



SAPIENZA
UNIVERSITÀ DI ROMA

DOCTORAL THESIS

Direct Error Formulation for Real-Time SLAM Applications Across Diverse Sensors

Author:
Leonardo Brizi

Advisor:
Prof. Giorgio Grisetti

*A thesis submitted in fulfillment of the requirements
for the degree of Doctor of Philosophy in Engineering in Computer Science
in the*

Department of Computer, Control and Management Engineering
“Antonio Ruberti” (DIAG)

June 2025

Thesis defended the 18th of September 2025.

Direct Error Formulation for Real-Time SLAM Applications Across Diverse Sensors

Ph.D. Thesis - Sapienza University of Rome

Advisor: Prof. Giorgio Grisetti

Reviewer: Prof. Dr. Cyrill Stachniss

Reviewer: Prof. Alberto Pretto

© 2025 Leonardo Brizi. All rights reserved.

Author's email: brizi@diag.uniroma1.it

To Arianna

Abstract

This thesis investigates direct error minimization methods for real-time Simultaneous Localization and Mapping (SLAM), a fundamental problem in robotics and autonomous systems. SLAM enables sensor-equipped systems to construct representations of unknown environments while simultaneously estimating their own pose, relying exclusively on onboard sensory data. Traditional SLAM approaches often depend on explicit data association and indirect optimization techniques; in contrast, direct methods optimize error functions defined directly on sensor measurements, improving robustness and versatility across various sensor modalities.

Our research begins with the formulation of a direct error minimization pipeline for LiDAR, as realized in the MD-SLAM framework. This approach highlights the key advantage of direct methods, namely, the avoidance of explicit data association, and its modular design permits straightforward adaptation to additional sensor types such as RGB-D cameras. To further support the development and benchmarking of state estimation algorithms, we present a novel dataset featuring high-precision ground truth derived from a sensor fusion pipeline that combines GNSS-inertial measurements with multi-view LiDAR optimization. Within this framework, the LiDAR optimization problem minimizes a composite error function, integrating both geometric and photometric (direct) terms. The use of bilinear interpolation in the projection function enables sub-pixel accuracy in the alignment process.

Expanding this paradigm to radar sensors, we formalize a direct approach to radar odometry based on correlation maximization between a local map and incoming radar scans for ego-motion estimation. While the field of radar odometry has historically been dominated by indirect approaches, considered faster and more amenable to real-time operation, our work demonstrates that direct methods can achieve competitive performance when equipped with a GPU-parallelized implementation. This enables real-time operation without sacrificing the accuracy and versatility inherent to direct approaches, setting a new state of the art in radar odometry.

Collectively, this thesis demonstrates that direct error formulation is a powerful and adaptable framework for SLAM across diverse sensing modalities, providing advancements in accuracy, versatility and robustness, and, when accelerated with GPU computation, can achieve real-time capability.

Acknowledgments

Concludendo questo percorso di dottorato, desidero esprimere la mia gratitudine a tutte le persone che, a vario titolo, hanno reso possibile questo traguardo.

Il mio primo grazie va al prof. Giorgio Grisetti, per avermi guidato con passione e per aver creduto in me, sempre facendomi sentire trattato da pari a pari. Questa fiducia e questo rispetto hanno reso ogni confronto un'occasione di crescita e mi hanno dato la libertà di affrontare la ricerca con indipendenza e responsabilità.

Un ringraziamento speciale va al mio gruppo di ricerca, che per me è stato anche una seconda famiglia: Emanuele, Luca, Omar, Simone, Lorenzo e Davide. Con voi ho condiviso codice, esperimenti e notti, ma soprattutto amicizia, risate e una quantità innumerevole di discussioni che hanno reso ogni giornata più leggera. Un grazie particolare va anche a Fabio, che pur da tesista ha avuto un ruolo importante in questo percorso, contribuendo con entusiasmo e diventando parte integrante del gruppo.

Ai miei amici di sempre, Simone, Leonardo, William e Giovanni, grazie per essere rimasti al mio fianco anche quando non ero il più presente del gruppo: sapere di potervi ritrovare è stato un punto fermo in questi anni.

Un grazie va anche a chi mi ha accolto dall'altra parte dell'oceano, Tim Barfoot e il suo gruppo ASRL, in particolare Cédric, per aver reso possibile un'esperienza di ricerca che mi ha aperto nuovi orizzonti e insegnato molto, dentro e fuori dal laboratorio.

Al gruppo di Sentinel Road grazie per aver reso il mio soggiorno in Canada piacevole e ricco di avventure e risate, e al gruppo della B12[0-1][1], con cui ho condiviso le vere "sofferenze da PhD": ansie, scadenze, e la strana complicità che nasce solo tra chi vive la stessa avventura.

Infine, il grazie più grande va a chi ha reso tutto questo possibile sul piano personale: la mia famiglia, che mi ha sempre sostenuto con amore e fiducia, e la famiglia della mia compagna, che considero ormai parte della mia. E soprattutto a lei, la mia compagna di vita: se oggi arrivo a scrivere queste pagine è anche grazie al suo sostegno costante, ai suoi consigli e alla pazienza infinita con cui ha ascoltato i miei monologhi al punto che forse ormai ne sa più lei di me.

Non so se i segni del mio passaggio resteranno a lungo al DIAG, ma sono sicuro che il DIAG resterà a lungo dentro di me.

Glossary

- ATE** Absolute Trajectory Error [34](#), [36](#), [37](#), [51](#), [53–55](#), [76](#), [77](#), [79](#), [80](#), [92](#), [93](#), [124](#), [125](#), [129](#)
- BA** Bundle Adjustment [29](#), [35](#), [44](#), [49](#), [54](#), [56](#), [69–72](#), [74–79](#), [82](#), [120](#), [125](#), [126](#)
- BST** Binary Search Tree [28](#), [33](#)
- CPU** Central Processing Unit [35](#), [38–40](#), [76](#), [103](#), [114](#)
- DoF** Degree-of-Freedom [4](#), [46](#), [50](#), [61](#), [111](#), [119](#)
- EKF** Extended Kalman Filter [18](#), [87](#)
- FMCW** Frequency-Modulated Continuous-Wave [5](#), [14](#), [17](#), [98](#), [100](#), [101](#), [116](#), [120](#), [121](#), [123](#)
- FoV** Field of View [47](#), [48](#), [51](#), [66](#), [81](#), [96](#), [106](#), [116](#), [125](#)
- GD** Gradient Descend [18–20](#), [22](#)
- GN** Gauss-Newton [18–22](#), [63](#), [123](#)
- GNSS** Global Navigation Satellite System [4](#), [5](#), [43–45](#), [47–51](#), [54](#), [56](#), [70](#), [106](#), [110](#), [111](#)
- GP** Gaussian Process [7](#), [17](#), [101](#), [103](#), [106](#), [107](#), [110](#), [127](#)
- GPU** Graphic Processor Unit [2](#), [5](#), [29](#), [35](#), [38](#), [40](#), [76](#), [77](#), [103](#), [114](#), [117](#), [119](#), [126](#)
- ICP** Iterative Closest Point [29](#), [37](#), [48](#), [49](#), [59](#), [60](#), [88](#), [98](#), [106](#), [107](#), [113](#)
- ILS** Iterative Least-Squares [22](#), [32](#)
- IMU** Inertial Measurement Unit [2](#), [4](#), [34](#), [43](#), [45–49](#), [86](#), [87](#), [93](#), [98](#), [106](#), [110–113](#), [116](#), [119](#), [120](#)
- INS** Inertial Navigation System [43](#), [45](#), [54](#), [56](#), [106](#), [110](#)
- IRLS** Iterative Reweighted Least-Squares [21](#), [22](#), [88](#), [90](#)
- LiDAR** Light Detection and Ranging [1–5](#), [7](#), [8](#), [10–15](#), [25](#), [27–31](#), [33](#), [34](#), [36](#), [37](#), [40](#), [41](#), [43–51](#), [54](#), [56–62](#), [64–67](#), [69](#), [70](#), [72](#), [76–83](#), [85–88](#), [90](#), [92–95](#), [97](#), [98](#), [106–108](#), [110–113](#), [116](#), [117](#), [119–121](#), [123–126](#)

LM Levenberg-Marquardt [18](#), [19](#), [22](#), [60](#), [75](#)

LMS Least Mean Squares [87](#)

MLE Maximum Likelihood Estimation [18](#)

mmWave millimeter wave [95](#), [98](#)

NDT Normal Distributed Transform [29](#), [72](#)

NLS Non-Linear Least-Squares [18](#)

PCA Principal Component Analysis [50](#)

PGO Pose-Graph Optimization [70](#), [72](#)

PnP Perspective-n-Points [60](#), [62](#)

PPS Pulse Per Second [49](#)

RADAR Radio Detection and Ranging [1–5](#), [7](#), [8](#), [14–17](#), [25](#), [45](#), [86](#), [93](#), [95–108](#), [110–114](#), [116](#), [117](#), [119–121](#), [123](#), [126](#), [127](#), [130](#)

RANSAC RANdom SAmple Consensus [33](#), [62](#), [97](#), [98](#), [107](#)

RMSE Root Mean Square Error [34](#), [35](#), [53–55](#), [76](#), [79](#), [80](#), [91](#), [106](#), [108](#), [124](#), [126](#), [129](#)

RPE Relative Pose Error [51](#), [53–55](#), [106](#), [113](#), [124](#), [130](#)

RTK Real-Time Kinematic [4](#), [43–45](#), [47](#), [49](#), [51](#), [106](#), [110](#), [111](#)

SDF Signed Distance Function [71](#)

SfM Structure from Motion [27](#), [59](#), [69–71](#)

SLAM Simultaneous Localization and Mapping [1–5](#), [7](#), [8](#), [24](#), [25](#), [27–30](#), [34–38](#), [40](#), [41](#), [43–46](#), [51](#), [53](#), [54](#), [56](#), [69](#), [70](#), [72](#), [73](#), [78–80](#), [82](#), [83](#), [92–95](#), [111](#), [117](#), [119–121](#), [126](#), [129](#), [130](#)

SOTA state-of-the-art [5](#), [96](#), [108](#), [112](#), [116](#), [120](#)

SVD Singular Value Decomposition [60](#)

TSDF Truncated Signed Distance Function [29](#)

VO Visual Odometry [97](#)

Contents

Glossary	ix
1 Introduction	1
1.1 Motivation	1
1.2 Direct vs Indirect Methods	2
1.2.1 Indirect Methods	2
1.2.2 Direct Methods	3
1.3 Thesis Organization	4
1.4 Publications	5
2 Fundamentals	7
2.1 Sensors in Robotics	7
2.1.1 Camera	8
2.1.2 LiDAR	10
2.1.3 RADAR	14
2.2 Gaussian Process Regression	17
2.3 Non-Linear Optimization	18
2.3.1 First-Order Methods	19
2.3.2 Gauss-Newton Optimization	20
2.3.3 Lie Theory and Optimization on Manifold	22
2.3.4 Factor Graph	24
3 MD-SLAM: Multi-cue Direct SLAM	27
3.1 Related Work	29
3.2 Multi-cue Direct SLAM	31
3.2.1 Input Preprocessing	31
3.2.2 Direct Error Minimization	32
3.2.3 Tracking	33
3.2.4 Loop Detection and Validation	33
3.2.5 Pose-graph Optimization	34
3.3 Experimental Evaluation	34
3.3.1 RGB-D Results	35
3.3.2 LiDAR Results	36
3.4 Numerical Stability analysis	38
3.5 Conclusion	40

4	VBR: A Vision Benchmark in Rome	43
4.1	Related Work	44
4.2	The Datasets	47
4.2.1	Sensors setup	47
4.2.2	Calibration	48
4.2.3	Synchronization	48
4.2.4	Ground truth generation	49
4.2.5	Data selection	50
4.3	Benchmark	51
4.3.1	Evaluation	54
4.4	Conclusion	54
5	<i>Ca²Lib</i>: Simple and Accurate L-RGB Calibration using Small Common Markers	57
5.1	Related Work	58
5.2	LiDAR-RGB calibration	61
5.3	Experimental Evaluation	63
5.3.1	Synthetic Case	64
5.3.2	Real Case	64
5.4	Conclusion	65
6	Photometric LiDAR and RGB-D Bundle Adjustment	69
6.1	Related Work	70
6.2	Photometric Bundle Adjustment	73
6.2.1	Graph Construction	73
6.2.2	Photometric Error Minimization	74
6.3	Experimental Evaluation	76
6.3.1	RGB-D	78
6.3.2	3D LiDAR	79
6.3.3	RGB-D + 3D LiDAR	80
6.4	Conclusion	82
7	Enhancing LiDAR performance: Robust de-skewing exclusively relying on range measurements	85
7.1	Related Work	86
7.2	Velocity Based De-skewing	87
7.2.1	Velocity Based De-skewing	88
7.2.2	Error Metric for the Optimization	89
7.2.3	Data Association	90
7.3	Experimental Validation	90
7.3.1	De-skewing Effects in SLAM application	92
7.4	Conclusion	93
8	DRO: Doppler-Aware Direct RADAR Odometry	95
8.1	Related work	97
8.2	Direct RADAR Odometry	98
8.2.1	Problem statement and overview	98

8.2.2	Direct intensity objective function	99
8.2.3	Doppler-based objective function	101
8.2.4	Motion models	102
8.3	Experiments	103
8.3.1	Implementation	103
8.3.2	Metrics	105
8.3.3	Doppler-enabled automotive odometry	106
8.3.4	Public automotive benchmarks	110
8.3.5	Off-road navigation	112
8.3.6	Lateral velocity bias estimate	113
8.3.7	Computation time	114
8.4	Limitations	116
8.5	Conclusion	116
9	Conclusion	119
9.1	Summary	119
9.2	Outlook	120

Chapter 1

Introduction

IN this chapter, we introduce Simultaneous Localization and Mapping (SLAM) as the core problem of estimating both a robot’s trajectory and the structure of its environment in real time. We then motivate the shift to direct error formulation, showing how it leverages raw sensor data for greater robustness and computational efficiency, and contrast it with traditional indirect, feature-based pipelines. Finally, we outline the organization of the thesis and summarize the main contributions presented in each chapter.

1.1 Motivation

The evolution of autonomous systems, from self-driving vehicles navigating urban environments to drones inspecting critical infrastructure, demands robust perception capabilities that can operate reliably across diverse and challenging conditions. While traditional **SLAM** approaches have achieved remarkable success in controlled settings, real-world deployment exposes fundamental limitations that motivate our investigation into direct error formulation methods.

Beyond Feature Dependency. Indirect methods’ reliance on distinctive visual or geometric features creates critical failure modes. In textureless environments like tunnels, foggy conditions, or scenes dominated by glass and reflective surfaces, feature extraction becomes unreliable or impossible. Yet these are precisely the scenarios where robust perception is most crucial for safety-critical applications. Direct methods circumvent this limitation by leveraging all available sensor data, not just sparse salient points.

Computational Efficiency Through Simplicity. The two-stage pipeline of indirect methods, feature extraction followed by geometric optimization, introduces computational overhead and potential points of failure. Each stage must be tuned for specific sensors and environments, creating a complex parameter space. Direct formulation unifies these stages into a single optimization problem, reducing system complexity while maintaining or improving accuracy. This simplification becomes particularly valuable when deploying perception systems on resource-constrained platforms or when processing high-frequency sensor data in real-time.

Sensor Agnosticism as a Design Principle. Modern autonomous systems increasingly rely on heterogeneous sensor suites: cameras for semantic understanding, Light Detection and Ranging (LiDAR)s for precise geometry, Radio Detection and Ranging (RADAR)s

for all-weather operation, and Inertial Measurement Unit (IMU)s for high-frequency motion tracking. Traditional **SLAM** pipelines require sensor-specific feature extractors and matching strategies, leading to a proliferation of specialized algorithms. Our direct approach demonstrates that a single algorithmic framework can seamlessly handle RGB-D cameras, **LiDARs**, and even millimeter-wave **RADARs** by simply adapting the projection model and error metric. This versatility accelerates the integration of new sensors and simplifies system design.

Exploiting Modern Computational Architectures. The dense, regular structure of direct error computation naturally maps to parallel processing architectures. While indirect methods struggle to parallelize sparse feature matching, direct methods can leverage Graphic Processor Unit (GPU) acceleration to process millions of pixels simultaneously. This architectural alignment transforms previously intractable dense optimization problems into real-time capable systems, as we demonstrate with our **GPU**-accelerated implementations achieving real-time processing times.

Towards Robust Autonomy. Perhaps most critically, the ability to operate without explicit data association makes direct methods inherently more robust to perceptual aliasing and ambiguous correspondences. In cluttered environments or during aggressive motions where feature tracking fails, direct methods continue to provide reliable motion estimates by exploiting the full information content of sensor measurements. This thesis demonstrates that direct error formulation is not merely an alternative to feature-based methods but represents a in some cases a fundamental shift in how we approach the **SLAM** problem, one that aligns with the computational resources and robustness requirements of next-generation autonomous systems.

1.2 Direct vs Indirect Methods

In most contemporary approaches, we begin by defining a probabilistic model that maps noisy sensor measurements, denoted by $\mathbf{Z}$, to an estimator $\mathbf{X}$ of the hidden quantities of interest (*e.g.*, the 3D scene structure and sensor motion). A common strategy is to adopt a Maximum Likelihood formulation, seeking the parameter set $\mathbf{X}$ that makes the observed data $\mathbf{Z}$ most probable:

$$\mathbf{X}^* = \arg \max_{\mathbf{X}} P(\mathbf{Z} | \mathbf{X}). \quad (1.1)$$

From this general framework two algorithmic families emerge: *indirect methods* and *direct methods*, which differ in how they handle the raw sensor data.

1.2.1 Indirect Methods

Indirect methods decompose the estimation task into two sequential stages:

1. **Pre-processing (feature extraction & matching).** The sensor’s raw outputs are first transformed into an intermediate representation. Typically, this involves detecting a sparse set of salient keypoints and establishing correspondences between sensor measurements (though one can also match dense optical-flow fields or fit and match other geometric primitives, such as lines, curves or conics).

2. **Probabilistic inference.** These intermediate correspondences, now treated as noisy measurements $\mathbf{Z}$, are fed into a probabilistic estimator that solves for the underlying scene geometry and motion parameters.

Because the intermediate values represent geometric quantities, *i.e.* point positions, flow vectors, or primitive parameters, indirect methods naturally pose the estimation problem as minimization of a *geometric error*.

1.2.2 Direct Methods

Direct methods bypass the explicit feature-extraction step altogether. Instead of converting pixel intensities into higher-level descriptors, they incorporate the *raw photometric measurements*, the actual recorded brightness values, as inputs to the probabilistic model. In *passive vision* setups (standard video cameras), this leads to formulations that directly minimize a *photometric error*, measuring the discrepancy in pixel intensity after projecting a hypothesized 3D scene and motion into the image plane. For sensors that intrinsically produce geometric readings (e.g., depth cameras or [LiDAR](#) or [RADAR](#)), a direct formulation may likewise optimize a *geometric error*, but one that operates on the raw depth or range measurements themselves, as done in this thesis.

The conceptual elegance of direct methods, operating on raw sensor data without intermediate abstractions, suggests potential advantages in terms of accuracy, robustness, and computational efficiency. However, their adoption in the [SLAM](#) community has been limited, particularly for non-visual sensors. Several factors have historically hindered their widespread use:

- **Computational complexity:** The dense nature of photometric error functions traditionally required processing orders of magnitude more data than sparse feature-based approaches.
- **Convergence challenges:** Direct optimization over high-dimensional measurement spaces can be sensitive to initialization and prone to local minima.

This thesis challenges these preconceptions by demonstrating that modern computational resources, careful algorithm design, and a unified mathematical framework can unlock the full potential of direct methods across diverse sensor modalities. We show that what was once considered computationally prohibitive can now achieve real-time performance, and that the same core principles can elegantly adapt from cameras to [LiDARs](#) to [RADARs](#) with minimal modification.

1.3 Thesis Organization

This thesis is organized as follows

Chapter 2	It lays the mathematical and sensing foundations that underpin our direct-error SLAM framework. It begins by reviewing the three primary sensor modalities, cameras, LiDARs and RADARs , deriving their geometric projection models (pinhole for cameras, spherical for spinning LiDAR and polar for RADAR), and discussing each modality's strengths and limitations. Next, the chapter introduces Gaussian Process regression as a flexible tool to interpolate or smooth sensor measurements by placing a prior over functions and inferring continuous signal estimates from discrete, noisy samples. Finally, it covers the core principles of nonlinear least-squares optimization on manifolds, from first-order gradient methods to Gauss–Newton and Levenberg–Marquardt, and shows how Lie theory lets us optimize over pose manifolds, all wrapped in a factor-graph abstraction for efficient SLAM back-ends.
Chapter 3	It presents MD-SLAM, a unified direct-error SLAM pipeline that works seamlessly with both RGB-D and LiDAR data. It formulates tracking, loop closure and global optimization as the minimization of a dense, multi-cue direct error solved via a Gauss–Newton/Levenberg–Marquardt solver on $\mathbb{SE}(3)$. The system builds image pyramids for coarse-to-fine registration, uses appearance-based proposals with geometric verification for loop detection, and fuses all relative-pose constraints in a pose graph for global consistency. Experiments demonstrate sub-centimeter RGB-D accuracy and competitive LiDAR SLAM , with strong numerical stability across modalities.
Chapter 4	We introduce a versatile, large-scale robotics perception dataset recorded in Rome. It comprises high-resolution RGB imagery, 3D LiDAR point clouds, IMU readings and Real-Time Kinematic (RTK)-Global Navigation Satellite System (GNSS) poses. Unlike existing collections, often limited to narrow environments, simple motion patterns or lacking standardized benchmarks, ours is tailored for the latest SLAM and 3D reconstruction research. We address key challenges such as environmental diversity, dynamic motion profiles and high-frequency, cross-sensor synchronization. All sensors undergo rigorous intrinsic and extrinsic calibration; timestamps are aligned to within a few milliseconds. Crucially, we supply accurate 6-Degree-of-Freedom (DoF) ground truth by refining RTK-GNSS trajectories via multi-view LiDAR registration and bundle adjustment. The dataset, split into training and validation sequences.

Chapter 5	We present a lightweight extrinsic calibration method for accurately estimating the rigid-body transform between LiDAR and camera frames. Precise alignment is essential for fusing their data, yet most robotics workflows rely on large, expensive targets to compensate for LiDAR 's sparse returns. By contrast, our approach uses readily available, compact markers (e.g., A3 chessboards) and casts calibration as a least-squares plane-to-plane registration. In both simulation and real-world experiments, we achieve sub-centimeter and sub-degree accuracy, matching or surpassing state-of-the-art techniques that require specialized tags.
Chapter 6	We present a unified photometric bundle-adjustment framework that works seamlessly with both RGB-D and LiDAR data. By formulating a single, dense photometric cost over all sensor poses, our method refines SLAM - or GNSS -derived trajectories to maximize intensity consistency across frames, implicitly handling data association for each modality. In standalone tests, it matches or exceeds state-of-the-art (SOTA) camera-only and LiDAR -only optimizers; when precise extrinsic calibration is available, it further improves accuracy by jointly leveraging both sensors.
Chapter 7	It tackles motion-distortion in spinning 2D LiDAR scans by moving beyond a single rigid-body pose per sweep. Instead, it models the sensor's trajectory over each scan with continuous linear and angular velocities, then jointly estimates those motion parameters to "undistort" the point cloud. Integrated into a standard 2D LiDAR odometry pipeline, this approach demonstrably reduces drift and scan mismatch, and lays the groundwork for the full Doppler-aware RADAR odometry developed in Chapter 8.
Chapter 8	We introduce DRO, a breakthrough $\mathbb{SE}(2)$ odometry method for spinning Frequency-Modulated Continuous-Wave (FMCW) RADAR s that directly aligns raw intensity scans to a local map, with no feature extraction needed, and compensates for motion- and Doppler-induced distortions. When the RADAR 's modulation allows Doppler velocities to be observed, DRO adds a dedicated Doppler constraint, dramatically improving performance in feature-poor or adverse-weather scenarios. Implemented in real time on the GPU , it sets a new state of the art in RADAR odometry across hundreds of kilometers of urban and off-road data.

This thesis presents a unified methodology applied to several heterogeneous problems. Because of that, instead of providing a single *Related Work* Chapter, we report the bibliography per Chapter.

1.4 Publications

Part of this thesis has been published in peer-reviewed conference and journal articles:

- L. Di Giammarino, **L. Brizi**, T. Guadagnino, C. Stachniss, G. Grisetti. Md-slam: Multi-cue direct slam. *IEEE International Conference on Intelligent Robots and Systems (IROS)*, Kyoto, 2022. [67]
- L. Di Giammarino, E. Giacomini, **L. Brizi**, O. Salem, G. Grisetti. Photometric lidar and rgb-d bundle adjustment. *IEEE Robotics and Automation Letters (RAL)*, 2023. [44]
- O. Salem, E. Giacomini, **L. Brizi**, L. Di Giammarino, G. Grisetti. Enhancing LiDAR performance: Robust de-skewing exclusively relying on range measurements. *International Conference of the Italian Association for Artificial Intelligence*, 2023. [137]
- E. Giacomini*, **L. Brizi***, L. Di Giammarino, O. Salem, P. Perugini, G. Grisetti. *Ca²Lib*: Simple and Accurate L-RGB Calibration using Small Common Markers. *MDPI sensors*, 2024. [64]
*authors have contributed equally
- **L. Brizi***, E. Giacomini*, L. Di Giammarino, S. Ferrari, O. Salem, L. De Rebotti, G. Grisetti. VBR: A Vision Benchmark in Rome. *IEEE International Conference on Robotics and Automation (ICRA)*, Yokohama, 2024. [22]
*authors have contributed equally
- S. Ferrari, L. Di Giammarino, **L. Brizi**, G. Grisetti. MAD-ICP: It is all about matching data—robust and informed LiDAR odometry. *IEEE Robotics and Automation Letters (RAL)*, 2024. [51]
- C. Le Gentil, **L. Brizi**, D. Lisus, X. Qiao, G. Grisetti, T. D. Barfoot. DRO: Doppler-Aware Direct Radar Odometry. *Robotics: Science and Systems (RSS)*, 2025. [97]

Chapter 2

Fundamentals

IN autonomous robotics, sensors are fundamental for perceiving and interacting with the environment. The ability of a robot to map its surroundings, localize itself, and make decisions hinges critically on the quality and interpretation of sensor data. This chapter lays the groundwork by exploring the main sensor modalities used in robotics, cameras, [LiDARs](#), and [RADARs](#), highlighting their operating principles, advantages, and limitations. We emphasize the need for a 2D representation of sensor data to properly formulate the direct optimization problem central to this work. Additionally, the chapter covers foundational mathematical tools such as Gaussian Process (GP) regression and nonlinear optimization techniques that support advanced perception algorithms discussed throughout this thesis.

2.1 Sensors in Robotics

In robotics, sensors are the foundation of autonomous behavior. They provide the only means by which an agent can perceive and interact with its environment. All higher-level capabilities, *i.e.* mapping, navigation, control, and decision-making, ultimately depend on the quality and interpretation of the raw data acquired from onboard sensors. In the context of [SLAM](#), sensors allow a robot to infer both its position in the world and the structure of the surrounding environment. Yet, interpreting sensor data is not trivial: measurements are inherently noisy, limited in resolution, and subject to occlusions and ambiguities.

A key distinction in robotic perception lies in the *nature* of the sensors themselves, specifically, whether they are categorized as *active* or *passive*, based on whether they emit energy into the environment.

Passive sensors, such as monocular or stereo cameras, capture ambient signals (*e.g.*, light) reflected by objects. They are lightweight, low-cost, and capable of capturing rich appearance information, making them ideal for tasks that benefit from texture, color, and dense imagery. However, their performance is sensitive to lighting conditions, scene texture, and motion blur, and they do not directly provide metric depth unless supplemented with active illumination or stereo baselines.

Active sensors, on the other hand, emit structured energy, such as laser beams ([LiDAR](#)) or radio waves ([RADAR](#)), and measure the returned signal to infer range, reflectivity, or even velocity through Doppler shifts. These sensors are generally more robust to illumination changes and provide direct geometric information, but tend to be more expensive and power-

hungry. They can also suffer from multipath interference and complex signal reflections.

Each sensor type presents different trade-offs in terms of resolution, field of view, noise characteristics, and observability. Consequently, the choice and combination of sensors, as well as how their data is interpreted, whether through filtering or optimization, are central design decisions in robotic state estimation systems.

Among the most commonly used modalities in robotics are *cameras*, *LiDARs*, and *RADARs*, each with distinct sensing principles and characteristics that influence their suitability for various tasks. Understanding their geometry is essential to properly interpret measurements, project data into 2D or 3D space, and integrate them into state estimation pipelines.

Cameras provide dense 2D imagery rich in appearance features such as color and texture. They are low-cost, lightweight, and high-resolution, making them ideal for object detection, visual *SLAM*, and semantic scene understanding. However, they are sensitive to lighting conditions and lack inherent depth information unless used in stereo or active configurations.

LiDARs actively measure depth by emitting laser pulses and computing the time-of-flight of reflected signals. They offer accurate 3D geometric information and are largely unaffected by illumination, which makes them well suited for outdoor and large-scale mapping applications. Still, they come with higher cost and power demands, and their data is typically sparser and limited in vertical resolution. Notably, some modern *LiDARs* are capable of measuring radial velocity via Doppler shifts, offering motion information similar to *RADAR* systems.

RADAR also emits signals, but operates at radio frequencies. It is especially robust in challenging environments such as fog, rain, or darkness, and can directly measure radial velocity through Doppler shifts. While *RADAR* data is generally lower in resolution and noisier compared to *LiDAR* and vision, it provides critical redundancy and dynamic awareness in adverse conditions.

The choice of sensor depends on system requirements and environmental constraints. Cameras are preferred where appearance information is crucial and conditions are favorable. *LiDARs* offer precise 3D measurements for structure-rich environments, while *RADAR* provides robustness in low-visibility or high-speed scenarios. To support the direct error formulations discussed in this thesis, we now formalize the geometric models associated with each sensor type.

2.1.1 Camera

Cameras capture 2D projections of a 3D scene by recording light rays that enter the lens and strike the image sensor. Each pixel measures the intensity of light arriving from a particular direction. For simplicity, this directional information is modeled as a single ray originating at a point in 3D space and passing through the camera's center of projection. This mapping between image pixels and spatial rays is defined by the camera model, which depends on a set of intrinsic calibration parameters.

In practice, most robotic systems adopt simplified models that balance expressiveness with computational efficiency. The most widely used among them is the *pinhole camera model*, which assumes all light rays pass through a single point, known as the *optical center* or *pinhole*, before intersecting a planar image sensor. The sensor is positioned perpendicular to the optical axis, and the point where this axis meets the image plane is called the *principal point*.

Under this model, a camera is characterized by four intrinsic parameters: the focal lengths f_x and f_y , which relate to the lens's magnification in the horizontal and vertical directions, and the coordinates c_x and c_y of the principal point. These parameters are grouped into the camera calibration matrix:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.1)$$

The values f_x and f_y are typically equal when the sensor has square pixels, though older or specialized sensors may use rectangular pixels, resulting in distinct focal lengths.

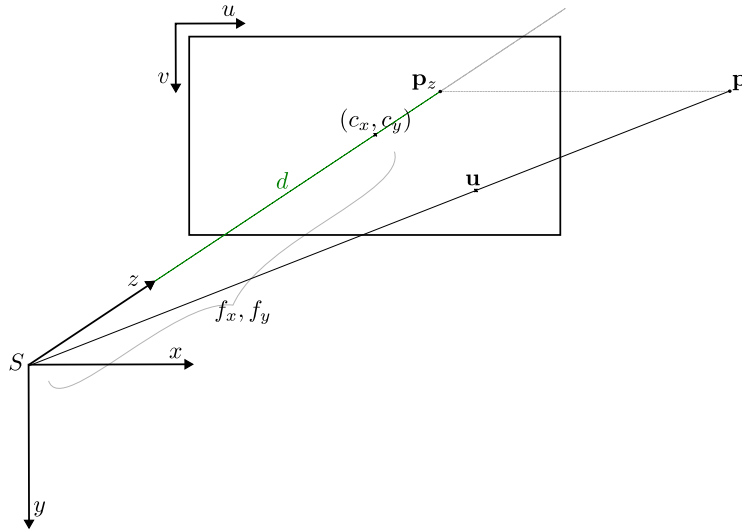


Figure 2.1. Illustration of the pinhole camera model. The sensor frame S (used instead of C to generalize the notation across sensors) is located at the optical center. The image plane is mirrored in front of the camera for clarity. Crosses indicate intersections with the image plane, while filled circles mark the corresponding 3D points in space. The depth d (in green) denotes the distance from the camera center to the 3D point along the z -axis. Illustration courtesy of Di Giammarino [43].

Projection

Given a 3D point $\mathbf{p} = [x, y, z]^\top$ expressed in the camera coordinate frame, the pinhole projection computes the corresponding pixel location $\mathbf{u} = [u, v]^\top$ as:

$$\pi_p(\mathbf{p}) = \phi(\mathbf{K} \mathbf{p}), \quad (2.2)$$

$$\phi(\mathbf{v}) = \frac{1}{v_z} \begin{bmatrix} v_x \\ v_y \end{bmatrix}, \quad \text{with } \mathbf{v} = \mathbf{K} \mathbf{p}. \quad (2.3)$$

The function ϕ performs homogeneous normalization by dividing the first two components by the depth z . This gives the 2D image coordinates corresponding to the 3D point. If $z \leq 0$, or if the resulting $\mathbf{u}$ falls outside the image dimensions, the point is not visible.

Inverse projection

The inverse projection recovers a 3D point from a 2D pixel $\mathbf{u}$ and a scalar depth d along the optical axis. This operation maps image coordinates back to 3D space as follows:

$$\mathbf{p} = d \cdot \mathbf{K}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}. \quad (2.4)$$

The result $\mathbf{p}$ is the unique 3D point lying along the ray associated with $\mathbf{u}$, at a depth d . If the depth is unknown, the inverse projection gives only the direction of the viewing ray, which can be normalized to obtain a unit vector.

Figure 2.1 illustrates the geometry of the pinhole model. The camera reference frame S is located at the optical center, with the z -axis pointing forward. For visualization purposes, the image plane is shown mirrored in front of the camera. The depth d is defined along the optical axis and corresponds to the distance from the pinhole to the 3D point.

2.1.2 LiDAR

LiDAR sensors estimate distances by emitting pulsed laser beams and measuring the time they take to reflect back from surrounding objects. These sensors are commonly categorized into two main types: *mechanical* and *solid-state* **LiDARs**.

Mechanical **LiDAR** systems utilize rotating components, such as servo motors, to sweep laser beams across the environment. In 3D **LiDARs**, the lasers are arranged in vertical stacks and the entire assembly rotates, enabling a full 360-degree field of view. This omnidirectional sensing capability is highly advantageous for applications requiring comprehensive environmental awareness, such as autonomous driving and 3D mapping. However, their moving parts introduce mechanical complexity, making them bulkier, more susceptible to wear, and often more costly.

Solid-state **LiDARs**, by contrast, have no moving parts. Instead, they use electronic beam-steering technologies, such as optical phased arrays or MEMS mirrors. These systems offer greater durability and a more compact form factor, making them ideal for embedded and rugged environments. Their field of view, however, tends to be narrower and less predictable due to their non-repetitive scanning patterns.

In this thesis, we concentrate on mechanical **LiDARs** due to their ability to generate consistent, panoramic and dense scans. These scans can be readily structured into 2D range images, thanks to their systematic vertical beam layout and rotational motion. Conversely, the data produced by solid-state **LiDARs** is more irregular, making it harder to interpret or image in a structured way.

Each laser in a **LiDAR** emits a pulse of light, and the sensor measures the round-trip time to calculate the distance to the reflecting surface. The return intensity $I_r(\rho, \lambda)$ received from an object depends on multiple factors and can be modeled as follows [131]:

$$I_r(\rho, \lambda) = I_0 \eta \frac{O}{4\pi\rho^2} \beta(\rho, \lambda) \exp\left(-2 \int_0^\rho \sigma(r, \lambda) dr\right) \quad (2.5)$$

Here, I_0 is the emitted intensity, η is the sensor's efficiency, O is the solid angle of the beam aperture, β represents surface reflectivity, and σ characterizes the medium's absorption.

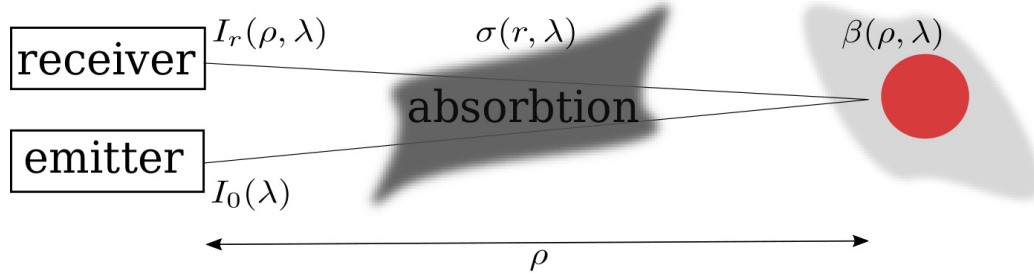


Figure 2.2. Single laser beam operation illustrating the measurement principle described in Eq. (2.5). Illustration courtesy of Di Giammarino [43].

Figure 2.2 visualizes this sensing principle. Reflectivity β is influenced by surface material, texture, moisture, and angle of incidence.

Range accuracy is primarily dictated by the resolution of the time-of-flight measurement. In high-precision systems, phase difference between emitted and received signals can also be analyzed. To reduce measurement noise, multiple pulses are typically emitted per direction, although this reduces acquisition frequency.

Mechanical constraints limit the rotational speed of the sensor. A 2D **LiDAR** typically uses a single rotating mirror, while a 3D system houses multiple lasers arranged vertically. Modern devices may include up to 128 channels.

Apart from range, **LiDARs** also measure the reflected intensity of each beam, often normalized and discretized to 8- or 16-bit values. The intensity information is the amount of laser radiation that reflect back from the object. While this intensity is useful for object identification, it is also sensitive to numerous variables. Nonetheless, all terms in Eq. (2.5) vary smoothly, implying that minor viewpoint changes yield minor variations in intensity.

Modern **LiDARs** generate dense point clouds that can be converted into structured images. A common method is to construct a panoramic image by arranging vertical beams horizontally. For instance, with 128 vertical beams and 1024 horizontal encoder steps, the resulting image has 128 rows and 1024 columns. This method is known as *projection by ID*. While it produces complete, hole-free images, the transformation is not differentiable, limiting its use in optimization-based state estimation.

To overcome this and maintain consistency with pinhole camera models, we use a *spherical projection* and its inverse, which map between 3D points and image coordinates in a differentiable manner.

Spherical Projection Model

The spherical projection maps 3D points onto a 2D spherical surface, with the reverse mapping reconstructing 3D coordinates from image pixels and range. Unlike the pinhole model, this transformation involves a conversion between Cartesian and spherical coordinates before flattening into the image space.

Projection. Let $\mathbf{p} \in \mathbb{R}^3$ be a point in **LiDAR** coordinates. We define a projection $\pi_s(\mathbf{p})$ via a mapping ψ from a 3D point to azimuth and elevation angles, followed by a camera matrix $\mathbf{K}$:

$$\pi_s(\mathbf{p}) = \mathbf{K}_{[1,2]} \psi(\mathbf{p}) \quad (2.6)$$

$$\psi(\mathbf{v}) = \begin{bmatrix} \text{atan2}(v_y, v_x) \\ \text{atan2}\left(v_z, \sqrt{v_x^2 + v_y^2}\right) \\ 1 \end{bmatrix} \quad (2.7)$$

Here, $\mathbf{K} \in \mathbb{R}^{2 \times 3}$ has the form used in Equation Eq. (2.1). We define its parameters, *i.e.* the azimuth and elevation resolution from LiDAR specifications:

$$\theta_{\text{res}} = \frac{\max h_{\text{FoV}} - \min h_{\text{FoV}}}{\text{rows}} \quad \phi_{\text{res}} = \frac{\max v_{\text{FoV}} - \min v_{\text{FoV}}}{\text{cols}} \quad (2.8)$$

where $\min v_{\text{FoV}}$ and $\max v_{\text{FoV}}$ are taken from the LiDAR datasheet, while $\min h_{\text{FoV}}$ and $\max h_{\text{FoV}}$ depends on the rotation done by the encoder (to acquire an omnidirectional point cloud, these are respectively 0 and 360 deg). The number of rows are equivalent to the number of beams vertically distributed, while the number of cols depends on the number of steps done by the encoder.

The focal lengths and principal points are simply parameterized by:

$$f_x = -\frac{1}{\theta_{\text{res}}} \quad c_x = \frac{\text{cols}}{2} \quad (2.9)$$

$$f_y = -\frac{1}{\phi_{\text{res}}} \quad c_y = \frac{\text{rows}}{2} \quad (2.10)$$

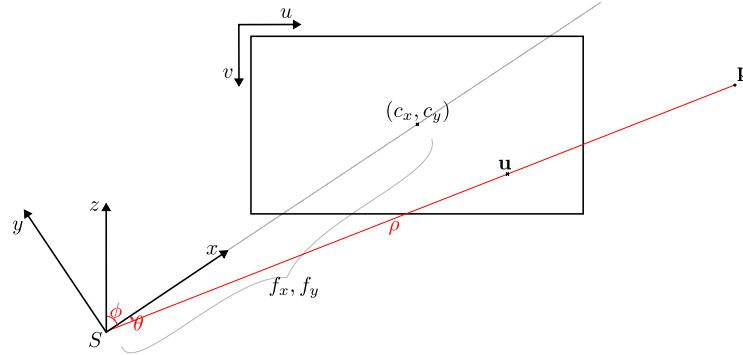


Figure 2.3. Spherical projection illustration. The LiDAR frame typically uses a z -up convention. The red lines represent spherical coordinates: azimuth θ , elevation ϕ , and range ρ . Illustration courtesy of Di Giammarino [43].

Inverse Projection. Given a pixel coordinate $\mathbf{u} = [u, v]^\top$ and a range ρ , we compute the corresponding 3D point $\mathbf{p}$ as:

$$\mathbf{p} = \rho \cdot \psi^{-1} \left(\mathbf{K}_{[1,2]}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \right) \quad (2.11)$$

where ψ^{-1} converts spherical coordinates (θ, ϕ) back to Cartesian:

$$\psi^{-1}(\theta, \phi) = \begin{bmatrix} \cos \theta \cos \phi \\ \sin \theta \cos \phi \\ \sin \phi \end{bmatrix} \quad (2.12)$$

Here, ρ denotes the range from the sensor origin to the point (see Figure 2.3). Unlike camera depth, which is defined along the z -axis, range is the full Euclidean distance.

Projection by ID

In the previous section, we discussed a differentiable spherical projection model. However, for tasks such as visualization or learning from image-like data (as for the work described in Chapter 5), we often use a discrete indexing strategy known as *projection by ID*.

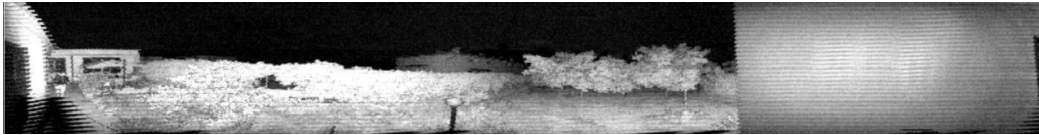
Let $\mathbf{p}$ be a 3D point. We define its projection as:

$$\pi_{\text{id}}(\mathbf{p}) = \mathbf{A}\psi(\mathbf{p}) \quad (2.13)$$

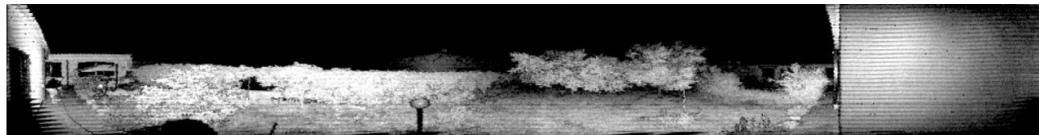
$$\mathbf{A} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & 1 & 0 \end{bmatrix} \quad (2.14)$$

$$\psi(v) = \begin{bmatrix} \text{atan2}(v_y, v_x) \\ \text{ring}(v) \\ 1 \end{bmatrix} \quad (2.15)$$

Here, the $\text{ring}(v)$ index indicates which vertical laser fired the point. This index is either provided by the sensor or inferred from the data ordering. Unlike the spherical model, projection by ID lacks differentiability and distorts geometric structure. Nevertheless, it provides clean, hole-free images useful for downstream tasks. The inverse mapping can be implemented with a simple lookup table.



(a) Projection by ID. The image is visually complete but introduces distortions and is non-differentiable.



(b) Spherical projection. Preserves geometry and differentiability, though it may contain holes due to calibration error.

Figure 2.4. Comparison between projection methods: ID-based (top) and spherical (bottom).

Depending on which quantity associated with each point, such as range, reflectance intensity, or estimated surface normals, is used to populate the corresponding image pixels, different types of **LiDAR** images can be generated. These images highlight various properties

of the scene and are useful for different downstream applications such as segmentation, registration, or surface analysis. A visible comparison between the two projections is shown in Fig. 2.4.

2.1.3 RADAR

Let us consider the type of **RADARs** commonly used in the automotive and robotic industries: **FMCW RADAR**. As their name suggests, these **RADARs** continuously emit Frequency-Modulated Electromagnetic Waves. Operating typically in the millimeter-wave bands, **FMCW RADARs** transmit radio waves whose frequency is modulated over time, enabling simultaneous measurement of both range and relative velocity of reflecting objects.

The sensor emits electromagnetic waves with a frequency that changes linearly over time in a chirp-like pattern. When these waves encounter an object, they reflect back to the sensor with a time delay proportional to the object's distance. By comparing the transmitted and received signals, the **RADAR** extracts the frequency difference, which encodes the distance information. This mechanism differs fundamentally from **LiDAR**, which relies on measuring the time-of-flight of light pulses. The longer wavelengths of **RADAR** waves allow for greater penetration through adverse weather conditions such as fog, rain, or dust, making **FMCW RADARs** highly valuable for robust sensing in automotive and robotic applications.

Thus, instead of directly measuring the time between the emission and reception of waves, the distance to an electromagnetic reflector corresponds to a frequency difference Δf between the signal transmitted and received. For the sake of simplicity, the following explanation will consider a single reflector at a certain distance r from the **RADAR**. In reality, the **FMCW RADARs** provide a reflection score/value for many 'range bins' along the emission beam.

Considering linear modulation of the frequency with a slope s as shown in Fig. 2.6, the distance r between the **RADAR** and reflector is

$$\tilde{r} = \frac{c\Delta f}{2s} \quad (2.16)$$

where c is the speed of light, and the divider 2 accounts for the fact that Δf represents twice the distance r due to the time for the wave to reach the reflector and return.

Projection

As with other sensors, it is useful to define a mapping between the raw **RADAR** data format, which is naturally represented in polar coordinates, and Cartesian coordinates for geometric interpretation and fusion with other sensors.

The **RADAR** data is typically collected in scans covering the full 360° field-of-view around the sensor. Each scan consists of N azimuthal beams indexed by n , each associated with an azimuth angle α_n and a timestamp t_n . Along each beam, the sensor records electromagnetic reflection intensity values ψ_{nm} at M discrete range bins indexed by m , corresponding to distances $\hat{r}_m$ from the sensor. Hence, a **RADAR** scan can be interpreted as a two-dimensional polar intensity image or equivalently as a set of tuples $\{\alpha_n, \hat{r}_m, \psi_{nm}\}$.

Let $\mathbf{p} = [p_x, p_y]^\top \in \mathbb{R}^2$ be a point in the **RADAR** coordinate frame. We define a projection function $\pi_s(\mathbf{p})$ that maps this 2D Cartesian point to polar **RADAR** image

coordinates by applying a transformation π from Cartesian to polar, followed by a linear scaling with matrix $\mathbf{K}$:

$$\pi_s(\mathbf{p}) = \mathbf{K} \pi(\mathbf{p}), \quad (2.17)$$

$$\pi(\mathbf{p}) = \begin{bmatrix} \text{atan2}(p_y, p_x) \\ \sqrt{p_x^2 + p_y^2} \end{bmatrix}. \quad (2.18)$$

The matrix $\mathbf{K} \in \mathbb{R}^{2 \times 2}$ encodes the **RADAR**'s angular and range resolution, derived from its datasheet specifications:

$$\mathbf{K} = \begin{bmatrix} \frac{1}{\alpha_{res}} & 0 \\ 0 & \frac{1}{\hat{r}_{res}} \end{bmatrix}, \quad (2.19)$$

where α_{res} is the azimuth angular resolution (in radians per pixel) and $\hat{r}_{res}$ is the range resolution (meters per pixel). These parameters determine the discretization of the **RADAR** image along the angular and range dimensions respectively.

This mapping corresponds to a straightforward polar coordinate transformation combined with a normalization step to convert the physical angular and range values into pixel indices within the **RADAR** intensity image.

Inverse projection

Conversely, given a pixel coordinate $\mathbf{u} = [u, v]^\top$ in the **RADAR** image, we can recover the corresponding point in Cartesian coordinates by inverting the projection:

$$\mathbf{p} = \pi^{-1}(\mathbf{K}^{-1}\mathbf{u}), \quad (2.20)$$

where π^{-1} converts from polar coordinates $(\alpha, \hat{r})$ back to Cartesian coordinates as

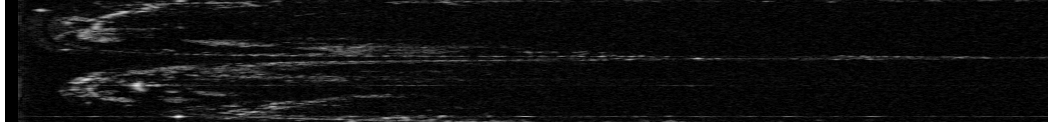
$$\pi^{-1}(\alpha, \hat{r}) = \begin{bmatrix} \hat{r} \cos \alpha \\ \hat{r} \sin \alpha \end{bmatrix}. \quad (2.21)$$

This inverse mapping enables the interpretation of **RADAR** pixel data in spatial terms, facilitating sensor fusion with other modalities (such as **LiDAR** or cameras) and allowing geometric reasoning about the scene (see Fig. 2.5).

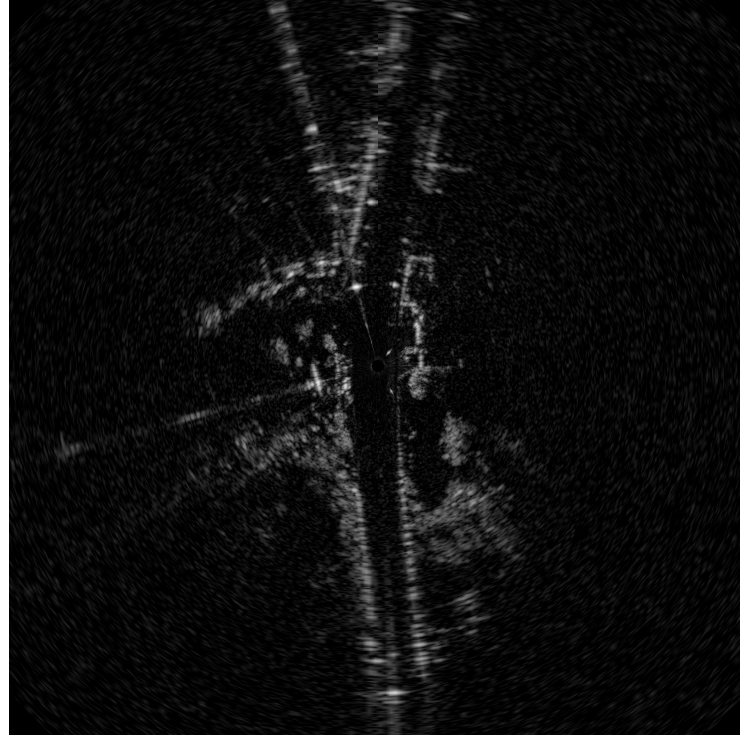
Doppler Effect in RADAR Measurements

This relationship in Eq. (2.16) is only true ($\tilde{r} = r$) if the relative velocity between the sensor and the reflector is null. In the presence of relative motion, the measured Δf is impacted by the Doppler effect that compresses or expands the electromagnetic wave, thus resulting in $\Delta f = \Delta f_t + \Delta f_d$, with Δf_t being the shift corresponding to the waves travelling time, and Δf_d the Doppler-induced frequency shift. This shift Δf_d is proportional to the signal frequency $\frac{1}{\lambda_t}$ and the velocity along the emission beam, denoted 'radial velocity' u , as

$$\Delta f_d = \frac{2u}{\lambda_t}. \quad (2.22)$$



(a) Raw **RADAR** returns plotted in polar coordinates, with azimuth on the y-axis and range on the x-axis.



