two components together to form a full SLAM solution. An intensity-assisted front-end odometry estimation and an ISC-based back-end optimization are running in parallel to provide real-time localization and global optimization. Thorough experiments are performed on both outdoor autonomous driving and indoor warehouse robot manipulation. Compared to state-of-the-art works, this work achieves the fastest processing speed while maintaining satisfactory accuracy in various environments. It is a cost-effective solution for mobile robots with limited computational power.

- We develop a lightweight LiDAR-based Odometry and Mapping framework that provides a practical real-time localization solution for autonomous vehicles. Specifically, we explore edge features and planar features based on the local geometry distribution and match them into globally maintained edge and plane maps respectively. Local smoothness is introduced to efficiently

evaluate the uniqueness of feature points and is subsequently used for iterative pose optimization to improve the localization accuracy. To reduce the computational cost, non-iterative distortion compensation is adopted to efficiently correct motion distortion. A thorough evaluation of the proposed method is presented to verify the robustness, which shows that our framework is able to achieve real time performance on a low power embedded computing unit. It is also worthy noting that the proposed method is one of the most accurate and fastest open-sourced methods for KITTI benchmark.

- We introduce a vision-based re-localization technique via binarized saliency map, which combines the advantage of fast binary operation and traditional loop closure detection approach. Compared to the traditional appearance-based approach, we analyze the saliency distribution of a place and build the binarized object relationship on top of the feature-based approach to reduce the computational cost. Moreover, no offline training is required in our method. The experiments show that by combining fast binary operation and existing feature-space retrieval, the proposed method achieves faster speed than existing feature-based methods without reducing recall rate and precision.
- We develop a vision-based re-localization technique based on the saliency analysis on images. Inspired by the fact that human beings always recognize a place by remembering objects or landmarks that are more appealing or interesting than others, we formulate loop closure detection as saliency re-identification. It is more natural and straightforward than feature matching. Specifically, we adopt saliency analysis in frequency domain to efficiently identify the saliency distribution. A translational and rotational invariant kernel-cross-correlation-based saliency re-identification is introduced to retrieve the repetitive information from historical databases. We show that both saliency detection and retrieval can be performed in frequency domain, which can further reduce the computational cost. The experiments indicate that our method is much faster than existing state-of-the-art methods with higher recall rate and precision. It does not require offline training or pre-trained vocabulary.
- We develop a LiDAR-based re-localization technique to achieve efficient and robust loop closure detection. A novel global descriptor, namely Intensity

Scan Context (ISC), is introduced that leverages both geometry and intensity characteristics. We explain how intensity information can be distinguishable in place recognition and present a procedure to construct the new descriptor. To further improve the efficiency for loop closure detection, a two-stage hierarchical place re-identification strategy is proposed, including binary-operation based fast geometry retrieval and intensity structure matching. A thorough evaluation of the proposed method is presented and the results show that the proposed approach achieves a higher recall rate and recall precision than existing geometric-only loop closure detection methods.

- A lightweight and robust SLAM system is proposed that integrates both LiDAR odometry and re-localization into a full solution for autonomous vehicles. Several techniques are used to further enhance the robustness and computational efficiency for practical applications. Specifically, a model-based intensity calibration approach is introduced to correct the reflectivity of the surrounding environment. Then we construct an intensity global map and combine an extra intensity cost and the existing geometric-only cost for odometry estimation to further improve the localization accuracy at a low computational cost. An ISC-based back-end factor graph optimization is also integrated to correct localization drifts. A thorough evaluation on the proposed method is presented on both warehouse and autonomous driving. The results indicate that our method outperforms the existing geometric-only methods across multiple scenarios and it is able to provide real-time localization for mobile robots with limited computational power.
- We further extend the proposed SLAM framework with solid-state LiDAR introduced recently. We propose a new rotation-invariant feature extraction method that explores both horizontal and vertical curvature. The proposed method provides real-time localization up to 30 Hz with 5 cm accuracy. Thorough experiments have been conducted to evaluate its performance. Specifically, we integrate an Intel L515 solid-state LiDAR to AGVs and test the proposed method in a complex warehouse environment. The proposed method is also validated on a hand-held device which is able to provide high-resolution mapping in real-time.

1.4 Overview of the Thesis

In this section, we will briefly introduce the overview of the proposed SLAM system. As discussed in Section 1.2, mobile robots often have limited computational resources to achieve a good SLAM performance. A good SLAM is supposed to achieve a good balance between computational cost and localization accuracy. Motivated by this challenge, this thesis presents a unified SLAM framework with high flexibility, practicality and stability for autonomous vehicles. Specifically, we explore improvement opportunities in the front-end and back-end SLAM systems separately, and then examine their integration in warehouse robots and autonomous driving cars. The rest of this thesis is organised in the following manner:

- Chapter 2 presents a literature review on different localization approaches for autonomous vehicles. Moreover, existing SLAM techniques used in odometry estimation and LCD are reviewed.
- Chapter 3 describes the details of building a computationally efficient LiDAR odometry and mapping system as well as the implementation on different types of mobile robots.
- Chapter 4 introduces a vision-based loop closure detection by combining fast binarized saliency map-based retrieval and feature based-method in order to reduce the computational cost of the existing methods.
- Chapter 5 introduces a vision-based loop closure detection via saliency analysis, where several computer vision techniques are used to reduce the computational cost.
- Chapter 6 introduces the ISC as a global descriptor and demonstrates the effectiveness of ISC in identifying loop closure under a two-stage retrieval.
- Chapter 7 details the integration of both LO and LCD systems into a complete SLAM solution, which is subsequently implemented into different types of autonomous vehicles to demonstrate the performance.
- Chapter 8 details the system structure of SLAM for solid-state LiDAR. The robustness of the proposed method is evaluated in both AGVs and hand-held devices.

- Chapter 9 presents the conclusion and future work.

The connection between each chapter is shown in Figure 1.1. We firstly look into the problem of odometry estimation. In Chapter 3, we focus on LiDAR odometry and target to improve the performance with geometric-only features. Then in Chapter 7, we target to fuse the intensity feature into geometric-only LiDAR odometry. By properly calibrating intensity information, we manage to improve the performance of LiDAR odometry from Chapter 3.

Then we focus on the back-end SLAM system. We first explore a vision-based system for loop closure detection. In Chapter 4, we propose to use saliency distribution to identify the distinctive landmarks from the image stream. This work is further extended in Chapter 5. The vision-based loop closure detection only explores the visual characteristic of a place, which can be easily affected by environmental changes such as illumination difference. Compared to visual information, the geometry feature is less sensitive to such variation. We next explore the LiDAR based loop closure detection in Chapter 6.

After exploring the front-end SLAM and back-end SLAM independently, we integrate the two components together to form a full SLAM solution in Chapter 7. Moreover, in recent years, the introduction of solid-state LiDAR provides a cost-effective solution for robot perception systems. In Chapter 8, we further extend

FIGURE 1.1: Overview of this thesis. Specifically, we explore improvement opportunities in the front-end and back-end SLAM systems separately, and then examine their integration in warehouse robots and autonomous driving cars.

the SLAM introduced in the previous chapters and propose a SLAM system for the new solid-state LiDAR.

Chapter 2

Literature Review

In this chapter, an extensive review related to different localization techniques as well as existing solutions to the SLAM problem is provided. We begin with explaining different localization approaches in Section 2.1, including GPS system, MCS, UWB Localization, marker-based localization and SLAM. Then we analyze the feasibility of different localization approaches in real applications. The remaining part of the chapter proceeds as below: Section 2.2 reviews the state-of-the-art methods that are used to solve the localization problems in unknown environments. Both Visual Odometry (VO) methods and LiDAR Odometry (LO) methods are included. Section 2.3 describes the problem formulation of LCD and the importance of LCD in a SLAM system. Then we illustrate the existing approaches to solve LCD problems. Finally, Section 2.5 summarizes this chapter.

2.1 Localization Techniques for Robotics

The localization is a fundamental component when designing a robotic system. Based on different application scenarios, the localization approaches also vary. In this section, we will review and compare both traditional and emerging localization techniques.

2.1.1 Global Navigation Satellite System

Global Navigation Satellite System (GNSS) is the most popular and cost-effective localization method that is used in various mobile applications such as hand-phones, UAVs and Unmanned Surface Vehicles (USVs). A GPS device collects microwave signals from the satellites orbiting the earth and achieves the localization by trilateration [14]. However, obstacles such as mountains and buildings will block and weaken GPS signals. Therefore, it is often used in large outdoor environments. A typically localization accuracy of GPS is 3-5 meters.

FIGURE 2.1: Operational principle of Differential-GPS [15].

An improved solution to GNSS is called Differential-GPS (D-GPS) which leverages a minimum of two GPS receivers to achieve millimeter-level measurement accuracy [16]. As shown in Figure 2.1, a stationary receiver is set up and calibrated as a reference ground station. The monitoring station on a mobile robot compares its current GPS signal with the correction signal from the reference station, so that the atmospheric variations can be identified and minimized by filtering techniques such as Kalman filter [17]. D-GPS is a mature technology used in air navigation, agriculture, hydrographic surveying, monitoring of surface deformations, etc. In some

robotic applications, it is often used as ground truth in outdoor environments due to its high accuracy, e.g., KITTI dataset [18]. Similar to GNSS, the performance drops in the presence of high buildings. Moreover, a calibration step is required to set the fixed ground station, which can be difficult in practice.

2.1.2 Motion Capture System

MCS is a popular approach for indoor localization when high accuracy is the first priority. A MCS utilizes multiple cameras from different angles to track the target object and computes the target location by synthesizing different observations. A typical MCS has a localization accuracy of millimeter and an updating frequency of over 100 *Hz* which is ideal for most robotic applications [19]. Therefore, it is popularly used in very difficult experiments, e.g., UAV formation and aggressive flight. Moreover, it is also used as a source of ground truth for performance evaluation such as TUM dataset [20] and EuRoC dataset [21]. Figure 2.2 gives two examples of MCS application. Specifically, Figure 2.2a presents a UAV swarm system in a small area covered by a VICON system¹. And Figure 2.2b shows an example of Wi-Fi based human gesture recognition via end-to-end training, where the human pose data is collected from the VICON system.

Despite its superior performance, the flexibility and mobility are major concerns. The coverage area of a MCS is often limited to a small room size and the whole system requires complicated calibration upon installation. In addition, the cost of setting up a MCS is several times higher than other localization techniques. Therefore, MCS is often used in some small scale proof-of-concept experiments rather than the practical applications.

2.1.3 Ultra-Wideband Localization

UWB-based localization is an emerging localization technique in recent years. It attracts great research attention due to its high mobility in both indoor and outdoor environments. UWB communication technology enables the measurement of relative localization via either Time Difference of Arrival (TDOA) or Time of Flight (TOF) [24]. Figure 2.3 illustrates the basic operational principle. Stationary UWB

¹<https://www.vicon.com/>

(A) UAV formation experiment in an indoor environment with VICON system [22].

(B) WiFi-based 3D human pose recognition that is achieved by supervision from a VICON system [23].

FIGURE 2.2: MCS used in different research scenarios.

anchors are set up in the field and the on-board UWB tag measures its distance towards the anchors. The relative localization can be decomposed by trilateration methods. The localization accuracy of UWB ranges from cm -level to $1\ m$ with a coverage area from $10\ m$ to $100\ m$. Similar to other wireless communication technology, UWB signals can be blocked by concrete or metals along Line of Sight (LOS). Thus it is often considered as a cost-effective source of localization when the localization area is fixed and free from barriers.

FIGURE 2.3: Operational principle of UWB-based localization.

2.1.4 Wireless Local Area Networks

Wireless Local Area Networks (Wi-Fi) is another emerging localization method via wireless communication techniques. A Wi-Fi receiver measures its signal strength that is received from nearby routers and converts to relative distance measurement by physical model of signal attenuation [25]. The location can be achieved in a way similar to UWB-based localization. It is reported that the Wi-Fi localization is able to provide localization accuracy at deci-meter level from the recent research [26]. Due to the popularity of the Internet, the Wi-Fi localization is often used in indoor commercial areas such as airports or shopping centers, where the Wi-Fi service is readily available.

2.1.5 Other Landmark-based Approaches

There are also some other localization techniques used for mobile robots. One of the limitations is the identification and tracking of landmarks. A QR code is highly distinguishable and has a low false detection rate, which makes it ideal as a landmark. Once the physical dimension of QR code is known, the relative position can be derived from the camera view. Therefore, as shown in Figure 2.4, QR code is also one of the localization methods for indoor robots. Based on the density of

QR codes, the localization accuracy varies from cm to sub-meter. One limitation is that the exact location of each QR code has to be input as the prior knowledge for mobile robots.

FIGURE 2.4: QR code localization for humanoid robot [27].

Magnetic stripe tracking is another localization method used for industrial robots. As shown in Figure 2.5, a magnetic track is pre-installed in the warehouse area. The magnetic sensor at the bottom of AGV will measure the robot location relative to the center of the track and provide the information to the motor controller to adjust the steering, so that the vehicle remains at the center of the track. The magnetic stripe is widely used in the conventional warehouses and has a high localization accuracy of 1-2 cm. However, the installation of magnetic stripe for the entire warehouse area is often costly, and once installed, no further modification is allowed.

2.1.6 Simultaneous Localization And Mapping

Simultaneous Localization And Mapping (SLAM) often refers to the task for a mobile robot to construct a map of an unknown environment while simultaneously localizing itself. A SLAM problem targets to solve the robot location via single or

FIGURE 2.5: AGV localization via magnetic stripes [28].

multiple on-board sensors with no dependency on external infrastructure setup. As firstly introduced by Smith Self and Cheeseman in 1988 [29], SLAM is considered as a filtering problem between motion and observation to minimize the uncertainty of the robot pose. The robot motion can be estimated via wheel odometer or IMU, while its observation is measured by perception sensors such as ultrasonic sensor and infrared sensor. By assuming Gaussian noise, EKF-SLAM [30] is introduced to solve SLAM problems via Extended Kalman Filtering (EKF), and the solution is proven to be optimal. It was a popular solution in the early days. In the 21st century, with the fast development of the robotic industry, robotic applications are no longer limited to small-scale environments. New robotic applications such as UAVs and ADS require localization in a large-scale environment. In such case, traditional EKF-based methods become computationally infeasible since new robot states and landmarks are continuously added into pose estimation with a computational complexity of $\mathcal{O}(n^2)$. In addition, the perception sensors have become more complicated, e.g., 3D LiDAR has thousands of observations at each scan input, which makes it infeasible to build observation models. Inspired by Structure From

Motion (SFM) method in computer vision area, Bundle Adjustment (BA) is introduced to SLAM. The SLAM problem is subsequently treated as an optimization problem rather than a filtering problem. Optimization-based approaches have become a mature solution to the SLAM problem nowadays. For example, one of the most well-known V-SLAM systems, ORB-SLAM, optimizes the reprojection error between visual features [31].

With the rapid development of both hardware and software over the past decades, the performance of SLAM has made significant progress. In hardware aspects, more reliable and accurate perceptual sensors are introduced, e.g., from low-resolution monocular camera to hyper-resolution stereo camera, from short-range 2D laser scanner to long-range 3D LiDAR. The Central Processing Unit (CPU) has also evolved from low-capability, single-core micro-controller to high-capability, multi-core embedded computer. The improvements on hardware have made SLAM feasible in actual environments. In terms of software, new frameworks have been proposed such as Large Scale Direct SLAM (LSD-SLAM) and IMplicit Least Squares SLAM (IMLS-SLAM). Due to these facts, SLAM has achieved many impressive results. For example, recent research on ORB-SLAM3 shows an average localization accuracy of 3.6 *cm* with only one stereo camera which is better than many existing localization techniques [32]. In addition, LiDAR Odometry And Mapping (LOAM) has achieved an average translational error of 0.55% in large-scale outdoor driving tests. SLAM is now a mature technology used in various robotic applications due to the high mobility and robustness. These SLAM applications include but not limited to:

- Augmented Reality/Virtual Reality. The SLAM technology provides a 3D mapping which is more authentic. The SLAM is used in some of the latest VR product to enhance the authenticity such as Google Tango² and Magic Leap³.
- UAV industry. SLAM can provide high-accuracy localization for UAV applications, e.g., precision landing and bridge inspection. Moreover, the UAV navigation is achieved by doing path planning on the constructed local map.

²<https://developers.google.com/ar>

³<https://www.magicleap.com/en-us>

- Autonomous driving. Most autonomous driving cars require high localization accuracy for safety purpose, e.g., Autonomous Valet Parking (AVP). The localization is often achieved by sensor fusion between GPS and SLAM.
- Smart Manufacturing. The SLAM technology frees AGVs from the traditional magnetic-track-based navigation, which is essential towards building a re-configurable smart factory.

2.1.7 Feasibility Analysis of Different Approaches

The design of a practical robotic system is a comprehensive process since localization accuracy is not the only factor to be considered. Many real-world applications are limited by numerous constraints such as lighting conditions, computational power and dynamic objects. Therefore, it is necessary to evaluate the feasibility in practical applications during selection of localization techniques. Table 2.1 lists the advantages and disadvantages of different localization techniques, including GPS, MCS, D-GPS, UWB, WiFi, SLAM, QR code and magnetic stripe. It can be seen that D-GPS and MCS have the highest accuracy. All methods except for SLAM and GPS require the installation of external devices. In many applications, the setup of fixed anchors is not feasible, e.g., cleaning robot, autonomous driving. In such cases, SLAM is a feasible solution with relatively high accuracy and large coverage area.

2.2 SLAM in Unknown Environments

Based on the on-board perception sensor type, the SLAM is divided into LiDAR SLAM and Visual SLAM [33]. In short, the LiDAR SLAM mainly leverages the geometric feature while Visual SLAM focuses on visual feature tracking. In this section we will review some state-of-the-art localization methods used in both LiDAR SLAM systems and Visual SLAM systems.

Technology	Localization Accuracy	Coverage Area	Indoor/Outdoor	Comments
GPS	Low	Large	Outdoor	Provide absolute localization
Differential GPS	Very High	Large	Outdoor	Setup of stationary reference GPS is required
Motion Capture System	Very High	Small	Indoor	High cost, system setup is fixed
UWB Localization	Moderate	Medium	Indoor & Outdoor	Setup of UWB anchors is required
Wi-Fi Localization	Low-Moderate	Medium	Indoor	Dense Wi-Fi networks are required
SLAM	High	Large	Indoor&Outdoor	No dependency of external setup, high computational cost
QR Code	Moderate	Small	Indoor	Setup of QR codes is required
Magnetic Stripe	High	Medium	Indoor	High cost, low flexibility

TABLE 2.1: Comparison of different localization approaches.

2.2.1 Visual Odometry

VO is a 2D-to-2D matching problem between images. The pose estimation can be solved by finding the essential matrix via an eight-point algorithm [34]. The matching error comes from many aspects in practice, e.g., false feature association due to the illumination change, calibration error of intrinsic parameter, measurement noise from camera, etc. Most state-of-the-art methods use optimization-based approaches to achieve better results. Feature-based and appearance-based approaches are the two main techniques used in VO, which target to minimize reprojection error and photometric error respectively.

2.2.1.1 Feature-based Approach

Visual features are mature techniques used for recognition and tracking in computer vision due to their superior properties such as translation, rotation or scaling invariance. An identical landmark has different projections on the image plane over consecutive observations. It is often formulated as a BA problem, which solves

FIGURE 2.6: Illustration of the BA problem [35]. The solid line represents the projection procedure, while the dotted line represents the reprojection procedure.

VO by keeping track of multiple features and minimizing the reprojection error. As shown in Figure 2.6, $\mathbf{p}_{X_i}$ is the landmark location of X_i in world coordinate, C_j is the camera pose, $\mathbf{u}_{ij}$ is the reprojected 2D image point of landmark X_i on camera pose C_j , $\mathbf{u}'_{ij}$ is the observation of X_i by camera pose C_j . Note that to be homogeneous, the 2D point in the image plane is represented as $\mathbf{u}_{ij} = [u_x, u_y, 1]$. The location of point $\mathbf{p}_{X_i}$ is related to reprojected 2D point $\mathbf{u}_{ij}$ by

$$\mathbf{u}_{ij} = \mathbf{K}^c \cdot \frac{1}{Z_{ij}} \cdot \mathbf{p}_{ij}, \quad (2.1)$$

where Z_{ij} is the depth of X_i in the local coordinate of C_j , $\mathbf{p}_{ij}$ is the landmark position $\mathbf{p}_{X_i}$ in the camera coordinate of C_j , and $\mathbf{K}^c$ is the calibration matrix given by

$$\mathbf{K}^c = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (2.2)$$

where f_x , f_y , c_x and c_y are the intrinsic parameters of the camera. The reprojection error is thus represented as

$$\mathbf{e}_{ij} = \mathbf{u}'_{ij} - \mathbf{u}_{ij}. \quad (2.3)$$

The objective of BA is to minimize the reprojection error by the least squares method

$$J = \sum_i \sum_j \mathbf{e}_{ij}^T \cdot \mathbf{e}_{ij}. \quad (2.4)$$

This is a non-linear and non-convex problem. Denote the parameter space as $\mathbf{x}$ with both camera pose and landmark location, the above equation can be re-written by Taylor expansion

$$J = \mathbf{e}(\mathbf{x})^T \cdot \mathbf{e}(\mathbf{x}) \approx (\mathbf{e}(\mathbf{x}) + \mathbf{J}_e \cdot \delta x)^T \cdot (\mathbf{e}(\mathbf{x}) + \mathbf{J}_e \cdot \delta x), \quad (2.5)$$

where $\mathbf{J}_e$ is the Jacobian matrix. The solution to the least squares problem is thus calculated by

$$\mathbf{J}_e^T \cdot \mathbf{J}_e \cdot \delta x = -\mathbf{J}_e^T \cdot \mathbf{e}(\mathbf{x}). \quad (2.6)$$

Thus we can update the optimal estimation by $\mathbf{x} \leftarrow \mathbf{x} + \delta x$ till convergence. BA is the major method used in SLAM such as ORB-SLAM [31] and Parallel Tracking And Mapping (PTAM) [36].

2.2.1.2 Appearance-based Approach

Another way to solve the pose estimation is to perform pixel-wise matching. Appearance-based method formulates the data association and the pose estimation into a single non-linear optimization problem rather than separating them into two independent topics. It is based on the assumption that in short duration, the light intensity that is observed in the camera does not change. Therefore, when the two images are properly aligned, the intensity difference is minimized. We reuse Figure 2.6 for illustration, while let $\mathbf{u}_{i1}$ be the observation of X_i in C_1 and $\mathbf{u}_{i2}$ be the observation of X_i in C_2 . To calculate the transformation between camera pose C_1 and C_2 , the photometric error is represented as

$$\mathbf{e}_i = I_1(\mathbf{u}_{i1}) - I_2(\mathbf{u}_{i2}) = I_1\left(\frac{1}{Z_{i1}} \mathbf{K}^c \mathbf{p}_{i1}\right) - I_2\left(\frac{1}{Z_{i2}} \mathbf{K}^c (\mathbf{R} \mathbf{p}_{i1} + \mathbf{t})\right), \quad (2.7)$$

where $I_j(\mathbf{u})$ is the pixel value of 2D point $\mathbf{u}$ in image observed by camera pose C_j , $\mathbf{R}$, $\mathbf{t}$ are the rotation and translation between C_1 and C_2 , Z_{ij} is the depth of X_i in the local coordinate of C_j , $\mathbf{p}_{i1}$ is the landmark position $\mathbf{p}_{X_i}$ in the camera coordinate of C_1 , $\mathbf{K}^c$ is the calibration matrix respectively. The equation can also be solved by applying linearization and Gaussian-Newton method similar to Equation (2.4).

The derivative can be derived by applying chain rule with the intensity derivative taken as image gradient

$$\mathbf{J}_I = \frac{\partial I}{\partial \mathbf{u}} \cdot \frac{\partial \mathbf{u}}{\partial \mathbf{p}_{i1}} \cdot \frac{\partial \mathbf{p}_{i1}}{\partial \xi}, \quad (2.8)$$

where $\frac{\partial \mathbf{p}}{\partial \xi}$ is the derivative of a 3D point w.r.t Lie Algebra.

Appearance-based approaches are often more computationally efficient compared to feature-based approaches, because feature extraction and matching are not necessary, e.g., DSO [37] and LSD-SLAM [38]. With the assumption of intensity consistency from the environment, they still require a photometric camera calibration to minimize the measurement noise in order to achieve competitive results [39]. For example, the image intensity measured is related to camera exposure time. Hence, appearance-based approaches cannot be directly applied to a video sequence that is acquired with an unknown camera or a camera with auto-exposure function. In addition, its performance can be severely degraded by unmodeled effects like rolling shutter or non-lambertian reflectance [31].

2.2.2 LiDAR Odometry

LiDAR SLAM focuses on the geometric characteristics of a place. In computer vision, visual feature extraction often requires the analysis of local image gradients. Similarly, many local geometric features need to compute the local geometry norm, e.g., FPFH [40] and SHOT [41]. However, it is not as straightforward as image analysis since point cloud is unordered. As a result, it is often computationally inefficient to extract and track local geometric features. Most LO methods use point cloud matching rather than feature-based matching. In this section we will review some of the common approaches used for point cloud matching.

2.2.2.1 Iterative Closest Point

ICP algorithm [42] remains as one of the most popular methods used for point cloud matching. For a source point cloud $\mathcal{P}_s$ and a target point cloud $\mathcal{P}_t$, ICP iterates in three steps: (1) a matching step where correspondences are estimated by nearest-neighbor search from target point cloud; (2) an optimization-based estimation step

to minimize the point-to-point distance that is represented by

$$\arg \min_{\mathbf{R}, \mathbf{t}} \sum_{\mathbf{p} \in \mathcal{P}_s, \mathbf{p}' \in \mathcal{P}_t} \|\mathbf{p}' - \mathbf{R}\mathbf{p} - \mathbf{t}\|. \quad (2.9)$$

(3) re-align the target point cloud with the result from the previous step. As shown in Figure 2.7, with a proper initial alignment guess, the estimated result converges to an optimal alignment.

FIGURE 2.7: Matching of two point cloud by ICP⁴.

ICP algorithm provides good results when the initial guess is proper. The interpretation of ICP is straightforward; therefore, ICP-based SLAM has been implemented into many applications such as [43] and [44]. However, ICP is sensitive to noise as the data association may not be ideal with the existence of numerous outliers. Moreover, the matching process is computationally inefficient since all points are used at each iteration.

2.2.2.2 Normal Distribution Transformation

Normal Distribution Transformation (NDT) is another scan matching method, which transforms a point cloud into a smooth surface representation. It was firstly proposed for 2D scan registration in 2003 [45] and later used in SLAM systems such as IRON [46] and NDT-Graph-SLAM [47]. The space occupied by a laser scan can be divided into grid cells, which are represented by squares for 2D LiDAR and cubes for 3D LiDAR. For each cell, a Probability Density Function (PDF) is

⁴<http://www.sanko-shoko.net>

calculated to describe the shape of the local surface. More specifically, for a grid cell of m points, its PDF is taken as Normal Distribution (ND) with mean and covariance represented by

$$\begin{aligned}\mu &= \frac{1}{m} \sum_i^m \mathbf{p}_i, \\ \Sigma &= \frac{1}{m-1} \sum_i^m (\mathbf{p}_i - \mu)(\mathbf{p}_i - \mu)^T,\end{aligned}\tag{2.10}$$

where $\mathbf{p}_i$ is the point within the cell. The probability of a 3D point $\mathbf{x}$ in the cell is thus represented as

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{3/2} \sqrt{|\Sigma|}} \exp\left(-\frac{(\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)}{2}\right).\tag{2.11}$$

Therefore, the point cloud is represented as a probability distribution with a set of ND. Figure 2.8 shows an example of 3D NDT transform result of a point cloud.

FIGURE 2.8: Example of 3D-NDT representation for a tunnel section [48]. Brighter and denser parts represent higher probabilities. (a) original point cloud; (b) NDT result.

The target of NDT-based scan-match is thus to find the pose to maximise the likelihood so that the points of the current scan lie on the target scan by

$$\arg \min_{\mathbf{R}, \mathbf{t}} \prod_{\mathbf{p}_i \in \mathcal{P}_s} p(\mathbf{R}\mathbf{p}_i + \mathbf{t}),\tag{2.12}$$

where $\mathbf{R}$ and $\mathbf{t}$ are the rotation and translation shifts between source and target frames. This is equivalent to the optimization of the log-likelihood function

$$\arg \min_{\mathbf{R}, \mathbf{t}} \sum_{\mathbf{p}_i \in \mathcal{P}_s} \log(p(\mathbf{R}\mathbf{p}_i + \mathbf{t})). \quad (2.13)$$

Since function $p(\mathbf{x})$ is a Gaussian function, the coefficient $\log \frac{1}{(2\pi)^{3/2} \sqrt{|\Sigma|}}$ is a constant and can be ignored. The problem can be further simplified as

$$\arg \min_{\mathbf{R}, \mathbf{t}} \sum_{\mathbf{p}_i \in \mathcal{P}_s} -\frac{(\mathbf{x} - \mu)^T \Sigma^{-1} (\mathbf{x} - \mu)}{2}. \quad (2.14)$$

This is a non-linear least squares optimization that can be solved by Gaussian-Newton method. In summary, NDT-based scan-matching consists of the following steps: (1) calculate NDT of the target scan; (2) find the data association by aligning the source scan to the grid in the target scan; (3) search for an optimal transform by solving a Maximum Likelihood Estimation (MLE) problem; (4) repeat step (2) and (3) until it converges. As mentioned in the last section, it is computationally expensive to search for nearest points during each iteration. Compared to ICP, NDT is only computed during the initialization, and the data association is much more computationally efficient than the nearest point searching. Therefore, the NDT-based scan-matching also exhibits good robustness to error in the initial pose estimate, especially regarding the rotation component.

2.2.2.3 Line and Plane Features Tracking

In an urban area, man-made architectures are often rich in line and plane features, e.g., buildings and trucks. These features are more consistent over consecutive scans and are less affected by measurement noise. Moreover, existing methods such as ICP can hardly achieve real-time performance on mobile robots. In comparison, representing the environment as a set of lines and planes is more efficient. In 2014, a novel framework using line and plane features was introduced and achieved real-time performance on hand-held tests[49]. It has become a major scan-matching method for LiDAR SLAM afterwards, e.g., LeGO-LOAM [50], V-LOAM [51]. The points are separated into edge features $\mathcal{E}$ and planar features $\mathcal{S}$ based on the local curvature which is collected from the nearest 5 points. Edge features are often the

points from lines, and planar features are often the points from flat surfaces. They are stored in a global map.

FIGURE 2.9: Illustration of global edge and plane estimation. (a) global edge estimated from two edge features; (b) global surface estimated from three planar features [49].

To match a current scan $\mathcal{P}_c$ with the global feature map $\mathcal{P}_m$, the edge features are aligned with nearest lines and planar features are aligned with nearest planes. This can be achieved by minimizing point-to-edge and point-to-plane distance. As illustrated in Figure 2.9, for each edge feature $\mathbf{p}_e$ from the current scan, we can search for two nearest edge features $\mathbf{p}_l, \mathbf{p}_j$ from the global map. The global line is estimated as the line crossing points $\mathbf{p}_l$ and $\mathbf{p}_j$. The distance between $\mathbf{p}_e$ and the line is thus calculated by

$$d_e(\mathbf{R}, \mathbf{t}, \mathbf{p}_e) = \frac{\|(\mathbf{p}_e - \mathbf{p}_l) \times (\mathbf{p}_e - \mathbf{p}_j)\|}{\|\mathbf{p}_l - \mathbf{p}_j\|}, \quad (2.15)$$

where the symbol $\times$ is cross product. Similarly, for each planar feature $\mathbf{p}_s$ in the current scan, we can search for three nearest plane features $\mathbf{p}_l, \mathbf{p}_j, \mathbf{p}_m$ from the global map. The global plane is estimated as the surface crossing points $\mathbf{p}_l, \mathbf{p}_j$ and $\mathbf{p}_m$ if they are not colinear. The distance between $\mathbf{p}_s$ and the plane is thus calculated by

$$d_s(\mathbf{R}, \mathbf{t}, \mathbf{p}_s) = (\mathbf{p}_s - \mathbf{p}_m) \cdot \frac{(\mathbf{p}_m - \mathbf{p}_l) \times (\mathbf{p}_m - \mathbf{p}_j)}{\|(\mathbf{p}_m - \mathbf{p}_l) \times (\mathbf{p}_m - \mathbf{p}_j)\|}. \quad (2.16)$$

Thus the objective becomes an optimization problem given by

$$\arg \min_{\mathbf{R}, \mathbf{t}} \sum_{\mathbf{p}_e \in \mathcal{E}} d_e(\mathbf{R}, \mathbf{t}, \mathbf{p}_e) + \sum_{\mathbf{p}_s \in \mathcal{S}} d_s(\mathbf{R}, \mathbf{t}, \mathbf{p}_s). \quad (2.17)$$

The non-linear optimization can be solved by an iterative calculation similar to ICP. The idea of using line and plane tracking has several advantages. Compared

to geometric features such as FPFH and SHOT, the extraction of edge and planar features is more computationally efficient. Compared to the ICP method, it is less sensitive to noise. It is also worthy mentioning that, existing works such as LOAM have achieved very impressive results on public dataset ranking⁵, which are more accurate than traditional scan-matching methods.

