Multisensor Fusion for State Estimation via Factor Graphs

David Wisth

St Hugh's College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy (Engineering Science)

2021

Abstract

State estimation is a fundamental requirement for mobile robots. An accurate and robust estimate of a platform's location and environment is a key input for many other robot systems, including control, navigation, and task planning.

The aim of this thesis is to develop a state estimation system based on probabilistic factor graph optimisation. The proposed algorithm is called VILENS (Visual Inertial Lidar Legged Navigation System). It uses different combinations of camera, lidar, inertial, and kinematic sensors to estimate the robot's state, and is designed to be robust to individual sensor failure.

This thesis presents several novel contributions. First is a new method of integrating kinematic measurements into a factor graph. The proposed preintegrated twist factor aims to reduce the drift caused by slippage or deformation by estimating kinematics biases online. These effects are not modelled in other works.

Second, a novel method for tracking and optimising sparse geometric features from lidar scans is presented. This differs from the literature which focuses on scan-to-scan or scan-to-map matching.

Third, the complete VILENS system, which is the first attempt at integrating visual, inertial, lidar, and kinematic measurements into a factor graph. This makes use of a novel camera-lidar synchronisation method to reduce the graph size and complexity. A key benefit of this tight fusion is the natural handling of degenerate scenarios without hard switching.

This algorithm has been widely tested on a variety of platforms, including legged robots and handheld sensor rigs. This thesis includes one of the first extensive real-world demonstrations of factor graphs for legged robot state estimation and has achieved relative position errors as low as 0.4 % per distance travelled in challenging underground environments.

Several large-scale deployments of mobile robots have either motivated its development or directly incorporated VILENS for state estimation. These include sewer inspection, 3D building reconstruction, and the DARPA Subterranean Challenge.

Multisensor Fusion for State Estimation via Factor Graphs



David Wisth

St Hugh's College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy (Engineering Science)

2021

Declaration

This thesis is submitted to the Department of Engineering Science, University of Oxford, in fulfilment of the requirements for the degree of Doctor of Philosophy. I declare that this thesis is entirely my own work, and except where otherwise stated, describes my own research.



David Wisth, St Hugh's College

Acknowledgements

First and foremost, I would like to thank my supervisor, Professor Maurice Fallon, for helping me to grow into the researcher and person I am today. His support, advice, and leadership have been instrumental in my DPhil. It has been an incredible experience seeing the Dynamic Robot Systems group develop into a world-leading research lab in just a few short years.

To Dr Marco Camurri, you have been the catalyst for this work. Your insightful questions, pertinent advice, programming knowledge, and late nights debugging and writing papers with me have provided the extra boost I often needed to get things over the line.

Thank you to the amazing group at the Oxford Robotics Institute and my co-supervisor Dr Ioannis Havoutis who have been a constant source of inspiration and mateship throughout my DPhil.

I would like to acknowledge my collaborators from around the world. In particular, thank you to Professor Marco Hutter and the Robotics System Lab at ETH Zürich who welcomed us with open arms into their community and their DARPA Subterranean Challenge team. Thank you to Dr Hendrik Kolvenbach who invited us to help bring legged robots to the sewers of Zürich.

Thank you to my examiners, Professor Jonathan Kelly and Professor Andrea Vedaldi, for taking time out of their busy schedules to provide valuable feedback and discussion on my research.

I gratefully acknowledge the generous backing of the University of Oxford, the Royal Society of Engineering, and Google DeepMind which made this possible.

Finally, thank you to my family for providing unconditional love and support throughout this entire process. Most importantly, I owe my deepest gratitude to my wife, Georgia Rose. She moved to the UK with me for my DPhil and has supported me every step of the way. Georgia Rose, thank you for being my champion.

Contents

List of Abbreviations **vii**

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Objective . . . 2
1.3 Contributions . . . 4
1.4 Publications and Highlights . . . 4
1.4.1 List of First-Authored Publications . . . 5
1.4.2 List of Co-Authored Publications . . . 5
1.5 Thesis Roadmap . . . 6

2 Background **8**
2.1 Terminology . . . 8
2.2 Hardware and Sensor Description . . . 9
2.3 State Estimation Algorithms . . . 15
2.3.1 Sensor Fusion Techniques . . . 16
2.3.2 Optimisation for State Estimation . . . 19
2.3.3 Performance Metrics . . . 20

3 Literature Review **22**
3.1 Exteroceptive State Estimation . . . 22
3.1.1 Vision . . . 23
3.1.2 Lidar . . . 28
3.2 Proprioceptive State Estimation . . . 32
3.2.1 Kinematics . . . 33
3.2.2 Modelling Contact Deformation and Slippage . . . 35
3.3 Multi-Sensor Fusion . . . 37
3.3.1 Vision, IMU, and Lidar . . . 37
3.3.2 Legged Robots . . . 41
3.4 Factor Graph Optimisation . . . 44
3.5 Conclusion . . . 45

4 Visual, Inertial, Kinematic State Estimation **46**
4.1 Robust Legged Robot State Estimation Using Factor Graph Optimization . . . 47
4.2 Preintegrated Velocity Bias Estimation to Overcome Contact Nonlinearities in Legged Robot Odometry . . . 57
4.3 Discussion . . . 67

5 Incorporating Lidar Information **68**
5.1 Unified Multi-Modal Landmark Tracking for Tightly Coupled Lidar-Visual-Inertial Odometry . . . 69
5.2 Extensions . . . 79
5.2.1 Point Cloud Feature Tracking . . . 79
5.2.2 Improving Plane and Line Factors . . . 81
5.3 Discussion . . . 84

6 Visual, Inertial, Lidar, Kinematic State Estimation **86**
6.1 VILENS: Visual, Inertial, Lidar, and Leg Odometry for All-Terrain Legged Robots . . . 86
6.2 Discussion . . . 106

7 Real-World Algorithm Deployment **107**
7.1 Software Architecture . . . 107
7.2 Ground Truth Generation using VILENS . . . 108
7.3 Practical Deployments . . . 110
7.3.1 Underground Sewer Inspection with Legged Robots . . . 110
7.3.2 Outdoor Mapping using a Handheld Sensor Rig . . . 111
7.3.3 DARPA Subterranean Challenge . . . 113
7.3.4 Multi-Camera Visual Inertial Odometry . . . 114
7.3.5 Integration with Lidar SLAM . . . 115
7.4 Discussion . . . 116

8 Conclusion and Future Work **117**
8.1 Future Work . . . 117
8.1.1 Automate Configuration for New Platforms . . . 117
8.1.2 Extend Lidar Odometry . . . 118
8.1.3 Exploit Correlation Between Sensors . . . 118
8.2 Conclusion . . . 119

A Derivation of Preintegrated Twist Factors **120**
A.1 Measurement Model 120
A.2 Preintegrated Measurement and Noise 122
A.2.1 Iterative Noise Model Update 123
A.3 Updating Under Small Bias Changes 125
A.4 Residual and Jacobians 126
A.5 Summary 127

B Additional Articles **128**
B.1 Towards Autonomous Inspection of Concrete Deterioration in Sewers with Legged Robots 128
B.2 The Newer College Dataset: Handheld LiDAR, Inertial and Vision with Ground Truth 148
B.3 CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge 158
B.4 Balancing the Budget: Feature Selection and Tracking for Multi-Camera Visual-Inertial Odometry 210

Bibliography **220**

List of Abbreviations

DOF Degrees of Freedom.

EKF Extended Kalman Filter.

GTSAM Georgia Tech Smoothing and Mapping [1].

ICP Iterative Closest Point.

IMU Inertial Measurement Unit.

iSAM2 Incremental Smoothing And Mapping 2 [2].

LOAM Lidar Odometry and Mapping [3].

RPE Relative Pose Error.

SLAM Simultaneous Localisation And Mapping.

TSIF Two State Implicit Filter [4].

VILENS Visual Inertial Lidar Legged Navigation System.

VIO Visual Inertial Odometry.

VO Visual Odometry.

Chapter 1

Introduction

1.1 Motivation

A key requirement for almost all mobile platforms, including robots, autonomous vehicles, and even virtual reality devices is *state estimation*. State estimation involves processing measurements from one or more sensors to estimate properties of the system and its surroundings – in robotics, this is often a platform's position, orientation, and velocity. This is also known as *motion tracking* or *odometry*. The *state estimate* is a key input to many other subsystems including control, navigation, and task planning.

The main challenge is producing accurate and robust estimates while dealing with measurement uncertainty across a wide range of environments and scenarios. All sensor measurements contain noise, which can be affected by many factors, including calibration, temperature, manufacturing defects, and quality of sensor construction. Additionally, sensors can fail, or their output can be degraded by environmental conditions such as darkness for cameras, or heavy fog for lidars [5].

Specifically, this thesis aims to estimate the 3D position, orientation, and velocity of mobile platforms, with particular emphasis on legged robots and handheld sensor rigs. Both platforms are challenging for state estimation algorithms due to their highly dynamic motion, wide range of operating conditions, and limited on-board computation.

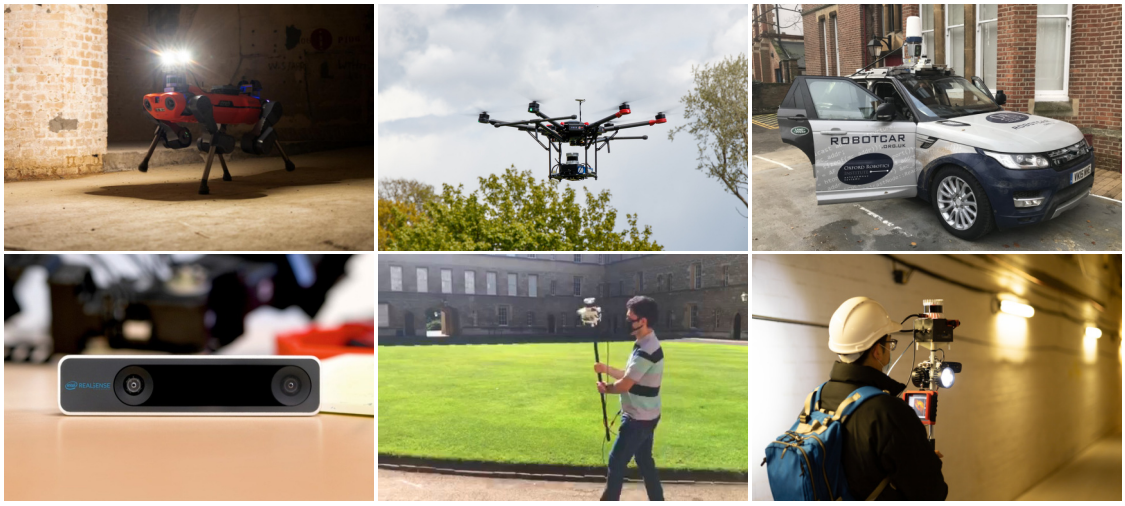


Figure 1.1: Examples of experimental platforms at the Oxford Robotics Institute used to evaluate the VILENS algorithm. *Top-Left:* ANYmal C quadruped robot [6], *Top-Middle:* Drone with sensor payload, *Top-Right:* Wheeled vehicles, *Bottom-Left:* Handheld Realsense visual-inertial sensor, *Bottom-Middle:* The "Rooster" sensor rig with lidar, camera, and IMU [7], *Bottom-Right:* The "HALO" sensor rig with multiple fisheye cameras, lidar, and IMU [8].

Legged robots also have strict requirements for state estimation to allow the robot to quickly react to the environment or regain balance. Typically, legged robots rely on both a smooth, low latency (<2.5 ms), and high frequency (100–1000 Hz) estimate for control, and a lower frequency (10–30 Hz), but higher accuracy estimate for other tasks, such as motion planning.

1.2 Objective

Currently, state-of-the-art multi-sensor fusion algorithms often rely on loosely-coupled methods (e.g., Extended Kalman Filters (EKF)) which do not usually exploit correlations between measurements, limiting their accuracy. Additionally, it is common for these approaches to run multiple independent odometry algorithms at once then perform "hard-switching" between these odometry sources based on heuristics [9–11]. In practice, this approach has significant downfalls such as increased computation, lower accuracy, and ignoring partial constraints from sensors. This thesis instead focuses on *tightly-coupled* methods, where sensor measurements are jointly optimized and correlations between sensor measurements are exploited



Figure 1.2: The VILENS algorithm logo.

(e.g., lidar measurements can assist with triangulation of visual features). This can make the estimation more robust and accurate, especially in challenging scenarios.

Mobile robots, including the ANYmal quadruped [6] shown in Figure 1.1, perform motion planning and obstacle avoidance based on 3D maps of their environment. However, drift in the state estimate can cause poor quality terrain reconstruction. For example, the default kinematic-inertial state estimator on the ANYmal robot [4] suffers from significant drift due to the uncertainty of foot contact with the ground. This can lead to highly distorted terrain maps which are a limiting factor in locomotion over rough terrain, where precise footstep execution is required.

Instead, this thesis addresses the state estimation problem by combining heterogeneous sensor measurements using Bayesian factor graph optimisation. These sensing modalities include visual, inertial, kinematic, and lidar sensors, which all vary greatly in accuracy, frequency, latency, and type of information. The variety of sensor measurements provides accuracy and robustness in a wide range of conditions. Measurements are tightly fused within a single factor graph for robustness, accuracy, and natural handling of degenerate scenarios.

The aim of this thesis is to create an odometry system that can estimate the robot's state (including position, orientation, and velocity) in real time with high accuracy. This algorithm should work with a variety of platforms and sensor configurations, as well as be able to run on-board computationally-limited platforms.

The proposed algorithm is titled *Visual Inertial Lidar Legged Navigation System (VILENS)*. It has been evaluated on a wide variety of platforms including quadruped robots, wheeled vehicles, drones, and handheld sensor rigs, as shown in Figure 1.1. The logo for the VILENS algorithm is shown in Figure 1.2.

1.3 Contributions

The main contribution of this thesis is the development of a robust factor graph-based state estimation system called VILENS. This algorithm is suitable for, and has been deployed on, a wide variety of platforms, with a particular emphasis on state estimation for legged robots.
Specifically, the contributions are:

- The first algorithm to tightly fuse visual, inertial, lidar, and leg kinematic measurements in a factor graph framework.
- Development of novel methods to efficiently and accurately incorporate leg kinematics in a factor graph, including a new preintegrated twist factor to account for contact non-linearities.
- Novel lidar landmark tracking and optimisation methods to allow visual and lidar landmarks to be tightly fused in a single factor graph. This formulation is robust to situations where other algorithms may fail.
- The first work to provide extensive experimental validation of a factor graph-based state estimation method for legged robots.
- Demonstration and analysis of VILENS across a wide variety of platforms and conditions, showing its wide applicability and robust performance.

1.4 Publications and Highlights

Shown below is a summary of the peer-reviewed publications submitted during this degree, including four first-author and four non-first-author peer-reviewed articles. It should be noted that TRO-2021 is an evolved version of the ICRA-2020 paper with significant additional contributions.

1.4.1 List of First-Authored Publications

RAL-2019 [12]: D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robotics and Automation Letters*, pp. 4507–4514, 2019.

ICRA-2020 [13]: D. Wisth, M. Camurri, and M. Fallon, "Preintegrated velocity bias estimation to overcome contact nonlinearities in legged robot odometry," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 392–398.

RAL-2021 [14]: D. Wisth, M. Camurri, S. Das, and M. Fallon, "Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1004–1011, 2021. **(Finalist, Best Student Paper Award, IEEE International Conference on Robotics and Automation (ICRA), 2021)**

TRO-2021 [15]: D. Wisth, M. Camurri, and M. Fallon, "VILENS: Visual, inertial, lidar, and leg odometry for all-terrain legged robots," *IEEE Transactions on Robotics*, 2021 (submitted).

1.4.2 List of Co-Authored Publications

IROS-2020 [7]: M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The Newer College Dataset: Handheld lidar, inertial and vision with ground truth," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

JFR-2020 [16]: H. Kolvenbach, D. Wisth, R. Buchanan, G. Valsecchi, R. Grandia, M. Fallon, and M. Hutter, "Towards autonomous inspection of concrete deterioration in sewers with legged robots," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1314–1327, 2020.

FR-2022 [17]: M. Tranzatto, F. Mascarich, L. Bernreiter, C. Godinho, M. Camurri, S. Khattak, T. Dang, V. Reijgwart, J. Löje, D. Wisth et al., "CERBERUS: Autonomous legged and aerial robotic exploration in the tunnel and Urban Circuits of the DARPA Subterranean Challenge," *Field Robotics*, vol. 2, pp. 274-324, 2022. **(CERBERUS team, Winners, Systems Track, DARPA Subterranean**

Ch.	Publication	IMU	Camera	Kinematics	Lidar
4	**RAL-2019**	✓	✓	✓	
4	**ICRA-2020**	✓	✓	✓(improved)	
5	**RAL-2021**	✓	✓(lidar depth)		✓(features)
6	**TRO-2021**	✓	✓(fisheye)	✓(integrated)	✓(features, ICP)
7	**RAL-2022**	✓	✓(multiple)		

Table 1.1: Thesis roadmap: Sensor modalities used for each publication.

Challenge Finals, 2021)

RAL-2022 [8]: L. Zhang, D. Wisth, M. Camurri, and M. Fallon, "Balancing the budget: Feature selection and tracking for multi-camera visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1182-1189, 2022.

1.5 Thesis Roadmap

This thesis will be presented in an "integrated thesis" style, consisting of peer-reviewed publications, alongside a more in-depth discussion of the work. Each article will be accompanied by a Statement of Authorship declaration, highlighting the author's contribution to each work. A high level summary of the modalities used in each paper is shown in Table 1.1.

The remainder of this thesis consists of the following chapters:

- **Chapter 2 – Background:** Overview of fundamentals for this thesis, including terminology, hardware description, and prerequisite material.
- **Chapter 3 – Literature Review:** General overview of relevant literature for this thesis. Specific literature reviews will also be presented in each chapter.
- **Chapter 4 – Visual, Inertial, Kinematic State Estimation:** Introduction to the VILENS factor graph framework and novel methods for integrating kinematic measurements into the factor graph. This chapter specifically addresses non-ideal contact events (e.g., slippage and deformation).

- **Chapter 5 – Incorporating Lidar Information:** Method of tightly fusing lidar, visual, and inertial measurements in a factor graph, enabling real-time performance and robust handling of degenerate scenarios.
- **Chapter 6 – Visual, Inertial, Lidar, Kinematic State Estimation:** This chapter introduces the complete VILENS framework for tightly coupling visual, lidar, kinematic, and inertial measurements. There is particular emphasis on integrating kinematic measurements directly into the factor graph.
- **Chapter 7 – Real-world algorithm deployment:** Implementation and performance evaluation of VILENS on a wide variety of real experimental platforms, with a focus on robust, reliable, and accurate performance across a range of conditions.
- **Chapter 8 – Conclusion and Future Work:** Summary and discussion of future research directions.
- **Appendix A – Derivation of Preintegrated Twist Factors:** Supporting material for Chapters 4 and 6.
- **Appendix B – Additional Articles:** Co-authored works published over the course of this thesis.

Chapter 2

Background

This chapter introduces important background concepts and principles for the remainder of the thesis and will be referred to in later chapters.

Section 2.1 will formally define the mathematical notation used, then Section 2.2 will highlight relevant properties of the hardware and sensors. Finally, important concepts and terminology in state estimation are outlined in Section 2.3.

2.1 Terminology

In this thesis we adopt the mathematical conventions defined in Table 2.1. We follow the frame decorator rules for physical quantities from [18], and where appropriate, simplify a time-dependent quantity as $a(t_i) = a_i$.

This thesis uses the common robotics terminology of *pose* to refer to the combination of position and orientation, and *twist* to refer to the combination

Quantity	**Description**	**Example**
Scalars	Lowercase italics	$a, b, c, \ldots$
States and landmarks	Bold italics	$\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \ldots$
Vectors	Lowercase Roman bold	$\mathbf{a}, \mathbf{b}, \mathbf{c}, \ldots$
Matrices	Uppercase Roman bold	$\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots$
Reference Frames	Typewriter typeface	$\mathtt{A}, \mathtt{B}, \mathtt{C}, \ldots$
Sensor measurements	Uppercase calligraphics	$\mathcal{A}, \mathcal{B}, \mathcal{C}, \ldots$
Set of measurement time indices	Uppercase sans	$\mathsf{A}, \mathsf{B}, \mathsf{C}, \ldots$

Table 2.1: Mathematical conventions used throughout this thesis [15].