(b) Corresponding Cartesian representation after applying the inverse projection Eq. (2.20).

Figure 2.5. An example of **RADAR** measurement: polar representation (top) and corresponding Cartesian (bottom).

Accordingly, the ‘range’ measurement provided by Eq. (2.16) is perturbed by the Doppler effect

$$\tilde{r} = \frac{c\Delta f_t}{2s} + \frac{c\Delta f_d}{2s} = r + r_d(u). \quad (2.23)$$

When considering a frequency modulation with a *sawtooth* pattern and multiple measurements of the same reflector, we cannot dissociate Δf_t and Δf_d from Δf .

However, if we use a triangle pattern consisting of a succession of ‘up-chirps’ and ‘down-chirps’ as illustrated in Fig. 2.6, two measurements of the same reflector allow for the recovery of the radial velocity:

$$\tilde{r}_\uparrow - \tilde{r}_\downarrow = \frac{c\Delta f_t}{2s} + \frac{c\Delta f_d}{2s} - \frac{-c\Delta f_t}{-2s} - \frac{c\Delta f_d}{-2s} \quad (2.24)$$

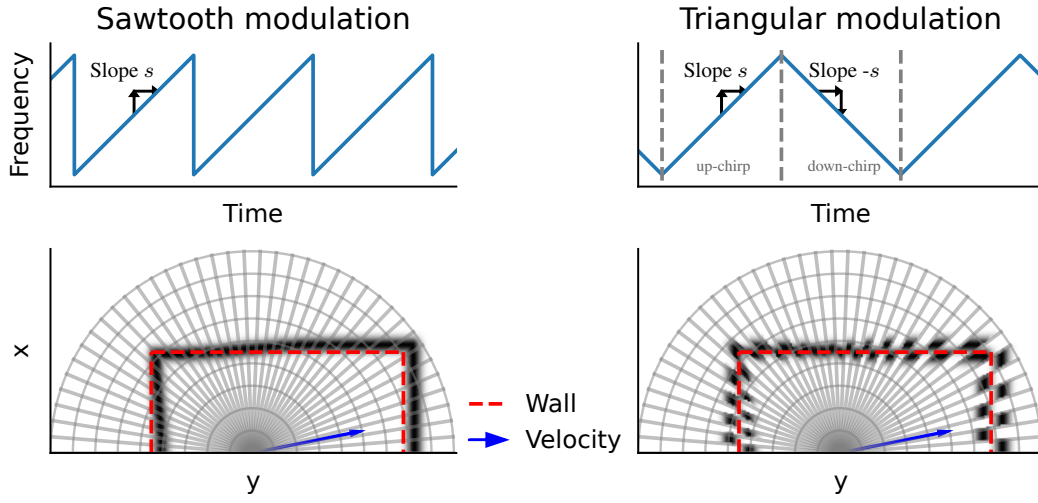


Figure 2.6. Impact of different frequency modulation patterns (top) and Doppler effect on the data collected with spinning FMCW radars (bottom).

where the components in Δf_t cancel out resulting in

$$\Delta \hat{r} = \tilde{r}_{\uparrow} - \tilde{r}_{\downarrow} = \frac{2uc}{s\lambda_y} = \beta u. \quad (2.25)$$

In reality, FMCW RADARs do not only expect one return per beam but collect ‘reflectivity’ information through a wide range of discrete ranges through a fast-Fourier transform of the received signal. Thus, the data from such RADARs can be interpreted as intensity/reflectivity images, in which rows correspond to different beam azimuths and columns to different ranges. Fig. 2.6 illustrates the effect of the Doppler effect on FMCW data. The data collected with a moving RADAR using a triangular pattern presents an alternating shift between each row of the scans, while all the rows are shifted according to the sign of the radial velocity for sawtooth RADARs. Depending on the RADAR model, the modulation pattern can be changed or not in the firmware. Thus, not all spinning FMCW RADARs enable the extraction of Doppler-based velocities.

2.2 Gaussian Process Regression

Gaussian Process regression [130] is a probabilistic, non-parametric interpolation method. Let us consider a signal $h(\mathbf{x})$ over an input space $\mathbf{x} \in \mathbb{R}^D$, modeled with a GP $h(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ with zero mean and the covariance function $k(\mathbf{x}, \mathbf{x}')$. The latest characterizes the covariance between two instances of h (at $\mathbf{x}$ and $\mathbf{x}'$). Given noisy samples $y_i = h(\mathbf{x}) + \eta$, with $\eta \sim \mathcal{N}(0, \sigma)$ and $i = (1, \dots, N)$, the GP regression aims at inferring the signal h and its variance at any new input location $\mathbf{x}^*$. The definition of a GP corresponds to the following multivariate normal distribution:

$$\begin{bmatrix} \mathbf{y} \\ h(\mathbf{x}^*) \end{bmatrix} \sim \mathcal{N} \left(\begin{bmatrix} \mathbf{0} \\ 0 \end{bmatrix}, \begin{bmatrix} \mathbf{K}_{\mathbf{X}\mathbf{X}} + \sigma^2 \mathbf{I} & \mathbf{k}_{\mathbf{X}\mathbf{x}^*} \\ \mathbf{k}_{\mathbf{x}^*\mathbf{X}} & k(\mathbf{x}^*, \mathbf{x}^*) \end{bmatrix} \right), \quad (2.26)$$

where $\mathbf{k}_{\mathbf{x}^*\mathbf{X}} = (\mathbf{k}_{\mathbf{X}\mathbf{x}^*})^\top = \begin{bmatrix} k(\mathbf{x}^*, \mathbf{x}_1) & \cdots & k(\mathbf{x}^*, \mathbf{x}_N) \end{bmatrix}$, $\mathbf{K}_{\mathbf{X}\mathbf{X}} = \begin{bmatrix} \mathbf{k}_{\mathbf{X}\mathbf{x}_1} & \cdots & \mathbf{k}_{\mathbf{X}\mathbf{x}_N} \end{bmatrix}$, and $\mathbf{y} = \begin{bmatrix} y_1 & \cdots & y_N \end{bmatrix}$.

By conditioning Eq. (2.26) with respect to the noisy observations we obtain the inference of h at any $\mathbf{x}^*$ and the covariance as

$$\begin{aligned} \hat{h}(\mathbf{x}^*) &= \mathbf{k}_{\mathbf{x}^*\mathbf{X}} \left(\mathbf{K}_{\mathbf{X}\mathbf{X}} + \sigma^2 \mathbf{I} \right)^{-1} \mathbf{y}, \\ \text{var} \left(\hat{h}(\mathbf{x}^*) \right) &= k(\mathbf{x}^*, \mathbf{x}^*) - \mathbf{k}_{\mathbf{x}^*\mathbf{X}} \left(\mathbf{K}_{\mathbf{X}\mathbf{X}} + \sigma^2 \mathbf{I} \right)^{-1} \mathbf{k}_{\mathbf{X}\mathbf{x}^*}. \end{aligned} \quad (2.27)$$

2.3 Non-Linear Optimization

State estimation in robotics often involves nonlinear problems, arising naturally from sensor measurements, robot motions, and environmental interactions. The robotics community typically addresses these challenges through two main paradigms: *filter-based* and *optimization-based* approaches. These paradigms fundamentally differ in how they represent and update the robot's internal state and its relationship with the external world.

Filtering-based approaches rely on recursively maintaining and updating a probabilistic belief about the robot's current state. This belief is commonly expressed as a joint probability distribution over latent variables, such as the robot's pose and relevant environmental features. The Bayes filter framework underpins these methods, performing two principal operations: a *prediction step*, which introduces uncertainty based on the robot's motion model, and a *correction step*, which refines the belief using sensor measurements. Central to this method is the Markov property, meaning future states depend solely on the current state, ignoring past states.

A prevalent filtering technique is the Extended Kalman Filter (EKF), which approximates the system's belief as a multivariate Gaussian, propagating both mean and covariance through linearized motion and measurement models. Despite its elegance and computational efficiency, the EKF is limited by its linear approximation and can degrade significantly under highly nonlinear conditions.

In contrast, *optimization-based approaches* reframe state estimation as a Maximum Likelihood Estimation (MLE) problem. Given a probabilistic measurement model conditioned on the robot's state, the goal is to find the state that maximizes the likelihood of the observed data, potentially involving the entire robot trajectory. Equivalently, this is posed as minimizing a negative log-likelihood, forming a Non-Linear Least-Squares (NLS) optimization task. Such methods explicitly model sensor measurements and noise characteristics, accommodating delayed measurements, re-linearization, and iterative refinement as new data arrives. The optimization is generally performed over a nonlinear cost function. Common optimization algorithms include [Gradient Descend](#), [Gauss-Newton](#), and [Levenberg-Marquardt](#).

Throughout subsequent chapters, we consistently address situations where closed-form solutions to estimate unknown state variables $\mathbf{x}$ from noisy and incomplete measurements $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^M$ are infeasible. Therefore, we define a unified and robust estimation approach using a cost function $E(\mathbf{x}, \mathbf{Z})$ to quantify how well a state hypothesis explains observed data. This cost function typically takes the following general form:

$$E(\mathbf{x}, \mathbf{Z}) = \sum_k \|\mathbf{e}_k(\mathbf{x})\|_{\Omega}^2, \quad (2.28)$$

where $\mathbf{e}_k(\mathbf{x})$ denotes the residual error of the k -th measurement, and Ω represents the covariance matrix weighting each residual. To enable effective numerical optimization, the residuals $\mathbf{e}_k(\mathbf{x})$ must be minimal at the optimal state and continuously differentiable.

The following subsections explore various optimization strategies, beginning from simpler, intuitive methods and progressing toward advanced techniques leveraging manifold and Lie group structures intrinsic to robotic states.

2.3.1 First-Order Methods

First-order iterative optimization methods extensively used in nonlinear optimization are based on following the gradient direction to maximize or minimize a function. **Gradient Descend** is the most fundamental of these methods. The idea is to take repeated steps in the opposite direction of the gradient (or approximate gradient) of the function at the current point because this is the direction of steepest descent. Conversely, stepping in the direction of the gradient will lead to a trajectory that maximizes that function; the procedure is then known as Gradient Ascent.

Formally, given a differentiable cost function $E(\mathbf{x})$, the iterative update rule for gradient descent is:

$$\mathbf{x}' = \mathbf{x} - \eta \nabla E(\mathbf{x}), \quad (2.29)$$

where η is a positive step size (learning rate), and $\nabla E(\mathbf{x})$ is the gradient of the cost function with respect to $\mathbf{x}$. If the step size η is sufficiently small and under suitable conditions (convexity of the cost function and Lipschitz continuity of its gradient), gradient descent guarantees convergence to a local minimum.

Selecting the step size η is critical: if η is too large, the algorithm may overshoot the minimum and fail to converge; if too small, convergence can be unnecessarily slow. Advanced variants of gradient descent, such as adaptive gradient methods (e.g., AdaGrad, RMSProp, Adam), dynamically adjust the step size to balance convergence speed and stability, effectively handling challenging optimization landscapes.

Moreover, Gradient Descend (GD) is particularly advantageous when dealing with high-dimensional problems or cases where computational simplicity is paramount. Unlike second-order methods (such as Gauss-Newton (GN) or Levenberg-Marquardt (LM)), **GD** does not require the computation or inversion of the Hessian matrix, making it computationally efficient and easier to implement, especially for large-scale optimization problems.

In certain scenarios, such as correlation maximization between signals, formulating the problem as a least-squares minimization, as in **GN** optimization, is not straightforward or even possible. This occurs because correlation-based objectives inherently maximize similarity measures rather than minimizing squared residuals. Hence, gradient-based methods like **GD** become indispensable, as they directly handle arbitrary differentiable objective functions without the necessity of a quadratic approximation inherent to least-squares methods. They are known as first-order methods because, at any point where the gradient is computed, the function is approximated as a linear function. Conversely, in **GN** optimization, the function is approximated by a quadratic function.

2.3.2 Gauss-Newton Optimization

The **Gauss-Newton** method is an iterative optimization strategy particularly suited for minimizing nonlinear cost functions, such as those encountered in state estimation problems. Unlike **GD**, which often demands meticulous tuning of the step size to avoid local minima traps, **GN** leverages second-order information, providing more stable and typically faster convergence.

To minimize a cost function $E(\mathbf{x})$, **GN** iteratively refines an initial state estimate $\mathbf{x}$ by determining small increments $\Delta\mathbf{x}$. At each iteration, it approximates the residuals using a first-order Taylor expansion around the current estimate $\mathbf{x}$:

$$\mathbf{e}_k(\mathbf{x} + \Delta\mathbf{x}) \approx \mathbf{e}_k(\mathbf{x}) + \mathbf{J}_k \Delta\mathbf{x}, \quad (2.30)$$

where $\mathbf{J}_k$ represents the Jacobian matrix of the residual $\mathbf{e}_k(\mathbf{x})$ at the current state estimate.

Incorporating this linear approximation into the cost function in Eq. (2.28) gives us:

$$E(\mathbf{x} + \Delta\mathbf{x}) = \sum_k |\mathbf{e}_k(\mathbf{x} + \Delta\mathbf{x})|_{\Omega_k}^2 \quad (2.31)$$

$$\approx \sum_k |\mathbf{e}_k(\mathbf{x}) + \mathbf{J}_k \Delta\mathbf{x}|_{\Omega_k}^2 \quad (2.32)$$

$$= \sum_k \left(\mathbf{e}_k(\mathbf{x})^T \Omega_k \mathbf{e}_k(\mathbf{x}) + 2\mathbf{e}_k(\mathbf{x})^T \Omega_k \mathbf{J}_k \Delta\mathbf{x} + \Delta\mathbf{x}^T \mathbf{J}_k^T \Omega_k \mathbf{J}_k \Delta\mathbf{x} \right). \quad (2.33)$$

By reorganizing terms, we introduce the standard notation used in nonlinear optimization:

$$\mathbf{c} = \sum_k \mathbf{e}_k(\mathbf{x})^T \Omega_k \mathbf{e}_k(\mathbf{x}), \quad (2.34)$$

$$\mathbf{b} = \sum_k \mathbf{J}_k^T \Omega_k \mathbf{e}_k(\mathbf{x}), \quad (2.35)$$

$$\mathbf{H} = \sum_k \mathbf{J}_k^T \Omega_k \mathbf{J}_k, \quad (2.36)$$

where $\mathbf{b}$ is half the gradient of the cost function at the current estimate, and $\mathbf{H}$ is the **GN** approximation to the Hessian matrix. Thus, the approximated cost can be strictly expressed as a quadratic function in $\Delta\mathbf{x}$:

$$E(\mathbf{x} + \Delta\mathbf{x}) \approx \mathbf{c} + 2\mathbf{b}^T \Delta\mathbf{x} + \Delta\mathbf{x}^T \mathbf{H} \Delta\mathbf{x}. \quad (2.37)$$

To Minimize this quadratic approximation requires taking the derivative with respect to $\Delta\mathbf{x}$ and setting it to zero, since it is a quadratic function. This is yielding the linear system:

$$\mathbf{H} \Delta\mathbf{x} = -\mathbf{b}, \quad (2.38)$$

which is solved to update the state:

$$\Delta\mathbf{x} = -\mathbf{H}^{-1} \mathbf{b}, \quad \mathbf{x} \leftarrow \mathbf{x} + \Delta\mathbf{x}. \quad (2.39)$$

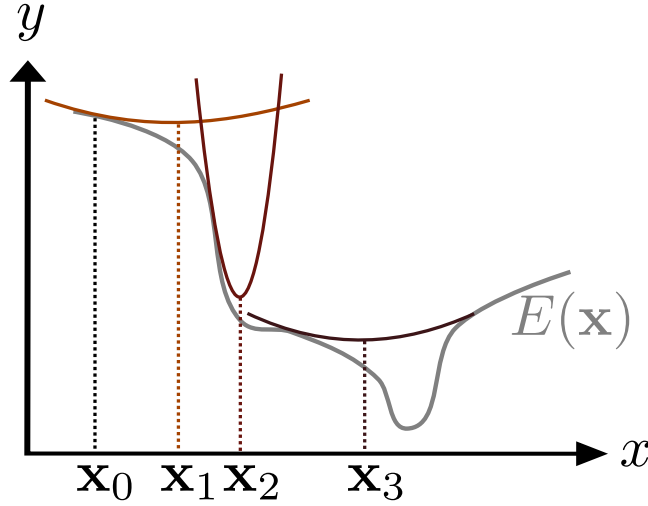


Figure 2.7. Visual illustration of the GN algorithm. Starting from an initial state $\mathbf{x}_0$, the cost function (gray curve) is approximated locally by a quadratic function (orange curve), represented as a parabola in one dimension. The updated state $\mathbf{x}_1$ is computed by adding the increment $\Delta \mathbf{x}$ to the previous state, and the procedure is iterated until convergence at the final state $\mathbf{x}_3$. Illustration courtesy of Di Giammarino [43].

Iteratively applying these updates refines the solution toward a stationary point (typically a local minimum) of the nonlinear cost function. The quality of GN convergence depends crucially on the accuracy of the linear approximation of residuals. Specifically, the GN approximation to the Hessian neglects second-order derivative terms involving residual curvature:

$$\nabla^2 E(\mathbf{x}) = 2 \sum_k \left(\mathbf{J}_k^T \boldsymbol{\Omega}_k \mathbf{J}_k + \sum_j (\mathbf{e}_k(\mathbf{x}))_j \nabla^2 (\mathbf{e}_k(\mathbf{x}))_j \right). \quad (2.40)$$

This approximation remains effective as long as residuals remain small and approximately linear near the optimum. Residuals that behave nearly linearly around their minima reduce the contribution of these neglected second-order terms, thus facilitating better convergence.

The iterative process of the GN method, emphasizing the successive quadratic approximations of the cost function and their respective minimizations, is visually represented in Fig. 2.7.

Iterative Reweighted Least Squares Sensor noise, mismatches, and dynamic outliers can significantly skew the solution if each residual is treated equally. Iterative Reweighted Least-Squares (IRLS) mitigates this by adapting the weight of each residual in proportion to its magnitude. A robust loss function $\rho(\cdot)$ is chosen, such as Huber or Tukey, to limit the

influence of large errors. At each iteration, residuals are scaled by:

$$\gamma_k(\mathbf{x}) = \frac{\rho'(|\mathbf{e}_k(\mathbf{x})| \Omega_k)}{|\mathbf{e}_k(\mathbf{x})| \Omega_k}, \quad (2.41)$$

and the modified information matrix becomes $\bar{\Omega}_k(\mathbf{x}) = \gamma_k(\mathbf{x}) \Omega_k$. This reweighting naturally integrates into the GN normal equations:

$$\mathbf{H}(\mathbf{x}) = \sum_k \mathbf{J}_k^T \bar{\Omega}_k(\mathbf{x}) \mathbf{J}_k, \quad \mathbf{b}(\mathbf{x}) = \sum_k \mathbf{J}_k^T \bar{\Omega}_k(\mathbf{x}) \mathbf{e}_k(\mathbf{x}). \quad (2.42)$$

By iterating weights and updates, IRLS delivers a solution that is robust to outliers while retaining the fast convergence properties of GN.

Levenberg-Marquardt Method Although GN can converge rapidly near the optimum, it may produce unstable or divergent steps in highly nonlinear regions. The LM algorithm interpolates between GN and GD by introducing a damping parameter $\lambda \geq 0$. The update rule becomes:

$$\Delta \mathbf{x} = -(\mathbf{H} + \lambda \mathbf{I})^{-1} \mathbf{b}, \quad (2.43)$$

For small λ , LM behaves like GN; as λ increases, it approaches a scaled gradient descent step. The update becomes $\Delta \mathbf{x} \approx -(\lambda \mathbf{I})^{-1} \mathbf{b} = -1/\lambda \mathbf{b}$. The damping factor is adjusted dynamically: if a step reduces the cost, λ is decreased (e.g., divided by 10); if the cost increases, λ is increased (e.g., multiplied by 10). This adaptive strategy ensures consistent descent while capitalizing on GN's fast convergence when the linear model is valid.

2.3.3 Lie Theory and Optimization on Manifold

Up to this point we have treated the state vector as if it lived in a Euclidean space. In many robotics applications, however, the state comprises rotation matrices in $\mathbb{SO}(3)$ or unit-length vectors on the sphere $\mathbb{S}^2$, both of which reside on curved manifolds rather than flat vector spaces. A manifold is a space that looks Euclidean only in an infinitesimal neighbourhood around every point. Consequently, during optimization we must ensure that algorithms such as LM, GN or GD update the state without letting it drift off its manifold. The present section reviews the mathematical tools that integrate manifold structure into Iterative Least-Squares (ILS) optimization routines.

A Lie group $\mathcal{G}$ is a smooth (differentiable) manifold $\mathcal{M}$ whose elements also form a group under some binary operation $\circ$. The group axioms are:

- **Closure:** Combining two elements with $\circ$ yields another element of the set.
- **Identity:** There exists a neutral element that leaves every element unchanged.
- **Inverse:** Each element possesses an inverse that returns the identity when composed.
- **Associativity:** The order in which three elements are grouped does not affect the result.

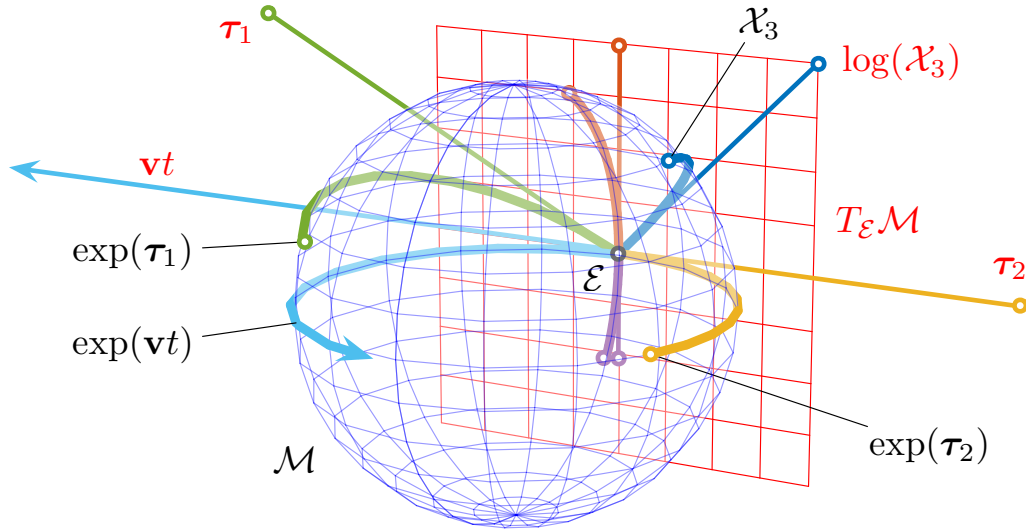


Figure 2.8. Relationship between a Lie group and its Lie algebra. The Lie algebra $\mathfrak{m} = T_{\mathcal{E}}\mathcal{M}$ (red plane) is the tangent space to the group manifold $\mathcal{M}$ (blue sphere) at the identity element $\mathcal{E}$. A straight line $\mathbf{v}t$ in the Lie algebra is mapped to a geodesic $\exp(\mathbf{v}t)$ on the manifold via the exponential map, establishing a one-to-one correspondence between the two. This link lets us perform calculations in the flat algebra and transfer the results back to the nonlinear group. The S^2 sphere sketched in $\mathbb{R}^3$ is a 2-D drawing aid; the actual Lie group is the 3-sphere $S^3 \subset \mathbb{R}^4$ that represents unit quaternions. Illustration courtesy of J. Solà [146].

Because $\mathcal{M}$ is differentiable, each point admits a unique tangent space; at the identity this space is called the *Lie algebra* $\mathfrak{m}$. The Lie algebra is a vector space and can therefore be identified with $\mathbb{R}^m$, where m equals the degrees of freedom of $\mathcal{M}$. To relate a Cartesian vector $\tau \in \mathbb{R}^M$ to an element of the Lie algebra $T_{\mathcal{E}}\mathcal{M}$, we define the *Hat* and *Vee* isomorphisms

$$\begin{aligned} \text{Hat} : \mathbb{R}^M &\rightarrow \mathfrak{m}; & \tau &\rightarrow \tau^\wedge = \sum_{i=1}^M \tau_i E_i \\ \text{Vee} : \mathfrak{m} &\rightarrow \mathbb{R}^M; & \tau^\wedge &\rightarrow (\tau^\wedge)^\vee = \tau = \sum_{i=1}^M \tau_i \mathbf{e}_i, \end{aligned} \quad (2.44)$$

where E_i are the Lie algebra generators and $\mathbf{e}_i$ their vector counterparts in $\mathbb{R}^M$ (so that $\mathbf{e}_i^\wedge = E_i$).

Let $\exp : \mathbb{R}^M \rightarrow \mathcal{M}$ map Lie algebra elements to the group and $\log : \mathcal{M} \rightarrow \mathbb{R}^M$ be its inverse:

$$\mathcal{X} \triangleq \exp(\tau^\wedge) \quad (2.45)$$

$$\tau \triangleq \log(\mathcal{X})^\vee. \quad (2.46)$$

The exponential map, also called a *retraction*, “wraps” the tangent vector onto the manifold along a geodesic, as sketched in Fig. 2.8.

For convenience we use capitalized operators that act directly on vectors $\tau \in \mathbb{R}^m (\cong T_{\mathcal{E}}\mathcal{M})$ and group elements $\mathcal{X} \in \mathcal{M}$:

$$\text{Exp} : \mathbb{R}^m \rightarrow \mathcal{M}; \quad \tau \mapsto \mathcal{X} = \text{Exp}(\tau) \quad (2.47)$$

$$\text{Log} : \mathcal{M} \rightarrow \mathbb{R}^m; \quad \mathcal{X} \mapsto \tau = \text{Log}(\mathcal{X}) \quad (2.48)$$

We now introduce manifold addition and subtraction via the right- and left-operators $\boxplus$ and $\boxminus$. Each is a composition of Exp/Log with the group operation, so their definitions depend on whether the update is applied from the right or from the left. The right-hand versions are

$$\text{right-}\boxplus : \quad \mathcal{Y} = \mathcal{X} \boxplus \tau \triangleq \mathcal{X} \text{Exp}(\tau) \in \mathcal{M} \quad (2.49)$$

$$\text{right-}\boxminus : \quad \tau = \mathcal{Y} \boxminus \mathcal{X} \triangleq \text{Log}(\mathcal{X}^{-1} \circ \mathcal{Y}) \in \mathbb{R}^M, \quad (2.50)$$

while the left-hand versions read

$$\text{left-}\boxplus : \quad \mathcal{Y} = \tau \boxplus \mathcal{X} \triangleq \text{Exp}(\tau) \mathcal{X} \in \mathcal{M} \quad (2.51)$$

$$\text{left-}\boxminus : \quad \tau = \mathcal{Y} \boxminus \mathcal{X} \triangleq \text{Log}(\mathcal{Y}^{-1} \circ \mathcal{X}) \in \mathbb{R}^M. \quad (2.52)$$

See [71, 146] for exhaustive treatments.

With these tools the Gauss–Newton linearization in Eq. (2.30) extends to a manifold-valued estimate $\hat{\mathbf{X}} \in \mathcal{M}$ as

$$\mathbf{e}_k(\hat{\mathbf{X}} \boxplus \Delta \mathbf{x}) \approx \mathbf{e}_k(\hat{\mathbf{X}}) + \left. \frac{\partial \mathbf{e}_k(\hat{\mathbf{X}} \boxplus \Delta \mathbf{x})}{\partial \Delta \mathbf{x}} \right|_{\Delta \mathbf{x}=0} \Delta \mathbf{x}. \quad (2.53)$$

Once the increment is found, it is composed with the current estimate via

$$\hat{\mathbf{X}} \leftarrow \hat{\mathbf{X}} \boxplus \Delta \mathbf{x}. \quad (2.54)$$

If the measurement error itself lies on a manifold (e.g. pose–pose constraints), a local subtraction using $\boxminus$ is required:

$$\mathbf{e}_k(\mathbf{X}) = \mathbf{h}(\mathbf{X}) \boxminus \mathbf{Z}_k, \quad (2.55)$$

where $\mathbf{h}$ denotes the measurement prediction function.

2.3.4 Factor Graph

In **SLAM** and 3D reconstruction, the estimation problem requires recovering a state composed of heterogeneous variables, *i.e.* robot poses in $\text{SE}(3)$, landmark positions $\mathbb{R}^3$, and sensor calibration parameters, often totaling thousands of dimensions. Despite this scale, each measurement constrains only a small subset of the state: a single relative motion reading, loop closure, or point reprojection involves just two or three variables. This locality induces a sparse optimization structure, which factor graphs make explicit.

A factor graph is a bipartite graph consisting of:

- **Variable Nodes** ($\mathbf{x}_i$): state elements to estimate.

- **Factor Nodes** (ϕ_j): sensor measurements or priors that impose constraints on one or more variables.

Each factor ϕ_j connects to its associated variables $\mathbf{x}_{j_1}, \dots, \mathbf{x}_{j_k}$ and encodes a residual function

$$\mathbf{e}_j(\mathbf{x}_{j_1}, \dots, \mathbf{x}_{j_k}) = \mathbf{z}_j - h_j(\mathbf{x}_{j_1}, \dots, \mathbf{x}_{j_k}),$$

weighted by an information matrix $\mathbf{\Omega}_j$. All residuals then combine into the unified least-squares formulation as in Eq. (2.28).

Crucially, the sparsity pattern of $\mathbf{H}$ matches the factor graph: nonzero blocks appear only for variable pairs sharing a factor. Exploiting this correspondence with sparse Cholesky factorization and graph-based variable ordering heuristics dramatically reduces memory usage and computation time.

Factor graphs are modular: adding a new sensor modality, be it inertial, [LiDAR](#), or [RADAR](#), simply introduces new factor types without changing the solver. Common factors include: **Pose–Pose**: odometry or loop-closure constraints between two frames; **Pose–Landmark**: reprojection errors linking 3D points to camera poses; **Priors**: anchors that fix gauge freedoms or initialize calibration.

By unifying all constraints under the least-squares framework and exploiting the resulting sparsity, factor graphs provide a scalable, extensible backbone for modern [SLAM](#) pipelines.

Chapter 3

MD-SLAM: Multi-cue Direct SLAM

SLAM is a popular field in robotics, and after roughly three decades of research, effective solutions are available. As many sectors rely on [SLAM](#), such as autonomous driving, augmented reality and space exploration, it still receives much attention in academia and industry. The advent of robust machine learning systems allowed the community to enhance purely geometric maps with semantic information or replace hard-coded heuristics with data-driven ones. Within the computer vision community, we have seen photometric (or direct) approaches used to tackle the [SLAM](#) (or Structure from Motion (SfM)) problem. The direct techniques address pairwise registration by minimizing the pixel-wise error between image pairs. By not relying on specific features and having the potential of operating at subpixel resolution on the entire image, direct approaches do not require explicit data association and offer the possibility to boost registration accuracy [139]. Whereas these methods have been successfully used on monocular, stereo, or RGB-D images, their use on 3D [LiDAR](#) data is less prominent; probably due to the comparably limited vertical resolution relation to cameras. Della Corte *et al.* [38] presented a multi-cue photometric registration methodology for RGB-D cameras. It is a system that extends photometric approaches to different projective models and enhances the robustness by considering additional cues such as normals and depth or range in the cost function. Recently released 3D [LiDAR](#)s sensors offer up to 128 beams of vertical resolution, making direct approaches also more attractive for [LiDAR](#) data. In addition, most [LiDAR](#)s provide intensity or reflectivity information besides range data. This intensity information can be used to sense a light reflectivity cue from the objects in the environment. Being able to assemble an intensity-like image out of a [LiDAR](#) scan has unleashed the possibility of using well-known computer vision appearance-based methods for place recognition [66].

The dominant paradigm for modern [SLAM](#) systems today is graph-based [SLAM](#). A graph-based [SLAM](#) system works by constructing a [SLAM](#) graph where each node represents the sensor position or a landmark, while edges encode a relative displacement between nodes. Pose-graphs are a particular case in which only poses are stored in the graph. These local transformations stored in the edges are commonly inferred by comparing and matching sensor readings. This paper investigates the fusion of multi-cue direct registration with graph-based [SLAM](#).

In this chapter, we are going to present a flexible, direct [SLAM](#) pipeline for 3D data. To the best of our knowledge, our approach is the only open-source [SLAM](#) system that can deal with RGB-D and [LiDAR](#) in a unified manner. We realized a revised version of

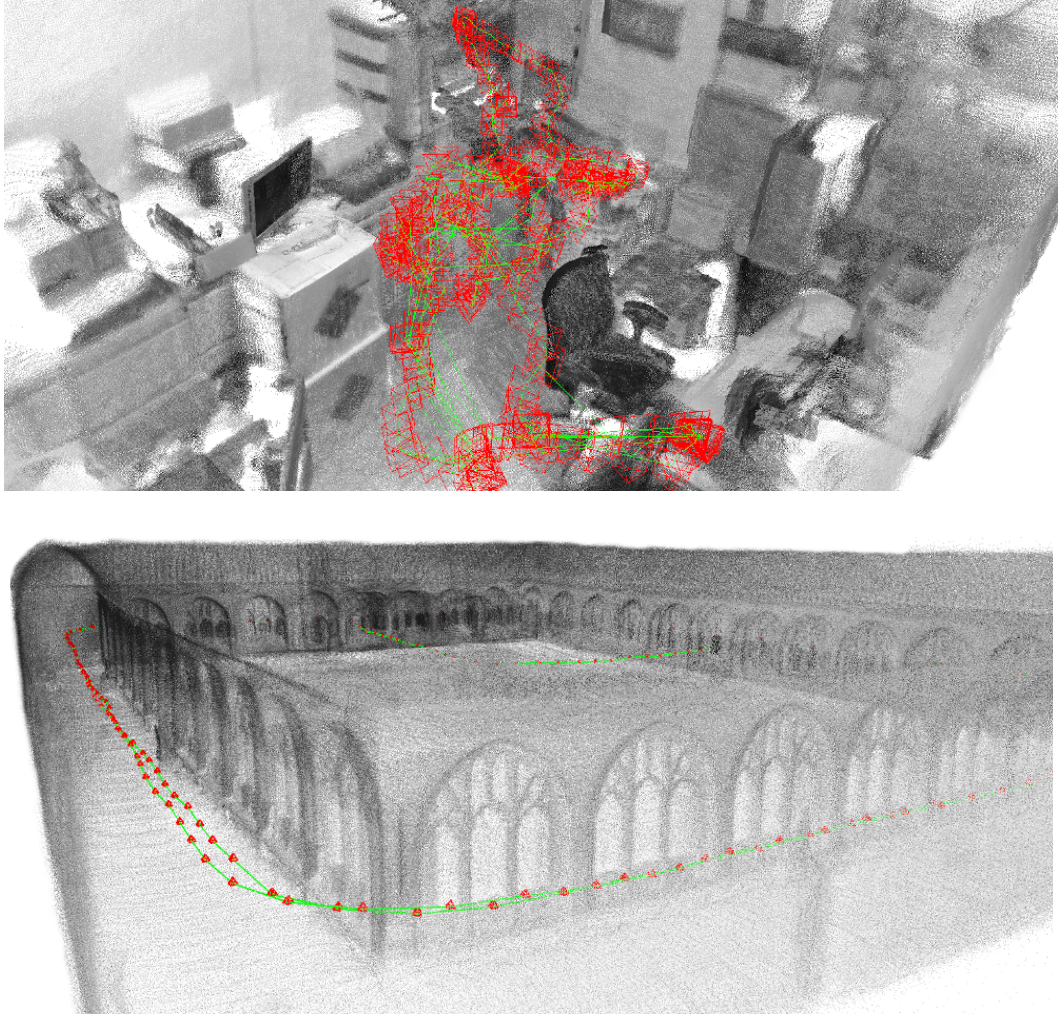


Figure 3.1. Scenes reconstructed using our pipeline. Top: results of a self-recorded dataset using Intel Realsense 455 RGB-D. Bottom: using **LiDAR** OS0-128 of the *cloister* sequence from the Newer College Dataset [170]. Red pyramids represent the keyframe poses of the sensor; green lines show the factors between them.

MPR [38] for computing the incremental motion of the sensor operating on RGB-D as well as **LiDAR** data. We detect loop closures by an appearance-based algorithm that uses a Binary Search Tree (BST) structure proposed by Schlegel *et al.* [138], populated with binary feature descriptors [136]. All components that require the solution of an optimization problem rely on the same framework [71], resulting in a compact implementation. It is designed for flexibility, hence not optimizing the **SLAM** system to a specific sensor. Our system has been tested on both, RGB-D and **LiDAR** data, using benchmark datasets. The accuracy is competitive concerning other sensor-specific **SLAM** systems, while it outperforms them if some assumptions about the structure of the environment are violated. An open-source CUDA/C++ implementation that runs in realtime for both sensors: <https://github.com/rvp-group/mdslam>. Fig. 3.1 illustrates example maps map built with our system using RGB-D (top) and **LiDAR** (bottom).

3.1 Related Work

3D SLAM has been widely addressed by the computer vision and robotics community and a large number of valid SLAM systems are available. The available computational resources limited early approaches to operate offline [119] in fairly limited environments [162]. After the Kinect sensor became available about 15 years ago, we observed a revamped interest in RGB-D SLAM. Newcombe *et al.* [81] were the first to leverage a dense tracking on a Truncated Signed Distance Function (TSDF) stored in the GPU while using massively parallel implementation to render the surface of the local scene perceived by the sensor. Meanwhile, Segal *et al.* [140] proposed a robust variant of Iterative Closest Point (ICP) relying on a point-to-plane metric. These initial methods addressed the open-loop registration approaches, tracking the pose of the sensor in a small neighborhood. The advent of efficient optimization systems such as *iSAM* [83] and *g²o* [72], made it possible to build an effective full-fledged 3D SLAM system supporting loop closures and providing an online globally consistent estimate. Novel efficient salient floating point [15] and binary image descriptors [136], paired with bag-of-words retrieval methods inspired from web search engines, lead to impressive place recognition approaches [59]. These methods were then employed within visual SLAM systems, ORB-SLAM by Mur Artal *et al.* [118] being one of the most popular ones. The pipeline fully relied on the stability of features (keypoints), minimizing the reprojection error of the reconstructed landmarks within the image. In contrast to these indirect methods, another line of research aimed at photometric error minimization. Keller *et al.* [87] use projective data matching in a dense model, relying on a surfel-based map for tracking. Others rely on keyframe-based technique [89]. As it happened for feature-based approaches, these works were assembled into full visual SLAM systems [48]. More recently, BAD-SLAM, a surfel-based direct Bundle Adjustment (BA) system that combines photometric and geometric error [139] using feature-based loop closures, shows that, for well-calibrated data, dense BA outperforms sparse BA. The accuracy and elegance shown by photometric approaches lead to further developments such as MPR [38] aiming at unifying both LiDAR and RGB-D devices into a unique registration method. Built maps can then be used for robot navigation [152], even in hazardous environments [148].

In parallel, the community approached LiDAR-based odometry by seeking alternative representations for the dense 3D point clouds. These include 3D salient features [168, 82], subsampled clouds [155] or Normal Distributed Transform (NDT) [147]. Nowadays, LiDAR Odometry and Mapping (LOAM) is perhaps one of the most popular methods for LiDAR odometry [168, 169]. It extracts distinct features corresponding to surfaces and corners, then used to determine point-to-plane and point-to-line distances to a voxel grid-based map representation. A ground optimized version (LeGO-LOAM) method has been later proposed [141], as it leverages the presence of a ground plane in its segmentation and optimization steps. In contrast to sparse methods, dense approaches suffer less in a non-structured environment [16]. Compared to RGB-D images, 3D LiDARs offer lower support for appearance-based place recognition. It is common for dense LiDAR SLAM systems to attempt a brute force registration with all neighborhood clouds to seek loop closures. Thanks to the typically small drift, this strategy is most successful; however, computational costs grow significantly in large environments. LiDAR loop closures have been addressed in different ways compared to RGB-D. Magnusson *et al.* proposed an approach suitable for NDT representations [113]. Röhling *et al.* [134] investigated the use of histograms computed directly from the 3D point cloud to define a measure of the similarity of two

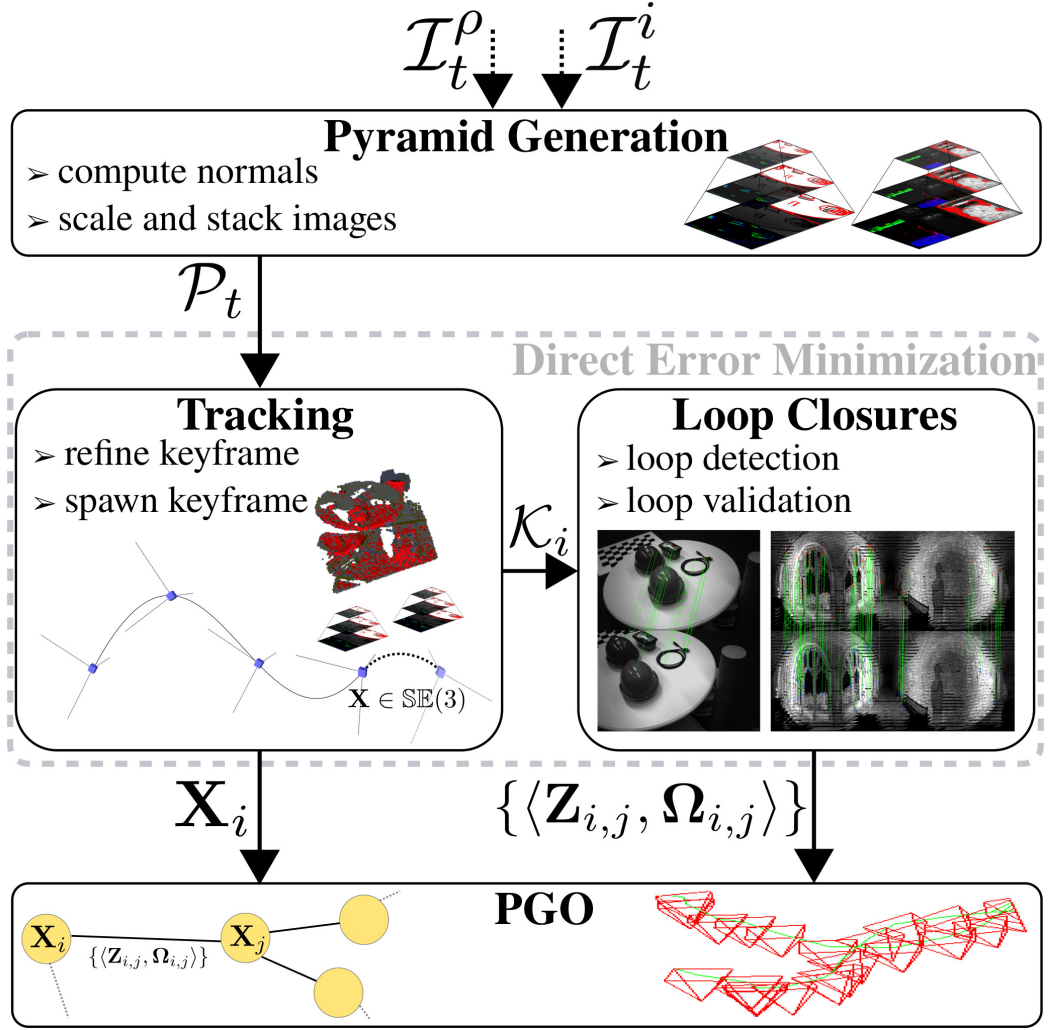


Figure 3.2. Illustration of our system. Range $\mathcal{I}_t^\rho$ and $\mathcal{I}_t^i$ images are taken as input from the system. An optimized trajectory within a map is produced as output. This system works independently both for RGB-D and LiDAR.

scans. Novel types of descriptors have been investigated, exploiting additional data gathered by the LiDAR sensor – *i.e.*, light emission of the beams [37, 73]. However, despite being very attractive, these descriptors are time-consuming to extract and match, resulting in a slower system overall. Recent works address loop-closures detection in a RGB-D fashion, relying on the visual feature matching extracted from the image obtained by using the LiDAR intensity channel [66].

Building on top of prior works [38, 66], the contribution of our study is a flexible and general SLAM approach. It is a direct method working on RGB-D and 3D LiDAR data alike providing a unified approach. Our results show that it is competitive with other sensor-specific systems.

3.2 Multi-cue Direct SLAM

Our approach relies on a pose-graph to represent the map. Nodes of the pose-graph store keyframes in the form of multi-cue image pyramids. Our pipeline takes as input intensity (grayscale) and depth images for RGB-D or intensity and range images for [LiDAR](#). For compactness, we will generalize, mentioning only range images. The pyramids are generated from the inputs images each time a new frame becomes available. By processing the range information, our system computes the surface normals and organizes them into a three-channel image, which is then stacked to the original input to form a five-channel image. Pyramids are generated by downscaling this input. This process is described in Sec. 3.2.1.

The pyramids are fed to the tracker, which is responsible for estimating the relative transform between the last keyframe and the current pyramid through the direct error minimization strategy summarized in Sec. 3.2.2. The tracker is in charge of spawning new keyframes and adding them to the graph when necessary, as discussed in Sec. 3.2.3.

Whenever a new keyframe is generated, the loop closure schema, described in Sec. 3.2.4, seeks for potential relocalization candidates between the past keyframes by performing a search in appearance space. Candidate matches are further pruned by geometric validation and direct refinement. Successful loop closures result in the addition of new constraints in the pose-graph and trigger a complete graph optimization as detailed in Sec. 3.2.5.

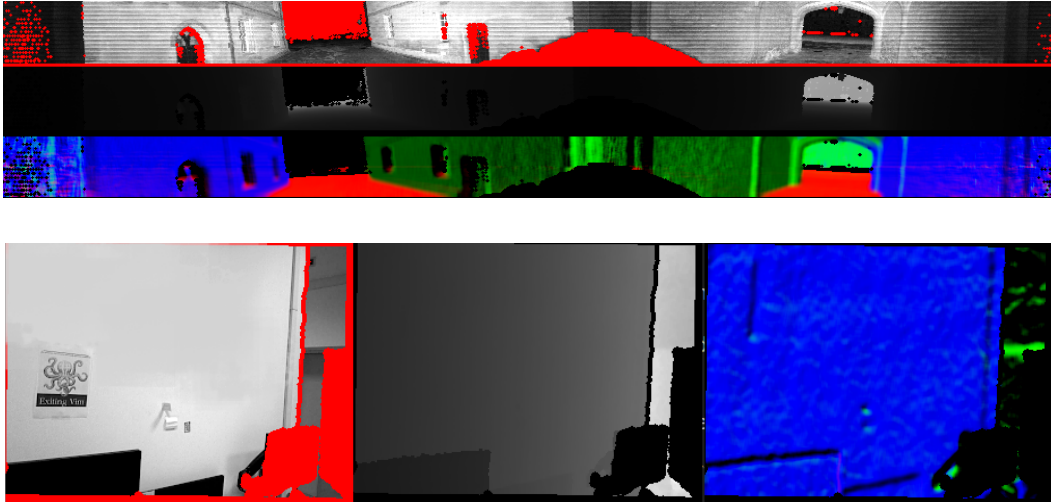


Figure 3.3. Cues generated for [LiDAR](#) (top) and RGB-D (bottom) images. The first row/column shows the intensity $\mathcal{I}^i$, the middle shows the range $\mathcal{I}^p$, and the last one illustrates the normals encoded by color $\mathcal{I}^n$. The red pixels on the intensity cues are invalid measurements (i.e., range not available).

3.2.1 Input Preprocessing

The first step to generate a pyramid from a pair of intensity $\mathcal{I}^i$ and range image $\mathcal{I}^p$ consists of extracting the normals. To calculate the normal at pixel $\mathbf{u}$ we unproject the pixels in the neighborhood $\mathcal{U} = \{\mathbf{u}' : \|\mathbf{u} - \mathbf{u}'\| < \tau_{\mathbf{u}}\}$ whose radius $\tau_{\mathbf{u}}$ is inversely proportional to the range at the pixel $I^p(\mathbf{u})$. The normal $\mathbf{n}_{\mathbf{u}}$ is the one of the plane that best fits the unprojected points from the set $\mathcal{U}$. All valid normals are assembled in a normal image $\mathcal{I}^n$,

so that $\mathcal{I}^n(\mathbf{u}) = \mathbf{n}_u$.

One level of a pyramid $\mathcal{I}$, therefore, consists of three images: $\mathcal{I}^i$, $\mathcal{I}^p$ and $\mathcal{I}^n$. Further channels such as curvature and semantics can easily be embedded into the representation by adding additional images. In the remainder, we will refer to one general image in the set $\mathcal{I}$ as a *cue* $\mathcal{I}^c$.

Pyramids are required to extend the basin of convergence of direct registration methods. This is due to the implicit data association, which operates only in the neighborhood of a few pixels. Hence, downscaling the images increases the convergence basin at the cost of reduced accuracy. However, the accuracy can be recovered by running the registration from the coarser to the finest level. Each time a level is changed, the initial guess of the transformation is set to the solution of the previous coarser level.

Another way to increase the convergence basin is by smoothing the images with a Gaussian filter and progressively reducing the kernel size as convergence improves. This approach is similar to the pyramid strategy, but since the image resolution is preserved, it is more computationally demanding.

A pyramid $\mathcal{P}$ is generated from all the cues $\mathcal{I} = \{\mathcal{I}^c\}$, by downscaling at user-selected resolutions. In our experiments, we typically use three scaling levels, each of them half the resolution of the previous level.

3.2.2 Direct Error Minimization

As in direct error minimization approaches, our method seeks to find the transform $\mathbf{X}^* \in \mathbb{SE}(3)$ that minimizes the photometric distance between the two images:

$$\mathbf{X}^* = \underset{\mathbf{X} \in \mathbb{SE}(3)}{\operatorname{argmin}} \sum_{\mathbf{u}} \underbrace{\left\| \hat{\mathcal{I}}^i(\mathbf{u}) - \mathcal{I}^i \left(\overbrace{\pi \left(\mathbf{X} \pi^{-1}(\mathbf{u}, \rho) \right)}^{\mathbf{u}'} \right) \right\|}_{\mathbf{e}_u}^2 \quad (3.1)$$

Where $\mathbf{e}_u^i$ denotes the error between corresponding pixels. The evaluation point $\mathbf{u}'$ of $\mathcal{I}^i$ is computed by inverse projecting the pixel $\mathbf{u}$, applying the transform $\mathbf{X}$ and projecting it back. To carry out this operation, the range at the pixel $\rho = \mathcal{I}^p(\mathbf{u})$ needs to be known.

Eq. (3.1) models classical photometric error minimization assuming that the cues are not affected by the transform $\mathbf{X}$. In our case, range and normal are affected by $\mathbf{X}$. Hence, we need to account for the change in these cues, and we will do it by introducing a mapping function $\mathcal{M}^c(\mathbf{X}, \hat{\mathcal{I}}^c(\mathbf{u}))$. This function calculates the *pixel* value of the c^{th} cue after applying the transform $\mathbf{X}$ to the original channel value $\hat{\mathcal{I}}^c(\mathbf{u})$. We can thus rewrite a more general form of Eq. (3.1) that accounts for all cues and captures this effect as follows:

$$\mathbf{X}^* = \underset{\mathbf{X} \in \mathbb{SE}(3)}{\operatorname{argmin}} \sum_c \sum_{\mathbf{u}} \underbrace{\left\| \mathcal{M}^c(\mathbf{X}, \hat{\mathcal{I}}^c(\mathbf{u})) - \mathcal{I}^c(\mathbf{u}') \right\|}_{\mathbf{e}_u^c}^2_{\Omega^c} \quad (3.2)$$

The squared Mahalanobis distance $\|\cdot\|_{\Omega^c}^2$ is used to weight the different cues. More details about a general methodology for direct registration can be found in [38].