2.2.3 Limitations of Existing Localization Methods in Applications

Computational resources are much limited in robotic applications such as UAVs or cleaning robots. Many existing methods are difficult to be integrated into mobile robots due to their huge computational cost. In LiDAR Odometry, the extraction of geometry features such as FPFH is often slow since the point cloud is unordered. It often takes a few seconds to process a frame. Similar problem also happens in BA, where the dimension of the optimization matrix grows with more landmarks and poses added. In recent years, many emerging robotic applications require faster localization updates for navigation tasks, e.g., autonomous driving cars and UAVs. Existing SLAM systems such as ORB-SLAM can only support a few *Hz*, which is not enough. Therefore, it remains a challenging topic to search for a computationally efficient and accurate SLAM algorithm for these new robots.

2.3 Loop Closure Detection and Re-Localization

The back-end SLAM mainly consists of loop closure detection and global optimization. Loop closure detection is the task to identify a re-visited place. With loop information, the localization drifts can be corrected by global optimization. The global optimization can be performed by factor graph optimization which is a mature topic. However, the loop closure detection remains a challenging topic in terms of computational cost and detection robustness.

⁵http://www.cvlibs.net/datasets/kitti/eval_odometry.php

2.3.1 Graph-based Optimization

The robot's trajectory is often interpreted as a graph representation

$$\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}, \quad (2.18)$$

where $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the set of edges. In a SLAM problem, vertex is the pose of robot $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ and edge refers to the observation of transition between two vertices $\mathbf{T}_{ij} \in SE(3)$. As shown in Figure 2.10, the edges of consecutive frames are provided by VO or LO which are plotted in black, and the edges of non-consecutive frames are provided by loop closure detection which are plotted in red. The objective of back-end SLAM is to optimize the robot trajectory

FIGURE 2.10: Illustration of graph-based optimization [52].

with the help of the loop closure detection, which is to minimize the sum of the squared error function [53]

$$F(\mathbf{X}) = \sum_i \sum_j f(\mathbf{x}_i, \mathbf{x}_j)^T \mathbf{\Omega}_{ij} f(\mathbf{x}_i, \mathbf{x}_j), \quad (2.19)$$

where $\mathbf{\Omega}_{ij}$ represents the information matrix of a constraint that is related to the parameters $\mathbf{x}_i$ and $\mathbf{x}_j$. The error function $f(\mathbf{x}_i, \mathbf{x}_j)$ is expressed as

$$f(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{T}_{ij}^{-1}(\mathbf{x}_i^{-1} \mathbf{x}_j). \quad (2.20)$$

The problem can be solved by linearization of function $\mathbf{F}(\mathbf{X})$:

$$\begin{aligned}
 F(\mathbf{X} + \delta\mathbf{X}) &= f(\mathbf{X} + \delta\mathbf{X})^T \boldsymbol{\Omega} f(\mathbf{X} + \delta\mathbf{X}) \\
 &\approx (f(\mathbf{X}) + \mathbf{J}(\mathbf{X})\delta\mathbf{X})^T \boldsymbol{\Omega} (f(\mathbf{X}) + \mathbf{J}(\mathbf{X})\delta\mathbf{X}) \\
 &= f(\mathbf{X})^T \boldsymbol{\Omega} f(\mathbf{X}) + 2f(\mathbf{X})^T \boldsymbol{\Omega} \mathbf{J}(\mathbf{X})\delta\mathbf{X} + \delta\mathbf{X}^T \mathbf{J}^T \boldsymbol{\Omega} \mathbf{J}(\mathbf{X})\delta\mathbf{X},
 \end{aligned} \tag{2.21}$$

where $\boldsymbol{\Omega}$ is the inverse covariance matrix formed by $\boldsymbol{\Omega}_{ij}$ and $\mathbf{J}$ is the Jacobian matrix of error function $F(\mathbf{X})$. This is a least squares optimization problem that can be solved by the Gaussian-Newton method. One of the examples using the pose graph optimization is shown in Figure 2.11. The dataset is collected at MIT Killian Court (courtesy of Mike Bosse and John Leonard) [54]. Figure 2.11 (a) shows the map that is obtained by rendering the laser scans according to the estimated robot positions in the graph. The loop closure detection results are marked as black lines on the left image. According to loop closure detection results, we can add the pose graph constraints, and the optimized mapping result is shown in Figure 2.11 (b). It can be seen that the mapping result after global optimization is more reasonable and meaningful towards robotic applications.

FIGURE 2.11: Global optimization with constraints from loop closure detection. (a) mapping result without loop closure; (b) mapping result after global optimization.

2.3.2 Loop Closure Detection via Visual Features

Among different perceptual sensors, a camera is closest to a human’s visual perception. With the rapid development of computer vision, image based LCD has become one of the main techniques used. In SLAM, the number of retrievals grows linearly with database size, leading to an incremental computational cost. Therefore, it is impossible to deal with raw image content under limited computational resources. Most LCD methods aim to translate an image content into more concise and precise feature representation and then perform retrieval on the feature space. These methods include appearance-based, deep-learning-based and saliency-based matching.

2.3.2.1 Appearance-based Matching

Appearance-based matching often extracts an image into some hand-crafted visual features. However, an image often contains many high-dimensional local features and it is often infeasible to apply brute-force matching between local features. Bag of Words (BoW) model is one of the common ideas used for local feature matching. As shown in Figure 2.12, a visual vocabulary is constructed based on the occurrence counts of local features from offline training. Each visual feature is associated with a specific location in the visual vocabulary. Therefore, an image with a certain amount of local features can be represented as a sparse histogram according to visual vocabulary. The comparison of the unordered visual features between two images is thus simplified as a comparison of two histograms.

The idea of BoW is commonly used in existing works, while the way of building a visual vocabulary can be different. One of the most well-known methods is FAB-MAP [55, 56], which detects loops with an omnidirectional camera. It uses SURF as a visual descriptor and utilizes Chow-Liu tree [57] to approximate the co-occurrences between feature points. A visual vocabulary of SURF features is trained by hierarchical k-means clustering to discretize its high dimensional representation. The result is tested with a dataset of 1,000 km in length, and achieves a satisfactory recall rate with only a few false positives [56]. However, the extraction of SURF features requires significant computational resources and hence this method is hard to implement on robots. Similar idea is employed in DBoW2 [58], in which the binary descriptor ORB [59] is used instead. The speed is greatly

improved due to the low complexity in computing the ORB descriptor. The result shows that it is able to work with real-time SLAM such as ORB-SLAM [31].

FIGURE 2.12: BoW model for LCD [60].

The BoW model requires complicated offline training to build visual vocabulary, which is troublesome in practice. For example, in FABMAP, it takes a few hours to train such visual vocabulary on a desktop computer. Moreover, the training data cannot contain any loop and has to be collected in a similar environment in order to achieve a good result. Instead of building an offline trained visual vocabulary to discretize local feature space, some recent works introduce more efficient feature retrieval to make online training possible. For example, Emilio *et al.* [61] introduce a vocabulary maintenance strategy called dynamic island that groups similar images. The dynamic island identifies repetitive registration of same descriptor across multiple frames. The database grows slower by removing those redundant descriptors. Schlegel *et al.* [62] introduce an online trained Hamming distance embedded binary search tree (HBST) to place retrieval. It is much faster than traditional Fast Library for Approximate Nearest Neighbors (FLANN) matching [63] method. However, the memory cost is huge since raw descriptors are used to build the incremental visual vocabulary tree.

Applying dimensional reduction on local features is an alternative solution. Carrasco *et al.* [64] adopt the idea from image encoding and extract local descriptors

into binary codes. Hash coding is applied to the local features array that is collected from a single image. As a result, the extracted features are more compact. Gehrig *et al.* [65] directly apply principal component analysis (PCA) for BRISK [66] feature. Instead of searching from the pre-trained visual vocabulary, k-nearest neighbor (K-NN) search is performed on the projected descriptors. It achieves faster speed at the level of millisecond for each query. However, these methods require local feature extraction, which is computationally expensive.

2.3.2.2 Deep-learning-based Approach

The recent success of Convolutional Neural Network (CNN) in computer vision [67] has triggered another research trend for visual place recognition. For example, in [68], a multi-scale feature encoding method is introduced by training two CNN architectures. In this work, the authors use a pre-trained CNN model to find weather/season-invariant features. Over 2.5 million images are used for training process, which are collected from more than 1000 places. Each place has hundreds of exemplars under different weather and season conditions. As shown in Figure 2.13, the trained CNN model is able to extract the same Region Of Interest (ROI) from the image under different weather conditions. It subsequently leads to a significant improvement on the performance of visual place recognition. Deep learning has also shown its great performance in detecting loop closure under different visual conditions in many other works. For example, Tang *et al.* use deep learning to solve the place recognition problem under seasonal change [69]; Yin *et al.* use deep learning to solve the place recognition problem under different weather conditions [70]; Sourav *et al.* use deep learning to solve the day-to-night place recognition problem [71].

Some other works target to use the CNN model to improve from the traditional hand-crafted features. Inspired by the traditional image retrieval method, Relja *et al.* propose the NetVLAD [72] to learn CNN parameters in an end-to-end manner for place recognition. The NetVLAD is a new generalized VLAD layer, inspired by the Vector of Locally Aggregated Descriptors (VLAD) which is commonly used in visual place recognition. As shown in Figure 2.14, a new weakly supervised ranking loss is introduced to learn the parameters of the CNN model, while the training data is collected from the Google street view time machine. The experiment results

FIGURE 2.13: Training a CNN framework with a massive (2.5M images) place recognition-specific dataset. The dataset contains more than 1000 places with hundreds of exemplars of each place under changing conditions. The trained CNN models is able to learn condition-invariant features for place recognition. (a) the CNN model used for training; (b) feature heat maps overlapped on top of the testing images. The images are collected on the same place but different seasons [68].

show that the proposed architecture outperforms the existing hand-crafted feature descriptors on standard visual place recognition benchmarks.

FIGURE 2.14: CNN architecture with the NetVLAD layer [72].

There are also some other works attempting to combine both feature-based methods and deep-learning based methods. Hausler *et al.* [73] combine both feature-based methods and the CNN technique. As shown in Figure 2.15, the query image is trained by combining sum of absolute difference (SAD), Histogram of Oriented gradients (HOG) [74], CNN spatial Max pooling and CNN spatial Arg-Max pooling. A feature vector is extracted by each method. It is subsequently compared with the database to find a distance vector, which indicates the similarity. However, the CNN-based methods achieve high accuracy at the expense of a high computational burden, which is sometimes infeasible for real-time systems due to the requirement of GPU with high power consumption.

FIGURE 2.15: The formulation of the emission matrix using the combination of both traditional methods and deep-learning based methods [73].

2.3.2.3 Saliency-based Approach

Compared to feature matching, saliency/landmark recognition is a more straightforward way for place representation. In [75], Kim *et al.* adopt inter-image saliency for place recognition with bag of words, which are shown to be effective in tackling challenging scenarios such as underwater ship hull inspection. It only takes a few milli-seconds to extract those salient patches from the image and outperforms feature based extraction in terms of speed. Birem *et al.* extract salient regions through a conspicuity map from intensity, color and orientation [76]. The similarity of image frames is measured based on the content and location of salient

regions. Taking advantage of the feature extraction ability of CNN, Xin *et al.* investigate how to localize discriminative visual landmarks that positively contribute to the similarity measurement [77]. In particular, a landmark localization network is designed to indicate the discriminative regions of an image. It has been shown that the saliency re-identification-based methods are more robust than the feature-based methods. However, it has not been widely applied to robotic systems since they are still too computationally expensive for robotic systems.

2.3.3 Loop Closure Detection via Geometric Features

Visual features are often susceptible to lighting conditions. In comparison, LiDAR detection is more robust due to its illumination and view-point invariant properties. However, the utilization of geometric features is more complicated than visual features and there are very limited works on LiDAR-based LCD. Another limitation is the computational cost since the point cloud is unordered and searching on nearby points is often slow. Existing works on LiDAR-based LCD target to find an efficient descriptor or place signature that can accurately and concisely present a place.

Fast Point Feature Histogram (FPFH) [40] is a common geometric feature used in LCD with low computational complexity at $\mathcal{O}(nk)$, where n is the number of points in point cloud and k is the number of nearby points used to calculate a single FPFH. As shown in Figure 2.16, for each query point $\mathbf{p}_q$ with normal $\mathbf{n}_q$ we can search for k nearest neighbour points (enclosed by the gray circle) within a radius r . For each neighbour point $\mathbf{p}_i$ its estimated normal $\mathbf{n}_i$, define a Darboux *uvn* frame ($\mathbf{u} = \mathbf{n}_i$, $\mathbf{v} = (\mathbf{p}_q - \mathbf{p}_i) \times \mathbf{u}$, $\mathbf{w} = \mathbf{u} \times \mathbf{v}$), the angular variations of $\mathbf{n}_i$ and $\mathbf{n}_q$ are computed as follows

$$\begin{aligned}\alpha &= \mathbf{v} \cdot \mathbf{n}_q, \\ \phi &= \frac{\mathbf{u} \cdot (\mathbf{p}_q - \mathbf{p}_i)}{\|\mathbf{p}_q - \mathbf{p}_i\|}, \\ \theta &= \arctan(\mathbf{w} \cdot \mathbf{n}_q, \mathbf{u} \cdot \mathbf{n}_q).\end{aligned}\tag{2.22}$$

The angular variations are next coded into Simplified Point Feature Histogram (SFPH) represented by function $SFPH(\mathbf{p}_q)$. Similarly, each neighbor is connected to its own neighbors and the resulted histograms $SFPH(\mathbf{p}_i)$ are weighted together

with the histogram of the query point to form the FPFH

$$FPFH(\mathbf{p}_q) = SPFH(\mathbf{p}_q) + \frac{1}{k} \sum_{i=1}^k \frac{1}{w_k} SPFH(\mathbf{p}_i), \quad (2.23)$$

where the weight w_k represents the distance between query point $\mathbf{p}_q$ and a neighbor point $\mathbf{p}_i$ in a given metric space.

FIGURE 2.16: The influence region diagram for a FPFH feature [40].

FPFH is often used to represent a place for LCD. For example, in SegMatch [78], the ground points are firstly removed and the filtered scan input is segmented into the distinctive elements based on the clustering method described in [79]. Each point cloud segment is considered as a landmark for place recognition. To compare segments, FPFH is computed to describe each point cloud. It takes more than 1 s to match two consecutive scans.

Re-identification on the local geometric feature descriptor usually requires key-point extraction and massive local geometry calculation, e.g., local surface normal. In comparison, matching on a global descriptor is more computationally efficient in place recognition. In [80], GLARE is proposed to encode spatial relations of landmarks for LCD. FLIRT features [81] are used to firstly extract a laser scan into feature points. The bearing angle and distance between each feature points pair are incorporated as a multivariate Gaussian and stored into a 2D histogram.

Approximate Nearest Neighbour Search (ANN) is used to efficiently search between high-dimensional histograms. GLARE was firstly used for 2D SLAM and it was later extended to 3D scenario with rotation invariance property taken into consideration [82]. In [83], Kim *et al.* propose scan context image (SCI) which projects laser scans into global descriptors. The 3D point cloud is subdivided into cubical cells by discretizing on a horizontal plane. For each cell, the point with the highest z-axis value is used to represent the height of the current cell. The height information from all cells subsequently forms a 2D image. The matching of scan context only requires element-wise multiplication so that the query speed is fast. However, the matching precision is not high enough and false positives occur during public datasets tests.

Recent works find that the performance can be further improved by integrating the intensity characteristics. Guo *et al.* justify that intensity information can be distinctive for places and propose a novel local descriptor called intensity signature of histograms of orientations (ISHOT) that consists of both geometry and intensity information. The place re-identification is solved by a probabilistic voting strategy similar to [84]. The experiment results show that the new descriptor outperforms geometrical-only descriptors in LCD.

2.3.4 Limitations of Loop Closure Detection in Robotic Applications

In Figure 2.11 we have shown the advantage of using LCD in SLAM. However, there are several challenges in practice. One of the challenges is the online performance. For example, in the BoW model, offline training is required to build a visual vocabulary. To build such visual vocabulary, it is required to collect training data from a similar environment, which is inconvenient. The limited computational resources is another challenge. As more places are visited, more features/landmarks are added into the database. As a result, the computational cost of searching from a database grows. Therefore, it is important to have an efficient place recognition strategy to improve the searching speed. Another challenge is the change of FoV. For example, in autonomous driving, a crossroad can be re-visited in different directions, so that the observed features are slightly different. Hence a robust LCD method should identify viewing angle change as well.

2.4 Mathematical Preliminaries

Before closing the chapter, we introduce some mathematical preliminaries that will be used in the thesis.

2.4.1 Lie Algebra and Lie Group

We recall some useful concepts related to the Special Orthogonal Group $\text{SO}(3)$ and the Special Euclidean Group $\text{SE}(3)$, which are also known as Lie group [85]. $\text{SO}(3)$ defines the rotation matrix between two poses and it is described as $\mathbf{SO}(3) = \{\mathbf{R} \in \mathbb{R}^{3 \times 3} : \mathbf{R} \cdot \mathbf{R}^T = \mathbf{I}, \det(\mathbf{R}) = 1\}$. The group $\text{SO}(3)$ forms a smooth manifold, and the tangent space to the manifold is denoted as $\mathfrak{so}(3)$. $\mathfrak{so}(3)$ is also known as Lie algebra, which is defined by $\mathfrak{so}(3) = \{\phi \in \mathbb{R}^3\}$. The relation between $\mathfrak{so}(3)$ and $\text{SO}(3)$ is defined as

$$\begin{aligned} \mathbf{R} &= \exp(\phi^\wedge) \\ &= \mathbf{I} + \frac{\sin \|\phi\|}{\|\phi\|} \phi^\wedge + \frac{1 - \cos \|\phi\|}{\|\phi\|^2} (\phi^\wedge)^2, \end{aligned} \quad (2.24)$$

where the symbol $\phi^\wedge$ (sometimes also noted as $[\phi]_\times$) is the skew symmetric matrix given by

$$\phi^\wedge = \begin{bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix}^\wedge = \begin{bmatrix} 0 & -\phi_3 & \phi_2 \\ \phi_3 & 0 & -\phi_1 \\ -\phi_2 & \phi_1 & 0 \end{bmatrix}. \quad (2.25)$$

Similarly, the inverse operation from $\text{SO}(3)$ to $\mathfrak{so}(3)$ is defined by

$$\begin{aligned} \log(\mathbf{R}) &= \frac{\theta \cdot (\mathbf{R} - \mathbf{R}^T)}{2 \cdot \sin \theta}, \quad \theta = \cos \frac{\text{tr}(\mathbf{R}) - 1}{2}, \\ \phi &= \log(\mathbf{R})^\vee, \end{aligned} \quad (2.26)$$

where the operator $\cdot^\vee$ is the inverse process of skew symmetric matrix. The Special Euclidean Group is defined as $\text{SE}(3) = \{\mathbf{T} \in \mathbb{R}^{4 \times 4} : \mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix}, \mathbf{R} \in \text{SO}(3), \mathbf{t} \in \mathbb{R}^3\}$. Similar to $\mathfrak{so}(3)$, let $\mathfrak{se}(3) = \{\xi \in \mathbb{R}^6 : \xi = [\rho, \phi]\}$, and we can define the operator $\cdot^\wedge$ as

$$\xi^\wedge = \begin{bmatrix} [\phi]_\times & \rho \\ \mathbf{0}^T & 0 \end{bmatrix}. \quad (2.27)$$

The relation of rigid body transformation matrix $\mathbf{T} \in \text{SE}(3)$ and $\xi \in \mathfrak{se}(3)$ is denoted as

$$\begin{aligned} \mathbf{T}_k &= \exp(\xi_k^\wedge) = \sum_{n=0}^{\infty} \frac{1}{n!} (\xi_k^\wedge)^n \\ &= \sum_{n=0}^{\infty} \frac{1}{n!} \begin{pmatrix} [\phi]_\times & \rho \\ \mathbf{0}^T & 0 \end{pmatrix}^n \\ &= \begin{bmatrix} \sum_{n=0}^{\infty} \frac{1}{n!} ([\phi]_\times)^n & \sum_{n=0}^{\infty} \frac{1}{(n+1)!} ([\phi]_\times)^n \rho \\ \mathbf{0}^T & 1 \end{bmatrix}. \end{aligned} \quad (2.28)$$

2.4.2 Kernel Cross Correlator

Recall that the cross-correlation of two 2D signals $\mathbf{g} = \mathbf{x} * \mathbf{h}$ becomes $\hat{\mathbf{g}} = \hat{\mathbf{x}} \odot \hat{\mathbf{h}}^*$ in the frequency domain, where the operator $*$, $\odot$ and $\mathbf{h}^*$ represent cross-correlation, element-wise multiplication and complex conjugate, respectively. Given correlation output $\mathbf{g}$, the kernel cross-correlation is defined as

$$\hat{\mathbf{g}} = \hat{\boldsymbol{\kappa}}_{\mathbf{z}}(\mathbf{x}) \odot \hat{\mathbf{h}}^*, \quad (2.29)$$

where $\boldsymbol{\kappa}_{\mathbf{z}}(\mathbf{x})$ is a kernel matrix given by

$$\boldsymbol{\kappa}_{\mathbf{z}}(\mathbf{x}) = \begin{bmatrix} \kappa(\mathbf{x}, \mathbf{z}_{11}) & \dots & \kappa(\mathbf{x}, \mathbf{z}_{1n}) \\ \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}, \mathbf{z}_{m1}) & \dots & \kappa(\mathbf{x}, \mathbf{z}_{mn}) \end{bmatrix}, \quad (2.30)$$

with $\kappa(\cdot, \cdot)$ being a kernel function. $\mathbf{x}$ and $\mathbf{z}$ are two 2D matrices. $\mathbf{z}_{ij}$ is the translational shift of $\mathbf{z}$ with i pixel shifts in horizontal and j pixel shifts in vertical direction. For a matrix $\mathbf{z}$ of size $m \times n$, there are $m \times n$ translational shifts in total. Hence we can have $m \times n$ kernel function $\kappa(\mathbf{x}, \mathbf{z}_{11}), \dots, \kappa(\mathbf{x}, \mathbf{z}_{mn})$, which forms a kernel matrix of dimension $m \times n$. Specifically, we use the Gaussian kernel as an example. Let

$$\begin{aligned} \boldsymbol{\kappa}(\mathbf{x}, \mathbf{z}_{ij}) &= \phi(\|\mathbf{x} - \mathbf{z}_{ij}\|^2), \\ \phi(\mathbf{x}) &= \exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right), \end{aligned} \quad (2.31)$$

where σ is a constant. Denote function $\Phi(\cdot)$ as the matrix form of Gaussian kernel function, where each element is a single Gaussian kernel $\phi(\cdot)$

$$\Phi(\mathbf{A}) = \begin{bmatrix} \phi(\mathbf{A}_{11}) & \dots & \phi(\mathbf{A}_{1n}) \\ \vdots & \ddots & \vdots \\ \phi(\mathbf{A}_{m1}) & \dots & \phi(\mathbf{A}_{mn}) \end{bmatrix}, \quad (2.32)$$

with $\mathbf{A}_{ij}$ being the i^{th} row and j^{th} column element of matrix $\mathbf{A}$. Since $\mathbf{z}_{ij}$ is the translational shifts of $\mathbf{z}$, the kernel matrix $\kappa_{\mathbf{z}}(\mathbf{x})$ becomes

$$\begin{aligned} \kappa_{\mathbf{z}}(\mathbf{x}) &= \begin{bmatrix} \phi(\|\mathbf{x} - \mathbf{z}_{11}\|^2) & \dots & \phi(\|\mathbf{x} - \mathbf{z}_{1n}\|^2) \\ \vdots & \ddots & \vdots \\ \phi(\|\mathbf{x} - \mathbf{z}_{m1}\|^2) & \dots & \phi(\|\mathbf{x} - \mathbf{z}_{mn}\|^2) \end{bmatrix} \\ &= \Phi(\|\mathbf{x}\|^2 + \|\mathbf{z}\|^2 - 2 \cdot [\text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})]_{mn}), \end{aligned} \quad (2.33)$$

where $\text{Tr}(\cdot)$ is the matrix trace

$$[\text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})]_{mn} = \begin{bmatrix} \text{Tr}(\mathbf{x}^T \mathbf{z}_{11}) & \dots & \text{Tr}(\mathbf{x}^T \mathbf{z}_{1n}) \\ \vdots & \ddots & \vdots \\ \text{Tr}(\mathbf{x}^T \mathbf{z}_{m1}) & \dots & \text{Tr}(\mathbf{x}^T \mathbf{z}_{mn}) \end{bmatrix}. \quad (2.34)$$

From the 2D correlation theory, we know that $\mathbf{x} * \mathbf{z} = [\text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})]_{mn}$, so that we have

$$\kappa_{\mathbf{z}}(\mathbf{x}) = \Phi(\|\mathbf{x}\|^2 + \|\mathbf{z}\|^2 - 2 \cdot [\text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})]_{mn}) \quad (2.35a)$$

$$= \Phi(\|\mathbf{x}\|^2 + \|\mathbf{z}\|^2 - 2 \cdot \mathbf{x} * \mathbf{z}) \quad (2.35b)$$

$$= \Phi\left(\frac{\|\hat{\mathbf{x}}\|^2}{m \cdot n} + \frac{\|\hat{\mathbf{z}}\|^2}{m \cdot n} - 2 \cdot \mathcal{F}^{-1}(\hat{\mathbf{x}} \odot \hat{\mathbf{z}}^*)\right). \quad (2.35c)$$

This reduces the complexity $\mathcal{O}(N^2)$ of Equation (2.33) to $\mathcal{O}(N \log N)$ of Equation (2.35c), where $N = m \times n$ is the number of pixels in the image. This implies that KCC can also be calculated efficiently due to the element-wise operation in the frequency domain [86].

2.5 Conclusion

In this chapter, relevant researches on different localization techniques have been reviewed. We firstly review the existing localization methods that are commonly used for robot localization, including SLAM, UWB, WiFi, GPS, etc. Then we compare the advantages and disadvantages of different methods. It can be seen that SLAM has several advantages over other localization solutions, e.g., independence of external setup and high localization accuracy. Then we look into the SLAM problem. A SLAM system is divided into two sub-modules: front-end and back-end system. In the front-end SLAM, existing methods do not achieve good balance between computational cost and localization accuracy. Existing works such as IMLS-SLAM achieve good localization results, but typically cost a few seconds to process a frame. It is often impractical for a real-time LiDAR odometry, where more than 10 Hz update frequency is required. In addition, visual odometry suffers from light illumination problem, where the lighting condition change may easily cause localization loss. In the back-end SLAM, the loop closure detection is challenged by the computational cost, since the database is growing with time. In addition, the change in viewing angle is also a big concern for loop closure detection. There are very limited works on LiDAR descriptors which are able to solve the case of reverse visit. Although the SLAM problem is well defined via the theoretical analysis, it can be seen that some issues remain unsolved in practical applications. In practice, real-time performance is one of the highest priorities for a robotic application. Based on the above challenges, this thesis targets to provide a light-weight and robust SLAM framework for practical applications, with improvements in both front-end and back-end systems.

Chapter 3

Lightweight LiDAR Odometry for Autonomous Vehicles

3.1 Introduction

In this chapter, we will explore the potential of using SLAM for the localization and navigation of autonomous vehicles. In Section 2.1.7 we have highlighted several advantages of SLAM compared to other localization approaches. According to the perceptual devices, SLAM can be roughly categorized as LiDAR and Visual SLAM. LiDAR SLAM is often more accurate in pose estimation and is robust to environmental variations such as illumination and weather change [87]. Therefore, we firstly focus on LiDAR-based localization systems in this chapter. To address the problem of limited computational resources in practice, we will adopt several strategies to reduce the computing time and demonstrate the performance on mobile robots.

The main contribution of this chapter is the introduction of a lightweight LiDAR-only localization system for autonomous vehicles. Specifically, we leverage the local geometric information to identify edge and planar features and match them with globally maintained feature maps via data association. As presented in Section 3.3, the proposed framework mainly consists of feature extraction, distortion compensation, pose optimization, and mapping. And we will explain how to efficiently track and align geometric features to achieve real-time localization on a mobile robot. To illustrate the performance of various systems, the proposed framework

is firstly verified in a simulated environment. Then it is also tested on public datasets collected from autonomous driving. It is eventually integrated into our AGV platform designed for warehouse manipulation. It is worthy of noting that the proposed method is one of the most accurate and fastest open-sourced methods on the KITTI benchmark.

This chapter is organized as follows: Section 3.2 reviews the related existing works on LiDAR SLAM approaches and their limitations in practical applications. Section 3.3 describes the details of the proposed framework, including feature point selection, laser points alignment, laser odometry estimation, and mapping. Section 3.4 firstly demonstrates the performance of the proposed system with both simulation validation and public datasets evaluation. Then it is integrated into an AGV for localization and navigation. Section 3.5 concludes this chapter.

3.2 Related Works

The most essential step in LiDAR SLAM is the matching of point clouds. Existing works mainly focus on finding the correspondence between point clouds, including two main categories: raw point cloud matching and feature point pairs matching. For raw point cloud matching, the ICP [88] method is the most classic one. As shown in Section 2.2.2.1, ICP measures the correspondent point by finding the closest point in Euclidean space. And the pose transition between two point clouds converges to the final location by iteratively minimizing the distance residual between corresponding points. An improved point cloud matching is IMLS-SLAM [89], where a weight value is assigned to each point. Each weight is derived by the IMPLICIT Least Square (IMLS) method based on the local surface normal of the target point. However, it is often computationally costly to apply raw point cloud matching, e.g., it takes 1.2 s to estimate one frame in IMLS-SLAM that is far from real-time performance requirement.

The feature-based method mainly leverages point-to-surface/point-to-edge matching and are popularly used. Introduced by LOAM [49], this has become the standard for robot navigation. The edges and surfaces are extracted in advance based on local smoothness analysis, and the points from the current point cloud are matched to the edges and surfaces from the map. The distance cost is formulated

as the Euclidean distance between the points and edges/surfaces. There are also many works that extend LOAM to achieve better performance, e.g., LeGO-LOAM separates ground plane before feature extraction [50]. For UGVs, the z-axis is not important and most of the time we can assume robots moving in 2D space. Hence, ground points are not contributing to the localization of the x-y plane. This approach is also widely implemented on ground vehicles.

Some works attempt to improve the performance by introducing extra modules. In some large-scale scenarios, the loop closure serves in the back-end to identify repetitive places in order to reduce localization drifts. For example, in LiDAR-only Odometry and Localization (LOL) the author extends LiDAR odometry into a full SLAM with loop closure [90]. By closing the loop, odometry drifting error can be corrected. Fusion with the Inertial Measurement Unit (IMU) is an alternative solution. Shan *et al* propose a fusion method with IMU and GPS named LIO-SLAM [91]. Different sensor inputs are synchronized and tightly coupled. The fused SLAM system is proven to be more accurate than LOAM in outdoor environments. Fusion-based approaches have shown improvement on the localization accuracy. However, multi-sensor fusion requires synchronization and complicated calibration.