Quantity	Sym.	Quantity	Sym.
State	$\boldsymbol{x}$	All states	$\mathcal{X}$
Measurement	$\boldsymbol{z}$	All measurements	$\mathcal{Z}$
Rotation	$\mathbf{R}$	Visual landmark	$\boldsymbol{m}$
Position	$\mathbf{p}$	Plane landmark	$\boldsymbol{p}$
Linear velocity	$\mathbf{v}$	Line landmark	$\boldsymbol{l}$
Bias – Accelerometer	$\mathbf{b}^a$	Twist bias – Linear	$\mathbf{b}^v$
Bias – Gyroscope	$\mathbf{b}^g$	Twist bias – Angular	$\mathbf{b}^\omega$
IMU measurements	$\mathcal{I}$	Kinematic measurements	$\mathcal{K}$
Camera measurements	$\mathcal{C}$	Lidar measurements	$\mathcal{L}$
Angular velocity	$\boldsymbol{\omega}$	Acceleration	$\mathbf{a}$
Joint angle	$\boldsymbol{\alpha}$	Joint velocity	$\dot{\boldsymbol{\alpha}}$
Joint torque	τ	Gaussian noise	$\boldsymbol{\eta}$
Normal distribution	$\mathcal{N}$	Covariance matrix	Σ

Table 2.2: Mathematical symbols used throughout this thesis.

of linear and angular velocity. Other mathematical notation commonly used in later chapters is defined in Table 2.2.

Throughout this thesis, we are interested in estimating the *state* of the platform, which is defined as,

$$\boldsymbol{x} \triangleq [\mathbf{R}, \mathbf{p}, \mathbf{v}, \mathbf{b}^g, \mathbf{b}^a] \in \mathrm{SO}(3) \times \mathbb{R}^{12} \tag{2.1}$$

where the individual elements are defined in Table 2.2. Unless otherwise specified, poses and landmarks are defined in the world frame `W`, velocities (and twist biases) are defined in the local base frame `B`, and IMU biases are defined in the IMU frame `I`.

2.2 Hardware and Sensor Description

In typical state estimation problems for mobile platforms, sensor inputs can be broken into two broad categories: *proprioceptive* and *exteroceptive* [19].

Proprioceptive sensors measure physical quantities relating to the robot's *interaction with the world* and include accelerometers, gyroscopes, joint encoders, and force/torque sensors. These sensors usually provide high frequency (>100 Hz), low latency (<1 ms) measurements of the robot's state. As these sensors do not directly



Figure 2.1: The main types of sensors used throughout this thesis. *Left to right:* IMU, joint/torque sensor, camera, and lidar.

measure the external environment, small measurement errors can accumulate over time, leading to drift in the state estimate.

Exteroceptive sensors measure physical quantities that describe a relationship between the robot and its *external environment*. Common examples are cameras, laser scanners, and radar. Exteroceptive sensors typically provide low frequency (<30 Hz), high latency (>10 ms) measurements of the robot's surroundings. If the environment is suitably descriptive, then these sensors can be used to estimate the robot's state. However, measurements from exteroceptive sensors can often be more complex and computationally demanding to process compared to proprioceptive sensors.

This thesis focuses on the problem of state estimation using a combination of proprioceptive and exteroceptive sensors—namely IMUs, cameras, lidars, and kinematic sensors (see Figure 2.1). This section will describe the important properties of these sensors and how they are typically integrated into robotic platforms.

Inertial Measurement Units

One of the simplest methods of performing robot state estimation is with an Inertial Measurement Unit (IMU). An IMU contains three orthogonal accelerometers and gyroscopes that measure proper acceleration $\mathbf{a}$ and angular velocity $\boldsymbol{\omega}$, respectively. *Proper acceleration* is defined as the physical acceleration acting on a body relative to freefall. This differs from *coordinate acceleration* which does not include the effects of gravity. IMUs can be small, inexpensive, and provide high frequency measurements that are independent of the external environment (except for gravity).

IMU measurements are typically modelled as having both additive white Gaussian noise ($\boldsymbol{\eta}^{\omega} \sim \mathcal{N}$ and $\boldsymbol{\eta}^{a} \sim \mathcal{N}$) and a slowly evolving bias ($\mathbf{b}^{g}, \mathbf{b}^{a}$) [20],

$$\tilde{\boldsymbol{\omega}} = \boldsymbol{\omega} + \mathbf{b}^{g} + \boldsymbol{\eta}^{\omega} \tag{2.2}$$

$$\tilde{\mathbf{a}} = \mathbf{a} + \mathbf{b}^{a} + \boldsymbol{\eta}^{a} \tag{2.3}$$

where $\tilde{\boldsymbol{\omega}}$ and $\tilde{\mathbf{a}}$ are the measured signals. Note that $\mathbf{a}$ includes the effects of gravitational acceleration. The IMU biases can be affected by temperature and other environmental conditions and are typically modelled using a slowly varying Gaussian random walk, $\dot{\mathbf{b}}^{g} \sim \mathcal{N}$ and $\dot{\mathbf{b}}^{a} \sim \mathcal{N}$.

IMU measurements can be integrated from a known starting position to estimate the robot's pose, however, the accuracy of IMU integration can vary by orders of magnitude depending on IMU quality [21]. Typically the largest source of error in IMU estimation is the IMU biases [22]. For example, if we use Equation (2.3) and assume $\boldsymbol{\eta}^{a} = 0$, $\mathbf{b}^{a} = 0.1\,\mathrm{m\,s^{-2}}$ (typical for a consumer-grade IMU [22]), and a stationary initial position, then there will be 5 m position error after just 10 s. This error is initially very small but grows quadratically with time.

Therefore, without extra information, the small, inexpensive IMUs used in most mobile platforms cannot be used alone for state estimation. A common solution is to couple IMUs with either prior motion models or other sensors [22].

Kinematic Sensors

We define a kinematic sensor as the combination of joint position, joint velocity, and torque sensors. There is most commonly one encoder and torque sensor per motor. Typically, a quadruped robot has twelve kinematic sensors, while a wheeled vehicle may only have two. It is common for legged robots to not have force/torque sensors installed on the feet due to hardware reliability issues.

Kinematic sensors are typically very accurate and provide high frequency, low latency measurements that are suitable for motor control. However, they only measure the relative motion between different parts of the robot's kinematic chain.



(a) IMU: Accelerometer biases can be unobservable under certain types of motion, including constant linear velocity.



(b) Kinematics: Slippery or deformable terrain (e.g., snow or mud) can cause drift in kinematic odometry.



(c) Camera: Dark or featureless environments can cause visual odometry to fail.



(d) Lidar: Environments with repetitive geometry (e.g., long tunnels) are challenging for lidar odometry [16].

Figure 2.2: Challenging environments for state estimation with different sensors.

State estimation algorithms need to infer the motion of the robot from the interaction of the wheels or legs with the environment.

Additionally, a critical input for control systems is an accurate, low latency velocity measurement. Kinematic sensors can infer the robot velocity either directly from the joint velocity or from joint torque measurements. IMU sensors can also measure velocity via integration of the IMU acceleration. This is in contrast to deriving position estimates from cameras and lidars which can produce noisy, high latency velocity signals that may not be suitable for control.

Cameras

Cameras measure the intensity of light on a photosensitive surface. They can detect light falling within particular spectral bands (e.g., greyscale, colour, thermal), and

Properties	IMU	Kinematics	Camera	Lidar
Type	Proprioceptive	Proprioceptive	Exteroceptive	Exteroceptive
Data	$\mathbf{a}$, $\boldsymbol{\omega}$	$\boldsymbol{\alpha}$, $\dot{\boldsymbol{\alpha}}$, τ	Image (mono/stereo)	Sparse point cloud
Freq. [Hz]	≥100	≥400	30	10
Dim. of meas.	6	36	6.2×10^6	2.8×10^4
Latency	Low	Low	Medium	High
Power [Watts]	<0.005	<0.15	3	8
Weight [g]	<20	30	40	830
Size [cm^3]	<0.015	25	50	600
Strengths	High frequency, independent of env.	High frequency, high accuracy	High resolution, 2D meas.	High accuracy, 3D meas.
Weaknesses	Accumulated error causes divergence of est.	Drift is difficult to observe and model	Odometry difficult in low-texture env.	Odometry difficult with repetitive structure

Table 2.3: Typical properties of different sensor modalities[1].

have various resolutions and lens shapes. The images produced are typically very high dimensional—for example, a single HD colour image contains $1920 \times 1080 \times 3 = 6.2 \times 10^6$ individual measurements. This high data rate requires carefully designed algorithms to process the signal in real time.

There are two main types of calibration required to use camera measurements for state estimation. The first is the *intrinsic calibration* which defines the internal properties of the camera, including focal length and lens distortion. The second is the *extrinsic calibration* which defines the transformation from the camera to another known frame on the platform (e.g., the IMU frame). This thesis makes use of the pinhole, radial-tangential, and Kannala-Brandt [23] (also known as fisheye or equidistant) distortion models. These parameters can be determined offline using tools such as Kalibr [24].

Since cameras only provide 2D information, multiple viewpoints of the same environment are required to estimate the 3D geometry of the world. Multiple viewpoints of a scene can be acquired using either a stereo camera pair, or by physically moving the same camera over time.

Lidar

[1]Values from a Bosch BMI055 MEMS IMU, Avago AEDM 5810 joint encoder, Flea3 FL3-U3-120S3C-C Camera, and Velodyne VLP-16 lidar. Values assume the camera is running at 1920×1080 resolution and a quadruped robot with three joints per leg, each measuring joint angle, velocity, and torque.

Platform	IMU	Kinematics	Camera	Lidar
Visual-Inertial Sensors	✓		✓	
Handheld Sensor Rigs	✓		✓	✓
Legged Robots	✓	✓(legs)	✓	✓
Wheeled Robots	✓	✓(wheel)	✓	✓
Drones	✓		✓	✓

Table 2.4: Typical sensor setups for different platforms. A variety of sensors are used to achieve reliable and accurate state estimation in a wide range of environments.

3D lidar (Light Detection and Ranging) sensors contain one or more distance-measuring lasers that continually move to measure different parts of the scene. Over time, individual laser measurements are accumulated into a sparse point cloud of the environment, commonly at 10 Hz. Since the measurements are collected over time (not simultaneously) the point cloud can suffer from motion distortion which needs to be corrected for accurate state estimation. These distance measurements can be very precise over a long range (up to 100 m) but can be influenced by dense fog, rain, and direct sunlight [25].

Lidars are often much heavier and have higher power consumption than cameras (see Table 2.3). This is mainly due to both the lasers and the components required to move them around (e.g., a motor and bearings for a rotational lidar).

Platform Description

As summarised in Table 2.3, each sensor modality provides complementary information for state estimation. This includes different frequencies, latencies, and failure modes. Therefore, to provide robust state estimation in all environments, a combination of sensors must be used. These sensors are typically time synchronised via software or a hardware-based triggering system (e.g., PTP or custom electronics (Tschopp et al. [26])).

Part of the motivation for this thesis is the deployment of mobile platforms in challenging environments including underground caverns, sewers, industrial facilities, and natural environments. Figure 2.2 shows common scenarios where individual sensor modalities can fail or become degenerate. *Degenerate scenarios* are defined

as situations in which the sensor measurements are insufficient to recover the full state of the robot [27]. For this reason, robotic platforms typically have a range of sensors to deal with different situations (Table 2.4). VILENS is designed to take advantage of all available sensor information for robust state estimation.

2.3 State Estimation Algorithms

This section will outline some of the key terminology used throughout this thesis. Listed below are some important definitions for state estimation systems:

Odometry vs Localisation: *Odometry* algorithms focus on producing a smooth, locally accurate state estimate, while *localisation* algorithms focus on global accuracy. This is often at the expense of update frequency and local smoothness. Mobile platforms frequently run both systems simultaneously for locally and globally accurate estimates. This thesis focuses on the first problem, *odometry*, but can be easily integrated with external localisation systems.

Loosely vs Tightly Coupled: These terms describe how the estimator handles correlations between different sensor measurements. *Loosely coupled* approaches process the measurements from each sensor (or set of sensors) independently to limit the algorithmic complexity. These separate estimates are then combined in a later step. This can potentially disregard correlations between different sensor states [28] or use the same measurement multiple times. On the other hand, *tightly coupled* approaches jointly estimate the state from all sensors simultaneously, which can improve accuracy. This thesis focuses on *tightly coupled* methods.

Front-end vs Back-end: State estimation algorithms are typically broken up into two main subsystems. The *front-end* is sensor-specific and processes the measurements to extract meaningful quantities for the optimiser. The *back-end* fuses the data from the front-end to produce a state estimate, often in a probabilistic manner. These subsystems can often be tightly coupled – for example, a sensor

processing algorithm may require an accurate state estimate to work correctly. This thesis introduces contributions for both the *front-end* and *back-end* subsystems.

Sparse vs Dense: Some sensors, including cameras and lidars, produce a vast amount of information about the environment. *Sparse* methods extract and track salient parts of the data to improve real-time operation, whereas *dense* methods use all measurements for state estimation. *Semi-dense* methods are a blend, often using a significant fraction of the data for estimation which can help create more informative gradients. This thesis focuses on *sparse* methods, but occasionally takes advantage of *dense* approaches. An example of a dense approach is Iterative Closest Point (ICP), which minimises the point-to-point distance between two point clouds [29].

2.3.1 Sensor Fusion Techniques

The aim of state estimation is to take a set of sensor measurements (or other inputs) $\mathcal{Z}$, and estimate the optimal value of a set of unknowns $\mathcal{X}$. For mobile platforms, we are interested in real-time operation to constantly produce updated estimates as soon as new sensor measurements are received.

These sensor fusion techniques can be broadly broken down into three distinct classes depending on the number of measurements being optimised at the same time. *Full batch smoothing* approaches recalculate the complete state history at every time step, while *fixed-lag smoothers* only estimate the previous N states of the robot. When $N = 1$, the approach is known as *filtering* and only the latest state is estimated at each time step.

This thesis focuses on *fixed-lag smoothing* as a trade off between speed and accuracy. This is important if the algorithm is to be run at high frequency on a mobile platform with limited computation. However, fixed lag smoothing approaches must be carefully designed to bound computation and keep the algorithm real time.

Filtering

The most common filtering method in the state estimation literature is the Kalman filter and its many variants [30]. Kalman filters aim to produce the optimal state estimate $\boldsymbol{x}_i^*$ at time t_i, given the previous state $\boldsymbol{x}_{i-1}$, and the current sensor inputs $\boldsymbol{z}_i$.

The algorithm is repeated at each time step to estimate the current state. It has been shown that the Kalman Filter (Kalman [31]) provides the optimal state estimate for linear systems. However, this assumption is unrealistic for the majority of robotics problems which often exhibit highly nonlinear behaviour.

To handle these nonlinearities, variants of the Kalman filter, including the Extended Kalman Filter (EKF), linearise the system around the current estimate at each time step. This approach can be simple to implement and computationally inexpensive. It assumes, however, that the system is approximately linear in the region of operation. If this is not true, linearisation errors can accumulate over time leading to *drift* in the state estimate. Notably, new measurements cannot affect previous state estimates, meaning that past errors cannot be corrected. This has been partially addressed in the Iterated EKF (Bloesch et al. [32]) where the linearisation point of past states can be updated if required.

The computation of the Kalman filter requires the inverse of an $n \times n$ matrix, where n is the number of states being estimated. The complexity of this operation is $O(n^3)$ (or $\Omega(n^2 \log n)$ for special cases [33]). This means that adding more states results in large increases in computation. This is the main computational bottleneck for Kalman filters (Stachniss et al. [34]) and limits how they can be used. This is significant if a large number of states need to be estimated, such as tracking landmark locations in a visual odometry algorithm.

Optimisation

Optimisation, similar to filtering, aims to find the Maximum A-Posteriori (MAP) state estimate $\mathcal{X}^*$, given the measurements $\mathcal{Z}$ [35]. This can be formalised as [36],

$$\mathcal{X}^* = \arg\max_{\mathcal{X}} P\left(\mathcal{X}|\mathcal{Z}\right) \tag{2.4}$$

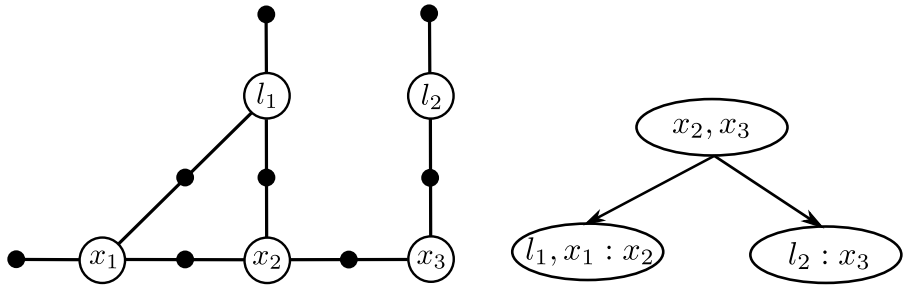


Figure 2.3: *Left:* A factor graph, consisting of nodes (circles) linked by factors (black dots). *Right:* The same factor represented as a Bayes Tree (Example from Dellaert and Kaess [36]).

Since problems can contain thousands of variables and measurements, computational considerations are important, especially as the problem gets larger.

Computation efficiency is often improved by exploiting any latent structure in the data. A common approach in state estimation is via *factor graphs* [36]. A factor graph is a bipartite graph between a set of factors and a set of nodes. Edges connect the nodes and factors, as shown in Figure 2.3.

For factor graphs, Barfoot [19] showed that the MAP estimate can be calculated from the product of each factor's likelihood function $\phi_i(\mathcal{X})$,

$$\mathcal{X}^* = \arg\max_{\mathcal{X}} \left(\prod_i \phi_i(\mathcal{X}) \right) \tag{2.5}$$

If we assume that each factor represents a measurement affected by white Gaussian noise then this is equivalent to the standard sum of least squares optimisation problem,

$$\mathcal{X}^* = \arg\min_{\mathcal{X}} \left(\sum_i \left(\| h_i(\mathcal{X}) \ominus \boldsymbol{z}_i \|^2_{\Sigma_i} \right) \right) \tag{2.6}$$

where $h_i(\mathcal{X})$ is the measurement model, Σ_i is the covariance associated with each measurement, and $\|\mathbf{r}\|^2_\Sigma \triangleq \mathbf{r}^T \Sigma^{-1} \mathbf{r}$. The $\ominus$ operator represents the generalised difference between two points on a manifold. For example, this is required to properly deal with differences between rotations over the SO(3) manifold [20].

Equation (2.6) can be solved efficiently using standard gradient descent-based methods, such as Gauss-Newton (GN) or Levenberg-Marquardt (LM) [37]. GN uses a second-order update step to iteratively reach the minima. This method

achieves almost a quadratic convergence to the minima when the objective function is approximately quadratic. However, GN optimisation can diverge if this is not the case [36]. The LM method extends GN with a damping parameter λ to limit the update step to a smaller *trust region*. This can improve converge for ill-conditioned problems.

2.3.2 Optimisation for State Estimation

State estimation problems (including *Simultaneous Localisation And Mapping (SLAM)*) require measurements to be received and processed in real time. The naive approach of continuing to add new measurements to the same graph means the graph, and thus the computation, will grow over time. This is a critical issue for deployment on mobile platforms where excessive resource usage can reduce estimation frequency, negatively affect other subsystems, or even cause the algorithm to fail.

There are several strategies for managing the computational complexity of state estimation problems. A common approach is *marginalisation*, where old states are regularly removed to maintain a fixed-size sliding window optimisation [28]. This limits the amount of computation required and ensures it does not grow over time.

A typical step in solving optimisation problems is factorising the information matrix λ. Two common approaches are Cholesky and QR factorisation [36] which produce the square-root information matrix $\sqrt{\Lambda}$. For example, the factor graph in Figure 2.3 (Left) can be represented by,

$$\sqrt{\Lambda} = \begin{array}{c} \begin{matrix} l_1 & l_2 & x_1 & x_2 & x_3 \end{matrix} \\ \begin{pmatrix} X & & X & X & \\ & X & & & X \\ & & X & X & \\ & & & X & X \\ & & & & X \end{pmatrix} \end{array} \tag{2.7}$$

This can be converted into a Bayes Tree (Figure 2.3, Right) for use by algorithms such as Incremental Smoothing And Mapping 2 [2] (iSAM2) to only update relevant states at each time step.