While approaching the problem in Eq. (3.2) with the ILS method described in Sec. 2.3.2, particular care has to be taken to the numerical approximations of floating-point numbers. In particular, since each pixel and cue contribute to constructing the quadratic form with an independent error $\mathbf{e}_u^c$, the summations in Eq. (2.34) might accumulate millions of terms.

Hence, to lessen the effect of these round-offs, Eq. (2.34) has to be computed using a stable algorithm. In our single-threaded implementation, we use the compensated summation algorithm [78]; its numerical-stability benefits are detailed in Sec. 3.4.

We use multi-cue direct alignment in incremental position tracking, explained in next section (Sec. 3.2.3) and in loop closure refinement and validation (Sec. 3.2.4).

3.2.3 Tracking

This module is in charge of estimating the open-loop trajectory of the sensor. To this extent, it processes new pyramids as they become available by determining the relative transform between the last pyramid $\mathcal{P}_t$, and the current keyframe $\mathcal{K}_i$. A keyframe stores a global transform $\mathbf{X}_i$, and a pyramid $\mathcal{P}_i$. The registration algorithm of Sec. 3.2.2 is used to compute a relative transform $\mathbf{Z}_{i,t}$ between the last two pyramids. Whenever the magnitude of such a transform exceeds a given threshold or the overlap between $\mathcal{P}_i$ and $\mathcal{P}_t$ becomes too small, the tracker spawns a new keyframe $\mathcal{K}_{i+1}$, with transform $\mathbf{X}_{i+1} = \mathbf{X}_i \mathbf{Z}_{i,i+1}$. Furthermore, it adds to the graph a new constraint between the nodes i and $i+1$, with transform $\mathbf{Z}_{i,i+1}$, and information matrix $\mathbf{\Omega}_{i,i+1}$. The latter is set to $\mathbf{H}$ matrix of the direct registration at the optimum. The generation of the new keyframe triggers the loop detection described in the next section. Using keyframes reduces the drift that would occur when performing subsequent pairwise registration since the reference frame stays fixed for a longer time. Potentially, if the sensor hovers at a distance smaller than the keyframe threshold, all registrations are done against the same pyramid, and no drift would occur.

3.2.4 Loop Detection and Validation

This module is responsible for relocalizing a newly generated keyframe with respect to previous ones. More formally, given a query frame $\mathcal{K}_i$, it retrieves a set of tuples $\{\langle \mathcal{K}_j, \mathbf{Z}_{i,j}, \mathbf{\Omega}_{i,j} \rangle\}$, consisting of a past keyframe $\mathcal{K}_j$, a transform $\mathbf{Z}_{i,j}$ between $\mathcal{K}_i$ and $\mathcal{K}_j$ and an information matrix $\mathbf{\Omega}_{i,j}$ characterizing the uncertainty of the computed transform. Our system approaches loop closing in multiple stages. At first, we carry on visual place recognition on the intensity channels. This approach leverages the results of previous work [66]. For visual place recognition, we rely on ORB feature descriptors, extracted from the $\mathcal{I}^i$ of each keyframe. Retrieving the most similar frame to the current one results in looking for the images in the database having the closest descriptor “close” to the one of the current image. To efficiently conduct this search, we use a hamming distance embedding BST (HBST) [138], a tree-like structure that allows for descriptor search and insertion in logarithmic time by exploiting particular properties of binary descriptors. A match from HBST also returns a set of pairs of corresponding points between the matching keypoints. Having the depth and inverse projecting the points, we can carry on a straightforward RANdom SAMple Consensus (RANSAC) registration. Finally, each candidate match is subject to direct refinement (Sec. 3.2.2). This step enhances the accuracy and it provides information matrices on the same scale as the ones generated by the tracker. The above strategy is applied independently to RGB-D or LiDAR data. These surviving pairs $\{\langle \mathcal{K}_j, \mathbf{Z}_{i,j}, \mathbf{\Omega}_{i,j} \rangle\}$, constitute potential loop closing constraints to be added to the graph. However, to handle environments with large sensor aliasing, we introduced a further check to preserve topological consistency. Whenever a loop closure is found, we carry on a direct registration between all neighbors that would result *after* accepting the closure. If the resulting error is within certain

bounds, the closure is finally added to the graph, and a global optimization is triggered.

3.2.5 Pose-graph Optimization

The goal of this module is to retrieve a configuration of the keyframes in the space that is maximally consistent with the incremental constraints introduced by the tracker and the loop closing constraints by the loop detector. A pose-graph is a special case of a factor graph [71, 83], as presented in Sec. 2.3.4. The nodes of the graph are the keyframe poses $\mathbf{X} = \{\mathbf{X}_i\}_{i=1:N}$, while the constraints encode the relative transformations between the connected keyframes, together with their uncertainty $\{\langle \mathbf{Z}_{i,j}, \mathbf{\Omega}_{i,j} \rangle\}$. Optimizing a factor graph consists in solving the following optimization problem:

$$\mathbf{X}^* = \underset{\mathbf{X} \in \mathbb{SE}(3)^N}{\operatorname{argmin}} \sum_{i,j} \underbrace{\|\mathbf{X}_i^{-1} \mathbf{X}_j \ominus \mathbf{Z}_{i,j}\|_{\mathbf{\Omega}_{i,j}}^2}_{\mathbf{e}_{i,j}} \quad (3.3)$$

Here, the error $\mathbf{e}_{i,j}$ is the difference between predicted displacement $\mathbf{X}_i^{-1} \mathbf{X}_j$ and result of the direct alignment $\mathbf{Z}_{i,j}$. The total perturbation vector $\Delta \mathbf{x} \in \mathbb{R}^{6N}$ results from stacking all variable perturbations $\{\Delta \mathbf{x}_i\}$.

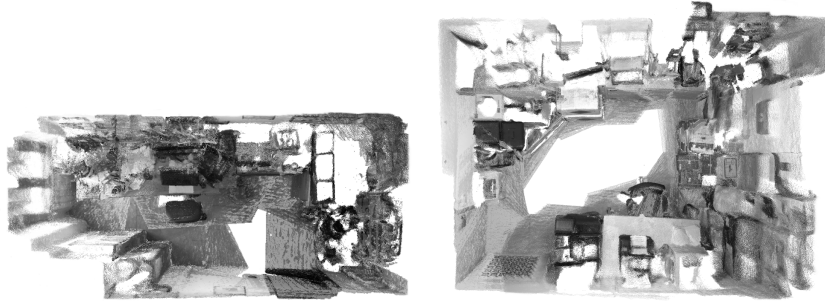


Figure 3.4. Qualitative RGB-D reconstructions showing the global consistency produced by our pipeline. Data has been self-recorded with an Intel Realsense 455.

3.3 Experimental Evaluation

In this section, we report the results of our pipeline on different public benchmark datasets. To the best of our knowledge, our approach is the only open-source **SLAM** system that can deal with RGB-D and **LiDAR** in a unified manner. Therefore, to evaluate our system, we compare with state-of-the-art **SLAM** packages developed specifically for each of these sensor types. For RGB-D we consider DVO-SLAM [89] and ElasticFusion [161] as direct approaches and ORB-SLAM2 [118] as indirect representative. For **LiDAR** we compare against LeGO-LOAM [141] as feature-based and SuMA [16] representing the dense category. We specify that our pipeline is purely photometric namely, not **IMU** or odometry source have been used as assistance. To run the experiments, we used a workstation with an Intel Core i7-7700K CPU @ 4.20GHz and 16GB of RAM. Since this work is focused on **SLAM**, we perform our quantitative evaluation using the Root Mean Square Error (RMSE) on the Absolute Trajectory Error (ATE) with $\mathbb{SE}(3)$ alignment. The alignment for the metric is computed by using the Horn method [79], and the timestamps are used to determine the

associations. Then, we calculate the **RMSE** of the translational differences between all matched poses.

The tracking module dominates the runtime of our approach since loop closures are detected and validated asynchronously within another thread. Hence, we report the average frequency at which the tracker runs for each sensor. At the core of the tracker, we have the photometric registration algorithm, whose computation is proportional to the size of the images. Runtimes summarized in Tab. 3.1

RGB-D (480x640)		LiDAR (128x1024)	
CPU	GPU	CPU	GPU
8	30	15	60

Table 3.1. MD-SLAM runtimes expressed in Hz.

3.3.1 RGB-D Results

We conducted several experiments with RGB-D sensor. Qualitative analysis have been done using self-recorded data and are shown in Fig. 3.4. As public benchmarks we used the TUM-RGB-D [149] and the ETH3D [139]. The TUM RGB-D dataset contains multiple real datasets captured with handheld Xbox Kinect. A rolling shutter camera provides RGB data. Further, the depth and color streams of the camera are not synchronized. Every sequence accompanies an accurate ground truth trajectory obtained with an external motion capture system. ETH3D benchmark is acquired with global shutter cameras and accurate active stereo depth. Color and depth images are synchronized. We select several indoor sequences for which ground truth, computed by external motion capture, is available.

On these datasets, we compare with DVO-SLAM, ElasticFusion and ORB-SLAM2. These three approaches are representative of different classes of **SLAM** algorithms. Tab. 3.2 shows the results on the TUM RGB-D datasets, while Tab. 3.3 presents the outcome on the ETH3D datasets. DVO SLAM implements a mixed geometry-based and direct registration. Internally the alignment between pairs of keyframes is obtained by jointly minimizing point-to-plane and photometric residuals. This is similar to ElasticFusion, whose estimate consists of a mesh model of the environment and the current sensor location instead of the trajectory. In contrast to these two approaches, ORB-SLAM2 implements a traditional visual **SLAM** pipeline, where a local map of landmarks around the RGB-D sensor is constructed from ORB features. This map is constantly optimized as the camera moves by performing local **BA**. Loop closures are detected through DBoW2 [59] and a global optimization on a $\text{Sim}(3)$ pose-graph to enforce global consistency is used.

The TUM dataset provides images 640×480 pixels, while ETH3D 740×460 pixels. From these images, we compute a 3 level pyramid with scales $1/2$, $1/4$, and $1/8$. Our system runs respectively at 8 and 7.5 Hz at these resolutions on Central Processing Unit (CPU). On **GPU**, differently, operates at 30 Hz and 28.5 Hz, making the graphic-card implementation suitable for real-time applications.

In Tab. 3.2 we can see that ORB-SLAM2 clearly outperforms all other pipelines. DVO-SLAM and ElasticFusion provide comparable results, and our approach is the worst in terms of accuracy. Yet, the largest error is 6.4 cm, which results in a usable map. As stated before, this dataset is subject to rolling shutter and asynchronous depth effects. ORB-SLAM2,

	fr1/desk	fr1/desk2	fr2/desk
DVO-SLAM	0.021	0.046	0.017
ElasticFusion	0.020	0.048	0.071
ORB-SLAM2	0.016	0.022	0.009
Ours	0.041	0.064	0.057

Table 3.2. ATE [m] results on TUM RGB-D datasets, recorded with non-synchronous depth using a rolling shutter camera.

	table3	table4	table7	cables1	plant2	planar2
DVO-SLAM	0.008	0.018	0.007	0.004	0.002	0.002
ElasticFusion	—	0.012	—	0.018	0.017	0.011
ORB-SLAM2	0.007	0.008	0.010	0.007	0.003	0.005
Ours	0.021	0.022	0.036	0.015	0.001	0.001

Table 3.3. ATE [m] on ETH3D, recorded with global shutter camera and synchronous streams. ElasticFusion fails in *table3* and *table7*.

being feature-based, is less sensitive to these phenomena. DVO-SLAM and ElasticFusion explicitly model these effects. Our approach does not attempt to address these issues since it would render the whole pipeline less consistent between different sensing modalities.

Tab. 3.3 presents the results on the ETH3D benchmark. In this case, our performances are on par with other methods, since intensity and depth are synchronous, and the camera is global shutter.

These results highlight the strength and weaknesses of a purely direct approach not supported by any geometric association. While being compact, it suffers from unmodeled effects and requires a considerable overlap between subsequent frames.

3.3.2 LiDAR Results

We conducted different experiments on public L benchmarks to show the performances of our SLAM implementation. For the LiDAR we use the Newer College Dataset [129, 170] recorded at 10 Hz with two models of Ouster LiDARs: OS1 and OS0. We conducted our evaluation on the *long*, *cloister*, *quad-easy* and *stairs* sequences. The OS1 has 64 vertical beams. We selected the *long* sequence that lasts approximately 45 minutes. It consists of multiple loops with viewpoint changes between buildings and a park.

The other three shorter sequences are recorded with the OS0, which has 128 vertical beams. The LiDAR *quad-easy* sequence contains four loops that explore quad, *cloister* mixes outdoor and indoor scenes while *stairs* is purely indoor and based on vertical motion through different floors.

Qualitative analysis have been performed to show the results obtained by our pipeline. Fig. 3.6 illustrates some reconstructions obtained with MD-SLAM from Newer College sequences. Fig. 3.5 and Fig. 3.7 show the global consistency of our estimate on *long* sequence.

Quantitatively, we compare against LeGO-LOAM and SuMA. These represent two different classes of LiDAR algorithms, respectively sparse and dense. Tab. 3.4 summarizes

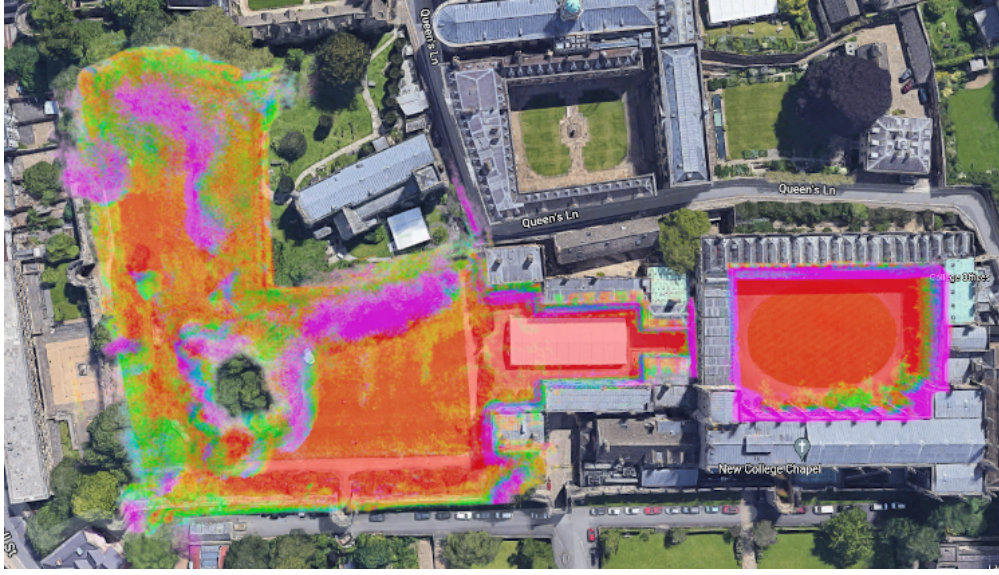


Figure 3.5. MD-SLAM map on *long* sequence from Newer College Dataset [129] aligned with Google Earth.

	OS0-128			OS1-64
	cloister	quad	stairs	long
LeGO-LOAM	0.20	0.09	3.20	1.30
SuMA	3.34	1.74	0.67	-
Ours	0.36	0.25	0.34	1.74

Table 3.4. ATE [m] results of all benchmarked approaches on the Newer College Dataset. SuMA fails on *long* sequence.

the results of the comparison. LeGO-LOAM is currently one of the most accurate **LiDAR SLAM** pipelines and represents a sparse class of **LiDAR** algorithms. In contrast to our approach, LeGO-LOAM is a pure geometric feature-based frame-to-model **LiDAR SLAM** work, where the optimization on roll, yaw and z-axis (pointing up) is decoupled from the planar parameters. SuMa constructs a surfel-based map and estimates the changes in the sensor’s pose by exploiting the projective data association in a frame-to-model or in a frame-to-frame fashion. For both the pipelines loop closures are handled through **ICP**. Being ground optimized, LeGO-LOAM shows impressive results mainly in chunks where ground occupies most of the scene, yet our approach provides competitive accuracy. The situation becomes challenging for LeGO-LOAM when its assumptions are violated, such as in the *stairs* sequence. In this case, our pipeline is the most accurate since it does not impose any particular structure on the environment being mapped. SuMA performances are the worst in terms of accuracy. We tried this pipeline both in a frame-to-frame and frame-to-model mode. The one reported in Tab. 3.4 represents SuMA frame-to-frame that always outperforms the frame-to-model on these datasets.

We use the OS1 to produce images of 64×1024 pixels while the OS0 to produce images of 128×1024 pixels. Since the horizontal resolution is much larger than the vertical one,

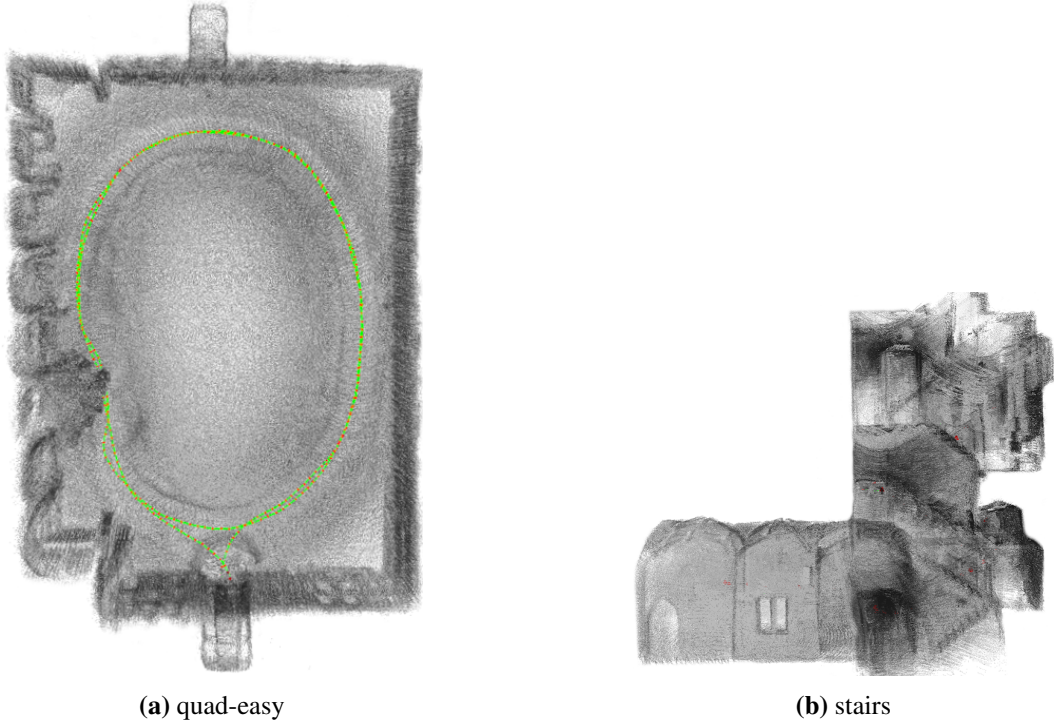


Figure 3.6. Some scenes from Newer College dataset reconstructed by our system.

to balance the aspect ratio for direct registration, initially, we downscale the horizontal resolution by $1/2$ for OS0 and by $1/4$ for OS1. Our approach generates a pyramid with the following scales: 1, $1/2$ and $1/4$. With these settings, our system operates in **CPU** at around 15 Hz on the OS0 and at approximately 24 Hz on the OS1. In **GPU** respectively at 60 Hz and 120 Hz. This makes both implementation suitable for online estimation.

3.4 Numerical Stability analysis

Numerical stability is an essential consideration in computationally intensive tasks such as **SLAM**, where precision and accuracy are critical. This section delves into numerical stability concerns relevant to our direct **SLAM** framework, emphasizing floating-point representation challenges and practical strategies to mitigate these issues.

Floating-point numbers, as represented in standard computing architectures, *e.g.* IEEE-754 standard [1] commonly implemented in C++, exhibit non-uniform density. This means the distribution of numbers that can be represented with a fixed number of bits is denser near zero and progressively sparser with increasing magnitude. Consequently, computations involving very small or very large values can lead to significant precision loss. This effect is particularly pronounced when many small numerical contributions are accumulated into a single computational structure.

In our **SLAM** system, the *tracker* Sec. 3.2.3 processes a single RGB-D frame of resolution 480×640 pixels. Each pixel contributes multiple residuals which are accumulated into a compact linear system—namely a Hessian matrix $\mathbf{H} \in \mathbb{R}^{6 \times 6}$ and a gradient vector $\mathbf{b} \in \mathbb{R}^6$.

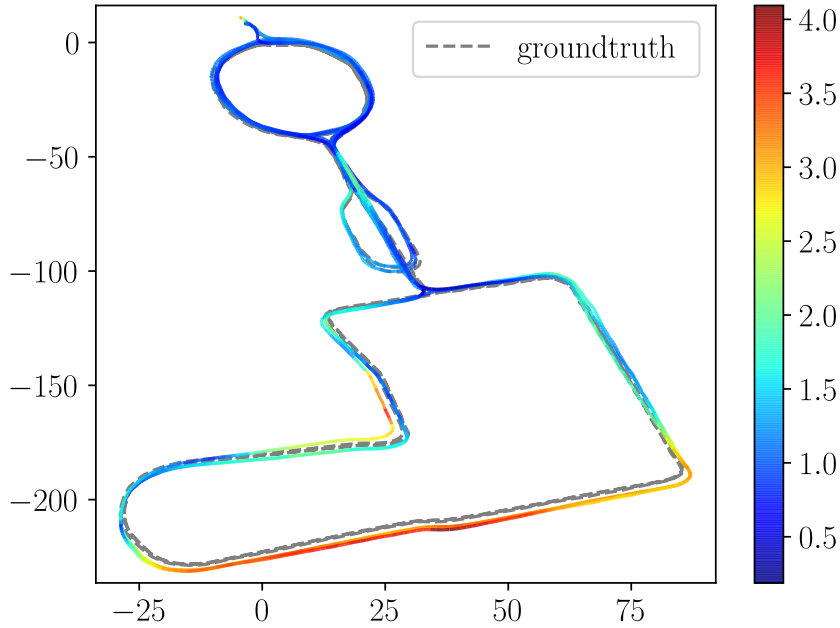


Figure 3.7. Alignment of our estimate with the ground truth in *long* sequence of Newer College. The color bar on the right shows the translational error [m] over the whole trajectory.

During this accumulation process of Hessians and gradient vectors, each new value added can be orders of magnitude smaller than the current sum. Over numerous operations, these small errors can compound, resulting in significant inaccuracies in the computed linear system.

To mitigate these accumulation errors in CPU computations, we employ the Kahan summation algorithm [78], designed explicitly to minimize floating-point errors. The algorithm introduces a compensation term, continuously adjusted during summation to correct accumulated rounding errors. The pseudo-code of the Kahan summation algorithm is shown in Alg. 1.

Algorithm 1 Kahan summation algorithm

Require: input_vec	▷ Input vector
Ensure: S	▷ Accurate sum
1: $S \leftarrow 0, c \leftarrow 0$	▷ The compensation c will store and correct the error
2: for each x in input_vec do	
3: $y \leftarrow x - c$	
4: $\text{temp} \leftarrow S + y$	▷ Temporarily store the sum
5: $c \leftarrow (\text{temp} - S) - y$	▷ Update the compensation
6: $S \leftarrow \text{temp}$	▷ Sum update

This method ensures the accumulation of small values does not degrade the overall accuracy, significantly improving numerical stability Fig. 3.8.

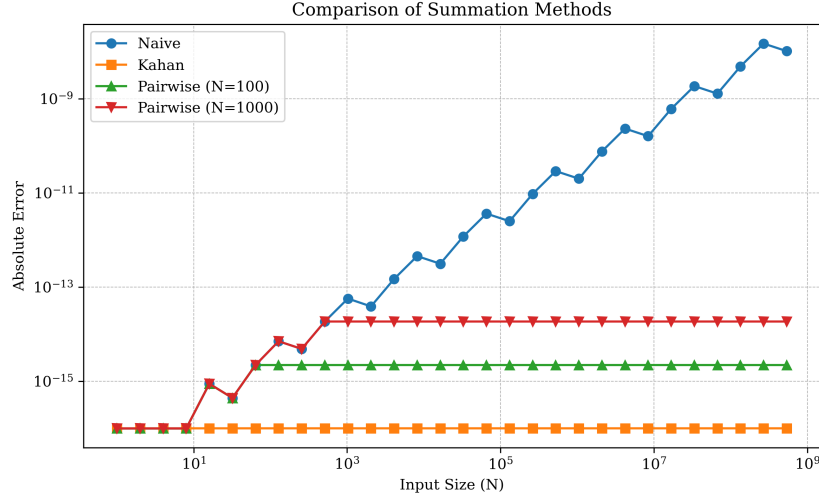


Figure 3.8. Absolute error comparison of summation algorithms. The y-axis marks 0 at the numerical precision threshold ($\sim 10^{-16}$), making it explicit in log scale. Kahan summation (blue) consistently maintains this level of accuracy, unlike naive (green) and pairwise (orange, black) approaches.

On GPU platforms, the parallel nature of computation inherently helps manage numerical stability through structured hierarchical summation, known as parallel reduction. CUDA, for example, provides efficient implementations of parallel reduction that organize additions in a tree-like hierarchy, significantly reducing the potential for accumulated rounding errors compared to sequential additions [75]. Each GPU thread initially handles a small subset of the data, combining these subsets in multiple stages to form a final stable summation. This structured reduction naturally reduces numerical inaccuracies.

Both CPU-based (Kahan summation) and GPU-based (parallel reduction) approaches contribute critically to the stability and reliability of our SLAM framework. By effectively addressing numerical precision concerns, these strategies ensure consistent and accurate performance across various sensor configurations and computational scenarios.

3.5 Conclusion

In this chapter, we introduced a comprehensive direct SLAM framework capable of integrating both RGB-D and LiDAR sensors. The central contribution lies in the unified handling of diverse sensor modalities by employing a shared internal representation that only requires adjustments in the projection models specific to each sensor. To the best of our knowledge, our method is uniquely positioned as an open-source solution that seamlessly bridges RGB-D and LiDAR data within a single iterative least squares solver. Comparative experiments show that our generic method can compete with sensor-specific state-of-the-art approaches. This architecture not only ensures compact and maintainable software but also maintains competitive accuracy when compared to specialized, sensor-dedicated state-of-the-art approaches.

While our approach represents significant progress in unifying sensor data processing, it remains preliminary, particularly regarding the application of direct methods to LiDAR-

based [SLAM](#). A crucial limitation identified in this chapter is the absence of compensation for motion distortion stemming from non-instantaneous data acquisition. Experimental results underscore this challenge: sequences captured with rolling shutter cameras demonstrated notably inferior performance compared to those recorded using global shutter cameras. Similarly, the relatively low acquisition rate of [LiDAR](#) sensors (around 10 Hz, compared to 30 Hz typical of cameras) intensifies motion distortion effects, significantly degrading the system’s accuracy. Consequently, our method underperforms compared to dedicated solutions that assume planar motion for simplifying the least squares optimization or facilitating data association, such as LeGO-LOAM and SuMa. However, precisely due to its motion-model agnostic nature, our system demonstrates superior versatility and accuracy in scenarios like handheld operations, where assumptions about movement cannot be reliably applied.

Additionally, we contextualized our contributions with respect to recent advances in frame-to-model approaches, which are recognized for their reduced drift compared to keyframe-based [SLAM](#) methods relying on pose graph structures. These approaches frequently utilize neural rendering techniques, such as Neural Radiance Fields (NeRF) [42, 135] and Gaussian Splatting [115], for model representation. Such methods synthesize novel viewpoints to match incoming measurements, effectively reducing drift. [LiDAR](#)-specific adaptations of these neural techniques have also been explored [65]. Nonetheless, these frame-to-model approaches typically rely on sensor-specific solutions, in contrast to the more generalized, sensor-independent nature of the framework presented in this chapter.

Given these insights and identified limitations, explicitly addressing motion distortion emerges as an essential future direction. Subsequent chapters will elaborate on advanced strategies to manage motion distortion more effectively Chapter 7. In particular, the Direct Radar Odometry (DRO) Chapter 8 framework, detailed in a subsequent chapter, explicitly addresses motion distortion and other sensor-induced artifacts, resulting in significant improvements in estimation accuracy and robustness. This enhancement further demonstrates the potential and flexibility of our direct estimation methods across diverse sensor modalities.

Chapter 4

VBR: A Vision Benchmark in Rome

STANDARDIZED datasets have played a central role in advancing computer vision and robotics by enabling rigorous benchmarking, algorithm comparison, and reproducible research. When ground truth is available, communities have been able to define meaningful metrics, pushing the state of the art through iterative development and competition. In recent years, with the proliferation of machine learning and data-driven techniques, such datasets have become essential, not only for evaluation but also as the input for training non-parametric models.

Among all, the KITTI dataset has had a transformative impact on both the robotics and computer vision communities, setting a de-facto standard for visual odometry and [SLAM](#) evaluation. Since its introduction, several other multi-sensor datasets have emerged [[112](#), [139](#), [91](#), [171](#)], yet none have achieved the same broad influence or completeness of KITTI [[62](#)].

Despite its continued relevance, KITTI shows its age. In the past decade, sensor technology, computing hardware, and ground truth systems have evolved significantly. Many modern datasets still rely on [GNSS](#)/Inertial Navigation System (INS) pipelines that are vulnerable to synchronization issues and often lack the precision required for evaluating cutting-edge [SLAM](#) systems, especially those relying on direct, dense representations. Furthermore, most available datasets are focused on road scenarios and autonomous driving applications, limiting their utility for general-purpose robotics and [SLAM](#) research.

Some recent works have tried to address these limitations by focusing on seasonality [[112](#)] or hand-held indoor trajectories with improved ground truth [[129](#)], but few have attempted to address all these issues simultaneously in a cohesive framework.

In this chapter, we introduce a dataset family designed to support the development and evaluation of high-precision state estimation systems in a wide variety of environments. We provide six datasets recorded in the city of Rome using a hardware-synchronized multi-sensor setup, including a 3D [LiDAR](#), a large-baseline stereo camera, an [RTK-GNSS](#), and a high-grade [IMU](#). The dataset spans more than 40 kilometers of trajectory and nearly four hours of continuous acquisition, amounting to approximately 2 TB of data.

The sequences intentionally cover a range of environments, *i.e.* urban, forested, and indoor, with consistent sensor configurations but varying acquisition modalities. This diversity is meant to challenge generalization and reduce overfitting to specific domains, thus offering a more robust benchmark for [SLAM](#) research.

A central contribution of this work is a procedure to generate accurate ground truth

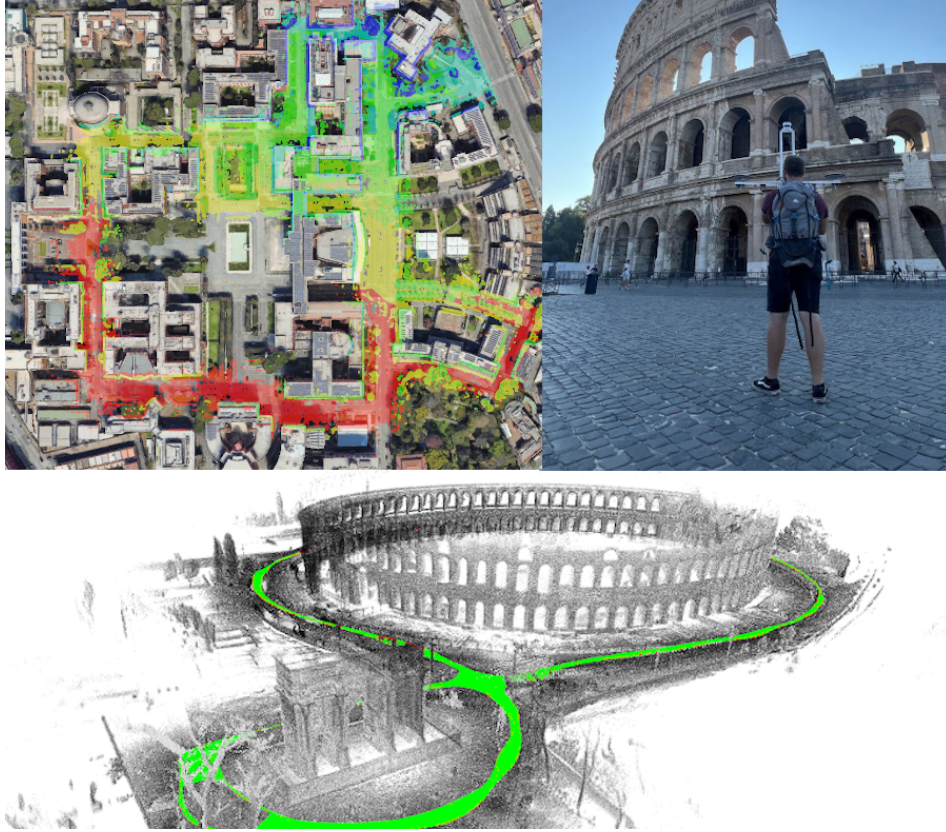


Figure 4.1. A summary of our dataset. Data illustrating some of the sequences recorded (top). 3D mapping done with of our ground truth (bottom).

trajectories by fusing [RTK-GNSS](#) with a global [BA](#) performed on the [LiDAR](#) data. This process leverages a photometric refinement stage based on [LiDAR](#)-image alignment and bilinear interpolation projection models, inspired by recent advances in direct methods. The resulting ground truth has been validated against a total station along a 1.5 km mixed indoor/outdoor trajectory, achieving an accuracy of approximately ± 3 cm.

Importantly, this chapter not only introduces a benchmark but also lays the groundwork for other works of this thesis, particularly the chapter dedicated to direct approaches in [Chapter 3](#) and [Chapter 6](#). The ground truth generation pipeline developed here directly reflects the core theme of the [Chapter 6](#). Thus, this dataset serves both as a research artifact and as a practical tool to validate the direct methods introduced throughout the thesis.

Following the KITTI convention, each sequence is released in two flavors: a training version with public ground truth and a benchmarking version without. The latter supports the establishment of an online leaderboard hosted at [our website](#), where results can be submitted for evaluation using hidden ground truth data.

4.1 Related Work

Within the domain of [SLAM](#) and 3D reconstruction, several datasets have played pivotal roles in benchmarking and advancing autonomous robotics systems and algorithms. These

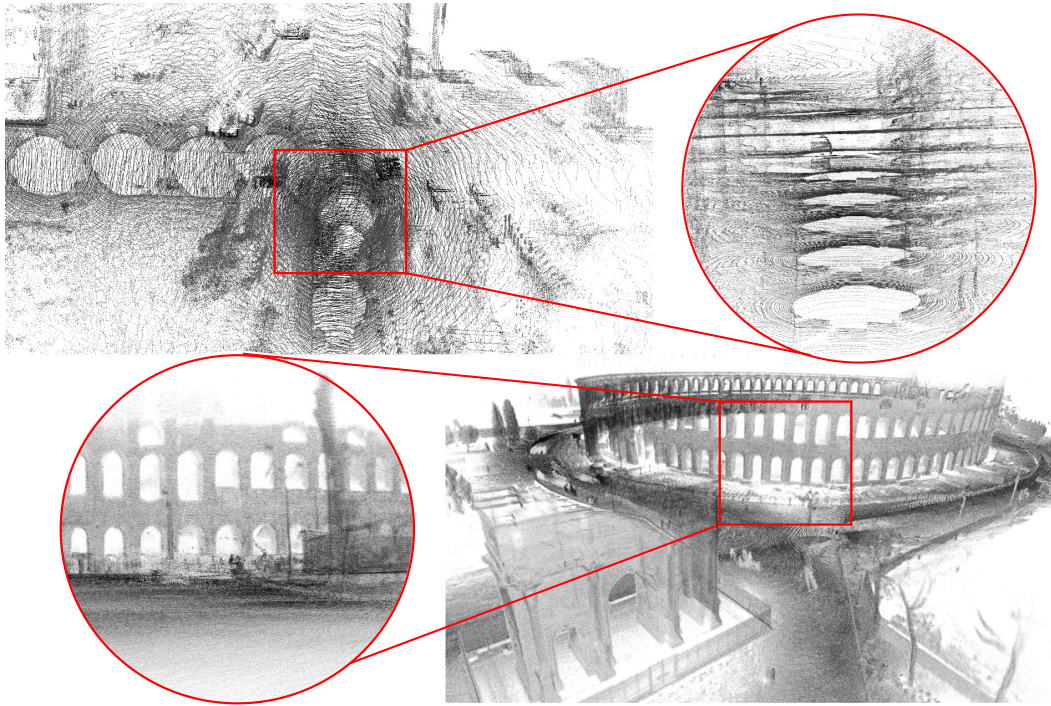


Figure 4.2. Comparison between [LiDAR](#) clouds attached to ground truth trajectories of KITTI (up) and ours (down). The zoom shows the elevation view.

datasets vary in terms of their operational contexts, sensory configurations, and the accuracy of their associated ground truth data. One of the seminal car datasets in this field is KITTI. Its strength is providing different benchmarks (*i.e.* visual odometry, optical flow, stereo matching, and object detection).

The KITTI datasets are publicly available and divided into training and evaluation sequences, fostering fair comparisons among the approaches. Despite being made available for more than a decade nowadays, KITTI is the most used benchmark in robotics perception and computer vision. However, KITTI presents synchronization issues between [IMU](#) readings and images, the ground truth data for visual odometry is produced only by fusing [RTK GNSS](#) receiver and [IMU](#), and the hardware used for recording data is nowadays outdated (Fig. 4.2, Fig. 4.3). Still, KITTI was pivotal in the development of many popular [SLAM](#) methods. Oxford RobotCar is another noteworthy car dataset [112]. In contrast to KITTI, it distinguishes itself by featuring the longest sequences among the datasets. Yet, the ground truth in the Oxford RobotCar dataset relies only on partial [GNSS](#) and [INS](#) data, which makes the baselines unreliable for benchmarking the accuracy of [SLAM](#) and localization methods. Furthermore, approaches like Mulran use the same ground truth generation process, leading to the same issue. However, this dataset is renowned for embracing multimodal sensor data, including [LiDAR](#) and [RADAR](#). While this adds diversity to the sequences, only the front half of the [LiDAR](#) field-of-view is included in the data collection [91].

In contrast to datasets collected from ground-based vehicles, certain research efforts have focused on data acquisition from micro aerial vehicles. For instance, the EuRoC dataset [28] stands out for its use of synchronized hardware and a laser tracking system to attain accurate ground truth data. However, the dataset does not provide [LiDAR](#) acquisition but only



Figure 4.3. Projection of the KITTI LiDAR point cloud into an image plane (up), projection of our LiDAR into an image plane (down). The many holes of the up image due to uneven distribution of the LiDAR beams and calibration issues make the KITTI LiDAR image unusable for computer vision tasks.

cline2-7	<i>Accurate GT</i>	<i>Indoor/Outdoor</i>	<i>Various Motion</i>	<i>Large Scale</i>	<i>Benchmark</i>
KITTI [62]				X	X
EuRoC [28]	X				
Oxford RobotCar [112]					
ETH3D [139]	X				X
MulRan [91]				X	
Newer College [129]	X				
Hilti [171]	X	X	X		X
Kitti-360 [103]	X			X	X
Ours	X	X	X	X	X

Table 4.1. The table summarizes the most important datasets in robotics perception and computer vision related to odometry estimation and SLAM. For "Accurate GT" we mean any ground truth recorded with motion capture, Laser Total Station, or globally refined.

a stereo-camera and an IMU. In addition, data is recorded only in industrial environments.

Recent advancements in handheld datasets, exemplified by Newer College [129] and Hilti [171], have achieved exceptional levels of accuracy in ground truth generation through the use of 3D imaging laser scanners. This innovative technique involves acquiring a prior map and registering LiDAR point clouds using a localization approach. Nevertheless, a notable limitation in this case is the impracticality of applying this method to large-scale scenarios, which constrains its broader utility.

This chapter introduces a diverse and heterogeneous dataset encompassing a wide range of environments. Our design accommodates various robotic platforms, including quadrupeds, quadrotors, and autonomous vehicles, making it a versatile resource for the robotics community.

We maintain hardware synchronization to ensure data accuracy and reliability and employ stereo cameras with a wide baseline to capture robust visual information. Furthermore, we provide an accurate 6-DoF ground truth even in large scale scenarios.

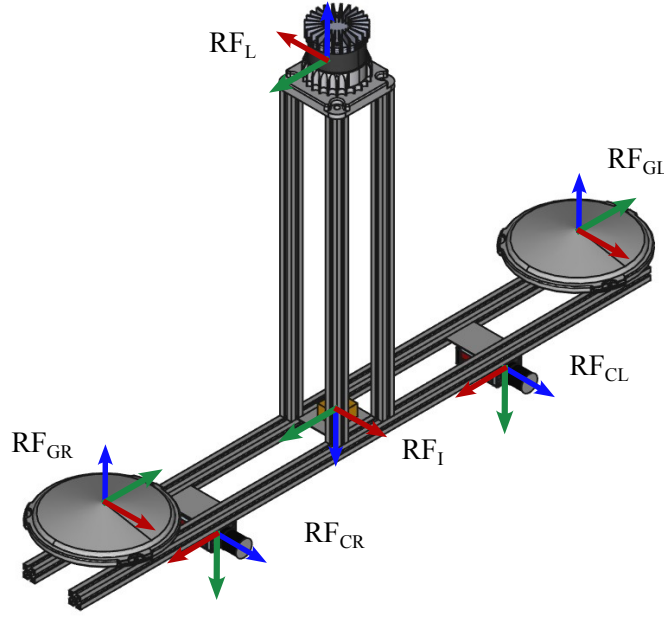


Figure 4.4. Sensor setup and reference frames. Our ground truth is expressed in the LiDAR reference frame RF_L . More details can be found in our website and supplementary materials.

4.2 The Datasets

Creating comprehensive and authentic benchmarks for the tasks mentioned earlier is challenging. These challenges encompass collecting vast data in real-time, calibrating different sensors operating at various speeds, producing accurate ground truths with minimal oversight, and choosing the right sequences and frames for every benchmark. The following section delves into our approaches to address these issues.

4.2.1 Sensors setup

Our sensor system is illustrated in Fig. 4.4 and consists of two RGBcameras, a 3D LiDAR, an RTK-GNSS, and an IMU. The cameras are two global shutter Manta G-145 capturing in RGB and arranged in a wide stereo fashion, with a baseline of approximately 50 cm. The cameras have a horizontal Field of View (FoV) of 45° and a vertical one of 40° . During the acquisition, we enable auto white-balance and auto-exposure, while maintaining a fixed focus. The maximum exposure is fixed at 20 ms. Each image is 1388×700 pixels and stored in Bayer pattern to reduce the memory footprint without losing information.

Two LiDARs were employed, tailored to the specific motion characteristics of the captured sequences. For hand-held sequences, an Ouster OS0-128 was used. This sensor offers a maximum range of 55 meters and a vertical FoV of 90° spanned by 128 beams. For car sequences, we used an Ouster OS1-64. This sensor provides a longer maximum range of 120 meters, a narrower vertical FoV of 45° spanned by 64 lasers.

The IMU is an SBG Ellipse-E IMU, with 0.05° of roll/pitch accuracy, whose firmware supports GNSS integration. The RTK-GNSS antennas are two Septentrio PolaNt-x MF GNSS antennas mounted in differential configuration. The GNSS receiver supports multi-

Sensor	Type	Details	Rate
LiDARs	Ouster, OS0-128	Vertical FoV: 90° Horizontal Res: 2048	10 Hz
	Ouster, OS1-64	Vertical FoV: 45° Horizontal Res: 1024	20 Hz
Cameras	Manta G-125B/C	Global Shutter	20 Hz
		Stereo configuration Wide baseline	30 Hz
IMU	SBG Ellipse-E	GNSS synchronization 0.05° precision	100 Hz

Table 4.2. Summary of the devices in our sensor setup, and the accuracy of the temporal synchronization.

frequency GPS, GLONASS, Galileo, BeiDou, QZSS, NavIC, Compass and L-band signal reception with an accuracy open-sky condition of 0.6 cm horizontally and 1 cm vertically. All sensors are rigidly attached to an aluminum frame. The relative position of the sensor is the same for both the hand-held datasets and for the driving datasets. Fig. 4.1 shows the sensor placement on the car. Tab. 4.2 summarizes the devices used in our system.

4.2.2 Calibration

The accuracy of intrinsic and extrinsic sensor calibration is fundamental in achieving dependable ground truth data. Our calibration process is outlined below.

Initially, we calibrate the stereo camera intrinsically and extrinsically. Subsequently, we determine the $\text{SE}(3)$ parameters that connect the coordinate systems of the laser scanner within the right camera. Finally, we align the LiDAR/camera system with GNSS/IMU reference frame. To calibrate the camera’s intrinsic and extrinsic parameters, we use an A3 checkerboard. Keeping the camera steady, we move the checkerboard and detect its corners in the calibration images. Minimizing the average reprojection error allows us to find optimal parameters for our setup [20]. Using the same target, we estimate the rigid transformation between the right camera (RF_{CR}) and the LiDAR. We achieve accurate results by minimizing plane-to-plane error [64]. A comprehensive description of this method is provided in Chapter 5. For each recorded sequence we acquire the calibration data which are public available.

Determining the relative pose between the GNSS/IMU and the LiDAR relies on the sensor systems’ motion, since the two devices cannot observe a common target. To this extent, we recover a trajectory from the LiDAR/camera system using a LiDAR odometry based on point-to-plane ICP. During the process, we ensure a wide range of orientations and translations essential for addressing the minimization issue. This technique is known as *hand-eye* calibration [46] and aims at computing the sensor offset that results in the maximum overlap between two LiDAR trajectories: the one computed by the odometry, and the one obtained by computing the LiDAR motion from the GNSS measurements (after applying the estimated offset).

4.2.3 Synchronization

A key challenge during acquisition involves synchronizing sensors to establish a shared temporal reference. Within our platform, two separate subsystems are at play: one comprises the LiDAR and RGB stereo pair, while the other includes the GNSS receiver and IMU.

In the first system, the **LiDAR** takes on the role of the master, generating synchronization pulses during its acquisition phase based on angle data from its encoder. For hand-held use, the **LiDAR** records at 10 Hz, and the synchronization pulse activates every 120 degrees, resulting in a 30 Hz signal. Moreover, in the automotive setup, the **LiDAR** operates at 20 Hz, with the synchronization pulse set to trigger once per revolution. The signal triggers the frame acquisition for both cameras, leading to sub-millisecond synchronization between the frames. Once the frames are received, their timestamp is overwritten with one of the **LiDAR** data packets received when the encoder was at the trigger angle. This ensures an accurate hardware synchronization between the two sensors.

In contrast, the **GNSS-IMU** system relies on Pulse Per Second (PPS) protocol for synchronization, which is directly addressed by the **IMU** firmware. The streams from the two subsystems are merged together by performing an offline time synchronization to determine the difference between the internal clocks of **GNSS** and **LiDAR**. We exploit the internal **LiDAR IMU** measurements to compute the temporal shift respect to the **IMU**, using cross-correlation. This technique allows us to obtain a maximum of 5 ms error considering possible un-observable phase shift between the **IMU**s signals that come every 10 ms. A temporal drift also affects the internal clocks of the **LiDAR** and **GNSS**. In the longest sequences, we observed a maximum shift between the two clocks of about 10 ms, which is neglectable compared to the scan/image frequency and the velocity of the sensor.

4.2.4 Ground truth generation

Generating accurate trajectories is the main objective of this work; hence, major attention was dedicated to this task. Some work produces very accurate ground truth using **ICP** localization within 3D Total Station reconstruction ([129], [171]). This is not always possible when moving in a very large environment. In such cases, a **GNSS RTK** system is usually used. These systems can reach an accuracy of a few centimeters, but the signal quality is not always optimal. In addition, these systems provide good global estimation, but poor locality.

In the remainder, we assume a 3D pose $\mathbf{X} \in \mathbb{SE}(3)$ be represented as a homogeneous 4×4 matrix where $\mathbf{R}$ is the rotation matrix, and $\mathbf{t}$ is the translation vector.

We combine the **GNSS** priors $\mathbf{Z}_t^g \in \mathbb{SE}(3)$ with a variation of the **BA** formulation proposed in [44], which will be described in detail in Chapter 6. This allows us to combine the global accuracy of the **GNSS** with the local precision of registration approaches. Our procedure works first by computing a **LiDAR** odometry that expresses the relative transform $\mathbf{Z}_{t,t+1}^{\text{sm}}$ between subsequent frames. To this extent, we use the same **ICP** algorithm used for temporal synchronization mentioned in Sec. 4.2.3. This odometry is accurate, reliable in the short term, and can cope with **GNSS** outages. Subsequently, we determine a global alignment of all the poses using the **RTK-GNSS** readings and considering the incremental measurements of the **LiDAR** odometry. In short, we solve the following optimization problem:

$$\begin{aligned} \mathbf{X}_{1:T}^* = \underset{\mathbf{X}_{1:T}}{\operatorname{argmin}} \quad & \sum \|\operatorname{Log}(\mathbf{Z}_{t,t+1}^{\text{sm}-1} \mathbf{X}_t^{-1} \mathbf{X}_{t+1})\|_{\Omega_{t,t+1}^{\text{sm}}}^2 \\ & + \sum \|\operatorname{Log}(\mathbf{Z}_t^{g-1} \mathbf{X}_t)\|_{\Omega_t^g}^2 \end{aligned} \quad (4.1)$$

Here $\|\mathbf{v}\|_{\Omega}^2 = \mathbf{v}^T \Omega \mathbf{v}$ denotes the Omega L2 norm of a vector $\mathbf{v}$, with Ω representing the information matrix encoding the accuracy of the measurement. Accordingly, in Eq. (4.1) Ω_t^{sm} is the information matrix resulting from scan matching, while Ω_t^g encodes the GNSS accuracy.

Once this process is completed and we have a reasonable initial guess, we look for the set of poses that is maximally consistent with all LiDAR scans. We solve the following problem using geometric and photometric error terms. The geometric part based on point-to-plane results in a configuration of the rigid motion which is close to the optimum. The second photometric step, increases the accuracy by ensuring subpixel consistency. Let $\langle i, j \rangle$ be the set of poses, $\langle k, l \rangle$ geometric associations, and u the image pixel generated by spherical projecting the LiDAR point cloud into an image (as illustrated in Sec. 2.1.2), and discussed more in detail in Chapter 6. The total residual can be expressed as follows:

$$E^{\text{ba}} = \sum_{i,j,k,l} \rho_{\text{geo}} \|e_{k,l}^{\text{geo}}\|_{\Omega_{\text{geo}}}^2 + \sum_{i,j,u} \rho_{\text{photo}} \|e_u^{\text{photo}}\|_{\Omega_{\text{photo}}}^2. \quad (4.2)$$

We employ a point-to-plane metric for the geometric error term, explicitly relying on efficient KD-tree data association based on Principal Component Analysis (PCA) splitting criteria. The geometric term can be compactly written as:

$$e_{k,l}^{\text{geo}}(\mathbf{X}_i, \mathbf{X}_j) = (\mathbf{X}_i \mathbf{p}_{i,j} - \mathbf{X}_j \mathbf{p}_{k,l}) \cdot (\mathbf{R}_i \mathbf{n}_{i,k}) \quad (4.3)$$

with $\mathbf{p}_{i,k}$ and $\mathbf{p}_{j,l}$ denoting corresponding points between the poses $\mathbf{X}_i$ and $\mathbf{X}_j$, and $\mathbf{n}_{i,k}$ be the corresponding normal. The photometric term, instead, follows the formulation illustrated in [44], but relies only on range and intensity images. The overall error function to minimize, taking into account GNSS information, will be therefore

$$\mathbf{X}_{1:T}^{gt} = E^{\text{ba}} + \sum \|\text{Log}(\mathbf{Z}_t^{g-1} \mathbf{X}_t)\|_{\Omega_t^g}^2 \quad (4.4)$$

We measured the accuracy of our ground truth using a Total Station and 6 highly reflective markers disposed as a hexahedron in the scene, to lock all redundantly all degrees of freedom. Since ranges are invariant of reference frame, we measure the differences between the distances measured from the points acquired from Total Station and the one detected in our estimated map. Our 6-DoF ground truth results in ± 3 cm accuracy on a trajectory of length of approximately 1.5 Km (indoor/outdoor). Our ground truth generation process has been shown to scale well to large environments while relying only on the onboard sensors. The final estimated global clouds are usually in the order of billions of points.