In the past few years, there are also some deep learning related works that attempt to use Convolution Neural Network (CNN) for point cloud processing. Rather than hand-crafted features, features are extracted by CNN training. For example, in CAE-LO [92], the author proposes a deep-learning-based feature point selection rule via unsupervised Convolutional Auto-Encoder (CAE) [93], while the LiDAR odometry estimation is still based on feature points matching. The experiment result shows that the learned features can improve the matching success rate. Although deep learning approaches have shown good performance in public datasets, it is not robust in different environments such as indoor/outdoor and city/rural scenarios. For example, [94] has a good performance on the semantic KITTI dataset. However, the same algorithm fails when moving to indoor environments or in some complex environments. Another challenge is the computational cost, i.e., real-time performance. For example, CAE-LO achieves the top 20 ranking on the KITTI dataset, but it takes 2 s on a high performance computer for each frame. In comparison, A-LOAM takes 0.1 s on a low-end computer.

3.3 System Overview

In this section, the proposed method is described in detail. We firstly present the sensor model and feature extraction. The extracted features are then calibrated with distortion compensation. Then, we present general feature extraction and global feature estimation. Lastly we explain the calculation of laser odometry as well as mapping.

3.3.1 Sensor Model and Feature Extraction

A mechanical 3D LiDAR perceives the surrounding environment by rotating a vertically aligned laser beam array of size M . Typical value of M is 16, 32, 64 or 128 based on different sensor configurations. It scans vertical planes with M readings in parallel. The laser array is rotating in a horizontal plane with N readings at a constant speed during each scan interval, while the laser measurements are taken in clockwise or counterclockwise order sequentially [95]. For a mechanical LiDAR sensor model, we denote the k^{th} LiDAR scan as P_k , and each point as $\mathbf{p}_k^{(m,n)}$, where $m \in [1, M]$ and $n \in [1, N]$. The points along vertical directions are detected in parallel while points along horizontal directions are detected continuously due to the rotation mechanism. The point id corresponds to the horizontal/vertical angle divided by horizontal/vertical angular resolution respectively. Raw point cloud matching methods such as ICP are sensitive to noise and dynamic objects such as humans for autonomous driving. Moreover, a LiDAR scan contains tens of thousands of points which makes these methods computationally inefficient. In comparison, feature point matching is more robust and computationally efficient in practice. To improve the matching accuracy and matching efficiency, we leverage the surface features and edge features, while discarding those noisy or less significant points. As mentioned above, the point cloud returned by a 3D mechanical LiDAR is sparse in vertical direction and dense in horizontal direction. Therefore, horizontal scans are more distinctive and it is less likely to have false feature detection in horizontal planes. Hence, for each point cloud, we focus on horizontal plane and evaluate the smoothness $\sigma_k^{(m,n)}$ of point $\mathbf{p}_k^{(m,n)}$ by

$$\sigma_k^{(m,n)} = \frac{1}{|\mathcal{S}_k^{(m,n)}|} \sum_{\mathbf{p}_k^{(m,j)} \in \mathcal{S}_k^{(m,n)}} (||\mathbf{p}_k^{(m,j)}|| - ||\mathbf{p}_k^{(m,n)}||), \quad (3.1)$$

where $\mathcal{S}_k^{(m,n)}$ is the adjacent points of $\mathbf{p}_k^{(m,n)}$ in horizontal direction. $\mathcal{S}_k^{(m,n)}$ can be easily collected based on point id n , which can reduce the computational cost compared to local searching. In the experiment we pick 5 points along the clockwise and counterclockwise, respectively. For a flat surface such as walls, the smoothness value is small while for a corner or edge point, the smoothness value is large. For each scan plane $\mathcal{P}_k$, edge feature points are selected with high σ , and surface feature points are selected with low σ . Therefore, we can formulate the edge feature set as $\mathcal{E}_k$ and the surface feature set as $\mathcal{S}_k$.

3.3.2 Motion Estimation and Distortion Compensation

The odometry is calculated by iterative pose estimation, where we iteratively calculate the pose until convergence. A good initial pose estimation can speed up pose convergence and reduce computational cost. In many works such as LOAM, the odometry is estimated by scan-to-scan matching followed by scan-to-map refinement. The scan-to-scan matching estimates the transformation between two adjacent scans, and scan-to-map refinement uses such initial estimation to align the current scan with the map and refine the transformation estimation. The initial pose alignment can fasten the pose convergence since the point alignment is performed in an iterative manner. However, this can be computationally expensive since initial pose is computed in an iterative manner. In this chapter, we use motion estimation to derive an initial pose which is more computationally efficient. Since time elapsed between two consecutive LiDAR scans is often very short, we can assume constant angular velocity and linear velocity during a short period. Although the estimation is not as accurate as scan-to-scan matching, the error can be corrected during scan-to-map matching. In our experiment, we find that it does not reduce much localization accuracy while the overall computational cost is reduced by 10-20%.

Denote the 6-DoF robot's pose at the k^{th} scan as $\xi_k = [\rho, \phi] \in \mathfrak{se}(3)$, where the ρ and ϕ are 3D vectors. The corresponding transformation matrix is denoted as $\mathbf{T}_k \in SE(3)$. The related transform between two neighbouring frames $k-1$ and $k-2$ can be represented as

$$\xi_{k-2}^{k-1} = \log(\mathbf{T}_{k-2}^{-1} \mathbf{T}_{k-1})^\vee. \quad (3.2)$$

By assuming linear velocity and constant angular velocity, i.e., $\hat{\xi}_{k-1}^k = \xi_{k-2}^{k-1}$, we can estimate the next robot pose prediction as

$$\hat{\xi}_k = \xi_{k-1} \oplus \hat{\xi}_{k-1}^k, \quad (3.3)$$

where the operator $\xi_1 \oplus \xi_2$ are the addition on manifold

$$\xi_1 \oplus \xi_2 = \log(\exp(\xi_1^\wedge) \exp(\xi_2^\wedge))^\vee. \quad (3.4)$$

A mechanical 3D LiDAR sweeps 360 degrees continuously. As shown in Figure 3.1, the points within a scan period between the $(k-1)^{th}$ and k^{th} are re-projected to the local coordinate at $t = k$ without motion compensation. Such distortion reduces both localization accuracy and mapping precision. It becomes more severe when the robot is moving at high speed, e.g., autonomous driving. Hence it is necessary to compensate for motion distortion before matching a point cloud into a global map.

FIGURE 3.1: Relation between sweeping angle and time.

The distortion can be corrected by the predicted motion from the last step. We can estimate the robot pose by assuming constant movement on $\mathfrak{sc}(3)$. Because the LiDAR is sweeping at a constant speed, the n^{th} point within a sweep is scanned at time $\delta t = \frac{n}{2\pi \cdot f}$, where f is the scanning frequency. Therefore, for the point detected at time δt of the k^{th} sweep, the pose can be estimated by

$$\hat{\mathbf{T}}_k(\delta t) = \exp(\xi_{k-1} \oplus \frac{N-n}{N} \hat{\xi}_{k-1}^k)^\wedge. \quad (3.5)$$

The distortion at k^{th} scan can be corrected as

$$\tilde{\mathcal{P}}_k = \{\hat{\mathbf{T}}_k(\delta t) \mathbf{p}_k^{(m,n)} \mid \mathbf{p}_k^{(m,n)} \in \mathcal{P}_k\}, \quad (3.6)$$

where $\tilde{\mathcal{P}}_k$ is the undistorted point cloud.

3.3.3 Feature Correspondence and Pose Estimation

The global feature map is separated into an edge map and a planar map corresponding to the edge feature and planar feature extracted from scan input. To reduce the searching computational cost, both edge and planar maps are stored in 3D KD-trees. Similar to [96], global line and plane are estimated by collecting nearby points from the edge and planar feature maps. For each scan, edge points are matched with edge map and planar points are matched with planar map respectively. To associate current points with a global edge or plane, we can collect a set of nearby points and estimate an optimal global edge/plane respectively. For each edge point $\mathbf{p}_{\mathcal{E}} \in \tilde{\mathcal{P}}_k$ and its transform in the global coordinate $\mathbf{p}'_{\mathcal{E}} = \mathbf{T}_k \cdot \mathbf{p}_{\mathcal{E}}$, we can find q nearest points from global edge map $\{\mathbf{p}_{\mathcal{E}}^1, \mathbf{p}_{\mathcal{E}}^2, \dots, \mathbf{p}_{\mathcal{E}}^q\}$. Let the corresponding global edge be a linear line in 3D space, with unit direction $\mathbf{n}_{\mathcal{E}}^g$ and a point in the edge $\mathbf{p}_{\mathcal{E}}^g$. We can derive the optimal global edge by minimizing the point to edge distance

$$\arg \min_{\mathbf{n}_{\mathcal{E}}^g, \mathbf{p}_{\mathcal{E}}^g} \sum_{i \in [1, q]} \|(\mathbf{p}_{\mathcal{E}}^i - \mathbf{p}_{\mathcal{E}}^g) \times \mathbf{n}_{\mathcal{E}}^g\|. \quad (3.7)$$

Similarly, for each planar point $\mathbf{p}_{\mathcal{S}} \in \tilde{\mathcal{P}}_k$ its transform in the global coordinate $\mathbf{p}'_{\mathcal{S}} = \mathbf{T}_k \cdot \mathbf{p}_{\mathcal{S}}$, we can find q nearest points from global planar map $\{\mathbf{p}_{\mathcal{S}}^1, \mathbf{p}_{\mathcal{S}}^2, \dots, \mathbf{p}_{\mathcal{S}}^q\}$. Denote the corresponding global plane normal $\mathbf{n}_{\mathcal{S}}^g$ and a point in the surface $\mathbf{p}_{\mathcal{S}}^g$, we can derive the optimal global plane by minimizing the point to plane distance

$$\arg \min_{\mathbf{n}_{\mathcal{S}}^g, \mathbf{p}_{\mathcal{S}}^g} \sum_{i \in [1, q]} \|(\mathbf{p}_{\mathcal{S}}^i - \mathbf{p}_{\mathcal{S}}^g) \cdot \mathbf{n}_{\mathcal{S}}^g\|. \quad (3.8)$$

Hence we can associate the features from the current scan to global edges and planes. To improve the accuracy, edge features with less than 2 nearest features or planar features with less than 3 nearest features are discarded due to the lack of enough correspondent points.

The pose estimation aligns the current undistorted edge features $\tilde{\mathcal{E}}_k$ and planar features $\tilde{\mathcal{S}}_k$ with the global feature map. From the last step, we can find the linked global edges or planes for each feature point from $\tilde{\mathcal{P}}_k$. Such correspondence

can be used to estimate the optimal pose between current frame and global map by minimizing the distance between feature points and global edges/planes. The distance between the edges features and global edge is

$$f_{\mathcal{E}}(\mathbf{p}_{\mathcal{E}}) = (\mathbf{T}_k \mathbf{p}_{\mathcal{E}} - \mathbf{p}_{\mathcal{E}}^g) \times \mathbf{n}_{\mathcal{E}}^g. \quad (3.9)$$

The distance between the planar features and global plane is

$$f_{\mathcal{S}}(\mathbf{p}_{\mathcal{S}}) = (\mathbf{T}_k \mathbf{p}_{\mathcal{S}} - \mathbf{p}_{\mathcal{S}}^g) \cdot \mathbf{n}_{\mathcal{S}}^g. \quad (3.10)$$

Traditional feature matching only optimizes the geometry distance mentioned above, while the local geometry distribution of each feature point is not considered. However, it is observed that edge features with higher local smoothness and planar features with lower smoothness are often consistently extracted over consecutive scans. These features are more important for matching. Therefore, a weight function is introduced to further balance the matching process. To reduce the computational cost, the local smoothness defined previously is re-used to determine the weight function. For each edge point $\mathbf{p}_{\mathcal{E}}$ with local smoothness $\sigma_{\mathcal{E}}$ and each plane point $\mathbf{p}_{\mathcal{S}}$ with local smoothness $\sigma_{\mathcal{S}}$, the weight is defined by

$$\begin{aligned} W(\mathbf{p}_{\mathcal{E}}) &= \frac{\exp(-\sigma_{\mathcal{E}})}{\sum_{\mathbf{p}^{(i,j)} \in \mathcal{E}_k} \exp(-\sigma_k^{(i,j)})}, \\ W(\mathbf{p}_{\mathcal{S}}) &= \frac{\exp(\sigma_{\mathcal{S}})}{\sum_{\mathbf{p}^{(i,j)} \in \mathcal{S}_k} \exp(\sigma_k^{(i,j)})}, \end{aligned} \quad (3.11)$$

where $\mathcal{E}_k$ and $\mathcal{S}_k$ are the edge features and plane features at the k^{th} scan. For an edge point with higher local smoothness, the weight is higher. For a planar point with lower local smoothness, the local surface is flatter so that the weight is higher. The new pose is estimated by minimizing the weighted sum of the point-to-edge and point-to-planar distance

$$\arg \min_{\mathbf{T}_k} \sum W(\mathbf{p}_{\mathcal{E}}) f_{\mathcal{E}}(\mathbf{T}_k \mathbf{p}_{\mathcal{E}}) + \sum W(\mathbf{p}_{\mathcal{S}}) f_{\mathcal{S}}(\mathbf{T}_k \mathbf{p}_{\mathcal{S}}). \quad (3.12)$$

The optimal pose estimation can be derived by solving the non-linear least squares problem through Gauss-Newton method mentioned in Section 2.2.1.1. However, it is not straightforward to calculate the derivative in the Lie group. Here, we

apply the Gauss-Newton method with a customized perturbation scheme. There are several advantages of using the perturbation scheme compared to a derivative model: (i) the rotation or pose is stored in a singularity-free format; (ii) at each iteration unconstrained optimization is performed; (iii) the manipulations occur at the matrix level so that there is no need to worry about taking the derivatives of massive scalar trigonometric functions, which can easily lead to errors [97]. Consider a non-linear quadratic function

$$\mathcal{C}(\mathbf{T}) = \sum_m (f_m(\mathbf{T}\mathbf{p}_m))^2, \quad (3.13)$$

where f_m is a non-linear function. We begin with an initial guess $\mathbf{T}_{op} \in SE(3)$ and multiply a small left perturbation $\delta\xi \in \mathfrak{se}(3)$

$$\mathbf{T} = \exp(\delta\xi^\wedge) \cdot \mathbf{T}_{op}. \quad (3.14)$$

The updated function f_m can be approximated as a linear function

$$\begin{aligned} f_m(\mathbf{T}\mathbf{p}_m) &= f_m(\exp(\delta\xi^\wedge)\mathbf{T}_{op}\mathbf{p}_m) \\ &\approx f_m((\mathbf{I} + \xi^\wedge)\mathbf{T}_{op}\mathbf{p}_m) \\ &\approx f_m(\mathbf{T}_{op}\mathbf{p}_m) + \frac{\partial f_m}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{T}_{op}\mathbf{p}_m} \cdot \mathbf{J}_p \cdot \delta\xi, \end{aligned} \quad (3.15)$$

where

$$\mathbf{J}_p = \frac{\partial \mathbf{T}_{op}\mathbf{p}_m}{\partial \delta\xi} = \begin{bmatrix} \mathbf{I}_{3 \times 3} & -[\mathbf{T}_{op}\mathbf{p}_m]_\times \\ \mathbf{0}_{1 \times 3} & \mathbf{0}_{1 \times 3} \end{bmatrix}, \quad (3.16)$$

with $[\mathbf{T}_{op}\mathbf{p}_m]_\times$ transforming 4-D homogeneous point expression $\{x, y, z, 1\}$ into 3D point expression $\{x, y, z\}$ and calculating its skew matrix. Let $\mathbf{J}_m = \frac{\partial f_m}{\partial \mathbf{x}}|_{\mathbf{x}=\mathbf{T}_{op}\mathbf{p}_m} \mathbf{J}_p$ and $\beta_m = f_m(\mathbf{T}_{op}\mathbf{p}_m)$, the cost function $\mathcal{C}$ is approximated by

$$\mathcal{C}(\mathbf{T}) = \sum_m (\beta_m + \mathbf{J}_m \delta\xi)^2, \quad (3.17)$$

which is quadratic in $\delta\xi$. Taking the derivative we have

$$\sum_m (\mathbf{J}_m^T \mathbf{J}_m) \delta\xi = - \sum_m \mathbf{J}_m \beta_m. \quad (3.18)$$

We can then apply this optimal perturbation to our initial guess

$$\mathbf{T}_{op} \leftarrow \exp(\delta\xi) * \mathbf{T}_{op}. \quad (3.19)$$

Similarly, we apply the left perturbation scheme to our optimization function. The Jacobian matrix of edge residual can be calculated by

$$\begin{aligned} \mathbf{J}_{\mathcal{E}} &= W(\mathbf{p}_{\mathcal{E}}) \frac{\partial f_{\mathcal{E}}}{\partial \mathbf{x}} \Big|_{\mathbf{x}=\mathbf{T}_{op}\mathbf{p}_{\mathcal{E}}} \frac{\partial \mathbf{T}_{op}\mathbf{p}_{\mathcal{E}}}{\partial \delta\xi} \\ &= W(\mathbf{p}_{\mathcal{E}}) \mathbf{J}_p \times \mathbf{n}_{\mathcal{E}}^g. \end{aligned} \quad (3.20)$$

For the point-to-plane residual, we can derive the Jacobian matrix by

$$\begin{aligned} \mathbf{J}_{\mathcal{S}} &= W(\mathbf{p}_{\mathcal{S}}) \frac{\partial f_{\mathcal{S}}}{\partial \mathbf{x}} \Big|_{\mathbf{x}=\mathbf{T}_{op}\mathbf{p}_{\mathcal{S}}} \frac{\partial \mathbf{T}_{op}\mathbf{p}_{\mathcal{S}}}{\partial \delta\xi} \\ &= W(\mathbf{p}_{\mathcal{S}}) \mathbf{J}_p \cdot \mathbf{n}_{\mathcal{E}}^g. \end{aligned} \quad (3.21)$$

And then we can find the small perturbation $\delta\xi$ and update the new pose by

$$(\sum \mathbf{J}_{\mathcal{S}}^T \mathbf{J}_{\mathcal{S}} + \sum \mathbf{J}_{\mathcal{E}}^T \mathbf{J}_{\mathcal{E}}) \delta\xi = -(\sum \beta_{\mathcal{E}} + \sum \beta_{\mathcal{S}}), \quad (3.22)$$

where

$$\begin{aligned} \beta_{\mathcal{E}} &= W(\mathbf{p}_{\mathcal{E}}) f_{\mathcal{E}}(\mathbf{T}_{op}\mathbf{p}_{\mathcal{E}}), \\ \beta_{\mathcal{S}} &= W(\mathbf{p}_{\mathcal{S}}) f_{\mathcal{S}}(\mathbf{T}_{op}\mathbf{p}_{\mathcal{S}}). \end{aligned} \quad (3.23)$$

Then the optimal pose can be updated by Equation (3.19). The current pose estimation can be solved by iterative pose optimization until it converges. The algorithm is shown in Algorithm 3.1.

3.3.4 Map Building

A dense map is reconstructed by integrating raw point clouds from keyframes. Each keyframe is selected when the displacement is greater than a predefined threshold, or when the time exceeds a certain period. The key-frame-based map update can reduce the computational cost compared to a frame-by-frame update. To increase the updating efficiency, an octotree is used to construct the global map. It only takes computational complexity of $\mathcal{O}(\log n)$ to search for a specific node from an octotree of depth n , which is computationally efficient [98]. For each cell in an

Algorithm 3.1: Fast LOAM

Input : Current Scan $\mathcal{P}_k$, global map $\mathcal{S}, \mathcal{E}$
Output: Pose estimation in global coordinate $\mathbf{T}_k$
begin
 if *Not Initialized* **then**
 $\mathbf{T}_0 \leftarrow \mathbf{0}$;
 end
 Calculate the local smoothness of current point cloud input $\mathcal{P}_k$ and extract edge features and planar features ;
 Compute initial alignment $\mathbf{T}_k$;
 for each point $\mathbf{p}_k^{(m,n)}$ in $\mathcal{P}_k$ **do**
 Find transform matrix $\hat{\mathbf{T}}_k(\delta\xi)$;
 Derive undistorted point cloud $\tilde{\mathcal{P}}_k$;
 end
 for a number of iterations **do**
 for each edge feature $\mathbf{p}_{\mathcal{E}}^{(m,n)}$ in $\tilde{\mathcal{P}}_{k+1}$ **do**
 Search for nearby points from global map $\mathcal{E}$;
 Estimate corresponding global edge $\mathbf{n}_{\mathcal{E}}^g, \mathbf{p}_{\mathcal{E}}^g$;
 end
 for each planar feature $\mathbf{p}_{\mathcal{S}}^{(m,n)}$ in $\tilde{\mathcal{P}}_{k+1}$ **do**
 Search for nearby points from global map $\mathcal{S}$;
 Estimate corresponding global plane $\mathbf{n}_{\mathcal{S}}^g, \mathbf{p}_{\mathcal{S}}^g$;
 end
 Compute loss function $f_{\mathcal{E}}(\mathbf{T}_k \mathbf{p}_{\mathcal{E}})$ and $f_{\mathcal{S}}(\mathbf{T}_k \mathbf{p}_{\mathcal{S}})$;
 Compute Jacobian matrix $J_{\mathcal{E}}$ and $J_{\mathcal{S}}$;
 if *nonlinear optimization converges* **then**
 Update $\mathbf{T}_k$
 end
 end
 Update current scan $\mathcal{P}_k$ to map $\mathcal{S}, \mathcal{E}$;
 Return odometry $\mathbf{T}_k$;
end

octotree, we use $P(n|z_{1:t})$ to present the probability of the existence of an object with [99]

$$P(n|z_{1:t}) = \left[1 + \frac{1 - P(n|z_t)}{P(n|z_t)} \cdot \frac{1 - P(n|z_{1:t-1})}{P(n|z_{1:t-1})} \cdot \frac{P(n)}{1 - P(n)} \right]^{-1}, \quad (3.24)$$

where z_t is the current measurement, $z_{1:t-1}$ is the historical measurements from keyframes, $P(n)$ is the prior probability, which is set to 0.5 for unknown areas. Let

$$L(t) = \log \frac{P(n|z_{1:t})}{1 - P(n|z_{1:t})}, \quad (3.25)$$

the new probability can be updated by

$$L(t) = L(t-1) + \log \frac{P(n|z_t)}{1 - P(n|z_t)}. \quad (3.26)$$

Therefore, the probability of each cell can be updated based on the new measurement.

3.4 Experiment Results

To validate the algorithm, we evaluate FLOAM on both a large-scale outdoor environment and a medium scale indoor environment. For the large-scale experiment, we evaluate our approach on the KITTI dataset [18] which is one of the most popular datasets for SLAM evaluation. Then the algorithm is integrated into warehouse logistics. It is firstly validated in a simulated warehouse environment and then it is tested on an AGV platform. The algorithm is implemented in C++ and the source code is available online¹.

3.4.1 Evaluation on Public Dataset

We firstly test our method on the KITTI dataset [100] that is popularly used for outdoor localization evaluation. The dataset is collected from a driving car equipped with Velodyne HDL-64 LiDAR, cameras, and GPS. Most state-of-the-art SLAM methods have been evaluated on this dataset, e.g., ORB-SLAM [31], VINS-Fusion [101], LIMO [102], and LSD-SLAM [103]. To validate the robustness, we evaluate the proposed algorithm on all sequences from the KITTI dataset, which includes different scenarios such as highway, city center, country road, residential area, etc. The ground truth is provided by an OXTS Inertial Navigation System (GPS/IMU), which has a localization accuracy of *cm*. The laser scanner spins at

¹<https://github.com/wh200720041/floam>

10 frames per second, capturing approximately 100k points per cycle. The vertical resolution of the laser scanner is 64. The details of setup are shown in Figure 3.2. Since the ground truth of sequence 11-21 is not open to the public, we mainly

FIGURE 3.2: Setup of the autonomous driving car on the KITTI dataset [100].

demonstrate the performance on sequence 00-11. Figure 3.3 shows the evaluation result of FLOAM on different sequences. In total there are 23,201 frames and the traveled length is 22 *km* over 11 sequences. The details of each sequence are shown in Table 3.1. The trajectory estimated by FLOAM is plotted in green color and the ground truth is plotted in red color. It can be seen that the localization result of FLOAM is very close to the ground truth.

In addition, we also demonstrate the mapping result of the proposed method. The point cloud is reconstructed based on the trajectory from FLOAM and it is colored based on the image view which is also provided by the KITTI dataset. We use KITTI sequence 05 to demonstrate our approach and the result is shown in Figure 3.4. Our mapping result can closely reveal the real scenarios.

Sequence	Environment	Total Frames	Travel Length (m)
00	Urban	4540	3724.2
01	Highway	1100	2453.2
02	Urban+Country	4660	5067.2
03	Country	800	560.9
04	Country	270	393.6
05	Urban	2760	2205.6
06	Urban	1100	1232.9
07	Urban	1100	694.7
08	Urban+Country	5170	3222.8
09	Urban+Country	1590	1705.1
10	Urban+Country	1200	919.5

TABLE 3.1: Information of KITTI sequence 00-10.

FIGURE 3.3: Result of the proposed FLOAM on the KITTI dataset. The estimated trajectory and ground truth are plotted in green and red respectively. (a)-(k) KITTI sequence 00-10.

FIGURE 3.4: Example of the proposed method on the KITTI dataset. (a) the mapping result on sequence 05; (b) the reconstructed 3D road scenery by integrating camera view; (c) result comparison of FLOAM and ground truth.

To quantify the performance of the proposed method, we calculate the Average Translational Error (ATE) and Average Rotational Error (ARE) which are defined by the KITTI dataset [100]

$$\begin{aligned}
 E_{rot}(\mathcal{F}) &= \frac{1}{|\mathcal{F}|} \sum_{i,j \in \mathcal{F}} \angle[\hat{T}_j \hat{T}_i^{-1} T_i T_j^{-1}], \\
 E_{trans}(\mathcal{F}) &= \frac{1}{|\mathcal{F}|} \sum_{i,j \in \mathcal{F}} \|\hat{T}_j \hat{T}_i^{-1} T_i T_j^{-1}\|_2,
 \end{aligned} \tag{3.27}$$

where $\mathcal{F}$ is a set of frames (i, j) , T and $\hat{T}$ are the estimated and true LiDAR poses respectively, $\angle[\cdot]$ is the rotation angle. The proposed method is tested on an Intel i7 3.2GHz processor based computer and the time is calculated from the start of feature extraction till the output of estimated odometry. The evaluation results are shown in Table 3.2. Our method achieves an average translational error of 0.80% and an average rotational error of 0.0048 deg / m over 11 sequences, which is accurate enough for most of robotic applications.

We also compare our method with state-of-the-art methods, including both LiDAR

Sequence	Runtime (ms/frame)	ATE (%)	ARE ($\times 10^{-2}$ deg/m)
00	76.2	0.5304	0.44
01	105.4	1.3183	0.29
02	70.9	0.6340	0.49
03	110.6	0.6223	0.61
04	93.4	0.3819	0.47
05	75.2	0.3457	0.35
06	95.7	0.2616	0.38
07	70.4	0.3430	0.62
08	87.8	0.8355	0.52
09	97.2	0.5785	0.40
10	81.4	0.8005	0.60

TABLE 3.2: Performance evaluation of FLOAM on the KITTI dataset.

SLAM and visual SLAM. The proposed method is compared with state-of-the-art LiDAR methods such as LOAM [49], A-LOAM, HDL-Graph-SLAM [104], IMLS-SLAM [89] and LeGO-LOAM [50]. Note that IMLS-SLAM is not open sourced and the results are collected from the original paper [89]. And the results of LOAM, A-LOAM, HDL-Graph-SLAM, LeGO-LOAM are collected by running the open-sourced algorithm on the same local computer. All algorithms are coded in C++ and tested on ROS. In addition, the VSLAM such as VINS-MONO [105] is also compared. For consistency, the IMU information is not used and the loop closure detection is removed. The testing environment is based on ROS Melodic [106] and Ubuntu 18.04. The results are shown in Figure 3.5. IMLS-SLAM achieves the highest accuracy. However, all points are used in iterative pose estimation in IMLS-SLAM so that the processing speed is slow, which is less than 2 Hz. In terms of computational cost, LeGO-LOAM is the fastest LiDAR SLAM since it only applies optimization on non-ground points. Among all methods compared, our method achieves the second highest accuracy at an average processing rate of more than 10 Hz, which is a good trade-off between computational cost and localization accuracy. Our method is able to provide both fast and accurate SLAM solution for real time robotic applications and is competitive among state-of-the-art approaches.

FIGURE 3.5: Comparison of the different localization approaches on the KITTI dataset.

Methods	Computing Time (ms/frame)	Accuracy (cm)
No Compensation	66.33	2.132
LOAM	84.52	2.052
FLOAM	69.07	2.037

TABLE 3.3: Ablation study of localization accuracy and computational cost.

3.4.2 Ablation Study

To further evaluate the performance of the proposed distortion compensation approach, we compare the results of different approaches in the warehouse environment. We firstly remove the distortion compensation in FLOAM and record both computing time and localization accuracy. Then we add the iterative distortion compensation method from LOAM into our method and record the result. Lastly we use the proposed motion compensation and record the result. The results are shown in Table 3.3. It can be seen that the proposed approach is much faster than the LOAM with motion compensation yet with a slightly better localization accuracy.

3.4.3 Experiment on Warehouse Logistics

In this experiment, we target to build an autonomous warehouse robot to replace human-labour-dominated manufacturing. The AGV is designed to carry out daily

tasks such as transportation. This requires the robot platform to actively localize itself in a complex environment.

3.4.3.1 Simulation

We firstly test our algorithm in a simulated environment. The simulation environment is built up on Gazebo and Linux Ubuntu 18.04. As shown in Figure 3.6 (a), we use a virtual Pioneer robot and a virtual Velodyne VLP-16 as the ground vehicle platform. The simulated environment reconstructs a complex warehouse environment including various objects such as moving human workers, shelves, machines, etc. The robot is controlled by a joystick to move around at the maximum speed of 2 m/s. We record the trajectory from FLOAM and compare it with the ground truth. The result is shown in Figure 3.6 (c), where the FLOAM trajectory and ground truth are plotted in green and red color respectively. Our method is able to track the AGV moving at high speed under the dynamic environment, where there are human operators walking in the warehouse environment.

3.4.3.2 Experiment

To further demonstrate the performance of our method, we implement FLOAM on an actual AGV used for smart manufacturing. As shown in Figure 3.7, the warehouse environment consists of three main areas: auto charging station, material handling area and manufacturing station. A fully autonomous factory requires a robot to deliver material to manufacturing machines for assembly of products. All these operations require precise localization to ensure reliability and safety. The robot platform for testing is shown in Figure 3.8 (a) and is equipped with an Intel NUC mini-computer and a Velodyne VLP-16 sensor. The localization and mapping result can be found in Figure 3.8. In this scenario, the robot automatically explores the warehouse and builds the map simultaneously using the proposed method.

3.4.3.3 Performance Evaluation

The proposed method is also tested in an indoor room equipped with a VICON system to evaluate its localization accuracy. The proposed method is integrated

FIGURE 3.6: Simulated warehouse environment with both static and dynamic objects. (a) simulated environment in Gazebo; (b) simulated Pioneer robot and Velodyne LiDAR for evaluation; (c) trajectory comparison of FLOAM and ground truth.

FIGURE 3.7: Advanced factory environment built for AGV manipulation, including operating machine, auto charging station and storage shelves.

FIGURE 3.8: Evaluation of the proposed method in a warehouse environment. (a) AGV used for the experiment; (b) FLOAM result of warehouse localization and mapping.

into the Pioneer robot with a Velodyne VLP-16. As shown in Figure 3.9, the robot is remotely controlled to move in a VICON-covered area. The results are shown in Figure 3.10, where the FLOAM trajectory and the ground truth trajectory are plotted in green and red color respectively. The ground truth trajectory is provided by the VICON system. It can be seen that our method can accurately track the robot pose. It achieves an average localization accuracy of 2 *cm* compared to the ground truth.

3.5 Conclusion

In this chapter, we have presented a computationally efficient LiDAR Odometry framework which targets to provide a solution to robotic applications with limited computational resources. Note that lines and planes are often distinctive, we adopt an efficient feature extraction via local smoothness analysis to search for edge and planar features respectively. The extracted features are associated with global lines and planes and the robot pose is achieved by minimizing point-to-edge

FIGURE 3.9: Comparison with ground truth. (a) Pioneer robot used for performance evaluation; (b) testing area.