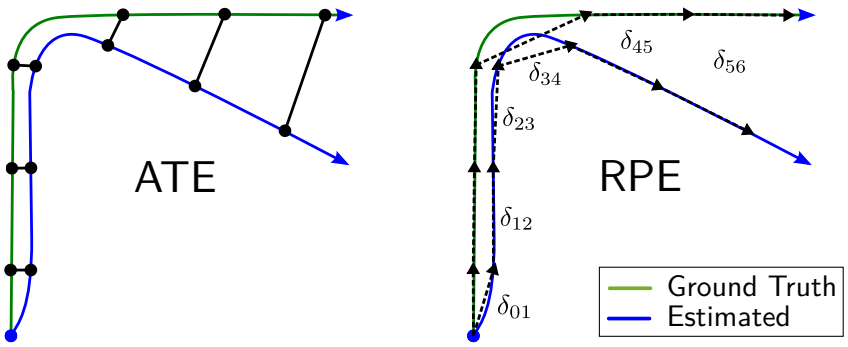


Figure 2.4: A comparison between ATE and RPE performance metrics. In this example, the estimated trajectory has a large ATE error while the RPE is low except for the one error at δ_{34}.

Another important consideration is how to handle outliers. One approach is via robust m-estimators, which can reduce the influence of measurements with very high error [38]. Some approaches can robustly handle up to 80 % outliers (GNC [39]) which can improve system performance in challenging environments.

For a detailed review of these topics please refer to Section 3.4 and Dellaert and Kaess [36].

2.3.3 Performance Metrics

An important consideration when evaluating state estimation algorithms is how to measure error. For localisation methods that estimate the *global* position of the robot (e.g., SLAM systems or systems with GPS), a common metric is the Absolute Trajectory Error (ATE) (See Figure 2.4). ATE measures the difference between each estimated and ground truth (GT) pose [40],

$$\text{ATE}_i = \mathbf{T}_{est,i} \ominus \mathbf{T}_{gt,i} \tag{2.8}$$

where the $\ominus$ operator is calculates the pose difference over SO(3) [41].

For algorithms that estimate the *relative* pose of the robot (e.g., odometry systems) ATE can be inappropriate. For example, a small odometry error at the start of the trajectory can cause a large ATE at the end of the trajectory, i.e., the specific location of the error can greatly affect the ATE metric (see Figure 2.4). A more appropriate metric for odometry is *Relative Pose Error (RPE)*,

which calculates the error over small segments of the trajectory δ_{ij} from t_i to t_j. It is formally defined as [40],

$$\text{RPE}_i = \delta_{est,ij} \ominus \delta_{gt,ij} \tag{2.9}$$

$$\text{RPE}_i = (\mathbf{T}_{est,i}^{-1}\mathbf{T}_{est,j}) \ominus (\mathbf{T}_{gt,i}^{-1}\mathbf{T}_{gt,j}) \tag{2.10}$$

In addition to accuracy, the robustness of an algorithm is very important in practice. This can be quantified as the *Success Rate (SR)* (Wang et al. [42]),

$$SR = \frac{N_{\text{successful}}}{N_{\text{total}}} \tag{2.11}$$

where $N_{\text{successful}}$ is the number of sequences without algorithm failure out of N_{total} runs. The ideal value is $SR = 1$ which means that algorithm always produced a valid state estimate.

These error metrics can be analysed using standard statistical techniques such as mean, standard deviation, or Root Mean Squared Error (RMSE). Since VILENS is an odometry system this thesis will focus on RPE error metrics.

Chapter 3

Literature Review

This chapter presents an overview of the relevant state estimation literature and is supported by specific literature reviews in each chapter.

This literature review will be broken down according to the sensor modality used. First, exteroceptive state estimation methods, including those based on camera and lidar sensors, will be explored in Section 3.1. Then, Section 3.2 will review proprioceptive methods, including those based on kinematics. Section 3.3 will present a detailed review of multi-sensor fusion techniques, including specific methods for legged robots. Finally, state-of-the-art methods for handling factor graph optimisation are reviewed in Section 3.4. A summary of the important methods discussed in this chapter is provided in Table 3.1.

For an overview of important terminology used in this literature review, please refer to Section 2.3.

3.1 Exteroceptive State Estimation

Both camera and lidar sensors capture vast amounts of data about the environment which can be used for state estimation. Handling of degenerate scenarios and performance on computationally limited on-board hardware are important considerations for mobile platforms.

Sensors	Loosely coupled systems	Tightly coupled systems
Vision + IMU	*	ROVIO [32], OKVIS [28], Kimera [43], VINS-Mono [44], VINS-Fusion [45], OpenVINS [46], ORB-SLAM3 [47]
Lidar + IMU	LOAM[1] [3], LeGO-LOAM[1] [48], Grant et al. [49], LiTAMIN2 [50], π-LSAM [51]	LIPS [52], LIOM [53], LIO-SAM [54], Yang et al. [55]
Kin. + IMU	Hartley et al. [56], Camurri et al. [57]	Lin et al. [58], Bloesch et al. [59], TSIF [4]
Vision + Lidar + IMU	V-LOAM [60], Wang et al. [61], CompSLAM [11], Super Odometry [9], Zhang et al. [62], R2LIVE [63]	LIMO [64], VIL-SLAM [65], LIC-Fusion 2.0 [66], LVI-SAM [67]
Vision + IMU + Kin.	Teng et al. [68]	Hartley et al. [69]
Vision + Lidar + IMU + Kin.	Pronto [70], HERO [10], LOCUS [71], Ma et al. [72]	**VILENS (proposed)**

Table 3.1: Summary of the most important methods discussed in this literature review. Note that mature fields such as VIO have converged on tightly coupled methods, while newer research areas often use simpler loosely-coupled approaches. Legend: [1]IMU input optional, *to the best of the author's knowledge most state-of-the-art methods are tightly coupled approaches [73].

3.1.1 Vision

Visual Odometry (VO) uses images from one or more cameras to estimate the camera's motion, and optionally, produce a 3D representation of the environment. For a detailed overview of visual SLAM, please refer to [34, 74, 75].

A more recent extension of VO is *Visual Inertial Odometry (VIO)*, which attempts to harness the different strengths of both camera and IMU sensors to provide high frequency and low drift state estimation (see Figure 3.1). This fusion is especially useful if there is highly dynamic motion, or if visual features cannot be robustly tracked. This is a relatively mature field and there are many commercial VIO systems available on the market.

The remainder of this section will explore different VO and VIO methods based on filtering and optimisation before exploring some alternative approaches, including those based on deep learning.

Filtering

As described in Section 2.3.1, filtering techniques can be simple and computationally efficient, but normally only use information from a single time-step, limiting their

accuracy and extensibility.

Bloesch et al. [32] proposed ROVIO, which used an iterated EKF framework to optimise the pose, linear velocity, IMU biases, and extrinsic calibration of a camera-IMU pair. Instead of using Cartesian coordinates, landmarks were modelled using bearing and distance to better handle the uncertainty in range. Visual features were detected using the FAST corner detector [76] then tracked using direct image methods, similar to the KLT feature tracker [77, 78]. OpenVINS [46] is a filtering-based VIO system that extended the Multi-State Constraint Kalman Filter (MSCKF) framework [79]. It achieved impressive results using online estimation of the camera extrinsics, intrinsics, and time offsets. However, since it is filter-based, the computation time would scale quadratically with the number of states (see Section 2.3). Therefore, to get the best accuracy, they split landmarks into two groups – one group is directly estimated as part of the state, and the other group is estimated using a separate cost function (to minimise the size of the state vector). This approach is challenging to extend to other sensor modalities, which is the core focus of this thesis.

Optimisation

According to a survey of VIO algorithms [80], filtering based methods, such as ROVIO [32], can strike a good balance of speed and accuracy on computationally constrained platforms. However, the most accurate and robust algorithms have used a fixed lag smoothing approach. Recent works based on fixed lag smoothing include VINS-Mono [44] (and later VINS-Fusion [45]), OKVIS [28], SVO+GTSAM [20], and Kimera [43].

VINS-Mono [44] used a factor graph back-end built on the Ceres Solver [81], which is a non-linear optimisation library. Similar to ROVIO, VINS-Mono directly tracked features in image space using KLT feature tracking. In a VIO benchmarking paper by Delmerico et al. [80], VINS-MONO was found to be the most accurate, but most computationally expensive approach of the VIO algorithms evaluated. To bound complexity, older states were marginalised out of the sliding window.

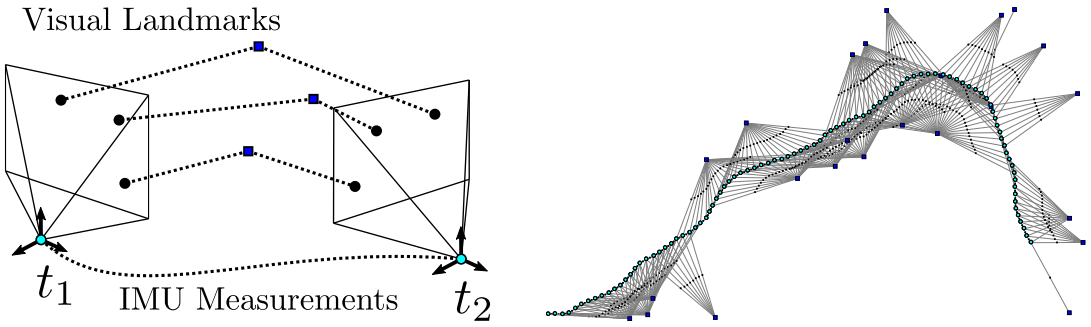


Figure 3.1: *Left:* In a VIO system, ego-motion is estimated by tracking visual landmarks over time and integrating IMU measurements between keyframes. *Right:* An example of a factor graph for visual-inertial odometry [36]. Visual factors link states to landmarks, which are tracked over time. Successive states are also linked with IMU measurements.

VINS-Mono, like other monocular VIO methods, used a complex initialisation procedure to estimate the scale and initial landmark locations. This required the camera to perform fast 3D rotations, which are not possible on many platforms.

Similarly, OKVIS [28] is based on factor graphs but instead matched image features across frames using the BRISK [82] feature detector and descriptor.

Forster et al. [83] presented Semi-Direct Visual Odometry (SVO) which tracked images features using direct methods, then used the g^{2}o optimisation library to minimise feature-based residuals [84]. SVO2 [85] extended this work with support for fisheye lenses, multi-camera systems, and edgelet features (small line segments). Their approach was very light-weight, requiring only 2.5 ms to compute the camera pose on a standard laptop, while providing similar accuracy to other state-of-the-art methods.

SVO+GTSAM [20] combines the visual features from SVO [83] with IMU measurements in an incremental optimisation based on iSAM2 [36]. This approach uses Smart Factors [86] to minimise computation by treating landmark locations as *auxiliary states* – these are states which are required for the calculations but are not the desired system output. These auxiliary states are estimated in separate, smaller optimisations and are only updated when necessary. By replacing multiple individual landmark factors with a single smart factor, the authors were able to achieve ∼80 % fewer factors and a ∼90 % reduction in computation time. However, this approach is less robust to noise and requires a good initialisation.

Kimera [87] also used a GTSAM backend, but instead focused on building a 3D mesh of the environment to improve state estimation by enforcing "structural regularities". In particular, Kimera first detected planes from landmark locations, then added a point-to-plane factor to constrain landmarks to the relevant plane. The authors showed that in situations where there were large planes in the environment this method increased accuracy by up to 27 % while having no impact on performance in non-planar environments. However, when only ∼10 planes were detected in the environment, the number of additional constraints added to the graph made the optimisation no longer real time. This was extended by Rosinol et al. [43] to incorporate semantic information, however, only simulation results were presented.

Campos et al. [47] presented ORB-SLAM3, an open-source VO/VIO algorithm. It tracked ORB features [88] and used DBoW2 [89] for loop closure detection. This is a mature system and has advanced features such as local feature mapping, multi-run localisation, and loop closure detection.

Deep Learning

In contrast to traditional visual odometry approaches, some authors have utilised deep neural networks for some or all of the deep learning pipeline. For an extensive review on deep learning for visual SLAM please refer to Chen et al. [73].

Some approaches have focused on end-to-end visual odometry, such as DeepVO by Wang et al. [90]. DeepVO used a combination of convolutional and recurrent neural networks to compute the relative transformation between pairs of successive images. They trained and tested on the KITTI dataset [91] and outperformed a traditional monocular VO method (VISO2 [92]). However, overfitting was an issue and their architecture was not generalisable to other platforms or cameras without retraining.

In contrast, Wang et al. [93] presented TartanVO, a learned monocular VO system that could generalise to new cameras and motions without retraining. Their pipeline was split into optical flow and pose networks. This simplified training and importantly, allowed camera intrinsics to be added to the network. To further enhance generalisability, they trained on the TartanAir simulated dataset [42] which

contains general 3D motion, unlike other vehicle-based datasets. They showed comparable performance to state-of-the-art traditional and learning-based methods even though the datasets and cameras were not seen during training.

Other approaches aimed to improve components of traditional feature-based systems. This includes improving feature matching (e.g., Superglue [94]), depth estimation from a single image (e.g., learning depth from monocular images using stereo ground truth [95]), image normalisation [96], and using a mixture of traditional and learned components for VIO [95]. A notable example is D3VO [97], which augmented a feature-based system with a network for depth prediction and pose estimation. The depth was used for 3D landmark initialisation and the depth uncertainty was used to adjust the weights of each feature in the optimisation. If feature tracking failed, the system relied on relative pose constraints from the pose network. If the calibration changed or different cameras were used, the neural networks may need retraining, limiting the algorithm's versatility.

Kaygusuz et al. [98] also used a learned fusion network to combine the outputs of separate VO networks in a multi-camera VO system. They showed their fusion network was more accurate than an EKF but it is unclear how these results generalise.

There have only been a few early works published on learned VIO. Almalioglu et al. [99] presented Self-VIO, a loosely coupled system with separate learned VO and IMU odometry networks. A learned adaptive fusion model weighted visual and IMU inputs based on the input data. They presented similar results to traditional methods on KITTI [91] and EUROC [100] datasets. However, their approach did not take into account calibration parameters or IMU biases, limiting it to trained datasets and platforms.

Methods based on deep learning can be computationally expensive, often requiring a GPU. For example, DROID-SLAM [101] required a server-grade GPU for online operation. Saputra et al. [102] focused real-time operation by reducing the size of their end-to-end VO system via knowledge distillation. By reducing the number of network weights by over 92 % the execution time dropped from 87 ms to 41 ms with only a 10 % loss in accuracy when tested on a low-power Nvidia Jetson TX2.

Overall, deep learning-based methods hold the promise of being more robust than traditional methods but suffer from overfitting, are expensive to train and run, and can be difficult to adapt to unknown environments.

Alternative Approaches

He et al. [103] recognised that traditional feature-based approaches can fail in common environments such as buildings with blank walls. The proposed PL-VIO algorithm utilised both point and line features, however, the authors found that state-of-the-art line detection and matching methods (LSD [104] and LBD [105]) were too time-consuming for online VIO.

Alternatively, Engel et al. [106] presented Direct Sparse Odometry (DSO), which directly uses the *photometric error* (instead of *re-projection error* used by other approaches) in a sliding window optimisation. This work presents competitive results across a range of datasets but requires careful photometric calibration of the camera.

Other approaches, such as KinectFusion [107], use VO to create a high density reconstruction of the environment, often at the expense of extra computation. KinectFusion, and other dense VO methods may not be suitable for some mobile platforms which do not have an onboard GPU.

Conclusion

VIO algorithms have been shown to achieve high accuracy and robustness in many situations, leading to several commercial products. A fundamental assumption of VIO is that there is sufficient visual texture in the scene to enable motion estimation. This assumption fails in dark, textureless, or highly dynamic environments which are a focus of this thesis.

3.1.2 Lidar

An alternative approach to VIO is lidar-based state estimation. These techniques are usually based on the geometry of the environment, and thus can fail when the geometry is either repetitive (e.g., long corridors), or not suitably descriptive

(e.g., large open fields). This section will explore four main types of lidar odometry: dense, semi-dense, sparse, and deep learning-based.

Dense

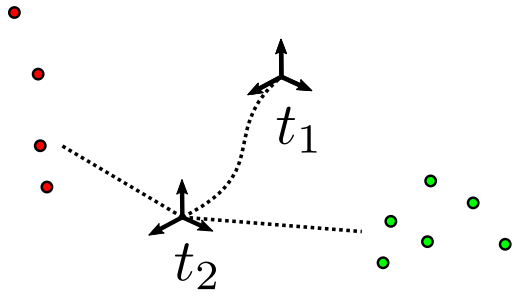
A classic dense lidar registration method is Iterative Closest Point (ICP) [108]. ICP aimed to find the relative transformation between two point clouds by matching and minimising the distance between individual points. It can be highly accurate but computationally expensive and relies on a good initialisation.

Park et al. [109] presented a dense lidar SLAM system based on a continuous time trajectory optimisation. The system computed odometry using point-to-plane residuals between the current lidar scan and a sparse local map, alongside IMU measurements. They achieved impressive mapping results, however, their system was not able to run in real time, even on a powerful desktop computer.

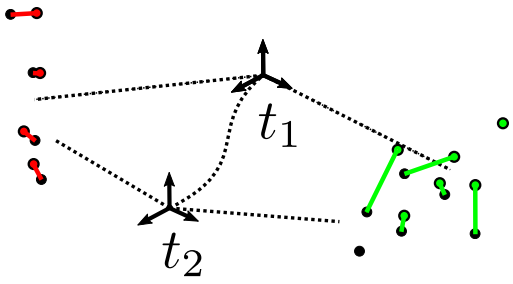
To reduce computation, some approaches such as LiTAMIN2 [50] used a voxel voting technique to dramatically reduce the number of points in the scan by over 99 %. When tested on the KITTI dataset [91] it achieved similar accuracy to state-of-the-art methods such as LOAM [3], but with odometry frame rates of up to 500 fps. The speed and accuracy depended on choosing an appropriate voxel size for each environment.

Semi-dense

Semi-dense lidar methods extract and match a large number of lidar feature points to estimate motion instead of using the entire scan. A pioneering work in this field was Lidar Odometry and Mapping [3] (LOAM) which filters the lidar scan to find points with very high or low *curvature*. This is roughly analogous to finding points that lie on *edges* or *planes*, respectively. The transform between two successive scans is found by minimizing point-to-line and point-to-plane residuals. A quick but low accuracy scan-to-scan matching typically runs at sensor frequency (e.g., 10 Hz). This estimate is then refined by comparing the current scan to a local map, which runs in a different thread at lower frequency (e.g., 2–5 Hz). It optionally included IMU measurements for frame-to-frame motion prediction, but



(a) Step 1: Edge (*red*) and plane (*green*) feature points are detected in each lidar scan.



(b) Step 2: Feature points are matched scan-to-scan (or scan-to-map) for egomotion estimation.

Figure 3.2: LOAM [3] used a *semi-dense* scan-to-scan and scan-to-map feature matching approach for motion estimation (illustrated above). However, it is important to note that individual features were not tracked over time.

did not incorporate the IMU measurements into the optimisation. An illustration of this method is shown in Figure 3.2.

Shan et al. [48] presented LeGO-LOAM, a variant of LOAM optimised for ground vehicles. Real-time performance was improved by segmentation-based outlier removal and assuming that a large ground plane is visible at all times. Similar to LOAM, it defines the curvature c_i for point p_i in the scan as,

$$c_i = \frac{\sum_{j \in S,\ j \neq i}(r_j - r_i)}{|S||r_i|} \tag{3.1}$$

where S is the set of 5 points on each side of p_i in the lidar scan line. When c_i is high then p_i is likely to be on an edge, and when c_i is low then p_i is likely to be on a plane. Note that both of these methods extract feature points by assuming an organised point cloud from a rotating lidar scanner.