We release the ground truth for each training sequence, always expressed in the LiDAR reference frame RF_L .

4.2.5 Data selection

In the context of this research study, the OS0-128 LiDAR system offers extensive capabilities for collecting spatial data. Specifically, it delivers precise range measurements across the entire horizontal plane, covering large distances. Furthermore, it employs an dense array of vertical beams distributed over a 90° , enabling comprehensive scans of a spherical area surrounding the sensor.

Given the nature of the chosen environments and to ensure a rich and diverse dataset, we used various configurations provided by the LiDARs with varying resolution and frequency. We used the OS1-64 for car sequences at 20 Hz to maximize the observation in wide scenarios and to reduce the skewing effect at higher speeds. Moreover, we employed the OS0-128 for hand-held sequences at 10 Hz to maximize the observation in narrow scenarios.

We provide 6 datasets split into different sequences. Among the datasets, 4 were acquired by walking using the hand-held device, while the other 2 collected by car. Each sequence was collected in a different environment, with different challenging scenarios such as dynamics, traffic, long sequences, and wide areas. Tab. 4.3 summarizes some parameters for the sequences and provides some illustration.

In the remainder, we shortly review each sequence, describing the scenario:

Spagna this sequence has been acquired in *Piazza di Spagna* and in the nearby streets. It features several large loops going up and down the stairs hence, the trajectory is non-planar. The narrow streets limit the FoV of the LiDAR, but the building facades are a rich source of structure.

Colosseum consists of two rounds around the *Colosseum* and the *Arco di Costantino*. The range of the LiDAR is not always sufficient to capture vertical structure. In some cases, the maximum range of the sensor is not sufficient to measure the entire surroundings, and the environment is repetitive, making some chunks difficult for state estimation.

Pincio several loops were collected in *Villa Borghese*. This dataset is characterized by rich vegetation and a repetitive environment.

DIAG this sequence is a mixed indoor/outdoor dataset. We traveled inside and outside our building, walking inside the corridors, through the courtyard, up to the stairs, and on the roof, which is the only part where the reception of RTK-GNSS was available.

Campus in contrast to all other sequences, this has been acquired at the main Campus of Sapienza University using the equipped car, and features several loops, spanning approximately all the streets that can be traversed by car. There are several narrow passage and some tunnels that pass under buildings. The dynamic is composed mainly of people walking composing a low percentage of the recorded data.

Ciampino these sequences have been recorded in the city of Ciampino (Fig. 4.5). It is the longest sequence so far: the length of the total trajectory is about 21 km, while being subject to moderate dynamics.

4.3 Benchmark

For a detailed assessment of SLAM and odometry estimation, we concentrate on the ATE and Relative Pose Error (RPE). As in [62], we assess rotation and translation errors independently rather than merging them into one metric.

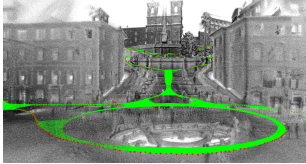
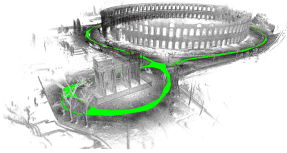
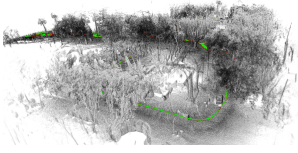
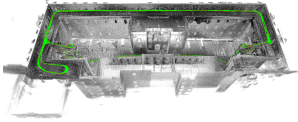
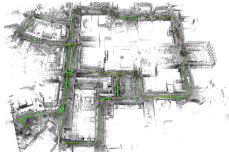
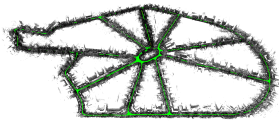
Dataset	Detail
Name: <i>Spagna</i> Motion: <i>hand-held</i> Type: <i>urban/vertical</i> Length: 2.045 km Duration: 1827 s	
Name: <i>Colosseum</i> Motion: <i>hand-held</i> Type: <i>urban/dynamics</i> Length: 2.159 km Duration: 1383 s	
Name: <i>Pincio</i> Motion: <i>hand-held</i> Type: <i>park/trees</i> Length: 2.541 km Duration: 2064 s	
Name: <i>DIAG</i> Motion: <i>hand-held</i> Type: <i>outdoor/indoor</i> Length: 1.480 km Duration: 1458 s	
Name: <i>Campus</i> Motion: <i>car</i> Type: <i>urban, underpasses</i> Length: 11.455 km Duration: 2290 s	
Name: <i>Ciampino</i> Motion: <i>car</i> Type: <i>urban/traffic</i> Length: 21.064 km Duration: 3688 s	

Table 4.3. Summary of our dataset sequences.



Figure 4.5. The image shows the overlay of the 3D model obtained from our ground truth system and a view from Google Maps.

[ATE](#) emphasizes [SLAM](#) performance over odometry. To compute its [RMSE](#), we first align the estimated trajectory with the ground truth using a $\text{SE}(3)$ transformation, matching poses with synchronized timestamps and employing the Horn method [79]. Subsequently, we calculate the [RMSE](#) of the [ATE](#) [m] among all the matched poses.

On the other hand, [RPE](#) emphasizes the odometry comparing local motion estimate chunks within the ground truth. It involves computing the [RPE](#) [%] (measured in percentage), over a set of subsequences of different lengths, as proposed by [62]. Afterwards, the translational part of the [RPE](#) [%] and the rotational part [deg/m] of a sequence are computed as the average of all chunks [RPE](#). Differently than any other benchmark, we have chosen to make chunk lengths adaptive to the total sequence length. In fact, local and global accuracy of our ground truth trajectories allows us to choose subsequences of arbitrary lengths, without biasing evaluation results.

Given a trajectory estimate for each sequence, a cumulative error curve is computed, like those in Fig. 4.6. For a given error value on the x -axis, the y -axis shows in how many sequences a method achieves a lower error. Therefore, the method ranking is determined as the area under curve, up to the selected maximum error and, for this metric, the larger the better. This metric rewards the robustness of evaluated methods, since a successful result on a sequence usually adds much more area under the curve than slightly improving the accuracy on many sequences.

Additional information, supplementary materials, and the leading table can be accessed on our website.

4.3.1 Evaluation

To assess our recorded data’s integrity, we evaluated different **LiDAR** odometry and visual **SLAM** systems on all our training sequences, specifically focusing on three notable solutions: KISS-ICP [157], F-LOAM [158] and ORB-SLAM3 [31]. The evaluation outcomes are reported in Fig. 4.6, showing cumulative **RPE** [%] and **ATE RMSE** [m], with threshold limits set to 10 % and 10 m respectively. As expected, **LiDAR** odometry methods are more robust and accurate than visual **SLAM**, in fact both KISS-ICP and F-LOAM perform successfully across all our 8 training sequences. The outcomes slightly change in the **ATE RMSE** [m] graph, where the area under the curve of every method is reduced, emphasizing the challenges in estimating the egomotion with global accuracy.

In the case of the Colosseum sequence, LeGO-LOAM demonstrated superior performance when compared to F-LOAM, managing to estimate a trajectory closely aligned with the ground truth. This outcome can be attributed to the sequence’s characteristic of traversing the same area twice, which significantly reduces the potential for **SLAM** drift. Conversely, the Campus sequence captured from the car is more extensive and presents a more challenging scenario. In this case, LeGO-LOAM exhibited limitations in closing larger loops, leading to unrecoverable drift. Notably, the performance of both pipelines in this sequence was fairly similar, with reasonable **RPE** but higher **ATE**. Finally, we report trajectory plots for LeGO-LOAM, LeGO-LOAM (SLAM mode), and F-LOAM on the Colosseum and Campus sequences. These were selected to represent two distinct motion profiles: Colosseum involves hand-held motion in a dense urban environment, while Campus features vehicle-mounted data in a semi-structured setting. The visual comparisons in Fig. 4.7 highlight the accuracy and drift behavior of each method under different operating conditions.

4.4 Conclusion

In this work, we introduced the Visual Benchmark in Rome (VBR), a dataset specifically designed to support the development and evaluation of high-accuracy state estimation methods in robotics. The dataset comprises a collection of urban driving sequences that include time-synchronized **LiDAR** and camera data, together with centimeter-level accurate ground truth trajectories. Unlike many existing datasets that rely on standard **GNSS/INS** pipelines or post-processed trajectories for ground truth, the VBR dataset is constructed using a novel multi-stage sensor fusion pipeline. This pipeline combines global positioning information with precise inertial data and performs a global optimization that fuses geometric and photometric constraints exploiting **LiDAR** observations.

A central element of the VBR pipeline is the incorporation of a photometric **BA** using **LiDAR**, where **LiDAR** point clouds are projected into a sequence of images and refined by minimizing the direct error between corresponding image patches. This direct formulation, which leverages bilinear interpolation for subpixel alignment, results in a more accurate and locally consistent pose estimate, particularly in areas where traditional geometric methods may suffer due to low structure or repetitive features. While the global consistent is ensured by the **GNSS/INS** system. This method is used in this work as a means to generate accurate ground truth, it also serves as a concrete application of direct error minimization principles applied to **LiDAR** data.

This aspect of the work aligns closely with the broader objectives of this thesis, which

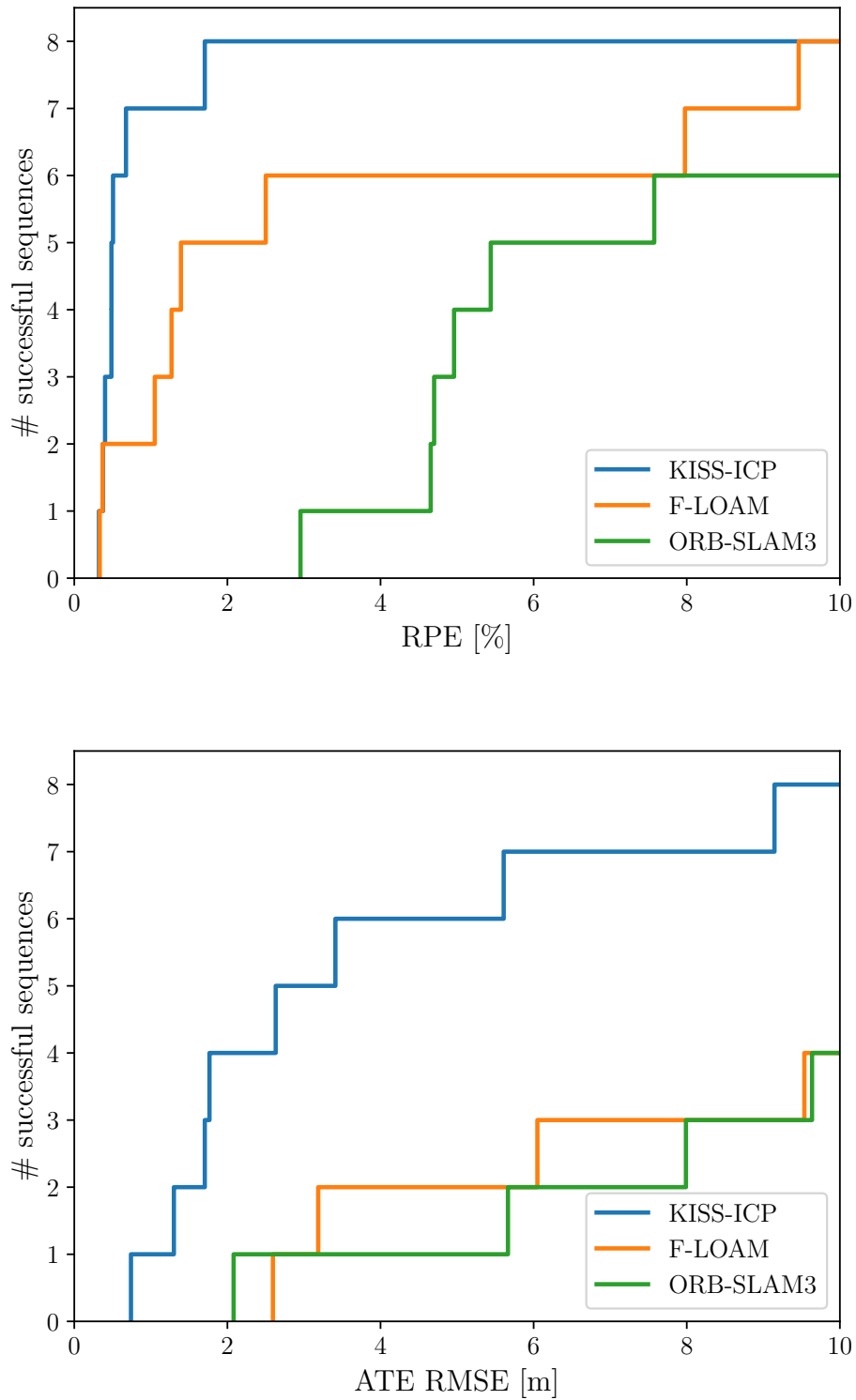


Figure 4.6. Our benchmark. Cumulative **RPE [%]** and **ATE RMSE [m]** across all training sequences in our dataset for KISS-ICP, F-LOAM and ORB-SLAM3.

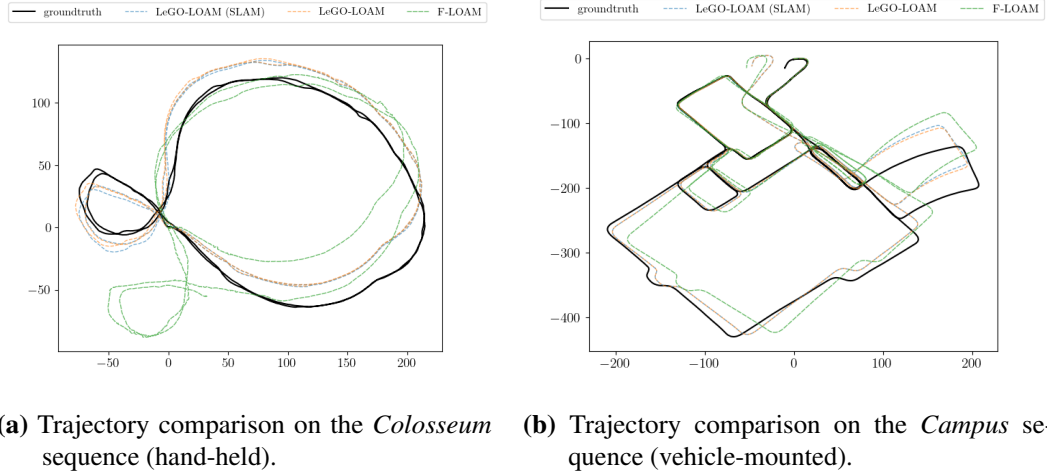


Figure 4.7. Trajectory plots of LeGO-LOAM, LeGO-LOAM (SLAM mode), and F-LOAM. Each plot is aligned to ground truth using a rigid transformation.

investigates direct methods as an alternative to traditional feature-based **SLAM** pipelines. In particular, the photometric **LiDAR** refinement process developed for VBR lays the methodological groundwork for the Chapter 6 that follows, where we formalize and extend this idea into a full photometric **BA** framework for **LiDAR**-based **SLAM**.

Moreover, VBR contributes to the thesis not only as a technical tool but also as a benchmark for evaluating the broader class of direct **SLAM** methods. The dataset’s combination of high-quality ground truth and dense sensor data makes it particularly suitable for validating the performance of direct optimization pipelines that operate on raw measurements. This stands in contrast to existing benchmarks, which often lack dense **LiDAR** data or rely solely on ground truth generated by **GNSS/INS** systems, making them less suitable for evaluating direct formulations. As such, VBR helps close the loop between theoretical developments in direct **SLAM** and empirical evaluation, ensuring that proposed methods can be tested under realistic and challenging conditions.

In summary, this work contributes a dataset that is both methodologically innovative and practically useful. By leveraging a direct, photometric approach to pose refinement during ground truth generation, it supports the core thesis goal of exploring sensor-agnostic, direct error minimization methods for **SLAM**. The lessons learned in this process, from error formulation and optimization to sensor calibration and synchronization, inform and motivate the subsequent chapters, reinforcing the value of direct methods not only as algorithmic tools but as a paradigm for unified multi-sensor state estimation.

Chapter 5

Ca²Lib: Simple and Accurate L-RGB Calibration using Small Common Markers

INTEGRATING [LiDAR](#) with RGB imaging systems significantly boosts the capabilities of robotic perception and state estimation. The complementary nature of these two vision sensors helps bridge the gap between geometric and visual understanding of the environment: [LiDAR](#) technology is renowned for its high-accuracy depth-sensing capabilities, which provide a reliable foundation for spatial reasoning, while RGBcameras offer high-resolution color information, enabling appearance-based scene understanding.

Within the broader context of this thesis, which investigates sensor-agnostic direct methods for state estimation and environment perception, combining [LiDAR](#) and RGBdata plays a critical role in enabling robust, multi-modal mapping pipelines. Recent advances demonstrate that these modalities can be fused through *guided depth completion* to produce unified dense RGB-D representations, widely adopted in downstream perception applications such as 3D reconstruction, object detection, and semantic segmentation [107, 163, 84, 122]. Moreover, in the domain of 3D reconstruction, coupling [LiDAR](#) and RGBdata has shown to enhance both the robustness and accuracy of the recovered scene geometry [43].

Despite their complementarity, integrating [LiDAR](#) and RGBsensors in practice introduces significant challenges due to their inherently different sensing modalities. This challenge stems from the need to align and synchronize data streams with different modalities. A core technical problem lies in determining the relative spatial transformation between these sensors, referred to as extrinsic calibration. This process requires extracting and matching meaningful features from their respective observations, which is complicated by differences in resolution, perspective, and sensing principles.

Current approaches often rely on artificial calibration targets, extending methods used in mono-modal systems (such as multi-RGB calibration with checkerboards [21]) to the multi-modal case. A calibration target typically refers to an object with known geometric and/or visual properties that can be reliably detected by both sensors. Extracting a set of mutually visible, well-localized features from these targets is typically sufficient to determine the extrinsic parameters.

However, in the [LiDAR](#)-RGB case, identifying such common features is non-trivial. Traditional visual markers like checkerboard corners, while easily detected in camera images,

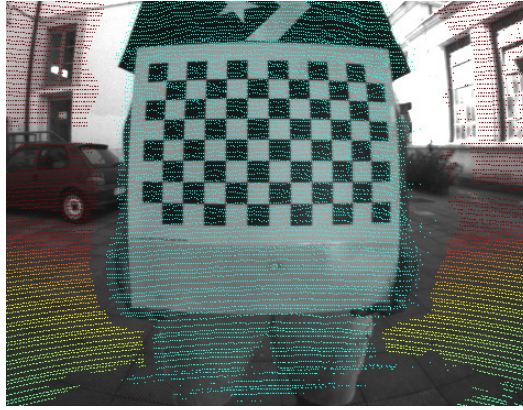


Figure 5.1. Reprojection of a *LiDAR* point cloud on a fisheye RGB camera rigidly attached to the former. The offset between the sensors leads to shadows on parts of the image.

are usually invisible or ambiguous in *LiDAR* data due to sparse point returns and low angular resolution. This necessitates the exploration of more complex features. One example is the use of holes with known dimensions: a *LiDAR* can infer the hole center by capturing points on its perimeter, while the camera can apply classic circle detection methods [80] to estimate the same feature. Although effective, these solutions typically require ad-hoc markers that are often expensive to manufacture (e.g., using CNC machining) and cumbersome to transport or deploy in field conditions.

Calibration in robotics is frequently performed on-site, especially for mobile or field-deployed systems. This makes portability, ease of use, and affordability of calibration targets critical considerations. Moreover, frequent recalibration may be required during development or after sensor realignment, further reinforcing the need for practical and repeatable solutions.

In this chapter, we present a calibration method that relaxes the need for specialized targets by enabling the use of standard planar patterns, such as checkerboards or ChArUco boards [61], which are already widely adopted for RGB camera calibration. The proposed approach builds upon robust planar feature extraction and employs a direct, non-linear optimization framework to estimate the extrinsic transformation between a *LiDAR* and a camera. Remarkably, high accuracy can be achieved with as few as three observations per sensor, supporting a lightweight and flexible calibration workflow.

An open-source implementation of the proposed method is available at <https://github.com/rvp-group/ca2lib>, facilitating reproducibility and integration into broader perception systems. This work contributes to the thesis by providing a modular calibration pipeline that strengthens the integration of heterogeneous sensors—an essential step toward building reliable, sensor-agnostic state estimation frameworks. This calibration method is the one we used to calibrate *LiDAR* and cameras for the dataset presented in Chapter 4.

5.1 Related Work

This paper’s section delves into *LiDAR*-RGB calibration and explores the two main classes of approaches: *target-based* and *target-less*. As the name suggests, target-based approaches

require the user to place artificial markers that both the camera and LiDAR can easily detect. This contrasts with target-less methods that free the user from this task. The core idea of calibration is common in the two approaches: computing common features between heterogeneous measurements and estimating the transformation that minimizes the distance between corresponding features.

First, an overview of *target-less* approaches is presented: Pandey *et al.* presents an automatic data-driven approach based upon the maximization of mutual information between the sensor-measured surface intensities [121]. Using different calibration parameters, the authors exploit the correlation coefficient for many scan-image pairs' reflectivity and intensity values. However, shadows of objects or colored surfaces that completely absorb infrared lights might result in a weaker correlation between scan-image pairs. Yoon *et al.* proposes a calibration method using region-based object pose estimation. Objects are segmented in both measurements, and then a 3D mesh is generated from the LiDAR measurements, while images are used to reconstruct the scene using SfM. The two models are then registered together to acquire an initial guess on the relative pose. The final solution is obtained iteratively by finding correspondences between the reconstructed objects from both measurements [166]. Bai *et al.* introduces an approach for calibration that relies on parallel lines commonly visible in urban environments (*i.e.*, edges of buildings)[11]. The relative pose estimate is obtained by aligning the directions of the lines observed by the two sensors to find the relative orientation and then by solving a set of linear point-on-line constraints to find the relative translation. In recent years, the development of learning-based methods has also spanned in this field: Lv *et al.* proposes a real-time self-calibration network that predicts the extrinsic parameters by constructing a cost volume between RGB and LiDAR features [110], while Sun *et al.* first estimates an initial guess by solving a hand-eye calibration method [150]. Moreover, the guess is fine-tuned by segmenting the image-cloud pair and aligning the distances between centroids. The advantage of the target-less method is that it can be used without preparing the environment. This comes at the cost of lower accuracy and robustness when compared to their target-based counterpart. *Target-based* methods estimate the relative pose using an observed known structure. Given the difference in resolution for the two sensors, it is highly unlikely that correspondences within the measurements can be established directly. For this reason, *point-to-point* methods tend to process LiDAR measurements to implicitly obtain virtual points, *i.e.* points that are not explicitly detected, but estimated from the LiDAR measurement, which are easily detectable from an RGB sensor. For instance, Park *et al.* utilizes a specially designed polygonal planar calibration board with known lengths of adjacent sides [123]. By estimating the 3D corresponding points from the LiDAR, vertices of the board can be determined as the meeting points of two projected sides. The vertices, along with the corresponding points detected from the color image, are used for calibration. Pusztai *et al.* introduces a methodology that utilizes cubic objects with predetermined side lengths [127, 126]. The corners of the cubes are estimated by initially detecting each side of the box and subsequently determining their intersection points. Furthermore, the corners and their corresponding RGB image are employed to calibrate the system by solving ICP. Zhou *et al.* propose a single-shot calibration method requiring a checkerboard [175]. The target is detected both in the RGB image and LiDAR measurement, using RANSAC[53] for the latter. Furthermore, the four edges of the checkerboard are estimated and aligned to compute the relative offset between the two sensors. Grammatikopoulos *et al.* uses a custom-made retro-reflective target paired with an AprilTag[120] fiducial marker to establish correspondences on the center of the target[69].

The relative pose is optimized by solving a Perspective-n-Points (PnP) problem. Tóth *et al.* introduces a fully automatic calibration technique that leverages the utilization of spheres, enabling accurate detection in both point clouds and camera images [154]. Upon successful detection, the algorithm aligns the set of sphere centers using Singular Value Decomposition (SVD). Beltrán *et al.* presents a methodology that utilizes a custom calibration target with four holes and Aruco markers specifically designed for monocular detection [17]. The methodology employs a set of techniques for each sensor to estimate the center points of the holes. Subsequently, the relative offset between sensors is determined by aligning the set of centers obtained from each sensor, Li *et al.* adopt a similar approach while using a checkerboard with four holes [102]. Fan *et al.* propose a two-stage calibration method using an auxiliary device with distinctive geometric features [50]. The method extracts lines from images and LiDAR point clouds, providing an initial estimation of the external parameters. Nonlinear optimization is then applied to refine these parameters. In the work of Singandhupe *et al.*, the authors first extract planar information from RGB and L measurements, then, two grids of points are extracted from the computed planar patches and aligned using a customized ICP algorithm [143]. Albeit these approaches provide accurate results with few measurements, care should be taken during the estimation of virtual correspondences, as they can cause significant errors in the estimation step. Moreover, these custom targets often require precise construction or expensive manufacturing.

Another group of approaches does not directly solve the calibration problem using point-to-point correspondences but rather exploits the flatness of the target to reduce the feasible set of solutions using *plane-to-plane* constraints. Mirzaei *et al.* addresses the challenge of accurate initial estimates by dividing the problem into two sub-problems and analytically solving each to obtain precise initial estimates [116]. The authors then refine these estimates through iterative minimization. They also discuss the identifiability conditions for accurate parameter estimation.

A method closely related to our proposed approach is presented by Kim *et al.* [90], who estimate the extrinsic transformation by first computing the relative orientation using observed surface normals and SVD. They then obtain an initial estimate of the translation by minimizing the pairwise distances between corresponding planar features, followed by a refinement step using non-linear optimization using LM. While conceptually straightforward, their method decouples rotation and translation estimation, which may result in suboptimal accuracy and a higher requirement on the number of observed features to achieve reliable convergence.

Building upon this line of research, we propose a tightly-coupled, joint formulation for non-linear optimization that simultaneously estimates relative rotation and translation by minimizing a plane-to-plane alignment error. In contrast to methods that rely on specialized or logistically demanding calibration targets, our approach supports standard checkerboard and ChArUco patterns of A3/A4 size, which are easy to produce and deploy. Furthermore, our calibration framework is modular and extensible: it separates the geometric optimization from the detection phase, allowing the use of different fiducial patterns or feature detectors. The implementation also supports a variety of camera models and lens distortion profiles, enhancing its applicability to both narrow and wide field-of-view systems.

An **open-source implementation** of the method is made available as part of this thesis, promoting reproducibility and facilitating its integration into broader perception and calibration pipelines.

5.2 LiDAR-RGB calibration

In this section, we provide a detailed description of the proposed extrinsic calibration pipeline between a LiDAR and an RGBcamera. Assuming prior familiarity with coordinate transformations, camera and LiDAR models, and non-linear least squares optimization, as introduced in Chapter 2, we outline each stage of the method in technical depth. The process begins with the acquisition and preprocessing of raw sensor data, followed by the detection of planar targets and the extraction of geometric features. Subsequently, we describe the matching strategy used to associate observations across modalities and formulate the joint optimization problem that estimates the rigid-body transformation between the sensors. The final output consists of the relative pose, expressed as a 6-DoF transformation in $\text{SE}(3)$, that aligns the two sensing frames.

Plane Representation: Let $\pi = (\mathbf{n}, d)$ be a 3D plane, where $\mathbf{n} \in \mathbb{S}^2$ represents its normal and $d \in \mathbb{R}$ is its orthogonal distance concerning the origin coordinate system (visible in Fig. 5.2a). This representation is also known as Hesse normal form. Applying a transform $\mathbf{X} \in \text{SE}(3)$ to a plane π yields new coefficients π' , as follows:

$$\mathbf{X}\pi = \begin{cases} \mathbf{R}\mathbf{n} \\ d + (\mathbf{R}\mathbf{n})^T \mathbf{t} \end{cases} \quad (5.1)$$

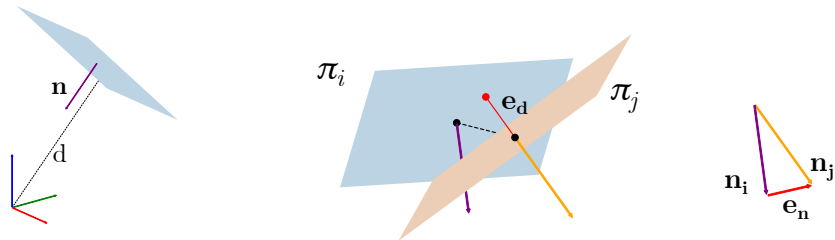
Here $\mathbf{X} = \langle \mathbf{R}; \mathbf{t} \rangle$ is represented by a rotation matrix $\mathbf{R} \in \text{SO}(3)$, and the translation vector $\mathbf{t} \in \mathbb{R}^3$.

The rotation matrix can be calculated from the perturbation vector using the *exponential map* at the identity $\text{Exp}(\Delta \mathbf{r}) \in \text{SO}(3)$. We extend the notation of the exponential map to refer to the transformation encoded in $\Delta \mathbf{x}$. If the transformation is modified by a small local perturbation $\Delta \mathbf{x} = (\Delta \mathbf{r} | \Delta \mathbf{t})$ then we can rewrite:

$$(\mathbf{X} \text{Exp}(\Delta \mathbf{x}))\pi = \begin{cases} \text{Exp}(\Delta \mathbf{r})\mathbf{R}\mathbf{n} \\ d + (\mathbf{R}\mathbf{n})^T \mathbf{t} + \mathbf{n}^T \mathbf{R}^T \text{Exp}(\Delta \mathbf{r})^T \Delta \mathbf{t} \end{cases} \quad (5.2)$$

Deriving the result for $\Delta \mathbf{x}$ leads to the following Jacobian:

$$\frac{\partial(\mathbf{X} \text{Exp}(\Delta \mathbf{x}))\pi}{\partial \Delta \mathbf{x}} = \begin{bmatrix} \mathbf{0}_{3 \times 3} & -[\mathbf{R}\mathbf{n}]_{\times} \\ \mathbf{n}^T \mathbf{R}^T & \mathbf{0}_{1 \times 3} \end{bmatrix}_{4 \times 6} \quad (5.3)$$



(a) Shows the Hesse plane representation used in this work. (b) A visual representation of the plane-2-plane error.

Figure 5.2. Here e_n is the error for the normal term while e_d represents the plane distances.

where $[\mathbf{v}]_{\times}$ maps the vector $\mathbf{v}$ into a skew-symmetric matrix defined as follows:

$$[\mathbf{v}]_{\times} = \begin{bmatrix} 0 & -v_z & v_y \\ v_z & 0 & v_x \\ -v_y & v_x & 0 \end{bmatrix} \quad (5.4)$$

The distance between two planes depends on the difference between their normals and the signed distance of the planes from the origin, as shown in Fig. 5.2b. These quantities can be captured by a 4D error vector $\mathbf{e}_p$ expressing the *plane-to-plane* error metric:

$$\mathbf{p}(\pi_k) = -\mathbf{n}_k d_k \quad (5.5)$$

$$\mathbf{e}_p(\pi_i, \pi_j) = \begin{bmatrix} \mathbf{n}_i - \mathbf{n}_j \\ \mathbf{n}_j^T (\mathbf{p}(\pi_j) - \mathbf{p}(\pi_i)) \end{bmatrix}. \quad (5.6)$$

Here $\mathbf{p}(\pi_k)$ is the point on the plane closest to the origin of the reference system, and it is obtained by taking a point along the normal direction $\mathbf{n}$ at a distance d .

Throughout the works presented in this thesis, we primarily adopt spherical projection for LiDAR data (see Sec. 2.1.2), as it preserves geometric consistency and maintains the spatial relationships between 3D points. This representation is particularly suitable for tasks involving registration and scan alignment (see Sec. 3.2.2). However, during the calibration procedure, our focus shifts toward enabling precise manual selection of the checkerboard plane within the LiDAR view. To facilitate this, we render the LiDAR data using an alternative projection method based on point identifiers (ID-based projection Sec. 2.1.2), which ensures a more stable and user-interpretable image for interaction. The calibration procedure is detailed below. First, we process the incoming raw LiDAR and RGB measurements to acquire planar information. Assuming the scene remains static throughout the acquisition of a single joint measurement, the LiDAR cloud is embedded in an image using the projection by ID. Moreover, the system awaits the user interaction to guess the position of the calibration target on the LiDAR image.

A parametric circular patch around the user's selection is used to estimate a plane using RANSAC, and concurrently, the calibration target detection is performed on the RGB image using a target-dependent method (*i.e.*, OpenCV[21]). Once the target is detected, the RGB plane is computed by solving a PnP problem with OpenCV. If the user is satisfied with both LiDAR and RGB planes, they are stored for processing.

Whereas a straightforward rank analysis of the Jacobians reveals that just three measurements are sufficient to constrain a solution, it is well known from the estimation theory that the accuracy grows with the number of measurements. From now on, a measurement represents a synchronized pair of LiDAR RGB measurements.

Once the set of measurements is acquired, we jointly estimate the relative orientation and translation of the LiDAR to the RGB sensor $\mathbf{X} \in \mathbb{SE}(3)$ by solving the following nonlinear minimization problem:

$$\mathbf{X} = \underset{\mathbf{X} \in \mathbb{SE}(3)}{\operatorname{argmin}} \sum_{i \in \mathbf{Z}} \underbrace{\|\mathbf{X}\pi_1^i - \pi_c^i\|}_{\mathbf{e}_p}}^2 \quad (5.7)$$

where $\mathbf{e}_p$ represent the plane-to-plane error.

During acquisition, the user may accept one or more wrongly estimated measurements. Due to the quadratic nature of the error terms, these *outliers* are often over-accounted,

Table 5.1. Average translation error in millimeters with different noise levels and number of measurements N .

N	$\sigma_l = 0$ $\sigma_c = 0$		$\sigma_l = 8e^{-3}$ $\sigma_c = 7e^{-3}$		$\sigma_l = 16e^{-3}$ $\sigma_c = 14e^{-3}$	
	mean	stdev	mean	stdev	mean	stdev
3	41.761	104.362	20.790	25.124	57.849	112.365
4	10.872	17.941	12.206	12.363	14.940	11.681
5	6.492	7.997	8.350	9.076	9.115	5.675
10	4.591	3.458	5.759	4.974	5.849	1.989
20	2.575	1.981	3.646	2.564	4.123	1.139
30	2.673	1.263	2.867	1.659	3.735	0.878
39	2.091	0.883	2.666	1.206	3.261	0.413

resulting in wrong estimations. We employ a Huber *robust estimator* $\rho(\cdot)$ that treats different measurements based on their error to account for this factor. We rewrite Eq. (5.7) as follows:

$$\mathbf{X} = \underset{\mathbf{X} \in \mathbb{SE}(3)}{\operatorname{argmin}} \sum_{i \in \mathbf{Z}} \rho(\|\mathbf{X}\pi_l^i - \pi_c^i\|). \quad (5.8)$$

To resolve Eq. (5.8), we employ the GN algorithm. The full pipeline is depicted in Fig. 5.3.

5.3 Experimental Evaluation

In this section, we describe the experiments we conducted to establish the quality of our calibration toolbox. We perform quantitative experiments in the simulated environment provided by [17] to compare our estimates with ground truth while we also conduct qualitative and quantitative experiments on real scenarios using our acquisition system. We directly compare our results with [90], as it is the work closest to ours. In addition, we compare to [17] that produce accurate results relying on a very complex target (CNC printed).

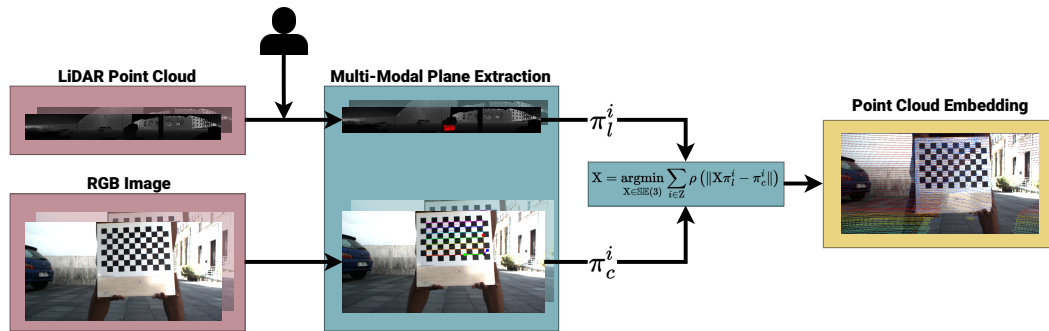


Figure 5.3. Diagram of our calibration pipeline. Measurements are acquired, and calibration target detection is performed (L planar detection is performed via human intervention). The set of planes is used to solve the non-linear optimization problem, leading to the optimal relative pose between the sensors.

5.3.1 Synthetic Case

We conducted experiments on *Gazebo* simulator [93] to evaluate the accuracy and robustness of our approach, injecting different noise figures into the sensor measurements. We also experiment with how the number of observations affects the final results. The scene setup includes a Velodyne HDL-64 LiDAR, a BlackFly-S RGB sensor, and a 6×8 checkerboard target with a corner size of 0.2 meters. We randomly generate and acquire 53 valid measurements, where a valid measurement for which both LiDAR and RGB sensor can detect the target.

To quantify the impact of the number of measurements on the accuracy of our approach, we run the calibration procedure with an increasing number of measurements $w_s = [3 \dots 39]$ and at three different LiDAR noise levels σ_1 (0 mm, 7 mm, and 14 mm). For every w_s , we sample 40 sets of measurements.

From Tab. 5.1, we observe a steady decrease of error for every noise level, reaching an average of 2.6 mm translation error in the intermediate noise case. In the case of three measurements, the high uncertainty is due to the potentially poorly conditioned system when using planes with similar normals. Nonetheless, we compare our best result with 3 measurements against the best results of the methods presented in [17] and [90]. Tab. 5.2 shows the results.

Method	e_t (cm)	e_r (10^{-2} rad)
<i>Beltrán et al.</i> [17]	0.82	0.24
<i>Kim et al.</i> [90]	10.2	129.56
Ours	0.11	0.25

Table 5.2. Quantitative results on synthetic data achieved through calibration using $N = 3$ measurements. We choose this measurement count for parity with the methodology proposed in [17]. In [17], a single measurement is deemed sufficient for parity with the methodology proposed in [17]. In [17], a single measurement is deemed sufficient for parity with the methodology proposed in [17]. In [17], a single measurement is deemed sufficient for parity with the methodology proposed in [17]. In [17], a single measurement is deemed sufficient for parity with the methodology proposed in [17]. Beyond 3 measurements, accuracy does not improve significantly. Both for our study and in alignment with the findings in [90], 3 measurements represent the minimum requirement for solution determination, and an increase in this count is expected to result in more precise outcomes. Our results show that with our minimum number of measurements, we perform on par with [17] in rotation while outperforming all methods on translation, using small commercial tags.

5.3.2 Real Case

In this section, we describe the experiments conducted on real measurements. We perform a quantitative test on our acquisition system shown in Fig. 5.4, that is the one reported also in Fig. 4.4. The system is equipped with an Ouster OS0-128 LiDAR with a resolution of 128×1024 , a RealSense T-265 stereo camera and two Manta-G145 RGB cameras arranged in a wide horizontal stereo configuration.

Since no ground truth information is available, we use the stereo extrinsic to estimate the calibration error. The offset between multiple cameras is measured using optical calibration procedures, which typically reach subpixel precision.

In the first experiment, we consider the LiDAR and the Realsense T-265 sensor, which provides two wide field-of-view fish-eye cameras with factory-calibrated intrinsic/extrinsic

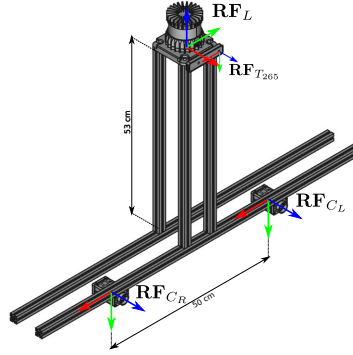


Figure 5.4. Acquisition system used for the Real Case experiments. We report nominal measurements between the sensors. The Realsense T265 (180° horizontal field of view) is installed closer to the **LiDAR**, while the two Manta cameras (90° horizontal field of view) are mounted in a further wide horizontal stereo baseline.

parameters. The task of the experiment is to demonstrate the accuracy of the calibrator in real-case scenarios and to understand how the number of measurements considered affects the quality of the solution. As for the synthetic case, we first acquire a set of 17 cloud-image **LiDAR** RGB measurements for both cameras. Moreover we perform 40 calibrations with w_s randomly selected measurements with $w_s \in \{3, 15\}$. Finally, for every w_s , we combine the computed extrinsic for both cameras to obtain an estimated stereo transform. Assuming approximately symmetrical errors in the two cameras, Fig. 5.5a shows the results of this experiment. We obtained, at best, an average error of 7.1 mm in translation and 0.01 rads in orientation.

The second experiment is conducted using the wide stereo setup for which we also calibrate the intrinsics and extrinsics of the cameras, providing expected results in a typical scenario. The large parallax between the **LiDAR** and each camera and a smaller field of view allows us to evaluate our approach in a stressful scenario. The acquisition procedure is the same as in the first experiment, and Fig. 5.5b shows the experimental result, where we obtain the best solution with 4.6 mm and 0.2×10^{-2} rads in orientation.

Moreover, Fig. 5.1 and Fig. 5.6 show the reprojection onto the right camera, respectively of the fisheye and wide baseline RGB sensor. In the latter, the large parallax between the sensors leads to strong occlusion effects that have been mitigated with a hidden point removal algorithm [86].

Our evaluation indicates that our method can generate extrinsic estimates comparable or superior to those obtained using other state-of-the-art approaches. It is important to note that careful consideration is required when selecting a minimal number of measurements. However, our experiments demonstrate that the accuracy of these estimates improves as the number of measurements increases.

5.4 Conclusion

The experiments presented in this chapter demonstrate that planar features represent a valid and robust alternative to existing methods for **LiDAR**-RGB calibration, particularly due to their resilience to the inherent noise characteristics of **LiDAR** measurements. As shown in

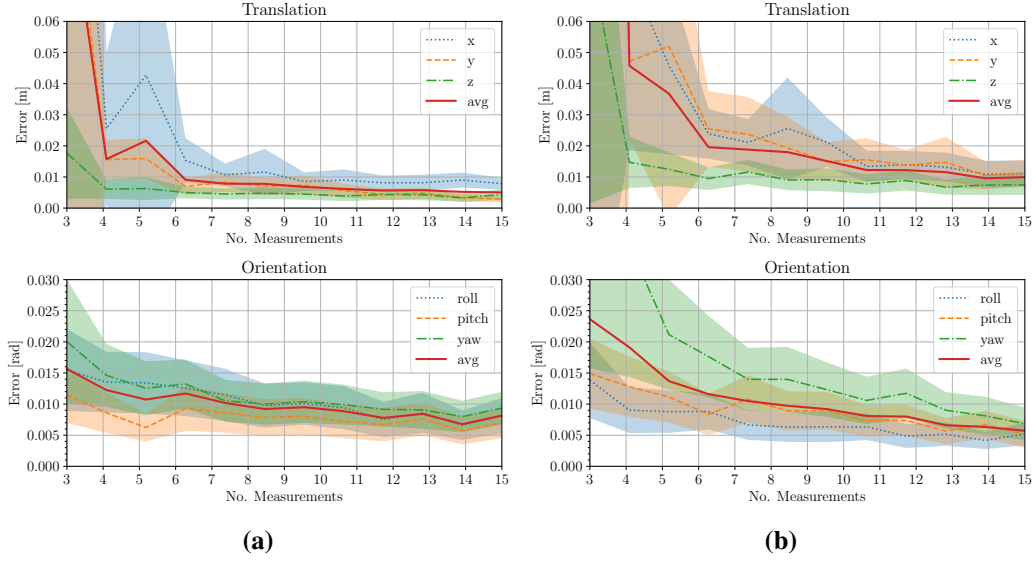


Figure 5.5. (a) Average camera-wise calibration error in the L-T265, wide FoV case. (b) shows the average camera-wise calibration error in the LiDAR-Manta case while

Tab. 5.1, the proposed approach maintains comparable translation error across varying noise levels, highlighting the stability and consistency of the calibration process under sensor uncertainty.

In addition to synthetic experiments, real-world evaluations further support our claims regarding the practicality of the method. Specifically, we show that the use of standard planar targets with dimensions reduced to A3/A4 formats remains effective for accurate extrinsic estimation. These targets are easily printable and portable, making them especially suited for field applications. Furthermore, the proposed framework enables seamless integration with different camera models, including the Kannala-Brandt model [85] used with the Intel RealSense T-265, and the Radial-Tangential model used with the Manta G-145. This demonstrates the generality and adaptability of the method across sensor configurations.

Based on these findings, we recommend the application of our method in scenarios where calibration must be performed on-site, particularly in environments that do not support dedicated calibration setups, or where transporting specialized calibration targets is not feasible. Nonetheless, several considerations must be taken into account to ensure successful calibration. One key factor is the configuration of the sensors and their shared field of view. Achieving an accurate estimate requires observing the calibration target from multiple poses, with sufficient coverage from both modalities. In cases where the overlap in field of view is limited, a single-shot calibration strategy, where all necessary information is captured from one position, might yield improved accuracy.

Another important aspect concerns the temporal synchronization of the sensor data. Our method assumes that the input measurements are time-aligned. Even small offsets between the LiDAR scan and the RGBimage can significantly degrade calibration accuracy. Synchronization is particularly challenging due to the differing operational characteristics of the sensors: LiDARs typically exhibit longer revolution periods compared to the exposure times of RGBcameras. As a practical recommendation, we suggest triggering image acquisition during periods where the LiDAR scan intersects the camera’s field of view to maximize

spatial overlap and minimize temporal discrepancy.

One possible enhancement to this work is the introduction of an automatic detection system for identifying calibration targets within the **LiDAR** point cloud. Such a system could operate directly on the 3D data, using spatial analysis to detect planar structures, or indirectly by projecting the point cloud onto a 2D embedding for image-based detection. Automating this step would significantly reduce the manual intervention required during calibration, either by fully replacing or by improving the initial guess for the current human-assisted plane detection process.

In summary, this chapter introduced a practical, flexible, and accurate method for estimating the extrinsic parameters between **LiDARs** and RGB sensors. By exploiting the inherent planarity of standard calibration targets, we enable reliable geometric alignment between these sensing modalities while simplifying the calibration process. Our experiments confirm that planar features not only mitigate the impact of **LiDAR** noise but also enable precise calibration even when using low-cost, easily available A3/A4-sized patterns. Finally, to encourage adoption and facilitate reproducibility, we release an open-source implementation of the proposed calibration pipeline as part of this thesis.



Figure 5.6. Qualitative samples showing L cloud projection on RGB image, in the setting sketched in Fig. 5.4, where the parallax between the sensor is approximately 66 cm.

Chapter 6

Photometric LiDAR and RGB-D Bundle Adjustment

SLAM has significantly advanced in robotics over recent decades, enabling robust solutions for accurate localization and environment reconstruction. **SLAM** applications span diverse and demanding domains, such as autonomous driving, augmented reality, robotics navigation, indoor mapping, and planetary exploration. This wide span of applications underscores **SLAM**'s significance and motivates ongoing academic research and substantial industrial investment.

Contemporary **SLAM** systems typically consist of two primary components. The first is the front-end, responsible for real-time sensor odometry estimation. The second is the back-end, which focuses on optimizing previously estimated sensor poses and refining the consistency of reconstructed 3D maps. **BA** [153] remains the gold standard optimization method for vision-based **SLAM** and **SfM**, offering a robust and accurate mechanism for refining poses and landmark positions simultaneously.

In recent years, photometric or direct methods have emerged as a critical advancement within the **SLAM** community, especially due to their capacity to directly minimize pixel-wise discrepancies across entire image regions. Unlike traditional feature-based techniques, photometric approaches do not depend on explicit handcrafted feature extraction, circumventing the challenges associated with feature sparsity, outliers, and ambiguous matches. These direct methods achieve enhanced registration accuracy through subpixel precision and implicit data association, effectively addressing limitations inherent in feature-dependent frameworks [89, 139].

Although photometric approaches have demonstrated considerable success with various sensor types such as monocular, stereo, and RGB-D cameras, their application to **LiDAR** data remains significantly less explored. Modern **LiDAR** sensors now offer unprecedented resolution, often up to 128 channels, and generate dense point clouds at a rate exceeding five million points per second. Such advancements in **LiDAR** sensor technology provide an opportunity to extend photometric approaches beyond traditional image-based sensors, making them viable and highly beneficial for **LiDAR** data processing. By projecting **LiDAR** scans into panoramic images that capture intensity and reflectivity information, photometric methods effectively leverage these high-density point clouds, paving the way for improved mapping accuracy and robustness. [38] presented a general photometric registration methodology that extends direct approaches to different projective models and

enhances the optimization robustness by considering additional channels such as normals.

Building upon the foundational concepts introduced in Chapter 3, this thesis proposes a unified photometric BA strategy capable of seamlessly integrating RGB-D and LiDAR sensors into a coherent optimization framework. This methodology aims to refine initial trajectories derived from SLAM systems or GNSS positioning solutions by explicitly maximizing photometric consistency across multi-sensor observations. Crucially, this approach operates without the need for explicit data associations, significantly reducing computational complexity and enhancing robustness to sensor noise and environmental variability.

The proposed framework inherently supports multi-sensor integration, accommodating the combination of heterogeneous sensors to enhance robustness and accuracy. Comprehensive comparative evaluations conducted on well-established benchmark datasets validate the effectiveness and versatility of the proposed approach. Results demonstrate competitive or superior performance compared to existing state-of-the-art sensor-specific BA techniques and established SLAM algorithms. Moreover, the thesis explores and illustrates the synergistic potential of fusing RGB-D and LiDAR data, further enhancing the proposed method’s overall efficacy.

In addition, the developed strategy serves a critical role within the ground truth generation pipeline used for the dataset detailed in Chapter 4. This practical application underscores the method’s effectiveness, particularly highlighted by an exemplary reconstruction of Viterbo city-center in Italy, as shown in Fig. 6.1. The reconstruction showcases the method’s capability to generate detailed, globally consistent, and accurate maps, illustrating the feasibility of practical deployment and emphasizing its potential for further academic and industrial exploration. The complete implementation, optimized in CUDA/C++, is publicly available, thereby facilitating wider adoption and continued research.

6.1 Related Work

Approaches for global refinement such as BA are widely used in Visual SLAM and SfM systems but are less common for LiDAR. In this chapter, we provide a unified photometric global registration method for both RGB-D and LiDAR that can improve the accuracy of the trajectory, and hence the map, obtained by standard SLAM systems that rely on Pose-Graph Optimization (PGO). PGO formulation reduces the optimization problem’s size but approximates the original problem by marginalizing the projective observations.

In the RGB-D SLAM field, BA can be classified into two categories: *direct* and *indirect*. In the following, we will explain the difference between these two paradigms and review the state-of-the-art of both approaches. We finally discuss BA applications in LiDAR SLAM.

Indirect SLAM methods estimate camera poses and 3D structure by minimizing re-projection error of feature points, making them faster and less sensitive to calibration and synchronization issues [139]. State-of-the-art implementations perform windowed BA to refine the map in a neighborhood of the sensor location, and global BA is invoked upon loop closures on the entire trajectory. To keep the problem tractable, the trajectory is subsampled in a set of keyframes, and only some salient feature points are considered in the optimization [118, 92].