FIGURE 3.10: Comparison of the proposed method and ground truth. Our method can precisely track robot's pose with an average localization error of 2 *cm*

and point-to-plane distances. Compared to traditional point cloud matching methods such as ICP, the computational cost is significantly reduced. Moreover, our solution achieves motion prediction via non-iterative calculation, which further reduces computational cost. To demonstrate the robustness of the proposed method in practical applications, thorough experiments have been done to evaluate the performance, including simulation, indoor AGV test and outdoor autonomous driving test. Our method achieves an average localization accuracy of 2 *cm* with more than 10 *Hz* processing rate in indoor tests and remains one of the most accurate and fastest open-sourced methods on the KITTI dataset. The results provide a basic real-time front-end SLAM solution for autonomous vehicles and will be further discussed and improved in the next chapters.

Chapter 4

Fast Loop Closure Detection via Binarized Saliency Map

4.1 Introduction

In the previous chapter, we introduced a LiDAR-based Odometry to achieve real-time localization on autonomous vehicles. As stated in Section 2.3.2, LiDAR odometry generates localization drifts and leads to localization failure in the long run. A good SLAM system should have the capability to re-localize itself under localization failure. In this chapter, we will explore the re-localization techniques on SLAM. Existing solutions to the LCD problem consist of vision-based LCD and LiDAR-based LCD. Compared to LiDAR-based methods, vision-based LCD or Visual Place Recognition (VPR) adopts image analysis to identify the repetitive scenes, which is more similar to human perception. Hence we firstly explore the potential of using vision-based loop closure detection as a method of re-localization.

Computational cost is one of the key bottlenecks in VPR [107, 108]. For example, in a SLAM system, the query time of incoming data grows as more places are visited and registered into the database. It can subsequently slow down the system or cause memory overflow in the long run. Most existing works down-scale an image (or a place) into lower dimensional features to improve computational efficiency. A popular strategy is to re-formulate the visual place recognition problem as the retrieval of the local hand-crafted features such as ORB [59] and BRISK [66].

However, to guarantee high accuracy, feature-based approaches such as iBoW-LCD [61] and FABMAP [55] require the extraction and matching of thousands of descriptors for each image, which is computationally expensive. Existing feature-based methods can achieve satisfactory recall rate results but are difficult to run in real-time. Moreover, discretization of local descriptors requires complicated offline training that is troublesome in practice.

We argue that extracting and comparing feature descriptors take too many computational resources and become a burden to the processing system over a long run. Note that the distribution of the objects or salient patterns is also a piece of important information in addition to feature points. In a scene, the geometrical distribution of objects as well as the shape of each object are often unique and hence they can be used for loop closure detection. However, these information are not utilized in feature-based methods. Hence we introduce these information to tackle the loop closure detection problem and represent the object distribution information as binarized saliency map. Thereafter, we can efficiently verify the loop closure places by checking the similarity of the binarized saliency maps of two images via fast logical operations. At the same time, we keep existing feature point methods on top of binarized saliency map indexing to achieve a high recall rate. The new framework consists of three parts: binarized saliency map construction, fast image retrieval and precise loop closure detection. It firstly introduces a binary map into loop closure detection to reduce the computational cost for indexing. Then precise image matching is applied to guarantee precision. Compared to the existing methods, no offline training is required for our method. The main contributions of this chapter are as follows:

- We propose a binarized saliency map based fast loop closure detection, which combines the advantage of both fast logical operation and traditional feature-based method. Specifically, we propose to represent a place using saliency distribution and further express the saliency distribution as a binarized saliency map.
- We propose a computationally efficient image retrieval strategy on saliency distribution via fast logical operation. Moreover, we use the traditional feature-based matching to avoid false matching. The experiment results show that our method is much faster than existing feature-based methods without reducing recall rate and recall precision.

- Compared to existing methods, the proposed method does not require any offline training. It can be easily implemented in the SLAM system.

This chapter is organized as follows: Section 4.2 reviews the related works on loop closure detection. Section 4.3 describes the details of the proposed method. Section 4.4 shows experiment results and comparisons with existing works, followed by the conclusion in Section 4.5.

4.2 Related Works

Feature matching is the most popular strategy for visual place recognition. In FABMAP, the visual vocabulary of SURF feature [109] is trained by hierarchical k-means clustering to discretize its high dimensional representation [110]. Revisited places are then identified by matching feature distribution over an offline-trained visual vocabulary. Similar idea is employed in DBoW2 [58], in which the binary descriptor ORB [59] is used instead. Compared to FABMAP, it achieves faster speed due to the binary feature representation. However, discretization of local descriptors requires complicated offline training for visual vocabulary that is troublesome in practice. For example, in FABMAP, it takes a few hours to train a visual vocabulary on a desktop computer. Moreover, the training data have to be collected in a similar environment and cannot contain any loop in order to achieve a good result.

Some other research works aim to find an effective and efficient image descriptor for loop closure detection. Mur-Artal *et al.* [111] uses SURF feature descriptor for loop detection. It achieves a satisfactory result but the computational cost is high. Huang *et al.* [112] extracts VLAD vectors from each image. The VLAD is a first order statistics of the non-probability Fisher Vector [113] and can be obtained by training a codebook of k visual words using k-means method. The similarity between two images is estimated by measuring the Euclidean distance of VLAD vectors. In recent years, there are also some binary descriptors used in loop closure detection such as Binary Robust Invariant Scalable Keypoints (BRISK) and Binary Robust Independent Elementary Features (BRIEF), [66, 109, 114–116]. They take the advantage of fast logical operation and use probability theory

to represent features. The computing speed is much faster compared to the non-binary descriptors.

4.3 System Overview

The proposed binarized saliency map based loop closure detection framework consists of three parts: binarized saliency map construction, fast image retrieval and precise loop closure detection. The framework is shown in Figure 4.1. To utilize the object distribution information, the binarized saliency map construction extracts the objects or salient regions from an image and compresses the extracted parts into compact binarized saliency map. Then the fast image retrieval module performs binary image indexing at high speed and filters out most unmatched pairs. Eventually, precise loop closure detection conducts further check on the result to remove any false positive. The details of each step will be explained in this section.

4.3.1 Binarized Saliency Map Construction

The extracted binarized saliency map is a highly representative signature of the original image. In this chapter we only focus on the distribution of salient regions. However, the definition of salient region is subjective and can be varied according to one's psychological process [117] and environmental changes [118]. For simplicity, we follow the observation that human eyes are more sensitive to salient regions in an image with specific spectrum properties [119]. The salient regions can be either objects or visual distinct areas that stand out. Although this simplified model cannot guarantee that all the saliency objects found are subjected to our psychological definition, we find that they are qualified enough for place recognition in SLAM. The location of the salient region and the shape of the salient regions are different in different images so that they can be a criterion to search for paired images. To extract salient regions, we adopt the Log spectral residual method [119], which has the advantage of low computational cost and high extraction capability. Moreover, no prior knowledge is required. Generally, given an input image $\mathbf{I}$, we define the following notations:

FIGURE 4.1: Framework of the proposed method.

- $\mathcal{A}(\hat{\mathbf{I}})$: the real part of Fast Fourier Transform of image $\mathbf{I}$, $\mathcal{A}(\hat{\mathbf{I}}) = \Re(\mathcal{F}(\mathbf{I}))$.
- $\mathcal{P}(\hat{\mathbf{I}})$: the imaginary part of Fast Fourier Transform of image $\mathbf{I}$, $\mathcal{P}(\hat{\mathbf{I}}) = \Im(\mathcal{F}(\mathbf{I}))$.
- $\mathcal{L}(\hat{\mathbf{I}})$: the log spectral of $\mathcal{A}(\hat{\mathbf{I}})$, $\mathcal{L}(\hat{\mathbf{I}}) = \log(\mathcal{A}(\hat{\mathbf{I}}))$.

The Log spectral residual $\mathcal{R}(\hat{\mathbf{I}})$ is defined as

$$\mathcal{R}(\hat{\mathbf{I}}) = \mathcal{L}(\hat{\mathbf{I}}) - f_{ave} * \mathcal{L}(\hat{\mathbf{I}}), \quad (4.1)$$

where f_{ave} is a normalized average filter of an $n \times n$ matrix and the symbol $*$ is the image cross-correlation. Salient region map $\mathcal{M}$ can be derived by recovering

(A) Raw Image

(B) Result after saliency extraction

FIGURE 4.2: An example of Log spectral residual approach.

Equation (4.1) with Gaussian filter $\mathcal{G}_\sigma$

$$S(x) = \mathcal{G}_\sigma * \mathcal{F}^{-1}[\mathcal{R}(\hat{\mathbf{I}}) \cdot \exp(\mathbf{j} \cdot \mathcal{P}(\hat{\mathbf{I}}))]^2, \quad (4.2a)$$

$$\mathcal{M} = \begin{cases} 1 & \text{if } S(x) > E(S(x)) \cdot \gamma, \\ 0 & \text{otherwise,} \end{cases} \quad (4.2b)$$

where the threshold γ indicates the level of salient region extraction and $\mathbf{j}$ is the imaginary unit. A larger γ implies that less salient area will be ignored, and only highly salient regions or objects will be retained. A demonstration of the Log spectral residual approach is shown in Figure 4.2, where only the crafts are kept after filtering. Salient regions contain the most representative information of the image and it is unique for each image in most cases. The saliency region map is a binary image and is stored into the database for later processing.

4.3.2 Fast Image Retrieval

Fast image retrieval aims to match the current saliency map with the database. The key idea is to make use of fast logical operations to reduce the computational cost. It is observed that similar scenes share a similar salient region distribution. Moreover, when a place is revisited, the light condition or view angle can be changed slightly, but the distribution will remain the same. Hence, we can perform an element-wise similarity check to compare the salient region map $\mathcal{M}_1$ and $\mathcal{M}_2$ by

$$\xi = \frac{\mathcal{F}(\text{AND}(\mathcal{M}_1, \mathcal{M}_2))}{\max\{\mathcal{F}(\mathcal{M}_1), \mathcal{F}(\mathcal{M}_2)\}}, \quad (4.3)$$

where ξ is the similarity factor of two binary images, $\text{AND}(\mathbf{a}, \mathbf{b})$ is the element-wise AND operator of two matrices $\mathbf{a}$ and $\mathbf{b}$, $\mathcal{F}(\mathbf{x})$ is the number of non-zero values in the binary image $\mathbf{x}$. The fast image retrieval can be performed by setting a threshold to ξ . A larger ξ will have less loop pairs identified. Moreover, we also define a binary image center $\mathbf{p}_{\mathcal{M}}$

$$\mathbf{p}_{\mathcal{M}} = \frac{\sum_{\mathbf{u} \in \mathcal{M}} \mathcal{M}(\mathbf{u}) \cdot \mathbf{u}}{\mathcal{F}(\mathcal{M})}, \quad (4.4)$$

where $\mathbf{u}$ is the 2D coordinate of each pixel in the binary image $\mathcal{M}$. The binary image center $\mathbf{p}_{\mathcal{M}}$ represents the center of saliency distribution of a place. Different places have different saliency distribution centers so that $\mathbf{p}_{\mathcal{M}}$ is unique. By setting a threshold on $\mathbf{p}_{\mathcal{M}}$ we can further filter out unmatched pairs. An example of binarized saliency map based fast indexing is shown in Figure 4.3. We randomly pick up 3 frames from the KITTI dataset [18]. The first and second frames are taken at the same place but at different times, while the third frame is taken at another similar place. The first and second frames are considered as a loop closure pair but the first and third frames are not. By applying the fast indexing, we can calculate the ξ between frames: $\xi_{12} = 67\%$ and $\xi_{13} = 20\%$. Intuitively we can tell that the second frame looks much more similar to the first frame.

4.3.3 Precise Loop Closure Detection

The fast image retrieval is able to remove most unmatched pairs. However, the binarized saliency map only considers the structure of the content which is fast but

FIGURE 4.3: Examples of binarized saliency map extraction.

not accurate enough. Note that the traditional methods using the SURF feature descriptor have achieved good performance in matching images, we implement feature-based comparison to further increase the precision.

The SURF feature points are extracted from each frame due to its high precision in image matching [109]. And we use SURF descriptors to examine each image pair. Figure 4.4 shows an example of feature matching. The number of matched feature points reveals the similarity of the image pair and we can conclude the loop closure by setting a threshold on the number of matched features.

FIGURE 4.4: An example of SURF feature points matching.

4.4 Experiment Results

To demonstrate the robustness, we test the proposed method with different datasets including the KITTI dataset, New College dataset and City Center dataset [18, 120, 121]. The information of respective dataset is given in Table 4.1. The important performance indexes for loop closure detection are recall rate, recall precision and speed. Recall precision refers to the ratio of correct loop closure detection against total loop closure detected. The recall precision must be high because any false positive may cause divergence in global optimization. Recall rate refers to the number of correct loop pairs detected against total loop pairs which can be collected from the ground truth. In this section, we provide a detailed analysis of the proposed method.

Dataset	Image Size	Source of Ground Truth
KITTI	370×1226	D-GPS
New College	640×480	GPS
City Center	640×480	GPS

TABLE 4.1: Information of different datasets.

4.4.1 Evaluation on Public Datasets

We conduct the test on an intel NUC mini computer which is popularly used in robotics related applications. The algorithm is coded in C++ and is tested on ROS. The proposed method is tested with different datasets mentioned above. The total number of loop closure pairs is counted based on the ground truth trajectory. For each query frame and candidate frame, we verify the Euclidean distance between two frames. When the distance of two frames is less than 5 m , it is considered as a loop pair. Note that the neighbour frames are not considered as a loop pair. An example of the proposed loop closure detection approach on the KITTI dataset is shown in Figure 4.5. In the Figure 4.5(a) we plot the trajectory of the camera in black and mark the ground truth of loop closure detection with black circle. The loop closure detection result is shown in Figure 4.5(b), where the detected loop pairs are marked with red circles. Each circle refers to a loop closure pair. No false positive detected and most loop closure places are identified. The proposed

method achieves a recall rate of 54.9% and a recall precision of 100%. This is satisfactory since most loop closure places are identified. More test results can be found in Table 4.2. In total there are 5 sequences with loop closure from the KITTI dataset and we test on these sequences. Our method achieves more than 50% recall rate on average without any false positives. It also achieves 20% recall rate on New College dataset and 27% recall rate on City Center dataset without false positive. In the meanwhile, our method is able to run at a high speed of 10 *Hz* on average. It is able to provide real-time loop information to a SLAM system and reduce localization drifts.

(A) Ground truth of KITTI sequence 00.

(B) Loop closure detection result.

FIGURE 4.5: Loop closure detection result of the proposed method.

Dataset	Mean Time (<i>ms</i>)	Average Recall Rate (%)	Precision (%)
KITTI	130	54.9	100
New College	92	20.9	100
City Center	86	27.7	100

TABLE 4.2: Loop detection results of our approach.

4.4.2 Comparison with Other Methods

We further compare our method with existing methods such as FABMAP, DBoW2 [55, 58]. To be consistent, all experiments are conducted on the same Intel NUC mini computer. FABMAP and DBoW are open-sourced in C++ and they are evaluated on the local computer. In order to have a clear comparison, we pick the largest dataset with loop closure for demonstration. Specifically, we use KITTI sequence 00, KITTI sequence 02 and KITTI sequence 05 with more than 10k frames in total.

We firstly test on KITTI sequence 05 and the results are shown in Figure 4.6. In the experiment, we fine-tuned the threshold in both FABMAP and DBoW2 in order to get the best recall rate and recall precision for different datasets. In comparison, our approach does not require fine tuning for a specific dataset. Besides, both FABMAP and DBoW2 require offline training of similar datasets in advance, while the proposed method does not. The details of the experiment results on other datasets are shown in Table 4.3. In terms of computational cost, the speed of the proposed method is 3 times faster than DBoW2 and 9 times faster than FABMAP. From the results, it can be seen that our approach also provides reliable precision and recall rate. A video demonstration of the proposed method can be found online¹. However, the extracted binarized saliency map only considers the saliency distribution, while the image content is ignored. When the viewing angle changes, the saliency distribution also changes. Thus, it is not robust to rotation change.

¹<https://youtu.be/YCRd3N0LwSA>

FIGURE 4.6: Comparison of binarized saliency map extraction with existing methods.

Dataset	Sequence 00			Sequence 02			Sequence 05		
	Time (ms)	Recall (%)	Precision (%)	Time (ms)	Recall (%)	Precision (%)	Time (ms)	Recall (%)	Precision (%)
Our Approach	130	54.9	100	129	47.7	100	118	62.5	100
FABMAP	1124	32.2	97.7	1162	23.4	49	1021	35.3	98
DBoW2	460	57.2	100	448	38.9	100	355	54.0	100

TABLE 4.3: Performance comparison of FABMAP, DBoW2 and the proposed method on different datasets.

4.5 Conclusion

In this chapter, we have reviewed different approaches used on back-end SLAM systems and presented a fast loop closure detection method via binarized saliency map. Traditional approaches such as FABMAP and DBoW2 use feature descriptors to compress the image content and build a descriptor vocabulary for indexing. However, these methods require intensive mathematical calculation to estimate the similarity of two images, which is less efficient than logical operation. Note that operation on binary images is faster than feature descriptor, we apply binarized saliency map based image retrieval on top of the existing feature-based method to reduce the computational cost. Specifically, a new framework has been proposed which consists of three parts: binarized saliency map construction, fast image retrieval and precise loop closure detection. The experiment result has demonstrated that the proposed method is able to detect loop closure places without false positives. In addition, no offline training is required in our approach so that it is easy for implementation.

Chapter 5

Online Visual Place Recognition via Saliency Re-identification

5.1 Introduction

In the previous chapter, we introduced a vision-based LCD method via binarized saliency map. We have shown that the retrieval of binarized saliency map is computationally efficient. However, as mentioned in Section 4.4, the extracted binarized saliency map only considers the saliency distribution, while the image content is ignored. It is not accurate enough and a feature-based method is required to guarantee the recall precision. When the viewing angle changes, the saliency distribution also changes. However, it is still computationally inefficient to extract local features such as SURF. In this chapter, we further explore the potential of using saliency information for LCD to improve the performance and robustness.

Inspired by the fact that human beings always recognize a place by remembering objects or landmarks that are more appealing or interesting than others, not simply by remembering point features [122], we further formulate place recognition as saliency re-identification, which is natural and straightforward. Rather than formulating the saliency information as binarized saliency map in the previous chapter, we introduce KCC [123] to achieve fast retrieval on the extraction saliency information. We will show that KCC is robust to image variations such as rotation change. Moreover, we find that both saliency detection and retrieval can be performed in the frequency domain, which can further reduce the computational cost.

The experiments show that the proposed method achieves competitive accuracy and much higher speed than state-of-the-art feature-based methods. The proposed method is open-sourced and publicly available¹.

The main contributions of this chapter are:

- We propose a novel framework for visual place recognition via saliency re-identification, which is coded in C++ and open sourced.
- We propose to detect and identify the salient regions in the frequency domain by taking advantage of computationally efficient element-wise operation.
- The experiments show that our method is much faster than existing feature-based methods without reducing recall rate and precision. It does not require offline training or loading pre-trained vocabulary.

This chapter is organized as follows: Section 5.2 reviews the related works on visual place recognition. Section 5.3 describes the details of the proposed framework. Section 5.4 shows experimental results and comparison with existing methods, followed by the conclusion in Section 5.5.

5.2 Related Works

In Section 4.2, we reviewed existing feature-based approaches such as DBoW2. However, offline training is required. In this section, we will review some online training-based methods that appeared in recent years. Moreover, we will also review the most recent deep learning-based methods and saliency-based methods.

5.2.1 Feature-based Methods via Online Training

Instead of building an offline-trained visual vocabulary to discretize local feature space, some recent works introduce more efficient feature retrieval to make on-line training possible. For example, Emilio *et al.* [61] introduce a vocabulary

¹<https://github.com/wh200720041/SRLCD.git>

maintenance strategy called dynamic island that groups similar images. The dynamic island identifies repetitive registration of the same descriptor across multiple frames. The database is kept at a small scale by removing those redundant descriptors. Schlegel *et al.* [62] introduce an online trained Hamming distance embedded binary search tree (HBST) to do image retrieval which is much faster than traditional FLANN matching methods. However, the memory cost is huge since raw descriptors are used to build the incremental visual vocabulary tree. Applying dimensional reduction on local features is also an alternative solution. Carrasco *et al.* [64] adopt the idea from image encoding to extract local descriptors. Hash coding is applied to the local features array collected from a single image so that the extracted features are more compact. Gehrig *et al.* [65] directly apply Principal Component Analysis (PCA) for BRISK [66] feature. Instead of searching on pre-trained visual vocabulary, k-nearest neighbor (K-NN) search is performed on the projected descriptors. It achieves retrieval speed at the level of a millisecond for each query. However, those methods still require local feature extraction, which is computationally expensive.

5.2.2 Deep Learning-based Methods

The recent success of Convolutional Neural Networks (CNN) in computer vision [67] has triggered another research trend for visual place recognition. For example, in [68], a multi-scale feature encoding method is introduced by training two CNN architectures. The generated CNN features are viewpoint invariant, hence a large performance improvement is achieved. Inspired by the traditional image retrieval method, i.e., Vector of Locally Aggregated Descriptors (VLAD), Relja *et al.* propose the NetVLAD [72] to learn CNN parameters in an end-to-end manner for place recognition. Hausler *et al.* [73] combine both feature-based methods and the CNN techniques. The query image is trained by combining Sum of Absolute Difference (SAD), Histogram of Oriented Gradients (HOG) [74], CNN spatial Max pooling and CNN spatial Arg-Max pooling. However, those CNN-based methods achieve high accuracy at the expense of a huge computational burden, which is sometimes infeasible for real-time systems.

5.2.3 Saliency Re-identification-based Methods

Compared to feature matching, saliency/landmark recognition is a more straightforward way for place recognition. Kim *et al.* adopt inter-image saliency for place recognition with BoW model [75], which is shown to be effective in tackling challenging scenarios such as underwater hull inspection. Birem *et al.* extract salient regions through a conspicuity map from intensity, color and orientation [76]. The similarity of image frames is measured based on the content and location of salient regions. By taking advantage of the feature extraction ability of CNN, Xin *et al.* investigate how to localize discriminative visual landmarks that positively contribute to the similarity measurement [77]. In particular, a landmark localization network is designed to indicate discriminative regions of an image. It has been shown that the saliency re-identification-based methods are more robust than the feature-based methods in some scenarios. However, they have not been widely applied to robotic systems due to the high computational cost.

5.3 System Overview

The proposed framework mainly consists of two components: saliency detection and loop identification, both of which will be processed in the frequency domain to take advantage of the efficiency of element-wise operation. The system overview can be found at Figure 5.1. In the first module, saliency extraction identifies the Visual Salient Regions (VSRs) from the input frame and removes those unqualified segments. In the second module, the VSR is then compared with a dataset via Kernel Cross-Correlation (KCC) [123] to examine the likelihood of re-occurrence.

5.3.1 Saliency Detection

As mentioned before, plenty of approaches for saliency detection have been proposed [122] to mimic the human psychological process. However, the psychological definition of saliency is subjective and can vary in different contexts [117]. In this work, we mainly focus on saliency re-identification for visual place recognition, thus it is not necessary to strictly follow the psychological definition. Similar to

FIGURE 5.1: System overview of the proposed method.

the previous chapter, we introduce a log-spectral residual approach to identify visual saliency which has been widely applied in practice [119]. Based on the saliency extraction result, it is suitable enough for visual place recognition. Moreover, log-spectral analysis can be computed in an efficient element-wise manner, which can subsequently reduce the computational cost.

5.3.1.1 Definition

In Chapter 4, we have presented the basic procedure of log-spectral residual for saliency detection. In this section, we further analyze the properties of log-spectral residual. Denote the 2D FFT $\mathcal{F} : \mathbb{C}^{m \times n} \mapsto \mathbb{C}^{m \times n}$ as $\hat{\cdot}$, given an input image $\mathbf{I}$, its FFT can be expressed as $\hat{\mathbf{I}}$. We denote the magnitude spectrum and phase spectrum as $\mathcal{A}(\hat{\mathbf{I}})$ and $\mathcal{P}(\hat{\mathbf{I}})$, respectively, thus the log-spectral residual $\mathcal{R}(\hat{\mathbf{I}})$ can be defined as:

$$\mathcal{L}(\hat{\mathbf{I}}) = \log(\mathcal{A}(\hat{\mathbf{I}})), \quad (5.1a)$$

$$\mathcal{R}(\hat{\mathbf{I}}) = \exp(\mathcal{L}(\hat{\mathbf{I}}) - \mathbf{f}_{ave} * \mathcal{L}(\hat{\mathbf{I}})), \quad (5.1b)$$

where $\mathbf{f}_{ave}$ is a normalized average filter of size $k \times k$ and $*$ denotes the cross-correlation. A saliency map $\mathcal{M}$ can be extracted by converting the residual back

to spatial domain via inverse fast Fourier transform (IFFT)

$$\mathcal{O}(x) = \mathcal{G}_\sigma * \mathcal{F}^{-1}[\mathcal{R}(\hat{\mathbf{I}}) \cdot \exp(\mathbf{j} \cdot \mathcal{P}(\hat{\mathbf{I}}))]^2, \quad (5.2a)$$

$$\mathcal{M} = \begin{cases} 1 & \text{if } \mathcal{O}(x) > E(\mathcal{O}(x)) \cdot \gamma, \\ 0 & \text{otherwise.} \end{cases} \quad (5.2b)$$

The saliency map (5.2) implies the saliency distribution of the original image [124]. With such distribution, we can derive saliency regions $\mathbf{x}_1, \mathbf{x}_2, \dots \in \mathbf{I}$ by carrying out pixel connectivity analysis on the saliency map $\mathcal{M}$.

The log-spectral residual approach provides a fast solution for saliency detection. However, not all extracted saliency information is suitable for place recognition, e.g., the regions that are too small or too dark or bright. Therefore, we need some extra rules to filter out the unqualified regions in order to improve the robustness.

It is observed that the detected saliency with high contrast and more edges is often unique and highly distinguishable, which is suitable for place recognition. Therefore, we adopt a discriminative strategy using contrast density ϕ_x and edge complexity ρ_x defined by

$$\phi_{\mathbf{x}} = \overline{(\mathbf{x} - \bar{\mathbf{x}})^2}, \quad (5.3a)$$

$$\rho_{\mathbf{x}} = \frac{\mathcal{C}(\mathbf{x})}{|\mathbf{x}|}, \quad (5.3b)$$

where $\mathbf{x} \in \mathbf{I}$ is the extracted salient region, $\bar{\mathbf{x}}$ is the mean pixel intensity, $\mathcal{C}$ is the total number of edge pixels after applying Canny edge filter, and $|\mathbf{x}|$ computes the total size of the salient region (i.e., pixel number). In experiment, we find that this strategy can remove most of the less informative regions and even perform better than more complicated rules.

The output of the log-spectrum residual algorithm is a mask with rough object shapes. For better matching, we enclose those masks with minimum bounding boxes. An example of saliency detection is shown in Figure 5.2. It can be seen that six distinct regions, including traffic signs, ad boards, phone booths, and cars are extracted. Although it is inevitable that some dynamic objects such as cars are extracted, they will be further removed by consistency verification in Section 5.3.3.

FIGURE 5.2: An example of saliency detection from the KITTI dataset. (a) mask of the detected saliency map; (b) the detected salient regions enclosed by red bounding boxes.

5.3.1.2 Property

One of the reasons that we take the log-spectral residual for saliency detection is that the detected results are translation, rotation and scale equivariant. This is crucial for visual place recognition. Recall that the corresponding FFT of translation-rotated image

$$\mathbf{I}_2(x, y) = \mathbf{I}_1(x \cos \theta + y \sin \theta - x_0, -x \sin \theta + y \cos \theta - y_0), \quad (5.4)$$

where θ is the rotation and (x, y) are the translation. It is related by [125]

$$\hat{\mathbf{I}}_2(u, v) = e^{-j2\pi(ux_0+vy_0)} \cdot \hat{\mathbf{I}}_1(u \cos \theta + v \sin \theta, -u \sin \theta + v \cos \theta) \quad (5.5)$$

It can be seen that the factor $e^{-j2\pi(ux_0+vy_0)}$ does not change the magnitude spectrum. This means that the log-spectrum residual $\mathcal{R}(\hat{\mathbf{I}}_2)$ is just a rotational replica of $\mathcal{R}(\hat{\mathbf{I}}_1)$, resulting in a rotated saliency map $\mathcal{M}$ and unchanged salient regions. This also holds for scale transforms. Suppose $\mathbf{I}_2$ is a scale transform of $\mathbf{I}_1$ with scale factor s , we have

$$\begin{aligned}\hat{\mathbf{I}}_2(u, v) &= \frac{1}{s^2} \hat{\mathbf{I}}_1(u/s, v/s), \\ \mathcal{L}(\hat{\mathbf{I}}_2(u, v)) &= \mathcal{L}(\hat{\mathbf{I}}_1(u/s, v/s)) - 2 \log s.\end{aligned}\tag{5.6}$$

This implies that the log-spectrum is scaled and the same salient region can be extracted. Those properties are crucial for robust saliency retrieval.

5.3.2 Saliency Retrieval

Although the saliency extraction is robust and consistent, the extracted salient regions can slightly differ from size, viewing angles, illumination, etc. Traditional fast image comparison methods such as hash coding method and Histogram of Oriented Gradient (HoG) method are easily affected by those factors. Therefore, a transform-invariant encoding algorithm is necessary for saliency retrieval. Considering the computational cost, we apply the KCC [123], which is able to match two image areas directly in the frequency domain and is invariant to affine transforms such as translation, rotation and scale. More specifically, we compare the current VSR $\mathbf{z}$ with the VSR $\mathbf{x}$ from the database. The purpose of training is to find an optimal correlator that is unique for each salient region. Then the correlator is used to examine the similarity of two salient regions in the retrieval stage.