Ye et al. [53] proposed LIOM, a method to jointly minimise the cost derived from lidar features and pre-integrated IMU measurements. This method performed worse than LOAM in slow-moving scenarios, but better in more dynamic ones, highlighting the importance of IMU during fast motion.

LIO-SAM [110] also extended the LOAM framework, this time by introducing scan matching at a local scale instead of a global scale. This allowed new keyframes to be registered to a sliding window of prior "sub-keyframes" merged into a voxel map. The extensive experimental results highlighted the quality of the SLAM

reconstruction. For long duration navigation they also used loop-closure and GPS factors to eliminate drift.

Lola-SLAM [111] instead used continuous scan slicing to produce high frequency state estimates by matching LOAM features from only small segments of the lidar scan as they are measured. They found that using 45° scan segments could achieve similar accuracy to LOAM with an eight-fold increase in state estimation frequency. There was only a ∼30 % increase in computation using this method.

Sparse

Sparse approaches extract high level features from the point cloud for motion estimation and often use a factor graph back-end. LIPS [52] was one of the first methods to tightly fuse 3D plane factors with IMU measurements in a factor graph framework. It used point-to-plane based cost functions, that did not directly represent the plane in the optimisation. Grant et al. [49] detected planes from a rotating lidar sensor using a voting method, combined with the geometric properties of the intersection of a conical lidar beam with a plane. These planes were combined in both a sliding window optimisation and a global pose graph. When insufficient planes were available, the algorithm fell back to a slower ICP-based method.

Motivated by Kaess [112] and the limited accuracy of LOAM's frame-to-frame tracking, π-LSAM [51] used a combination of planar tracking and ICP for real-time global point cloud registration. They found the use of large planar features dramatically reduced the map size and allowed for real-time operation. Since only planar features were used, this method may not work in unstructured environments.

Alternatively, Bernreiter et al. [113] performed a spectral analysis on the spherical projection of each lidar scan. The resultant distribution was used for global point cloud registration. Utilising extra information in the lidar scans, including intensity and reflectivity, allowed them to outperform other approaches that only considered range information. Optimal performance depended on selecting an appropriate frequency for adding scans to the map.

Deep Learning

Similar to deep learning for VO, there are several approaches to learning for lidar motion estimation. Nubert et al. [114] presented a self-supervised end-to-end network that takes two lidar range images and estimates the transform between them. This estimate was refined using the scan-to-map feature of LOAM [3]. It is, however, not clear how degenerate scenarios can be handled in this framework.

Similarly, Wang et al. [115] estimated the relative transformation between two point clouds in an end-to-end manner. After training on the KITTI dataset [91], their method, DeepPCO, outperformed other learning-based odometry methods but achieved lower accuracy than LOAM [3].

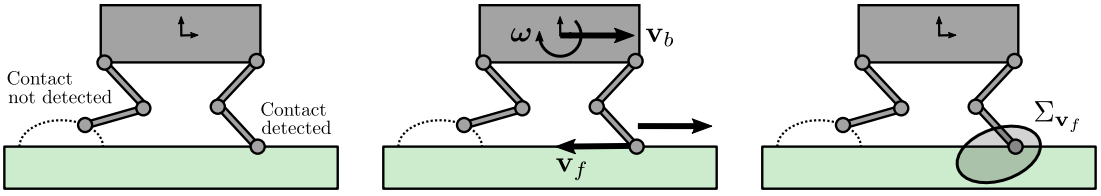
DLMO by Li et al. [116] detected feature points in a spherical projection of a lidar scan using a CNN. After refinement features were matched using Singular Value Decomposition (SVD). They achieved competitive performance to other state-of-the-art methods however did not provide details on run-time performance. Many other works that used deep learning-based lidar keypoints for localisation (e.g., [117]) required a GPU for online operation, making them challenging to deploy on mobile platforms.

Conclusion

Lidar-based state estimation can produce highly accurate pose estimates and detailed metrically-accurate maps of the environment. All lidar-based methods rely on the assumption that the geometry of the environment is suitably descriptive for egomotion estimation. This fails in geometrically repetitive environments (e.g., long corridors, open fields) [27] or if the sensor is occluded. Lidar odometry methods can also fail in dynamic environments unless this has been explicitly modelled in the algorithm.

3.2 Proprioceptive State Estimation

This section focuses on two main proprioceptive sensors – IMU and kinematic sensors. As mentioned in Section 2.2, IMU integration by itself is unstable and



(a) Step 1: Detect which feet are in stable (rigid) contact with the ground.

(b) Step 2: Estimate the robot's motion via the contact feet and angular velocity.

(c) Step 3: Estimate the contact point noise, which in general, can be non-Gaussian.

Figure 3.3: Overview of the fundamental steps in leg odometry. The motion of the robot's base can be estimated from the motion of the feet in stable contact with the ground (using forward kinematics) and the angular velocity of the robot's base. These measurements can be corrupted by non-ideal contact events, e.g., slippage, ground deformation, or kinematic chain flexibility. These contact events are a major source of error in kinematic methods [119].

can diverge after just a few seconds with the typical small, inexpensive IMUs found in mobile platforms.

Some approaches, such as TLIO [118], use neural networks to learn a motion model from IMU data alone. This can achieve impressive results when the platform is exhibiting similar motion to the training data, however, this approach may not generalise well to new environments and motions.

The remainder of this section will focus on state estimation using kinematic sensors, with a focus on handling contact nonlinearities (e.g., deformation and slippage), which are one of the major sources of error in kinematic odometry [119].

3.2.1 Kinematics

Kinematic sensors (including joint encoders, and force/torque sensors) provide high frequency, low latency measurements and thus are well suited for legged robot state estimation. Accurately estimating the robot's motion from kinematics is non-trivial and relies on complex modelling of the robot's interaction with the ground.

Using kinematic sensing on legged robots is often referred to as *leg odometry*. The fundamental principles of leg odometry are outlined in Figure 3.3. One of the largest sources of drift in kinematic methods is non-ideal ground contacts which may be caused by mechanical flexibility, slippage, and terrain deformation [119].

These effects are dependent on both the gait and terrain properties which makes them very difficult to model analytically.

One of the first papers to study this problem was Roston et al. [120] using the Ambler hexapod robot. The authors used the kinematics of the robot's legs to find the robot's motion via a least squares problem using Singular Value Decomposition (SVD). This achieved a 2 % error rate over a soil-covered flat surface in a small lab environment. This method assumed the feet had no slippage. It also exploited the fact that the robot often had more than three feet in contact with the ground simultaneously, greatly simplifying the state estimation problem.

Most leg odometry methods assume each foot has a point contact with the ground. This means that only one or two contact points are not enough to fully constrain the motion of the robot's base (at least one angular component of the robot's base will be unobservable). A logical extension is to incorporate another high frequency sensor to make the full state observable—usually an IMU.

One of the first examples of kinematic-inertial state estimation was presented by Lin et al. [58] on the RHex six Degrees of Freedom (DOF) hexapod. They used four accelerometers and a single gyroscope for sensing instead of a traditional integrated IMU, which enabled the measurement of the base's angular acceleration. These sensors were combined in an EKF framework and significantly reduced the state estimation error compared to using each sensor modality in isolation. This framework was specifically tuned for a single "jogging" gait where different process models were used during each phase of the gait. It also did not estimate the yaw of the robot. Similarly, Reinstein et al. [121] created a state estimator for a quadruped robot that was designed only for a specific gait cycle and stride length.

Bloesch et al. [122] presented a kinematic-inertial estimator for a 12-DOF quadruped robot using joint encoders and an on-board IMU. The state included the pose and linear velocity of the robot's base frame, foot contact positions, and the IMU biases. The authors fused these measurements using an Observability Constrained EKF (OCEKF). This approach suffered from singular states in certain unlikely robot configurations which meant that certain parts of the state became

unobservable. It also showed significant drift, even over a time period of only 15 s. Bloesch et al. [59] improved this method with an Unscented Kalman Filter (UKF), but it was still possible to enter singular states in some unlikely robot configurations.

In later work, Bloesch et al. [4] presented a new filter-based state estimator for mobile robots called the Two State Implicit Filter [4] (TSIF). This approach avoided the need for an explicit process model, and instead used measurement models that only depended on the previous and current states. This was the default state estimator provided for the ANYmal quadruped robot [6]. Unfortunately, it suffered from significant drift making it unusable for planning over longer time horizons [123].

Hartley et al. [56] presented an invariant EKF for kinematic-inertial state estimation. They provided detailed proof of the convergence properties of their filter under different initial conditions. They assumed the contact point was fixed and only evolved with Brownian motion, which is not a valid assumption when there is significant slippage or terrain deformation.

All of the above methods assume ideal contact behaviour. The next section will highlight works that address non-ideal contact events, which are one of the major sources of drift in leg odometry.

3.2.2 Modelling Contact Deformation and Slippage

Most kinematic-inertial estimators are based on the assumption that the stance foot location is entirely static throughout the stance period (yet affected by Gaussian noise). This assumption is violated every time there is slippage, kinematic chain flexibility, or terrain deformation. This is usually handled by using *contact detection* algorithms, which aim to detect and ignore feet that are not in fixed contact with the ground. This can be effective for sporadic slippage or deformation events. When these occur regularly, they need to be modelled.

Fahmi et al. [119] have shown that incorrect contact detections on soft ground (i.e., incorrectly detecting a "rigid" contact while the leg and ground are still deforming) are a key contributor to leg odometry drift.

In Bloesch et al. [59], a threshold on the Mahalanobis distance of the filter innovation was used to detect and ignore outlier leg odometry measurements, i.e., measurements from legs that were incorrectly classified as being in rigid contact with the ground.

Hwangbo et al. [124] presented a probabilistic approach where kinematic and dynamic measurements were fused in a Hidden Markov Model (HMM). This approach was later integrated with a dynamic trotting controller and the robust slip detection allowed the ANYmal robot to walk on slippery ice [125]. Notably, this work focused on improving control, not state estimation.

Also motivated by improving control performance, Bledt et al. [126] proposed a probabilistic contact detector to enable contact-based control policies (i.e., changing the control policy when different feet are in contact with the ground). They estimated the contact probability using a Kalman Filter. The prediction is based on the control input, and the update is based on a model of the contact force and ground height (assuming the ground height follows a Gaussian distribution). This is combined with a finite state machine to minimise false detections. They showed >99 % correct contact detections in controlled scenarios, using a force plate for ground truth.

Kim et al. [127] proposed a fixed lag smoother that fused kinematic and inertial measurements using costs for both position and velocity from kinematics. They proposed a slip detection method based on a threshold of the contact foot velocity. This required the threshold to be hand-tuned for different environments.

In contrast, Camurri et al. [57] proposed a contact detector that used maximum likelihood estimation to determine the contact force threshold. The authors also presented a probabilistic impact detector that adapted the measurement covariance online to reject unreliable measurements. These thresholds needed to be adapted for specific gaits and were estimated offline.

On the other hand, Lin et al. [128] proposed a deep learning-based method for estimating the contact state on the MIT Mini-Cheetah quadruped [129]. Their network estimated the contact state from a combination of kinematic and IMU

measurements. It achieved an average of 94 % accuracy across different datasets, but required over one million data points for supervised training of the neural network.

Conclusion

Most leg odometry methods in the literature have focused on high-frequency, smooth state estimation for control, often at the expense of long term accuracy. This drift is partially caused by ignoring the significant non-Gaussian noise arising from non-ideal ground contacts. Additionally, the majority of existing works ignore modelling errors, such as incorrect leg lengths, which can also lead to non-Gaussian errors. In contrast, this thesis reduces leg odometry drift by explicitly modelling the non-Gaussian contact noise, and through the fusion of other sensor modalities.

3.3 Multi-Sensor Fusion

Multi-sensor fusion algorithms combine complementary sensor information to improve the robustness and accuracy of state estimation. This requires careful consideration to get the benefits of each different sensor, without inheriting their weaknesses. A key differentiator is how different systems handle sensor failures (or degenerate modes). Loosely coupled approaches tend to design metrics to detect and reject unreliable inputs. On the other hand, tightly coupled approaches tend to assume that combining all available sensor information will be sufficient to constrain the estimate.

This section will first discuss visual-inertial-lidar methods, then focus on specific multi-sensor fusion algorithms for legged robots.

3.3.1 Vision, IMU, and Lidar

The aim of visual-inertial-lidar state estimation is to provide the robustness of visual tracking with the metric accuracy of lidar. This can help to avoid degeneracy and improve the accuracy and smoothness of the state estimate.

Loosely Coupled Approaches

In many recent works [11, 60, 61, 70] vision has been combined with lidar and IMU in a loosely coupled manner.

In [60], Zhang et al. presented V-LOAM, an extension of LOAM [3] that used a monocular VIO system to generate a prior to improve lidar scan matching. Lidar scans were only matched frame-to-frame and no loop closures were performed, which meant the system could suffer from drift during long term operation. To improve this, a Visual-Inertial-Lidar SLAM system was introduced by Wang et al. [61], who used a similar approach for odometry but performed a global pose graph optimisation by maintaining a keyframe database.

Palieri et al. [71] presented LOCUS, which was deployed as part of the DARPA Subterranean Challenge. LOCUS accepted a variety of different odometry inputs (visual, kinematic, or wheel odometry) as a prior for lidar scan-to-scan and scan-to-map odometry methods based on GICP [130]. Similarly, HERO [10] ran multiple odometry algorithms in parallel then selected only the best odometry source to use. This switch was based on a wide range of hand-crafted metrics, including the number of visual features or the number of invalid lidar scan points. This approach can make algorithms very difficult to adapt to different scenarios because these thresholds may need to be retuned. A similar approach was also proposed in Reinke et al. [131] for autonomous vehicles.

Zhao et al. [9] proposed Super Odometry, a loosely coupled factor graph with relative pose constraints from different odometry sources. Degeneracy was handled by increasing the covariance on particular measurement inputs by analysing the eigenvalues of each system's information matrix. This means that important measurements may be ignored during degenerate scenarios, which is precisely when extra measurements may be required.

An alternative approach to sensor fusion has been demonstrated by Zhang et al. [62]. A cascaded filtering approach produced successively more accurate state estimates through IMU, VIO, and lidar scan matching modules. Their approach achieved a relative position error of just 0.22 % over a 9.3 km dataset. They used a

complex degeneracy handling procedure that aimed to only use the non-degenerate directions of each modules' estimate. This was highly dependent on their specific sensing architecture and may be non-trivial to adapt to different systems.

CompSLAM by Khattak et al. [11] was a loosely coupled approach that used a visual/thermal-inertial prior for lidar scan matching in underground environments. Degeneracy was handled by hard-switching between sensor modalities which can cause loss of accuracy and may be effective if multiple odometry sources are degenerate at the same time. Leg odometry could be optionally used as a prior if VIO failed.

In contrast, Valente et al. [132] used a deep learning-based odometry system to fuse a 2D lidar scanner and camera. The two sensors were first pre-processed by individual CNNs before being fused by a third network. However, their system was only tested on the KITTI dataset [91] and was limited to 2D estimation.

Tightly Coupled Approaches

To avoid degeneracy and make state estimation more robust, tight integration of multi-modal sensing capabilities (vision, lidar, and IMU) has been explored by some authors [55,64–66,133]. In LIMO [64], Graeter et al. presented a bundle adjustment-based visual odometry system. They estimated visual feature depth from lidar measurements by re-projecting lidar points into image space and associating them to visual features. This helped to maintain accurate scale but only used a very small fraction of the lidar information. Shao et al. [65] introduced VIL-SLAM where they combined VIO with lidar-based odometry as separate sub-systems rather than doing a joint optimisation. Notably, this work only used the lidar for mapping, not odometry.

To perform joint state optimisation, many approaches [55, 66, 133] used the Multi-State Constraint Kalman Filter (MSCKF) framework [79]. Yang et al. [55] tightly integrated plane features from an RGB-D sensor, point features from vision, and IMU measurements using an MSCKF. To limit the state vector size, most point features were treated as MSCKF features and linearly marginalised, while only a few point features that enforced point-on-plane constraints were kept in the

state vector as SLAM features. This method could only use planes within a 3.5 m range due to the limited accuracy of the RGB-D camera.

LIC-Fusion, introduced by Zuo et al. [133], used an MSCKF to tightly combine IMU measurements, extracted lidar edge features, and sparse visual features. In a more recent follow up, LIC-Fusion 2.0 [66], Zuo et al. introduced plane features into their pipeline, tracking them over time using the same MSCKF framework. They efficiently detected planes using point-to-point correspondences but only presented results from indoor experiments with large planar walls.

LVI-SAM [67] introduced a state estimation system split into two tightly coupled factor graph estimators. The visual-inertial (VI) and lidar-inertial (LI) subsystems were based on VINS-Mono [44] and LIO-SAM [54], respectively. The two estimators were connected at multiple points, including visual loop detection from VI, initialisation from LI, and an odometry prior from VI for scan matching. These connections make the system difficult to adjust to new sensor setups. Degeneracy was handled by being robust to failure of either subsystem.

Similarly, in R2LIVE [63] the authors combined three separate subsystems for state estimation. The first was an IEKF to combine IMU measurements, lidar plane features (similar to LOAM [3]) and visual features. The output of this module was fed into a factor graph (similar to VINS-Mono [44] with prior constraints from lidar odometry), and a lidar mapping module. They demonstrated an improvement in accuracy compared to using the subsystems independently. The intertwined architecture meant that some measurements were used more than once, potentially biasing the results.

LVIO-Fusion [134] also proposed state estimation based on multiple different subsystems – VIO, LVIO, and a global pose graph optimisation. This complex structure handled degenerate scenarios by training an actor-critic framework to dynamically weight odometry sources depending on the input data. It presented comparable results to state-of-the-art methods, however, parts of the algorithm were customised for wheeled vehicles and were not generalisable to other platforms.

Conclusion

In contrast with previous works, this thesis presents a joint optimisation of the above modalities within a single, consistent factor graph optimisation. This unified structure allows for the natural handling of degenerate scenarios and simple, flexible addition of new sensor inputs, including kinematics.

3.3.2 Legged Robots

Multi-sensor fusion techniques for legged robots can be broadly broken down into filtering and optimisation-based approaches.

Filtering

An early example of heterogeneous sensor fusion on legged robots was by Chilean et al. [135] on the DLR Crawler hexapod. An Indirect Information Filter (IIF) was used to combine IMU, stereo visual odometry, and leg odometry measurements. By combining these inputs, the robot was able to achieve accurate localisation in a small, highly textured lab environment.

Wooden et al. [136] developed a loosely coupled state estimator for Boston Dynamics' 110 kg Big Dog quadruped robot. It combined the output of separate kinematic and visual odometry systems with an EKF. The IMU was only used for orientation estimation. This estimate was primarily used to develop a terrain map for obstacle avoidance.

This was extended by Ma et al. [72] for the Boston Dynamics' LS3 quadruped. They used an EKF to fuse leg odometry, IMUs, and a stereo camera. The IMUs were tactical and navigation grade and thus provided high accuracy measurements. They achieved a translation drift rate of 1 %, however, this system used GPS whenever there was reliable coverage. For long-term operation, they included different system modes, such as Zero Velocity Updates (ZUPTs) to constrain the estimate while stationary. Leg odometry was only used as a backup, specifically, when visual odometry failed and the robot was not falling.

Similarly, Fallon et al. [137] presented Pronto, an efficient modular EKF state estimation system. For the Atlas robot, two instances of the same EKF were used – one for control and the other for footstep planning. The control estimator only used kinematic and inertial inputs to produce a low latency, smooth estimate, while the other additionally used stereo camera and lidar information for long term low-drift state estimation. This is because the inclusion of high latency exteroceptive information led to non-smooth state estimates which were not suitable for control. The EKFs used an IMU driven process model in the update step, instead of a traditional dynamics model. This estimator enabled the robot to walk for tens of minutes without stopping.

Pronto was extended by Nobili et al. [138] and Camurri et al. [70] with demonstrations on the ANYmal [6] and HyQ [139] quadrupeds. Nobili et al. was one of the first to consider how discrete updates from high latency sensors can be incorporated in a control loop and obtained an error of just 0.72 % with all sensing modalities. It also showed graceful performance degradation as sensor inputs were removed from the system. The loosely coupled structure meant that correlations between measurements were ignored. For example, the IMU measurements did not aid the visual odometry.