Direct methods directly minimize the photometric error between overlapping images without relying on feature detection and matching. The photometric error is calculated as the difference between the measured and predicted pixel intensities based on an estimate of

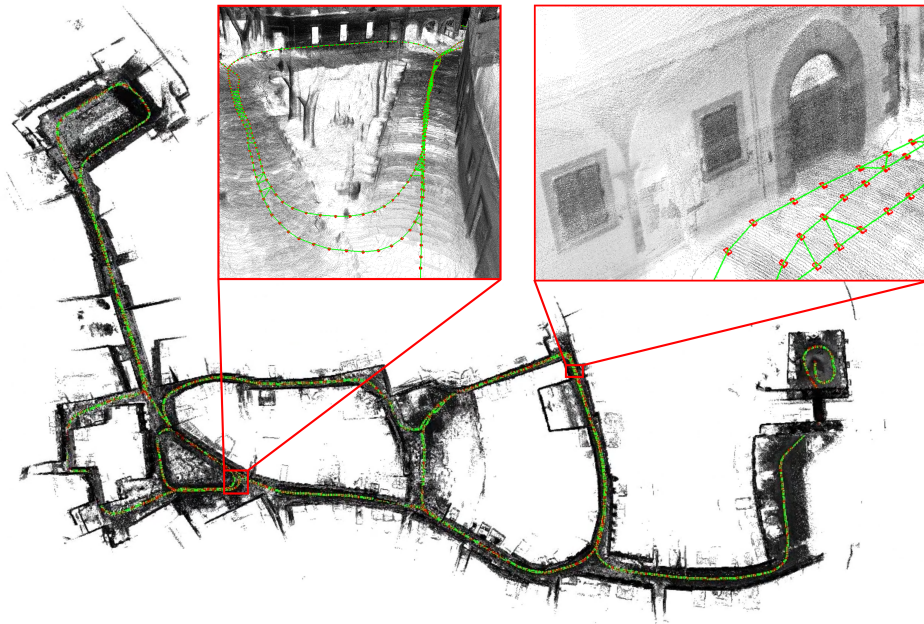


Figure 6.1. Reconstruction of Viterbo city-center (Italy) using our data recorded with an OS0-128. The trajectory, which is about 2 km long, has been estimated first with MD-SLAM Chapter 3 [67] and then refined with our photometric BA strategy. This image highlights both the global and local consistencies. We show reconstruction details with multiple scans of the same places acquired over time.

the 3D structure and camera parameters. Delaunoy and Pollefeys propose an offline dense 3D reconstruction methodology that refines the scene and camera parameters to minimize photometric error. The scene is represented by triangular meshes, which require frequent remeshing, making the approach computationally demanding [40]. Goldlücke *et al.* present a variational framework that estimates a super-resolution curved surface to precisely represent the 3D scene, leading to improved estimates and high-quality texture output [68]. Slavcheva *et al.* perform pairwise alignment by registering two consecutive Signed Distance Function (SDF)s generated from the depth images, which is used as global refinement [144]. The direct methods presented are computationally heavy, being mainly used for offline 3D reconstructions.

To enhance the computation, some researchers exploited the decomposability of the problem and investigated data reduction strategies to reduce the computation of BA in SfM applications. Hatem Alismail *et al.* consider only pixels with reasonable gradient in the error function to reduce the problem's size [7]. Eriksson *et al.* propose a consensus-based optimization to parallelize BA for large-scale problems [49]. Demmel *et al.* propose a novel solution in Distributed BA by breaking down the problem into smaller subparts using the k-means clustering method, and structuring the resulting subgraphs into well-constrained connected segments for more efficient processing [41].

In between the class of direct and indirect methods, we find the work of Forster *et al.* [56], which proposes a hybrid method to estimate the camera's motion. First, an initial guess of the sensor location is computed by minimizing the reprojection error of the world

points. Then the estimate is refined by minimizing the photometric error of the patches around the feature points. The final map refinement step is done by performing direct BA.

Recently, the community has focused on embedding BA refinement in SLAM applications. Hybrid approaches have gained traction to combine the accuracy of direct methods with the robustness of feature-based ones by mixing direct and indirect error terms in optimization strategies. One example is Bundle Fusion [39], which interleaves feature-based and photometric BA to refine the global estimate. The photometric refinement considers only camera poses and not the structure. Similarly, BAD-SLAM [139] minimizes a cost function that accounts for geometric and photometric errors in motion estimation and global refinement processes, with three main steps: refining the 3D scene modeled by surfel, optimizing camera poses with a fixed model, and refining the camera's intrinsics.

In parallel, the community approached LiDAR-based SLAM by seeking alternative representations for the dense 3D point clouds. Given the accuracy of these measurements, the robotics community addressed the problem of building a map incrementally registering new scans.

Many registration techniques have been exploited using LiDAR data. These include 3D salient features [168], subsampled clouds [155] or NDT [147]. Nowadays, LiDAR Odometry and Mapping (LOAM) is perhaps one of the most popular methods for LiDAR odometry [168, 169]. It extracts distinct features corresponding to surfaces and corners, then used to determine point-to-plane and point-to-line distances to a voxel grid-based map representation. A revised approach (Lego-LOAM) has been suggested [141], which takes advantage of a ground surface in its segmentation and optimization steps. Odometry estimation techniques, or more generally 3D point clouds registration routines, coupled with place recognition and PGO show satisfying results within LiDAR SLAM [141, 67]. This makes PGO the gold standard optimization method in LiDAR community. A pose-graph represents the trajectory, and observations between pairs of poses along the path are computed by registering the two overlapping clouds. Hence, an observation is a relative transform and potentially a covariance matrix. With this approximation, the constraints between the poses can be represented in a relatively compact manner. The optimum of a PGO is the configuration of poses that is maximally consistent with the transforms in the measurements. Albeit efficient, PGO approaches operate on approximating the original problem since pairwise measurements are computed once during the SLAM phase and never revised. Unavoidable drifts will accumulate, and wrong behavior of the place recognition might lead to inconsistent graphs as shown in [167].

In order to remove these inconsistencies, recently Liu and Zhang presented a global geometrical optimization methodology that considers cloud measurements [108]. In particular, it formulates a cost function based on LOAM features (i.e., edges and planes) and globally optimizes the trajectory to maximize the features' overlap. This approach performs a static data association based on the co-visibility of landmarks; thus, it requires a good initial guess to operate.

In recent years, the vertical resolution of modern 3D LiDARs increased, and these devices provide scans that resemble more and more dense panoramic images. This allows us to transfer results about direct global refinement from vision to LiDAR. In this work, we present a unified BA strategy that works independently for LiDAR and RGB-D in the same way. By operating directly on images, our method constantly refines the data association during the optimization; hence it is less sensitive to poor initial guesses. Our algorithm's capabilities are demonstrated through quantitative and qualitative analyses, consistently

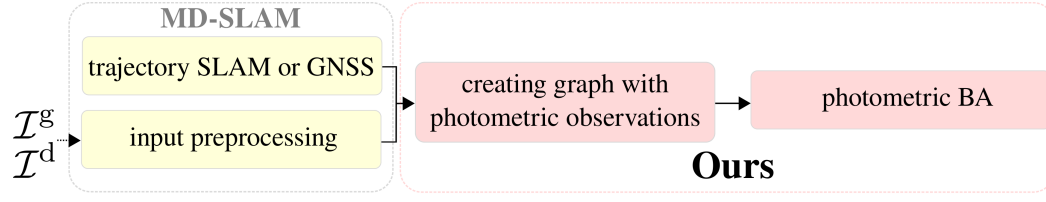


Figure 6.2. The flow of our approach. The input of our system contains the initial guess of the sensor pose, the intensity image $\mathcal{I}^g$ and the depth image $\mathcal{I}^d$. MD-SLAM already provides the input in the correct format since the preprocessing step is embedded in the SLAM system. The core of our refinement strategy, highlighted in red in this graph, consists in creating a graph with photometric observations (Sec. 6.2.1) and running a global optimization on the set of poses $\mathbf{X}_{1:N}$, as detailed in Sec. 6.2.2.

improving the initial estimates provided by any SLAM systems.

6.2 Photometric Bundle Adjustment

The goal of our approach is to compute the set of sensor poses $\mathbf{X}_{1:N}$ whose photometric error between overlapping frames is minimized. The input of our system is a set of triplets $\langle \mathbf{X}_i, \mathcal{I}^g, \mathcal{I}^d \rangle$, containing the initial guess of the sensor pose, the grayscale/intensity image $\mathcal{I}^g$ and the depth image $\mathcal{I}^d$. The workflow of our method is illustrated in Fig. 6.2. The first step is to generate an augmented image pyramid from each pair of images in a triplet, this part is equal to the previous work presented in Sec. 3.2.1. The second step is determining which pairs of images observe a common structure, which is discussed in Sec. 6.2.1. Finally, these pairs will be used to instantiate a photometric optimization problem presented in Sec. 6.2.2. Without using geometrical error functions or structure optimization, our approach is entirely free from any feature association. This makes our method simple, compatible with multiple depth sensors, and competitive with ad-hoc state-of-the-art approaches.

6.2.1 Graph Construction

Our system takes as input a collection of triplets $\langle \mathbf{X}_i, \mathcal{I}^g, \mathcal{I}^d \rangle$, that consist of the initial estimate of the sensor’s pose and pairs of images for both intensity and depth. To instantiate Eq. (6.3) we need to determine the matching pairs $\langle i, j \rangle$. The problem can be visualized as an undirected graph, where each node is a triplet, and an edge between two nodes encodes a potential match.

To compute the matches, we start from the initial assignment of poses $\{\mathbf{X}_n\}$ and add edges to the graph based on the input data. To this extent, we use a straightforward criterion that generates a matching pair if two poses $\mathbf{X}_i, \mathbf{X}_j$ are close in space, and their orientations are similar. More specifically, we create a pair between $\mathbf{X}_i, \mathbf{X}_j$ if all these conditions are satisfied:

1. the angle between the poses is below a threshold (typically 30 deg);
2. the translation between the poses is below a threshold (typically below 1 meter);

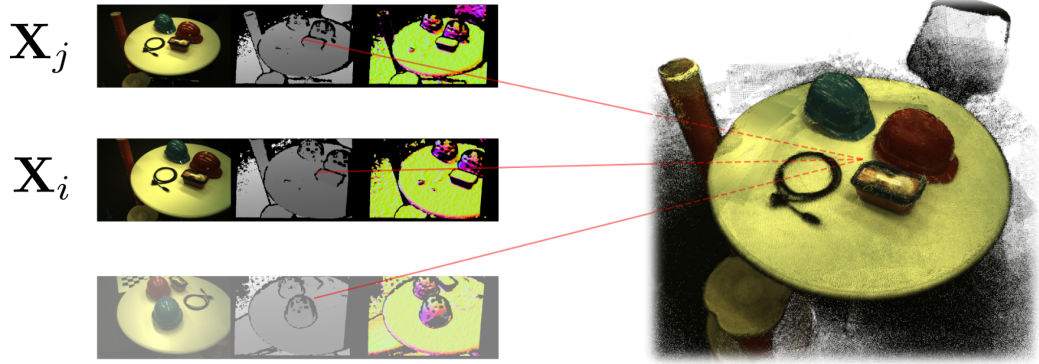


Figure 6.3. An example of pose-pairs associated, our **BA** strategy relies on the association of $\mathbf{X}_i$ and $\mathbf{X}_j$ if they share observations. In the picture above, the input images with depth and normals are on the left, and on the right is the reconstructed model using our methodology. Note that in reality, inputs of our **BA** are pyramids (as discussed in Sec. 3.2.1, i.e., images of different resolutions). Here, we show just one level for simplicity. The data used is from ETH3D.

3. the ratio of reprojected valid points from $\mathbf{X}_i$ onto $\mathbf{X}_j$ is sufficiently high (typically $1/3$).

In addition to these criteria, if the data come from a sequential acquisition, we add matches between subsequent triplets to model odometry-like constraints. An example of pose-pair associated is shown in Fig. 6.3.

6.2.2 Photometric Error Minimization

Our method seeks to find the set of transformations $\mathbf{X}_{1:N}^* \in \mathbb{SE}(3)^N$ that minimizes the photometric error between each candidate pair of sensor poses that observe a common portion of the environment. Note that the formulation is very similar to the one explained in Sec. 3.2.2, however this involves multiple variables rather the one. For clarity, we provide full details of this methodology below. Let $\mathcal{I}_i$ and $\mathcal{I}_j$ be two images acquired from poses $\mathbf{X}_i$ and $\mathbf{X}_j$ that form a matching pair. Let $\mathcal{I}(\mathbf{u})$ be the value of the pixel $\mathbf{u}$ in the image $\mathcal{I}$.

The photometric error at image coordinates $\mathbf{u}$ in the matching pair is the difference between $\mathcal{I}_i^g(\mathbf{u})$ and the pixel $\mathcal{I}_j^g(\mathbf{u}')$ of the second image. The evaluation point $\mathbf{u}'$ is computed by unprojecting the pixel $\mathbf{u}$ from $\mathcal{I}_i^g$ onto the image plane of $\mathcal{I}_j^g$. This accounts for the relative transform $\mathbf{X}_{j,i} = \mathbf{X}_j^{-1}\mathbf{X}_i$ between the two frames, as follows:

$$\mathbf{u}' = \pi \left(\mathbf{R}_j^T \left(\mathbf{R}_i \pi^{-1}(\mathbf{u}, \text{depth}) + \mathbf{t}_i - \mathbf{t}_j \right) \right). \quad (6.1)$$

To carry out this operation, the depth at the pixel $\text{depth} = \mathcal{I}^d(\mathbf{u})$ needs to be known. Standard photometric optimization seeks to find the following minimum:

$$\mathbf{X}_{1:N}^* = \underset{\mathbf{X}_{1:N}}{\operatorname{argmin}} \sum_{\langle i,j \rangle} \sum_{\mathbf{u}} \|\mathcal{I}_i^g(\mathbf{u}) - \mathcal{I}_j^g(\mathbf{u}')\|^2. \quad (6.2)$$

Here the inner summation computes the photometric error of a matching pair $\langle i, j \rangle$ as the squared norm of the error of all pixels $\mathbf{u}$ while the outer summation spans over all matching image pairs.

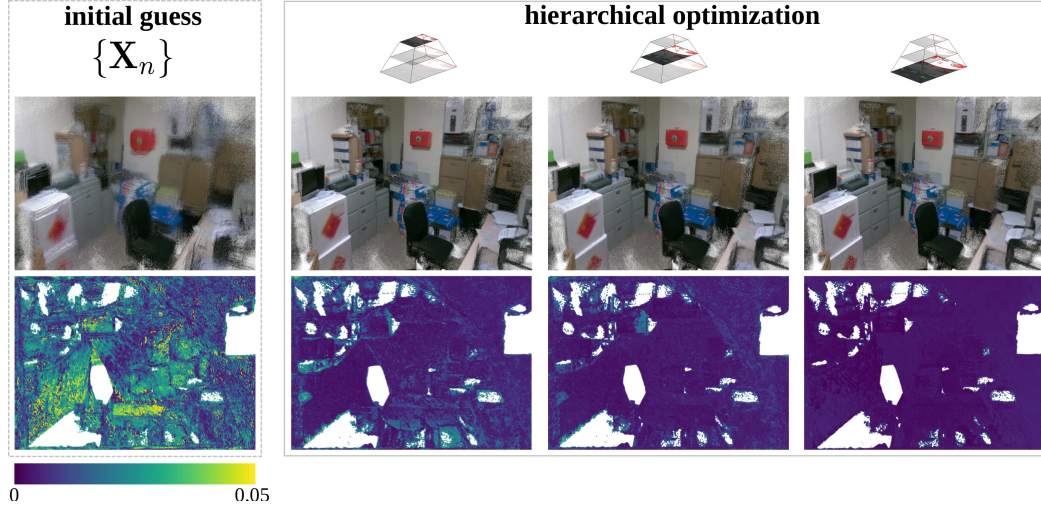


Figure 6.4. The effect of hierarchical optimization when performing BA. From left to right, we show how the quality of the estimate improves, starting from a bad initial guess. The heatmap is normalized between 0 and 0.005 [m] for better visualization. The data is self-recorded using an Intel D455.

Eq. (6.2) models classical photometric error minimization assuming that the cues are unaffected by $\mathbf{X}_{j,i}$. Whereas this is true when operating with pure intensity/grayscale values, normals, and depths change when mapped from the frame $\mathbf{X}_i$ to the frame $\mathbf{X}_j$. As in in [38] these mappings can be encapsulated by the function $\mathcal{M}^c(\mathbf{X}_{j,i}, \mathcal{I}_i^c(\mathbf{u}))$ that calculates the *pixel* value of the c^{th} cue after applying the transform $\mathbf{X}_{j,i}$ to the original channel value $\mathcal{I}_i^c(\mathbf{u})$. By extension, let $\hat{\mathcal{I}}_i^c = \mathcal{M}^c(\mathbf{X}_{j,i}, \mathcal{I}_i^c)$ be the image obtained by remapping $\mathcal{I}_i^c$, according to $\mathbf{X}_{j,i}$. We can thus rewrite a more general form of Eq. (6.2) that accounts for all cues and captures this effect as follows:

$$F(\mathbf{X}_{1:N}) = \sum_{i,j} \sum_{\mathbf{u}} \rho \left\| \sum_c \hat{\mathcal{I}}_i^c(\mathbf{u}) - \mathcal{I}_j^c(\mathbf{u}') \right\|_{\Omega^c}^2 \quad (6.3)$$

$$\mathbf{X}_{1:N}^* = \underset{\mathbf{X}_{1:N}}{\operatorname{argmin}} E(\mathbf{X}_{1:N}) \quad (6.4)$$

where ρ is a Huber robust M-estimator. The squared Mahalanobis distance $\|\cdot\|_{\Omega^c}^2$ is used to weight the different cues. In our experiment we typically set $\Omega^g = 0.6$, $\Omega^n = 0.8$ and $\Omega^d = 1$, giving more importance to depth and surface normals compared to grayscale channel. To carry on the minimization in Eq. (6.4) we employ the LM algorithm implemented in the `srrg2_solver` [71].

At each iteration, we suppress the occluded portions of the images before evaluating Eq. (6.3). The optimization proceeds by seeking the optimum of all poses, starting from the coarser level. Once convergence is reached at one level, our system switches to the next finer one, and the optimization proceeds by choosing the solution computed so far as an initial guess. Fig. 6.4 shows the effect of this hierarchical approach applied to a BA problem. Generally, the worse the initial guess, the more the required levels.

6.3 Experimental Evaluation

In this section, we report the results of our method on different public benchmark datasets. To the best of our knowledge, our approach is the only open-source photometric BA strategy that can deal with RGB-D and LiDAR in a unified manner. Therefore, to evaluate our system, we compare it with state-of-the-art SLAM and BA packages developed specifically for each of these sensor types.

To run the experiments, we used a PC with an Intel Core i7-7700K CPU @ 4.20GHz, 32GB of RAM and a Zotac Geforce GTX 1070 X 8G. Our BA schema is implemented on the GPU using CUDA 11. Since this work is focused on global consistency, we perform our quantitative evaluation using the RMSE on the ATE with $\mathbb{SE}(3)$ alignment. The metric's alignment is computed using the Horn method [79], and the timestamps are used to determine the associations. Then, we calculate the RMSE of the translational differences between all matched poses.

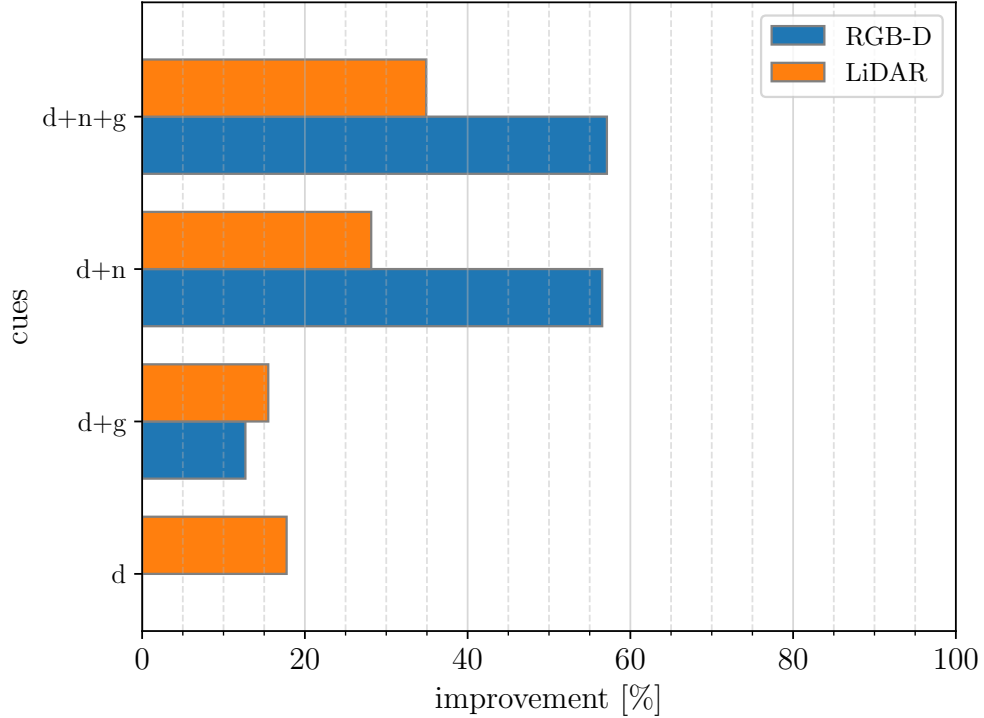


Figure 6.5. The contribution of each cue in our BA strategy. Our strategy uses three types of information (grayscale/intensity g, depth d, and surface normals n), the histogram shows the contribution of pairing each of them for RGB-D and L with respect to the initial guess in terms of ATE. Overall, using the three information together perform always better compared to coupling only few of them, for both the sensors.

In Sec. 6.3.1, we discuss the approaches and the datasets used for comparison with RGB-D sensors, while in Sec. 6.3.2 we present the results for LiDAR data. Since our implementation can run both on GPU and CPU, we report the runtimes of our algorithm for various pyramid resolutions using different commercial GPUs and our processor (Fig. 6.6).

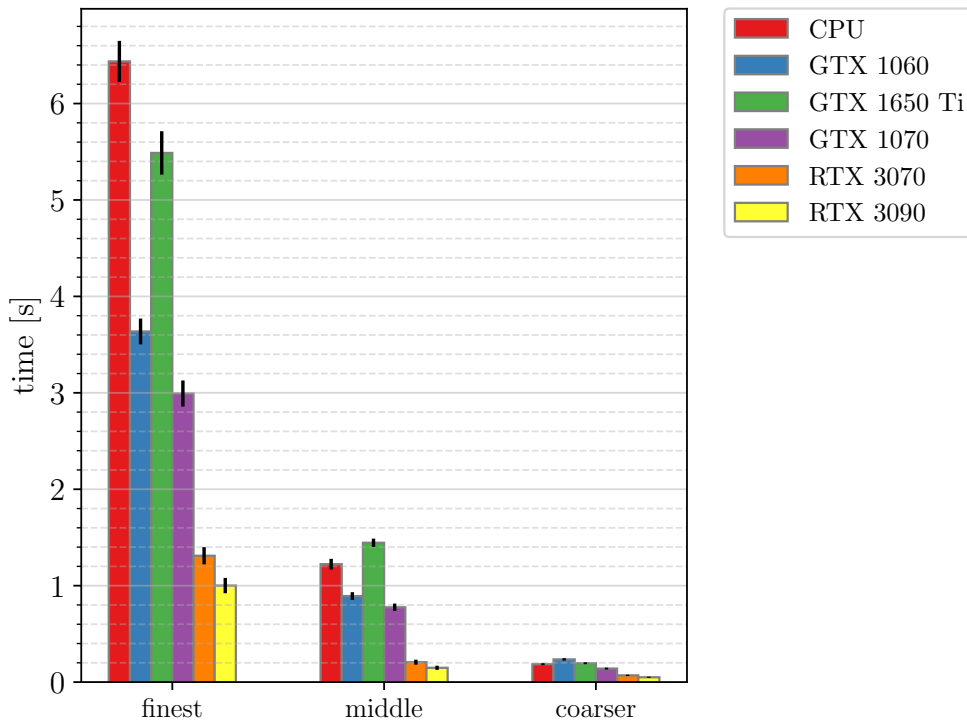


Figure 6.6. Runtimes of our BA strategy evaluated on multiple commercial GPUs and our Intel Core i7-7700K CPU. Cumulative time and standard deviation spent on an iteration of optimization. In this experiment, the finest level includes around 86M pixels, the middle level includes 22M pixels, and the coarser level has around 5M pixels. Time is expressed in seconds.

The image reports timings for each complete optimization iteration at different resolutions.

In our experiments, to switch from one level to the other, we use a simple termination criterion involving the variation of the error in Eq. (6.3) over the iterations. The number of iterations for each level differs from the quality of the initial guess. In our experiments, we tend to achieve success after 10 iterations on coarser levels, 5 iterations on middle levels, and merely a few iterations on the finest level.

In Sec. 6.3.3, we present a demonstration illustrating the impact of employing RGB-D and LiDAR simultaneously on the final estimate. Our findings clearly indicate that the use of these two sensors in conjunction, as part of our unified approach, consistently outperforms the single sensor modality.

In Fig. 6.5, we show the contribution of different cues (depth, grayscale, and surface normals) on the estimation results for both RGB-D and LiDAR. Our findings reveal that leveraging all available information enhances the performance of both sensors, with a more pronounced effect observed for LiDAR. The inclusion of surface normals plays a vital role in accurately determining the orientation of the estimate, thereby influencing the ATE. Additionally, we observe that the improvement in LiDAR performance diminishes when intensity values are combined with depth values. This can be attributed to the active nature of the LiDAR sensor, which yields contrasting results compared to RGB-D, where relying solely on noisy depth data is insufficient for achieving significant improvements.

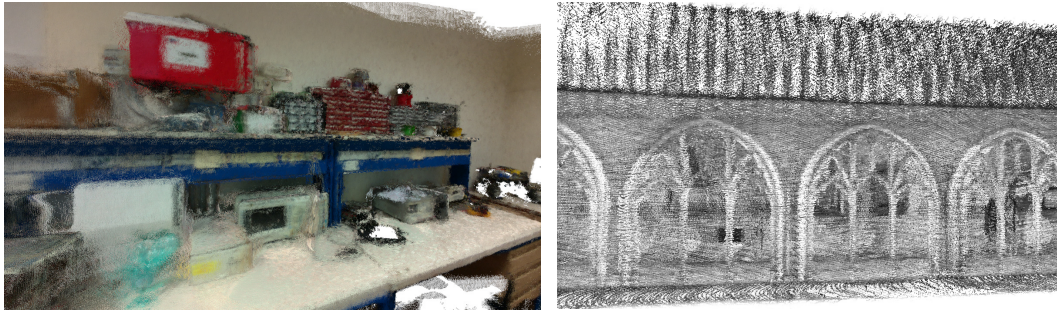


Figure 6.7. Some views of reconstructions using our photometric BA schema. On the left reconstruction of working table using an Intel L515 belonging to the RGB-D family. On the right a view of the cloister sequence of Newer College Dataset [170] recorded with OS0-128, belonging to the spinning LiDAR family.

6.3.1 RGB-D

As a public benchmark for RGB-D we used several sequences of ETH3D [139]. This dataset is acquired with global shutter cameras and accurate active stereo depth. Modeling rolling shutter effects and light changes cannot be encapsulated in the projection function $\pi(\cdot)$, our only pipeline component that differs between the RGB-D and LiDAR. For this reason, we restrict our comparison to the setting mentioned above.

The work proposed in this paper refines the map from a reasonable initial guess. We compute such initial configurations employing an improved online CUDA version of MD-SLAM [67], our previous SLAM system that unifies depth sensors through hierarchical photometric odometry estimation and feature-based loop-closures.

We compare different approaches representative of different classes of SLAM and BA algorithms specific for RGB-D sensors: DVO-SLAM[89], ElasticFusion [161], BundleFusion[39] BAD-SLAM[139], ORB-SLAM2 [118]. DVO SLAM implements a mixed geometry-based and direct registration. Internally the alignment between pairs of keyframes is obtained by jointly minimizing point-to-plane and photometric residuals. This is similar to ElasticFusion, whose estimate consists of a mesh model of the environment and the current sensor location instead of the trajectory. BundleFusion refines the global estimate by interleaving feature-based and photometric BA. Similar to us, their photometric refinement does not consider the structure, but only the sensor poses. This method highly depends on data association, employing correspondences based on sparse features and dense geometric/photometric matching. BAD-SLAM is a surfel-based direct BA SLAM system that combines photometric and geometric errors alternating optimization of motion and structure. In contrast to these approaches, ORB-SLAM2 implements a traditional visual SLAM pipeline, where a local map of landmarks around the RGB-D sensor is constructed from ORB features [136]. The map is constantly optimized as the camera moves by performing local and global BA.

Most of the compared approaches run global refinement on a separate thread in an anytime fashion. The work presented in this chapter addresses only this global aspect. Reporting the timings of this experiment would be unfair since our method addresses only a part of the problem.

ETH3D provides images of 740×460 pixels. We compute a 3-level pyramid from these images with scales 1/2, 1/4, and 1/8. In Tab. 6.1, we can see that our photometric refinement performs on par (second after BAD-SLAM) with other state-of-the-art ad-hoc RGB-D SLAM

	ElasticFusion	ORB-SLAM2	DVO-SLAM	BundleFusion	BAD-SLAM	MD-SLAM	MD-SLAM + Ours
table3	–	0.007	0.008	0.017	0.002	0.016	0.009
table4	0.012	0.008	0.018	–	0.002	0.023	0.008
table7	–	0.010	0.007	0.010	0.003	0.018	0.009
cables1	0.018	0.007	0.004	0.022	0.007	0.021	0.006
plant2	0.017	0.003	0.003	0.004	0.001	0.005	0.001
planar2	0.011	0.005	0.002	0.003	0.003	0.009	0.004
mean	0.014	0.007	0.007	0.011	0.003	0.015	0.006
std	0.003	0.002	0.005	0.008	0.002	0.007	0.003

Table 6.1. ATE RMSE [m] on ETH3D benchmark, recorded with global shutter camera and synchronous streams. ElasticFusion fails in *table3* and *table7*, BundleFusion fails in *table4*.

and BA systems. Our method reduces the trajectory error to a few millimeters. More importantly, it improves by 60% the accuracy of the initial guess provided by MD-SLAM. Fig. 6.4 and Fig. 6.3 illustrate the effect of the hierarchical optimization on self-recorded data.

Summarizing, using our general pipeline of MD-SLAM and photometric BA presented in this paper provides results comparable with other RGB-D specific approaches, being second only to BAD-SLAM.

6.3.2 3D LiDAR

To validate our approach on LiDAR measurements we used both our data and public benchmarks. We used the Newer College Dataset [170] as a public benchmark. The dataset is recorded at 10 Hz with Ouster OS0-128. More specifically, we used the *cloister*, *quad* (easy), and *stairs* sequences. The *quad* sequence contains two loops that explore the Oxford campus courtyard, *cloister* mixes outdoor and indoor scenes while *stairs* captures an indoor scenario with multiple floors. Being based on image comparison, our approach operates well on LiDAR data having a good vertical resolution. LiDARs with fewer beams (i.e., 64, 32, 16) would produce an unbalanced horizontal image, reducing the converge basin of the algorithm.

We compare our method with BALM2 [108], which, to the best of our knowledge, is the only publicly available LiDAR global refinement approach. BALM2 is based on the overall consistency of points, lines, and planes. This system requires the same input as our method, namely an initial guess trajectory and the point clouds.

To compute the initial guess, we used several SLAM algorithms specific for LiDAR: LeGO-LOAM [141], SuMA [16] and our unified MD-SLAM. LeGO-LOAM is a pure geometric feature-based frame-to-model LiDAR SLAM system, where the optimization

	LeGO-LOAM			MD-SLAM			SuMA		
	SLAM	BALM2	Ours	SLAM	BALM2	Ours	SLAM	BALM2	Ours
cloister	0.20	0.25	0.16	0.36	0.37	0.32	3.34	2.67	2.53
quad	0.09	0.09	0.09	0.25	1.98	0.17	1.74	1.72	1.68
stairs	3.20	5.13	3.48	0.23	0.38	0.10	0.67	0.86	0.60

Table 6.2. ATE RMSE [m] on Newer College Dataset, recorded with OS0-128. We always improve the SLAM baseline, a part for *stairs* starting from LeGO-LOAM estimate.

on roll, yaw, and z-axis (pointing up) is decoupled from the planar parameters. SuMa constructs a surfel-based map and estimates the changes in the sensor’s pose by exploiting the projective data association in a frame-to-model or frame-to-frame fashion.

Tab. 6.2 reports the accuracy of our method and BALM2, for each dataset, and each initial guess. The ATE of the SLAM solution measures the quality of an initial guess. We observe that LeGO-LOAM provides a good guess on all planar data but fails on the *stairs* dataset resulting in an ATE of more than 3 meters. MD-SLAM performs reasonably well with a maximum ATE of 0.36 meters, while SuMA yields an acceptable initial guess only in the stairs dataset. If the initial guess is good, both BALM2 and our method perform well, improving the initial estimate. However, as the initial guess degrades, our unified global refinement’s accuracy remains stable by systematically improving the trajectory estimate. On these data, we observed BALM2 to be particularly sensible to rotations and less dense trajectories (i.e., trajectories sampled with fewer keyframe poses). Quantitative results show that our strategy is successful within LiDAR data, raising the initial estimate close to 60% improvement when a good initial guess is provided (i.e., *stairs* with MD-SLAM).

The convergence basin of these global refinement strategies are bounded by inconsistency in the laser measurements (i.e., skewed point clouds); this is why none of the systems can further enhance the trajectory in the LeGO-LOAM *quad* experiment.

Fig. 6.1 shows our large-scale reconstruction of the historical part of Viterbo (Italy) from self-recorded data using a hand-held Ouster OS0-128 LiDAR. The trajectory is about 2 km long, and the dataset is available from our repository.

6.3.3 RGB-D + 3D LiDAR

Thanks to the unified nature of our pipeline, it allows the straightforward integration of multiple heterogeneous sensors. In Sec. 6.2.2, we formulated an optimization problem that estimates the sensor’s trajectory $\mathbf{X}_{1:N}$.

In a multiple sensors configuration, we can express the optimization problem as a function of the trajectory of a multi-device platform, to which all sensors are rigidly attached. The multiple sensors setting slightly modify the cost function presented in Eq. (6.3) to include the constant sensor offsets. The cost function for multi-sensor photometric refinement is just the sum of the cost functions of each device. In this section, we report an analysis of our optimization strategy with both RGB-D and LiDAR. For these experiments, we mounted an Intel D455 on the top of an OS0-128 and accurately calibrate the offset between the two sensors using a robust point-to-plane alignment [35]. We recorded some static indoor

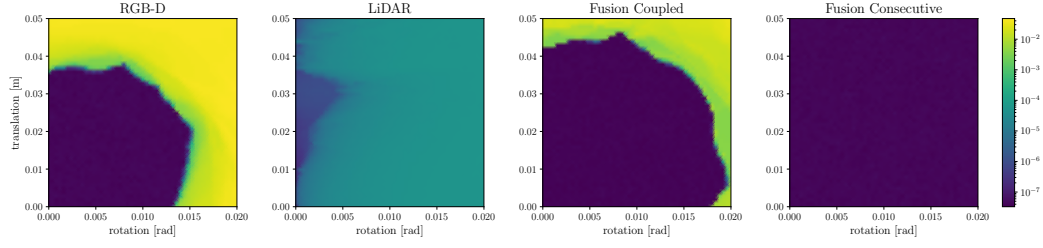


Figure 6.8. Results of fusion experiments. Plots show how the initial perturbation affects the convergence basin and the algorithm’s accuracy. On the x-y axis, respectively, translation [m] and rotation [rad] perturbations, the value is denoted by the mean between rotation and translation error in log scale. Fusing the two sensors with our straightforward approach always shows better results compared to single sensors.

sequences, each of them containing synchronized grayscale and depth images from D455 and its corresponding intensity and range images from the OS0-128.

We carried out an experiment to evaluate the accuracy and converge basin of our approach in this configuration. To this extent, we perform our photometric alignment strategy using a single sensor frame consisting of a RGB-D and a LiDAR measurement pair. Since we align a sensor frame with respect to itself, we expect the computed estimate to be as close as possible to the identity. The optimization is carried on starting from increasingly wrong initial guesses.

In Fig. 6.8, we show the result of this experiment, starting from single sensor optimization as a baseline. Due to their different characteristics, the two sensors behave very differently during photometric alignment. The 3D LiDAR has an inferior resolution but a very large 360 deg horizontal FoV. This results in a relatively large convergence basin, but the limited resolution affects the minimum. Conversely, RGB-D, provides a high-resolution image with a much smaller FoV of around 70 deg. This results in better minima at the expense of a smaller converge basin.

We can refine the data in two ways that we name *coupled* and *consecutive*. The coupled approach aims to simultaneously finding the minimum of both photometric errors of RGB-D $F^r(\mathbf{X}_{1:N})$ and LiDAR $F^l(\mathbf{X}_{1:N})$, as follows

$$\mathbf{X}_{1:N}^{\text{coupled}} = \underset{\mathbf{X}_{1:N}}{\operatorname{argmin}} F^r(\mathbf{X}_{1:N}) + F^l(\mathbf{X}_{1:N}) \quad (6.5)$$

The *consecutive* optimization combines the strengths of the two sensors, and operates by first carrying on an optimization where only the LiDAR is used. Subsequently, it uses the solution of this first run to find a minimum for the RGB-D problem.

The results suggest that fusing the two sensors always performs better than single sensors (Fig. 6.8). Specifically, *coupled* is generally more accurate compared to RGB-D only, having the converge basin incremented by the LiDAR. Similarly, *consecutive* seems to be the best since it takes full advantage of the large convergence basin of the LiDAR and benefits from the resolution of RGB-D. We believe that this proof of concept opens ways for 3D reconstruction algorithms that symmetrically fuse the two sensors, taking benefits from both.

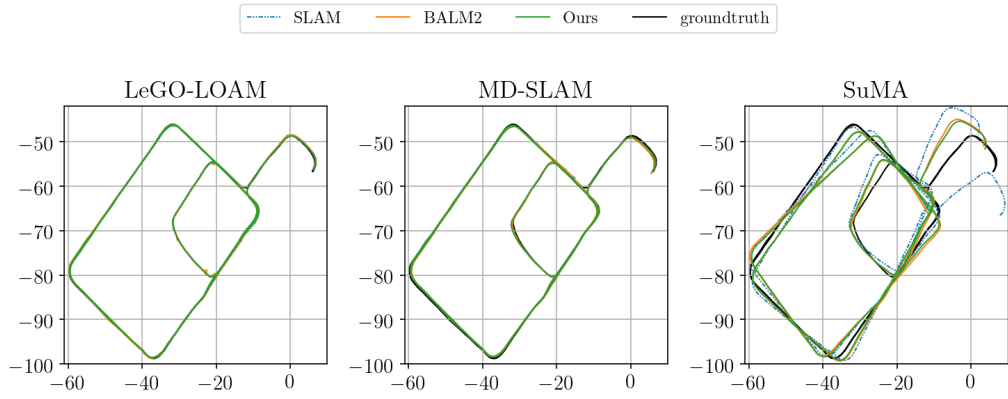
6.4 Conclusion

This chapter presented a unified photometric BA methodology capable of seamlessly refining sensor trajectories derived from RGB-D and LiDAR data. Unlike conventional approaches reliant on explicit feature extraction and data associations, our photometric optimization strategy minimizes pixel-wise difference directly, leveraging the dense intensity, depth, and surface normal information. This capability enables robust and precise refinement of trajectories and maps across different sensor modalities.

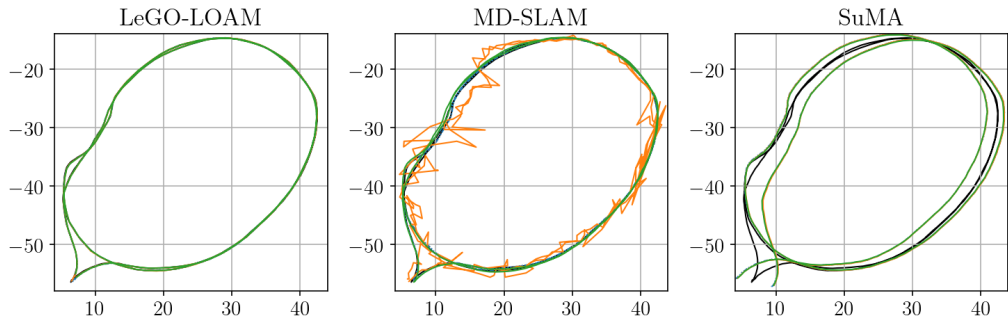
Through comprehensive experimental evaluations on publicly available datasets and self-recorded data, our approach demonstrated significant improvements over initial trajectory estimates produced by state-of-the-art SLAM systems. Specifically, the methodology achieved competitive or superior performance compared to sensor-specific BA and SLAM frameworks, underlining its versatility and robustness. The experiments also highlighted the complementary strengths of RGB-D cameras and modern high-resolution LiDAR sensors, demonstrating substantial improvements in accuracy and convergence properties when jointly optimizing both sensor types.

Furthermore, the proposed photometric BA strategy has been validated through practical application in generating accurate and globally consistent ground truth maps, as showcased by the detailed reconstruction of Viterbo city-center in Italy. The full CUDA/C++ implementation has been made publicly available, facilitating widespread adoption and enabling further research.

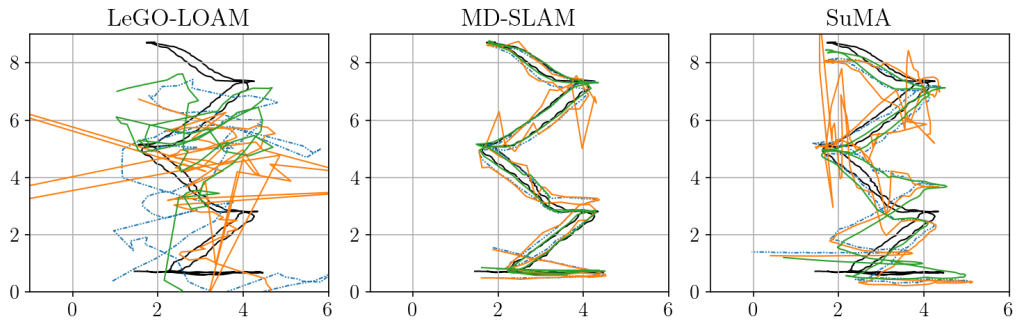
Future work aims to extend this unified framework to incorporate generic structure optimization methods, further exploiting the strengths of combined RGB-D and LiDAR data for more accurate and robust 3D reconstructions. This direction promises to enhance the adaptability of photometric approaches across diverse robotics and autonomous navigation scenarios.



(a) Trajectory estimates of *cloister* sequence. SLAM estimate in blue, BALM2 estimate in orange, Ours in green and ground truth in black.



(b) Trajectory estimates of *quad* sequence. SLAM estimate in blue, BALM2 estimate in orange, Ours in green and ground truth in black. BALM2 suffer of less dense trajectories, since MD-SLAM spawn less keyframe poses to reduce odometry drift.



(c) Trajectory estimates of *stairs* sequence. SLAM estimate in blue, BALM2 estimate in orange, Ours in green and ground truth in black. BALM2 suffer of rotations, as it is observable from MD-SLAM and SuMA initial guess. For completeness we include the LeGO-LOAM plot, however both our and BALM2 fail due to bad initial guess.

Figure 6.9. Plots of trajectory estimates in different LiDAR sequences of Newer College [170].

Chapter 7

Enhancing LiDAR performance: Robust de-skewing exclusively relying on range measurements

A significant challenge in robotic perception is the phenomenon known as motion distortion or skewing, which arises when sensors, such as rotating LiDARs, move during the acquisition of a measurement sweep. This effect results from the non-instantaneous nature of the sensor's data acquisition: each measurement in a single scan is effectively captured from a different sensor position and orientation. When neglected, the skewing effect introduces inaccuracies into the perceived environment, undermining the precision of subsequent processing steps. This issue becomes particularly pronounced in direct error minimization techniques, where high precision and subpixel accuracy convergence are crucial for algorithmic robustness and accuracy.



Figure 7.1. Effect of de-skewing. (a) Unprocessed (raw) scan acquired by a rotating robot, while (b) is the corresponding ground truth scan, and finally (c) is the same scan in (a), processed (de-skewed) using our proposed approach.

In this chapter, we specifically address motion distortion in low-cost, planar 2D LiDAR

sensors, characterized by relatively slow rotation speeds and the absence of integrated inertial sensors. Typically operating at frequencies of approximately 5 to 10 Hz, these devices lack proprioceptive feedback mechanisms, significantly intensifying the distortion effects from sensor motion during a sweep. Given the slow rotation, each complete LiDAR scan involves measurements acquired from substantially different sensor poses, causing noticeable distortion in the perceived geometry of the environment.

Our approach tackles this challenge by proposing a velocity estimation method that relies exclusively on the LiDAR range measurements, processed in a streaming manner. We employ a non-rigid plane-to-plane registration algorithm, estimating sensor velocity by maximizing geometric consistency across sequential measurements. Through comprehensive experiments involving both synthetic and real-world datasets, we demonstrate that our method effectively mitigates the impact of motion distortion, significantly improving the accuracy of the LiDAR scans with minimal computational overhead.

Critically, this foundational study on motion distortion in 2D LiDAR sensors establishes essential insights and robust methodologies relevant to a broader range of sensor types and perception techniques. In particular, addressing motion distortion becomes even more critical when working with radar sensors. Typical automotive RADAR sensors, for instance, acquire full 360-degree scans at relatively lower frequencies, commonly around 4 Hz. Due to this slow acquisition rate, RADAR scans inherently span a greater duration compared to LiDAR scans, thereby amplifying the motion distortion effect. This greater distortion profoundly affects the precision and effectiveness of direct estimation approaches, making it imperative to carefully model and compensate for motion during RADAR data acquisition.

Hence, accurately modeling motion distortion becomes not merely beneficial but essential for effectively deploying direct error minimization methods in RADAR approaches, which we will thoroughly explore and evaluate in Chapter 8. Our investigation in this chapter thus provides critical groundwork, highlighting fundamental strategies to mitigate motion distortion in 2D LiDAR data, enhancing the robustness and accuracy, and significantly advancing robotic perception capabilities.

In Fig. 7.1 we illustrate the effects of our approach on measurements acquired with a constantly rotating LiDAR, with a rotational velocity 3 rad/s.

7.1 Related Work

In this section, we review the most recent approaches to de-skew LiDAR data. Existing approaches for de-skewing mostly use either wheeled odometry or IMU to estimate the relative motion of the sensor/robot while a scan is being acquired. This process involves motion integration relying on estimators that process the raw proprioceptive data such as a filter or an integrator that directly provides an estimate of the sensor position each time a single range is measured.

These methods fall in the class of *loosely-coupled*, since they do not use the LiDAR information to refine the proprioceptive estimate. Among this class, we find the work of Tang *et al.* [151] that proposes to match subsequent scans to compute the relative motion, where the initial relative orientations are provided by an IMU. Subsequently, He *et al.* [76] proposed a method to estimate relative motion between IMU poses and de-skew subsequent ranges by using these smoothed poses.

In contrast to loosely-coupled approaches, *tightly-coupled* methods jointly process

LiDAR and **IMU**. These methods are typically based on either smoothing or filtering. Wei *et al.* [164] uses a pre-integration scheme to estimate the **LiDAR**'s ego-motion based on the inertial measurements, and updates the **IMU** biases in the correct step of an iterative **EKF** whenever a scan is completed. Shan *et al.* [142] includes the states of **IMU** in the factor-graph, updating **IMU** biases through the optimization process, adapting a well-known computer vision work [54] to the **LiDAR** case. These ideas have been further extended in [159, 165] where the authors resort to full factor-graph optimization to obtain a state configuration that is maximally consistent with all the **IMU** measurements and **LiDAR** ranges.

For coupled approaches, accurate time synchronization between the sensors is essential. Unfortunately, this is not straightforward to obtain on inexpensive small devices due to the unpredicted communication latencies that affects the usb channels. These issues, however, can be completely avoided when using only **LiDAR** data to perform de-skewing. At their core **LiDAR** only methods have a registration algorithm, which aims to compute the velocities of the sensor while the robot moves. The basic intuition is that if the correct velocities were found, the sequence of range measurements could be assembled in a maximally consistent scan. In this context, Moosman *et al.* [117] propose to linearly interpolate scans between the last known pose, and the currently estimated pose, de-skewing the scan at both poses. However, this double approximation might hinder the accuracy when the initial registration fails, resulting in even worse estimates.

Al-Nuaimi *et al.* [5] propose a weighting schema based on their adapted Geometric Algebra Least Mean Squares (LMS) solver, where they assume that consecutive relative translation and angular motion are equal, which might be inaccurate due to the existence of drift and friction.

In this work, we propose a planar **LiDAR**-only approach that addresses the de-skewing problem by continuously registering the the sequence of range measurements. The registration is carried on by estimating the **LiDAR**'s planar velocity, which iteratively minimizes a plane-to-plane metric between the de-skewed endpoints. Our results demonstrate that our methodology is accurate in estimating the velocity of the robot, only based on the laser measurements.

7.2 Velocity Based De-skewing

Our method leverages some mild assumptions about the motion of the sensor mounted on the robot and the structure of the environment to operate on a continuous stream of range measurements. More specifically we assume to have a 2D slow **LiDAR** sensor, hence, we define each measurement as $\mathbf{z}_i = \langle r_i, \alpha_i, t_i \rangle$. Here, r_i is the range, α_i is the angle of the laser beam with respect to the origin of the sensor and t_i is the timestamp. We assume the environment consists of a locally smooth surface and that the robot velocities change mildly within a **LiDAR** beam revolution.

Our method estimates these velocities by registering the stream of measurements $\{\mathbf{z}_i\}$ onto itself. The most likely velocities are the ones that if applied for de-skewing renders the measurement maximally consistent. Consistency is measured by a plane-to-plane metric applied between the corresponding **LiDAR** endpoints. More formally, let $\mathbf{x} = (v \ \omega)^T$ be the translational and angular velocity of the robot we want to estimate $\mathbf{x}^*$, such that

$$\mathbf{x}^* = \underset{\mathbf{x}}{\operatorname{argmin}} \sum_{\langle i,j \rangle \in \mathcal{C}} \rho \|\mathbf{e}(\mathbf{x}, t_i, t_j)\|^2. \quad (7.1)$$

Where $\mathbf{e}(\mathbf{x}, t_i, t_j)$ denotes an error vector between two planar scan patches computed around two corresponding range measurements at time t_i and t_j . The optimization step estimates new velocities $\mathbf{x} = (v \ \omega)^T$ under the current set of correspondences $\mathcal{C}$, using [IRLS](#). In Eq. (7.1) ρ denotes the Huber robust estimator.

Our algorithm is an instance of [ICP](#) since it proceeds by alternating between the data association (Sec. 7.2.3) and optimization steps (Sec. 7.2.2). In the data association step, the corresponding endpoints are found using nearest neighbour strategy based on current velocity estimates. In the optimization step, the velocities are refined from the new correspondences found by the data association.

7.2.1 Velocity Based De-skewing

Whereas our approach can be applied to more complex kinematics, for the sake of simplicity we detail the common case of a unicycle mobile base. Our goal is to estimate the location $\mathbf{T}(\mathbf{x}, t_i) = [\mathbf{R}(t_i) | \mathbf{t}_i]$ of the mobile base at time t_i where $\mathbf{R}(t_i) \in \mathbb{SE}(2)$, and $\mathbf{t}_i = (x_i, y_i)^T$, assuming it starts from the origin and progresses with constant translation and angular velocities over sampling time $\mathbf{x} = (v \ \omega)^T$. After a time t_i , the robot would have traveled for a distance $l_i = v \cdot t_i$ and its angle is $\theta_i = \omega \cdot t_i$.