5.3.2.1 VSR Training

Recall that the cross-correlation of two 2D signals $\mathbf{g} = \mathbf{x} * \mathbf{h}$ becomes $\hat{\mathbf{g}} = \hat{\mathbf{x}} \odot \hat{\mathbf{h}}^*$ in the frequency domain, where the operator $*$, $\odot$ and $\mathbf{h}^*$ represent cross-correlation, element-wise multiplication and complex conjugate, respectively. Given correlation output $\mathbf{g}$, the kernel cross-correlation is defined as

$$\hat{\mathbf{g}} = \hat{\kappa}_{\mathbf{z}}(\mathbf{x}) \odot \hat{\mathbf{h}}^*,\tag{5.7}$$

where $\kappa_{\mathbf{z}}(\mathbf{x})$ is a kernel matrix given by

$$\kappa_{\mathbf{z}}(\mathbf{x}) = \begin{bmatrix} \kappa(\mathbf{x}, \mathbf{z}_{11}) & \dots & \kappa(\mathbf{x}, \mathbf{z}_{1n}) \\ \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}, \mathbf{z}_{m1}) & \dots & \kappa(\mathbf{x}, \mathbf{z}_{mn}) \end{bmatrix}, \quad (5.8)$$

with $\kappa(\cdot, \cdot)$ being a kernel function. $\mathbf{x}$ and $\mathbf{z}$ are the candidate VSR and target VSR. $\mathbf{z}_{ij}$ is the translational shift of $\mathbf{z}$ with i pixel shifts in horizontal and j pixel shifts in vertical direction. For a VSR $\mathbf{z}$ of size $m \times n$, there are $m \times n$ translational shifts in total. Hence we can have $m \times n$ kernel function $\kappa(\mathbf{x}, \mathbf{z}_{11}), \dots, \kappa(\mathbf{x}, \mathbf{z}_{mn})$, which forms a kernel matrix of dimension $m \times n$. Specifically, we use the Gaussian kernel as an example. Let

$$\begin{aligned} \kappa(\mathbf{x}, \mathbf{z}_{ij}) &= \phi(\|\mathbf{x} - \mathbf{z}_{ij}\|^2), \\ \phi(\mathbf{x}) &= \exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right), \end{aligned} \quad (5.9)$$

where σ is a constant. Denote function $\Phi(\cdot)$ as the matrix form of Gaussian kernel function, where each element is a single Gaussian kernel $\phi(\cdot)$

$$\Phi(\mathbf{A}) = \begin{bmatrix} \phi(\mathbf{A}_{11}) & \dots & \phi(\mathbf{A}_{1n}) \\ \vdots & \ddots & \vdots \\ \phi(\mathbf{A}_{m1}) & \dots & \phi(\mathbf{A}_{mn}) \end{bmatrix}, \quad (5.10)$$

with $\mathbf{A}_{ij}$ being the i^{th} row and j^{th} column element of matrix $\mathbf{A}$. Since $\mathbf{z}_{ij}$ is the translational shifts of $\mathbf{z}$, the kernel matrix $\kappa_{\mathbf{z}}(\mathbf{x})$ becomes

$$\begin{aligned} \kappa_{\mathbf{z}}(\mathbf{x}) &= \begin{bmatrix} \phi(\|\mathbf{x} - \mathbf{z}_{11}\|^2) & \dots & \phi(\|\mathbf{x} - \mathbf{z}_{1n}\|^2) \\ \vdots & \ddots & \vdots \\ \phi(\|\mathbf{x} - \mathbf{z}_{m1}\|^2) & \dots & \phi(\|\mathbf{x} - \mathbf{z}_{mn}\|^2) \end{bmatrix} \\ &= \Phi(\|\mathbf{x}\|^2 + \|\mathbf{z}\|^2 - 2 \cdot [\text{Tr}(\mathbf{x}^T \mathbf{z}_{ij})]_{mn}), \\ &= \Phi(\|\mathbf{x}\|^2 + \|\mathbf{z}\|^2 - 2 \cdot \mathbf{x} * \mathbf{z}) \\ &= \Phi\left(\frac{\|\hat{\mathbf{x}}\|^2}{m \cdot n} + \frac{\|\hat{\mathbf{z}}\|^2}{m \cdot n} - 2 \cdot \mathcal{F}^{-1}(\hat{\mathbf{x}} \odot \hat{\mathbf{z}}^*)\right). \end{aligned} \quad (5.11)$$

Given a desired correlation output $\mathbf{g}$, we expect an optimal $\mathbf{h}$ that maps the kernel matrix $\kappa_{\mathbf{z}}(\mathbf{x})$ to $\mathbf{g}$ such that the sum of squared error (SSE) is minimized in the

frequency domain

$$\min_{\hat{\mathbf{h}}^*} \|\hat{\kappa}_{\mathbf{z}}(\mathbf{x}) \odot \hat{\mathbf{h}}^* - \hat{\mathbf{g}}\|^2 + \lambda \|\hat{\mathbf{h}}^*\|^2. \quad (5.12)$$

The solution of $\hat{\mathbf{h}}^*$ can be derived by setting the partial derivative to zero

$$\frac{\partial}{\partial \hat{\mathbf{h}}^*} (\|\hat{\kappa}_{\mathbf{z}}(\mathbf{x}) \odot \hat{\mathbf{h}}^* - \hat{\mathbf{g}}\|^2 + \lambda \|\hat{\mathbf{h}}^*\|^2) = 0. \quad (5.13)$$

Thus it has a closed-form solution

$$\hat{\mathbf{h}}^* = \frac{\hat{\mathbf{g}}}{\hat{\kappa}_{\mathbf{z}}(\mathbf{x}) + \lambda}, \quad (5.14)$$

where operator $\div$ is the element-wise division. In the training stage, the optimal correlator $\mathbf{h}$ is solved with $\mathbf{x} = \mathbf{z}$. Note that due to the efficiency of Equation (5.14), we can obtain $\hat{\mathbf{h}}^*$ online, hence no pre-trained database is required.

5.3.2.2 VSR Retrieval

As shown in [123], KCC is equivariant to affine transforms, i.e., if input $\mathbf{x}$ is transformed (translation, scale and rotation), the output $\mathbf{g}$ will be translated accordingly. Therefore, the transform of a test sample $\mathbf{x}$ can be estimated by examining the translation of correlation output in (2.29). For simplicity, we set $\mathbf{g}$ in (5.14) as

$$\mathbf{g}(x, y) = \begin{cases} 1, & \text{if } x = 0, y = 0 \\ 0, & \text{otherwise} \end{cases} \quad (5.15)$$

Because of noise and distortion, we cannot always obtain a pure translational response of $\mathbf{g}$ in (6.6). Therefore, we can take the maximum value in the test output as the similarity ζ between training and test samples, which will be invariant to the affine transforms of $\mathbf{x}$.

$$\zeta = \max \mathcal{F}^{-1} \left(\hat{\kappa}_{\mathbf{z}}(\mathbf{x}) \odot \hat{\mathbf{h}}^* \right). \quad (5.16)$$

The procedure of saliency retrieval using KCC is presented in Figure 5.3, in which a loop closure candidate is detected.

FIGURE 5.3: The procedure of saliency retrieval. The same traffic sign is identified when a car re-visits the same place.

5.3.3 Consistency Verification

As mentioned before, re-identification through saliency can improve computational efficiency. However, the matched salient region may also cause false positive because the approach only identifies the similarity between query and candidate VSRs. Therefore, it is necessary to check consistency before closing the loop since any false positive will lead to localization and mapping failure easily.

The consistency verification is performed only on detected loop pairs by brute force matching using ORB features. Despite being computationally expensive, matching on raw feature descriptors is able to remove most of the false positives produced by saliency re-identification, implying that traditional feature matching is a good complementary technique for saliency matching. We extract 1000 descriptors from the current image and the candidate image. A loop closure is determined when the number of matched descriptor pairs exceeds a certain threshold. Fortunately, the loop closure does not occur frequently so that the computational cost is less significant in comparison to the entire process of place recognition. In the practical experiment, it costs less than 5% of total computational expense on average so that such computational cost is worthy to guarantee the geometrical consistency.

5.4 Experiment Results

5.4.1 Setup and Metrics

In our experiment, the proposed method and the compared methods are coded in C++ and tested on an Intel i7-8700 3.2 GHz CPU. The source codes of the proposed

method and the compared methods are publicly available². The algorithms of the compared methods are downloaded from their released source code respectively.

The parameter list used in this chapter is presented in Table 5.1. Three metrics, namely recall precision, recall rate and computational cost, that are widely used to evaluate place recognition, are adopted in this evaluation.

Parameter	Description	Value
σ	Standard deviation of Gaussian filter $\mathcal{G}_\sigma$	2.0
n	Size of average filter $\mathbf{f}_{ave}$	7
ϕ_{th}	Threshold of contrast density	58
ρ_{th}	Threshold of edge complexity	0.5
η_{th}	Threshold of KCC similarity test η	0.4

TABLE 5.1: Parameter list.

5.4.2 Performance Evaluation

The evaluation is performed based on various public datasets that are popularly used for loop closure detection, including KITTI dataset[18], TUM dataset[20], New College dataset[126], City Center dataset[126] and Oxford RobotCar dataset[127]. They include various scenarios such as indoor localization and autonomous driving, etc. The datasets contain large scale collections of up to 35,000 frames (RobotCar dataset), medium scale collections of 5,000 frames (KITTI dataset) and small scale collections of 1,000 frames (New College dataset). Specifically, TUM dataset [20] is mainly collected in the indoor environment, while the rest of datasets are collected in the outdoor environment. The details are listed in Table 5.2. Similar to Chapter 4, the ground truth of loop closure detection is collected based on D-GPS/Vicon information. For each query frame and candidate frame, we verify the Euclidean distance between two frames. When the distance of two frames is less than 5 *m* in outdoor environments or 50 *cm* in indoor environments, they are considered as a loop pair.

²<https://github.com/wh200720041/SRLCD>

Dataset	Description	Image Resolution	Ground Truth
KITTI [18]	Outdoor, dynamic	370×1226	D-GPS
Freiburg3 [20]	Indoor, static	1280×1024	VICON
New College [126]	Outdoor, dynamic	640×480	GPS
City Center [126]	Outdoor, dynamic	640×480	GPS
RobotCar [127]	Outdoor, dynamic	1280×960	D-GPS

TABLE 5.2: Details of datasets used for evaluation.

5.4.2.1 Results

We firstly use the KITTI dataset for demonstration. Specifically, we test the proposed method on KITTI sequence 00 and sequence 05 since these two sequences are large compared to the rest sequences. Figure 5.4 shows the examples of loop closure detection by the proposed method and the comparison with the ground truth. It can be seen that most revisited places are identified by our approach.

More results can be found in Figure 5.5. We plot the results from large datasets for illustration. The datasets include more than 60,000 images in total and the largest dataset contains around 35,000 images. It can be seen that most of the re-visited places are identified and there is no false positive.

5.4.2.2 Precision-Recall Curve

We next evaluate the proposed method on all datasets and record the precision-recall curve. The results are shown in Figure 5.6. In particular, RobotCar 2014.05, 2014.06, 2014.08, and 2014.12 refer to RobotCar sequence 2014_05_14_13_50_20, 2014_06_26_09_53_12, 2014_08_11_10_22_21, and 2014_12_02_15_30_08, respectively. It is notable that our method achieves 100% recall precision over all datasets, which is accurate. In some challenging outdoor dynamic environments, e.g., Oxford RobotCar dataset with 35,000 images, our approach achieves a high recall rate up to 90%. More importantly, our approach is able to run up to 50Hz, which is computationally cost-effective for many robotic applications. Compared to the method discussed in Chapter 4, the recall rate is improved from 54.9% to 71% on KITTI sequence 00, from 20.9% to 40% on the New College dataset and from

FIGURE 5.4: Examples of place recognition on the KITTI dataset. (a, b) and (c, d) are the results of sequence 00 and sequence 05, respectively. The trajectory color changes from light to dark with respect to time. The matches from ground truth are marked with black, while the detected matches are highlighted with red. It can be seen that most of the re-visited places are recognized.

27.7% to 42% on the City Center dataset. It can be seen that the recall rate is higher by using saliency information instead of binarized saliency map.

5.4.3 Comparison

To further illustrate the efficiency of the proposed method, we compare it with state-of-the-art feature-based methods such as iBoW [61], HALOC [64], DBoW2 [58], and HBST [62]. In particular, DBoW2 was proposed in the early stage and yet is still one of the most popular methods. It has been used for ORB SLAM [111] and LDSO [128]. The rest of methods were introduced in recent years and they have been shown to be effective in loop closure detection. More specifically,

FIGURE 5.5: Examples of place recognition on public datasets. The trajectory from GPS is plotted from light grey to dark grey with respect to time. The loop closure reported by the proposed method is marked with a red dot. (a, b, c) are collected from KITTI sequence 02, 07 and 09 respectively; (d, e, f) are collected from Oxford RobotCar sequence 2014_05_14_13_50_20, 2014_08_11_10_22_21 and 2014_12_02_15_30_08 respectively. Our method achieves 80% recall rate with no false positive.

DBoW2 and iBoW require offline training while the rest of the methods perform online training. For all compared methods, 1000 local features, which is a typically recommended feature extraction number, are extracted for each frame. To note that, HBST is evaluated at a quarter of the original video size due to its huge memory consumption and time cost.

5.4.3.1 Efficiency Evaluation

For computational efficiency, we use KITTI sequence 00 for example since the speed comparison is similar across different datasets. Figure 5.7 (a) shows the performance of the saliency extraction compared to feature extraction from other methods. As mentioned before, saliency extraction is only of linear computational complexity in the frequency domain. Hence we can see that it is faster than local feature extraction from the results. In iBoW and HBST, raw ORB features are used and hence the computational cost is also low. In comparison, HALOC and DBoW2 require further dimension reduction hence the feature extraction time is

FIGURE 5.6: Precision-recall curve of the proposed method on different datasets.

higher. Figure 5.7 (b) shows the overall computational cost of respective methods. Our approach achieves the fastest speed among all the approaches, i.e., a few times faster than DBoW2 and HALOC. Our method can be used on mobile robots with limited computational power.

5.4.3.2 Overall Performance

Figure 5.8 compares the overall performance of different methods on KITTI sequence 00. HBST performs feature matching on raw image descriptors and achieves the highest recall rate. However, both the computational cost and memory cost are too high to build the binary search tree so that it can be difficult to be implemented in practical applications. In comparison, our approach provides a feasible solution for visual place recognition that is computationally efficient and accurate enough. Therefore, our approach achieves a good trade-off between precision and efficiency. It is also notable that our method requires no offline training or vocabulary loading, making it more feasible for real-time robotic applications.

(A) Feature/Saliency Extraction.

(B) Running Time.

FIGURE 5.7: Performance evaluation. (a) comparison of saliency extraction with feature extraction methods on KITTI sequence 00; (b) time cost comparison of different approaches on KITTI sequence 00. Our approach achieves much faster speed than existing methods.

FIGURE 5.8: Overall performance of different approaches.

5.5 Conclusion

In this chapter, we have presented a novel framework for visual place recognition via saliency re-identification. In the previous chapter, we proposed an LCD method via saliency distribution. However, the performance was limited since the image content is ignored. To further improve the performance, we leveraged the VSRs which contain the salient image contents. Moreover, we introduced KCC rather than binary operation to identify the loop pairs to increase the matching precision. The proposed framework mainly consists of two components: saliency detection and retrieval. It was also found that both tasks can be performed in the frequency domain to take advantage of the efficiency of element-wise operation, resulting in overall computational complexity of only $\mathcal{O}(N \log N)$. The experiments have shown that our method achieves a good trade-off between computational cost and accuracy.

Chapter 6

Intensity Scan Context: Coding Intensity and Geometry Relations for Loop Closure Detection

6.1 Introduction

In the previous chapter, we introduced a vision-based loop closure detection approach for re-localization. VPR often explores the visual features of a place. However, a revisited place usually varies from light illumination, weather and viewing angle, so that the visual features can be much different. In comparison, LiDAR is less sensitive to environmental changes. In this chapter, we will explore the potential of using LiDAR information for SLAM re-localization.

Existing works on LiDAR-based LCD often leverage the matching of 3D descriptors such as Fast Point Feature Histogram (FPFH) [40], Fast Laser Interest Region Transform (FLIRT) [81] and Signature of Histograms of Orientations (SHOT) [129]. These descriptors explore either global or local geometry information such as surface normal or neighbour points distribution, leaving the intensity information unused. The main justification is that intensity information is less straightforward than geometry measurement [130], since it is affected by not only target surface characteristics (e.g., roughness, surface reflectance), but also acquisition geometry (e.g., distance) and instrument effects (e.g., transmitted energy) [131]. However, the intensity channel reveals the reflectance structure of the surrounding

environment, e.g., retro-reflective material such as metal plate usually returns a high value and concrete returns a low value. This information is often unique for different places. Moreover, recent works have obtained some preliminary results that intensity reading can be effective for place recognition [132, 133].

In this chapter, we propose a novel global descriptor, Intensity Scan Context (ISC), that integrates both geometry and intensity characteristics for loop closure detection. We firstly explain how intensity information can be distinguishable to represent a place. Then we use ISC as a global signature for place recognition. To further improve the efficiency for loop closure detection, a two-stage hierarchical place re-identification strategy is proposed, including binary-operation based fast geometry retrieval and intensity structure matching. To evaluate the performance of the proposed ISC descriptor, our method is tested under different scenarios including outdoor autonomous driving and indoor warehouse robot navigation. The results show that the proposed approach achieves a higher recall rate and recall rate than existing geometric-only loop closure detection methods.

The main contributions of this chapter are as follows:

- We propose a novel global descriptor for the 3D LiDAR scan that integrates both geometry and intensity characteristics.
- An efficient loop closure detection strategy based on a two-stage hierarchical ISC-based re-identification is proposed. It only costs 1.2 *ms* per query on average.
- A thorough evaluation of the proposed descriptor, including both local experiment and public dataset test, is conducted.

This chapter is organized as follows: Section 6.2 reviews the related works on LiDAR-based approaches for loop closure detection. Section 6.3 describes the idea of using Intensity Scan Context for place recognition, followed by the two-stage hierarchical place re-identification. Section 6.4 shows experiment results and comparison with existing works, followed by the conclusion in Section 6.5.

6.2 Related Works

Existing works on loop closure detection consist of vision-based methods and LiDAR-based methods according to different perception systems used. Vision-based approaches were developed for place recognition in the early stage. As mentioned in Chapter 4 and Chapter 5, these methods often leverage on BoW model, e.g., FAB-MAP [110] and DBoW2 [58]. They are widely used in visual SLAM such as ORB SLAM [111] and LDSO [128]. However, image stream is not resistant to light illumination or view-point, so that vision based place recognition is not robust enough in practice. Recently some works target to tackle environmental changes such as [134, 135], but are limited to some specific scenarios.

In comparison, LiDAR detection is more robust due to its illumination and view-point invariant properties and is subsequently introduced for LCD. Existing works on LiDAR-based place recognition strive for an efficient local descriptor or place signature that can accurately and concisely represent a place. One of the most popular local descriptors is FPFH [40] which explores the local surface normal of each neighbour point. It is effective in estimating affine transforms between two point clouds and is used for place recognition in the later work such as [78]. Bosse *et al.* propose a probabilistic voting approach based on Gasalt3D descriptor [84]. Despite the good performance achieved, the point cloud retrieval is inefficient since the Gasalt3D descriptor is high dimensional.

Re-identification on the local descriptor usually requires key-point extraction and massive local geometry calculation. In comparison, matching on a global descriptor is more efficient in place recognition. Rizzini introduces a novel descriptor named GLAROT that encodes the relative geometric position of key-point pairs into a histogram [82]. The experiment result achieves very satisfactory recall precision and recall rate. However, building key-point relations is still computationally expensive. Kim *et al.* propose a Scan Context Image (SCI) which projects a laser scan into a global descriptor. The matching of SCIs only requires element-wise multiplication so that the query speed is fast. However, the matching precision is not satisfactory and false positives occur during public dataset evaluation.

The global descriptors are more efficient but the performance is not competitive. Recent works find that the performance can be improved by integrating intensity characteristics. Guo *et al.* justify that intensity information can be distinctive for

places and propose a novel local descriptor called intensity signature of histograms of orientations (ISHOT) that consists of both geometry and intensity information. The place re-identification is solved by a probabilistic voting strategy similar to [84]. Despite being computationally expensive, the new descriptor outperforms geometric-only descriptors in place recognition. This inspires our work on finding a more efficient and more accurate descriptor for loop closure detection.

6.3 System Overview

In this section, the proposed method is described in detail. We firstly present the concept of intensity measurement and briefly explain how intensity characteristics can be used for place recognition. Then we present the idea of using ISC as a global 3D descriptor. Furthermore, an efficient two-stage hierarchical ISC-based retrieval is introduced, consisting of fast binary-operation based geometry retrieval and intensity structure matching.

6.3.1 Intensity Calibration and Pre-processing

LiDAR perceives the environment by emitting and receiving laser beams. Generally, the distance value is measured by traveling time while the surface reflectance can be estimated by returned energy level (i.e., intensity). The intensity reading reveals surrounding surface reflectance structure. Existing works on LiDAR have shown that the returned intensity readings vary for different objects [131]. In Figure 6.1, we show an example from the KITTI dataset [18] for demonstration. The point cloud and image present the same place and the intensity structure is interpreted as color in the first image. We pick 3 landmarks including car, road sign and building respectively and highlight them with red rectangles in both Figure 6.1 (a) and Figure 6.1 (b). It is observed that the road sign is highly distinguishable with low energy loss while the building structure (concrete) returns medium intensity. In addition, the reflectance is consistent within the object.

The intensity channel is noisy since it is affected by not only target surface characteristics (e.g., roughness, surface reflectance), but also acquisition geometry (e.g.,

(A) Intensity scan reading of a crossroad.

(B) Camera view of the same place.

FIGURE 6.1: An example of intensity reading from the KITTI dataset. The relationships of intensity information and landmarks are highlighted with red rectangles.

distance) and instrument effects (e.g., transmitted energy) [131]. Hence calibration is necessary in order to reduce the disturbance by other factors. Similar to [133, 136], in this work we calibrate the intensity reading η_r with a mapping function φ

$$\eta_{cal} = \varphi(\eta_r, d), \quad (6.1)$$

where d is the distance reading. The mapping function φ describes the influence of distance on the received energy and can be collected based on an offline experiment. Besides remapping with function φ , the intensity value returned from a 3D LiDAR is an eight-byte integer, e.g., Velodyne VLP-16. It is re-scaled into $[0,1]$ as a floating number for convenience.

In practical applications, some pre-processing of the LiDAR scan is necessary in

order to remove the redundant information. Based on our observation, the LiDAR noise increases with distance. Therefore, the LiDAR reading is firstly filtered by setting a distance threshold L_{max} to remove unreliable points. Moreover, the ground points are often not significant for place recognition so that they are optimized by a method similar to [50] in advance.

6.3.2 Intensity Scan Context

Extraction of local descriptors is often computationally expensive since we need to identify local geometry characteristics such as local norm for every single key-point. Hence to increase the computational efficiency, the intensity information is interpreted as a global descriptor in this work. Inspired by scan context [83, 137] and shape context [138], we introduce intensity scan context that can efficiently integrate both geometry and intensity characteristics into a global signature.

Denote the intensity reading η , geometry reading $[x, y, z]$, number of points n , a LiDAR scan is defined as $\mathcal{P} = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n\}$ with each point $\mathbf{p}_k = [x_k, y_k, z_k, \eta_k]$ in the local Cartesian coordinate. Each point $\mathbf{p}_k$ can be converted into polar coordinate, but only in x-y plane, so that

$$\begin{aligned}\mathbf{p}_k &= [\rho_k, \theta_k, z_k, \eta_k], \\ \rho_k &= \sqrt{x_k^2 + y_k^2}, \\ \theta_k &= \arctan \frac{y_k}{x_k}.\end{aligned}\tag{6.2}$$

The point cloud is then segmented by equally dividing polar coordinates in azimuthal and radial directions into N_s sectors and N_r rings. Each segment is represented by

$$\begin{aligned}\mathcal{S}_{ij} = \{\mathbf{p}_k \in \mathcal{P} \mid & \frac{i \cdot L_{max}}{N_r} \leq \rho_k < \frac{(i+1) \cdot L_{max}}{N_r}, \\ & \frac{j \cdot 2\pi}{N_s} - \pi \leq \theta_k < \frac{(j+1) \cdot 2\pi}{N_s} - \pi\},\end{aligned}\tag{6.3}$$

where $i \in [1, N_s]$ and $j \in [1, N_r]$. The point cloud is then divided into $N_s \times N_r$ subspace. As discussed before, the intensity reading is often consistent for the same object. Since each subspace is much smaller than the whole point cloud, we can assume that the intensity reading does not vary too much within a subspace. Hence for each subspace, a coding function κ can be applied to reduce the intensity

dimension. It is defined as

$$\begin{aligned}\eta_{ij} &= \kappa(\mathcal{S}_{ij}) \\ &= \max_{p_k \in \mathcal{S}_{ij}} \eta_k.\end{aligned}\tag{6.4}$$

Note that if $\mathcal{S}_{ij} \in \emptyset$ (i.e., no scan data), $\eta_{ij} = 0$. Up to this point, we can generate the ISC descriptor Ω by

$$\Omega(i, j) = \eta_{ij}.\tag{6.5}$$

The global signature Ω is a 2D matrix that reveals both geometry and intensity distribution of the environment. As shown in Figure 6.2, we pick a LiDAR scan from the KITTI dataset [18] for illustration. The top figure is the point cloud from top down view with the subspace $\mathcal{S}_{ij}$ marked in red. The bottom figure is the coded ISC descriptor with each pixel calculated by intensity coding function.

6.3.3 Place Re-identification

Place recognition targets to match current place with the previously visited places from the historical database. As more places are visited, the scale of database inevitably increases so that the computational cost grows accordingly. To reduce the computational cost, in this section we propose a two-stage hierarchical intensity scan context retrieval strategy that makes use of fast binary operation to speed up the process of place re-identification.

6.3.3.1 Fast Geometry Re-identification

Most 3D descriptors are histogram based such as unique shape context (USC) [129], ISHOT [133], etc. Matching between histograms can be slow in practice since mathematical operation is inevitable, e.g., multiplication of floating numbers. As mentioned in Chapter 4, binary operation (or logical operation) achieves much faster speed than those mathematical operations. Inspired by [139], we introduce an efficient binary-operation-based geometry re-identification for fast indexing. Given an ISC descriptor Ω , its geometry distribution on the local coordinate can be represented as a binary matrix $\mathcal{I}$

$$\mathcal{I}(x, y) = \begin{cases} false, & \text{if } \Omega(x, y) = 0, \\ true, & \text{otherwise.} \end{cases}\tag{6.6}$$

FIGURE 6.2: A visual illustration of the proposed intensity scan context. Top figure: original point cloud is decomposed into subspace based on geometry characteristics. Bottom figure: derived intensity scan context by intensity projection on each subspace.

For a query ISC descriptor Ω^q , a candidate ISC descriptor Ω^c and their binary transform $\mathcal{I}^q, \mathcal{I}^c$, the geometry similarity can be derived as

$$\varphi_g(\mathcal{I}^q, \mathcal{I}^c) = \frac{XOR(\mathcal{I}^q, \mathcal{I}^c)}{|\mathcal{I}^q|}, \quad (6.7)$$

where $|\mathbf{x}|$ is the total number of elements in $\mathbf{x}$ and $XOR(\mathbf{x}, \mathbf{y})$ refers to element-wise exclusive OR operation between matrix $\mathbf{x}$ and $\mathbf{y}$. Because the column vector in intensity scan context represents azimuthal direction, the rotation of the laser

scan becomes column shift in the ISC descriptor [83]. Hence for place recognition, the viewing angle change can be interpreted as column-shifts of Ω . Therefore, to detect the view-point change, the final score is calculated by

$$\Phi_g(\mathcal{I}^q, \mathcal{I}^c) = \max_{i \in [1, N_s]} \varphi_g(\mathcal{I}_i^q, \mathcal{I}^c), \quad (6.8)$$

where $\mathcal{I}_i^q$ is $\mathcal{I}^q$ shifted by i columns. In the meantime, we can identify the best matched $\mathcal{I}_k^q$ with column-shifts of k that can be used to correct viewing angle change. The unmatched pair can be filtered out by setting an empirically determined threshold ϵ_g . In experiments we find that binary matching only costs 0.5 *ms* per query on a desktop computer that is very computationally efficient.

6.3.3.2 Intensity Structure Matching

The second stage mainly identifies the intensity similarity between two ISC descriptors Ω^q and Ω^c by column-wise comparison. Let $\mathbf{v}_i^q$ and $\mathbf{v}_i^c$ be the i^{th} column of Ω^q and Ω^c , the score can be found by taking cosine distance

$$\varphi_i(\Omega^q, \Omega^c) = \frac{1}{N_s} \sum_{i=0}^{N_s-1} \left(\frac{\mathbf{v}_i^q \cdot \mathbf{v}_i^c}{\|\mathbf{v}_i^q\| \cdot \|\mathbf{v}_i^c\|} \right). \quad (6.9)$$

Similar to geometry matching, we have to correct viewing angle change. Since the viewing angle change k is already identified from geometry re-identification, we compare Ω_k^q and Ω^c where Ω_k^q is Ω^q shifted by the k^{th} column. This also significantly reduces the computational cost because comparison of intensity involves mathematical operation rather than logical operation. The final score is calculated as

$$\Phi_i(\Omega^q, \Omega^c) = \varphi_i(\Omega_k^q, \Omega^c). \quad (6.10)$$

The unmatched pair can be also filtered out by setting an empirically determined threshold ϵ_i .

6.3.4 Consistency Verification

As discussed in the previous section, the global descriptor is a highly simplified representation of the original point cloud. Hence it is inevitable to have some

features ignored, which can lead to false positive. Therefore it is necessary to do consistency verification before closing the loop.

6.3.4.1 Temporal Consistency Check

In a SLAM system, it is observed that the occurrence of single loop closure often implies high similarity on the neighbour LiDAR scans [140]. We can verify the loop closure by measuring the temporal consistency

$$P(\mathcal{P}_m, \mathcal{P}_n) = \frac{1}{N} \sum_{k=1}^N (\Phi_g(\mathcal{I}_{m-k}, \mathcal{I}_{n-k}) + \Phi_i(\Omega_{m-k}, \Omega_{n-k})), \quad (6.11)$$

where N is the number of frames included for temporal consistency verification. Note that in case of reverse visit (viewing angle change detected in geometry re-identification), $\mathcal{I}_{m-i}$ becomes $\mathcal{I}_{m+i}$ accordingly. The loop candidate can be accepted by taking threshold ξ on the final temporal consistency score.

6.3.4.2 Geometrical Consistency Check

The geometrical consistency verifies the raw scan-to-scan similarity. Similar to [133], Fast Point Feature Histogram (FPFH) features are extracted and matched to get an initial guess of the rigid transform matrix. Starting from this initial estimate, ICP [88] is applied to find the minimal distance error between the query scan and candidate scan. The loop candidate can be accepted by taking threshold on the minimal distance error.

6.4 Experiment Results

In this section, we present evaluation results from the proposed method, including both indoor warehouse robot navigation and outdoor autonomous driving experiments. The proposed method is implemented in C++ and is integrated into the robot operating system (ROS) on an Intel NUC mini computer. For indoor warehouse navigation, an AGV equipped with a Velodyne VLP-16 and an Intel Realsense r200 is used for the experiment. As shown in Figure 6.3, the robot is

developed for warehouse manipulation tasks such as packaging and transportation which require high accuracy on long-term localization. The maximum speed of AGV is 1 m/s . For outdoor autonomous driving, KITTI dataset [18] is used for evaluation. The proposed place recognition approach is used to reduce localization drifts and improve the mapping accuracy. A list of parameters used in the experiment is shown in Table 6.1.

FIGURE 6.3: Autonomous warehouse robot platform used for our experiment.

6.4.1 Experiment on Autonomous Guided Vehicle

The robot is tested in a real warehouse environment consisting of operating machines, shelves, humans, etc. LiDAR odometer is estimated based on ICP via point cloud library (PCL) [141] and the robot trajectory (front-end SLAM) are collected from the fusion of both wheel odometer and LiDAR odometer by Extended Kalman Filter (EKF). To illustrate, we simulate a common task where the robot leaves the docking station to fetch materials and comes back in the reverse direction. Our approach is compared with a vision-based approach via a front-mounted camera. In particular, we use DBoW2 [58] which is implemented in ORB SLAM [111]. The results are shown in Figure 6.4 with the estimated trajectory plotted in green. For DBoW2, the reverse visit changes the view angle significantly so that there is not enough similarity to conclude loop closure. Hence the trajectory collides with shelves according to the final result, which is incorrect. In comparison, the proposed method is able to identify the revisited place. By applying global optimization, the optimized trajectory and the mapping result are much more reasonable. This is due to the view angle invariant property of our proposed two-stage hierarchical ISC-based retrieval.

6.4.2 Evaluation on Public Dataset

To further demonstrate the performance of the proposed method, we test the algorithm on the KITTI dataset which is commonly used for place recognition. KITTI dataset is collected from an autonomous car equipped with various perception systems including front-mounted cameras, Velodyne HDL-64E LiDAR, GPS, etc. The

Parameter	Description	Value
L_{max}	Maximum radius	50
N_s	Number of sectors	20
N_r	Number of rings	60
ϵ_g	Geometry matching threshold	0.9
ϵ_i	Intensity matching threshold	0.92
N	Frames used in temporal consistency check	5
ξ	Temporal consistency checking threshold	1.8

TABLE 6.1: Parameter list.

(A) Loop not detected with visual place recognition.

(B) Loop identified with the proposed approach.

FIGURE 6.4: Local experiment result of the proposed method.

proposed method is tested with multiple sequences including sequence 00, 02 and 05. Sequence 00 and 05 are commonly used for place recognition since most loop closure places are forward visited. Sequence 02 contains both forward and reverse visit and is more challenging for place recognition. Similar to the previous chapters, recall rate and precision are collected from each test. The results of the proposed approach can be found at Figure 6.5, where the GPS trajectory plotted from light

to dark with respect to time. Each of the loop closure place detected is marked with red.

FIGURE 6.5: Loop closure detection result. The left images are camera view, the right images are loop closure detection result. (a) KITTI sequence 00; (b) KITTI sequence 02; (c) KITTI sequence 05.