Teng et al. [68] extended the Invariant EKF from Hartley et al. [56] with pose measurements from a Realsense T265 tracking camera. They achieved accurate velocity tracking during their short indoor slip experiments. Their system was very sensitive to the covariance used for the pose measurement, and a poor choice could lead to divergence. This required an empirical online covariance tuning system.

Optimisation

Hartley et al. [140] proposed one of the first methods to incorporate kinematics into a factor graph. They extended the floating base state with the foot contact location and defined two new factors to incorporate forward kinematics (shown in Figure 3.4). In effect, this imposed a zero velocity constraint on the contact point. These factors were combined with preintegrated IMU factors [20] and relative pose measurements

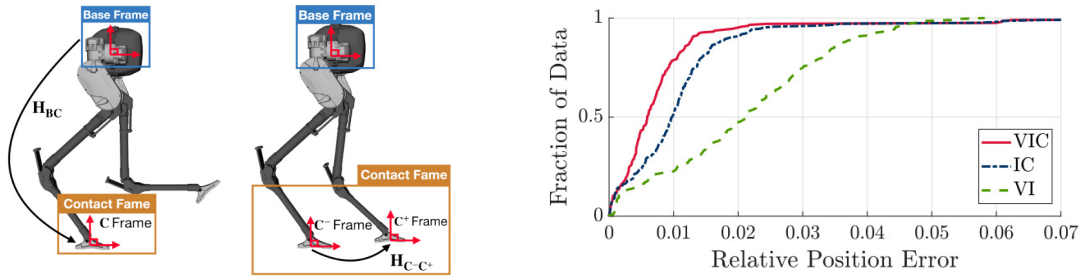


Figure 3.4: Hartley et al. [140] presented a novel method of adding kinematic information into a factor graph. *Left:* The contact frame was explicitly estimated as part of the state. *Right:* Fusing visual, inertial, and kinematic measurements (VIC) reduced the RPE compared to using only two sensor modalities.

from an external VO system (SVO [85]). This method was designed for a biped robot with a single contact point and did not consider multiple simultaneous foot contacts.

The authors extended this work in [69] to support multiple footsteps in the same kinematics factor. Both works provided limited experimental results, only demonstrating the algorithm on the Cassie bipedal robot in controlled environments, with experiments lasting less than 2 min.

Fourmy et al. [141] proposed a factor graph based state estimation system for the Solo12 quadruped [142]. It fused both kinematic and dynamic information to estimate the base frame of the robot and performed online extrinsic calibration of the Centre of Mass (CoM) location. The authors claimed this was important for control since the CoM of their robot was not known precisely from the design. Only very limited experimental results were presented, where only the robot's torso moved while the feet remained stationary.

In the related field of motion planning, Xie et al. [143] modelled both the kinematics and dynamics of a three DOF manipulator using factor graphs. This approach took into account dynamics, contact forces, and joint actuation limits.

Conclusion

Current research into multi-sensor fusion for legged robots has been mainly limited to loosely-coupled filtering-based methods. These methods ignore important correlations between measurements, limiting their accuracy. This is in contrast to mature fields such as VIO where tight fusion and optimisation-based methods are common.

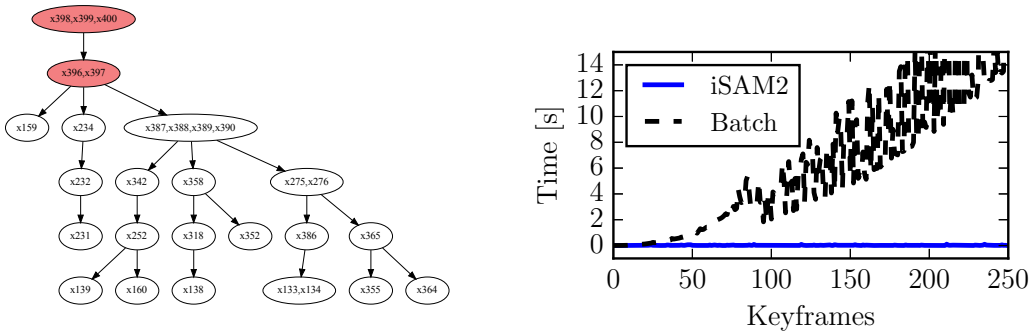


Figure 3.5: *Left:* iSAM2 [2] used a Bayes tree representation to perform incremental optimisation. As new measurements were received only affected parts of the graph are recalculated (highlighted in red). *Right:* This approach greatly reduced the computation time compared to batch optimisation [20].

3.4 Factor Graph Optimisation

Computation and real-time performance are critical for deployment on mobile robots. For many state estimation algorithms, the most computationally expensive component is factor graph optimisation.

A common strategy is to use marginalisation to perform sliding window optimisation. For example, Leutenegger et al. [28] marginalised non-keyframe and older states to bound the computational complexity of OKVIS.

Alternatively, iSAM (Incremental Smoothing And Mapping) by Kaess et al. [144] performed incremental updates using QR factorisation of the square root information matrix, avoiding the need for marginalisation. Unfortunately, iSAM required periodic batch updates to perform variable reordering and relinearisation. This caused non-linear errors to accumulate between batch updates, and real-time performance was challenging due to highly variable optimisation time.

In a follow up work, iSAM2 [2] linked the matrix factorisation to the novel *Bayes Tree* graphical model (shown in Figure 3.5) for efficient real-time performance. It supported both QR and Cholesky factorisation methods.

Similarly, AprilSAM [145] by Wang et al. incrementally updated the Cholesky factorisation and adaptively switched between incremental and batch updates. Specifically, the authors introduced a new dynamic ordering to reduce the com-

putation required for incremental Cholesky. At low computation levels, which are desired for mobile platforms, iSAM2 achieved lower error on a simulated pose graph dataset with loop closures.

Most state estimation systems use an external optimisation library. Some of the most popular libraries include General Graph Optimisation (g^2o) [84], Ceres [81], and Georgia Tech Smoothing and Mapping [1] (GTSAM). In a recent comparison paper, Juric et al. [146] evaluated several optimisation libraries on a visual-SLAM dataset consisting of both poses and landmarks. GTSAM achieved the lowest ATE error out of all the solvers, even outperforming Ceres and g^2o by an order of magnitude in one dataset. The optimisation time of g^2o and GTSAM was always lower than Ceres, but was highly dependent on the dataset. Importantly, this comparison did not consider the effect of GTSAM's iSAM2 implementation which is likely to have been significantly quicker in some problems.

3.5 Conclusion

This thesis differs from previous works by using a method to jointly optimise visual, lidar, kinematic, and inertial measurements in a single factor graph. Unlike other approaches, VILENS directly adds all landmarks to the same graph, allowing the measurement noise to be accurately modelled. Importantly, this tight fusion allows degenerate scenarios to have handled naturally, without the need for hand-crafted metrics to determine when measurements should be ignored. This approach is easily reconfigurable to new robots and sensors by changing the types of factors added to the graph.

Chapter 4

Visual, Inertial, Kinematic State Estimation

This chapter introduces the VILENS state estimation system and presents a novel method of fusing visual, inertial, and kinematic measurements.

This work was motivated by the need to improve state estimation on the ANYmal quadruped robot [6]. TSIF [4], ANYmal's in-built kinematic-inertial state estimator, is locally accurate and suitable for high frequency control, but suffers from significant drift over time. This drift rate increases when traversing rough terrain or performing dynamic manoeuvres. Overcoming challenging terrain is one of the key benefits of legged robots, and thus, this drift is a limiting factor in deployment. For example, both local terrain mapping and motion planning depend on an accurate state estimate to operate correctly. By improving the on-board state estimation, this work aims to enable legged robots to perform more dynamic motions over rougher terrain, and thus unlock the true potential of legged platforms.

Specifically, this chapter describes two novel methods of integrating kinematics into a factor graph. Factor graph-based state estimation for legged robots is still in its infancy, but this approach is mature in other subfields (e.g., VIO for Micro Aerial Vehicles (MAV)).

There are two articles presented in this chapter. The first presents one of the earliest attempts to tightly fuse leg odometry, vision, and IMU measurements into a factor graph framework. It was also the first work to validate the use of factor graphs on legged robots through extensive real-world experiments.

The second article extends the VILENS framework to include online bias estimation for leg odometry. This aimed to estimate and reject the characteristics biases present in leg odometry by defining a new *preintegrated twist factor*. This was the first work to explicitly handle non-ideal contacts within leg odometry, rather than just assuming that contact points had zero-mean Gaussian error.

Note that almost all quadruped robots run a high frequency, low latency state estimator for control purposes. To avoid recomputation of the kinematics, the works in this chapter use the output of this estimator as an input to the factor graph. A method of more tightly integrating the kinematics is presented in Chapter 6.

4.1 Robust Legged Robot State Estimation Using Factor Graph Optimization

The following article was published in the IEEE Robotics and Automation Letters (2019) and presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (2019) [12]. A video summary of this publication can be found at: `https://youtu.be/p8o7mJPy4_w`.

IEEE ROBOTICS AND AUTOMATION LETTERS. PREPRINT VERSION. ACCEPTED JULY, 2019 1

Robust Legged Robot State Estimation Using Factor Graph Optimization

David Wisth, Marco Camurri, and Maurice Fallon

***Abstract*—Legged robots, specifically quadrupeds, are becoming increasingly attractive for industrial applications such as inspection. However, to leave the laboratory and to become useful to an end user requires reliability in harsh conditions. From the perspective of state estimation, it is essential to be able to accurately estimate the robot's state despite challenges such as uneven or slippery terrain, textureless and reflective scenes, as well as dynamic camera occlusions. We are motivated to reduce the dependency on foot contact classifications, which fail when slipping, and to reduce position drift during dynamic motions such as trotting. To this end, we present a factor graph optimization method for state estimation which tightly fuses and smooths inertial navigation, leg odometry and visual odometry. The effectiveness of the approach is demonstrated using the ANYmal quadruped robot navigating in a realistic outdoor industrial environment. This experiment included trotting, walking, crossing obstacles and ascending a staircase. The proposed approach decreased the relative position error by up to 55% and absolute position error by 76% compared to kinematic-inertial odometry.**

***Index Terms*—Legged Robots; Sensor Fusion; Localization**

I. Introduction

FOR legged robots to become truly autonomous and useful they must have a consistent and accurate understanding of their location in the world. This is essential for almost every aspect of robot navigation, including control, motion generation, path planning, and local mapping.

Legged robots pose unique challenges to state estimation. First, the dynamic motions generated by the robot footsteps can induce motion blur on camera images as well as slippage or flexibility in the kinematics. Second, the strict real-time requirements of legged locomotion require low latency, high frequency estimates which are robust. Third, the sensor messages are heterogeneous, with different frequencies and latencies. Finally, the conditions where legged robots are expected to operate are far from ideal: poorly lit or textureless areas, self-similar structures, muddy or slippery ground are some examples.

For these reasons, legged robots have traditionally relied on filter-based state estimation, using *proprioceptive* inputs (IMUs, force/torque sensors and joint encoders) [1], [2], [3]. While these approaches give reliable and high frequency estimates, they are limited in their ability to reject linear and angular position drift.



Fig. 1. Experiments were conducted using an ANYbotics ANYmal quadruped [5] in outdoor environments, including the Oil Rig training facility shown above. The ANYmal robot has 12 actuated degrees of freedom, an IMU, dual forward-facing RealSense D435 cameras and a Velodyne VLP-16 LIDAR. Video: `https://youtu.be/p8o7mJPy4_w`.

For statically stable walking, leg odometry drift is low enough that terrain mapping can be used for continuous footstep planning and execution [4]. However for dynamic locomotion, position drift is much higher which makes such mapping ineffective, as illustrated in Fig. 2. The result of this is that we do not know what the shape of the terrain is under the robot due to drift and cannot properly plan motions.

To overcome this limitation, some previous works have incorporated *exteroceptive* inputs (cameras and LIDAR) into filtering estimators in a loosely coupled fashion. This has been successfully demonstrated on legged machines operating in field experiments [6], [7].

However, because these filters marginalize all previous states of the robot, it is not possible to fully exploit a (recent) history of measurements, as smoothing methods can.

Research into smoothing approaches applied to Visual-Inertial Navigation Systems (VINS) is now well established in the Micro-Aerial-Vehicle (MAV) community. On a MAV this approach has been successful due to careful time synchronization of the IMU and cameras, smooth vehicle motion, and also the absence of the challenges of articulated legged machines.

A recent work by Hartley et al. [8] demonstrated that a VINS approach could be adapted to legged robots — in their case, a biped. Their initial results were promising, but only tested in a controlled scenario. Additionally, vision was inte-

Manuscript received: February, 24, 2019; Revised June, 13, 2019; Accepted July, 20, 2019.

This paper was recommended for publication by Editor Nikos Tsagarakis upon evaluation of the Associate Editor and Reviewers' comments. This research has been conducted as part of the ANYbotics research community. It was part funded by the Innovate UK-funded ORCA Robotics Hub (EP/R026173/1), the EU H2020 Project THING, a Royal Society University Research Fellowship (Fallon) and a Google DeepMind studentship (Wisth).

The authors are with the Oxford Robotics Institute at the University of Oxford, United Kingdom {davidw, mcamurri, mfallon}@robots.ox.ac.uk

Digital Object Identifier (DOI): see top of this page.

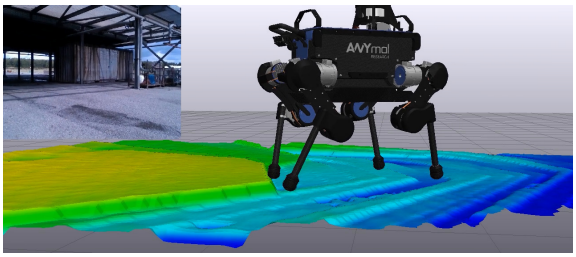


Fig. 2. An elevation map created by the GridMap package [9] while walking over flat terrain shows discrete ridges due to position drift in the current kinematic state estimator. This makes dynamic locomotion and footstep planning much more challenging.

grated as relative pose constraints, rather than incorporating the feature residuals directly into the optimization.

Contribution

This paper aims to progress the deployment of state estimation smoothing methods in realistic application scenarios. Compared to previous research, we present the following contributions:

- We present the first state estimation method based on factor graphs that tightly integrates visual features directly into the cost function (rather than adding a pose constraint from a separate visual inertial module), together with preintegrated inertial factors and relative pose constraints from the kinematics. We will refer to our proposed method as VILENS (Visual Inertial LEgged Navigation System);
- We demonstrate the performance and robustness of our method with extensive experiments on two field scenarios. Challenges included motion blur, dynamic scene occludants, textureless and reflective scenes, and locomotion on uneven, muddy and slippery terrain;
- We demonstrate that a low-cost consumer-grade depth camera, the RealSense D435, is sufficient to significantly improve the state estimate in these conditions.

The remainder of the article is presented as follows: in Section II we describe the previous research in the field; the theoretical background of the algorithm is described in Sections III and IV. Section V outlines the details of our implementation. The experimental results and their discussion are shown in Sections VI and VII. Finally, Section VIII concludes the article.

II. Related Work

Multi-sensor fusion for mobile robot state estimation has been widely described in the literature [10]. Here, we limit our discussion to filtering and smoothing approaches with particular focus on dynamic legged robots.

A. Filtering Approaches

After the spread of filtering methods for proprioceptive state estimation [1], there has been interest in including exteroceptive data, particularly Visual Odometry (VO).

Ma et al. [6] presented a method based on an Extended Kalman Filter (EKF) with an error state formulation developed for the Boston Dynamics LS3. The system was primarily driven by inertial predictions with VO updates: a modular sensor head performs the fusion of a very high quality tactical grade IMU with two hardware synchronized stereo cameras. The leg odometry (fused with an additional navigation grade IMU in the body) was used only in case of VO failure. Their extensive evaluation (over several km) achieved $1\,\%$ error per distance traveled.

Nobili et al. [7] recently presented a state estimator for the HyQ quadruped robot which used an EKF to combine inertial, kinematic, visual, and LIDAR measurements. In contrast to [6], the EKF was driven by an inertial process model with the primary corrections coming from leg odometry (synchronized in EtherCAT) at the nominal control rate ($1\,\mathrm{kHz}$). The VO updates and an ICP-based matching algorithm were run on a separate computer at lower frequency and integrated into the estimator when available. This allowed the use of the estimator inside the control loop and has been recently demonstrated in dynamic motions with local mapping [11].

B. Smoothing Approaches

There has been a significant body of work on visual inertial navigation, especially for use with MAVs. A recent benchmark paper [12] evaluated a number of state-of-the-art methods on the EuRoC dataset [13]. The maturity of the field was highlighted by the fact that many algorithms achieved an average Relative Position Error (RPE) of less than $20\,\mathrm{cm}$ per $35\,\mathrm{m}$ traveled. The authors concluded that the best performing algorithms were OKVIS [14], ROVIO [15], and VINS-Mono [16]. All of these algorithms perform windowed optimization to achieve the most accurate state estimate while bounding computation time. An exception was SVO+GTSAM [17] which loosely coupled the SVO visual odometry algorithm with IMU data using iSAM2 as the smoothing back-end [18].

The methods were, however, typically designed assuming that the IMU and cameras were synchronized in hardware, and that the vehicle motions were smooth. They are difficult to implement on legged platforms due to their hardware complexity and the high vibrations caused by locomotion.

As mentioned above, the first approach to use smoothing/optimization on legged robots was the work of Hartley et al. [8] which presented a fusion of kinematic, inertial, and visual information, again using iSAM2 as the smoothing back-end. They presented a mathematical framework to model contact points as landmarks in the environment, similar to [1]. Their system incorporated contact information from only a single contact point at a time, and directly integrates (as pose constraints) the relative motion estimate of the SVO2 [19] algorithm. The approach was tested using inertial and visual input from a MultiSense S7 sensor (which was hardware synchronized) mounted on a Cassie biped (from Agility Robotics). A short indoor experiment of $60\,\mathrm{s}$ showed that adding vision into the optimization reduced the relative position error.

Our approach aims to combine best practice from the mature field of VINS – including windowed optimization and

TABLE I
SENSOR SPECIFICATIONS

Sensor	Model	Hz	Specs
IMU	Xsens MTi-100	400	Init Bias: $0.2\,^\circ/\mathrm{s}$ | $5\,\mathrm{mg}$ Bias Stab: $10\,^\circ/\mathrm{h}$ | $15\,\mathrm{mg}$
Camera	RealSense D435	30	*Keble College Dataset:* Resolution: 848×480 px FoV: $91.2^\circ \times 65.5^\circ$ Imager: IR global shutter *Oil Rig Dataset:* Resolution: 640×480 px FoV: $69.4^\circ \times 42.5^\circ$ Imager: RGB rolling shutter
Encoder	ANYdrive	400	Resolution: $<0.025^\circ$
Torque	ANYdrive	400	Resolution: $<0.1\,\mathrm{N\,m}$

tight integration of visual features – with legged odometry to provide robust state estimation for legged robots. In Section VI we show our system outperforming both kinematic-inertial and visual-inertial approaches in large-scale outdoor urban and industrial experiments.

III. Problem Statement

We wish to track the linear position and velocity of a 12 Degrees of Freedom (DoF) legged robot equipped with an industrial grade MEMS IMU, an RGB/IR camera, joint encoders and torque sensors. The sensor specifications are detailed in Table I.

In Fig. 3 we provide a schematic of the reference frames involved. The pose of the robot's base B expressed in the fixed-world, inertial frame W is defined as:

$$\mathbf{T}_{\mathtt{WB}} = \begin{bmatrix} \mathbf{R}_{\mathtt{WB}} & {}_{\mathtt{W}}\mathbf{p}_{\mathtt{WB}} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathrm{SE}(3)$$

The IMU and camera sensing frames are I and C, respectively. The relative transformations between B, I and C are assumed to be known from CAD design. The location of a foot in base coordinates is expressed as ${}_{\mathtt{B}}\mathbf{p}_{\mathtt{BK}}$.

A. State Definition

Borrowing the notation from [17], we define the state of the system at time t_i as:

$$\boldsymbol{x}_i \triangleq [\mathbf{R}_i, \mathbf{p}_i, \mathbf{v}_i, \mathbf{b}_i] \tag{1}$$

where the couple $(\mathbf{R}_i, \mathbf{p}_i)$ expresses the robot pose and $\mathbf{v}_i \in \mathbb{R}^3$ is the robot linear velocity. As is common in this field, the stack of gyro and accelerometer IMU biases $\mathbf{b}_i = [\mathbf{b}_i^\omega \;\; \mathbf{b}_i^a] \in \mathbb{R}^6$ replaces the angular velocity, which is directly measured by the IMU.