The base will move along an arc of radius R , such that:

$$R = \frac{v_i}{\omega_i} = \frac{l_i}{\theta_i}. \quad (7.2)$$

From this consideration, we can easily compute $\mathbf{T}$ as

$$\mathbf{T}(\mathbf{x}, t_i) = \begin{pmatrix} R \sin \theta_i \\ R(1 - \cos \theta_i) \end{pmatrix} = \underbrace{v \cdot t_i}_{l_i} \begin{pmatrix} \frac{\sin \theta_i}{\theta_i} \\ \frac{1 - \cos \theta_i}{\theta_i} \end{pmatrix} \quad (7.3)$$

Assuming the sensor ([LiDAR](#)) is located at the center of the mobile base, if at time t_i the beam has a relative angle α_i and reports a range measurement r_i , we can straightforwardly find the 2D laser endpoint $\mathbf{p}_i$ as follows:

$$\mathbf{p}_i(\mathbf{x}, t_i) = \mathbf{R}(\theta_i + \alpha_i) \begin{pmatrix} r_i \\ 0 \end{pmatrix} + \begin{pmatrix} x_i \\ y_i \end{pmatrix}. \quad (7.4)$$

Here $\mathbf{R}(\theta_i + \alpha_i) \in SO(2)$ is 2D rotation matrix of $\theta_i + \alpha_i$. Applying this process to all the measurements and mapping all reconstructed points back to the pose where the acquisition of the current scan was started, results in the desired de-skew operation.

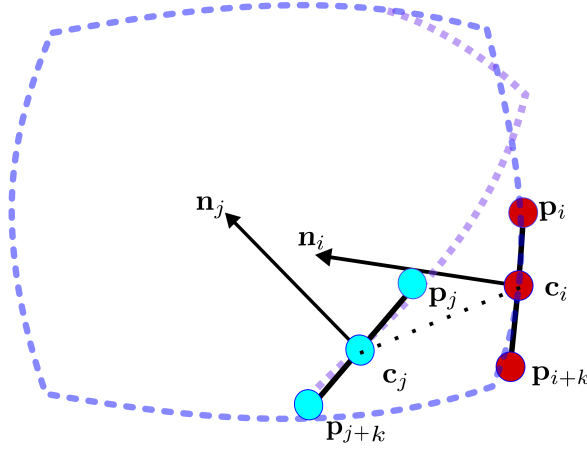


Figure 7.2. Illustration of the planar patches calculation and of the correspondence search between two matching patches $\mathbf{m}_i$ and $\mathbf{m}_j$.

7.2.2 Error Metric for the Optimization

To evaluate a plane-to-plane distance the algorithm needs to compute the normal vectors, which are based on the endpoints. Since the position of the endpoints is a function of the estimated velocities $\mathbf{x}$, also the normal vectors are. Hence the algorithm has to recompute both endpoints and normals at each iteration. Furthermore, when subsequent endpoints are too close, the noise affecting the range might result in an unstable normal vector, which hinders the error metric. To lessen this effect, before each iteration, we regularize the scan to retain only temporally subsequent measurements that are sufficiently far from each other to ensure a stable normal. In our experiments, we set this threshold to 0.15 m. Similarly, if there is a large distance gap between two subsequent endpoints (>0.4 m), it is likely that the surface is not continuous at that point, hence we drop those measurements too.

At the end of this regularization step, we end up with a set of reasonably stable measurements we can use for the remainder of the computation. For each temporally subsequent pair of endpoints, we compute a planar patch $\mathbf{m}_i = \langle \mathbf{c}_i, \mathbf{n}_i, t_i \rangle$, characterized by a center $\mathbf{c}_i$, a normal vector $\mathbf{n}_i$, and a timestamp t_i , such that:

$$\mathbf{c}_i = \frac{1}{2} (\mathbf{p}_{i+k} + \mathbf{p}_i), \quad (7.5)$$

$$\mathbf{n}_i = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \frac{\mathbf{p}_{i+k} - \mathbf{p}_i}{\|\mathbf{p}_{i+k} - \mathbf{p}_i\|}, \quad (7.6)$$

$$t_i = \frac{1}{2} (t_{i+k} + t_i). \quad (7.7)$$

Here, the index $k > 0$ accounts for endpoints suppressed during regularization. If two planar patches $\mathbf{m}_i$ and $\mathbf{m}_j$ corresponds to the same portion of the environment, we can

calculate an error vector $\mathbf{e}(\mathbf{x}, t_i, t_j)$ accounting for both their differences in position and orientation. The error vector is a function of the velocities, since the patches $\mathbf{m}_i$ and $\mathbf{m}_j$ are computed based on the endpoints. The latter is related to the velocities by Eq. (7.4).

Let $\mathbf{e}(\mathbf{x}, t_i, t_j) \in \mathbb{R}^3$ be the error vector, whose components are defined as follows:

$$\mathbf{e}(\mathbf{x}, t_i, t_j) = \begin{pmatrix} \frac{1}{2}(\mathbf{c}_i - \mathbf{c}_j)^T(\mathbf{n}_i + \mathbf{n}_j) \\ \mathbf{n}_j - \mathbf{n}_i \end{pmatrix}. \quad (7.8)$$

Here the first dimension accounts for the distance between two corresponding planar patches $\mathbf{m}_i$ and $\mathbf{m}_j$ projected along the average of their normals. The other two account for the difference between the normal vectors. Eq. (7.8) is differentiable in the velocities $\mathbf{x}$, hence we can minimize Eq. (7.1) by [IRLS](#).

7.2.3 Data Association

To determine the correspondence we proceed at each iteration by de-skewing the sequence of measurements, to get a set of updated endpoints $\mathbf{p}_i$. Subsequently, we apply the regularization to discard those measurements whose endpoints fall either too close or too far from their temporal neighbors. This gives us a set of sequential stable measurements we can use to extract the planar patches.

Once this is done, for each $\mathbf{m}_i$ we seek for those other patches $\mathbf{m}_j$ that fulfill all the following criteria:

- their centers are close enough $|\mathbf{c}_i - \mathbf{c}_j| < \tau_c$,
- their normals are sufficiently parallel $\mathbf{n}_i \cdot \mathbf{n}_j > \tau_n$,
- their timestamp are sufficiently distant $|t_i - t_j| > \tau_t$.

Within this set we select as correspondence for $\mathbf{m}_i$, which is the $\mathbf{m}_j$ having the smallest projective distance $(\mathbf{c}_i - \mathbf{c}_j)^T(\mathbf{n}_i + \mathbf{n}_j)$. Fig. 7.3 shows subsequent iterations of our proposed approach.

7.3 Experimental Validation

In this section, we present quantitative evaluations to demonstrate the accuracy of the velocities estimated by our approach, and its effect on the processed data, using only [LiDAR](#) measurements. To this extent, we carried out statistical experiments under changing velocities.

The goal of the first experiment is to compare the velocities of the robot calculated by our approach with the ground truth applied.

Virtual synthetic scans were created with a skewing effect based on the addressed velocities. We proceed by de-skewing twice, the first time we compute the ground truth measurement by setting the applied velocities, and the second time we compute the de-skewed scan by using our velocity estimate. The consistency is measured as the distance between corresponding endpoints in the two scans. In Tab. 7.1 each cell contains 3 rows in which: the first row is the estimated translational velocity $\bar{v} \pm std(m/s)$, while the second row is the estimated angular velocity $\bar{\omega} \pm std(rad/s)$, and finally the third row is the point

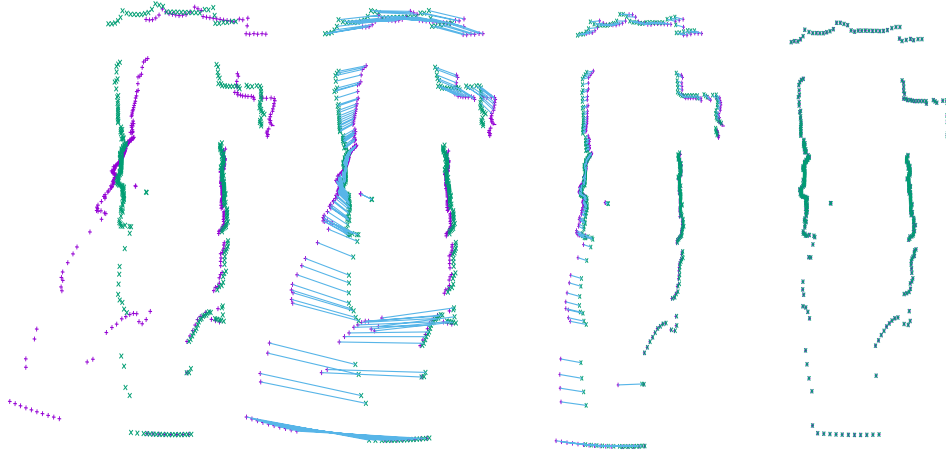


Figure 7.3. Iterative evolution of the de-skewing process. Purple points represent skewed points, green points are the ground truth points, and the blue lines represents the data association. (a): initial configuration, instance of full scan acquisition, with $RMSE = 0.4$ m. (b): after 4 iterations of processing (de-skewing) the raw scan with the proposed approach, decreasing error to $RMSE = 0.2$ m. (c): after 20 iterations, ending up with a de-skewed scan closer to the ground truth, with $RMSE = 0.074$ m.

to point $RMSE$ (m) for the de-skewed/skewed respectively with respect to their ground truth counterpart, for each combination of velocities.

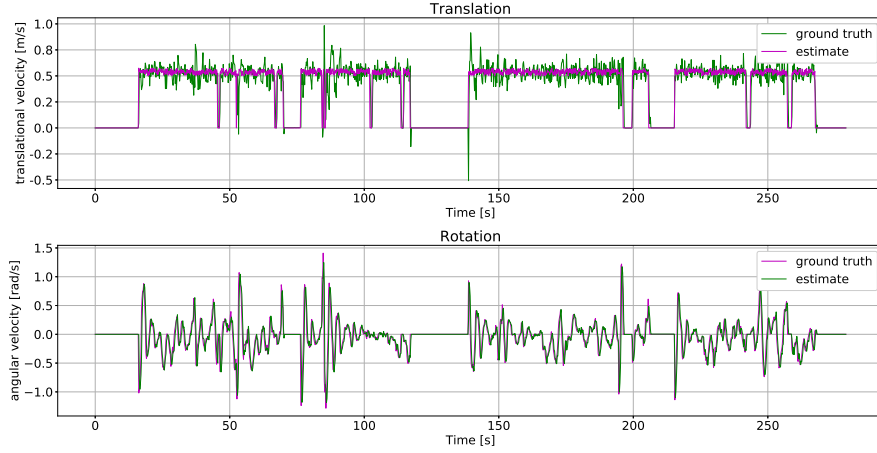
$\omega(rad/s) \backslash v(m/s)$	-2.000	-1.000	-0.500	0.500	1.000	2.000
-2.000	-1.936 $\pm$ 0.090	-0.950 $\pm$ 0.068	-0.471 $\pm$ 0.050	0.470 $\pm$ 0.047	0.962 $\pm$ 0.052	1.910 $\pm$ 0.092
	-1.952 $\pm$ 0.081	-1.933 $\pm$ 0.115	-1.958 $\pm$ 0.066	-1.962 $\pm$ 0.052	-1.952 $\pm$ 0.070	-1.932 $\pm$ 0.103
	0.090 /0.404	0.083 /0.399	0.059 /0.351	0.061 /0.414	0.055 /0.460	0.081 /0.579
-1.000	-1.890 $\pm$ 0.131	-0.979 $\pm$ 0.031	-0.477 $\pm$ 0.035	0.482 $\pm$ 0.034	0.946 $\pm$ 0.069	1.897 $\pm$ 0.073
	-0.949 $\pm$ 0.069	-0.986 $\pm$ 0.023	-0.973 $\pm$ 0.037	-0.979 $\pm$ 0.026	-0.953 $\pm$ 0.059	-0.950 $\pm$ 0.056
	0.067 /0.297	0.058 /0.308	0.055 /0.296	0.049 /0.354	0.054 /0.336	0.062 /0.399
-0.500	-1.929 $\pm$ 0.085	-0.978 $\pm$ 0.063	-0.479 $\pm$ 0.054	0.492 $\pm$ 0.033	0.935 $\pm$ 0.064	1.906 $\pm$ 0.062
	-0.487 $\pm$ 0.020	-0.493 $\pm$ 0.022	-0.477 $\pm$ 0.035	-0.495 $\pm$ 0.009	-0.484 $\pm$ 0.026	-0.482 $\pm$ 0.022
	0.040 /0.308	0.035 /0.345	0.041 /0.188	0.043 /0.200	0.060 /0.218	0.084 /0.338
0.500	-1.955 $\pm$ 0.024	-0.954 $\pm$ 0.044	-0.474 $\pm$ 0.071	0.481 $\pm$ 0.048	0.969 $\pm$ 0.042	1.967 $\pm$ 0.036
	0.483 $\pm$ 0.012	0.495 $\pm$ 0.018	0.476 $\pm$ 0.045	0.485 $\pm$ 0.026	0.488 $\pm$ 0.020	0.496 $\pm$ 0.012
	0.119 /0.158	0.029 /0.140	0.044 /0.112	0.052 /0.132	0.059 /0.161	0.159 /0.271
1.000	-1.844 $\pm$ 0.130	-0.976 $\pm$ 0.064	-0.472 $\pm$ 0.057	0.495 $\pm$ 0.028	0.969 $\pm$ 0.059	1.98 $\pm$ 0.038
	0.933 $\pm$ 0.077	0.976 $\pm$ 0.053	0.940 $\pm$ 0.053	0.992 $\pm$ 0.015	0.970 $\pm$ 0.048	0.992 $\pm$ 0.017
	0.063 /0.261	0.063 /0.225	0.024 /0.231	0.055 /0.302	0.058 /0.303	0.039 /0.335
2.000	-1.906 $\pm$ 0.116	-0.941 $\pm$ 0.076	-0.465 $\pm$ 0.081	0.480 $\pm$ 0.065	0.940 $\pm$ 0.083	1.947 $\pm$ 0.069
	1.904 $\pm$ 0.142	1.919 $\pm$ 0.120	1.912 $\pm$ 0.104	1.905 $\pm$ 0.137	1.922 $\pm$ 0.116	1.963 $\pm$ 0.069
	0.074 /0.416	0.071 /0.368	0.081 /0.358	0.075 /0.435	0.076 /0.494	0.091 /0.424

Table 7.1. Evaluation of the proposed approach using different combinations of translation and angular velocities.

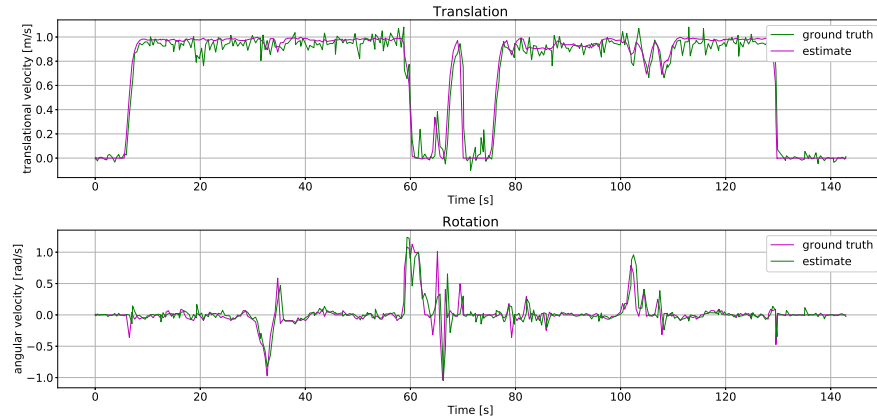
We repeated this experiment on the real robot and the velocity plots in Fig. 7.4 confirm that our system can effectively recover the robot's velocities, correctly de-skewing the scan, in real life indoor scenarios.

7.3.1 De-skewing Effects in SLAM application

In our final experiment, we used our de-skewing mechanism to pre-process the input of our 2D LiDAR SLAM system [36]. We run SLAM on three different inputs: the raw (skewed) scans, the ones de-skewed by using the proposed approach, and finally on the data de-skewed by using the ground truth. We simulated different exploration runs of the same environment while changing the velocity bounds of the robot. Fig. 7.5 illustrates the three different maps obtained.



(a) Velocity estimated using synthetic data.



(b) Velocity estimated using real data.

Figure 7.4. Evolution of estimated and real velocities.

Furthermore we measure the ATE between the ground truth trajectory computed with by the simulator and the ones estimated by SLAM using the two types of scans. ATE measures the distance between corresponding points of the trajectories, after computing a transformation that makes them as close as possible. For trajectory registration we used Horn method [79], while the corresponding poses are associated based on the timestamps. The results are summarized in Tab. 7.2. Consistently, the trajectories obtained by using de-skewed data are substantially close to the ground truth than their skewed counterpart as shown in Fig. 7.5d. This is confirmed by the maps generated based on the trajectories and

Velocities		ATE(m)	
$v_{\max}$ (m/s)	$\omega_{\max}$ (rad/s)	w/o deskewing	deskewed
0.5	0.5	0.329	0.113
0.5	1.0	0.246	0.232
1.0	1.0	1.018	0.977
1.0	1.5	1.586	1.520
2.0	2.0	11.18	1.805

Table 7.2. SLAM comparison.

illustrated in Fig. 7.5. These results are consistent with the intuition that using skewed scans when doing SLAM is not recommended, since these measurements might induce systematic unrecoverable errors in the registration process.

7.4 Conclusion

This chapter introduced a minimalistic yet effective de-skewing approach for 2D LiDAR sensors, tailored for scenarios where no additional proprioceptive sensors (such as IMUs or wheel encoders) are available. By exploiting a plane-to-plane registration framework, we demonstrated the feasibility of estimating the velocity of the sensor platform directly from range-only measurements, leading to significantly improved scan consistency even with slow and noisy sensors.

While the method was developed and validated in the context of 2D LiDAR, the broader implications of this work reach into more general sensor modalities. In particular, RADAR sensors pose an even greater challenge in terms of motion distortion, as they often acquire data at lower frequencies, around 4 Hz, resulting in longer scan durations. This extended temporal window increases the range of motion covered during scan acquisition, making the resulting distortions more severe and the correction more critical.

The insights provided in this chapter serve as a technical and conceptual stepping stone toward robust motion compensation strategies that can be extended to RADAR data. While the specifics of RADAR sensing, such as its sparse and noisy returns, Doppler measurements, and angular uncertainty, differ substantially from LiDAR, the core problem of temporally inconsistent data remains the same.

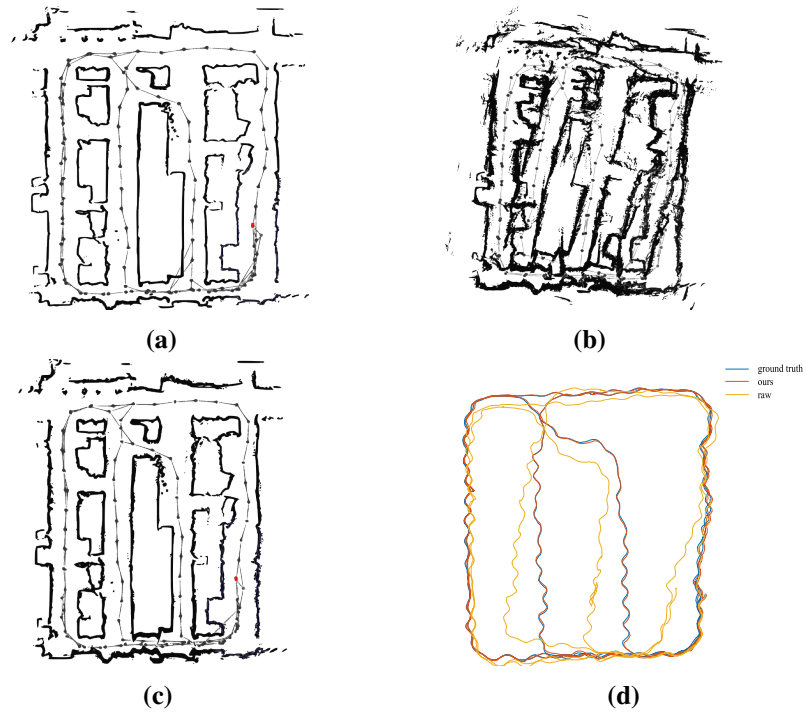


Figure 7.5. SLAM Results. (a) Ground truth map (reference). (b) Map constructed using raw scans acquired directly from the LiDAR. (c) Map constructed using processed LiDAR measurements de-skewed by our proposed approach. (d) Estimated trajectories from SLAM, produced by scan matching between consecutive measurements.

Chapter 8

DRO: Doppler-Aware Direct RADAR Odometry

BUILDING on the direct-error paradigm developed for [LiDAR](#) and RGB-D sensors in the preceding chapters (Chapter 3, Chapter 6), we now turn to millimetre-wave [RADAR](#).

From air traffic control to weather forecasting via marine navigation, [RADAR](#) sensing is a cornerstone of modern societies. During the early days of mobile robotics, [RADAR](#) were used to detect specific landmarks and perform [SLAM](#) [45]. While the advent of [LiDAR](#) technologies and the ubiquity of cameras drove the state estimation research community away from [RADAR](#) sensing [29], we are currently experiencing a ‘new wave’ of interest toward this modality [74]. This interest is partly driven by the robustness of [RADARs](#) with respect to adversarial weather conditions. In this chapter, we present a novel algorithm for [RADAR](#)-based odometry and benchmark its performance across different datasets in both automotive and off-road robotic scenarios as shown in Fig. 8.1.

The working principle of [RADAR](#) sensing consists of emitting radio waves and analyzing the signal returned through various reflections of the waves from objects present in the environment to collect geometric information about the surroundings. This work leverages data from millimeter wave (mmWave) [RADAR](#) that use electromagnetic waves with a length scale that is in the order of magnitude of one millimetre. Waves in such frequency bands can penetrate through plastic, textiles, glass, etc. Accordingly, in the context of mobile robotics, [RADAR](#) allows the system to sense its environment in a wide range of conditions, even under intensely adverse weather conditions such as fog, rain, and snow. Like any wave, the [RADAR](#)-emitted signals are subject to the Doppler effect when the relative velocity between the system and part of its environment is not zero. Thus, when using specific frequency patterns and processing, [RADAR](#) data can allow for the extraction of Doppler-based radial velocity information. These benefits represent a competitive advantage of [RADARs](#) compared to other modalities, such as vision or commonly used [LiDARs](#). The proposed odometry method accounts for the Doppler-based distortion of the [RADAR](#) data and can integrate a Doppler-specific velocity constraint in its optimization, allowing it to robustly perform odometry even in geometrically feature-deprived environments, such as featureless tunnels (see Fig. 8.1 (left)).

Two types of [RADAR](#) sensors are typically used in robotics applications: ‘fixed’ phased-array [RADARs](#) and spinning [RADARs](#). In this work, we focus on spinning [RADARs](#), as they

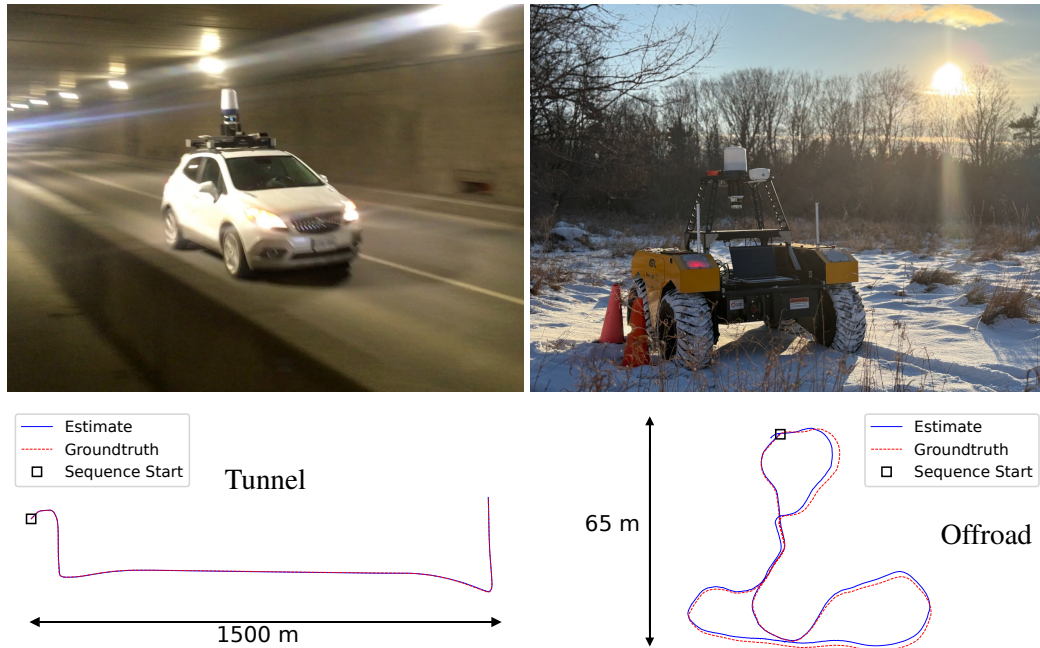


Figure 8.1. This chapter presents a Direct **RADAR** Odometry (DRO) method that provides state-of-the-art performance in a wide range of environments, including unstructured ones (right) and geometrically degenerated ones such as featureless tunnels (left).

provide a 360° **FoV** that contains dense intensity information from up to several hundred meters away. This makes spinning **RADAR** enticing for navigational tasks such as odometry, localization, and object tracking, where a view of the entire surroundings is required for robustness and safety. The most common spinning-**RADAR**-odometry approaches are indirect, they use the dense intensity information to extract a set of points or features [33, 4, 24, 13, 26, 104, 128]. These approaches work well, with point-based methods currently being the **SOTA** [4, 101], but they require computationally costly data association, critical parameter tuning, and they ultimately discard a large amount of the gathered data in favour of sparse representations. Methods that have used the entire scan directly for odometry have done scan-to-scan correlation matching on the intensity image [124], sometimes requiring pre-processing using a deep learning network [14, 160]. However, these methods do not explicitly account for the motion and Doppler distortion of the **RADAR** data, and discretize the relative transform search space to lower the computational cost. Additionally, the need for a learned processing component limits the transferability of these approaches to different environments.

In this work, we present the first Direct Radar Odometry (DRO) framework that accounts for the continuous motion and Doppler distortion of the incoming data in a principled way. It can be aided by a yaw-axis gyroscope and does not rely on deep-learning techniques or hand-tuned discretization parameters. As demonstrated on more than 250 km of data, it is applicable to a wide range of environments. With the appropriate **RADAR** emission pattern, it can also perform in geometrically challenging scenarios that generally lead to degenerated state estimates. We introduce the first direct radar odometry framework that performs gradient-based scan registration while explicitly modelling continuous motion and Doppler-

induced distortions, without discretizing the search space. The method leverages Doppler measurements for velocity estimation without relying on beam-to-beam correspondences, achieves state-of-the-art performance when fused with gyroscopic data, and runs in real time on GPU hardware, with an open-source implementation available at <https://github.com/utiasASRL/dro>.

8.1 Related work

A radar odometry system [74, 2, 156, 32], like other odometry systems, aims to estimate the ego-motion of the sensor, in this case, a **RADAR**, thus determining the trajectory of the platform on which the sensor is mounted. The motion estimation process typically involves formulating an optimization problem, where the objective is to find the sensor's state and an environment model that maximizes the probability of observing the measured data. This principle is known as maximum-likelihood estimation.

In the last decade, **RADAR** odometry has been dominated by indirect approaches [2], which rely on selecting salient points or features from radar measurements. Early works, such as [30], started by leveraging computer vision approaches, such as SIFT [109] features, from **RADAR** Cartesian images, while more recent state-of-the-art methods, such as CFEAR [4], select points based on each individual azimuths of a **RADAR** signal. There exist two main categories for feature extraction methods [125]: signal-based and spatial-based extractors. Signal-based methods identify points by applying thresholds to individual azimuths in the **RADAR** signal. A well-known example is the Constant False Alarm Rate (CFAR) method [52], which processes each azimuth of a **RADAR** scan using a sliding window. The power readings within the window are used to calculate a detection threshold. Many variants of CFAR exist [111, 133, 60, 174, 172, 145, 18]. Another signal-based extractor, K-strongest [3], selects the K strongest power returns above a static threshold along each azimuth. In contrast, spatial-based extractors rely on extracting features and descriptors from **RADAR** scans transformed into Cartesian images, some of those leverage computer vision algorithms, such as ORB [136] or SIFT [109].

Indirect odometry systems, extracting points from the **RADAR** data, rely on matching these features. The data association step is one of the most critical building blocks of these systems. Burnett et al. [23] introduced motion-compensated **RANSAC** to ensure robust association. Subsequently, they developed Hero [26], a method that leverages learned key-point locations, uncertainties, and descriptors to address outliers and facilitate matching within the ESGVI [12] batch state estimation framework. In CFEAR [4], the authors extract the surface points and demonstrate that the point-to-line and point-to-distribution perform better than point-to-point, as these metrics mitigate the problem of erroneous associations between individual points. After an initial scan filtering step, Kung et al. [95] extract sets of local Gaussian distributions and perform point-to-distribution registration. Finally, some other systems exploit data-driven approaches to downscale noisy point contributions in the optimization problem [106] or to select others suitable for the data association [6].

Alternatively, direct methods do not need explicit data association. While this kind of method has been well studied for **LiDAR** [38, 67, 44] and Visual Odometry (VO) systems [48, 55, 70, 40, 47], the literature on direct **RADAR**-based odometry is still somewhat limited and methods are slower than feature-based methods. Barnes et al. [14] proposed a brute-force approach that samples and selects rotations to maximize the dense correlation

between **RADAR** scans. Similarly, Checchin et al. [34] and Park et al. [124] addressed the need for sampling by decoupling rotation and translation. They utilized the Fourier-Mellin Transform, maximizing phase correlation between log-polar and Cartesian images to estimate rotation and translation, respectively. A coarse-to-fine phase correlation refinement on Cartesian images was then applied to improve the translation estimate. While Barnes et al. [14] achieved remarkable performance in odometry estimation, their method suffered from high computational complexity. To address this, Weston et al. [160] replaced the brute-force rotation search with an FFT-based approach, significantly reducing computation time at the cost of slightly lower accuracy. Overall, existing direct methods either decouple the estimation of rotation and translation or sub-sample the problem within a discrete search space. These strategies can result in susceptibility to local minima and poor scalability across different resolutions. In contrast, our approach optimizes the locally continuous $\mathbb{SE}(2)$ **RADAR** trajectory without discretizing the search space.

For both direct and indirect methods, motion distortion can significantly degrade system performance. In the literature, this issue has been addressed using continuous-time trajectory estimation [9], [8, 10, 57] for **LiDAR** [19, 137] and rolling-shutter cameras [77]. This work applies similar concepts to **RADAR** data while also correcting Doppler distortion in a continuous-time optimization, estimating the velocity of the sensor. The Doppler information channel provides valuable additional data that can be leveraged for odometry and ego-motion estimation. Several works have explored its integration with different models and estimation techniques to enhance motion estimation accuracy. Kellner et al. [88] demonstrated that automotive **RADAR**, combined with an Ackermann vehicle model, can estimate ego-motion by fitting least-squares to stationary point Doppler velocities identified via **RANSAC**. In [63], a gyroscope is used to alleviate the need for an Ackermann platform. Galeote-Luque et al. [58] extended this by using Doppler measurements for 3D odometry with **RANSAC** and a kinematic model. Kubelka et al. [94] showed that integrating high-quality Doppler data with an **IMU** outperforms **ICP** in feature-deprived environments, and Lisus et al. [105] implemented a continuous-time approach with a white-noise-on-acceleration prior. In [132], the authors extend the deep-learning-based preprocessing from [14] and trained an additional network for per-scan velocity regression. In contrast, our model-based approach does not require any training data and does not suffer from any generalization issues.

8.2 Direct RADAR Odometry

The proposed method addresses the issue of **RADAR** odometry optionally aided by a gyroscope. The main concept is the direct use of **FMCW** intensity data without any point cloud or feature extraction, or data association.

8.2.1 Problem statement and overview

Let us consider a 2D **mmWave FMCW** spinning **RADAR** and a rigidly mounted gyroscope. Without loss of generality, the extrinsic calibration (rotation matrix) between the two sensors is assumed to be known and will not appear in the following derivations. The gyroscope provides measurements $\tilde{\omega}_i$, at times t_i , of the system's angular velocity around the spinning axis of the **RADAR**. The **RADAR** data is assumed to be collected in scans covering the 360° field-of-view. Each scan contains information collected from N beams, each associated with

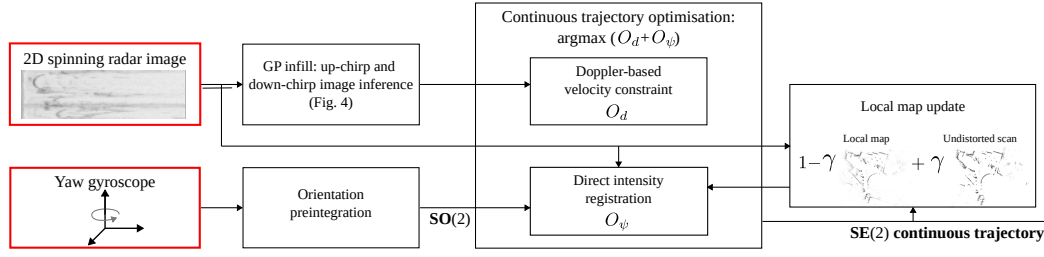


Figure 8.2. Overview diagram of the proposed direct RADAR odometry method. To leverage the Doppler-based velocity constraint, the RADAR must use a triangular frequency modulation pattern. Note that O_ψ accounts for motion and Doppler-induced distortion.

an azimuth α_n and timestamp t_n , and each beam consists of the electromagnetic reflection intensity ψ_{nm} at M different ranges $\hat{r}_m$. Accordingly, a RADAR scan can be interpreted as a polar intensity image or a set of tuples $\{\alpha_n, \hat{r}_m, \psi_{nm}\}$.

The rotation, position, and velocity of the RADAR reference frame with respect to an Earth-fixed frame are described with continuous functions $t \in \mathbb{R} \mapsto \mathbf{R}_W^t \in \mathbb{SO}(2)$, $\mathbf{p}_W^t \in \mathbb{R}^2$, and $\mathbf{v}_W^t \in \mathbb{R}^2$, respectively. The details of the various motion models used to obtain $\mathbf{R}_W^t$, $\mathbf{p}_W^t$, and $\mathbf{v}_W^t$ from a finite set of state variables $\mathcal{S}$ will be discussed in Sec. 8.2.4. The proposed method aims to estimate the state variables $\mathcal{S}$, thus the sensor’s trajectory, during the collection time of the last RADAR scan using the RADAR intensity data and optionally the angular velocity from the gyroscope. The estimation is done with the numerical optimization of a combination of objective functions as

$$\mathcal{S}^* = \underset{\mathcal{S}}{\operatorname{argmax}} \quad O_\psi + O_d, \quad (8.1)$$

with O_ψ an objective function that relates to the direct registration of the RADAR intensity data (cf., Sec. 8.2.2, and O_d to the Doppler effect (cf., Sec. 8.2.3). Fig. 8.2 provides an overview of the proposed method. As detailed later in this section, both O_ψ and O_d can be interpreted as cross-correlation scores between different parts of the present and past RADAR data.

8.2.2 Direct intensity objective function

This objective function is inspired by direct methods in traditional monocular state estimation. The goal is to estimate the continuous pose of the sensor to register/align the last RADAR scan with past RADAR data. In this work, we opted for a local map that is updated ‘on-the-fly’ as a way to combine past RADAR scans into a single image representation of the sensor’s environment with intensity in Cartesian coordinates (as opposed to raw RADAR data in polar coordinates). The motivation for using a local map instead of solely the last RADAR scan is the higher robustness to outliers such as moving vehicles or multipath echoes.

Direct registration

Let us assume the availability of an image-like local map $\mathcal{M}$ of the sensor’s environment expressed in the RADAR’s reference frame at time t_1 . The rows and columns of $\mathcal{M}$ correspond to discrete Cartesian coordinates. The function $\Gamma_{\mathcal{B}}(\mathcal{M}, x, y)$ allows querying

the local map intensity for any x and $y \in \mathbb{R}$ with the radar projection (Sec. 2.1.3) and the bilinear interpolation.

To compute the direct registration objective between the incoming **RADAR** scan and the local map $\mathcal{M}$, we need to correct for the Doppler effect and the motion of the **RADAR** as

$$\begin{bmatrix} \tilde{x}_{nm} \\ \tilde{y}_{nm} \end{bmatrix} = \mathbf{R}_{t_1}^{t_n} \left(\left(\hat{r}_{nm} \pm \frac{\Delta r_n}{2} \right) \begin{bmatrix} \cos(\alpha_n) \\ \sin(\alpha_n) \end{bmatrix} \right) + \mathbf{p}_{t_1}^{t_n}, \quad (8.2)$$

with Δr_n the Doppler-induced shift Eq. (2.25) as a function of the body-centric velocity and the beam azimuth:

$$\Delta r_n = \beta \begin{bmatrix} \cos(\alpha_n) \\ \sin(\alpha_n) \end{bmatrix}^\top \mathbf{R}_W^{t_n \top} \mathbf{v}_W^{t_n}. \quad (8.3)$$

The sign $\pm$ is defined by the chirp direction for the azimuth α_n . Please note that this objective function accounts for the Doppler effect but does not require an **FMCW RADAR** with a triangular pattern. Once the **RADAR** scan is corrected using the current estimate of the trajectory, the objective function is simply the cross-correlation score,

$$O_\psi = \sum_{n=1}^N \sum_{m=1}^M \psi_{nm} \Gamma_B(\mathcal{M}, \tilde{x}_{nm}, \tilde{y}_{nm}), \quad (8.4)$$

between the corrected scan and the local map.

In this chapter, we adopt the term objective function rather than error function to reflect a fundamental shift in the estimation paradigm. Unlike the approach taken in previous chapters, where the goal was to minimize a residual error between predicted and observed measurements, here we frame the state estimation problem as a maximization task. Specifically, we aim to find the trajectory that maximizes the alignment between the corrected **RADAR** scan and the local map, as measured by the cross-correlation score. This formulation emphasizes data agreement and provides a more direct means of exploiting the spatial consistency in the **RADAR** signal, especially when incorporating Doppler information.

On-the-fly local map

The proposed local map consists of an image-like representation of the past **RADAR** data using a per-pixel low-pass filter to update $\mathcal{M}$ with the last **RADAR** scan and the corresponding state estimate. Concretely, given the convergence of the optimization problem Eq. (8.1), the **RADAR** scan is corrected to the first timestamp of the following scan in Cartesian coordinates (using Eq. (8.2) replacing $\mathbf{R}_{t_1}^{t_n}$ and $\mathbf{p}_{t_1}^{t_n}$ with $\mathbf{R}_{t_{N+1}}^{t_n}$ and $\mathbf{p}_{t_{N+1}}^{t_n}$, respectively). The corrected **RADAR** data is converted into an image-structure $\tilde{\mathcal{I}}_{\mathcal{M}}$ with the help of bilinear interpolation. Then the local map update is

$$\mathcal{M} \leftarrow (1 - \gamma)\mathcal{M} + \gamma\tilde{\mathcal{I}}_{\mathcal{M}}, \quad (8.5)$$

with $\gamma \in [0, 1]$ a user-defined parameter ($\gamma = 0.1$ in our implementation). Note that after receiving the very first **RADAR** frame, the local map is directly initialized with $\tilde{\mathcal{I}}_{\mathcal{M}}$.

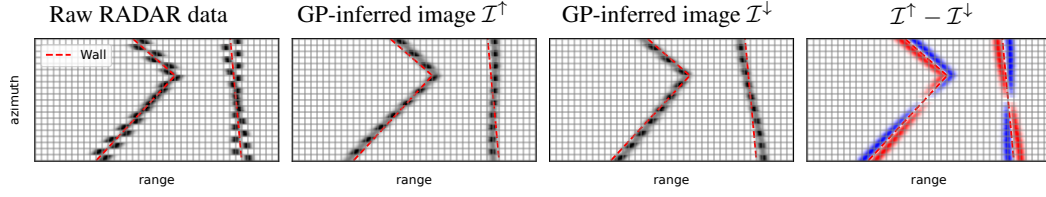


Figure 8.3. Illustration of the proposed infilling. The raw Doppler distorted RADAR data (left) is split between up and down-chirp azimuths. GP regression is used to interpolate every second azimuth (middle two, the interpolated portion is in grey). Both $\mathcal{I}^\uparrow$ and $\mathcal{I}^\downarrow$ virtually observe the same geometry without assuming that two consecutive azimuths of the raw data observe the same part of the environment (right, with blue positive and red negative).

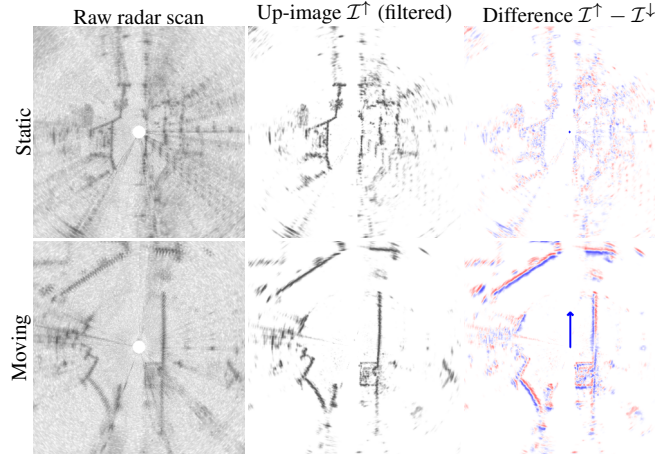


Figure 8.4. Illustration of the GP-based infill highlighting the Doppler effect on RADAR data collected with a triangular frequency modulation pattern (the blue arrow is the sensor velocity).

8.2.3 Doppler-based objective function

As presented in 2.1.3, an FMCW RADAR using a triangular modulation pattern (alternating up-chirp and down-chirp for each collected beam) allows for observability of the radial velocity along that beam if the consecutive beams observe the same part of the environment. Unfortunately, this is not quite true in the context of mobile RADAR sensing, since the vehicle and the RADAR dish move between consecutively recorded returns. We propose to split the incoming RADAR scan into two image-like structures based on chirp direction and use GP regression Eq. (2.27) to ‘infill’ the missing rows in both images. This is illustrated in Fig. 8.3. We denote the up-chirp and down-chirp infilled images as $\mathcal{I}^\uparrow = \{\alpha_n, \hat{r}_m, \psi_{nm}^\uparrow\}$ and $\mathcal{I}^\downarrow = \{\alpha_n, \hat{r}_m, \psi_{nm}^\downarrow\}$, respectively. Thanks to the GP regression step, the part of the environment observed in each row of $\mathcal{I}^\uparrow$, be it observed or interpolated, is the same as in the corresponding row of $\mathcal{I}^\downarrow$.

We show in Fig. 8.4 that when the sensors are static, both $\mathcal{I}^\uparrow$ and $\mathcal{I}^\downarrow$ overlap well. However, the Doppler shift is clearly visible in the difference between $\mathcal{I}^\uparrow$ and $\mathcal{I}^\downarrow$ when the sensors move. The objective function O_d aims to constrain the estimated velocity $\mathbf{v}(t)$ using $\mathcal{I}^\uparrow$ and $\mathcal{I}^\downarrow$. First, we must compute the ‘range shift’, with Eq. (8.3), between each row of the two images as a function of the sensor velocity. Then, we define the function $\Gamma_{\mathcal{L}}(\mathcal{I}^\uparrow, \alpha_n, r)$ that queries the intensity value of $\mathcal{I}^\uparrow$ for any value of $r \in \mathbb{R}$ using linear interpolation for

each of the image's row. The objective function is a cross-correlation score between $\mathcal{I}^\downarrow$ and the Doppler-rectified version of $\mathcal{I}^\uparrow$:

$$O_d = \sum_{n=1}^N \sum_{m=1}^M \psi_{nm}^\downarrow \Gamma_{\mathcal{L}}(\mathcal{I}^\uparrow, \alpha_n, \hat{r}_m + \Delta r_n). \quad (8.6)$$

The objective function O_d is maximised when the body-centric velocity estimate $\mathbf{R}(t_n)^\top \mathbf{v}(t_n)$ corresponds to the actual velocity of the sensor.

8.2.4 Motion models

In this section, we present different ways to obtain the continuous trajectory functions $\mathbf{R}_W^t$, $\mathbf{p}_W^t$, and $\mathbf{v}_W^t$ from the state variables $\mathcal{S}$. As $\mathbf{R}_W^t$ is in $\mathbb{SO}(2)$, the rotation can easily be represented with a simple rotation angle $\theta(t)$ as

$$\mathbf{R}_W^t = \mathbf{R}_W^{t_1} \begin{bmatrix} \cos(\theta(t)) & \sin(\theta(t)) \\ -\sin(\theta(t)) & \cos(\theta(t)) \end{bmatrix}, \quad (8.7)$$

with $\mathbf{R}_W^{t_1}$ being the starting rotation obtained from the state estimate of the previous [RADAR](#) scan. We introduce two different models to obtain $\theta(t)$ in Subsections [8.2.4](#), and [8.2.4](#). Regarding the translation and velocity, we use a simple constant body-centric velocity model in [Sec. 8.2.4](#). For convenience, we define $\mathcal{S} = \{\mathcal{S}_R, \mathbf{v}_b\}$ with $\mathcal{S}_R$ the rotational state variables, and $\mathbf{v}_b$ the body-centric velocity.

Constant angular velocity

In the absence of a gyroscope, the proposed method can rely on the assumption of constant angular velocity ω for the duration of any [RADAR](#) scan with $\theta(t) = \omega(t - t_1)$. Therefore, the state variable is $\mathcal{S}_R = \{\omega\}$.

Gyro preintegration

Inspired by [\[98\]](#), when using a gyroscope, the change of orientation of the sensor suite is defined as a function of the angular rate measurements. To account for the frequency discrepancy between the [RADAR](#) azimuths and the gyroscope measurements, it leverages continuous preintegration concepts drawn from [\[100\]](#) and [\[99\]](#). For the sake of simplicity and lightweight computation, we use a simple piece-wise linear model

$$\theta(t) = \theta(t_i) + \frac{(\omega_{i+1} + \omega_i)(t - t_i)}{2(t_{i+1} - t_i)} + \omega_i(t - t_i), \quad (8.8)$$

with $\omega_i = \tilde{\omega}_i - b$, $t \in [t_i, t_{i+1}]$. We decided not to include the gyroscope bias b in the estimated state. In our experiments, biases range from $\approx 5e^{-5}$ to $\approx 4e^{-3}$ corresponding to a maximum of 0.057° drift over the 250 ms of a scan. With the typical 0.9° [RADAR](#) angular resolution, the signal-to-noise ratio is too high to enable robust bias estimation in a ‘frame-to-frame’ approach like ours. This is especially true as the proposed objective functions are not probabilistic, thus impeding the integration of additional residuals/objective functions from other modalities and the probabilistic prior/previous knowledge of the bias. Accordingly, when using a gyroscope, the rotational state is empty $\mathcal{S}_R = \emptyset$. However,

we use a separate online bias estimation strategy based on a simple yet effective heuristic: when the vehicle's velocity is zero, so is the angular rate. Accordingly, the bias estimate is initialized by averaging the first Q raw gyroscope measurements collected when the velocity estimate is under 5 cm/s. Then it is updated with a low-pass filter whenever the velocity estimate is under 5 cm/s.

Constant body-centric velocity

This model assumes a constant body-centric velocity $\mathbf{v}_b \in \mathbb{R}^2$ during any **RADAR** scan. The velocity in the global frame is $\mathbf{v}_W^t = \mathbf{R}_W^t \mathbf{v}_b$. The position is

$$\mathbf{p}_W^t = \mathbf{p}_W^{t_1} + \left(\int_{t_1}^t \mathbf{R}_W^s ds \right) \mathbf{v}_b, \quad (8.9)$$

with $\mathbf{p}_W^{t_1}$ the position resulting from the state estimation during the previous **RADAR** scan. The integral in Eq. (8.9) is computed analytically when using the constant angular velocity assumption. Otherwise, when using a gyroscope, it is numerically preintegrated using a piece-wise linear model similar to Eq. (8.8) upon each of the matrix elements.

8.3 Experiments

8.3.1 Implementation

The proposed method has been implemented in Python using PyTorch for matrix and algebra operations. Accordingly, it can easily be run on either a **GPU** or a **CPU**. Note that we have implemented Eq. (8.1) with analytical Jacobians and our solver without relying on PyTorch's automatic differentiation and integrated solvers. When using a gyroscope and the Doppler-based velocity constraints, our method converges in 13.5 iterations on average.

Gaussian process regression

GP regression is known to suffer cubic computational complexity due to the matrix inversion in Eq. (2.27). With around 2 million points in each **RADAR** scan, naive **GP** regression to generate $\mathcal{I}^\uparrow$ and $\mathcal{I}^\downarrow$ is intractable. Fortunately, efficient computation strategies are enabled by the grid-pattern (image-like) nature of both the **RADAR** data and the inference locations in the proposed **GP**-based infill step. By only considering the data in a $U \times V$ neighbourhood around each inference location, $\mathbf{k}_x * \mathbf{x} (\mathbf{K}_{\mathbf{x}\mathbf{x}} + \sigma^2 \mathbf{I})^{-1}$ can be precomputed and the multiplication with $\mathbf{y}$ is a simple convolution operation. Accordingly, with a **GPU** implementation, the inference of $\mathcal{I}^\uparrow$ and $\mathcal{I}^\downarrow$ is extremely efficient.

Similarly to [105], each row of the **GP**-inferred image is independently filtered to attenuate multipath and specular noise. First, the intensity standard deviation is computed and any range bin with a value inferior to twice the deviation is nullified. Then the intensity values are normalized so that the maximum value equals one. Per-azimuth Gaussian blur is applied to smooth the discontinuities introduced in the previous steps. Finally, each intensity value is cubed to augment contrast.

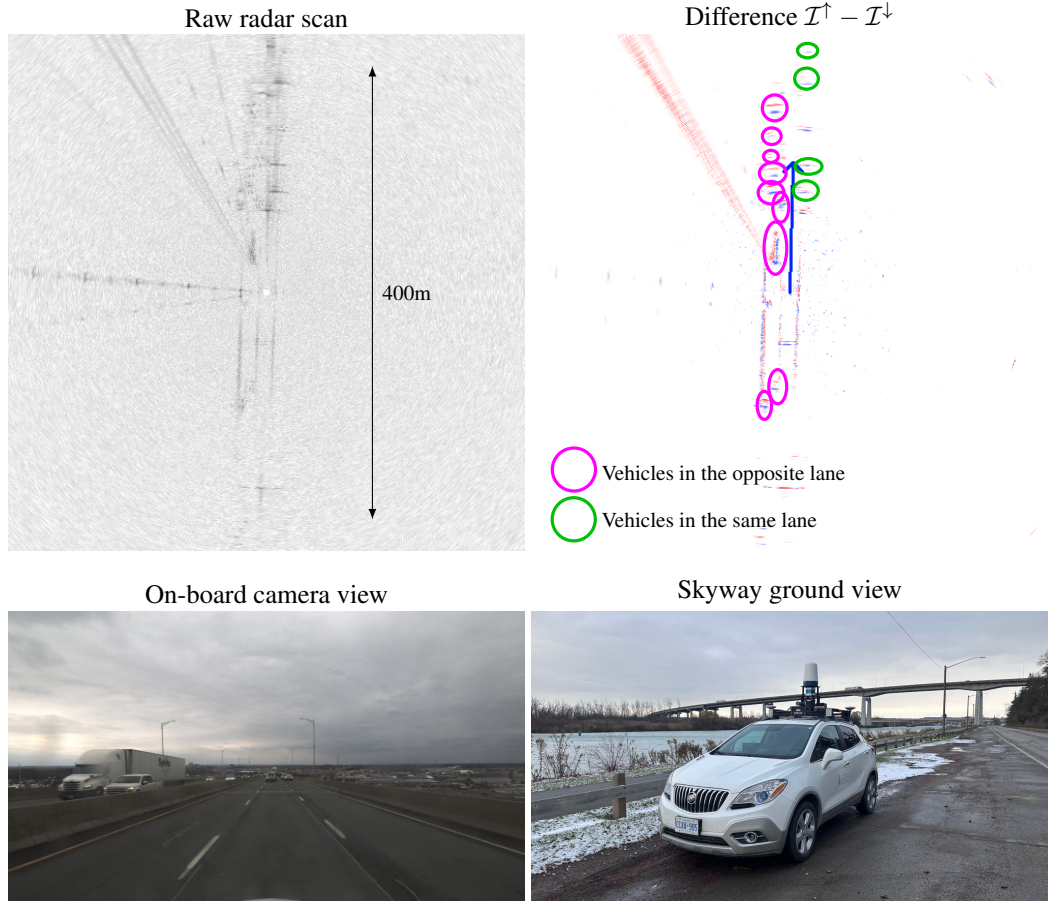


Figure 8.5. Illustration of a challenging situation with a line of vehicles coming the other way while driving over a skyway (very few static elements visible in the **RADAR** data).