The results are also compared with existing works such as [82, 83, 139] that are commonly used in SLAM applications. We also include state-of-the-art vision-based place recognition methods such as [58] for comparison. The experiment results of [82, 83, 139] are collected from respective papers due to unavailability of source code and [58] is tested with local experiment. The results are listed in Table 6.2. Compared to vision based approach, our approach achieves competitive precision and recall rate on both sequence 00 and sequence 05. On a more challenging dataset, i.e., KITTI sequence 02, our approach achieves a much higher recall rate.

This is because in sequence 02 vision-based approach fails to identify the reverse visit so that the recall rate drops significantly. However, false positives are reported for LiDAR-based approaches such as our method and [83]. The failure case comes from non-residential areas where both sides of roads are trees so that the geometry and intensity characteristics are very limited for place recognition. Compared to other LiDAR-based approaches such as [82, 83, 139], we achieve both higher recall precision and recall rate across all experiments. In addition, the proposed intensity scan context only costs 1.2 *ms* for each query which is very efficient in practice.

Dataset	Approaches	Precision (%)	Recall Rate (%)
sequence 00	Kim [83]	100	87
	GLAROT3D [82]	86	40
	Cieslewski [139]	92	80
	Gálvez-López [58]	100	92
	<i>Proposed</i>	100	90.2
sequence 02	Kim [83]	90	73
	Gálvez-López [58]	100	80.6
	<i>Proposed</i>	98	91
sequence 05	Kim [83]	100	90
	Cieslewski [139]	93	60
	GLAROT3D [82]	80	80
	Gálvez-López [58]	100	87.6
	<i>Proposed</i>	100	91.2

TABLE 6.2: Comparison with existing methods.

We also calculate the average performance of recall rate and recall precision. The results are summarized in Figure 6.6. Note that, SCI is the scan context image [83].

6.5 Conclusion

In this chapter, we have reviewed existing works on LiDAR-based back-end SLAM systems and presented a robust LCD approach by integrating both geometry and

FIGURE 6.6: Performance comparison between the different methods.

intensity information. Existing works on LiDAR-based LCD mainly leverage the geometric-only descriptor and ignore intensity reading. Inspired by the recent research on LiDAR intensity, we argued that the intensity information can be effective for place recognition and proposed a global 3D descriptor named intensity scan context. To reduce the computational cost, an efficient two-stage hierarchical intensity scan context retrieval was proposed, consisting of fast binary-operation based geometry indexing and intensity structure re-identification. It costs only 1.2 *ms* for each query on a normal computer in practice. Thorough experiments have been conducted including local run tests with an autonomous warehouse robot and public dataset tests to evaluate the proposed method. Compared to the existing VPR methods, the proposed method is able to tackle some challenging scenarios such as reverse visits. Compared to existing state-of-the-art LiDAR-based methods, our proposed method has achieved competitive recall precision and recall rate.

Chapter 7

Intensity-SLAM: Intensity Assisted Localization and Mapping for Large Scale Environment

7.1 Introduction

In the previous chapters, we presented the steps of building a LiDAR-based front-end odometry and both vision-based and LiDAR-based LCD. We also analyzed the feasibility and robustness of various robotic applications such as autonomous driving cars and AGVs. In this chapter, they are integrated together to form a full SLAM solution for autonomous vehicles. In Chapter 6, we have demonstrated that intensity information is very useful for robust loop closure detection. In this chapter, we further explore the intensity information and propose a full SLAM framework that uses both geometry and intensity information for odometry estimation. We firstly analyze the physical model of intensity measurement. Then we introduce an extra intensity cost to the existing geometric-only cost for odometry estimation. Finally, we combine the intensity-based loop closure detection and back-end optimization to further improve the performance. The proposed framework is tested in both an indoor warehouse environment robot and an outdoor autonomous driving car. The results show that our method provides reliable

and accurate localization in multiple environments and outperforms geometric-only methods. The main contributions of this chapter are as below:

- We propose a novel SLAM framework that uses both intensity and geometry information for localization estimation, which involves front-end odometry estimation and back-end factor graph optimization. The proposed method is open sourced¹.
- We propose to calibrate the intensity information via a physical model. Moreover, we minimize both intensity cost and geometry cost to estimate the robot location.
- A thorough evaluation on the proposed method is presented. More specifically, our method is tested on both warehouse and autonomous driving. The results show that our method outperforms the existing geometric-only methods for multiple scenarios.

This chapter is organized as follows: Section 7.2 reviews related works on existing LiDAR SLAM approaches. Section 7.3 describes the details of the proposed approach, including intensity calibration, feature selection, odometry estimation, and ISC-based graph optimization. Section 7.4 shows experimental results and comparison with existing works, followed by the conclusion in Section 7.5.

7.2 Related Works

Many existing works on LiDAR SLAM mainly focus on geometry information of the environment. One of the most popular methods for point cloud matching is ICP which is popularly used in SLAM systems such as HDL-Graph SLAM [104] and LiDAR-Only Odometry and Localization (LOL) [90]. However, all points are used for calculation which is computationally expensive for a laser scan with tens of thousands of points. A more robust and computationally efficient way is to leverage the geometric features. In Chapter 3.1, a simple feature extraction strategy is introduced by analyzing the local smoothness. The feature points are segmented into edge features and planar features based on local smoothness, and the robot

¹https://github.com/wh200720041/intensity_slam

pose is subsequently calculated by matching edge and planar features respectively. However, the intensity channel is ignored and only the geometry channel is used in those methods.

In recent years, some works attempt to introduce the intensity channel into SLAM via deep learning. In [142] the author proposes an end-to-end architecture based on a deep localization network for ground robots. The intensity channel and global intensity map are trained through a CNN which outputs a 3-DoF pose (i.e., x , y position and yaw angle). The network layers are trained by collecting over 4,000 *km* driving data with a variety of urban and highway environments. The experimental results show that the proposed method reduces translational error by 20% compared to the conventional ICP algorithm. The deep-learning-based method adopts end-to-end training without further analysis of intensity information which is straightforward. However, it is often difficult and time consuming to collect, label and train data in practice. Moreover, the performance under environment changes may not be consistent, i.e., the trained network performs well in the environment similar to the training data, but fails when it is transferred to another environment.

Some other works attempt to use the intensity channel and integrate intensity into pose estimation without end-to-end training. Tian *et al* propose an intensity-assisted ICP for fast registration of 2D-LiDAR [143]. Compared to the traditional ICP algorithm, a target function is introduced to determine the initial rigid-body transformation estimation based on the spatial distance and intensity residual. The number of iterations is reduced by 10 times so that the proposed algorithm is able to operate in real time with a single core Central Processing Unit (CPU). In [144], an intensity model is incorporated into the Sparse Bundle Adjustment (SBA) [145] estimation problem. The authors analyze the physical model of intensity measurement and propose a new measurement model that augments 3D localization, intensity and surface normal. Simulation results show that the addition of intensity measurements improves the SLAM accuracy and the same performance can be achieved by fewer landmarks. Unfortunately, it is only validated by simulation and is not evaluated on real robotic applications. Khan *et al* propose a data driven approach to calibrate the intensity information [146]. The scan matching is solved by minimizing the intensity residual of the point pairs rather than the geometry residual. It is subsequently integrated into Hector SLAM [147], which significantly

reduces the drifting error compared to the original Hector SLAM. However, it is limited to 2D localization and mapping in a small scale environment.

7.3 System Overview

The proposed framework of Intensity-SLAM consists of both front-end laser odometry and back-end loop closure detection. The system overview is shown in Figure 7.1. The front-end system includes intensity calibration, feature extraction and odometry estimation, while the back-end system includes loop closure detection and pose graph optimization.

FIGURE 7.1: System overview of the proposed Intensity-SLAM. In the front-end, we analyze both geometry and intensity features to estimate odometry. In the back-end, we use intensity scan context for loop closure detection and global pose graph optimization.

7.3.1 Intensity Calibration

A LiDAR emits laser beams and measures the arrival time as well as the energy of the reflected signals. The object's position in sensor coordinates is determined by emitting angle and distance (i.e., arrival time). The intensity value is determined by the ratio between the energy received from the reflected signals and the laser power emitted. The physical principles of the received power P_r can be determined as [144]

$$P_r = \frac{P_e D_r^2 \rho}{4R^2} \eta_{sys} \eta_{atm} \cos \alpha, \quad (7.1)$$

where P_e is the power of emitted laser beam, D_r is the receiver aperture diameter, η_{sys} is the system transmission factor, η_{atm} is the atmospheric transmission factor, α is the incident angle between the object surface and laser beam, and ρ is the material reflectivity of the object. Specifically, the distance measurement R and the incident angle α are extrinsic parameters, while η_{sys} and η_{atm} are constant parameters. Therefore, the intensity measurement I is determined by [144]

$$\begin{aligned} I &= \frac{P_r}{P_e} = \frac{D_r^2 \rho}{4R^2} \eta_{sys} \eta_{atm} \cos \alpha \\ &= \eta_{all} \frac{\rho \cos \alpha}{R^2}, \end{aligned} \quad (7.2)$$

where η_{all} is a constant. Therefore, the surface reflectivity ρ is only related to the incident angle and measured distance by

$$\rho \propto \frac{IR^2}{\cos \alpha} \quad (7.3)$$

For a LiDAR scan, the distance can be easily measured. Hence the incident angle can be estimated by analyzing the local normal. For each point $\mathbf{p} \in \mathcal{R}^3$ we can search for the nearest two points $\mathbf{p}_1$ and $\mathbf{p}_2$, so that the local surface norm $\mathbf{n}$ can be expressed by

$$\mathbf{n} = \frac{(\mathbf{p} - \mathbf{p}_1)}{\|\mathbf{p} - \mathbf{p}_1\|} \times \frac{(\mathbf{p} - \mathbf{p}_2)}{\|\mathbf{p} - \mathbf{p}_2\|}, \quad (7.4)$$

The incident angle is related by

$$\cos \alpha = \frac{\mathbf{p} \cdot \mathbf{n}}{\|\mathbf{p}\|}. \quad (7.5)$$

Therefore, we can derive the calibrated intensity scan $\tilde{\mathcal{I}}$ from a raw LiDAR scan with geometric reading $\mathcal{P}$ and intensity reading $\mathcal{I}$. Specifically, the intensity is partially calibrated with distance measurement by default in some sensors, hence we only apply the calibration to the incident angle. Moreover, low intensity value often leads to a lower Signal-to-Noise Ratio (SNR), which deteriorates the ranging accuracy for intensity measurement with low value [148]. Points with low intensity are ignored in practice.

7.3.2 Salient Point Selection and Feature Extraction

A LiDAR scan often consists of tens of thousands of points. The raw data contains noisy measurements that can reduce the matching accuracy, e.g., measurements from trees alongside a road in autonomous driving. Hence it is more robust and computationally efficient to match the points clouds with features [49, 50]. Instead of using geometric-only features, in this chapter we explore the feature based on both geometry and intensity information. The calibrated intensity information contains the reflectivity structure of the environment that reveals the distribution of different objects. Therefore, the intensity information also helps to recognize identical features across multiple frames. For each point $\mathbf{p}_i \in \mathcal{P}$ and its intensity value $\tilde{\eta}_i \in \tilde{\mathcal{I}}$, we search for the nearby points $\mathcal{N}_i \in \mathcal{P}$ and calculate the local distance distribution $\sigma_i^{\mathcal{G}}$ and intensity distribution $\sigma_i^{\mathcal{I}}$ by

$$\begin{aligned}\sigma_i^{\mathcal{G}} &= \frac{1}{|\mathcal{N}_i|} \cdot \sum_{p_j \in \mathcal{N}_i} (\|\mathbf{p}_i - \mathbf{p}_j\|), \\ \sigma_i^{\mathcal{I}} &= \frac{1}{|\mathcal{N}_i|} \cdot \sum_{p_j \in \mathcal{N}_i} (\eta_i - \eta_j).\end{aligned}\tag{7.6}$$

The features are calculated based on the weighted local smoothness defined by

$$\sigma_i = w^{\mathcal{G}} \cdot \sigma_i^{\mathcal{G}} + w^{\mathcal{I}} \cdot \sigma_i^{\mathcal{I}},\tag{7.7}$$

where $w^{\mathcal{G}}$ and $w^{\mathcal{I}}$ are the weights of geometry and intensity distribution, respectively. Note that for flat surfaces such as wall, the smoothness values are small; while for corner or edge points, the smoothness values are large. Thus, for each scan, edge features are selected from the points with large σ , and planar features are selected from the points with low σ .

7.3.3 Intensity Map Building

The intensity map $\mathcal{M}$ contains the reflectivity distribution of the surrounding environment. For most geometric-only SLAM, the map is maintained and updated via occupancy grid [149] or Octomap [150]. A 3D space is segmented into grid cells and each cell is represented by a probability function. Similar ideas can be used to construct and update an intensity map. Instead of a probability function, we

use the intensity observation $\mathcal{M}(m_i|z_{1:t})$ to represent each grid cell m_i [146]. More specifically, for each observation of grid cell at time t , the surface reflectivity can be updated by

$$\mathcal{M}(m_i|z_{1:t}) = \mathcal{M}(m_i|z_{1:t-1}) + \frac{\eta_{m_i} - \mathcal{M}(m_i|z_{1:t-1})}{n_{m_i}}, \quad (7.8)$$

where $\mathcal{M}(m_i|z_{1:t})$ is the current intensity observation and n_{m_i} is the total number of observation times on cell m_i . Note that, if the grid is not occupied with an object, the intensity value is marked as 0 since there is no signal reflected.

7.3.4 Scan to Map Matching

The laser odometry is the task to estimate the transformation matrix $\mathbf{T} \in SE(3)$ between the current frame and the global map. The optimal pose estimation is calculated by minimizing both geometry error and intensity error.

7.3.4.1 Geometry Residual

Similar to Chapter 3, the geometric error is calculated by matching the current edge and planar feature with the global map. It can be achieved by minimizing the point-to-edge and point-to-plane residuals. Given an edge feature $\mathbf{p}_\mathcal{E} \in \mathcal{P}_\mathcal{E}$ and the transformed point $\hat{\mathbf{p}}_\mathcal{E} = \mathbf{T} \cdot \mathbf{p}_\mathcal{E}$, we can search for two nearest points $\mathbf{p}_1^\mathcal{E}$ and $\mathbf{p}_2^\mathcal{E}$ from the global map. The point-to-edge residual is defined by

$$f_\mathcal{E}(\hat{\mathbf{p}}_\mathcal{E}) = \frac{(\hat{\mathbf{p}}_\mathcal{E} - \mathbf{p}_1^\mathcal{E}) \times (\hat{\mathbf{p}}_\mathcal{E} - \mathbf{p}_2^\mathcal{E})}{\|\mathbf{p}_1^\mathcal{E} - \mathbf{p}_2^\mathcal{E}\|}. \quad (7.9)$$

Similarly, given a planar feature point $\mathbf{p}_\mathcal{S} \in \mathcal{P}_\mathcal{S}$ and the transformed point $\hat{\mathbf{p}}_\mathcal{S} = \mathbf{T} \cdot \mathbf{p}_\mathcal{S}$, we can search for three nearest points $\mathbf{p}_1^\mathcal{S}$, $\mathbf{p}_2^\mathcal{S}$, and $\mathbf{p}_3^\mathcal{S}$ from the global map. The point-to-plane residual is defined by

$$f_\mathcal{S}(\hat{\mathbf{p}}_\mathcal{S}) = (\hat{\mathbf{p}}_\mathcal{S} - \mathbf{p}_1^\mathcal{S}) \cdot \frac{(\mathbf{p}_1^\mathcal{S} - \mathbf{p}_2^\mathcal{S}) \times (\mathbf{p}_1^\mathcal{S} - \mathbf{p}_3^\mathcal{S})}{\|(\mathbf{p}_1^\mathcal{S} - \mathbf{p}_2^\mathcal{S}) \times (\mathbf{p}_1^\mathcal{S} - \mathbf{p}_3^\mathcal{S})\|}. \quad (7.10)$$

7.3.4.2 Intensity Residual

The intensity residual is calculated by matching the features with the intensity map. It can be achieved by minimizing the intensity residual between the current point $\mathbf{p}_i$ (including both edge features and planar features) and the transformed point $\hat{\mathbf{p}}_i$ in the intensity map

$$f_{\mathcal{I}}(\hat{\mathbf{p}}_i) = \eta_i - \mathcal{M}(\hat{\mathbf{p}}_i), \quad (7.11)$$

where $\mathcal{M}(\hat{\mathbf{p}}_i)$ is the point intensity value in the intensity map $\mathcal{M}$. To search for

FIGURE 7.2: Trilinear interpolation.

the intensity information in the intensity map, we introduce trilinear interpolation. Although it is more straightforward to represent the intensity by the nearest grid cell, the intensity information is less accurate especially for large scale mapping where the grid resolution is low. For each transformed point $\hat{\mathbf{p}}_i = [x_i, y_i, z_i]^T$, we can find the surrounding eight grid cells. As shown in Figure 7.2, the intensity measurements of these grid cells are noted as $\mathcal{M}_1(\hat{\mathbf{p}}_i), \mathcal{M}_2(\hat{\mathbf{p}}_i), \dots, \mathcal{M}_8(\hat{\mathbf{p}}_i)$. Let the center position of cell 1 (closest to the origin) be $\mathbf{p}_1 = [x_1, y_1, z_1]^T$ and the center position of cell 8 (farthest to the origin) be $\mathbf{p}_2 = [x_2, y_2, z_2]^T$, where $x_2 - x_1$, $y_2 - y_1$, and $z_2 - z_1$ are the width, height and depth of each grid cell, respectively.

The intensity estimation of the target point is calculated as

$$\mathcal{M}_{12}(\hat{\mathbf{p}}_i) = \frac{x_i - x_1}{x_2 - x_1} \cdot \mathcal{M}_1(\hat{\mathbf{p}}_i) + \frac{x_i - x_1}{x_2 - x_1} \cdot \mathcal{M}_2(\hat{\mathbf{p}}_i), \quad (7.12)$$

Similarly, we can derive $\mathcal{M}_{34}$, $\mathcal{M}_{56}$ and $\mathcal{M}_{78}$. The intensity value of target point $\hat{\mathbf{p}}_i$ is estimated as

$$\begin{aligned} \mathcal{M}_9(\hat{\mathbf{p}}_i) &= \frac{y_i - y_1}{y_2 - y_1} \cdot \mathcal{M}_{12}(\hat{\mathbf{p}}_i) + \frac{y_i - y_1}{y_2 - y_1} \cdot \mathcal{M}_{34}(\hat{\mathbf{p}}_i) \\ \mathcal{M}_{10}(\hat{\mathbf{p}}_i) &= \frac{y_i - y_1}{y_2 - y_1} \cdot \mathcal{M}_{56}(\hat{\mathbf{p}}_i) + \frac{y_i - y_1}{y_2 - y_1} \cdot \mathcal{M}_{78}(\hat{\mathbf{p}}_i) \\ \mathcal{M}(\hat{\mathbf{p}}_i) &= \frac{z_i - z_1}{z_2 - z_1} \cdot \mathcal{M}_9(\hat{\mathbf{p}}_i) + \frac{z_i - z_1}{z_2 - z_1} \cdot \mathcal{M}_{10}(\hat{\mathbf{p}}_i). \end{aligned}$$

7.3.4.3 Pose Estimation

The final pose can be estimated by minimizing both geometry residual and intensity residual

$$\arg \min_{\mathbf{T}} \sum_{\mathbf{p}_i \in \mathcal{P}_{\mathcal{E}}} |f_{\mathcal{E}}(\hat{\mathbf{p}}_i)| + \sum_{\mathbf{p}_i \in \mathcal{P}_{\mathcal{S}}} |f_{\mathcal{S}}(\hat{\mathbf{p}}_i)| + \sum_{\mathbf{p}_i \in \mathcal{P}} |f_{\mathcal{I}}(\hat{\mathbf{p}}_i)|. \quad (7.13)$$

It can be solved by the Levenberg–Marquardt algorithm [151]. Note that the initial pose alignment is estimated by assuming constant angular velocity and linear velocity which can increase convergence speed compared to assuming an identical transformation matrix.

7.3.5 Loop Closure Detection and Global Optimization

In the previous chapters, we have demonstrated the use of LCD in a SLAM system. To reduce the computational cost, we use keyframe selection to reduce the retrieval time since the computational cost of loop closure detection is often high. The keyframes are selected based on the following criteria: (1) the displacement of the robot is significant, i.e., greater than a pre-defined threshold; (2) the change of rotation angles, including roll, pitch, and yaw is large; (3) the time elapsed is more than a certain period.

For each keyframe, we use our previous work on ISC descriptor to extract the current frame into a global signature. Compared to geometric-only descriptors such as GLAROT3D [82], NBLD [139] and Scan Context [83], the ISC is robust

Algorithm 7.1: Laser Odometry**Input** : Current scan $\mathcal{P}_k$, $\mathcal{I}_k$, robot's trajectory $T_{1:k-1}$ **Output:** Current pose T_k and intensity map**begin** **if** *Not Initialized* **then** $T_0 \leftarrow \mathbf{0}$; **end** Calculate calibrated intensity scan $\tilde{\mathcal{I}}_k$; Compute initial alignment $T_k^0 \leftarrow T_{k-1} \cdot T_{k-2}^{-1} \cdot T_{k-1}$;

Feature extraction;

for *a number of iterations* i **do** **for** *each feature point* $\mathbf{p}_i \in \mathcal{P}_k$ **do** Transform into global coordinate $\hat{\mathbf{p}}_i \leftarrow T_k^{i-1} \cdot \mathbf{p}_i$; Compute intensity residual $f_{\mathcal{I}}(\hat{\mathbf{p}}_i)$; **if** $\mathbf{p}_i$ *is an edge feature* **then** Compute edge residual $f_{\mathcal{E}}(\hat{\mathbf{p}}_i)$ **end** **if** $\mathbf{p}_i$ *is a planar feature* **then** Compute planar residual $f_{\mathcal{S}}(\hat{\mathbf{p}}_i)$; **end** **end**

Applying Levenberg-Marquard method;

if *nonlinear optimization converges* **then** Compute T_k^i ; **end** **end**

Update intensity map;

end

to rotation change in identifying a loop closure. Given a keyframe, we can extract the ISC descriptor Ω from both intensity and geometry information. To compare a query ISC descriptor Ω^q with a candidate ISC descriptor Ω^c , let $\mathbf{v}_i^q$ and $\mathbf{v}_i^c$ be the i^{th} column of Ω^q and Ω^c . The similarity score $\varphi(\Omega^q, \Omega^c)$ is found by taking cosine distance

$$\varphi(\Omega^q, \Omega^c) = \frac{1}{N_s} \sum_{i=0}^{N_s-1} \left(\frac{\mathbf{v}_i^{qT} \cdot \mathbf{v}_i^c}{\|\mathbf{v}_i^q\| \cdot \|\mathbf{v}_i^c\|} \right). \quad (7.14)$$

The loop closure can be determined by setting a threshold on $\varphi(\Omega^q, \Omega^c)$. Moreover, to prevent false positives, geometry consistency verification is used to check the similarity of candidate frames. With the revisited place identified, we can add the edge between two frames into the pose graph and global optimization can be applied to correct the drifts [54].

7.4 Experiment Results

7.4.1 Experiment Setup

In this section, we present extensive experiments of the proposed Intensity-SLAM, including evaluation on public dataset and practical experiments on a real robot platform. For public dataset test, the proposed method is evaluated on the KITTI dataset [100] that is commonly used for SLAM evaluation. In the practical test, the proposed method is integrated into an AGV in a warehouse environment. The algorithm is implemented in C++ on ROS Melodic and Linux Ubuntu 18.04. In order to measure the computational cost of different approaches, all algorithms are tested on a desktop computer equipped with an Intel i7 CPU.

7.4.2 Evaluation on Public Dataset

Similar to Chapter 3, we evaluate the proposed method on the KITTI dataset. In total we test over more than 10,000 frames with more than 10 *km* travelling distance. Some of the experiment results are shown in Figure 7.3, in which the sequences 00, 02, 05, 07, and 09 are selected for demonstration. The trajectory of the proposed method is in green and the ground truth is in red. It can be seen that our method is able to track the robot pose accurately.

We also evaluate the performance of the proposed full SLAM system with the front-end LO on the KITTI dataset. We use FLOAM which is proposed in Chapter 3 for comparison. The results are shown in Figure 7.4. It can be seen that the trajectory estimated from intensity-SLAM is closer to the ground truth and the results are greatly improved compared to FLOAM.

7.4.3 Ablation Study of Intensity Information

We also demonstrate the effect of intensity information in localization. To be consistent, the loop closure detection is removed and only front-end laser odometry is compared. In the experiment, the “geometric-only laser odometry” method only includes point-to-edge and point-to-plane residual, while “odometry with intensity”

FIGURE 7.3: Result of the proposed method on the KITTI dataset. In the first row from left to right are sequences 00, 02, in the second row are sequences 05, 07, and 09. The result of intensity-SLAM is plotted in green color and the ground truth is plotted in red color.

method integrates intensity residual into the geometric-only laser odometry. All other parameters are kept unchanged. Moreover, we also include the result of ICP for comparison. The results are shown in Figure 7.5 and Table 7.1. In particular, we use KITTI sequence 05 for illustration. It can be seen that by adding the intensity residual, the overall ATE is improved from 0.69% to 0.62% and the overall ARE is improved from 5.7×10^{-5} deg / m to 5.3×10^{-5} deg / m . Compared to the geometric-only method, the intensity information improves the translational and rotational accuracy by around 10%, while it does not increase the computing time too much.

7.4.4 Comparison with Existing Methods

We also compare our approach with state-of-the-art methods including LeGO-LOAM [50], A-LOAM, and HDL-GRAPH-SLAM [152]. LeGO-LOAM, A-LOAM and HDL-GRAPH-SLAM are open-sourced and we directly use the source code and

FIGURE 7.4: Comparison of FLOAM and intensity-SLAM. The ground truth is plotted in red and the estimated trajectory is plotted in green. (a) and (b) are the results of FLOAM on KITTI Sequences 00 and 05. (c) and (d) are the results of Intensity-SLAM on KITTI Sequences 00 and 05.

test the algorithms on the local computer. To evaluate the performance, all methods are implemented on the same computer with ROS installed. All approaches are tested using the same settings and the results are shown in Table 7.2. It can be seen from the table that our method is competitive in terms of computational cost. And it is worth noting that our method achieves the best accuracy and outperforms existing geometric-only methods.

FIGURE 7.5: Performance comparison on KITTI sequence 05.

Method	Time (<i>ms</i>)	ATE (%)	ARE (<i>deg/m</i>)
ICP	142	1.031	0.0092
Odometry(geometric-only)	92	0.691	0.0057
Odometry(with intensity)	113	0.627	0.0053
Intensity-SLAM	115	0.529	0.0044

TABLE 7.1: Result analysis of intensity-aided method and non-intensity-aided method on the KITTI dataset.

Method	Time (<i>ms</i>)	ATE (%)	ARE (<i>deg/m</i>)
Intensity-SLAM	115	0.529	0.0044
LeGO-LOAM	84	0.724	0.0072
HDL-Graph-SLAM	346	1.127	0.0098
A-LOAM	182	0.702	0.0058

TABLE 7.2: Performance comparison of different approaches on the KITTI dataset.

FIGURE 7.6: Localization and mapping result in a warehouse environment. (a) AGV platform for warehouse manipulation; (b) localization and mapping result of the proposed method. The robot's trajectory is plotted in red color.

7.4.5 Validation on Practical Robots

In this section, we test the performance of our method on a practical robot in a warehouse environment. To this end, we implement our algorithm on an industrial AGV that is used for grabbing and transporting materials. As shown in Figure 7.6 (a), the AGV is equipped with an Intel NUC mini computer and a Velodyne VLP-16 LiDAR for perception and localization. In this experiment, the robot uses Intensity-SLAM for localization and it is supposed to fetch the item from the target storage shelf and return it to another place. The localization and mapping result is shown in Figure 7.6 (b), with the robot trajectory plotted in red. It can be seen that our method is able to generate reliable localization and a dense 3D map for a complex indoor environment. In particular, the robot is controlled to return to the same location and we record the final localization drift. The final localization drift is about 5 *cm*, which is accurate enough for many indoor robotic applications.

7.5 Conclusion

In this chapter, we have presented a novel full SLAM framework for LiDAR-based localization systems, which is an integration of both front-end and back-end SLAM systems that were proposed in the previous chapters. Furthermore, we introduced intensity features for both odometry estimation and global optimization to seek improvement. We began with the physical model of intensity measurement and illustrated the use of intensity information. We introduced the intensity map and intensity residual for pose estimation to improve the localization accuracy. Moreover, we have proposed a full SLAM structure that consists of both front-end and back-end systems, namely Intensity-SLAM. Thorough experiments were performed, including indoor AGV mapping in the warehouse and outdoor autonomous driving in urban areas. The experiment results showed that by integrating intensity information, the localization accuracy is significantly improved. The proposed method has achieved lower translational error and rotational error compared to the state-of-the-art geometric-only methods.

Chapter 8

Lightweight 3D Localization and Mapping for Solid-State LiDAR

8.1 Introduction

In the previous chapter, we have designed a full SLAM solution for autonomous vehicles. It mainly targets to provide solutions based on conventional mechanical LiDAR. Recently, the introduction of solid-state LiDAR provides a cost-effective and lightweight solution for LiDAR SLAM systems. The solid-state LiDAR is a system that is built entirely on a silicon chip with no moving parts involved [148]. Hence both size and weight can be made much smaller than mechanical LiDAR. Moreover, the solid-state LiDAR is resistant to vibrations by removing the rotating mechanical structure. The latest solid-state LiDAR only costs 10% price of a mechanical LiDAR and is as small as a smartphone, which has a great potential to become the dominant perception system for small scale application such as VR [153], drone inspection, and indoor navigation. In terms of performance, solid-state LiDAR is able to achieve accuracy up to 1-2 *cm* and a detectable range up to a few hundreds of meters. The difference between a mechanical LiDAR and a solid-state LiDAR is shown in Figure 8.1.

Although the principle of the mechanical LiDAR and the solid-state LiDAR is similar, the implementation of LiDAR SLAM is different. To illustrate the difference between two LiDARs, we use Velodyne VLP-16 and Realsense L515 for example. The specifications can be found at Table 8.1. The solid-state LiDAR has higher

FIGURE 8.1: Comparison of mechanical LiDAR and Solid-state LiDAR¹.

angular resolution, implying that the points are denser within the same scanning area. Therefore, traditional LiDAR odometry methods such as ICP [88] can be computationally inefficient since there are more points involved for calculation. Secondly, the update frequency of solid-state LiDAR is higher, while traditional LiDAR SLAM such as LOAM [49] is not computationally efficient enough to reach real-time performance. Another challenge is the pyramid-like coverage view that can cause severe tracking loss during large rotation. Therefore, a more computationally efficient and robust algorithm has to be designed in order to achieve real-time performance for solid-state LiDAR-based SLAM.

In this chapter, we propose a novel and lightweight SLAM framework for solid-state LiDAR, consisting of feature extraction, odometry estimation and probability map construction. Inspired by the works proposed in the previous chapters, we propose a new rotation-invariant feature extraction method that explores both horizontal and vertical curvature. The proposed method provides real-time localization up to 30 *Hz*. Thorough experiments have been conducted to evaluate its performance. The main contributions of this chapter are as follows:

¹<https://www.youtube.com/watch?v=Lv0jKIZ5aIk>

Tech Specs	Velodyne VLP-16	Realsense L515
Type	Mechanical	Solid-state
Frequency	5-20Hz	30Hz
FoV	$360^\circ \times 32^\circ$	$70^\circ \times 55^\circ$
Horizontal Resolution	$0.1\text{-}0.4^\circ$	0.07°
Vertical Resolution	2.0°	0.07°
Detection Range	0.5-100 m	0.25-9 m
Accuracy	3 cm	1.4 cm
Dimension	103×72 mm	61×26
Weight	860 g	95 g

TABLE 8.1: Difference of mechanical and solid-state LiDAR.

- We propose a full SLAM framework for solid-state LiDAR, which targets to tackle perception system with small FoV and high updating frequency. We make the proposed method open sourced ².
- We introduce an improved feature extraction strategy that is able to search for consistent features under significant rotations. Moreover, left Lie derivative is used for iterative pose estimation so that the pose is stored in a singularity-free format.
- A thorough evaluation of the proposed method is presented. More specifically, we integrate an Intel L515 solid state LiDAR to AGVs and test the proposed method in a complex warehouse environment. The proposed method can provide real time localization and is robust under large rotations.