Let K_k be the set of camera keyframe indices up to time $t_k \in \mathbb{R}$. We assume that for each keyframe image $\mathcal{C}_i$ (with $i \in \mathsf{K}_k$) a number of landmark points ${}_{\mathtt{W}}\mathbf{m}_j$ are visible, where $j \in \mathsf{M}_i \subseteq \mathsf{M}$; M_i indicates the set of landmark indices visible from keyframe $\mathcal{C}_i$ out of the full set of landmarks, M. We

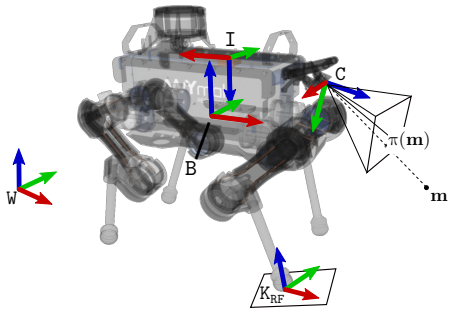


Fig. 3. Reference frames conventions. The world frame W is fixed to earth, while the base frame B, the camera's optical frame C, and the IMU frame, I are rigidly attached to the robot's chassis. When a foot touches the ground (e.g., the Right Front, RF), a contact frame K (perpendicular to the ground and parallel to W's y-axis) is defined. The projection of a landmark point $\mathbf{m}$ onto the image plane is $\pi(\mathbf{m})$.

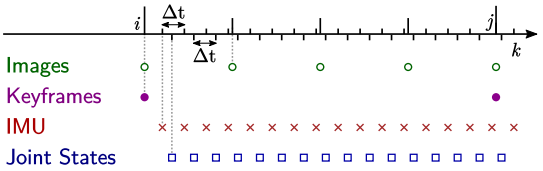


Fig. 4. The sensors inputs are images, IMU measurements, and joint states, which are in general unsynchronized with each other.

then define the objective of our estimation problem $\mathcal{X}_k$ as the history of robot states and landmarks detected up to t_k:

$$\mathcal{X}_k \triangleq \bigcup_{\forall i \in \mathsf{K}_k} \left[\{\boldsymbol{x}_i\}, \bigcup_{\forall j \in \mathsf{M}_i} \{\mathbf{m}_j\} \right] \tag{2}$$

B. Measurements

The input measurements consist of camera images, IMU readings, and joint sensing (position, velocity and torque). The measurements are not assumed to be synchronized. However, we assume they have a common time frame. The IMU and joint states have the same frequency (see Fig. 4). For each pair of consecutive keyframe indices $\Delta i = i-1, i \in \mathsf{K}_k$ we define $\mathsf{I}_{\Delta i}$ as the set of IMU measurements indices such that $\forall\; m \in \mathsf{I}_{\Delta i}$ we have $t_{i-1} \leq t_m < t_i$. We then indicate with $\mathcal{I}_{\Delta i} = \bigcup_{\forall j \in \mathsf{I}_{\Delta i}} (\boldsymbol{\omega}_j, \mathbf{a}_j)$ all angular velocity and proper acceleration measurements collected between time t_{i-1} and t_i. Analogous definitions apply to joint states $\mathcal{Q}_{\Delta i} = \bigcup_{\forall j \in \mathsf{Q}_{\Delta i}} (\mathbf{q}_j, \dot{\mathbf{q}}_j, \boldsymbol{\tau}_j)$, which include all joint positions, velocities and torques collected between time t_{i-1} and t_i. Strategies to account for synchronization issues are discussed in Section V-A.

Finally, we then let $\mathcal{Z}_k$ denote the set of all measurements up to time t_k:

$$\mathcal{Z}_k \triangleq \bigcup_{\forall i \in \mathsf{K}_k} \{\mathcal{I}_{\Delta i}, \mathcal{C}_i, \mathcal{Q}_{\Delta i}\} \tag{3}$$

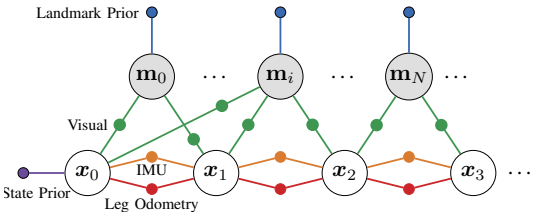


Fig. 5. The factor graph consists of state and landmark nodes linked by prior, visual, inertial, and leg odometry factors.

C. Maximum-A-Posteriori Estimation

The aim of the factor graph framework is to maximize the posterior the history of states and landmarks $\mathcal{X}_k$, given the history of measurements $\mathcal{Z}_k$:

$$\mathcal{X}_k^* = \arg\max_{\mathcal{X}_k} p(\mathcal{X}_k|\mathcal{Z}_k) \propto p(\mathcal{X}_0)p(\mathcal{Z}_k|\mathcal{X}_k) \tag{4}$$

Where the last member of (4) is the likelihood function, which is proportional to the posterior and therefore can be used as a cost function. If the measurements are conditionally independent and corrupted by zero mean Gaussian noise, then (4) is equivalent to a least squares problem of the form:

$$\mathcal{X}^* = \arg\min_{\mathcal{X}_k} \sum_{\mathcal{T}} \sum_{\forall i \in \mathsf{K}_k} \|\mathbf{r}_{\mathcal{T}_i}\|^2_{\Sigma_{\mathcal{T}_i}} \tag{5}$$

where $\mathbf{r}_{\mathcal{T}_i}$ is the residual of the error between the predicted and measured value of type $\mathcal{T}$ (e.g., IMU $\mathcal{I}$) at keyframe index $i \in \mathsf{K}_k$. The quadratic cost of each residual is weighted by the corresponding covariance $\Sigma_{\mathcal{T}_i}$.

From (2) and (3) the optimization becomes the following:

$$\begin{aligned}\mathcal{X}^* = \arg\min_{\mathcal{X}} &\sum_{j\in M} \|\mathbf{r}_{\mathbf{m}_{j,0}}\|^2_{\Sigma_{\mathbf{m}_{j,0}}} + \sum_{i\in\mathsf{K}_k} \|\mathbf{r}_{\mathcal{I}_{\Delta i}}\|^2_{\Sigma_{\mathcal{I}_{\Delta i}}} + \\ &+ \|\mathbf{r}_0\|^2_{\Sigma_0} + \sum_{i\in\mathsf{K}_k}\sum_{j\in\mathsf{M}_i} \|\mathbf{r}_{\mathbf{m}_j}\|^2_{\Sigma_{\mathbf{m}_j}} + \sum_{i\in\mathsf{K}_k} \|\mathbf{r}_{\mathcal{Q}_{\Delta i}}\|^2_{\Sigma_{\mathcal{Q}_{\Delta i}}}\end{aligned} \tag{6}$$

where the residuals are from: landmark prior, IMU, state prior, camera and leg odometry factors, respectively. These factors will be used to create the factor graph structure shown in Fig. 5. In the next section we define each residual of (6).

IV. Factor Definitions

A. Prior Factors

Since this is an odometry system, prior factors are required during initialization to anchor the unobservable modes (i.e., position and yaw) to a fixed reference frame. The residual is defined as the error between the estimated state $\boldsymbol{x}_0$ and the prior $\boldsymbol{x}_{p_0}$:

$$\mathbf{r}_0(\boldsymbol{x}_0, \mathcal{Z}) = \begin{pmatrix} \Phi(\mathbf{T}_0^{-1}\mathbf{T}_{p_0}) \\ \mathbf{v}_0 - \mathbf{v}_{p_0} \\ \mathbf{b}_0^a - \mathbf{b}_{p_0}^a \\ \mathbf{b}_0^\omega - \mathbf{b}_{p_0}^\omega \end{pmatrix} \tag{7}$$

where $\Phi : \mathrm{SE}(3) \mapsto \mathbb{R}^6$ is the lifting operator [17]. The prior state of the system is determined by IMU initialization, assuming the robot is stationary.

B. Visual Odometry Factors

The visual odometry residual consists of two components. The first is the difference between the measured landmark pixel location, $(u_{i,j},\ v_{i,j})$, and the re-projection of the estimated landmark location into image coordinates, (π_u, π_v) using the standard radial-tangential distortion model. The residual is defined as:

$$\mathbf{r}_{\mathbf{m}_j} = \begin{pmatrix} \pi_u(\mathbf{R}_i, \mathbf{p}_i, \mathbf{m}_j) - u_{i,j} \\ \pi_v(\mathbf{R}_i, \mathbf{p}_i, \mathbf{m}_j) - v_{i,j} \end{pmatrix} \tag{8}$$

The second is the error between the prior on the landmark location $\mathbf{m}_{j,0}$ and the estimated landmark location $\mathbf{m}_j$:

$$\mathbf{r}_{\mathbf{m}_{j,0}} = \mathbf{m}_j - \mathbf{m}_{j,0} \tag{9}$$

Legged robots typically move dynamically with velocities parallel to the optical axis of the camera. This motion can lead to underconstrained landmark positions. To this end, a landmark prior is generated online through an initial triangulation procedure (see Section V). The covariance, $\Sigma_{\mathbf{m}_{i,j}}$, is determined by the typical feature tracking accuracy (in our case, we use 1 pixel).

C. Preintegrated IMU Factors

We use the IMU preintegration algorithm described by Forster et al. [17]. This approach preintegrates the IMU measurements between nodes in the factor graph to provide high frequency state updates between optimization steps. The preintegrated IMU measurements are then used to create a new IMU factor between two consecutive keyframes. This will use an error term of the form:

$$\mathbf{r}_{\mathcal{I}_{\Delta i}} = \left[\mathbf{r}^\mathsf{T}_{\Delta\mathbf{R}_{\Delta i}}, \mathbf{r}^\mathsf{T}_{\Delta\mathbf{p}_{\Delta i}}, \mathbf{r}^\mathsf{T}_{\Delta\mathbf{v}_{\Delta i}}, \mathbf{r}^\mathsf{T}_{\Delta\mathbf{b}_{\Delta i}}\right] \tag{10}$$

where $\mathcal{I}_{\Delta i}$ are the IMU measurements between times $i-1$ and i. The individual elements of the residual are defined as:

$$\mathbf{r}_{\Delta\mathbf{R}_{\Delta i}} = \log\left(\Delta\tilde{\mathbf{R}}_{\Delta i}(b^g_{i-1})\right)\mathbf{R}^T_{i-1}\mathbf{R}_i \tag{11}$$

$$\begin{aligned}\mathbf{r}_{\Delta\mathbf{p}_{\Delta i}} = {} & \mathbf{R}^T_{i-1}\left(\mathbf{p}_i - \mathbf{p}_{i-1} - \mathbf{v}_{i-1}\Delta t_{\Delta i} - \frac{1}{2}\mathbf{g}\Delta t^2_{\Delta i}\right) \\ & -\Delta\tilde{\mathbf{p}}_{\Delta i}(b^g_{i-1}, b^a_{i-1})\end{aligned} \tag{12}$$

$$\begin{aligned}\mathbf{r}_{\Delta\mathbf{v}_{\Delta i}} = {} & \mathbf{R}^T_{i-1}\left(\mathbf{v}_i - \mathbf{v}_{i-1} - \mathbf{g}\Delta t_{\Delta i}\right) \\ & -\Delta\tilde{\mathbf{v}}_{\Delta i}(b^g_{i-1}, b^a_{i-1})\end{aligned} \tag{13}$$

$$\mathbf{r}_{\Delta\mathbf{b}_{\Delta i}} = \mathbf{b}_i - \mathbf{b}_{i-1} \tag{14}$$

where $\Delta\tilde{\mathbf{R}}_{\Delta i}, \Delta\tilde{\mathbf{p}}_{\Delta i}, \Delta\tilde{\mathbf{v}}_{\Delta i}$ are the preintegrated IMU measurements defined in [17]. The covariance, $\Sigma_{\mathcal{I}_{\Delta i}}$, is based on an offline Allan Variance analysis.

D. Leg Odometry Factors

Leg Odometry (LO) estimates the incremental motion of a walking robot given its kinematic sensing and the perceived contacts between the robot's feet and the ground. The main assumption behind LO measurements is that the absolute velocity of a contact point is zero. This condition is met when the Ground Reaction Force (GRF) at a contact point is inside a hypothetical friction cone. Torques at contact points are neglected as quadruped robots' feet are idealized as points.

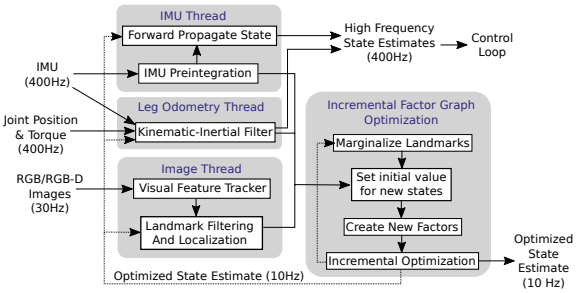


Fig. 6. The proposed state estimation architecture consists of measurement handlers which process the sensor input and the current state from the optimizer to create the factors. These factors are then smoothed using iSAM2 for incremental factor graph optimization.

Since the GRF, the terrain inclination, and the coefficient of static friction are typically unknown, an appropriate fusion of the dynamic model, kinematics, and IMU is required. In this work, we rely on the state estimator provided by ANYbotics, the Two-State Implicit Filter (TSIF) [3]. We use the relative robot motion estimated by the TSIF to formulate relative pose factors so as to further constrain the robot's motion estimate from the factor graph.

Given two keyframes at times t_{i-1} and t_i, the corresponding pose estimates $\widetilde{\mathbf{T}}_{i-1}$, $\widetilde{\mathbf{T}}_i$ are computed via linear/slerp interpolation from the filter history and used to produce the relative pose constraint:

$$\mathbf{r}_{\mathcal{Q}_{\Delta i}} = \Phi\left((\mathbf{T}_{i-1}^{-1}\mathbf{T}_i)^{-1}\widetilde{\mathbf{T}}_{i-1}^{-1}\widetilde{\mathbf{T}}_i\right) \tag{15}$$

where the covariance $\Sigma_{\mathcal{Q}_{\Delta i}}$ is provided by the filter, and Φ is the lifting operator defined in [17].

E. Zero Velocity Update Factors

To limit drift and factor graph growth when the robot is stationary, we detect zero velocity motion from camera frames by calculating the average feature motion:

$$\overline{\Delta x} = \frac{\sum_{i=k-N}^{k}\left(\sqrt{(u_i - u_{i-1})^2 + (v_i - v_{i-1})^2}\right)}{N+1} \tag{16}$$

If $\overline{\Delta x}$ is below a certain threshold $\beta = 0.5$ pixels over $N = 10$ successive frames, we stop adding image landmark measurements to the graph and simply add a zero relative pose factor to the graph of the same form as (15). The thresholds were tuned by calculating $\overline{\Delta x}$ when the camera was stationary. These factors reduce computation when the robot is stationary and can reduce overall drift by up to 20%.

V. Implementation

The factor graph optimization was implemented using the iSAM2 incremental optimization library [18]. The structure of the system is shown in Fig. 6. The algorithm consists of a series of measurement handlers running in separate threads that process the different sensor inputs. When a new node is created (e.g., when an image measurement is received) each of the measurement handlers adds new factors to the graph.

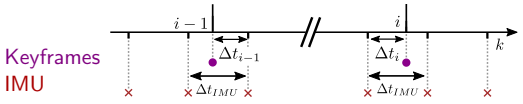


Fig. 7. We consider and account for camera/IMU asynchronousity so as to improve the accuracy of the IMU factors.

The output of the factor graph optimization is then fed back to the measurement handlers.

A. Synchronization

Given two consecutive keyframe indices $i-1, i \in \mathsf{K}_k$, preintegrating all IMU measurements between times t_{i-1} and t_i would result in an incorrect motion estimation as the IMU measurement timestamps may not be aligned with the image frames (see Fig. 7). To avoid this, we correct the Δt of the IMU measurements directly before and after the camera image timestamp t_i:

$$\Delta t_{i-1} = t_{\text{IMU}} + \Delta t_{\text{IMU}} - t_i \tag{17}$$

$$\Delta t_i = t_i - t_{\text{IMU}} \tag{18}$$

where we assume constant acceleration and angular velocity between IMU measurements.

B. Visual Feature Tracking

A core component of the VILENS system is the tight integration of visual features into the optimization. It can provide lower drift state estimates by tracking and optimizing the robot state and landmark positions over many observations. Our visual feature tracking method is based upon the robust pixel-based tracking approaches used in ROVIO [20] and VINS-Mono [16].

Visual features are first detected using the Harris Corner Detector and then tracked through successive images using the Kanade-Lucas-Tomasi feature tracker. This method provides sub-pixel accuracy and is well-suited to the constrained but jerky motions typical of a legged robot. After the tracking, outliers are rejected using RANSAC with a fundamental matrix model similar to [16]. New features are then detected in the image to maintain a minimum number of tracked features. These new features are constrained to be a minimum distance (in image space) from existing features, to ensure an even distribution of features across the image.

To limit the graph growth, we estimate the location of a feature in the world frame only after it has been observed more than $N_{\text{obs}} = 30$ times. When depth is available, it is used for the initial landmark location estimate. When only monocular data are available, we triangulate the landmark location $\mathbf{m}_{j,0}$ using the last N_{obs} frames with the Direct Linear Transformation (DLT) algorithm from [21]. If the landmark is successfully triangulated with a depth smaller than $d_{\max} = 50\,\text{m}$ then this and successive measurements of the same landmark are added to the graph.

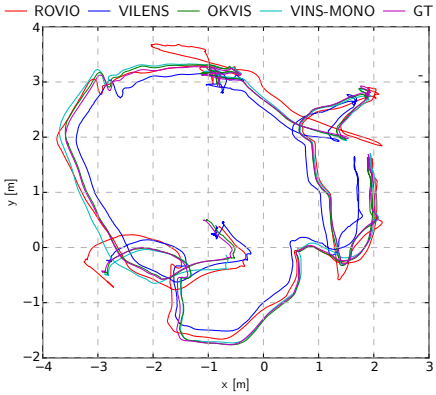


Fig. 8. Performance comparison between our baseline VILENS system (blue), ROVIO (red), OKVIS (green), and VINS-Mono (cyan) compared to VICON ground truth for EuRoC V2_01 dataset.

C. Marginalization

An important consideration in a legged robot system is latency, and without marginalization, the time taken by iSAM2 optimization increases over time [18]. This becomes an important consideration when operating the robot for extended periods of time.

We marginalize states older than a threshold (typically, $10\,\mathrm{s}$) and landmarks which are no longer observed, whilst keeping a minimum number of nodes in the factor graph. These marginalized states and landmarks are replaced with a simple linear Gaussian factor based on the current linearization point of the node. This Gaussian factor has the same form as the residual defined in (7).

VI. Experimental Results

In this section, we present the experimental evaluation of the proposed algorithm on three different datasets: EuRoC, Keble College and Oil Rig. The first dataset is a purely VINS dataset collected on a MAV. The other two datasets were collected using our ANYmal robot in different outdoor environments: a college campus and an industrial oil rig firefighter training facility.

A. EuRoC Dataset

To demonstrate that our approach builds upon a stand-alone VINS system, we evaluated it on a portion of the EuRoC benchmark dataset [13] and compared it to several state-of-the-art VINS algorithms including OKVIS [14], ROVIO [20], and VINS-Mono [16]. The estimated trajectory from the EuRoC V2_01 dataset is shown in Fig. 8.

In brief, we found that our core system can achieve comparable performance to these VINS algorithms. This demonstrates that the estimator can function without leg odometry information, which is important when that modality becomes unreliable (e.g., on slippery or soft ground).



Fig. 9. The Keble College dataset involved the ANYmal trotting around an open urban environment.

B. Outdoor Datasets Experimental Setup

The outdoor experiments were conducted using the 12-DoF ANYmal quadruped [5] (Fig. 1).

The measurements from the motors (i.e., joint states) were synchronized via EtherCAT, while the other sensors were synchronized by Network Time Protocol (NTP). The sensor configurations used for the two datasets are specified in Table I.