Optimization

The optimization problem Eq. (8.1) is solved with a gradient ascent algorithm with an update step

$$\mathcal{S} \leftarrow \mathcal{S} + c \frac{\nabla(O_d + O_\psi)}{\|\nabla(O_d + O_\psi)\|}, \quad (8.10)$$

where ∇ is the gradient operator, and c is a constant equal to 0.1 at the start of the optimization and halved for every non-ascending step. Note that the objective functions are element-wise products of intensity values. Thus, the computations linked to intensity values that are equal to zero do not contribute to the optimization problem and can be omitted to lower the computational burden.

Robust weighting

The proposed objective functions O_d and O_ψ are generally robust to outliers such as moving vehicles and specular noise. However, on very rare occasions (2 frames among 42k in our automotive experiments), a very high proportion of consistent outliers can lead to

erroneous state estimates. A typical example is illustrated in Fig. 8.5 when the sensing platform is moving over a bridge/skyway with a very small amount of static reflectors in the surroundings, and a line of vehicles comes the other way. It can make the velocity estimate suddenly jump unrealistically. Such scenarios are easily detected with a simple threshold on the sensor’s acceleration from one scan estimate to the next. When detected, we re-run the optimization weighting the objective functions O_ψ and O_d as

$$O_\bullet = \sum_{n=1}^N \sum_{m=1}^M \rho_{nm}^\bullet \psi_{nm} \Gamma(\cdot, \cdot, \cdot), \quad (8.11)$$

$$\rho_{nm}^\bullet = (|\psi_{nm} - \Gamma(\cdot, \cdot, \cdot)| - 1)^6,$$

with $\bullet = d$ or ψ . Intuitively, the weighting ρ acts as an outlier-rejection mechanism by down-weighting the objective components that are too dissimilar in terms of intensity.

Velocity bias in Doppler constraint

Similarly to [105], we have empirically observed a velocity bias when performing Doppler-only velocity estimation. This bias mostly concerns the lateral velocity of the sensing platform. We believe this bias is partly explained by the disparity between the actual measurement process and the simple measurement model that assumes the information contained in a row of the **RADAR** data corresponds to a single azimuth/timestamp: in reality, the **RADAR**’s scanning pattern is continuous. Thus, every measurement from a row could correspond to a different azimuth and timestamp. We think that the motion model’s ‘inaccuracy’ leads to a velocity bias that appears as environment-dependent due to the ignored correlation between the data range and azimuth/timestamp. The authors of [105] addressed this issue by fitting a linear model to the velocity error using held-out data with accurate velocity ground-truth. Then, the linear model is used to correct the velocity estimates directly. In this work, we propose to leverage the non-slip kinematics constraint to estimate and compensate for the lateral component of the velocity bias online without a ‘calibration’ phase.

Using the angular rate and the extrinsic calibration between the **RADAR** and the vehicle’s rear axle, the velocity estimated based solely on the Doppler-based constraint O_d is projected into the axle reference frame. The y component is projected back to the **RADAR** frame to update a low-pass filter that tracks the lateral velocity bias. The filter’s output is then used to correct the radial velocities when computing the shifts Eq. (8.3) for Doppler distortion correction. This estimation and correction process is only applied when O_d is part of the optimization problem Eq. (8.1). It is important to note that this mechanism does not strictly enforce the non-slip constraint thanks to the low-pass filter. Accordingly, the vehicle can be subject to momentary side slips without altering the state estimates.

8.3.2 Metrics

To benchmark our algorithm, we use well-established metrics in the odometry literature. For datasets with ground-truth orientation, the KITTI relative translation and rotation errors are adopted. Succinctly, for one pose estimate, the estimated trajectory is aligned with the ground truth. Then, after a certain distance travelled from the aligned poses (segments varying from 100 to 800 m), the estimated and ground truth poses are used to compute

position and orientation errors relative to the segment lengths. This process is repeated every 5 poses of the estimated trajectory, and the results are reported as the average relative errors in % and $^{\circ}/100$ m. In the absence of ground truth orientation, we adapted the RPE introduced in [173] to $bbSE(2)$ trajectories. It corresponds to the position RMSE between aligned segments of the ground truth and the estimated trajectory. The length of segments [50 m, 100 m, 150 m, 200 m]. Similarly to the KITTI metric, we display the RMSE error as a percentage of the segment lengths.

8.3.3 Doppler-enabled automotive odometry

In this section, we provide a detailed analysis of our method’s performance on a dataset collected with our automotive sensing platform. Note that the RADAR used in this set of experiments uses a triangular frequency modulation pattern, thus enabling the use of the Doppler-based velocity constraints O_d in Eq. (8.1).

Dataset description

The goal of this set-up is to replicate experiments from [105]. To do so, we have collected data with an automotive platform equipped with a Navtech RAS6 RADAR, a Silicon Sensing DMU41 IMU, a Velodyne Alpha Prime lidar, and an accurate Applanix RTK-GNSS/INS solution for ground truth with post-processing. All the sensors have been accurately synchronized. The data used for benchmarking has been collected repeatedly four times along four routes (16 sequences in total) displaying an increasing level of difficulty: *Suburbs*, *Highway*, *Tunnel*, and *Skyway*. The first two environments provide a relatively large amount of geometric constraints via the presence of many buildings and other human-made structures in the vehicle’s surroundings. In *Tunnel* and *Skyway*, the geometric features are either degenerate (not constraining all the movement’s axes) or close to nonexistent, respectively. In these scenarios, only Doppler-induced information can lead to usable odometry performances. The skyway scenario is especially challenging as for a couple of kilometres the RADAR does not observe any significant static features except for sparse lamp posts and weak returns from the side barriers (the RADAR is mounted on the top of the vehicle and the barriers are relatively short, thus the bulk of the emission cone passes over the barriers). Accordingly, the ratio of moving to static elements in the RADAR’s FoV is very high. The length and velocity characteristics of each sequence type are given in Tab. 8.1 along with the results discussed later in Sec. 8.3.3. Fig. 8.1 (left) shows our collection vehicle while recording a *Tunnel* sequence.

Baselines

To benchmark the proposed algorithm, we have replicated four baselines from [25] and [105], three based on RADAR data and one on LiDAR. The first baseline, denoted CT-R, corresponds to the odometry component of the publicly available ‘teach and repeat’ framework [25]. It consists of an ICP-based continuous-time scan registration using point clouds extracted from the raw RADAR data. The trajectory estimates are computed via a sliding window optimization based on a continuous-time GP state representation. Note that the velocity estimates are used to compensate for the Doppler-induced shift of the RADAR points. The second baseline is DG and replicates the work from [105] on per-azimuth radial

velocity extraction, followed by a **RANSAC**-based outlier rejection and robust optimization. With the constant-velocity assumption, it allows for the estimation of the vehicle’s velocity for each **RADAR** scan. Combined with the gyroscope’s angular rate, it enables association-free **RADAR** odometry, even in geometrically feature-deprived environments. Despite not extracting any point cloud out of the **RADAR** data, one can consider the radial velocity as ‘features’ extracted in a ‘front-end’ before an optimization-based state estimation step. Third is CT-RDG, the integration of the per-scan velocity estimates from [105] and the gyroscope measurements within CT-R. Finally, the **LiDAR** baseline denoted CT-LG consists of the continuous-time **ICP** from [25] with additional gyroscope constraints. The various parameters of the baseline, as well as the **RADAR**-specific value of β , have been tuned using an extra sequence of each type.

Results

Sequence type (length, avg. / max. vel.)	CT-R [25]	DG [105]	CT-RDG [105]	CT-LG [25]	DRO-D (ours)	DRO-G (ours)	DRO-GD (ours)
Suburbs (4×7.9 km, 8.1 / 18.6 m/s)	1.22 / 0.37	0.60 / 0.04	0.32 / 0.04	0.15 / 0.07	0.72 / 0.02	0.18 / 0.02	0.19 / 0.02
Highway (4×9.3 km, 11.1 / 27.0 m/s)	1.92 / 0.47	0.74 / 0.04	0.49 / 0.04	0.19 / 0.06	0.84 / 0.02	0.16 / 0.02	0.24 / 0.02
Tunnel (4×1.9 km, 8.9 / 28.2 m/s)	35.90 / 0.96	1.35 / 0.04	0.70 / 0.04	3.35 / 0.11	0.54 / 0.02	7.12 / 0.02	0.34 / 0.02
Skyway (4×11.1 km, 18.2 / 31.7 m/s)	35.24 / 1.77	1.17 / 0.02	0.86 / 0.02	16.98 / 0.24	0.71 / 0.01	45.76 / 0.01	0.40 / 0.01

KITTI odometry metric reported as XX/YY with XX [%] and YY [°/100 m].

Table 8.1. Average relative pose accuracy of the proposed method and several baselines on our dataset (best **RADAR**-based method in bold).

Tab. 8.1 shows the average odometry error (KITTI metrics) obtained with each baseline and our algorithm in the four environments. While a thorough ablation study is conducted in the following subsection, here, we consider three variations of DRO that all rely on the gyroscope integration for the orientation estimation. The first variant, denoted DRO-D, solely uses the Doppler-based velocity constraint O_d to estimate the linear dynamics of the system. The second one, DRO-G, uses solely the direct registration objective function O_ψ . Note that DRO-G still leverages Doppler compensation of the **RADAR** data as part of O_ψ . Ultimately, DRO-GD combines both O_d and O_ψ as presented in Eq. (8.1).

Overall, DRO-GD significantly outperforms every other **RADAR** methods. While the performance gap between DRO-GD and DRO-G is minimal in the well-structured suburban environment, the difference increases as the data becomes more challenging. The best **RADAR** baseline, CT-RDG, shows errors around twice as large as ours throughout all the sequences. When considering ‘Doppler-only’ approaches, DG [105] and DRO-D perform similarly in *Suburbs* and *Highway*. In the more challenging environments, DRO-D displays a noticeable advantage. All these observations suggest that the direct nature of the proposed approach (considering all the **RADAR** information in a single optimization without any feature extraction) and our novel **GP**-based infill for Doppler data provide a greater level of robustness. Without the Doppler-based velocity constraint, DRO-G is not able to reliably perform odometry in challenging scenarios, similar to CT-R. Despite not completely failing on all the *Tunnel* sequences (3 out of 4 runs present a translation error under 2.5%), the Doppler-compensation that happens in O_ψ is not sufficient for robust

estimation in feature-deprived environments. Only the [LiDAR](#) baseline outperforms DRO-G in some feature-dense environments (*Suburbs*). However, CT-LG poorly performs in feature-deprived sequences (*Tunnel* and *Skyway*) while DRO-GD keeps a similar level of performance.

Fig. 8.6 provides a sample of the estimated trajectories for each sequence type. It is interesting to note that despite relatively similar KITTI metrics, CT-RDG and DRO-DG result in fairly different trajectories, with DRO-DG being much closer to the ground truth. Part of the explanation is that the KITTI metrics only consider segments up to 800 m where the small orientation drift of CT-RDG has a weak impact on the position at the end of each segment. However, along the full trajectory length, the orientation drift leads to large position errors that are visually noticeable. This observation supports the fact that for kilometre-long odometry, the direct integration of the gyroscope data can provide better orientation estimates than [SOTA](#) odometry.

Ablation study

We have conducted a thorough ablation study to determine which of the presented mechanisms impacts the algorithm’s accuracy the most. Due to the extremely challenging nature of the *Skyway* sequences, we have chosen to leave them out of this study, as other factors, such as the high proportion of outliers, can have a greater impact on the results. Tab. 8.2 shows the relative pose error of the proposed DRO-GD when various elements are removed from the pipeline. For the ‘no local map’ row, DRO-GD is run replacing the proposed local map updated on the fly with simply the last motion and Doppler-corrected scan. As expected, removing the gyroscope corresponds to the largest performance degradation, leading to accuracy slightly better than CT-R in Tab. 8.1 (also without a gyroscope) for feature-dense environments. For the gyroscope-less operation, we correct the estimated state with an angular velocity bias calibrated using held-off data ($0.034^\circ/\text{s}$ for *Suburbs* and *Highway*, and $0.057^\circ/\text{s}$ for *Tunnel*). We have also applied the robust weighting when the change in orientation between consecutive scans is not realistic. Without the bias correction, we obtain relative translation errors of 1.37%, 1.61%, and 3.67% for *Suburbs*, *Highway*, and *Tunnel*, respectively.

Estimating the system’s orientation in a continuous manner using only [RADAR](#) data is challenging due to the low angular resolution of the sensor (0.9°) and the beam width (1.8°), even when leveraging a motion prior like CT-R. Interestingly, in more challenging scenarios like the *Tunnel* sequences, the drop in our method’s accuracy is not as large as for CT-R thanks to the proposed Doppler-based velocity constraint. Regardless, the velocity [RMSE](#) is not significantly impacted by the absence of a gyroscope. As for the other variations, the difference with DRO-GD is limited, and on average, our algorithm still outperforms the best [RADAR](#) baseline CT-RDG.

In Fig. 8.7, we study the sensitivity of DRO-GD with respect to the γ parameter used in the local map update. One can see that the method’s accuracy is not very sensitive to γ . We also analyze the estimated side velocity bias and empirical evidence that, given the assumption made in our formulation, it depends on the observed environment in Sec. 8.3.6. Overall, this ablation study demonstrates that the proposed direct [RADAR](#) registration is sound and robust.

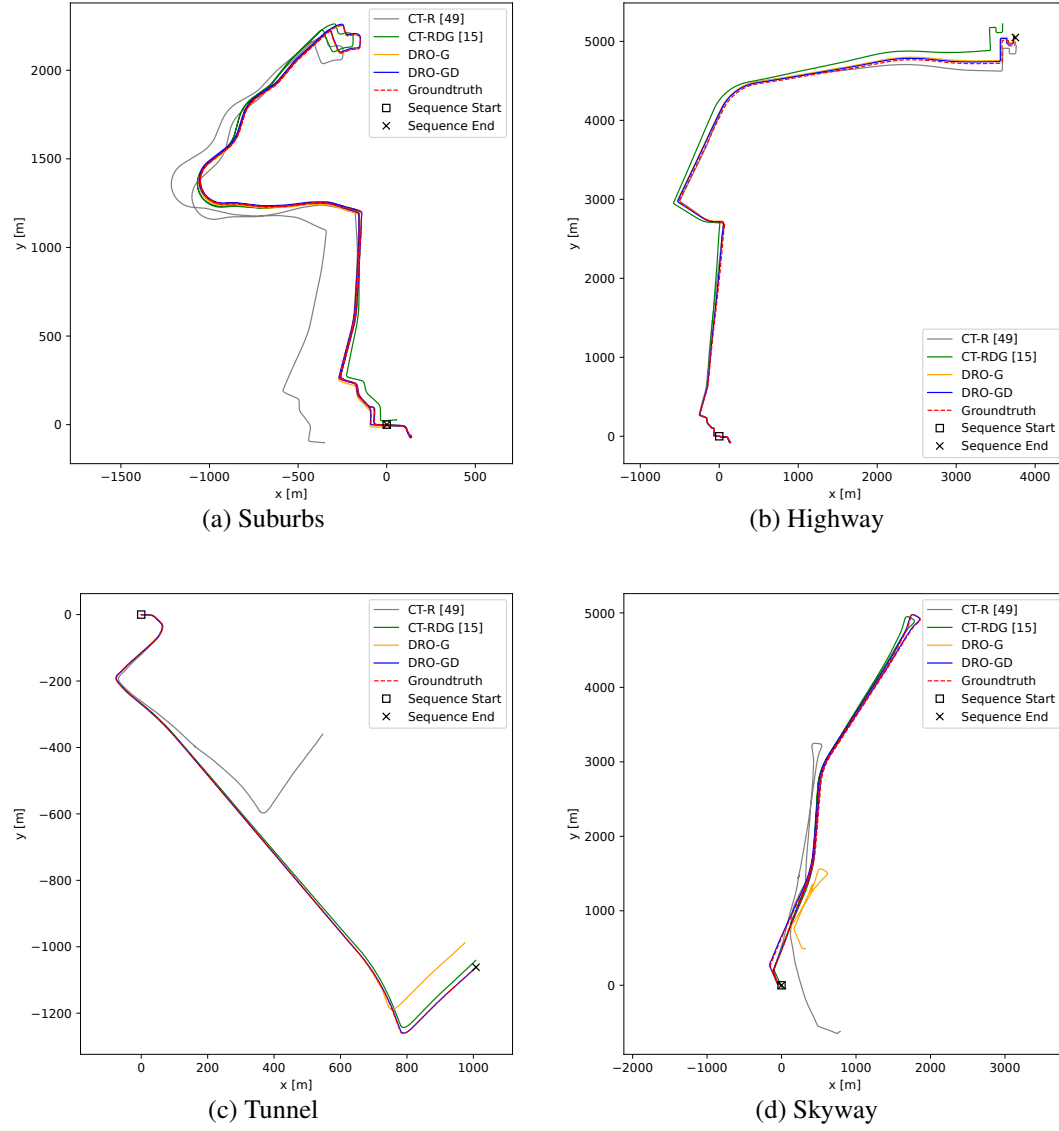


Figure 8.6. Trajectory estimate samples for the different sequence types.

Table 8.2. Pose accuracy ablation study: each row corresponds to a marginal difference with DRO-GD.

Method	Suburbs	Highway	Tunnel	Vel. er.
DRO-GD (ours)	0.19 / 0.02	0.24 / 0.02	0.34 / 0.02	0.119
No gyr. bias	0.27 / 0.03	0.31 / 0.04	0.42 / 0.04	0.119
No vel. bias	0.22 / 0.02	0.29 / 0.02	0.31 / 0.02	0.120
No local map	0.35 / 0.02	0.43 / 0.02	0.40 / 0.02	0.107
No gyroscope	1.00 / 0.24	1.56 / 0.38	2.92 / 0.99	0.132
KITTI odometry metric reported as XX/YY with XX [%] and YY [$^\circ/100$ m] the trans. and rot. errors, respectively.				Avg RMSE [m/s]

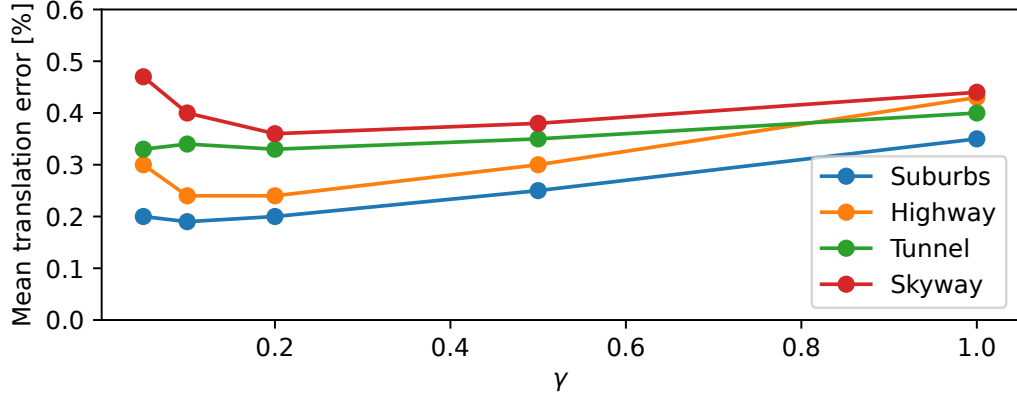


Figure 8.7. Sensitivity analysis of DRO-GD with respect to the local map update parameter γ .

8.3.4 Public automotive benchmarks

We have benchmarked the proposed method against state-of-the-art frameworks over the Boreas and MulRan datasets. In both cases, the [RADAR](#) uses a sawtooth modulation pattern. Thus, none of the results presented in this section include the Doppler-based objective function O_d .

Boreas dataset

The Boreas dataset [27] was collected by driving a vehicle equipped with a sensor suite composed of a 3D [LiDAR](#), a 2D spinning [RADAR](#) (sawtooth pattern), a front-facing camera, and an [RTK-GNSS/INS](#) solution for ground truthing. The dataset consists of 44 sequences collected over one year along more than 350 km of roads in Toronto, Canada. Accordingly, the data corresponds to various weather conditions ranging from clear summer sky to heavy snow and rain. A specificity of Boreas is that most sequences follow a unique route in a suburban environment, allowing one to test an algorithm’s robustness with respect to changing conditions. The Boreas dataset also offers a leaderboard for odometry benchmarking that leverages 13 of the sequences above. Note that the [GNSS/INS](#) ground truth is not publicly available for these sequences.

After testing our algorithm on 6 sequences with ground truth, we submitted our results to the leaderboard with DRO-G. Our algorithm significantly outperforms all the other methods. We have reported the four best [RADAR](#)-based methods in Tab. 8.4. The core principle of CFEAR [3] and STEAM-RIO++ [24] is similar as both methods extract a point cloud from the incoming [RADAR](#) data before performing registration to multiple previous scans. A difference is that CFEAR first performs motion distortion correction based on the previous estimate and uses point-to-distribution residuals, while STEAM-RIO++ leverages a [GP](#) motion prior, point-to-point constraints and [IMU](#) data. CFEAR++ [101] builds atop CFEAR by embedding the use of the [IMU](#) and some semantic information.

Our results confirm the ability of DRO-G to provide state-of-the-art performances without requiring a [RADAR](#) with a triangular modulation pattern and O_d as long as the environment contains sufficient geometric cues. In Tab. 8.3, we present the detailed results obtained on the 13 sequences of the Boreas dataset leaderboard.

Table 8.3. Per-sequence errors of DRO-G on the Boeras dataset leaderboard.

Sequence ID	Weather conditions	Trans. err. [%]	Rot. err. [°/100m]
2020-12-04-14-00		0.24	0.05
2021-01-26-10-59		0.20	0.04
2021-02-09-12-55		0.22	0.02
2021-03-09-14-23		0.23	0.06
2021-04-22-15-00		0.25	0.05
2021-06-29-18-53		0.36	0.02
2021-06-29-20-43		0.44	0.04
2021-09-08-21-00		0.30	0.02
2021-09-09-15-28		0.53	0.19
2021-10-05-15-35		0.12	0.02
2021-10-26-12-35		0.13	0.01
2021-11-06-18-55		0.32	0.10
2021-11-28-09-18		0.10	0.01

: Overcast, : Snow coverage, : High snow coverage
: Snowing, : Sun, : Rain, : Dusk, : Night

Table 8.4. Boreas dataset SE(2) odometry leaderboard.

Method	Trans. err. [%]	Rot. err. [°/100m]
DRO-G (ours)	0.26	0.05
CFEAR++ [101]	0.51	0.14
CFEAR [4]	0.61	0.21
STEAM-RIO++ [24]	0.62	0.18

As expected, there seems to be no correlation between the weather conditions and the translation error. An interesting observation is that the orientation estimates of DRO-G using solely the gyroscope integration outperform the other inertial-aided methods. This means that the **RADAR** constraints in CFEAR++ and STEAM-RIO++ play a negative role in the orientation estimates.

MulRan dataset

The MulRan dataset [91] has been recorded with a **LiDAR**, 2D spinning **RADAR**, and an **IMU** mounted atop a car while driving in 4 different locations in Daejeon and Sejong, South Korea. This dataset originally targeted the task of place recognition, but the availability of a 6-DoF trajectory ground-truth also enables the benchmarking of odometry and **SLAM** algorithms. However, unlike in the previous experiments, the ground truth is not obtained with an **RTK-GNSS** but via the optimization of a pose graph that leverages wheel odometry, an optical-fibre gyroscope, and a VRS-GPS. Additionally, the quality of the **IMU** is significantly lower than in the Boreas and our automotive datasets.

Table 8.5. Average error per sequence type of the MulRan dataset.

Sequences	CFEAR-3 [4]	CFEAR-3-50 [4]	DRO-G (ours)
KAIST	1.65 / 0.70	1.53 / 0.68	1.44 / 0.42
DCC	1.75 / 0.49	1.58 / 0.50	1.54 / 0.41
Riverside	1.46 / 0.51	1.39 / 0.50	1.38 / 0.31
Average	1.62 / 0.57	1.50 / 0.56	1.45 / 0.38

KITTI odometry metric reported as XX / YY with XX [%]
and YY [°/100 m] the translation and orientation errors, respectively.

Tab. 8.5 shows the KITTI odometry errors of the proposed method and variations of CFEAR as reported in [4]. As introduced in Sec. 8.3.4, CFEAR relies on a scan-to-local-keyframes registration. C-FEAR-3 and C-FEAR-3-50 refer to [4] using a different number of past keyframes in the registration process: 4 and 50, respectively. The table shows the ability of DRO-G to outperform SOTA methods even with a low quality IMU. The level of rotational accuracy highlights the soundness of the proposed integration-only orientation and reiterates the observation from the Boreas leaderboard that RADAR data can have a detrimental impact on the system’s rotation estimate compared to pure integration and a simplistic bias estimation strategy.

Surprisingly, both CFEAR and our method display errors significantly larger with MulRan data than with the Boreas dataset. We believe that the nature of the ground truth generation process leads to noisier poses, but also that extrinsic calibration and synchronization are suboptimal. For example, we observed that the difference between consecutive IMU data timestamps presents a variation of $\pm 35\%$ around the nominal period of 10 ms and that there is a constant offset with respect to the other sensors. In our experiments, we corrected the IMU synchronization by removing 50ms to each inertial timestamp of all sequences. We also adjusted the RADAR heading by 0.172° (without the heading correction DRO-G leads to errors of 1.46% and $0.38^\circ/100\text{m}$). For completeness, without the gyroscope bias estimation, we obtain errors of 6.00% and $1.57^\circ/100\text{m}$ on average. Due to the lower quality of the dataset, we believe no strong conclusion can be drawn from this experiment other than the demonstration that DRO-G does not require a high-end gyroscope to perform at the level of SOTA methods.

8.3.5 Off-road navigation

To demonstrate the versatility of the proposed method, we collected a series of RADAR-inertial data sequences with an off-road mobile robot.

Dataset description

Our off-road dataset comprises 16 test sequences divided into four locations ranging progressively from highly structured to unstructured environments. The robot’s maximum velocity is approximately 1.2 m/s. The sensor suite comprises a Navtech RAS3 RADAR and an Ouster LiDAR with its embedded IMU. The robot is driven over the same loop three or four times for each environment. The easiest one, *Parking*, takes place in a parking lot with many artificial geometric features such as buildings, fences, and parked cars. The second

Table 8.6. Average RPE in various environments spanning over different levels of structure.

Sequence type	CT-RG ^[128] no-bias-est	CT-RG ^[128]	DRO-G no-bias-est.	DRO-G (ours)
Parking (4×368 m)	3.45	1.03	3.44	1.04
Industrial (4×390 m)	3.63	0.77	3.57	0.62
Grass (4×175 m)	5.63	1.05	5.29	0.67
Woods (4×421 m)	4.70	1.30	3.42	0.71

RPE as a percentage [%] of the distance travelled.

location *Industrial* takes place around a large industrial building with part of the trajectory going over a grass patch. This environment contains a combination of geometric structures (fences and buildings) and sparse open areas (trees and bushes) at the back of the industrial building. For the third trajectory, *Grass*, the robot moves over a grass field and occasionally can observe a metallic fence. The last and most challenging environment, *Woods*, is entirely off-road in the woods. Fig. 8.1 (right) shows our Clearpath Warthog robot within the *Woods* environment.

Baseline

In this experiment, we benchmark DRO-G against a version of CT-R [25] with additional inertial constraints as in [128]. We denote this ICP-based baseline as CT-RG. For each CT-RG run, the gyroscope measurements have been corrected with a constant bias estimated from five seconds of data where the robot is static. CT-RG’s parameters have been tuned using held-out *Parking* sequences.

Results

We show the RPE obtained with and without gyroscope bias estimation in Tab. 8.6. In structured environments (*Parking* and *Industrial*), both the direct and point-cloud-based methods perform similarly. The discrepancy between the two methods increases when dealing with more challenging environments. Overall, DRO-G maintains a satisfactory level of accuracy across all scenarios, demonstrating that the level of structure of the environment does not seem to impact the accuracy significantly. It also shows that direct RADAR registration is robust to the nature of the environment without requiring any tedious parameter tuning. Fig. 8.8 provides a visualization of estimated trajectories in each environment. Tab. 8.6 and Fig. 8.8 also show the efficacy of the proposed gyroscope bias estimation (DRO-G no-bias-est. vs DRO-G). Unlike the automotive scenario, where the gyroscope bias is nearly negligible, the lower quality of the LiDAR’s IMU has a significant impact on the results if the bias is not accounted for. Both methods perform poorly without any bias correction.

8.3.6 Lateral velocity bias estimate

This section discusses the lateral velocity bias estimation introduced in Sec. 8.3.1. Fig. 8.9 presents the lateral velocity estimates obtained across the different sequences used in Sec. 8.3.3. The value of the bias is shown as a function of the vehicle’s position along the

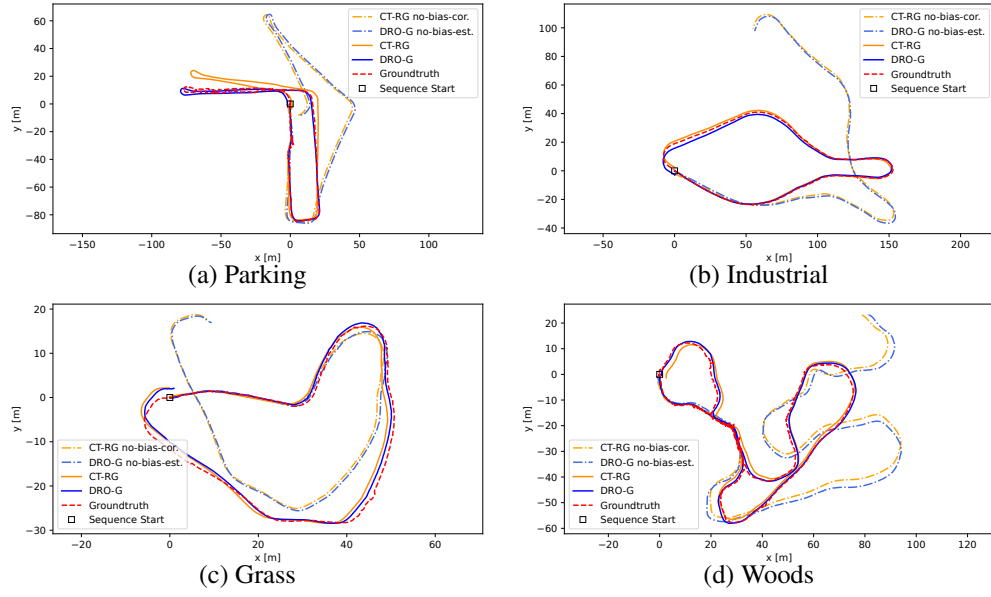


Figure 8.8. Visualization of trajectory estimates in the off-road scenarios (aligned with the ground truth for visualization using the first 20% of the trajectory).

trajectory. Note that the *Highway* and *Tunnel* sequences are split into two plots as they have been collected separately in one direction and the other. As one can see, there is a clear correlation between the location of the vehicle and the value of the velocity bias estimate. This observation supports our hypothesis that the Doppler-based velocity bias is impacted by the structure of the environment. Note that the extrinsic calibration between the **RADAR** and the vehicle rear axle has been thoroughly verified, and thus, the miscalibration hypothesis has been discarded. Further work will be required to obtain a deeper characterization of this bias and a clearer understanding of the physical phenomenon behind it.

8.3.7 Computation time

All the experiments in this paper were performed using a laptop equipped with an Intel i7-13850HX **CPU** and an Nvidia RTX 5000 Mobile **GPU**. On our automotive dataset, the computation of DRO-GD took on average 89 ms per **RADAR** frame using the **GPU**. This is far below the real-time limit of 250 ms (4 Hz **RADAR**). When considering solely the Doppler-based velocity objective function, DRO-D required only 34 ms per scan. For comparison, we also changed the target device to **CPU** in Pytorch. We obtained averages of 738 ms and 59 ms for DRO-GD and DRO-D, respectively. On the Boreas and MulRan datasets, the **GPU** computations needed 79 ms and 95 ms per scan, respectively. The off-road experiments were the ones that required the most computation time with 134 ms per frame. To summarise, all our experiments have run in real-time. However, the current implementation is somewhat modular, allowing for different combinations of objective functions and motion models. We believe that specializing the code and leveraging a compiled programming language would lead to significant improvement in the proposed method's efficiency, especially for **CPU**-based operations.

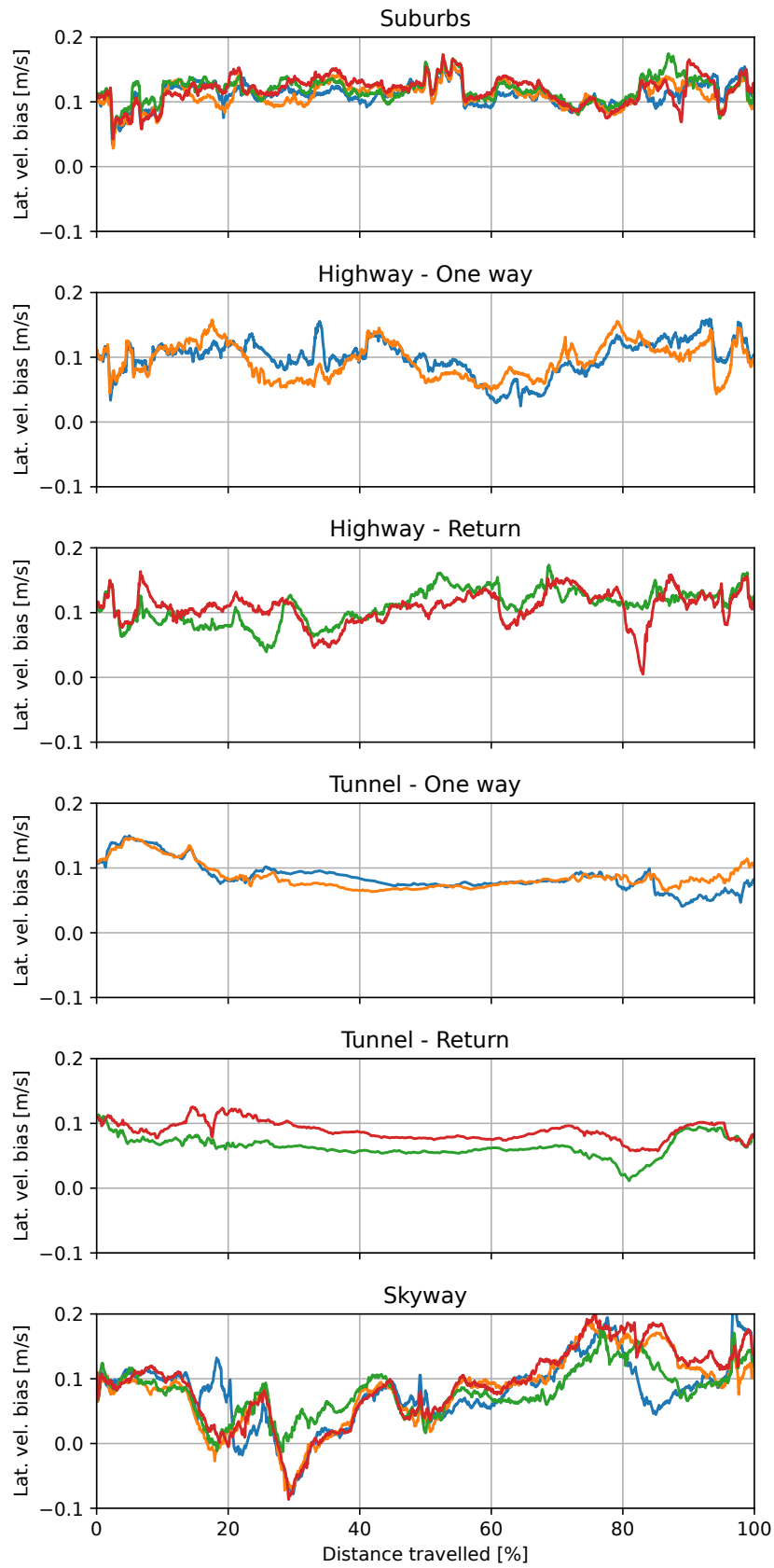


Figure 8.9. Lateral velocity bias estimate obtained by running DRO-GD on our automotive dataset. The bias is expressed as a function of the distance travelled in the sequence.

8.4 Limitations

Currently, the proposed framework strongly relies on the gyroscope data as the orientation is obtained via direct integration of the angular velocity. We demonstrated that it was sufficient to obtain [SOTA](#) performance in both automotive and off-road scenarios with high-quality and low-quality [IMU](#). However, significant degradation of the inertial data (saturation, long dropouts, erroneous measurements, etc), while unlikely, would have a catastrophic impact on the estimated state. On a similar topic, the proposed approach to estimate the gyroscope bias requires the vehicle/robot to be static occasionally. Further work is needed to introduce a principled approach to gyroscope bias estimation in DRO.

Our method and experiments focused on spinning [FMCW RADARs](#). Thus, our findings are limited to this particular type of sensor. However, we believe that DRO applies to other [RADAR](#) types (e.g. phased arrays) both in 2D and 3D given a sufficient [FoV](#) and access to the intensity returns (unfortunately, most off-the-shelf automotive [RADARs](#) do not provide this information). A practical limitation of using 3D data is the potential computational load increase if the [RADAR](#) provides much more information than the [RADARs](#) used in this work.

8.5 Conclusion

In this chapter, we presented **Direct RADAR Odometry (DRO)**, a direct method for real-time odometry estimation using spinning millimetre-wave [RADAR](#). This work extends the direct error minimization paradigm introduced earlier in the thesis, initially applied to [LiDAR](#) sensors through the MD-SLAM framework (Chapter 3), to the [RADAR](#) domain, which is inherently more challenging due to its lower spatial resolution, non-trivial signal distortions, and the presence of Doppler effects.

Unlike most [RADAR](#) odometry systems in the literature, which operate through indirect pipelines based on sparse point or feature extraction [4, 101], DRO avoids this intermediate representation entirely. Instead, it works directly on the [RADAR](#)'s raw polar intensity images, formulating odometry as a scan-to-map registration problem solved through continuous, gradient-based optimization in $\mathbb{SE}(3)$. This direct approach retains the full information available in the sensor data and eliminates the need for computationally expensive data association, fragile thresholding, or tuning-heavy filtering stages.

One of the key contributions of this work is the incorporation of sensor-specific physics into the estimation pipeline. DRO explicitly models both the motion distortion due to continuous movement during scanning and the Doppler-induced frequency shifts that arise from the relative motion between the [RADAR](#) and scene elements. In scenarios where the [RADAR](#) uses a triangular frequency modulation pattern, common in many automotive-grade sensors, these Doppler shifts become asymmetric, rendering the radial component of velocity observable. We exploit this property to constrain motion estimation in scenes that lack reliable geometric cues, such as featureless tunnels, where conventional point-based methods tend to fail catastrophically.

Orientation is obtained through integration of yaw-axis gyroscope measurements, while body-centric velocity is estimated via the maximization of one or two objective functions, depending on the [RADAR](#) emission profile. Unlike prior work that assumes correspondence between consecutive [RADAR](#) beams [33], our method treats each beam independently,

enabling Doppler exploitation even in unstructured or dynamic scenes. To our knowledge, DRO is the first method to integrate both Doppler and motion distortion effects into a unified, fully differentiable estimation framework for RADAR odometry.

To validate our approach, we implemented a real-time GPU-based version of DRO and evaluated its performance on more than 250 km of RADAR data. This includes sequences from public benchmarks such as Boreas [27], which contains varied urban and suburban driving conditions, and a new off-road dataset that we collected. The results show that DRO outperforms current state-of-the-art RADAR odometry pipelines both in structured urban environments and in unstructured terrain, while maintaining real-time performance at 10–30 Hz on a standard NVIDIA GPU. These experiments underscore the core message of this thesis: that direct methods, when combined with physically grounded sensor models and efficient numerical solvers, offer a powerful and generalizable alternative to traditional indirect state estimation pipelines.

Future work on DRO will focus on several key extensions. First, the integration of a principled bias estimation mechanism for the gyroscope would improve long-term orientation estimation and reduce drift, especially during slow or stationary phases. Second, we aim to embed DRO within a full SLAM system, enabling loop closure detection, map reuse, and global trajectory optimization. This would allow RADAR-based systems to perform large-scale navigation and mapping in conditions where vision and LiDAR fail, such as fog, dust or heavy precipitation. Moreover, ongoing developments in RADAR hardware, including higher angular resolution and Doppler sensitivity, may further enhance the utility of direct RADAR odometry in the years to come.

Chapter 9

Conclusion

IN this concluding chapter, we revisit the core challenge of [SLAM](#) and the rationale for embracing direct error minimization across heterogeneous sensor modalities. We then recap the principal contributions of each chapter. Finally, we reflect on the broader implications of our work and outline promising directions for future research.

9.1 Summary

In this thesis, we have developed and evaluated direct error minimization techniques for sensors beyond conventional cameras. Although direct methods are well established for visual odometry and [SLAM](#), they have seen little application to [LiDAR](#) and [RADAR](#) data. By formulating alignment and motion estimation directly on raw range and reflectivity measurements, we eliminate the need for explicit feature extraction and matching. Our results show that this approach delivers reliable performance in challenging scenarios, and in the case of [RADAR](#) it achieves a clear breakthrough over the previous state of the art. Finally, leveraging [GPU](#) acceleration played a crucial role in realizing real-time implementations of these methods for autonomous systems.

Chapter 3 presents a novel, direct 3D [SLAM](#) pipeline that operates seamlessly with both RGB-D and [LiDAR](#) sensors. Only minor modifications to the projection model are needed to accommodate each modality. The system combines dense pose tracking with an appearance-based relocalization module to handle large loop closures, validating each closure using the same direct registration algorithm employed for odometry. It is the first pipeline to apply direct error minimization to [LiDAR](#) data. A C++/CUDA implementation enables real-time performance on both sensor types.

In Chapter 4, we introduce a new benchmark dataset that provides RGBimagery, 3D [LiDAR](#) scans, [IMU](#) measurements with an high-precision 6 [DoF](#) ground truth. The sequences span both indoor and outdoor environments and cover diverse motion patterns—hand-held and vehicle-mounted. For the [LiDAR](#) data, we provide two acquisition modes: a denser point cloud for hand-held operation and a higher frequency rate for car-mounted runs. This variability lets us systematically analyze how sensor configuration affects the performance of our direct [LiDAR](#) direct methods.

To produce reliable data, we had to tackle two core challenges: accurate multi-sensor calibration and provision of precise 6 [DoF](#) ground truth. These are addressed in Chapter 5 and Chapter 6, respectively.

In Chapter 5, we focus on LiDAR–RGB calibration. Accurate knowledge of the relative pose between sensors is a prerequisite for fusion, but existing techniques often depend on large or specialized targets to overcome the sparsity of LiDAR returns. Instead, we employ a simple A3 chessboard pattern and exploit its planar geometry to establish correspondences between LiDAR points and image features. A nonlinear optimization then refines the relative pose, yielding robust, high-accuracy calibration with minimal manual intervention.

Then in Chapter 6, we extend the work presented in Chapter 3 with a unified photometric BA method, refining SLAM-derived trajectories for maximized global photometric consistency.

In Chapter 7, we present a preliminary study on the effect of skewing, or motion distortion, in 2D LiDAR scans. Rather than estimating the robot’s pose as a single rigid-body transform, we parameterize its state by continuous linear and angular velocities. This formulation enables us to reconstruct the sensor’s motion over each sweep and remove distortion from the raw scans. We demonstrate the method’s effectiveness within a 2D LiDAR pipeline, establishing a foundation for our final work on RADAR.

In Chapter 8, we present our final contribution: a direct $\mathbb{SE}(2)$ odometry method for spinning FMCW RADAR. The pipeline registers each incoming scan directly against a local map using the full RADAR intensity data, without extracting features or generating point clouds. Trajectories are estimated in a locally continuous manner, with compensation for motion and Doppler distortions within each sweep. When the RADAR’s frequency modulation pattern makes radial Doppler velocities observable, we add a Doppler-based constraint that improves velocity estimation and enables reliable odometry in geometrically sparse environments (for example, featureless tunnels). We validate this approach on over 250 km of on-road data and show that, with gyroscope assistance, it outperforms existing methods, achieving an average relative translation error of 0.26 %, which is half that of the current SOTA. With Doppler-enabled modulation, the translation error further drops to 0.18 %.

9.2 Outlook

A key prerequisite for the success of direct error–minimization methods is a sufficiently dense data representation. Sensors that produce sparse or irregular measurements, such as many solid-state LiDARs or certain 4D RADAR designs, are not directly compatible with our framework without additional interpolation, sensor fusion or measurements stacking. Future work should investigate how to adapt direct methods to these emerging modalities, perhaps by combining them with complementary high-rate sensors or learned upsampling of sparse returns.

Motion distortion during data acquisition is another critical factor. Our MD-SLAM experiments in Chapter 3 demonstrated that global-shutter cameras significantly outperform rolling-shutter devices in direct alignment, achieving state-of-the-art accuracy only when shutter-induced distortions are eliminated. The effect is even more pronounced for LiDAR: with scan rates much lower than camera frame rates, raw point clouds suffer from visible skew unless corrected. As shown in Chapter 7 and Chapter 8, compensating for continuous motion, for example using onboard IMU data in the LiDAR case, is essential. Extending these continuous-time formulations to other sensors and improving the tight coupling between motion estimation and distortion correction remain promising research directions

for the LiDAR case.

RADAR introduces a distinct form of motion-induced error: Doppler distortion. Although LiDARs, operating at light frequencies, experience negligible Doppler effects at typical platform speeds, spinning FMCW RADARs, operating at radio frequencies, can exhibit measurable range shifts when moving rapidly. As discussed in Chapter 8, incorporating Doppler-based constraints into the direct registration process improves odometry in feature-deprived scenarios. Future RADAR approaches may benefit from adaptive modulation schemes that maximize Doppler observability, and from joint estimation of motion and reflectivity changes. Regarding LiDAR, compensating for Doppler effects is generally unnecessary unless the platform is moving at very high speeds.

Map representation is another open challenge. For RADAR, our current map is simply an accumulation of Cartesian intensity scans over time, sufficient for local-map registration but limited in expressiveness. Correlating also the space in the local-map update could significantly improve the robustness of the pipeline. LiDARs maps, by contrast, demand more sophisticated representations (e.g., surfels, octrees, or implicit functions) to support direct registration at scale. Investigating compact yet dense map structures that balance memory footprint, update speed, and registration accuracy will be crucial for long-term autonomy.

Looking ahead, emerging volumetric reconstruction techniques offer exciting possibilities. Neural Radiance Fields (NeRFs) learn a continuous volumetric scene representation from sparse 2D images, encoding radiance as a function of 3D position and viewing direction. Their ability to synthesize high-quality novel views and capture fine details makes them a compelling candidate for generating dense, photometrically consistent maps. Similarly, Gaussian splatting provides a sample-based volumetric approach where each Gaussian's weight decays with distance, enabling fast and smooth merging of data from multiple sensors or viewpoints. Integrating such learned or sample-based volumetric maps with direct pose estimation could yield novel-view guided error metrics and improve robustness in visually challenging conditions.

Finally, our work has assumed a static environment, treating moving objects as outliers. A truly comprehensive SLAM system must instead model dynamic elements, like vehicles and nonrigid, deformable objects such as humans. Early efforts in dynamic scene reconstruction (e.g., [114], [96]) have shown the feasibility of jointly estimating motion and shape. Extending direct error-minimization techniques to these settings, perhaps by incorporating object-centric motion priors or online shape deformation models, represents a significant but rewarding frontier for future research.