This chapter is organized as follows: Section 8.2 reviews the related works on existing LiDAR SLAM approaches. Section 8.3 describes the details of the proposed approach, including feature point selection, laser odometry estimation, and probability map construction. Section 8.4 shows experimental results and comparison with existing works, followed by the conclusion in Section 8.5.

²https://github.com/wh200720041/ss1_slam

8.2 Related Works

As discussed previously, the solid-state LiDAR has higher angular resolution. Traditional LiDAR odometry methods such as ICP [88] can be computationally inefficient since there are more points involved for calculation. Secondly, the update frequency of solid-state LiDAR is higher, while traditional LiDAR SLAM such as LOAM [49] is not computationally efficient enough to reach real-time performance. Moreover, the pyramid-like coverage view can cause severe tracking loss during large rotation. Therefore, a more computationally efficient and robust algorithm has to be designed in order to achieve real-time performance for solid-state LiDAR-based SLAM. Recently some works aim to use some specially designed LiDAR with small FoV. In [148] the author proposes a new LiDAR SLAM framework for small FoV LiDAR, i.e., the DJI Livox. DJI Livox spins a laser beam in cone shape direction and provides smaller FoV but denser points. To tackle the localization problem for small FoV sensors, the author introduces a new feature extraction method with the analysis of local LiDAR intensity. Compared to traditional feature extraction, the proposed feature extraction is more consistent over adjacent frames which can subsequently reduce the tracking loss. However, it is mainly designed for a large scale outdoor environment, while the performance in the indoor complex environment is not demonstrated.

8.3 System Overview

In this section, the proposed method is described in detail. As shown Figure 8.2, the system consists of three main modules, namely feature extraction, odometry estimation and probability map construction. We firstly present a rotation invariant feature extraction method and then introduce odometry estimation via local feature matching. Finally we present the probability map building and scene reconstruction.

8.3.1 Feature Extraction

A solid-state LiDAR integrates all sensors on a single silicon chip without moving parts. It often has higher resolution and higher update frequency compared to a

FIGURE 8.2: System overview of the proposed method. It consists of three main modules: feature extraction, odometry estimation and probability map construction, which are highlighted in yellow, blue and red color respectively.

mechanical LiDAR. Hence it might be too computationally heavy to match the raw point clouds. Inspired by the works proposed in previous chapters, we leverage the edge and planar matching which is much more computationally efficient. Before processing the data, we remove the noisy points based on the measured distance. It is observed that readings near the maximum detection range are often less accurate due to the low reflected energy and thus we pre-filter these noisy points.

The point cloud from a LiDAR scan is unordered. To compute the edge and planar features, we firstly project the point cloud into a 2D point matrix. For the k^{th} LiDAR scan input $\mathcal{P}_k$, it is segmented based on the vertical angle and horizontal angle of each point. Given a point $\mathbf{p}_i = \{x_i, y_i, z_i\} \in \mathcal{P}_k$, the vertical angle α_i and horizontal angle θ_i are calculated by

$$\begin{aligned}\alpha_i &= \arctan \frac{y_i}{x_i}, \\ \theta_i &= \arctan \frac{z_i}{x_i}.\end{aligned}\tag{8.1}$$

The point cloud is then segmented by equally dividing vertical detection range $[\alpha_{min}, \alpha_{max}]$ and horizontal detection range $[\theta_{min}, \theta_{max}]$ into M sectors and N sectors, where α_{min} , α_{max} , θ_{min} and θ_{max} are the minimum vertical angle, maximum vertical angle, minimum horizontal angle, maximum horizontal angle according to the sensor specifications. Hence, the point cloud is segmented into $M \times N$ cells. For a solid-state LiDAR with vertical angular resolution of α_r and horizontal resolution of θ_r , M and N are selected as half of total points in a single direction, i.e., $M = \frac{\alpha_{max} - \alpha_{min}}{2 \times \alpha_r}$ and $N = \frac{\theta_{max} - \theta_{min}}{2 \times \theta_r}$. In general, larger M and N will take more computational resources. In each cell (m, n) , where $m \in [1, M]$, $n \in [1, N]$,

we calculate the mean measurement $\mathbf{p}_k^{(m,n)}$ by finding the geometry center of all points in the cell. To extract the edge and planar features, we search for its nearby points and define the local smoothness by

$$\sigma_k^{(m,n)} = \frac{1}{\lambda^2} \cdot \sum_{\mathbf{p}_k^{(i,j)} \in \mathcal{S}_k^{(m,n)}} (||\mathbf{p}_k^{(i,j)}|| - ||\mathbf{p}_k^{(m,n)}||), \quad (8.2)$$

with

$$\mathcal{S}_k^{(m,n)} = \{\mathbf{p}_k^{(i,j)} | i \in [m - \lambda, m + \lambda], j \in [n - \lambda, n + \lambda]\}, \quad (8.3)$$

where λ is a pre-defined searching radius. A larger λ means searching for more surrounding points and requires more computational resources. The local smoothness indicates the sharpness of surrounding information. A higher $\sigma_k^{(m,n)}$ indicates that the local surface is sharp, thus the corresponding point $\mathbf{p}_k^{(m,n)}$ is taken as edge feature. A smaller $\sigma_k^{(m,n)}$ indicates that the local surface is flat, thus the corresponding point $\mathbf{p}_k^{(m,n)}$ is taken as a plane feature.

8.3.2 Odometry Estimation

The odometry estimation is a task to estimate the robot's current pose $\mathbf{T}_k \in SE(3)$ in global coordinate based on the historical laser scans $\mathcal{P}_1, \mathcal{P}_2, \dots, \mathcal{P}_{k-1}$. To reduce the computational cost, a sliding window approach is used. The edge features and planar features from the nearby frames are used to build the local feature maps. For the current input P_k , we define the local map $M_k = \{P_{k-1}, P_{k-2}, \dots, P_{k-q}\}$, where q is the number of frames used to build a local map.

As mentioned above, matching on raw point clouds is less efficient and sensitive to noise. Thus, we leverage the matching of edge points and planar points in feature space. Similar to Chapter 7, for each edge point $\mathbf{p}_\mathcal{E} \in P_k$ and its transform in the global coordinate $\hat{\mathbf{p}}_\mathcal{E} = \mathbf{T}_k \cdot \mathbf{p}_\mathcal{E}$, we search for the nearest edge from the local map. Note that the local map is divided into edge map and planar map, and each map is built by K-D tree in order to increase searching efficiency. We can select two nearest edge feature points $\mathbf{p}_1^\mathcal{E}$ and $\mathbf{p}_2^\mathcal{E}$ from the local map for each edge point. The point-to-edge residual $f_\mathcal{E}(\hat{\mathbf{p}}_\mathcal{E})$ is defined as the distance between $\hat{\mathbf{p}}_\mathcal{E}$ and the edge crossing $\mathbf{p}_1^\mathcal{E}$ and $\mathbf{p}_2^\mathcal{E}$

$$f_\mathcal{E}(\hat{\mathbf{p}}_k) = \frac{(\hat{\mathbf{p}}_k - \mathbf{p}_1^\mathcal{E}) \times (\hat{\mathbf{p}}_k - \mathbf{p}_2^\mathcal{E})}{\|\mathbf{p}_1^\mathcal{E} - \mathbf{p}_2^\mathcal{E}\|}. \quad (8.4)$$

Note that if the number of nearby points is less than 2, the point-to-edge residual is not taken into account by the final cost. Similarly, for each planar point $\mathbf{p}_k \in P_k$, we search for the nearest planar features from the local map. To estimate a plane in 3D space, it is necessary to have 3 points. Hence for a given planar feature point $\mathbf{p}_S$ and its transform in the global coordinate $\hat{\mathbf{p}}_S = \mathbf{T}_k \cdot \mathbf{p}_S$, we can find 3 nearest points $\mathbf{p}_S^1$, $\mathbf{p}_S^2$, and $\mathbf{p}_S^3$ from the local map. The point-to-plane residual $f_S(\hat{\mathbf{p}}_k)$ is defined as the distance between $\hat{\mathbf{p}}_k$ and the plane crossing $\mathbf{p}_S^1$, $\mathbf{p}_S^2$, and $\mathbf{p}_S^3$

$$f_S(\hat{\mathbf{p}}_S) = (\hat{\mathbf{p}}_S - \mathbf{p}_S^1) \cdot \frac{(\mathbf{p}_S^1 - \mathbf{p}_S^2) \times (\mathbf{p}_S^1 - \mathbf{p}_S^3)}{\|(\mathbf{p}_S^1 - \mathbf{p}_S^2) \times (\mathbf{p}_S^1 - \mathbf{p}_S^3)\|}. \quad (8.5)$$

Similar to the edge residual, the point-to-plane residual is not taken into account when the number of nearby points is less than 3. The final odometry is estimated by minimizing the point-to-plane residual and point-to-edge residual

$$\arg \min_{\mathbf{T}_k} \sum |f_E(\hat{\mathbf{p}}_k)| + \sum |f_S(\hat{\mathbf{p}}_k)|. \quad (8.6)$$

This nonlinear optimization problem can be solved by the Gauss–Newton optimization method. In LOAM, the Jacobian matrix is calculated using a differential model of BCH formula, where the increment is directly defined in the Lie Algebra [49]. However, it is computationally expensive to compute the derivative. Instead, we use the left perturbation scheme and apply increment on the Lie Group from Chapter 3. Define $\xi_k = [\rho, \phi] \in \mathfrak{se}(3)$ and the transformation matrix

$$\mathbf{T}_k = \exp(\xi_k^\wedge), \quad (8.7)$$

where the notation $\xi^\wedge$ converts a 6D vector into a 4×4 matrix by

$$\xi^\wedge = \begin{bmatrix} [\rho]_\times & \phi \\ \mathbf{0}_{1 \times 3} & 0 \end{bmatrix}. \quad (8.8)$$

The left perturbation model can be calculated by

$$\begin{aligned} \mathbf{J}_p &= \frac{\partial \mathbf{T}_k \mathbf{p}_k}{\partial \delta \xi} = \lim_{\delta \xi \rightarrow \mathbf{0}} \frac{(\exp(\delta \xi^\wedge) \cdot \mathbf{T}_k \mathbf{p}_k - \mathbf{T}_k \mathbf{p}_k)}{\delta \xi} \\ &= \begin{bmatrix} \mathbf{I}_{3 \times 3} & -[\mathbf{T}_k \mathbf{p}_k]_\times \\ \mathbf{0}_{1 \times 3} & \mathbf{0}_{1 \times 3} \end{bmatrix}. \end{aligned} \quad (8.9)$$

Note that here $[\mathbf{T}_k \mathbf{p}_k]_\times$ transforms 4D homogeneous point expression $\{x, y, z, 1\}$

into 3D point expression $\{x, y, z\}$ before calculating the skew matrix. The Jacobian matrix of point-to-edge residual is defined by

$$\begin{aligned} \mathbf{J}_\mathcal{E} &= \frac{\partial f_\mathcal{E}(\hat{\mathbf{p}}_k)}{\partial \hat{\mathbf{p}}_k} \cdot \mathbf{J}_p \\ &= \frac{(\mathbf{p}_1^\mathcal{E} - \mathbf{p}_2^\mathcal{E})}{\|\mathbf{p}_1^\mathcal{E} - \mathbf{p}_2^\mathcal{E}\|} \times \mathbf{J}_p. \end{aligned} \quad (8.10)$$

Similarly, we can derive the Jacobian matrix of point-to-plane residual

$$\begin{aligned} \mathbf{J}_\mathcal{S} &= \frac{\partial f_\mathcal{S}(\hat{\mathbf{p}}_k)}{\partial \hat{\mathbf{p}}_k} \cdot \mathbf{J}_p \\ &= \frac{[(\mathbf{p}_1^\mathcal{S} - \mathbf{p}_2^\mathcal{S}) \times (\mathbf{p}_1^\mathcal{S} - \mathbf{p}_3^\mathcal{S})]}{\|(\mathbf{p}_1^\mathcal{S} - \mathbf{p}_2^\mathcal{S}) \times (\mathbf{p}_1^\mathcal{S} - \mathbf{p}_3^\mathcal{S})\|} \cdot \mathbf{J}_p. \end{aligned} \quad (8.11)$$

The alignment of the current scan and the local map may not be ideal at the beginning. Therefore, it is necessary to search for better feature correspondences. The optimal feature correspondence can be found through an iterative calculation, i.e. with an initial pose $\mathbf{T}_k^0$ and initial correspondence based on $\mathbf{T}_k^0$, we can derive the odometry estimation $\mathbf{T}_k^1$ then $\mathbf{T}_k^2$ and $\mathbf{T}_k^3$, etc. This finally converges to the optimal estimation of current pose. Although iterative calculation is computationally inefficient, a good estimation of initial pose alignment is able to speed up the convergence. To find a better initial alignment, we assume constant angular velocity and linear translation

$$\begin{aligned} \mathbf{T}_k^0 &= \mathbf{T}_{k-1} \cdot \mathbf{T}_{k-2}^{k-1} \\ &= \mathbf{T}_{k-1} \cdot \mathbf{T}_{k-2}^{-1} \cdot \mathbf{T}_{k-1}. \end{aligned} \quad (8.12)$$

The process of iterative odometry estimation is listed in Algorithm 8.1.

8.3.3 Probability Map Construction

The global map is often of large size and it is computationally inefficient to update it with each frame. Therefore, we only use keyframes to update and reconstruct maps similar to Chapter 3. To increase the searching efficiency, an octotree is used to construct the global map. It only takes computational complexity of $\mathcal{O}(\log n)$ to search for a specific node from an octotree of depth n , which can significantly reduce mapping cost [98]. Recall that for each cell in octotree, we use probability

Algorithm 8.1: Pose estimation for Solid-state LiDAR

Input : Current scan $\mathcal{P}_k$, robot's trajectory $\mathbf{T}_{1:k-1}$
Output: Current pose $\mathbf{T}_k$
begin
 if *Not Initialized* **then** $\mathbf{T}_0 \leftarrow \mathbf{0}$;
 Divide the current scan into cells;
 Calculate the geometry center point $\mathbf{p}_k^{(m,n)}$ of each cell;
 Apply feature extraction from geometry center points;
 Compute initial alignment $\mathbf{T}_k^0 \leftarrow \mathbf{T}_{k-1} \mathbf{T}_{k-2}^{-1} \mathbf{T}_{k-1}$;
 for *Pose optimization is not converged* **do**
 for *each point* $\mathbf{p}_k \in \mathcal{P}_k$ **do**
 Transform to map coordinate $\hat{\mathbf{p}}_k \leftarrow \mathbf{T}_k^{i-1} \mathbf{p}_k$;
 if $\mathbf{p}_k$ *is an edge feature* **then**
 Compute edge residual $f_{\mathcal{E}}$ and Jacobian matrix $J_{\mathcal{E}}$;
 end
 if $\mathbf{p}_k$ *is a planar feature* **then**
 compute planar residual $f_{\mathcal{S}}$ and Jacobian matrix $J_{\mathcal{S}}$;
 end
 end
 if *nonlinear optimization converges* **then**
 Compute $\mathbf{T}_k^i$;
 end
 end
 Update current scan into local map and remove the oldest scan from the local map;
 if *the current scan is a key frame* **then**
 Align current scan to global coordinate;
 Update probability $P(n|z_{1:t})$;
 end
 Return current pose $\mathbf{T}_k$;
end

representation [99] $P(n|z_{1:t})$ to indicate the existence of an object

$$P(n|z_{1:t}) = \left[1 + \frac{1 - P(n|z_t)}{P(n|z_t)} \cdot \frac{1 - P(n|z_{1:t-1})}{P(n|z_{1:t-1})} \cdot \frac{P(n)}{1 - P(n)} \right]^{-1}, \quad (8.13)$$

where z_t is the current measurement, $z_{1:t-1}$ is the historical measurement from keyframes. $P(n)$ is the prior probability and it is set to 0.5 for an unknown area. Therefore, the probability of each cell can be updated based on new measurements.

8.4 Experiment Results

8.4.1 Experiment Setup

In this section, we present experiment results of the proposed method. Our method is firstly tested on an AGV that is used for warehouse logistics. Then we analyze the performance of the proposed method and compare it with the existing LiDAR SLAM. To further illustrate the robustness, the proposed method is integrated into a hand-held device used for 3D scanning. In our experiment, an Intel Realsense L515 is used for demonstration. It is a small FoV solid-state LiDAR with $70^\circ \times 55^\circ$ viewing angle and 30 *Hz* update frequency. It is smaller and lighter than a smartphone so that it can be used on many mobile robotic platforms. The algorithm is coded in C++ and is implemented on an Intel NUC i7 mini computer with Ubuntu 18.04 and ROS Melodic [154]. For performance comparison, we use A-LOAM and LOAM. Two methods are open-sourced and we implement the methods on the same computer. All methods are coded in C++ and tested on ROS.

8.4.2 Performance Evaluation and Comparison

To evaluate the localization results, our method is compared with the ground truth, which is provided by a VICON system. The AGV is manually controlled to move around the VICON-covered room of size $4\text{m} \times 4\text{m}$ as shown in Figure 8.3. The result is shown in Figure 8.4, where the trajectories of the ground truth and our method are plotted in red and green, respectively. Our method achieves a localization error of 5 *cm*.

We also compared our method with LOAM [49] that is widely used for LiDAR SLAM. To be consistent, both algorithms are tested on an intel NUC mini PC and we record the average computing time per frame. Our method achieves an average processing speed of 31 *ms*, while LOAM requires 74 *ms* to process each frame. The result of LOAM is plotted in orange. It is obvious that tracking loss happens to LOAM when there is large rotation, while our method is still able to track accurately.

FIGURE 8.3: Performance evaluation with ground truth. (a) Pioneer robot equipped with a solid-state LiDAR; (b) testing field with VICON system.

FIGURE 8.4: Comparison of the proposed method, LOAM and ground truth. LOAM loses track during large rotation while the proposed method is able to track accurately. And our approach achieves 5 *cm* localization accuracy.

8.4.3 Performance on Warehouse Robot

The algorithm is then evaluated on an AGV running in a warehouse environment. Smart manufacturing is one of the main applications for SLAM. In an advanced factory, the robot is supposed to transport, process and assemble products automatically. This requires the robot to efficiently localize itself in complex and highly dynamic environments with moving operators and other robots. In this experiment, the proposed method is integrated into an industrial AGV that is shown in Figure 8.5 (a). The robot is used for gripping and transporting the materials with a robot arm on the top. A solid-state LiDAR is mounted in the front to provide localization. In this task, the AGV platform automatically explores the warehouse and reconstructs the environment at the maximum speed of 1 m/s . The mapping result is shown in Figure 8.5 (b). It can be seen that our method is able to provide dense mapping in complex environments. We compare the 3D mapping results with the images. The image view of the original warehouse is shown in Figure 8.6 (a), (b) and the built 3D map is shown in Figure 8.6 (c), (d). It can be seen that our approach is also able to build high resolution 3D maps that are close to the real environment.

FIGURE 8.5: An example of indoor localization and mapping in a warehouse environment. (a) the AGV platform used for warehouse manipulation with a solid-state LiDAR mounted in the front; (b) reconstructed map.

FIGURE 8.6: Map quality evaluation. The reconstructed map is shown in the middle. We randomly picked two places for illustration. (a) and (b) are the original camera view; (c) and (d) are reconstructed scenes based on the proposed approach.

8.4.4 Performance on Hand-held Device

8.4.4.1 3D Mapping

To further demonstrate the robustness, the proposed method is also evaluated on a hand-held device. With the growing VR, AR, and gaming industry, SLAM has been implemented on various mobile devices such as smartphones and VR glasses. However, most mobile platforms have limited computational resources. Compared to the implementation on warehouse robots, the hand-held device suffers from vibration and large viewing angle change which can cause tracking loss and localization failure. Our system is built as a mobile scanner which is shown in Figure 8.7 (a). The hand-held scanner consists of three modules: perception module, rotation platform, and computing module. It is less than 500 *g* and can be implemented on many applications such as drone navigation. In the experiment, we hold the 3D scanner and scan the indoor environment at normal walking speed. The localization and mapping results are shown in Figure 8.7 (b), with the trajectory plotted in green. Our method can accurately localize itself and perform mapping in real time.

8.4.4.2 Rotation Test

The hand-held device often has higher rotation changes which can result in tracking loss. In order to demonstrate the performance of the proposed method under large rotation, we put the solid-state LiDAR in horizontal orientation and randomly rotate the solid-state LiDAR with maximum rotation speed of 1.57 rad/s . The solid-state LiDAR returns to horizontal orientation and we record the angle deviation. Tracking loss is considered when the final angle deviation is greater than 10 degrees. We compare our method with A-LOAM and the results are shown in Table 8.2. It can be seen that our method has a higher success rate compared to A-LOAM.

Method	Success	Tracking loss
The proposed method	6	0
A-LOAM	1	5

TABLE 8.2: Results of rotation test.

8.5 Conclusion

In this chapter, we have presented a SLAM system based on the recently introduced solid-state LiDAR. Solid-state LiDAR is an emerging LiDAR system with higher update frequency and smaller FoV compared to traditional mechanical LiDAR. We made some improvements on our SLAM system that was proposed in the previous chapter. To tackle the tracking loss problem, we introduced an improved feature extraction strategy that is able to search for consistent features under significant rotations. Our system mainly consists of rotation invariant feature extraction, odometry estimation, and probability map construction. Thorough experiments have been performed to evaluate the proposed method, including experiments on a warehouse AGV and a hand-held mobile device. The results have shown that the proposed method is able to provide reliable and accurate localization and mapping at high frequency. It can be implemented on most mobile platforms such as UAV and hand-held scanners, where the computational resources are limited. We have made the source codes publicly available.

FIGURE 8.7: Integration of the proposed method on a hand-held scanner. (a) lightweight hand-held scanner using solid-state LiDAR as perception system; (b) localization and mapping result.

Chapter 9

Conclusion and Future Work

9.1 Conclusion

In this thesis, we have developed a lightweight and robust full SLAM system for autonomous vehicles. We analyzed the limitations of the existing SLAM systems in robotic applications and aimed to improve from both front-end odometry estimation and back-end re-localization aspects. We performed experiments on autonomous driving cars, warehouse AGVs and hand-held devices to evaluate the robustness of the proposed SLAM system in practical applications. In summary, we have introduced front-end LiDAR odometry FLOAM, back-end saliency-based loop closure detection, LiDAR-based ISC-LCD, and full SLAM named intensity-SLAM. Many important theoretical and practical results have been achieved which are recalled below:

- Chapter 3 presented a computationally efficient LiDAR Odometry framework which provides a LiDAR-based localization solution to robotic applications with limited computational resources. Note that lines and planes are often distinctive, we adopt an efficient feature extraction via local smoothness analysis to search for edge and planar features respectively. The extracted features were associated with global lines and planes and the robot pose was achieved by minimizing point-to-edge and point-to-plane distances. Compared to the traditional point cloud matching methods such as ICP, the computational cost was significantly reduced. Moreover, we solved motion

prediction via non-iterative calculation to further reduce the computational cost. To demonstrate the robustness of the proposed method in practical applications, thorough experiments have been done to evaluate the performance, including simulation, indoor AGV test and outdoor autonomous driving test. Our method has achieved an average localization accuracy of 2 *cm* in indoor tests. It remains one of the most accurate and fastest open-sourced methods on the KITTI dataset.

- Chapter 4 presented a fast loop closure detection method via binarized saliency map. Traditional approaches such as FABMAP and DBoW2 use feature descriptors to compress the image content and build a descriptor vocabulary for indexing. However, these methods require intensive mathematical calculation to estimate the similarity of two images, which is less efficient than binary operation. It was observed that operation on binary images is faster than matching feature descriptor. Hence, we introduced an efficient binary-content-based retrieval strategy on top of existing feature-based methods to improve the searching efficiency. The experiment results have demonstrated that it is able to identify loop closure places without false positives. The proposed method was also compared with state-of-the-art methods such as FAB and DBoW2. The results have shown that it achieves faster computing speed than other approaches. In addition, no offline training is required in our approach, which makes it easy for implementation.
- Chapter 5 presented a novel framework for visual place recognition via saliency re-identification. It mainly consists of two components, namely saliency detection and retrieval. Compared to the previous chapter, we introduced KCC for saliency retrieval to improve the recall rate and recall precision. To reduce the computational cost, both tasks were performed in frequency domain to take advantage of the efficiency of element-wise operation, resulting in overall computational complexity of $\mathcal{O}(N \log N)$. The proposed method is open sourced as a C++ library. The experiments have shown that the proposed method is more computationally efficient and accurate compared with other state-of-the-art methods.
- Chapter 6 presented a robust LiDAR-based loop closure detection approach by integrating both geometry and intensity information. Compared to the vision-based LCD, LiDAR is more robust to illumination change and viewing

angle change. Existing works on LiDAR based loop closure detection mainly leverage the geometric-only descriptor and ignore intensity reading. Inspired by the recent research on LiDAR intensity, we argued that the intensity information can be effective for place recognition and proposed a global 3D descriptor named intensity scan context. To reduce the computational cost, an efficient two-stage hierarchical intensity scan context retrieval has been proposed, consisting of fast binary-operation based geometry indexing and intensity structure re-identification. It costs only 1.2 *ms* for each query on a normal computer in practice. Thorough experiments have been conducted, including local run tests with an autonomous warehouse robot and public dataset tests to evaluate the proposed method. The results have shown that our method achieves competitive recall precision and recall rate among existing state-of-the-art methods.

- Chapter 7 presented a novel full SLAM framework for LiDAR-based localization system, which integrated both front-end and back-end SLAM systems that were proposed in the previous chapters. Furthermore, we introduced intensity features for both LiDAR odometry estimation and global optimization to seek improvement. We began with the physical model of intensity measurement and illustrated the use of intensity information. We introduced the intensity map and intensity residual for pose estimation to improve localization accuracy. Thorough experiments were performed, including the indoor AGV mapping in the warehouse and the outdoor autonomous driving in urban areas. The experiment results have shown that by integrating intensity information, the localization accuracy is significantly improved. The proposed method has achieved much lower translational error and rotational error compared to geometric-only methods.
- Chapter 8 presented a SLAM system based on the recently introduced solid-state LiDAR. Solid-state LiDAR is an emerging LiDAR system with higher update frequency and smaller FoV compared to traditional mechanical LiDAR. We made some improvements on our SLAM system that was proposed in the previous chapter. To tackle the tracking loss problem, we introduced an improved feature extraction strategy which was able to search for consistent features under significant rotations. Our system mainly consisted of rotation invariant feature extraction, odometry estimation, and probability map construction. We also showed that the proposed SLAM system is able to be

used on many mobile devices with limited computational resources. Thorough experiments have been performed to evaluate the proposed method, including experiments on a warehouse AGV and a hand-held mobile device. The results showed that the proposed method is able to provide reliable and accurate localization and mapping at high frequency. It can be implemented on most mobile platforms such as UAV and hand-held scanners.

9.2 Limitations of Current Research

In this thesis, we have introduced several contributions in SLAM research in Section 9.1. However, there are also some limitations of current research. We will discuss the limitations in this section and briefly introduce some potential solutions in Section 9.3.

9.2.1 SLAM in Degraded Environments

For most applications such as autonomous driving, building inspections, the SLAM algorithm works perfectly. However, as we mentioned in the literature review, existing methods (and our method) mainly leverage the matching of geometric features. The target is often to minimize the point-to-point, point-to-edge or point-to-plane residuals. These methods fail when there are not enough features. For example, for the SLAM task in a coal mine (like the DARPA challenge), or the SLAM under different weather conditions (the sensor measurement noise significantly increases on the rainy days). In such case, the proposed SLAM cannot work. It is also important to solve the SLAM problem in such degraded environments.

9.2.2 Loop Closure Detection under Illumination Changes

In Chapter 4 and Chapter 5, we have proposed different approaches to solve the loop closure detection problem. However, the loop closure detection methods developed in the thesis only work during a relatively short period of time. For example, the evaluation on the KITTI dataset only validates the performance during 1-2 hours driving, where the weather change is very limited. However, Chapter 4

and Chapter 5 are not able to tackle the long-term place recognition, where the environmental changes may be significant. For example, an autonomous driving car may have to run 365 days a year, and the car may revisit the same place at night, rainy/snowy/cloudy day, or different seasons. The saliency information can be much different. It remains a challenge topic for research.

9.2.3 SLAM in Dynamic Environments

In Chapter 7 and Chapter 8, we have proposed SLAM systems for different kinds of sensors. The performance is good under a static environment. The localization accuracy decreases when there are dynamic objects in the environment. In the presence of dynamic objects, the current research is not able to distinguish dynamic features from static features. Therefore, both static and dynamic features will be included for iterative pose estimation, which does not guarantee convergence. However, in many practical applications, dynamic objects are inevitable. For example, walking human operators in a warehouse environment and nearby moving cars for autonomous driving.

9.3 Future Work

In Section 9.2, we have mentioned some limitations of current research. In this section, we will briefly introduce some interesting ideas or applications that can be used to solve the limitations and further extend this research as future work.

9.3.1 Multi-sensor Fusion

Rather than using a single perception system, one possible solution to improve the localization accuracy is to fuse multiple perception systems together.

9.3.1.1 Integration with Inertial Measurement Unit Sensor

Inertial Measurement Unit (IMU) is a perception sensor used for measuring the orientation of the robot body. An IMU often consists of a magnetometer, a gyroscope and an accelerometer. As stated in Section 2.2, the iterative pose optimization requires operation in Lie Group to find the orientation, which is a non-linear problem. In such cases, it does not guarantee a global optimal solution. Moreover, for perception sensors with small FoV such as cameras, tracking loss may occur under large rotation change which is a critical problem to a localization system. A possible solution is to combine IMU and SLAM to achieve Visual Inertial SLAM (VISLAM) or LiDAR Inertial SLAM (LISLAM), where the IMU can be fused in either loosely-coupled or tightly-coupled manner.

9.3.1.2 Combination of Both LiDAR and Visual SLAM

In Chapter 4, Chapter 5 and Chapter 6 we have studied both vision-based and LiDAR-based solutions for LCD. LiDAR features are more robust to illumination change while visual features are more computationally efficient. A possible solution towards a long-term and reliable re-localization is to combine both geometric and visual information to improve the robustness and efficiency of existing methods. Both geometric and visual information can be utilized in order to avoid false detection.

9.3.1.3 Integration with Other Localization Techniques

As stated in Chapter 2.1, SLAM is limited by the computational resources and perception sensor. For example, the update frequency of a VSLAM or a LSLAM is capped at 10-20 *Hz*. Moreover, for a fast moving vehicle, motion distortion (for LiDAR) or image blurring (for camera) becomes significant. On the other hand, other localization methods can be used to work together with SLAM in some scenarios. For example, we could possibly combine SLAM and UWB to achieve large-scale and long-term localization, where the UWB can be used to correct the localization drifts.

9.3.2 Machine Learning

Deep learning has shown its great advantage in robotics during the past years. Different CNNs have been proposed which have better performance than traditional methods in some scenarios.

9.3.2.1 Loop Closure Detection via Incremental Learning

Incremental learning is a method of machine learning in which input data is continuously used to extend the existing model's knowledge. Existing methods often use offline training to learn the knowledge of landmarks prior to applications, e.g., DBoW2. However, the trained knowledge is fixed and existing methods are not able to learn new objects online. A more intelligent system is expected to grow its knowledge with new places visited. In the recent research on incremental learning, knowledge distillation could be applied to adjust the complexity of neural networks with more data or more classes added into the database. Similar ideas can be used in the field of LCD, so that the SLAM could change its knowledge in different environments.

9.3.2.2 Semantic SLAM

Existing SLAM systems mainly focus on the feature level. In the real world, it is more straightforward to formulate the SLAM at the object level and it is unlikely to have false matching at the object level compared to features. Existing machine learning frameworks such as PointNet [155] and SOLO [156] have already demonstrated the ability to recognize multiple objects. Therefore, one possible solution to a more robust and reliable SLAM system is to apply semantic recognition. Moreover, in most scenarios, dynamic objects are inevitable, e.g., walkers in autonomous driving scenarios. Existing feature based approaches are not able to distinguish between dynamic features and static features. Once the semantic information is available, we can achieve better localization results by distinguishing static and dynamic information.