To generate ground truth, we collected a dense gravity-aligned prior map of the site using the Leica BLK-360 3D laser scanner. Afterwards, we performed ICP localization (using the AICP algorithm [22]) against the prior map using data from ANYmal's Velodyne VLP-16 LIDAR. Both map and LIDAR sensor were used as the ground truth only. This provided a ground truth trajectory with approximately $5\,\mathrm{cm}$ accuracy, but only at $2\,\mathrm{Hz}$.

C. Keble College Dataset

The first outdoor dataset was collected in a urban environment at Keble College, Oxford, UK. The dataset consists of the robot trotting on a concrete path around a $28\,\mathrm{m} \times 60\,\mathrm{m}$ open lawn surrounded by a residential building (Fig. 9). The main challenges were vegetation moving in the wind, long distances to visual features ($>10\,\mathrm{m}$), and limited angular motion, which made feature triangulation difficult.

We ran four trials each approximately $22\,\mathrm{m}$ in length and evaluated the mean and standard deviation of the Relative Position Error (RPE) over a $10\,\mathrm{m}$ distance (see Table II). Compared to the kinematic-inertial estimator, our algorithm reduces the RPE by $15\,\%$ to $55\,\%$ and yaw error by $42\,\%$ to $85\,\%$, as visual features tracked over many frames constrain pose drift.

D. Oil Rig Dataset

A second outdoor dataset was collected at an industrial firefighter training facility in Moreton-In-Marsh, UK (Fig. 1). The facility closely matches the locations where ANYmal is likely to be deployed in future.

This $110\,\mathrm{m}$ ($22\,\mathrm{min}$) long dataset involves the ANYmal robot trotting through the facility, climbing over a slab, and

TABLE II
MEAN (AND STANDARD DEVIATION) PERFORMANCE ON THE KEBLE COLLEGE AND OIL RIG DATASETS.

	RPE $\mu(\sigma)$ [m]		Yaw Error $\mu(\sigma)$ [deg]	
Dataset	**TSIF** [3]	**VILENS**	**TSIF** [3]	**VILENS**
Keble 1	0.53 (0.21)	**0.30** (0.12)	6.64 (2.23)	**0.99** (0.80)
Keble 2	0.51 (0.10)	**0.23** (0.10)	5.72 (0.94)	**1.47** (1.07)
Keble 3	0.67 (0.10)	**0.52** (0.15)	6.68 (0.80)	**3.86** (1.90)
Keble 4	0.47 (0.11)	**0.40** (0.10)	3.32 (1.15)	**1.13** (1.46)
Oil Rig	0.44 (0.37)	**0.41** (0.18)	4.89 (3.38)	**3.68** (4.10)

TABLE III
MEAN (AND STANDARD DEVIATION) PERFORMANCE OF TSIF, VILENS AS VINS SYSTEM, AND VILENS BEFORE VINS FAILURE ($\sim$70 s)

	RPE $\mu(\sigma)$ [m]			Yaw Error $\mu(\sigma)$ [deg]		
Dataset	**TSIF**	**VINS**	**VILENS**	**TSIF**	**VINS**	**VILENS**
Keble 1	0.30 (0.06)	0.36 (0.12)	**0.25** (0.13)	4.28 (1.07)	**0.74** (0.61)	0.75 (0.56)
Oil Rig	1.09 (0.09)	5.33 (0.53)	**0.34** (0.12)	10.38 (0.75)	5.03 (3.21)	**1.21** (0.90)

walking up a staircase into a smoke-blackened room. Challenging situations include: featureless areas, stationary periods with intermittent motion, dynamic obstacles occluding large portions of the image, non-flat terrain traversal, and foot slip caused by a combination of mud, oil, and water on the ground (Fig. 11).

Figure 10 shows the estimated trajectory from VILENS, compared to TSIF [3] and ground truth. The Absolute Translation Error (ATE) for VILENS is 76 % lower compared to the TSIF (1.65 m and 6.88 m, respectively).

Looking at the relative performance over 10 m, the error reduction is 7 % for RPE and 25 % for yaw (Table III). This suggests that the Oil Rig dataset is more challenging than Keble, since the accuracy at small scale is closer to the TSIF (see Section VII).

Note that performance evaluation against the VINS algorithms mentioned in Section VI-A was not possible because they either failed to initialize due to the lack of motion or diverged after a short period. This is also confirmed in Table III, where we evaluated the performance of VILENS as standalone VINS system against TSIF and full VILENS. Without Leg Odometry factors, VILENS fails for all the datasets except the first 70 s of Keble 1 and Oil Rig, where it performs worse or same.

E. Timing

A summary of the key computation times in the proposed algorithm are shown in Table IV. Since the kinematic-inertial filter, the image processing, and the optimization are run in separate threads, the system is capable of outputting a high-frequency kinematic-inertial state estimate (for control purposes) at 400 Hz, and an optimized estimate incorporating visual features at approximately 10 Hz.

VII. DISCUSSION

In the previous section, we have demonstrated that VILENS outperforms kinematic-inertial and visual-inertial methods for

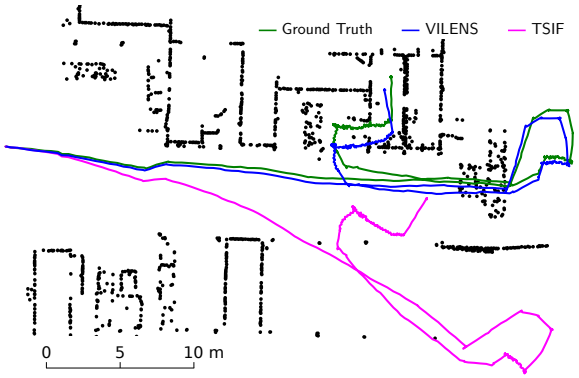


Fig. 10. Top view of the estimated trajectories of VILENS (blue), TSIF (magenta), and ground truth (green) on the Oil Rig Dataset. The map is not used in the algorithm and is shown for illustrative purposes only.

TABLE IV
MEAN (AND STANDARD DEVIATION) PROCESSING TIME FOR COMPONENTS OF THE VILENS SYSTEM, ON THE OIL RIG DATASET.

Thread	**Module**	$\mu(\sigma)$ [ms]
Optimization	Factor Creation	10.80 (4.50)
	Optimization	10.05 (7.69)
	Marginalization	0.82 (0.97)
	Total	**21.67** (**13.12**)
Image Proc	Image Equalization	0.87 (0.51)
	Feature Tracking	2.04 (0.98)
	Outlier Rejection	1.74 (1.87)
	Feature Detection	5.33 (0.91)
	Total	**9.99** (**4.27**)



Fig. 11. Notable situations within the Oil Rig dataset. *Top-Left:* VILENS outperforms TSIF where there are many visual features. *Top-Right:* VILENS and TSIF perform similarly where there is no structure in front of the robot. *Bottom:* Moving objects and reflections are robustly handled by VILENS.

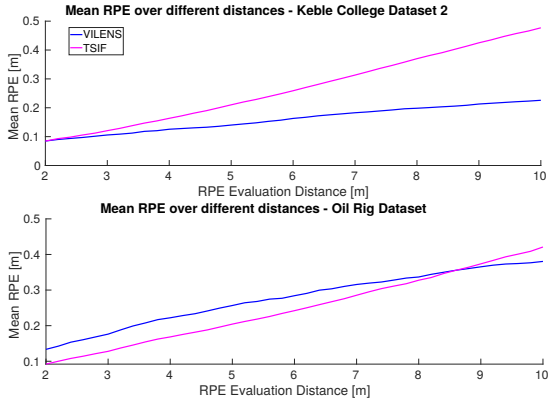


Fig. 12. Mean RPE for VILENS and TSIF at different distance scales. *Top:* Keble College Dataset. *Bottom:* Oil Rig Dataset. The Oil Rig dataset is more challenging and therefore VILENS outperforms TSIF at larger distance ($8.5\,\mathrm{m}$) than for Keble.

all the datasets. However, from Table II we can see that the gap between VILENS and TSIF is not uniform across the datasets.

An in-depth analysis of performance was limited by the accuracy and frequency of our LIDAR ground truth. Nonetheless, we see that the feature quality has an influence of the drift rate over small scales.

In Fig. 12 we compare the drift rate of VILENS and TSIF over different distance scales for the Keble College and the Oil Rig datasets. The latter experiment is more challenging from the perspective of exteroception and as a result the scale at which VILENS starts to outperform TSIF is larger. This is due to the tracking of poor quality features in the conditions highlighted in Fig. 11.

In future work, we are motivated to improve performance in these challenging scenarios by exploring the use of redundant or wider field-of-view cameras as well as different methods of incorporating the kinematics directly into the factor graph. Additionally, as part of an ongoing project, we intend to test performance in soft and compliant surface materials such as sand, mud and gravel where we envisage the visual part of the estimator predominating during sinking, sliding and slipping.

VIII. Conclusion

In this paper, we have presented VILENS (Visual Inertial LEgged Navigation System), a robust state estimation method for legged robots based on factor graphs, which incorporates kinematic, inertial and visual information. This method outperforms the robot's kinematic-inertial estimator and robustly estimates the robot trajectory in challenging scenarios, including textureless areas, moving occludants, reflections and slippery ground. Under the same conditions, current state-of-the-art visual-inertial algorithms diverge rapidly.

References

[1] M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. David Remy, and R. Siegwart, "State Estimation for Legged Robots - Consistent Fusion of Leg Kinematics and IMU," in *Robotics: Science and Systems VIII*, 2012.

[2] M. Camurri, M. Fallon, S. Bazeille, A. Radulescu, V. Barasuol, D. G. Caldwell, and C. Semini, "Probabilistic Contact Estimation and Impact Detection for State Estimation of Quadruped Robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1023–1030, 2017.

[3] M. Bloesch, M. Burri, H. Sommer, R. Siegwart, and M. Hutter, "The Two-State Implicit Filter - Recursive Estimation for Mobile Robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 573–580, 2017.

[4] P. Fankhauser, M. Bjelonic, C. Dario Bellicoso, T. Miki, and M. Hutter, "Robust Rough-Terrain Locomotion with a Quadrupedal Robot," in *IEEE International Conference on Robotics and Automation*, 2018.

[5] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. A. Hoepflinger, "ANYmal - A Highly Mobile and Dynamic Quadrupedal Robot," in *IEEE International Conference on Intelligent Robots and Systems*, 2016, pp. 38–44.

[6] J. Ma, M. Bajracharya, S. Susca, L. Matthies, and M. Malchano, "Real-time pose estimation of a dynamic quadruped in GPS-denied environments for 24-hour operation," *International Journal of Robotics Research*, vol. 35, no. 6, pp. 631–653, 2016.

[7] S. Nobili, M. Camurri, V. Barasuol, M. Focchi, D. Caldwell, C. Semini, and M. Fallon, "Heterogeneous Sensor Fusion for Accurate State Estimation of Dynamic Legged Robots," in *Robotics: Science and Systems*, 2017.

[8] R. Hartley, M. G. Jadidi, L. Gan, J.-K. Huang, J. W. Grizzle, and R. M. Eustice, "Hybrid Contact Preintegration for Visual-Inertial-Contact State Estimation within Factor Graphs," in *IEEE International Conference on Intelligent Robots and Systems*, 2018.

[9] P. Fankhauser and M. Hutter, "A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation," in *Robot Operating System (ROS) – The Complete Reference (Volume 1)*. Springer, 2017, ch. 5.

[10] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

[11] O. A. Villarreal Magana, V. Barasuol, M. Camurri, L. Franceschi, M. Focchi, M. Pontil, D. G. Caldwell, and C. Semini, "Fast and Continuous Foothold Adaptation for Dynamic Locomotion through CNNs," *IEEE Robotics and Automation Letters*, 2019.

[12] J. Delmerico and D. Scaramuzza, "A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots," *IEEE International Conference on Robotics and Automation*, 2018.

[13] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[14] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[15] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust Visual Inertial Odometry Using a Direct EKF-Based Approach," in *IEEE International Conference on Intelligent Robots and Systems*, 2015.

[16] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," in *IEEE International Conference on Robotics and Automation*, 2017.

[17] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-Manifold Preintegration for Real-Time Visual-Inertial Odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[18] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "ISAM2: Incremental smoothing and mapping using the Bayes tree," *International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[19] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO : Semidirect Visual Odometry for Monocular and Multicamera Systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[20] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback," *International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[21] L. Carlone, Z. Kira, C. Beall, V. Indelman, and F. Dellaert, "Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors," in *IEEE International Conference on Robotics and Automation*, 2014.

[22] S. Nobili, R. Scona, M. Caravagna, and M. Fallon, "Overlap-based ICP tuning for robust localization of a humanoid robot," in *IEEE International Conference on Robotics and Automation*, 2017.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	Robust Legged Robot State Estimation Using Factor Graph Optimization
Publication Status	Published
Publication Details	D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robotics and Automation Letters*, pp.4507-4514, 2019.

Student Confirmation

Student Name:	David Wisth		
Contribution to the Paper	My contributions to the paper were: • Developed the idea behind the paper • Collected and processed the data • Performed the experiments with co-authors • Wrote the paper with co-authors		
Signature		Date	30/09/2021

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Maurice Fallon		
Supervisor comments This is an accurate desciption		
Signature 	Date	4/10/2021

This completed form should be included in the thesis, at the end of the relevant chapter.

4.2 Preintegrated Velocity Bias Estimation to Overcome Contact Nonlinearities in Legged Robot Odometry

The previous article presented a method of incorporating kinematics as a relative pose constraint, using the output of an external kinematic-inertial (KI) filter. This is very effective when the leg odometry module has a low drift rate with Gaussian noise. This typically only occurs when the robot uses slow, statically stable gaits.

In contrast, when the robot performs more dynamic gaits, especially over soft or deformable terrain, the drift rate of the KI filter can significantly increase. Additionally, the noise usually is non-Gaussian, with the state estimate often drifting in the positive z-direction (i.e., upwards) and in yaw. This non-Gaussian error is not modelled by the relative pose constraints in the previous article. Thus, a biased leg odometry input can lead to a biased optimisation output.

The main reason for this drift is the assumption of static foot contact points. Any slippage or deformation violates this assumption and is a major cause of drift (Fahmi et al. [119]). Contact detection methods can help to reject sporadic slippage or deformation events when they are significant enough to trigger relevant thresholds. This approach ignores smaller, but more frequent slippage or deformation, which fall under the preset thresholds, yet can still cause significant long-term drift. Existing approaches also ignore other sources of systematic error, such as inaccurate calibration or non-zero contact point size.

To overcome this problem, the following paper presents a novel *preintegrated twist factor* that takes an external twist measurement (e.g., from an external KI filter) to provide relative pose constraints. Importantly, this factor also performs online bias estimation of the twist measurements, inspired by how VIO systems estimate IMU biases online [20].

This new factor allows biased leg odometry inputs to be used within the factor graph to improve performance, rather than degrading it. Leg odometry is critical for improving system robustness, especially when exteroceptive sensors (e.g., cameras or

lidar) fail, become obstructed, or are in degraded environments. The full derivation of the preintegrated twist factors is provided in Appendix A.

This paper was presented at the IEEE International Conference on Robotics and Automation (2020) [13]. A video summary of this publication can be found at: `https://youtu.be/w1Sx6dIqgQo`. The associated conference presentation can be found at: `https://youtu.be/AqfZ0RTRBBM`.

Preintegrated Velocity Bias Estimation to Overcome Contact Nonlinearities in Legged Robot Odometry

David Wisth, Marco Camurri, Maurice Fallon

***Abstract*— In this paper, we present a novel factor graph formulation to estimate the pose and velocity of a quadruped robot on slippery and deformable terrain. The factor graph introduces a preintegrated velocity factor that incorporates velocity inputs from leg odometry and also estimates related biases. From our experimentation we have seen that it is difficult to model uncertainties at the contact point such as slip or deforming terrain, as well as leg flexibility. To accommodate for these effects and to minimize leg odometry drift, we extend the robot's state vector with a bias term for this preintegrated velocity factor. The bias term can be accurately estimated thanks to the tight fusion of the preintegrated velocity factor with stereo vision and IMU factors, without which it would be unobservable. The system has been validated on several scenarios that involve dynamic motions of the ANYmal robot on loose rocks, slopes and muddy ground. We demonstrate a 26% improvement of relative pose error compared to our previous work and 52% compared to a state-of-the-art proprioceptive state estimator.**

I. Introduction

The increased maturity of legged robotics has been demonstrated by the initial industrial deployments of quadruped robots, as well as impressive results achieved by academic research. State estimation plays a key role in field deployment of legged machines: without an accurate estimate of its location and velocity, the robot cannot build a useful representation of its environment or plan/execute trajectories to reach goal positions.

Most legged robots are equipped with a high frequency ($>$250 Hz) proprioceptive state estimator for control and local mapping purposes. These are typically implemented as nonlinear filters fusing high frequency signals such as kinematics and IMU [1]. In ideal conditions (i.e. high friction, rigid terrain), these estimators have a limited (yet unavoidable) drift that is acceptable for local mapping and control.

However, deformable terrains, leg flexibility and slippage can degrade the estimation performance up to a point where local terrain reconstruction is unusable and multi-step trajectories cannot be executed, even over short ranges. This problem is more evident when a robot is moving dynamically.

Recent works have attempted to improve kinematic-inertial estimation accuracy by detecting unstable contacts and reducing their influence on the overall estimation [2], [3]. Alternatively, some works have focused on incorporating additional exteroceptive sensing modalities into the estimator to help reduce the pose error [4].

The authors are with the Oxford Robotics Institute, Department of Engineering Science, University of Oxford, UK.
{davidw, mcamurri, mfallon}@robots.ox.ac.uk



Fig. 1. ANYmal trotting over a field of small rocks at the Swiss Military Rescue Centre in Wangen an der Aare (Switzerland). The ground truth was collected using a Leica TS16 laser tracker (visible in the background). Video: http://youtu.be/w1Sx6dIqgQo

These approaches model the contact locations as being fixed and affected only by Gaussian noise. Both assumptions fail in conditions such as non-rigid terrain, kinematic chain flexibility, and foot slippage.

A. Motivation

Our work is motivated by the observation that there is an approximately constant velocity bias from the kinematic-inertial state estimator on the ANYmal robot during dynamic locomotion. An example is shown in Fig. 2, where the robot's estimated altitude grows linearly as the robot moves. We attribute this behavior to the compression of legs and the ground during the contact events.

One approach would be to further model the dynamic properties of the robot [5] or the terrain directly within the estimator. However, this is likely to be robot specific and terrain dependent: improving performance in one situation but degrading it elsewhere. Instead, we propose to extend the state of the estimator with a velocity bias term which is estimated using vision and then to reject all such effects.

Inspired by the IMU bias estimation and preintegration methods from [6], we propose a novel leg odometry factor that performs online velocity preintegration and bias estimation to compensate for characteristic drift in leg odometry.

This factor was implemented as a concurrent thread within our VILENS framework [7], a visual-inertial-legged estimator which uses GTSAM for optimization [8]. Thanks to forward propagation, the thread can output the best pose and velocity estimates at 400 Hz directly, or just update the bias terms of the estimator running inside the robot's control loop. The optimized estimate is available from the optimizer thread at at 30 Hz for effective local mapping.

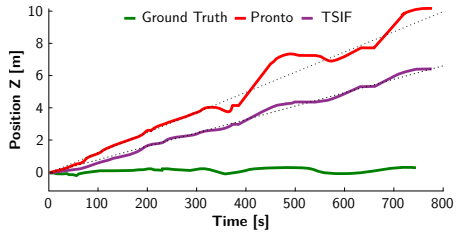


Fig. 2. Comparison between estimated robot altitude by Pronto [2] (blue) and TSIF [9] (magenta) kinematic-inertial state estimators, against ground truth (green) on the SMR1 dataset. Despite local fluctuations, the drift has a characteristic linear growth.