List of Figures

2.1	Illustration of the pinhole camera model. The sensor frame S (used instead of C to generalize the notation across sensors) is located at the optical center. The image plane is mirrored in front of the camera for clarity. Crosses indicate intersections with the image plane, while filled circles mark the corresponding 3D points in space. The depth d (in green) denotes the distance from the camera center to the 3D point along the z -axis. Illustration courtesy of Di Giammarino [43].	9
2.2	Single laser beam operation illustrating the measurement principle described in Eq. (2.5). Illustration courtesy of Di Giammarino [43].	11
2.3	Spherical projection illustration. The LiDAR frame typically uses a z -up convention. The red lines represent spherical coordinates: azimuth θ , elevation ϕ , and range ρ . Illustration courtesy of Di Giammarino [43]. . . .	12
2.4	Comparison between projection methods: ID-based (top) and spherical (bottom).	13
2.5	An example of RADAR measurement: polar representation (top) and corresponding Cartesian (bottom).	16
2.6	Impact of different frequency modulation patterns (top) and Doppler effect on the data collected with spinning FMCW radars (bottom).	17
2.7	Visual illustration of the GN algorithm. Starting from an initial state $\mathbf{x}_0$, the cost function (gray curve) is approximated locally by a quadratic function (orange curve), represented as a parabola in one dimension. The updated state $\mathbf{x}_1$ is computed by adding the increment $\Delta\mathbf{x}$ to the previous state, and the procedure is iterated until convergence at the final state $\mathbf{x}_3$. Illustration courtesy of Di Giammarino [43].	21
2.8	Relationship between a Lie group and its Lie algebra. The Lie algebra $\mathfrak{m} = T_{\mathcal{E}}\mathcal{M}$ (red plane) is the tangent space to the group manifold $\mathcal{M}$ (blue sphere) at the identity element $\mathcal{E}$. A straight line $\mathbf{v}t$ in the Lie algebra is mapped to a geodesic $\exp(\mathbf{v}t)$ on the manifold via the exponential map, establishing a one-to-one correspondence between the two. This link lets us perform calculations in the flat algebra and transfer the results back to the nonlinear group. The S^2 sphere sketched in $\mathbb{R}^3$ is a 2-D drawing aid; the actual Lie group is the 3-sphere $S^3 \subset \mathbb{R}^4$ that represents unit quaternions. Illustration courtesy of J. Solà [146].	23

3.1	Scenes reconstructed using our pipeline. Top: results of a self-recorded dataset using Intel Realsense 455 RGB-D. Bottom: using LiDAR OS0-128 of the <i>cloister</i> sequence from the Newer College Dataset [170]. Red pyramids represent the keyframe poses of the sensor; green lines show the factors between them.	28
3.2	Illustration of our system. Range $\mathcal{I}_t^p$ and $\mathcal{I}_t^i$ images are taken as input from the system. An optimized trajectory within a map is produced as output. This system works independently both for RGB-D and LiDAR.	30
3.3	Cues generated for LiDAR (top) and RGB-D (bottom) images. The first row/column shows the intensity $\mathcal{I}^i$, the middle shows the range $\mathcal{I}^p$, and the last one illustrates the normals encoded by color $\mathcal{I}^n$. The red pixels on the intensity cues are invalid measurements (i.e., range not available).	31
3.4	Qualitative RGB-D reconstructions showing the global consistency produced by our pipeline. Data has been self-recorded with an Intel Realsense 455.	34
3.5	MD-SLAM map on <i>long</i> sequence from Newer College Dataset [129] aligned with Google Earth.	37
3.6	Some scenes from Newer College dataset reconstructed by our system. . . .	38
3.7	Alignment of our estimate with the ground truth in <i>long</i> sequence of Newer College. The color bar on the right shows the translational error [m] over the whole trajectory.	39
3.8	Absolute error comparison of summation algorithms. The y-axis marks 0 at the numerical precision threshold ($\sim 10^{-16}$), making it explicit in log scale. Kahan summation (blue) consistently maintains this level of accuracy, unlike naive (green) and pairwise (orange, black) approaches.	40
4.1	A summary of our dataset. Data illustrating some of the sequences recorded (top). 3D mapping done with of our ground truth (bottom).	44
4.2	Comparison between LiDAR clouds attached to ground truth trajectories of KITTI (up) and ours (down). The zoom shows the elevation view.	45
4.3	Projection of the KITTI LiDAR point cloud into an image plane (up), projection of our LiDAR into an image plane (down). The many holes of the up image due to uneven distribution of the LiDAR beams and calibration issues make the KITTI LiDAR image unusable for computer vision tasks. . .	46
4.4	Sensor setup and reference frames. Our ground truth is expressed in the LiDAR reference frame RF_L . More details can be found in our website and supplementary materials.	47
4.5	The image shows the overlay of the 3D model obtained from our ground truth system and a view from Google Maps.	53
4.6	Our benchmark. Cumulative RPE [%] and ATE RMSE [m] across all training sequences in our dataset for KISS-ICP, F-LOAM and ORB-SLAM3. . .	55
4.7	Trajectory plots of LeGO-LOAM, LeGO-LOAM (SLAM mode), and F-LOAM. Each plot is aligned to ground truth using a rigid transformation. . .	56
5.1	Reprojection of a LiDAR point cloud on a fisheye RGB camera rigidly attached to the former. The offset between the sensors leads to shadows on parts of the image.	58

5.2	Here $\mathbf{e}_n$ is the error for the normal term while $\mathbf{e}_d$ represents the plane distances.	61
5.3	Diagram of our calibration pipeline. Measurements are acquired, and calibration target detection is performed (L planar detection is performed via human intervention). The set of planes is used to solve the non-linear optimization problem, leading to the optimal relative pose between the sensors.	63
5.4	Acquisition system used for the Real Case experiments. We report nominal measurements between the sensors. The Realsense T265 (180° horizontal field of view) is installed closer to the LiDAR, while the two Manta cameras (90° horizontal field of view) are mounted in a further wide horizontal stereo baseline.	65
5.5	(a) Average camera-wise calibration error in the L-T265, wide FoV case. (b) shows the average camera-wise calibration error in the LiDAR-Manta case while	66
5.6	Qualitative samples showing L cloud projection on RGB image, in the setting sketched in Fig. 5.4, where the parallax between the sensor is approximately 66 cm.	68
6.1	Reconstruction of Viterbo city-center (Italy) using our data recorded with an OS0-128. The trajectory, which is about 2 km long, has been estimated first with MD-SLAM Chapter 3 [67] and then refined with our photometric BA strategy. This image highlights both the global and local consistencies. We show reconstruction details with multiple scans of the same places acquired over time.	71
6.2	The flow of our approach. The input of our system contains the initial guess of the sensor pose, the intensity image $\mathcal{I}^g$ and the depth image $\mathcal{I}^d$. MD-SLAM already provides the input in the correct format since the preprocessing step is embedded in the SLAM system. The core of our refinement strategy, highlighted in red in this graph, consists in creating a graph with photometric observations (Sec. 6.2.1) and running a global optimization on the set of poses $\mathbf{X}_{1:N}$, as detailed in Sec. 6.2.2.	73
6.3	An example of pose-pairs associated, our BA strategy relies on the association of $\mathbf{X}_i$ and $\mathbf{X}_j$ if they share observations. In the picture above, the input images with depth and normals are on the left, and on the right is the reconstructed model using our methodology. Note that in reality, inputs of our BA are pyramids (as discussed in Sec. 3.2.1, i.e., images of different resolutions). Here, we show just one level for simplicity. The data used is from ETH3D.	74
6.4	The effect of hierarchical optimization when performing BA. From left to right, we show how the quality of the estimate improves, starting from a bad initial guess. The heatmap is normalized between 0 and 0.005 [m] for better visualization. The data is self-recorded using an Intel D455.	75
6.5	The contribution of each cue in our BA strategy. Our strategy uses three types of information (grayscale/intensity g , depth d , and surface normals n), the histogram shows the contribution of pairing each of them for RGB-D and L with respect to the initial guess in terms of ATE. Overall, using the three information together perform always better compared to coupling only few of them, for both the sensors.	76

6.6	Runtimes of our BA strategy evaluated on multiple commercial GPUs and our Intel Core i7-7700K CPU. Cumulative time and standard deviation spent on an iteration of optimization. In this experiment, the finest level includes around 86M pixels, the middle level includes 22M pixels, and the coarser level has around 5M pixels. Time is expressed in seconds.	77
6.7	Some views of reconstructions using our photometric BA schema. On the left reconstruction of working table using an Intel L515 belonging to the RGB-Dfamily. On the right a view of the cloister sequence of Newer College Dataset [170] recorded with OS0-128, belonging to the spinning LiDAR family.	78
6.8	Results of fusion experiments. Plots show how the initial perturbation affects the convergence basin and the algorithm's accuracy. On the x-y axis, respectively, translation [m] and rotation [rad] perturbations, the value is denoted by the mean between rotation and translation error in log scale. Fusing the two sensors with our straightforward approach always shows better results compared to single sensors.	81
6.9	Plots of trajectory estimates in different LiDAR sequences of Newer College [170].	83
7.1	Effect of de-skewing. (a) Unprocessed (raw) scan acquired by a rotating robot, while (b) is the corresponding ground truth scan, and finally (c) is the same scan in (a), processed (de-skewed) using our proposed approach. . .	85
7.2	Illustration of the planar patches calculation and of the correspondence search between two matching patches $\mathbf{m}_i$ and $\mathbf{m}_j$	89
7.3	Iterative evolution of the de-skewing process. Purple points represent skewed points, green points are the ground truth points, and the blue lines represents the data association. (a): initial configuration, instance of full scan acquisition, with RMSE = 0.4 m. (b): after 4 iterations of processing (de-skewing) the raw scan with the proposed approach, decreasing error to RMSE = 0.2 m. (c): after 20 iterations, ending up with a de-skewed scan closer to the ground truth, with RMSE = 0.074 m.	91
7.4	Evolution of estimated and real velocities.	92
7.5	SLAM Results. (a) Ground truth map (reference). (b) Map constructed using raw scans acquired directly from the LiDAR. (c) Map constructed using processed LiDAR measurements de-skewed by our proposed approach. (d) Estimated trajectories from SLAM, produced by scan matching between cosecutive measurements.	94
8.1	This chapter presents a Direct RADAR Odometry (DRO) method that provides state-of-the-art performance in a wide range of environments, including unstructured ones (right) and geometrically degenerated ones such as featureless tunnels (left).	96
8.2	Overview diagram of the proposed direct RADAR odometry method. To leverage the Doppler-based velocity constraint, the RADAR must use a triangular frequency modulation pattern. Note that O_ψ accounts for motion and Doppler-induced distortion.	99

8.3	Illustration of the proposed infilling. The raw Doppler distorted RADAR data (left) is split between up and down-chirp azimuths. GP regression is used to interpolate every second azimuth (middle two, the interpolated portion is in grey). Both $\mathcal{I}^\uparrow$ and $\mathcal{I}^\downarrow$ virtually observe the same geometry without assuming that two consecutive azimuths of the raw data observe the same part of the environment (right, with blue positive and red negative). . . .	101
8.4	Illustration of the GP-based infill highlighting the Doppler effect on RADAR data collected with a triangular frequency modulation pattern (the blue arrow is the sensor velocity).	101
8.5	Illustration of a challenging situation with a line of vehicles coming the other way while driving over a skyway (very few static elements visible in the RADAR data).	104
8.6	Trajectory estimate samples for the different sequence types.	109
8.7	Sensitivity analysis of DRO-GD with respect to the local map update parameter γ	110
8.8	Visualization of trajectory estimates in the off-road scenarios (aligned with the ground truth for visualization using the first 20% of the trajectory). . . .	114
8.9	Lateral velocity bias estimate obtained by running DRO-GD on our automotive dataset. The bias is expressed as a function of the distance travelled in the sequence.	115

List of Tables

3.1	MD-SLAM runtimes expressed in Hz.	35
3.2	ATE [m] results on TUM RGB-D datasets, recorded with non-synchronous depth using a rolling shutter camera.	36
3.3	ATE [m] on ETH3D, recorded with global shutter camera and synchronous streams. ElasticFusion fails in <i>table3</i> and <i>table7</i>	36
3.4	ATE [m] results of all benchmarked approaches on the Newer College Dataset. SuMA fails on <i>long</i> sequence.	37
4.1	The table summarizes the most important datasets in robotics perception and computer vision related to odometry estimation and SLAM. For "Accurate GT" we mean any ground truth recorded with motion capture, Laser Total Station, or globally refined.	46
4.2	Summary of the devices in our sensor setup, and the accuracy of the temporal synchronization.	48
4.3	Summary of our dataset sequences.	52
5.1	Average translation error in millimeters with different noise levels and number of measurements N	63
5.2	Quantitative results on synthetic data achieved through calibration using $N = 3$ measurements. We choose this measurement count for parity with the methodology proposed in [17]. In [17], a single measurement is deemed sufficient for calibration determination, with 3 measurements considered the optimal scenario. Beyond 3 measurements, accuracy does not improve significantly. Both for our study and in alignment with the findings in [90], 3 measurements represent the minimum requirement for solution determination, and an increase in this count is expected to result in more precise outcomes. Our results show that with our minimum number of measurements, we perform on par with [17] in rotation while outperforming all methods on translation, using small commercial tags.	64
6.1	ATE RMSE [m] on ETH3D benchmark, recorded with global shutter camera and synchronous streams. ElasticFusion fails in <i>table3</i> and <i>table7</i> , BundleFusion fails in <i>table4</i>	79
6.2	ATE RMSE [m] on Newer College Dataset, recorded with OS0-128. We always improve the SLAM baseline, a part for <i>stairs</i> starting from LeGO-LOAM estimate.	80

7.1	Evaluation of the proposed approach using different combinations of translation and angular velocities.	91
7.2	SLAM comparison.	93
8.1	Average relative pose accuracy of the proposed method and several baselines on our dataset (best RADAR-based method in bold).	107
8.2	Pose accuracy ablation study: each row corresponds to a marginal difference with DRO-GD.	109
8.3	Per-sequence errors of DRO-G on the Boeras dataset leaderboard.	111
8.4	Boreas dataset SE(2) odometry leaderboard.	111
8.5	Average error per sequence type of the MulRan dataset.	112
8.6	Average RPE in various environments spanning over different levels of structure.	113

Bibliography

- [1] Ieee standard for floating-point arithmetic. *IEEE Std 754-2019 (Revision of IEEE 754-2008)*, (2019), 1. doi:10.1109/IEEESTD.2019.8766229. 38
- [2] ABU-ALRUB, N. J. AND RAWASHDEH, N. A. Radar odometry for autonomous ground vehicles: A survey of methods and datasets. *IEEE Trans. on Intelligent Vehicles (T-IV)*, **9** (2024), 4275. doi:10.1109/TIV.2023.3340513. 97
- [3] ADOLFSSON, D., MAGNUSSON, M., ALHASHIMI, A., LILIENTHAL, A. J., AND ANDREASSON, H. Cfear radarodometry - conservative filtering for efficient and accurate radar odometry. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 5462–5469 (2021). doi:10.1109/IROS51168.2021.9636253. 97, 110
- [4] ADOLFSSON, D., MAGNUSSON, M., ALHASHIMI, A., LILIENTHAL, A. J., AND ANDREASSON, H. Lidar-level localization with radar? the cfear approach to accurate, fast, and robust large-scale radar odometry in diverse environments. *IEEE Trans. on Robotics (TRO)*, **39** (2023), 1476. doi:10.1109/TRO.2022.3221302. 96, 97, 111, 112, 116
- [5] AL-NUAIMI, A., LOPES, W., ZELLER, P., GARCEA, A., LOPES, C., AND STEINBACH, E. Analyzing lidar scan skewing and its impact on scan matching. In *2016 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–8. IEEE (2016). 87
- [6] ALDERA, R., MARTINI, D. D., GADD, M., AND NEWMAN, P. Fast radar motion estimation with a learnt focus of attention using weak supervision. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 1190–1196 (2019). doi:10.1109/ICRA.2019.8794014. 97
- [7] ALISMAIL, H., BROWNING, B., AND LUCEY, S. Photometric bundle adjustment for vision-based slam. In *Proc. of the Asian Conf. on Computer Vision (ACCV)*, pp. 324–341. Springer (2017). 71
- [8] ANDERSON, S., BARFOOT, T., TONG, C., AND SÄRKKÄ, S. Batch nonlinear continuous-time trajectory estimation as exactly sparse gaussian process regression. *Autonomous Robots*, **39** (2015), 221. doi:10.1007/s10514-015-9455-y. 98
- [9] ANDERSON, S. AND BARFOOT, T. D. Ransac for motion-distorted 3d visual sensors. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 2093–2099. IEEE (2013). 98

- [10] ANDERSON, S. AND BARFOOT, T. D. Full steam ahead: Exactly sparse gaussian process regression for batch continuous-time trajectory estimation on $se(3)$. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 157–164 (2015). doi:10.1109/IROS.2015.7353368. 98
- [11] BAI, Z., JIANG, G., AND XU, A. Lidar-camera calibration using line correspondences. *Sensors*, **20** (2020). Available from: <https://www.mdpi.com/1424-8220/20/21/6319>, doi:10.3390/s20216319. 59
- [12] BARFOOT, T. D., FORBES, J. R., AND YOON, D. J. Exactly sparse gaussian variational inference with application to derivative-free batch nonlinear state estimation. *Intl. Journal of Robotics Research (IJRR)*, **39** (2020), 1473. doi:10.1177/0278364920937608. 97
- [13] BARNES, D. AND POSNER, I. Under the radar: Learning to predict robust keypoints for odometry estimation and metric localisation in radar. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)* (2020). 96
- [14] BARNES, D., WESTON, R., AND POSNER, I. Masking by Moving: Learning Distraction-Free Radar Odometry from Pose Information. In *Conference on Robot Learning (CoRL)* (2019). Available from: <https://arxiv.org/pdf/1909.03752>. 96, 97, 98
- [15] BAY, H., TUYTELAARS, T., AND GOOL, L. V. Surf: Speeded up robust features. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pp. 404–417 (2006). 29
- [16] BEHLEY, J. AND STACHNISS, C. Efficient surfel-based slam using 3d laser range data in urban environments. In *Proc. of Robotics: Science and Systems (RSS)* (2018). 29, 34, 79
- [17] BELTRÁN, J., GUINDEL, C., DE LA ESCALERA, A., AND GARCÍA, F. Automatic extrinsic calibration method for lidar and camera sensor setups. *IEEE Trans. on Intelligent Transportation Systems (ITS)*, **23** (2022), 17677. doi:10.1109/TITS.2022.3155228. 60, 63, 64, 129
- [18] BLAKE, S. Os-cfar theory for multiple targets and nonuniform clutter. *IEEE Trans. on Aerospace and Electronic Systems (T-AES)*, **24** (1988), 785. doi:10.1109/7.18645. 97
- [19] BOSSE, M. AND ZLOT, R. Map matching and data association for large-scale two-dimensional laser scan-based slam. *Intl. Journal of Robotics Research (IJRR)*, **27** (2008), 667. doi:10.1177/0278364908091366. 98
- [20] BRADSKI, G. The opencv library. *Dr. Dobb's Journal: Software Tools for the Professional Programmer*, **25** (2000), 120. 48
- [21] BRADSKI, G. AND KAEHLER, A. *Learning OpenCV: Computer vision with the OpenCV library*. "O'Reilly Media, Inc." (2008). 57, 62
- [22] BRIZI, L., GIACOMINI, E., GIAMMARINO, L., FERRARI, S., SALEM, O., REBOTTI, L., AND GRISETTI, G. Vbr: A vision benchmark in rome. pp. 15868–15874 (2024). doi:10.1109/ICRA57147.2024.10611395. 6

- [23] BURNETT, K., SCHOELLIG, A. P., AND BARFOOT, T. D. Do we need to compensate for motion distortion and doppler effects in spinning radar navigation? *IEEE Robotics and Automation Letters (RA-L)*, **6** (2021), 771. Available from: <https://doi.org/10.1109/lra.2021.3052439>, doi:10.1109/lra.2021.3052439. 97
- [24] BURNETT, K., SCHOELLIG, A. P., AND BARFOOT, T. D. Continuous-time radar-inertial and lidar-inertial odometry using a gaussian process motion prior. *IEEE Trans. on Robotics (TRO)*, (2024). 96, 110, 111
- [25] BURNETT, K., WU, Y., YOON, D. J., SCHOELLIG, A. P., AND BARFOOT, T. D. Are We Ready for Radar to Replace Lidar in All-weather Mapping and Localization? *IEEE Robotics and Automation Letters (RA-L)*, **7** (2022), 10328. 106, 107, 113
- [26] BURNETT, K., YOON, D. J., SCHOELLIG, A. P., AND BARFOOT, T. D. Radar Odometry Combining Probabilistic Estimation and Unsupervised Feature Learning. In *Proc. of Robotics: Science and Systems (RSS)* (2021). 96, 97
- [27] BURNETT, K., ET AL. Boreas: A multi-season autonomous driving dataset. *Intl. Journal of Robotics Research (IJRR)*, **42** (2023), 33. doi:10.1177/02783649231160195. 110, 117
- [28] BURRI, M., NIKOLIC, J., GOHL, P., SCHNEIDER, T., REHDER, J., OMARI, S., ACHELNIK, M. W., AND SIEGWART, R. The euroc micro aerial vehicle datasets. *Intl. Journal of Robotics Research (IJRR)*, **35** (2016), 1157. 45, 46
- [29] CADENA, C., CARLONE, L., CARRILLO, H., LATIF, Y., SCARAMUZZA, D., NEIRA, J., REID, I., AND LEONARD, J. Past, present, and future of simultaneous localization and mapping: Towards the robust-perception age. *IEEE Trans. on Robotics (TRO)*, **32** (2016), 1309–1332. 95
- [30] CALLMER, J., TÖRNQVIST, D., GUSTAFSSON, F., SVENSSON, H., AND CARLBOM, P. Radar slam using visual features. *EURASIP Journal on Advances in Signal Processing*, **2011** (2011). doi:10.1186/1687-6180-2011-71. 97
- [31] CAMPOS, C., ELVIRA, R., RODRÍGUEZ, J. J. G., MONTIEL, J. M., AND TARDÓS, J. D. Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam. *IEEE Trans. on Robotics (TRO)*, **37** (2021), 1874. 54
- [32] CEN, S. H. AND NEWMAN, P. Precise ego-motion estimation with millimeter-wave radar under diverse and challenging conditions. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 6045–6052 (2018). doi:10.1109/ICRA.2018.8460687. 97
- [33] CEN, S. H. AND NEWMAN, P. Radar-only ego-motion estimation in difficult settings via graph matching. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 298–304 (2019). doi:10.1109/ICRA.2019.8793990. 96, 116
- [34] CHECCHIN, P., GÉROSSIER, F., BLANC, C., CHAPUIS, R., AND TRASSOUDAIN, L. Radar scan matching slam using the fourier-mellin transform. In *Springer Tracts in Advanced Robotics*, vol. 62, pp. 151–161 (2009). ISBN 978-3-642-13407-4. doi:10.1007/978-3-642-13408-1_14. 98

- [35] CHEN, Y. AND MEDIONI, G. Object modelling by registration of multiple range images. *Image and Vision Computing*, **10** (1992), 145. [80](#)
- [36] COLOSI, M., ALOISE, I., GUADAGNINO, T., SCHLEGEL, D., CORTE, B. D., ARRAS, K. O., AND GRISETTI, G. Plug-and-play slam: A unified slam architecture for modularity and ease of use. In *IROS*, pp. 5051–5057 (2020). [doi:10.1109/IROS45743.2020.9341611](#). [92](#)
- [37] COP, K. P., BORGES, P. V., AND DUBÉ, R. Delight: An efficient descriptor for global localisation using lidar intensities. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 3653–3660 (2018). [30](#)
- [38] CORTE, B. D., BOGOSLAVSKYI, I., STACHNISS, C., AND GRISETTI, G. A general framework for flexible multi-cue photometric point cloud registration. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 1–8 (2018). [27](#), [28](#), [29](#), [30](#), [32](#), [69](#), [75](#), [97](#)
- [39] DAI, A., NIESSNER, M., ZOLLHÖFER, M., IZADI, S., AND THEOBALT, C. Bundle-fusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Trans. on Graphics*, **36** (2017), 1. [72](#), [78](#)
- [40] DELAUNOY, A. AND POLLEFEYS, M. Photometric bundle adjustment for dense multi-view 3d modeling. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1486–1493 (2014). [71](#), [97](#)
- [41] DEMMEL, N., GAO, M., LAUDE, E., WU, T., AND CREMERS, D. Distributed photometric bundle adjustment. In *Proc. of the International Conference on 3D Vision (3DV)*, pp. 140–149. IEEE (2020). [71](#)
- [42] DENG, J., WU, Q., CHEN, X., XIA, S., SUN, Z., LIU, G., YU, W., AND PEI, L. Nerf-loam: Neural implicit representation for large-scale incremental lidar odometry and mapping. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)* (2023). [41](#)
- [43] DI GIAMMARINO, L. Unified multimodal 3d reconstructions: leveraging parallels between camera and lidar. [9](#), [11](#), [12](#), [21](#), [57](#), [123](#)
- [44] DI GIAMMARINO, L., GIACOMINI, E., BRIZI, L., SALEM, O., AND GRISETTI, G. Photometric lidar and rgb-d bundle adjustment. *IEEE Robotics and Automation Letters (RA-L)*, (2023). [6](#), [49](#), [50](#), [97](#)
- [45] DISSANAYAKE, M., NEWMAN, P., CLARK, S., DURRANT-WHYTE, H., AND CSORBA, M. A solution to the simultaneous localization and map building (slam) problem. *IEEE Trans. on Robotics and Automation*, **17** (2001), 229. [doi:10.1109/70.938381](#). [95](#)
- [46] DORNAIKA, F. AND HORAUD, R. Simultaneous robot-world and hand-eye calibration. *IEEE Trans. on Robotics and Automation*, **14** (1998), 617. [48](#)
- [47] ENGEL, J., KOLTUN, V., AND CREMERS, D. Direct sparse odometry. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, **PP** (2016). [doi:10.1109/TPAMI.2017.2658577](#). [97](#)

- [48] ENGEL, J., SCHÖPS, T., AND CREMERS, D. Lsd-slam: Large-scale direct monocular slam. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pp. 834–849 (2014). 29, 97
- [49] ERIKSSON, A., BASTIAN, J., CHIN, T., AND ISAKSSON, M. A consensus-based framework for distributed bundle adjustment. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 1754–1762 (2016). 71
- [50] FAN, S., YU, Y., XU, M., AND ZHAO, L. High-precision external parameter calibration method for camera and lidar based on a calibration device. **11**, 18750. Available from: <https://ieeexplore.ieee.org/document/10049434/>, doi: 10.1109/ACCESS.2023.3247195. 60
- [51] FERRARI, S., GIAMMARINO, L. D., BRIZI, L., AND GRISETTI, G. Mad-icp: It is all about matching data – robust and informed lidar odometry. *IEEE Robotics and Automation Letters*, **9** (2024), 9175. doi:10.1109/LRA.2024.3456509. 6
- [52] FINN, H. M. Adaptive detection mode with threshold control as a function of spatially sampled clutter level estimates. *Rca Rev.*, **29** (1968), 414. Available from: <https://api.semanticscholar.org/CorpusID:208092087>. 97
- [53] FISCHLER, M. A. AND BOLLES, R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, **24** (1981), 381–395. Available from: <https://doi.org/10.1145/358669.358692>, doi:10.1145/358669.358692. 59
- [54] FORSTER, C., CARLONE, L., DELLAERT, F., AND SCARAMUZZA, D. On-manifold preintegration for real-time visual–inertial odometry. *IEEE Transactions on Robotics*, **33** (2016), 1. 87
- [55] FORSTER, C., PIZZOLI, M., AND SCARAMUZZA, D. Svo: Fast semi-direct monocular visual odometry. *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, (2014), 15. Available from: <https://api.semanticscholar.org/CorpusID:206850490>. 97
- [56] FORSTER, C., ZHANG, Z., GASSNER, M., WERLBERGER, M., AND SCARAMUZZA, D. Svo: Semidirect visual odometry for monocular and multicamera systems. *IEEE Trans. on Robotics (TRO)*, **33** (2016), 249. 71
- [57] FURGALE, P., TONG, C. H., BARFOOT, T. D., AND SIBLEY, G. Continuous-time batch trajectory estimation using temporal basis functions. *Intl. Journal of Robotics Research (IJRR)*, **34** (2015), 1688. Available from: <https://doi.org/10.1177/0278364915585860>, doi:10.1177/0278364915585860. 98
- [58] GALEOTE-LUQUE, A., KUBELKA, V., MAGNUSSON, M., RUIZ-SARMIENTO, J.-R., AND GONZALEZ-JIMENEZ, J. Doppler-only single-scan 3d vehicle odometry. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 13703–13709 (2024). doi:10.1109/ICRA57147.2024.10611199. 98
- [59] GÁLVEZ-LÓPEZ, D. AND TARDOS, J. D. Bags of binary words for fast place recognition in image sequences. *IEEE Trans. on Robotics (TRO)*, **28** (2012), 1188. 29, 35

- [60] GANDHI, P. AND KASSAM, S. Analysis of cfar processors in nonhomogeneous background. *IEEE Trans. on Aerospace and Electronic Systems (T-AES)*, **24** (1988), 427. doi:10.1109/7.7185. 97
- [61] GARRIDO-JURADO, S., MUÑOZ-SALINAS, R., MADRID-CUEVAS, F., AND MARÍN-JIMÉNEZ, M. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition*, **47** (2014), 2280. Available from: <https://www.sciencedirect.com/science/article/pii/S0031320314000235>, doi:<https://doi.org/10.1016/j.patcog.2014.01.005>. 58
- [62] GEIGER, A., LENZ, P., AND URTASUN, R. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 3354–3361. IEEE (2012). 43, 46, 51, 53
- [63] GHABCHELOO, R. AND SIDDIQUI, S. Complete odometry estimation of a vehicle using single automotive radar and a gyroscope. In *Mediterranean Conference on Control and Automation (MED)*, pp. 855–860 (2018). doi:10.1109/MED.2018.8442474. 98
- [64] GIACOMINI, E., BRIZI, L., DI GIAMMARINO, L., SALEM, O., PERUGINI, P., AND GRISETTI, G. Ca2lib: Simple and accurate lidar-rgb calibration using small common markers. *IEEE Sensors Journal*, **24** (2024). Available from: <https://www.mdpi.com/1424-8220/24/3/956>, doi:10.3390/s24030956. 6, 48
- [65] GIACOMINI, E., GIAMMARINO, L. D., REBOTTI, L. D., GRISETTI, G., AND OSWALD, M. R. Splat-loam: Gaussian splatting lidar odometry and mapping (2025). Available from: <https://arxiv.org/abs/2503.17491>, arXiv: 2503.17491. 41
- [66] GIAMMARINO, L. D., ALOISE, I., STACHNISS, C., AND GRISETTI, G. Visual place recognition using lidar intensity information. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4382–4389 (2021). 27, 30, 33
- [67] GIAMMARINO, L. D., BRIZI, L., GUADAGNINO, T., STACHNISS, C., AND GRISETTI, G. Md-slam: Multi-cue direct slam. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 11047–11054. IEEE (2022). 6, 71, 72, 78, 97, 125
- [68] GOLDLÜCKE, B., AUBRY, M., KOLEV, K., AND CREMERS, D. A super-resolution framework for high-accuracy multiview reconstruction. *Intl. Journal of Computer Vision (IJCV)*, **106** (2014), 172. 71
- [69] GRAMMATIKOPOULOS, L., PAPANAGNOU, A., VENIANAKIS, A., KALISPERAKIS, I., AND STENTOUMIS, C. An effective camera-to-lidar spatiotemporal calibration based on a simple calibration target. *Sensors*, **22** (2022). Available from: <https://www.mdpi.com/1424-8220/22/15/5576>, doi:10.3390/s22155576. 59

- [70] GRIDSETH, M. AND BARFOOT, T. Towards direct localization for visual teach and repeat. In *16th Conference on Computer and Robot Vision (CRV)*, pp. 97–104 (2019). doi:[10.1109/CRV.2019.00021](https://doi.org/10.1109/CRV.2019.00021). 97
- [71] GRISETTI, G., GUADAGNINO, T., ALOISE, I., COLOSI, M., CORTE, B. D., AND SCHLEGEL, D. Least squares optimization: From theory to practice. *Robotics*, **9** (2020), 51. [24](#), [28](#), [34](#), [75](#)
- [72] GRISETTI, G., KÜMMERLE, R., STRASDAT, H., AND KONOLIGE, K. g2o: A general framework for (hyper) graph optimization. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 9–13 (2011). [29](#)
- [73] GUO, J., BORGES, P. V., PARK, C., AND GAWEL, A. Local descriptor for robust place recognition using lidar intensity. *IEEE Robotics and Automation Letters (RA-L)*, **4** (2019), 1470. [30](#)
- [74] HARLOW, K., JANG, H., BARFOOT, T. D., KIM, A., AND HECKMAN, C. A new wave in robotics: Survey on recent mmwave radar applications in robotics. *IEEE Trans. on Robotics (TRO)*, **40** (2024), 4544. doi:[10.1109/TRO.2024.3463504](https://doi.org/10.1109/TRO.2024.3463504). [95](#), [97](#)
- [75] HARRIS, M. ET AL. Optimizing parallel reduction in cuda. *Nvidia developer technology*, **2** (2007), 70. [40](#)
- [76] HE, L., JIN, Z., AND GAO, Z. De-skewing lidar scan for refinement of local mapping. *Sensors*, **20** (2020), 1846. [86](#)
- [77] HEDBORG, J., FORSSÉN, P.-E., FELSBURG, M., AND RINGABY, E. Rolling shutter bundle adjustment. *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, (2012), 1434. Available from: <https://api.semanticscholar.org/CorpusID:15591674>. [98](#)
- [78] HIGHAM, N. J. *Accuracy and Stability of Numerical Algorithms*. SIAM (2002). [33](#), [39](#)
- [79] HORN, B., HILDEN, H., AND NEGAHDARIPOUR, S. Closed-form solution of absolute orientation using orthonormal matrices. *JOSA A*, **5** (1988), 1127. [34](#), [53](#), [76](#), [92](#)
- [80] ILLINGWORTH, J. AND KITTLER, J. The adaptive hough transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **PAMI-9** (1987), 690. doi:[10.1109/TPAMI.1987.4767964](https://doi.org/10.1109/TPAMI.1987.4767964). [58](#)
- [81] IZADI, R. A. N. S., HILLIGES, O., MOLYNEAUX, D., KIM, D., DAVISON, A. J., KOHI, P., SHOTTON, J., HODGES, S., AND FITZGIBBON, A. Kinectfusion: Real-time dense surface mapping and tracking. In *Proc. of the Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 127–136 (2011). [29](#)
- [82] J. SERAFIN, E. O. AND GRISETTI, G. Fast and robust 3d feature extraction from sparse point clouds. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4105–4112 (2016). [29](#)

- [83] KAEISS, M., RANGANATHAN, A., AND DELLAERT, F. isam: Incremental smoothing and mapping. *IEEE Trans. on Robotics (TRO)*, **24** (2008), 1365. [29](#), [34](#)
- [84] KAM, J., KIM, J., KIM, S., PARK, J., AND LEE, S. Costdcnet: Cost volume based depth completion for a single rgb-d image. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pp. 257–274. Springer (2022). [57](#)
- [85] KANNALA, J. AND BRANDT, S. A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **28** (2006), 1335. [doi:10.1109/TPAMI.2006.153.66](#)
- [86] KATZ, S., TAL, A., AND BASRI, R. Direct visibility of point sets. *ACM Transactions on Graphics*, **26** (2007), 24–es. Available from: <https://doi.org/10.1145/1276377.1276407>, [doi:10.1145/1276377.1276407](#). [65](#)
- [87] KELLER, M., LEFLOCH, D., LAMBERS, M., AND IZADI, S. Realtime 3d reconstruction in dynamic scenes using pointbased fusion. In *Proc. of the International Conference on 3D Vision (3DV)*, pp. 1–8 (2013). [29](#)
- [88] KELLNER, D., BARJENBRUCH, M., KLAPPSTEIN, J., DICKMANN, J., AND DIETMAYER, K. Instantaneous ego-motion estimation using doppler radar. In *Intl. IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 869–874 (2013). [doi:10.1109/ITSC.2013.6728341](#). [98](#)
- [89] KERL, C., STURM, J., AND CREMERS, D. Dense visual slam for rgb-d cameras. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 2100–2106 (2013). [29](#), [34](#), [69](#), [78](#)
- [90] KIM, E.-S. AND PARK, S.-Y. Extrinsic calibration between camera and lidar sensors by matching multiple 3d planes. *IEEE Sensors Journal*, **20** (2020). Available from: <https://www.mdpi.com/1424-8220/20/1/52>, [doi:10.3390/s20010052](#). [60](#), [63](#), [64](#), [129](#)
- [91] KIM, G., PARK, Y. S., CHO, Y., JEONG, J., AND KIM, A. Mulran: Multimodal range dataset for urban place recognition. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 6246–6253 (2020). [43](#), [45](#), [46](#), [111](#)
- [92] KLEIN, G. AND MURRAY, D. Parallel tracking and mapping for small ar workspaces. In *6th IEEE and ACM international symposium on mixed and augmented reality*, pp. 225–234. IEEE (2007). [70](#)
- [93] KOENIG, N. AND HOWARD, A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, vol. 3, pp. 2149–2154. IEEE (2004). [64](#)
- [94] KUBELKA, V., FRITZ, E., AND MAGNUSSON, M. Do we need scan-matching in radar odometry? In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 13710–13716 (2024). [doi:10.1109/ICRA57147.2024.10610666](#). [98](#)

- [95] KUNG, P.-C., WANG, C., AND LIN, W.-C. A normal distribution transform-based radar odometry designed for scanning and automotive radars. *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, (2021), 14417. Available from: <https://api.semanticscholar.org/CorpusID:232233173>. 97
- [96] LAMARCA, J., PARASHAR, S., BARTOLI, A., AND MONTIEL, J. M. M. DefSLAM: Tracking and mapping of deforming scenes from monocular sequences. *IEEE Transactions on Robotics*, **37** (2021), 291. 121
- [97] LE GENTIL, C., BRIZI, L., LISUS, D., QIAO, X., GRISETTI, G., AND BARFOOT, T. Dro: Doppler-aware direct radar odometry (2025). doi:10.48550/arXiv.2504.20339. 6
- [98] LE GENTIL, C., FALQUE, R., AND VIDAL-CALLEJA, T. Real-time truly-coupled lidar-inertial motion correction and spatiotemporal dynamic object detection. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 12565–12572 (2024). doi:10.1109/IROS58592.2024.10802233. 102
- [99] LE GENTIL, C. AND VIDAL-CALLEJA, T. Continuous latent state preintegration for inertial-aided systems. *Intl. Journal of Robotics Research (IJRR)*, **42** (2023), 874. Available from: <https://doi.org/10.1177/02783649231199537>, arXiv:<https://doi.org/10.1177/02783649231199537>, doi:10.1177/02783649231199537. 102
- [100] LE GENTIL, C., VIDAL-CALLEJA, T., AND HUANG, S. Gaussian process preintegration for inertial-aided state estimation. *IEEE Robotics and Automation Letters (RA-L)*, **5** (2020), 2108. doi:10.1109/LRA.2020.2970940. 102
- [101] LI, S., CHEN, Y., HONG, Z., AND HU, L. Cfear++: An efficient radar-inertial semantic odometry. *ICRA Workshop, Radar in Robotics: Resilience from Signal to Navigation*, (2024). 96, 110, 111, 116
- [102] LI, X., HE, F., LI, S., ZHOU, Y., XIA, C., AND WANG, X. Accurate and automatic extrinsic calibration for a monocular camera and heterogenous 3d lidars. *IEEE Sensors Journal*, **22** (2022), 16472. doi:10.1109/JSEN.2022.3189041. 60
- [103] LIAO, Y., XIE, J., AND GEIGER, A. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *Pattern Analysis and Machine Intelligence (PAMI)*, (2022). 46
- [104] LIM, H., HAN, K., SHIN, G., KIM, G., HONG, S., AND MYUNG, H. ORORA: Outlier-robust Radar Odometry. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 2046–2053 (2023). doi:10.1109/ICRA48891.2023.10160997. 96
- [105] LISUS, D., BURNETT, K., YOON, D. J., POULTON, R., MARSHALL, J., AND BARFOOT, T. D. Are doppler velocity measurements useful for spinning radar odometry? *IEEE Robotics and Automation Letters (RA-L)*, **10** (2025), 224. doi:10.1109/LRA.2024.3505821. 98, 103, 105, 106, 107

- [106] LISUS, D., LACONTE, J., BURNETT, K., AND BARFOOT, T. D. Pointing the way: Refining radar-lidar localization using learned icp weights. *ArXiv*, **abs/2309.08731** (2023). Available from: <https://api.semanticscholar.org/CorpusID:262043844>. 97
- [107] LIU, T. Y., AGRAWAL, P., CHEN, A., HONG, B.-W., AND WONG, A. Monitored distillation for positive congruent depth completion (2022). *arXiv:2203.16034*. 57
- [108] LIU, Z. AND ZHANG, F. Balm: Bundle adjustment for lidar mapping. *IEEE Robotics and Automation Letters (RA-L)*, **6** (2021), 3184. 72, 79
- [109] LOWE, D. G. Distinctive Image Features from Scale-Invariant Keypoints. *Intl. Journal of Computer Vision (IJCV)*, **60** (2004), 91. 97
- [110] LV, X., WANG, B., DOU, Z., YE, D., AND WANG, S. Lccnet: Lidar and camera self-calibration using cost volume network. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 2894–2901 (2021). 59
- [111] MACHADO FERNÁNDEZ, J., MOJENA-HERNÁNDEZ, N., AND BACALLAO VIDAL, J. Evaluation of cfar detectors performance. *ITECKNE Innovación e Investigación en Ingeniería*, **14** (2017), 170. doi:10.15332/iteckne.v14i2.1772. 97
- [112] MADDERN, W., PASCOE, G., LINEGAR, C., AND NEWMAN, P. 1 year, 1000 km: The oxford robotcar dataset. *Intl. Journal of Robotics Research (IJRR)*, **36** (2017), 3. 43, 45, 46
- [113] MAGNUSSON, M., ANDREASSON, H., NÜCHTER, A., AND LILIENTHAL, A. J. Automatic appearance-based loop detection from three-dimensional laser data using the normal distributions transform. *Journal of Field Robotics (JFR)*, **26** (2009), 892. 29
- [114] MATSUKI, H., BAE, G., AND DAVISON, A. 4dtam: Non-rigid tracking and mapping via dynamic surface gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2025). 121
- [115] MATSUKI, H., MURAI, R., KELLY, P. H. J., AND DAVISON, A. J. Gaussian splatting slam. (2024). 41
- [116] MIRZAEI, F. M., KOTTAS, D. G., AND ROUMELIOTIS, S. I. 3d lidar-camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization. **31**, 452. Available from: <http://journals.sagepub.com/doi/10.1177/0278364911435689>, doi:10.1177/0278364911435689. 60
- [117] MOOSMANN, F. AND STILLER, C. Velodyne slam. In *2011 IEEE intelligent vehicles symposium (iv)*, pp. 393–398. IEEE (2011). 87
- [118] MUR-ARTAL, R. AND TARDÓS, J. D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. on Robotics (TRO)*, **33** (2017), 1255. 29, 34, 70, 78

- [119] NÜCHTER, A., LINGEMANN, K., HERTZBERG, J., AND SURMANN, H. Heuristic-based laser scan matching for outdoor 6d slam. In *Annual Conf. on Artificial Intelligence*, pp. 304–319 (2005). 29
- [120] OLSON, E. Apriltag: A robust and flexible visual fiducial system. In *2011 IEEE International Conference on Robotics and Automation*, pp. 3400–3407 (2011). doi:10.1109/ICRA.2011.5979561. 59
- [121] PANDEY, G., MCBRIDE, J. R., SAVARESE, S., AND EUSTICE, R. M. Automatic extrinsic calibration of vision and lidar by maximizing mutual information. **32** (2015), 696. 59
- [122] PARK, J., JOO, K., HU, Z., LIU, C.-K., AND KWEON, I. S. Non-local spatial propagation network for depth completion. In *Proc. of European Conference on Computer Vision (ECCV)* (2020). 57
- [123] PARK, Y., YUN, S., WON, C. S., CHO, K., UM, K., AND SIM, S. Calibration between color camera and 3d lidar instruments with a polygonal planar board. *IEEE Sensors Journal*, **14** (2014), 5333. Available from: <https://www.mdpi.com/1424-8220/14/3/5333>. 59
- [124] PARK, Y. S., SHIN, Y.-S., AND KIM, A. Pharao: Direct radar odometry using phase correlation. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 2617–2623 (2020). doi:10.1109/ICRA40945.2020.9197231. 96, 98
- [125] PRESTON-KREBS, E., LISUS, D., AND BARFOOT, T. The finer points: A systematic comparison of point-cloud extractors for radar odometry. *arXiv preprint arXiv:2409.10491*, (2024). doi:10.48550/arXiv.2409.12256. 97
- [126] PUSZTAI, Z., EICHHARDT, I., AND HAJDER, L. Accurate calibration of multi-LiDAR-multi-camera systems. **18** (2018), 2139. Available from: <http://www.mdpi.com/1424-8220/18/7/2139>, doi:10.3390/s18072139. 59
- [127] PUSZTAI, Z. AND HAJDER, L. Accurate calibration of lidar-camera systems using ordinary boxes. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, pp. 394–402 (2017). doi:10.1109/ICCVW.2017.53. 59
- [128] QIAO, X., KRAWCIW, A., LILGE, S., AND BARFOOT, T. D. Radar teach and repeat: Architecture and initial field testing. *arXiv preprint arXiv:2409.10491*, (2024). 96, 113
- [129] RAMEZANI, M., WANG, Y., CAMURRI, M., WISTH, D., MATTAMALA, M., AND FALLON, M. The newer college dataset: Handheld lidar, inertial and vision with ground truth. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4353–4360 (2020). doi:10.1109/IROS45743.2020.9340849. 36, 37, 43, 46, 49, 124
- [130] RASMUSSEN, C. E. AND WILLIAMS, C. K. I. *Gaussian Processes for Machine Learning*. The MIT Press (2006). 17
- [131] RASSHOFFER, H. R., SPIES, M., AND SPIES, H. Influences of weather phenomena on automotive laser radar systems. *Advances in Radio Science*, **9** (2011), 49. 10

- [132] RENNIE, F., WILLIAMS, D., NEWMAN, P., AND DE MARTINI, D. Doppler-Aware Odometry from FMCW Scanning Radar. In *Intl. IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 5126–5132 (2023). 98
- [133] ROHLING, H. Radar cfar thresholding in clutter and multiple target situations. *IEEE Trans. on Aerospace and Electronic Systems (T-AES)*, **AES-19** (1983), 608. doi:10.1109/TAES.1983.309350. 97
- [134] RÖHLING, T., MACK, J., AND SCHULZ, D. A fast histogram-based similarity measure for detecting loop closures in 3-d lidar data. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 736–741 (2015). 29
- [135] ROSINOL, A., LEONARD, J. J., AND CARLONE, L. Nerf-slam: Real-time dense monocular slam with neural radiance fields. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 3437–3444 (2023). doi:10.1109/IROS55552.2023.10341922. 41
- [136] RUBLEE, E., RABAU, V., KONOLIGE, K., AND BRADSKI, G. Orb: An efficient alternative to sift or surf. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, pp. 2564–2571 (2011). 28, 29, 78, 97
- [137] SALEM, O. A. A. K., GIACOMINI, E., BRIZI, L., DI GIAMMARINO, L., AND GRISETTI, G. Enhancing lidar performance: Robust de-skewing exclusively relying on range measurements. In *AIxIA 2023 - Advances in Artificial Intelligence*. Springer-Verlag, Berlin, Heidelberg (2023). ISBN 978-3-031-47545-0. Available from: https://doi.org/10.1007/978-3-031-47546-7_21, doi:10.1007/978-3-031-47546-7_21. 6, 98
- [138] SCHLEGEL, D. AND GRISETTI, G. Hbst: A hamming distance embedding binary search tree for feature-based visual place recognition. *IEEE Robotics and Automation Letters (RA-L)*, **3** (2018), 3741. 28, 33
- [139] SCHOPS, T., SATTler, T., AND POLLEFEYS, M. Bad slam: Bundle adjusted direct rgbd-d slam. In *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 134–144 (2019). 27, 29, 35, 43, 46, 69, 70, 72, 78
- [140] SEGAL, A., HAEHNEL, D., AND THRUN, S. Generalized-icp. In *Proc of Robotics: Science and Systems* (2009). 29
- [141] SHAN, T. AND ENGLot, B. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4758–4765 (2018). 29, 34, 72, 79
- [142] SHAN, T., ENGLot, B., MEYERS, D., WANG, W., RATTI, C., AND RUS, D. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *2020 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 5135–5142. IEEE (2020). 87
- [143] SINGANDHUPE, A., LA, H. M., AND HA, Q. P. Single frame lidar-camera calibration using registration of 3d planes. pp. 395–402. IEEE. ISBN 978-1-66547-260-9. Available from: <https://ieeexplore.ieee.org/document/10023616/>, doi:10.1109/IRC55401.2022.00076. 60

- [144] SLAVCHEVA, M., KEHL, W., NAVAB, N., AND ILIC, S. Sdf-2-sdf: Highly accurate 3d object reconstruction. In *Proc. of the Europ. Conf. on Computer Vision (ECCV)*, pp. 680–696. Springer (2016). 71
- [145] SMITH, M. AND VARSHNEY, P. K. Vi-cfar: a novel cfar algorithm based on data variability. *Proceedings of the 1997 IEEE National Radar Conference*, (1997), 263. Available from: <https://api.semanticscholar.org/CorpusID:140690935>. 97
- [146] SOLÀ, J., DERAY, J., AND ATCHUTHAN, D. A micro lie theory for state estimation in robotics (2021). Available from: <https://arxiv.org/abs/1812.01537>, [arXiv:1812.01537](#). 23, 24, 123
- [147] STOYANOV, T., MAGNUSSON, M., ANDREASSON, H., AND LILIENTHAL, A. J. Fast and accurate scan registration through minimization of the distance between compact 3d ndt representations. *Intl. Journal of Robotics Research (IJRR)*, **31** (2012), 1377. 29, 72
- [148] STRÖM, D. P., BOGOSLAVSKYI, I., AND STACHNISS, C. Robust exploration and homing for autonomous robots. *Journal on Robotics and Autonomous Systems (RAS)*, (2016). 29
- [149] STURM, J., ENGELHARD, N., ENDRES, F., BURGARD, W., AND CREMERS, D. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 573–580 (2012). 35
- [150] SUN, C., WEI, Z., HUANG, W., LIU, Q., AND WANG, B. Automatic targetless calibration for lidar and camera based on instance segmentation. *IEEE Robotics and Automation Letters (RA-L)*, **8** (2022), 981. 59
- [151] TANG, J., CHEN, Y., NIU, X., WANG, L., CHEN, L., LIU, J., SHI, C., AND HYYPPÄ, J. Lidar scan matching aided inertial navigation system in gnss-denied environments. *Sensors*, **15** (2015), 16710. 86
- [152] TRAHANIAS, P., BURGARD, W., ARGYROS, A., HÄHNEL, D., BALTZAKIS, H., PFAFF, P., AND STACHNISS, C. TOURBOT and WebFAIR: Web-operated mobile robots for tele-presence in populated exhibitions. *IEEE Robotics and Automation Magazine (RAM)*, **12** (2005), 77. 29
- [153] TRIGGS, B., MCCLAUCHLAN, P. F., HARTLEY, R. I., AND FITZGIBBON, A. W. Bundle adjustment - a modern synthesis. In *International Workshop on Vision Algorithms*, pp. 298–372. Springer (2000). 69
- [154] TÓTH, T., PUSZTAI, Z., AND HAJDER, L. Automatic lidar-camera calibration of extrinsic parameters using a spherical target. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 8580–8586 (2020). [doi:10.1109/ICRA40945.2020.9197316](https://doi.org/10.1109/ICRA40945.2020.9197316). 60
- [155] VELAS, M., SPANEL, M., AND HEROUT, A. Collar line segments for fast odometry estimation from velodyne point clouds. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 4486–4495 (2016). 29, 72

- [156] VENON, A., DUPUIS, Y., VASSEUR, P., AND MERRIAUX, P. Millimeter wave fmcw radars for perception, recognition and localization in automotive applications: A survey. *IEEE Trans. on Intelligent Vehicles (T-IV)*, **7** (2022), 1. doi:10.1109/TIV.2022.3167733. 97
- [157] VIZZO, I., GUADAGNINO, T., MERSCH, B., WIESMANN, L., BEHLEY, J., AND STACHNISS, C. Kiss-icp: In defense of point-to-point icp—simple, accurate, and robust registration if done the right way. *IEEE Robotics and Automation Letters (RA-L)*, **8** (2023), 1029. 54
- [158] WANG, H., WANG, C., CHEN, C.-L., AND XIE, L. F-loam : Fast lidar odometry and mapping. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 4390–4396 (2021). 54
- [159] WANG, Z., ZHANG, L., SHEN, Y., AND ZHOU, Y. D-liom: Tightly-coupled direct lidar-inertial odometry and mapping. *IEEE Transactions on Multimedia*, (2022). 87
- [160] WESTON, R., GADD, M., DE MARTINI, D., NEWMAN, P., AND POSNER, I. Fast-mbym: Leveraging translational invariance of the fourier transform for efficient and accurate radar odometry. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 2186–2192 (2022). doi:10.1109/ICRA46639.2022.9812063. 96, 98
- [161] WHELAN, T., LEUTENEGGER, S., SALAS-MORENO, R., GLOCKER, B., AND DAVISON, A. Elasticfusion: Dense slam without a pose graph. In *Proc. of Robotics: Science and Systems* (2015). 34, 78
- [162] WILLIAMS, B., I., G. K., AND REID. Real-time slam relocalisation. In *Proc. of the IEEE Intl. Conf. on Computer Vision (ICCV)*, pp. 1–8 (2007). 29
- [163] WONG, A. AND SOATTO, S. Unsupervised depth completion with calibrated back-projection layers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12747–12756 (2021). 57
- [164] XU, W. AND ZHANG, F. Fast-lio: A fast, robust lidar-inertial odometry package by tightly-coupled iterated kalman filter. *IEEE Robotics and Automation Letters*, **6** (2021), 3317. 87
- [165] YE, H., CHEN, Y., AND LIU, M. Tightly coupled 3d lidar inertial odometry and mapping. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 3144–3150. IEEE (2019). 87
- [166] YOON, B.-H., JEONG, H.-W., AND CHOI, K.-S. Targetless multiple camera-lidar extrinsic calibration using object pose estimation. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 13377–13383. IEEE (2021). 59
- [167] ZHANG, J., KAESSE, M., AND SINGH, S. On degeneracy of optimization-based state estimation problems. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 809–816 (2016). 72
- [168] ZHANG, J. AND SINGH, S. Loam: Lidar odometry and mapping in real-time. In *Proc. of Robotics: Science and Systems (RSS)* (2014). 29, 72

- [169] ZHANG, J. AND SINGH, S. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, pp. 2174–2181 (2015). [29](#), [72](#)
- [170] ZHANG, L., CAMURRI, M., AND FALLON, M. Multi-camera lidar inertial extension to the newer college dataset. *arXiv*, 2112.08854 (2021). [arXiv:2112.08854](#). [28](#), [36](#), [78](#), [79](#), [83](#), [124](#), [126](#)
- [171] ZHANG, L., HELMBERGER, M., FU, L. F. T., WISTH, D., CAMURRI, M., SCARAMUZZA, D., AND FALLON, M. Hilti-oxford dataset: A millimeter-accurate benchmark for simultaneous localization and mapping. *IEEE Robotics and Automation Letters (RA-L)*, **8** (2022), 408. [43](#), [46](#), [49](#)
- [172] ZHANG, R., SHENG, W., AND MA, X. Improved switching cfar detector for non-homogeneous environments. *Signal Processing*, (2013), 35–48. [doi:10.1016/j.sigpro.2012.06.015](#). [97](#)
- [173] ZHANG, Z. AND SCARAMUZZA, D. A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry. In *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)* (2018). [106](#)
- [174] ZHAO, L., LIU, W., WU, X., AND FU, J. S. A novel approach for cfar processors design. In *Proceedings of the 2001 IEEE Radar Conference*, pp. 284–288. IEEE (2001). [97](#)
- [175] ZHOU, L., LI, Z., AND KAESE, M. Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences. pp. 5562–5569 (2018). [doi:10.1109/IROS.2018.8593660](#). [59](#)