9.3.3 Applications

9.3.3.1 Virtual Reality

VR is one of the very promising technologies. In Chapter 8 we have shown that the proposed LiDAR SLAM can reconstruct dense mapping which is close to the real environment. The constructed map can be integrated into a VR system for visualization.

9.3.3.2 Drone Inspection

UAVs are widely used in various fields such as agriculture, architecture inspection, etc. In Chapter 7 and Chapter 8 we have presented a light-weight and robust localization solution for autonomous vehicles. UAV systems are often constrained by the low computing power and payload limit. In such cases, our SLAM system is lightweight and hence is suitable to be implemented on UAVs to provide real-time localization.

List of Author's Publications

Journal articles related to the thesis

- **Han Wang**, Chen Wang and Lihua Xie, “Intensity-SLAM: Intensity Assisted Localization and Mapping for Large Scale Environment,” published in *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1715-1721, April 2021, doi: 10.1109/LRA.2021.3059567 (the contents of this paper are also selected by ICRA’21 Program Committee for presentation at the conference).
- **Han Wang**, Chen Wang, and Lihua Xie, “Lightweight 3-D Localization and Mapping for Solid-State LiDAR,” published in *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1801-1807, April 2021, doi: 10.1109/LRA.2021.3060392 (the contents of this paper are also selected by ICRA’21 Program Committee for presentation at the conference).

Other journal articles

- Juncheng Li, Maopeng Ran, **Han Wang** and Lihua Xie, “A Behavior-Based Mobile Robot Navigation Method with Deep Reinforcement Learning,” published in *Unmanned Systems*, vol. 09, no. 03, pp. 201-209, 2021, doi: 10.1142/S2301385021410041.

Conference articles related to the thesis

- **Han Wang**, Chen Wang, and Lihua Xie, “F-LOAM: Fast Lidar Odometry And Mapping in Large Scale Environment,” submitted to *2021/2021*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2021.

- **Han Wang**, Chen Wang, and Lihua Xie, “Online Visual Place Recognition via Saliency Re-identification,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Las Vegas, NV, USA, 2020, pp. 5030-5036, doi: 10.1109/IROS45743.2020.9341703.
- **Han Wang**, Chen Wang, and Lihua Xie, “Intensity Scan Context: Coding Intensity and Geometry Relations for Loop Closure Detection,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, Paris, France, 2020, pp. 2095-2101, doi: 10.1109/ICRA40945.2020.9196764.
- **Han Wang**, Juncheng Li, Maopeng Ran, and Lihua Xie, “Fast Loop Closure Detection via Binary Content,” in *IEEE International Conference on Control and Automation 2019 (ICCA)*, Edinburgh, United Kingdom, 2019, pp. 1563-1568, doi: 10.1109/ICCA.2019.8899937.

Other conference articles

- Juncheng Li, Maopeng Ran, **Han Wang**, and Lihua Xie, “MPC-based Unified Trajectory Planning and Tracking Control Approach for Automated Guided Vehicles,” in *IEEE International Conference on Control and Automation 2019 (ICCA)*, Edinburgh, United Kingdom, 2019, pp. 374-380, doi: 10.1109/ICCA.2019.8899955.
- **Han Wang**, Muqing Cao, Hao Jiang, and Lihua Xie, “Feasible Computationally Efficient Path Planning for UAV Collision Avoidance,” in *IEEE International Conference on Control and Automation 2018 (ICCA)*, Anchorage, AK, USA, 2018, pp. 576-581, doi: 10.1109/ICCA.2018.8444284.

Bibliography

- [1] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, “Simultaneous localization and mapping: A survey of current trends in autonomous driving,” *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194–220, 2017. [1](#)
- [2] M. Campbell, M. Egerstedt, J. P. How, and R. M. Murray, “Autonomous driving in urban environments: approaches, lessons and challenges,” *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1928, pp. 4649–4672, 2010. [1](#)
- [3] J. Mehami, M. Nawi, and R. Y. Zhong, “Smart automated guided vehicles for manufacturing in the context of industry 4.0,” *Procedia manufacturing*, vol. 26, pp. 1077–1086, 2018. [1](#)
- [4] J. Theunissen, H. Xu, R. Y. Zhong, and X. Xu, “Smart agv system for manufacturing shopfloor in the context of industry 4.0,” in *2018 25th International Conference on Mechatronics and Machine Vision in Practice (M2VIP)*. IEEE, 2018, pp. 1–6. [1](#)
- [5] S. Lee and S. Lee, “Embedded visual slam: Applications for low-cost consumer robots,” *IEEE Robotics & Automation Magazine*, vol. 20, no. 4, pp. 83–95, 2013. [1](#)
- [6] N. Mir-Nasiri, H. Siswoyo, and M. H. Ali, “Portable autonomous window cleaning robot,” *Procedia computer science*, vol. 133, pp. 197–204, 2018. [1](#)
- [7] S.-h. Qian, S.-r. Ge, Y.-s. Wang, Y. Wang, and C.-q. Liu, “Research status of the disaster rescue robot and its applications to the mine rescue [j],” *Robot*, vol. 28, no. 3, pp. 350–354, 2006. [1](#)
- [8] A. H. Reddy, B. Kalyan, and C. S. Murthy, “Mine rescue robot system—a review,” *Procedia Earth and Planetary Science*, vol. 11, pp. 457–462, 2015. [1](#)
- [9] K. Zheng, Q. Zheng, H. Yang, L. Zhao, L. Hou, and P. Chatzimisios, “Reliable and efficient autonomous driving: the need for heterogeneous vehicular networks,” *IEEE Communications Magazine*, vol. 53, no. 12, pp. 72–79, 2015. [1](#)

- [10] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of rgb-d slam systems,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 573–580. [1](#)
- [11] T. M. Nguyen, A. H. Zaini, K. Guo, and L. Xie, “An ultra-wideband-based multi-uav localization system in gps-denied environments,” in *2016 International Micro Air Vehicles Conference*, 2016. [1](#)
- [12] Z. Chen, H. Zou, H. Jiang, Q. Zhu, Y. C. Soh, and L. Xie, “Fusion of wifi, smartphone sensors and landmarks using the kalman filter for indoor localization,” *Sensors*, vol. 15, no. 1, pp. 715–732, 2015. [1](#)
- [13] E. Garcia-Fidalgo and A. Ortiz, *Methods for Appearance-based Loop Closure Detection*. Springer, 2018. [3](#)
- [14] R. Bajaj, S. L. Ranaweera, and D. P. Agrawal, “Gps: location-tracking technology,” *Computer*, vol. 35, no. 4, pp. 92–94, 2002. [14](#)
- [15] H. Rütters, I. Möller, F. May, K. Flornes, V. Hladik, A. Arvanitis, N. Gülec, C. Bakiler, A. Dudu, L. Kucharic *et al.*, “State-of-the-art of monitoring methods to evaluate storage site performance,” *CGS Europe Key Report*, vol. 1, p. 109, 2013. [14](#)
- [16] G. Morgan-Owen and G. Johnston, “Differential gps positioning,” *Electronics & Communication Engineering Journal*, vol. 7, no. 1, pp. 11–21, 1995. [14](#)
- [17] G. Welch, G. Bishop *et al.*, “An introduction to the kalman filter,” 1995. [14](#)
- [18] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The kitti dataset,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013. [15](#), [56](#), [73](#), [75](#), [92](#), [93](#), [102](#), [105](#), [109](#)
- [19] M. Field, D. Stirling, F. Naghdy, and Z. Pan, “Motion capture in robotics review,” in *2009 IEEE International Conference on Control and Automation*. IEEE, 2009, pp. 1697–1702. [15](#)
- [20] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of rgb-d slam systems,” in *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012. [15](#), [92](#), [93](#)
- [21] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, “The euroc micro aerial vehicle datasets,” *The International Journal of Robotics Research*, 2016. [Online]. Available: <http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.abstract> [15](#)
- [22] A. Kushleyev, D. Mellinger, C. Powers, and V. Kumar, “Towards a swarm of agile micro quadrotors,” *Autonomous Robots*, vol. 35, no. 4, pp. 287–300, 2013. [16](#)

- [23] W. Jiang, H. Xue, C. Miao, S. Wang, S. Lin, C. Tian, S. Murali, H. Hu, Z. Sun, and L. Su, "Towards 3d human pose construction using wifi," in *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*, 2020, pp. 1–14. [16](#)
- [24] F. Mazhar, M. G. Khan, and B. Sällberg, "Precise indoor positioning using uwb: a review of methods, algorithms and implementations," *Wireless Personal Communications*, vol. 97, no. 3, pp. 4467–4491, 2017. [15](#)
- [25] S. Boonsriwai and A. Apavatjrut, "Indoor wifi localization on mobile devices," in *2013 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*. IEEE, 2013, pp. 1–5. [17](#)
- [26] M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "Spotfi: Decimeter level localization using wifi," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 2015, pp. 269–282. [17](#)
- [27] L. George and A. Mazel, "Humanoid robot indoor navigation based on 2d bar codes: Application to the nao robot," in *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. IEEE, 2013, pp. 329–335. [18](#)
- [28] W. X. Wang, "Agv magnetic sensor data acquisition system," in *Applied Mechanics and Materials*, vol. 713. Trans Tech Publ, 2015, pp. 1056–1060. [19](#)
- [29] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *The international journal of Robotics Research*, vol. 5, no. 4, pp. 56–68, 1986. [19](#)
- [30] S. Se, D. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *The international Journal of robotics Research*, vol. 21, no. 8, pp. 735–758, 2002. [19](#)
- [31] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015. [20](#), [24](#), [25](#), [34](#), [56](#)
- [32] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual-inertial and multi-map slam," *arXiv preprint arXiv:2007.11898*, 2020. [20](#)
- [33] A. R. Khairuddin, M. S. Talib, and H. Haron, "Review on simultaneous localization and mapping (slam)," in *2015 IEEE International Conference on Control System, Computing and Engineering (ICCSCE)*. IEEE, 2015, pp. 85–90. [21](#)
- [34] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003. [22](#)

- [35] Y. Chen, Y. Chen, and G. Wang, “Bundle adjustment revisited,” *arXiv preprint arXiv:1912.03858*, 2019. [23](#)
- [36] G. Klein and D. Murray, “Parallel tracking and mapping for small ar workspaces,” in *2007 6th IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234. [24](#)
- [37] J. Engel, V. Koltun, and D. Cremers, “Direct sparse odometry,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017. [25](#)
- [38] J. Engel, T. Schöps, and D. Cremers, “Lsd-slam: Large-scale direct monocular slam,” in *European conference on computer vision*. Springer, 2014, pp. 834–849. [25](#)
- [39] P. Bergmann, R. Wang, and D. Cremers, “Online photometric calibration of auto exposure video for realtime visual odometry and slam,” *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 627–634, 2017. [25](#)
- [40] R. B. Rusu, N. Blodow, and M. Beetz, “Fast point feature histograms (fpfh) for 3d registration,” in *2009 IEEE international conference on robotics and automation*. IEEE, 2009, pp. 3212–3217. [25](#), [38](#), [39](#), [99](#), [101](#)
- [41] S. Salti, F. Tombari, and L. Di Stefano, “Shot: Unique signatures of histograms for surface and texture description,” *Computer Vision and Image Understanding*, vol. 125, pp. 251–264, 2014. [25](#)
- [42] Z. Zhang, “Iterative point matching for registration of free-form curves and surfaces,” *International journal of computer vision*, vol. 13, no. 2, pp. 119–152, 1994. [25](#)
- [43] E. Mendes, P. Koch, and S. Lacroix, “Icp-based pose-graph slam,” in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2016, pp. 195–200. [26](#)
- [44] R. Tiar, N. Ouadah, O. Azouaoui, M. Djehaich, H. Ziane, and N. Achour, “Icp-slam methods implementation on a bi-steerable mobile robot,” in *2013 IEEE 11th International Workshop of Electronics, Control, Measurement, Signals and their application to Mechatronics*. IEEE, 2013, pp. 1–6. [26](#)
- [45] P. Biber and W. Straßer, “The normal distributions transform: A new approach to laser scan matching,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)(Cat. No. 03CH37453)*, vol. 3. IEEE, 2003, pp. 2743–2748. [26](#)
- [46] T. Schmiedel, E. Einhorn, and H.-M. Gross, “Iron: A fast interest point descriptor for robust ndt-map matching and its application to robot localization,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 3144–3151. [26](#)

- [47] W. Wen, L.-T. Hsu, and G. Zhang, “Performance analysis of ndt-based graph slam for autonomous vehicle in diverse typical driving scenarios of hong kong,” *Sensors*, vol. 18, no. 11, p. 3928, 2018. [26](#)
- [48] M. Magnusson, “The three-dimensional normal-distributions transform: an efficient representation for registration, surface analysis, and loop detection,” Ph.D. dissertation, Örebro universitet, 2009. [27](#)
- [49] J. Zhang and S. Singh, “Loam: Lidar odometry and mapping in real-time.” in *Robotics: Science and Systems*, vol. 2, 2014. [28](#), [29](#), [46](#), [60](#), [120](#), [132](#), [134](#), [137](#), [140](#)
- [50] T. Shan and B. Englot, “Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4758–4765. [28](#), [47](#), [60](#), [104](#), [120](#), [126](#)
- [51] J. Zhang and S. Singh, “Visual-lidar odometry and mapping: Low-drift, robust, and fast,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2174–2181. [28](#)
- [52] F. Dellaert, “Factor graphs and gtsam: A hands-on introduction,” Georgia Institute of Technology, Tech. Rep., 2012. [31](#)
- [53] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “g 2 o: A general framework for graph optimization,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3607–3613. [31](#)
- [54] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard, “A tutorial on graph-based slam,” *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010. [32](#), [124](#)
- [55] M. Cummins and P. Newman, “Fab-map: Probabilistic localization and mapping in the space of appearance,” *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008. [33](#), [68](#), [77](#)
- [56] M. Cummins, “Highly scalable appearance-only slam-fab-map 2.0,” *Proc. Robotics: Sciences and Systems (RSS), 2009*, 2009. [33](#)
- [57] C. Chow and C. Liu, “Approximating discrete probability distributions with dependence trees,” *IEEE transactions on Information Theory*, vol. 14, no. 3, pp. 462–467, 1968. [33](#)
- [58] D. Gálvez-López and J. D. Tardos, “Bags of binary words for fast place recognition in image sequences,” *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012. [33](#), [69](#), [77](#), [94](#), [101](#), [110](#), [112](#), [113](#)
- [59] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, “Orb: An efficient alternative to sift or surf,” in *Computer Vision (ICCV), 2011 IEEE international conference on*. IEEE, 2011, pp. 2564–2571. [33](#), [67](#), [69](#)

- [60] S. Bazeille and D. Filliat, “Combining odometry and visual loop-closure detection for consistent topo-metrical mapping,” *RAIRO-Operations Research-Recherche Opérationnelle*, vol. 44, no. 4, pp. 365–377, 2010. [34](#)
- [61] E. Garcia-Fidalgo and A. Ortiz, “ibow-led: An appearance-based loop-closure detection approach using incremental bags of binary words,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3051–3057, 2018. [34](#), [68](#), [82](#), [94](#)
- [62] D. Schlegel and G. Grisetti, “Hbst: A hamming distance embedding binary search tree for visual place recognition,” *arXiv preprint arXiv:1802.09261*, 2018. [34](#), [83](#), [94](#)
- [63] M. Muja and D. G. Lowe, “Scalable nearest neighbor algorithms for high dimensional data,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 11, pp. 2227–2240, 2014. [34](#)
- [64] P. L. N. Carrasco, F. Bonin-Font, and G. Oliver-Codina, “Global image signature for visual loop-closure detection,” *Autonomous Robots*, vol. 40, no. 8, pp. 1403–1417, 2016. [34](#), [83](#), [94](#)
- [65] M. Gehrig, E. Stumm, T. Hinzmann, and R. Siegwart, “Visual place recognition with probabilistic voting,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3192–3199. [35](#), [83](#)
- [66] S. Leutenegger, M. Chli, and R. Siegwart, “Brisk: Binary robust invariant scalable keypoints,” in *2011 IEEE international conference on computer vision (ICCV)*. Ieee, 2011, pp. 2548–2555. [35](#), [67](#), [69](#), [83](#)
- [67] C. Wang, J. Yang, L. Xie, and J. Yuan, “Kervolutional neural networks,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 31–40. [35](#), [83](#)
- [68] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford, “Deep learning features at scale for visual place recognition,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3223–3230. [35](#), [36](#), [83](#)
- [69] L. Tang, Y. Wang, Q. Luo, X. Ding, and R. Xiong, “Adversarial feature disentanglement for place recognition across changing appearance,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1301–1307. [35](#)
- [70] P. Yin, L. Xu, X. Li, C. Yin, Y. Li, R. A. Srivatsan, L. Li, J. Ji, and Y. He, “A multi-domain feature learning method for visual place recognition,” in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 319–324. [35](#)

- [71] S. Garg, N. Suenderhauf, and M. Milford, “Lost? appearance-invariant place recognition for opposite viewpoints using visual semantics,” *arXiv preprint arXiv:1804.05526*, 2018. [35](#)
- [72] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, “Netvlad: Cnn architecture for weakly supervised place recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5297–5307. [35](#), [36](#), [83](#)
- [73] S. Hausler, A. Jacobson, and M. Milford, “Multi-process fusion: Visual place recognition using multiple image processing methods,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1924–1931, 2019. [36](#), [37](#), [83](#)
- [74] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005. [36](#), [83](#)
- [75] A. Kim and R. M. Eustice, “Real-time visual slam for autonomous underwater hull inspection using visual saliency,” *IEEE Transactions on Robotics*, vol. 29, no. 3, pp. 719–733, 2013. [37](#), [84](#)
- [76] M. Birem, J.-C. Quinton, F. Berry, and Y. Mezouar, “Sail-map: Loop-closure detection using saliency-based features,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 4543–4548. [37](#), [84](#)
- [77] Z. Xin, Y. Cai, T. Lu, X. Xing, S. Cai, J. Zhang, Y. Yang, and Y. Wang, “Localizing discriminative visual landmarks for place recognition,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019. [38](#), [84](#)
- [78] R. Dubé, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, “Seg-match: Segment based place recognition in 3d point clouds,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5266–5272. [39](#), [101](#)
- [79] B. Douillard, J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, and A. Frenkel, “On the segmentation of 3d lidar point clouds,” in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 2798–2805. [39](#)
- [80] A. Zaganidis, A. Zerntev, T. Duckett, and G. Cielniak, “Semantically assisted loop closure in slam using ndt histograms,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4562–4568. [39](#)
- [81] G. D. Tipaldi and K. O. Arras, “Flirt-interest regions for 2d range data,” in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 3616–3622. [39](#), [99](#)

- [82] D. L. Rizzini, “Place recognition of 3d landmarks based on geometric relations,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 648–654. [40](#), [101](#), [112](#), [113](#), [123](#)
- [83] G. Kim and A. Kim, “Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4802–4809. [40](#), [104](#), [107](#), [112](#), [113](#), [123](#)
- [84] M. Bosse and R. Zlot, “Place recognition using keypoint voting in large 3d lidar datasets,” in *2013 IEEE International Conference on Robotics and Automation*, 2013, pp. 2677–2684. [40](#), [101](#), [102](#)
- [85] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, “Imu preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation,” in *Proc. Robotics: Sciences and Systems (RSS), 2015*. Georgia Institute of Technology, 2015. [41](#)
- [86] C. Wang, T. Ji, T.-M. Nguyen, and L. Xie, “Correlation flow: Robust optical flow using kernel cross-correlators,” in *2018 International Conference on Robotics and Automation (ICRA)*, 2018, pp. 836–841. [43](#)
- [87] C. Debeunne and D. Vivet, “A review of visual-lidar fusion based simultaneous localization and mapping,” *Sensors*, vol. 20, no. 7, p. 2068, 2020. [45](#)
- [88] P. J. Besl and N. D. McKay, “Method for registration of 3-d shapes,” in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. International Society for Optics and Photonics, 1992, pp. 586–606. [46](#), [108](#), [132](#), [134](#)
- [89] J.-E. Deschaud, “Imls-slam: scan-to-model matching based on 3d data,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2480–2485. [46](#), [60](#)
- [90] D. Rozenberszki and A. L. Majdik, “Lol: Lidar-only odometry and localization in 3d point cloud maps,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4379–4385. [47](#), [116](#)
- [91] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and R. Daniela, “Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020. [47](#)
- [92] D. Yin, Q. Zhang, J. Liu, X. Liang, Y. Wang, J. Maanpää, H. Ma, J. Hyypää, and R. Chen, “Cae-lo: Lidar odometry leveraging fully unsupervised convolutional auto-encoder for interest point detection and feature description,” *arXiv preprint arXiv:2001.01354*, 2020. [47](#)

- [93] J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, “Stacked convolutional auto-encoders for hierarchical feature extraction,” in *International conference on artificial neural networks*. Springer, 2011, pp. 52–59. [47](#)
- [94] H. Shi, G. Lin, H. Wang, T.-Y. Hung, and Z. Wang, “Spsequencenet: Semantic segmentation network on 4d point clouds,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4574–4583. [47](#)
- [95] R. Bergelt, O. Khan, and W. Hardt, “Improving the intrinsic calibration of a velodyne lidar sensor,” in *2017 IEEE SENSORS*. IEEE, 2017, pp. 1–3. [48](#)
- [96] L. Zhang and P. N. Suganthan, “Robust visual tracking via co-trained kernelized correlation filters,” *Pattern Recognition*, vol. 69, pp. 82–93, 2017. [51](#)
- [97] T. D. Barfoot, “State estimation for robotics: A matrix lie group approach,” *Draft in preparation for publication by Cambridge University Press, Cambridge*, 2016. [53](#)
- [98] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, “Octomap: An efficient probabilistic 3d mapping framework based on octrees,” *Autonomous robots*, vol. 34, no. 3, pp. 189–206, 2013. [54](#), [138](#)
- [99] H. Moravec and A. Elfes, “High resolution maps from wide angle sonar,” in *Proceedings. 1985 IEEE international conference on robotics and automation*, vol. 2. IEEE, 1985, pp. 116–121. [55](#), [139](#)
- [100] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for autonomous driving? the kitti vision benchmark suite,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. [56](#), [57](#), [59](#), [125](#)
- [101] T. Qin, J. Pan, S. Cao, and S. Shen, “A general optimization-based framework for local odometry estimation with multiple sensors,” *arXiv preprint arXiv:1901.03638*, 2019. [56](#)
- [102] J. Graeter, A. Wilczynski, and M. Lauer, “Limo: Lidar-monocular visual odometry,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7872–7879. [56](#)
- [103] J. Engel, J. Stückler, and D. Cremers, “Large-scale direct slam with stereo cameras,” in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 1935–1942. [56](#)
- [104] K. Koide, J. Miura, and E. Menegatti, “A portable three-dimensional lidar-based system for long-term and wide-area people behavior measurement,” *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, p. 1729881419841532, 2019. [60](#), [116](#)
- [105] T. Qin, P. Li, and S. Shen, “Vins-mono: A robust and versatile monocular visual-inertial state estimator,” *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018. [60](#)

- [106] J. M. O’Kane, “A gentle introduction to ros,” 2014. [60](#)
- [107] R. Arroyo, P. F. Alcantarilla, L. M. Bergasa, and E. Romera, “Towards life-long visual localization using an efficient matching of binary sequences from images,” in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 6328–6335. [67](#)
- [108] T. Cieslewski and D. Scaramuzza, “Efficient decentralized visual place recognition using a distributed inverted index,” *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 640–647, 2017. [67](#)
- [109] H. Bay, T. Tuytelaars, and L. Van Gool, “Surf: Speeded up robust features,” in *European conference on computer vision*. Springer, 2006, pp. 404–417. [69](#), [74](#)
- [110] A. Glover, W. Maddern, M. Warren, S. Reid, M. Milford, and G. Wyeth, “Openfabmap: An open source toolbox for appearance-based loop closure detection,” in *2012 IEEE international conference on Robotics and automation (ICRA)*, 2012, pp. 4730–4735. [69](#), [101](#)
- [111] R. Mur-Artal and J. D. Tardós, “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras,” *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017. [69](#), [94](#), [101](#), [110](#)
- [112] Y. Huang, F. Sun, and Y. Guo, “Vlad-based loop closure detection for monocular slam,” in *Information and Automation (ICIA), 2016 IEEE International Conference on*, 2016, pp. 511–516. [69](#)
- [113] Y. Uchida, S. Sakazawa, and S. Satoh, “Image retrieval with fisher vectors of binary features,” *ITE Transactions on Media Technology and Applications*, vol. 4, no. 4, pp. 326–336, 2016. [69](#)
- [114] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004. [69](#)
- [115] D. G. Viswanathan, “Features from accelerated segment test (fast),” 2009.
- [116] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” in *European conference on computer vision*. Springer, 2010, pp. 778–792. [69](#)
- [117] M. Donk and W. van Zoest, “Effects of salience are short-lived,” *Psychological Science*, vol. 19, no. 7, pp. 733–739, 2008. [70](#), [84](#)
- [118] C. Wang, W. Wang, Y. Qiu, Y. Hu, and S. Scherer, “Visual Memorability for Robotic Interestingness via Unsupervised Online Learning,” in *European Conference on Computer Vision (ECCV)*, 2020. [70](#)
- [119] X. Hou and L. Zhang, “Saliency detection: A spectral residual approach,” in *Computer Vision and Pattern Recognition, 2007. CVPR’07. IEEE Conference on*, 2007, pp. 1–8. [70](#), [85](#)

- [120] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, “The new college vision and laser data set,” *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 595–599, May 2009. [Online]. Available: <http://www.robots.ox.ac.uk/NewCollegeData/> 75
- [121] J. Engel, V. Usenko, and D. Cremers, “A photometrically calibrated benchmark for monocular visual odometry,” *arXiv preprint arXiv:1607.02555*, 2016. 75
- [122] R. Cong, J. Lei, H. Fu, M.-M. Cheng, W. Lin, and Q. Huang, “Review of visual saliency detection with comprehensive information,” *IEEE Transactions on Circuits and Systems for Video Technology*, 2018. 81, 84
- [123] C. Wang, L. Zhang, L. Xie, and J. Yuan, “Kernel cross-correlator,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 81, 84, 88, 90
- [124] A. Srivastava, A. B. Lee, E. P. Simoncelli, and S.-C. Zhu, “On advances in statistical modeling of natural images,” *Journal of mathematical imaging and vision*, vol. 18, no. 1, pp. 17–33, 2003. 86
- [125] B. S. Reddy and B. N. Chatterji, “An fft-based technique for translation, rotation, and scale-invariant image registration,” *IEEE transactions on image processing*, vol. 5, no. 8, pp. 1266–1271, 1996. 87
- [126] M. Cummins and P. Newman, “FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance,” *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008. 92, 93
- [127] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, “1 year, 1000 km: The oxford robotcar dataset,” *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017. 92, 93
- [128] X. Gao, R. Wang, N. Demmel, and D. Cremers, “Ldso: Direct sparse odometry with loop closure,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 2198–2204. 94, 101
- [129] F. Tombari, S. Salti, and L. Di Stefano, “Unique shape context for 3d data description,” in *Proceedings of the ACM workshop on 3D object retrieval*, 2010, pp. 57–62. 99, 105
- [130] C. Wang, J. Yuan, and L. Xie, “Non-iterative slam,” in *Advanced Robotics (ICAR), 2017 18th International Conference on.* IEEE, 2017, pp. 83–90. 99
- [131] A. Kashani, M. Olsen, C. Parrish, and N. Wilson, “A review of lidar radiometric processing: From ad hoc intensity correction to rigorous radiometric calibration,” *Sensors*, vol. 15, no. 11, pp. 28 099–28 128, 2015. 99, 102, 103
- [132] K. P. Cop, P. V. Borges, and R. Dubé, “Delight: An efficient descriptor for global localisation using lidar intensities,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3653–3660. 100

- [133] J. Guo, P. V. Borges, C. Park, and A. Gawel, “Local descriptor for robust place recognition using lidar intensity,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1470–1477, 2019. [100](#), [103](#), [105](#), [108](#)
- [134] A. Anoosheh, T. Sattler, R. Timofte, M. Pollefeys, and L. Van Gool, “Night-to-day image translation for retrieval-based localization,” in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 5958–5964. [101](#)
- [135] T. Naseer, M. Ruhnke, C. Stachniss, L. Spinello, and W. Burgard, “Robust visual slam across seasons,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015, pp. 2529–2535. [101](#)
- [136] J. Levinson and S. Thrun, “Unsupervised calibration for multi-beam lasers,” in *Experimental Robotics*. Springer, 2014, pp. 179–193. [103](#)
- [137] G. Kim, B. Park, and A. Kim, “1-day learning, 1-year localization: Long-term lidar localization using scan context image,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1948–1955, 2019. [104](#)
- [138] S. Belongie, J. Malik, and J. Puzicha, “Shape matching and object recognition using shape contexts,” *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 4, pp. 509–522, 2002. [104](#)
- [139] T. Cieslewski, E. Stumm, A. Gawel, M. Bosse, S. Lynen, and R. Siegwart, “Point cloud descriptors for place recognition using sparse visual information,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 4830–4836. [105](#), [112](#), [113](#), [123](#)
- [140] M. J. Milford and G. F. Wyeth, “Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights,” in *2012 IEEE International Conference on Robotics and Automation*. IEEE, 2012, pp. 1643–1649. [108](#)
- [141] R. B. Rusu and S. Cousins, “3D is here: Point Cloud Library (PCL),” in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011. [110](#)
- [142] I. A. Barsan, S. Wang, A. Pokrovsky, and R. Urtasun, “Learning to localize using a lidar intensity map,” in *CoRL*, 2018, pp. 605–616. [117](#)
- [143] Y. Tian, X. Liu, L. Li, and W. Wang, “Intensity-assisted icp for fast registration of 2d-lidar,” *Sensors*, vol. 19, no. 9, p. 2124, 2019. [117](#)
- [144] R. A. Hewitt and J. A. Marshall, “Towards intensity-augmented slam with lidar and tof sensors,” in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 1956–1961. [117](#), [118](#), [119](#)
- [145] D. P. Frost, O. Kähler, and D. W. Murray, “Object-aware bundle adjustment for correcting monocular scale drift,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4770–4776. [117](#)

- [146] S. Khan, D. Wollherr, and M. Buss, “Modeling laser intensities for simultaneous localization and mapping,” *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 692–699, 2016. [117](#), [121](#)
- [147] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, “A flexible and scalable slam system with full 3d motion estimation,” in *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, November 2011. [117](#)
- [148] J. Lin and F. Zhang, “Loam_livox: A fast, robust, high-precision lidar odometry and mapping package for lidars of small fov,” *arXiv preprint arXiv:1909.06700*, 2019. [119](#), [131](#), [134](#)
- [149] S. Khan, D. Wollherr, and M. Buss, “Adaptive rectangular cuboids for 3d mapping,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2132–2139. [120](#)
- [150] K. M. Wurm, A. Hornung, M. Bennewitz, C. Stachniss, and W. Burgard, “Octomap: A probabilistic, flexible, and compact 3d map representation for robotic systems,” in *Proc. of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation*, vol. 2, 2010. [120](#)
- [151] J. J. Moré, “The levenberg-marquardt algorithm: implementation and theory,” in *Numerical analysis*. Springer, 1978, pp. 105–116. [123](#)
- [152] K. Koide, J. Miura, and E. Menegatti, “A portable 3d lidar-based system for long-term and wide-area people behavior measurement,” *IEEE Trans. Hum. Mach. Syst*, 2018. [126](#)
- [153] G. C. Burdea and P. Coiffet, *Virtual reality technology*. John Wiley & Sons, 2003. [131](#)
- [154] M. Quigley, B. Gerkey, and W. D. Smart, *Programming Robots with ROS: a practical introduction to the Robot Operating System*. ” O’Reilly Media, Inc.”, 2015. [140](#)
- [155] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660. [153](#)
- [156] N. Bernstein, N. Fong, I. Lam, M. Roy, D. G. Hendrickson, and D. R. Kelley, “Solo: doublet identification via semi-supervised deep learning,” *bioRxiv*, p. 841981, 2019. [153](#)