B. Contribution

This paper builds upon state-of-the-art methods for online IMU bias estimation [6] and the authors' previous work [7] to improve state estimation performance of legged robots in a variety of difficult scenarios where kinematic-inertial estimates would drift significantly. Compared to previous research, we present the following contributions:

- We present a novel factor graph approach that tightly fuses leg odometry as velocity constraints (as opposed to position constraints), with stereo vision and IMU measurements;
- We present the first visual-inertial-legged odometry solution that explicitly accounts for error in leg odometry (that can be caused by terrain/leg deformation and slippage) by extending the state with a velocity bias term;
- We show that estimating leg odometry error can reduce RPE by 26% in extensive outdoor experiments on muddy ground, slopes, and rockbeds with the ANYmal robot (Fig. 1).

The remainder of this article is presented as follows: in Section II we review the literature on mobile state estimation with a focus on challenging, outdoor conditions; Section III formally defines the problem addressed by the paper and provides the required mathematical background; Section IV describes the factors used in our proposed formulation; Section V presents the implementation details of our physical system; Section VI presents the experimental results and their interpretation; Section VII concludes with final remarks.

II. Related Work

In legged robotics, slippage and/or deformation have been typically addressed by assuming the contact location of a stance foot is always static throughout the stance period (yet affected by Gaussian noise). Thus, the main focus has been on detecting and ignoring the feet that are not in fixed contact with the ground. These methods would typically perform filtering using only proprioceptive sensing, with a few exceptions.

Bloesch et al. [1] proposed an Unscented Kalman Filter design that fuses IMU and differential kinematics. The approach used a threshold on the Mahalanobis distance of the filter innovation to infer outliers which were then ignored.

Sensor	Model	Hz	Specs
IMU	Xsens MTi-100	400	*Init Bias:* $0.2\,^\circ/\mathrm{s}$ | $5\,\mathrm{mg}$ *Bias Stab:* $10\,^\circ/\mathrm{h}$ | $15\,\mathrm{mg}$
Stereo Camera	RealSense D435i	30	*Resolution*: 848×480 px *FoV:* $91.2^\circ \times 65.5^\circ$ *Imager:* IR global shutter
Encoder	ANYdrive	400	*Resolution:* $<0.025^\circ$
Torque	ANYdrive	400	*Resolution:* $<0.1\,\mathrm{N\,m}$

TABLE I

Ma et al. [10] proposed an Extended Kalman Filter (EKF) design that was Visual Odometry (VO) driven. They incorporated kinematics only when VO failed or a simple heuristic criteria was met (e.g., when roll or pitch are greater than 45° slippage was assumed and leg odometry ignored). Using high-grade sensors, they were able to achieve 1% error over several kilometers of experiments.

Camurri et al. [2] proposed an EKF fusing IMU and differential kinematics similar to [1]. Instead of the Mahalanobis distance on the filter innovation, they developed probabilistic contact and impact detectors. The contact detector learns the optimal force threshold to detect a foot in contact for a specific gait, while the impact detector adapts the measurement covariance to reject unreliable measurements. We used this approach to fuse each leg's kinematic measurements into a single velocity measurement for our proposed factor graph method.

Recently, Jenelten et al. [3] presented a probabilistic contact and slip detector which used a Hidden Markov Model for the ANYmal quadruped robot. Using differential kinematics, the authors were able to successfully detect slippage events and robustify locomotion on slippery surfaces. However, they did not address pose estimate drift.

III. Problem Statement

Our quadruped robot has 12 active Degrees-of-Freedom (DoF) and is equipped with a stereo camera, an IMU, joint encoders and torque sensors (see Table I for the specifications). We aim to estimate the history of the robot's base link pose and its velocity (linear and angular) over time. In contrast to previous works, we propose to estimate velocity biases (in addition to IMU biases) to compensate for leg odometry drift, as detailed in the following section.

The relevant reference frames are specified in Fig. 3 and include: the left camera frame $\mathtt{C}$, the IMU frame $\mathtt{I}$, the fixed-world frame $\mathtt{W}$, and the base frame $\mathtt{B}$. When a foot is in contact with the ground, a contact frame $\mathtt{K}$ is also defined.

Unless otherwise specified, position ${}_{\mathtt{W}}\mathbf{p}_{\mathtt{WB}}$ and orientation $\mathbf{R}_{\mathtt{WB}}$ of the base are expressed in world coordinates, velocities of the base ${}_{\mathtt{B}}\mathbf{v}_{\mathtt{WB}}, {}_{\mathtt{B}}\boldsymbol{\omega}_{\mathtt{WB}}$ are in base coordinates (see [11]), IMU biases ${}_{\mathtt{I}}\mathbf{b}^{g}$, ${}_{\mathtt{I}}\mathbf{b}^{a}$ are expressed in the IMU frame, and the velocity biases ${}_{\mathtt{B}}\mathbf{b}^{\omega}, {}_{\mathtt{B}}\mathbf{b}^{v}$ are expressed in the base frame.

A. State Definition

The robot state at time t_i is defined as follows:

$$\boldsymbol{x}_i \triangleq [\mathbf{R}_i, \mathbf{p}_i, \mathbf{v}_i, \mathbf{b}_i] \tag{1}$$

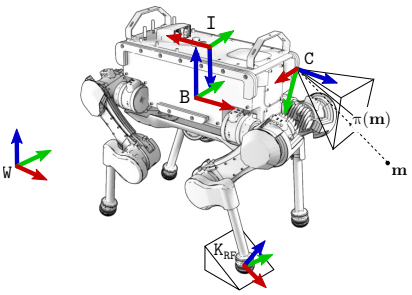


Fig. 3. Reference frames conventions. The world frame W is fixed to earth, while the base frame B, the camera's optical frame C, and the IMU frame, I are rigidly attached to the robot's chassis. When a foot touches the ground (e.g., the Right Front, RF), a contact frame K (perpendicular to the ground and parallel to W's y-axis) is defined. The projection of a landmark point $\mathbf{m}$ onto the image plane is $\pi(\mathbf{m})$.

where: $\mathbf{R}_i \in \mathrm{SO}(3)$ is the orientation, $\mathbf{p}_i \in \mathbb{R}^3$ is the position, $\mathbf{v}_i \in \mathbb{R}^3$ is the linear velocity. The bias vector $\mathbf{b}_i$ is composed as follows:

$$\mathbf{b}_i = [\mathbf{b}_i^g \;\; \mathbf{b}_i^a, \;\; \mathbf{b}_i^\omega, \;\; \mathbf{b}_i^v] \in \mathbb{R}^{12} \tag{2}$$

where the first two elements are the usual IMU gyro and accelerometer biases, and the last two $[\mathbf{b}_i^\omega, \;\; \mathbf{b}_i^v] = \mathbf{b}_i^Q$ are our proposed angular/linear velocity biases from leg odometry.

This builds upon the formulation from our previous work [7], by incorporating a preintegrated velocity factor (as opposed to a relative pose factor where the integration was operated by an external filter).

In addition to the robot state, we estimate the position of all observed visual landmarks $\mathbf{m}_\ell$. The objective of our estimation problem is then the union of all the robot states and landmarks visible up to the current time t_k:

$$\mathcal{X}_k \triangleq \bigcup_{\forall i \in \mathsf{K}_k} \left[\{\boldsymbol{x}_i\}, \bigcup_{\forall \ell \in \mathsf{M}_i} \{\mathbf{m}_\ell\} \right] \tag{3}$$

where $\mathsf{K}_k, \mathsf{M}_i$ are the lists of all the keyframe indices up to time t_k and all landmark indices visible at time t_i, respectively.

B. Measurements Definition

For each new stereo camera frame $\mathcal{C}_i$, collected at time t_i, we receive a number of IMU measurements $\mathcal{I}_{ij}$ collected between t_i and t_j. We also define $\mathcal{V}_{ij}$ as the angular and linear velocity measurements from an external source of leg odometry (Pronto or TSIF). This source would account for the fusion of multiple legs in contact into one velocity measurement per joint state measurement. The set of all measurements up to time t_k is therefore defined as:

$$\mathcal{Z}_k \triangleq \bigcup_{\forall i \in \mathsf{K}_k} \{\mathcal{C}_i, \mathcal{I}_{ij}, \mathcal{V}_{ij}\} \tag{4}$$

C. Maximum-a-Posteriori Estimation

We wish to maximize the likelihood of the measurements $\mathcal{Z}_k$ given the history of states $\mathcal{X}_k$:

$$\mathcal{X}_k^* = \arg\max_{\mathcal{X}_k} p(\mathcal{X}_k|\mathcal{Z}_k) \propto p(\mathcal{X}_0)p(\mathcal{Z}_k|\mathcal{X}_k) \tag{5}$$

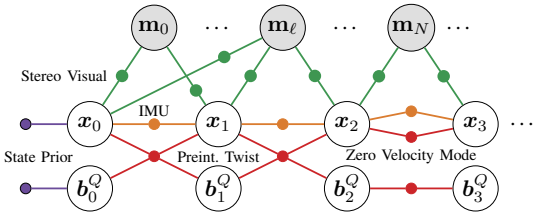


Fig. 4. VILENS factor graph structure, showing initial prior, visual, IMU, and preintegrated velocity factors. When a zero velocity state is detected (e.g. between x_2 and x_3) then the velocity bias is not used and is kept constant (we assume the bias is only present when the robot is moving).

As the measurements are formulated as conditionally independent and corrupted by white Gaussian noise, Eq. (5) can be formulated as a least squares minimization problem:

$$\mathcal{X}_k^* = \arg\min_{\mathcal{X}_k} \|\mathbf{r}_0\|^2_{\Sigma_0} + \sum_{i \in \mathsf{K}_k} \|\mathbf{r}_{\mathcal{I}_{ij}}\|^2_{\Sigma_{\mathcal{I}_{ij}}} + \sum_{i \in \mathsf{K}_k} \|\mathbf{r}_{\mathcal{V}_{ij}}\|^2_{\Sigma_{\mathcal{V}_{ij}}} + \sum_{i \in \mathsf{K}_k} \|\mathbf{r}_{\mathbf{b}_{ij}}\|^2_{\Sigma_{\mathbf{b}}} + \sum_{i \in \mathsf{K}_k} \sum_{\ell \in \mathsf{M}_i} \|\mathbf{r}_{i,\mathbf{m}_\ell}\|^2_{\Sigma_{\mathbf{m}}} \tag{6}$$

where each term is the residual associated to a factor type, weighted by its covariance matrix; specifically the residuals are: state prior, IMU, velocity, biases and landmarks.

IV. Factor Graph Formulation

In the following sections we describe the measurements, residuals and covariances of the factors which compose the factor graph shown in Fig. 4. For convenience, we summarize the IMU factors from [6] in Section IV-A; our novel velocity factor is detailed in Section IV-B; Sections IV-C and IV-D describe the bias and stereo visual residuals, which are adapted from [6], [7] to include the velocity bias term and stereo cameras, respectively.

A. Preintegrated IMU Factors

In the standard manner, the IMU measurements are preintegrated to constrain the pose and velocity between two consecutive nodes of the graph, and provide high frequency state updates between them. This uses a residual of the form:

$$\mathbf{r}_{\mathcal{I}_{ij}} = \left[\mathbf{r}^\mathsf{T}_{\Delta\mathbf{R}_{ij}}, \mathbf{r}^\mathsf{T}_{\Delta\mathbf{v}_{ij}}, \mathbf{r}^\mathsf{T}_{\Delta\mathbf{p}_{ij}} \right] \tag{7}$$

where $\mathcal{I}_{ij}$ are the IMU measurements between t_i and t_j. The individual elements of the residual are defined as:

$$\mathbf{r}_{\Delta\mathbf{R}_{ij}} = \mathrm{Log}\left(\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^g) \right) \mathbf{R}_i^\mathsf{T} \mathbf{R}_j \tag{8}$$

$$\mathbf{r}_{\Delta\mathbf{v}_{ij}} = \mathbf{R}_i^\mathsf{T} \left(\mathbf{v}_j - \mathbf{v}_i - \mathbf{g}\Delta t_{ij} \right) - \Delta\tilde{\mathbf{v}}_{ij}(\mathbf{b}_i^g, \mathbf{b}_i^a) \tag{9}$$

$$\mathbf{r}_{\Delta\mathbf{p}_{ij}} = \mathbf{R}_i^\mathsf{T} \left(\mathbf{p}_j - \mathbf{p}_i - \mathbf{v}_i\Delta t_{ij} - \frac{1}{2}\mathbf{g}\Delta t_{ij}^2 \right) - \Delta\tilde{\mathbf{p}}_{ij}(\mathbf{b}_i^g, \mathbf{b}_i^a) \tag{10}$$

for the definition of the preintegrated IMU measurements $\Delta\tilde{\mathbf{R}}_{ij}, \Delta\tilde{\mathbf{p}}_{ij}, \Delta\tilde{\mathbf{v}}_{ij}$, the noise terms $\delta\boldsymbol{\phi}, \delta\mathbf{v}, \delta\mathbf{p}$, and the covariance matrix $\Sigma_{\mathcal{I}_{ij}}$, the reader is invited to consult [6].

B. Preintegrated Velocity Factors

1) Leg Odometry: When a leg is in rigid, non-slipping contact with the ground, the robot's linear velocity can be computed from the foot velocity and position in base frame:

$$ {}_{\mathtt{B}}\mathbf{v}_{\mathtt{WB}} = -{}_{\mathtt{B}}\mathbf{v}_{\mathtt{BK}} - {}_{\mathtt{B}}\boldsymbol{\omega}_{\mathtt{WB}} \times {}_{\mathtt{B}}\mathbf{p}_{\mathtt{BK}} \tag{11} $$

From the sensed joint positions and velocities $\tilde{\mathbf{q}}, \dot{\tilde{\mathbf{q}}}$ and noise $\boldsymbol{\eta}^q, \boldsymbol{\eta}^{\dot{q}}$ we can rewrite Eq. (11) as a linear velocity measurement [1]:

$$ \tilde{\mathbf{v}} = -J(\tilde{\mathbf{q}} - \boldsymbol{\eta}^q) \cdot (\dot{\tilde{\mathbf{q}}} - \boldsymbol{\eta}^{\dot{q}}) - \boldsymbol{\omega} \times \mathrm{f}(\tilde{\mathbf{q}} - \boldsymbol{\eta}^q) \tag{12} $$

where $\mathrm{f}(\cdot)$ and $J(\cdot)$ are the forward kinematics function and its Jacobian, respectively.

Eq. (12) is valid only when the corresponding leg is in contact with the ground. However, this happens intermittently while the robot moves. Since multiple legs can be in contact simultaneously, measurement fusion is necessary. To do this we take advantage of the contact detection and sensor fusion features of the EKF filter in [2] and use it as an independent source of unified velocity measurements $\tilde{\mathbf{v}}, \tilde{\boldsymbol{\omega}}$.

2) Velocity Bias: On slippery/deformable terrains, the constraint from Eq. (11) might not be respected, leading to leg odometry drift and inconsistency with visual odometry. In our experience this drift is constant and gait dependent.

For these reasons, we relax Eq. (11) by adding a slowly varying bias term $\mathbf{b}^v$ to Eq. (12). As in [1], we also collect all the effects of encoder noise into a single term, leading to:

$$ \tilde{\mathbf{v}} = -J(\tilde{\mathbf{q}})\dot{\tilde{\mathbf{q}}} - \boldsymbol{\omega} \times \mathrm{f}(\tilde{\mathbf{q}}) + \mathbf{b}^v + \boldsymbol{\eta}^v \tag{13} $$

$$ \tilde{\boldsymbol{\omega}} = \boldsymbol{\omega} + \mathbf{b}^\omega + \boldsymbol{\eta}^\omega \tag{14} $$

where the parameters for $\boldsymbol{\eta}^v, \boldsymbol{\eta}^\omega$ are provided by the source of velocity measurements.

3) Preintegrated Measurements: In the following, we derive the the preintegrated position and noise only. For the respective rotational quantities $\Delta\tilde{\mathbf{R}}$ and $\delta\boldsymbol{\phi}$, we refer to [6], as they have the same form as for IMU measurements.

Assuming constant velocity between t_i and t_j, we can iteratively calculate the position at time t_j as:

$$ \mathbf{p}_j = \mathbf{p}_i + \sum_{k=i}^{j-1} [\mathbf{R}_k(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v - \boldsymbol{\eta}_k^v)\Delta t] \tag{15} $$

From Eq. (15) a relative measurement can be obtained:

$$ \Delta\mathbf{p}_{ij} = \mathbf{R}_i^\mathsf{T}(\mathbf{p}_j - \mathbf{p}_i) = \sum_{k=i}^{j-1} [\Delta\mathbf{R}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_k^v - \boldsymbol{\eta}_k^v)\Delta t)] \tag{16} $$

With the substitution $\Delta\mathbf{R}_{ik} = \Delta\tilde{\mathbf{R}}_{ik}\mathrm{Exp}\left(-\delta\boldsymbol{\phi}_{ik}\right)$ to include the preintegrated rotation measurement, and the approximation $\mathrm{Exp}\left(\boldsymbol{\phi}\right) \simeq \mathbf{I} + \boldsymbol{\phi}^\wedge$, Eq. (16) becomes:

$$ \Delta\mathbf{p}_{ij} \simeq \sum_{k=i}^{j-1} \left[\Delta\tilde{\mathbf{R}}_{ik}(I - \delta\boldsymbol{\phi}_{ik}^\wedge)(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v - \boldsymbol{\eta}_k^v)\Delta t\right] \tag{17} $$

By separating the measurement and noise components of Eq. (17) and ignoring the higher order terms, we can define the preintegrated position *measurement* $\Delta\tilde{\mathbf{p}}$ and *noise* $\delta\mathbf{p}$ as:

$$ \Delta\tilde{\mathbf{p}}_{ij} \triangleq \sum_{k=i}^{j-1} \left[\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v)\Delta t\right] \tag{18} $$

$$ \delta\mathbf{p}_{ij} \triangleq \sum_{k=i}^{j-1} \left[\Delta\tilde{\mathbf{R}}_{ik}\boldsymbol{\eta}_k^v\Delta t - \Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v)^\wedge\delta\boldsymbol{\phi}_{ij}\Delta t\right] \tag{19} $$

Note that both quantities still depend on the bias states $\mathbf{b}^Q = [\mathbf{b}^\omega, \mathbf{b}^v]$; when these change, we would like to avoid the recomputation of Eq. (18). Given a small change $\delta\mathbf{b}^Q$ such that $\mathbf{b}^Q = \bar{\mathbf{b}}^Q + \delta\mathbf{b}^Q$, we approximate the measurement as:

$$ \Delta\tilde{\mathbf{p}}_{ij}(\mathbf{b}^Q) \simeq \Delta\tilde{\mathbf{p}}_{ij}(\bar{\mathbf{b}}^Q) + \frac{\partial\Delta\mathbf{p}_{ij}}{\partial\mathbf{b}^\omega}\delta\mathbf{b}^\omega + \frac{\partial\Delta\mathbf{p}_{ij}}{\partial\mathbf{b}^v}\delta\mathbf{b}^v \tag{20} $$

4) Residuals: the velocity residual can be expressed as:

$$ \mathbf{r}_{\mathcal{V}_{ij}} = \left[\mathbf{r}_{\Delta\mathbf{R}_{ij}}^\mathsf{T}, \mathbf{r}_{\Delta\mathbf{p}_{ij}}^\mathsf{T}\right] \tag{21} $$

where $\mathbf{r}_{\Delta\mathbf{R}_{ij}}$ has the same form as Eq. (8) and $\mathbf{r}_{\Delta\mathbf{p}_{ij}}$ is:

$$ \mathbf{r}_{\Delta\mathbf{p}_{ij}} = \mathbf{R}_i^\mathsf{T}(\mathbf{p}_j - \mathbf{p}_i) - \Delta\tilde{\mathbf{p}}_{ij}(\mathbf{b}_i^\omega, \mathbf{b}_i^v) \tag{22} $$

5) Covariance: After simple manipulation of Eq. 19, the covariance of the residual $\mathbf{r}_{\Delta\mathbf{p}_{ij}}$ can be expressed as a linear combination of the preintegrated and current sensor noise:

$$ \Sigma_{\mathcal{V}_{ik+1}} = A\,\Sigma_{\mathcal{V}_{ik}}\,A^\mathsf{T} + B\,\Sigma_{\mathcal{V}}^\eta\,B^\mathsf{T} \tag{23} $$

$\Sigma_{\mathcal{V}_{ik}}$ evolves over time while $\Sigma_{\mathcal{V}}^\eta$ is fixed and depends sensor specifications. The multiplicative terms are:

$$ A = \begin{bmatrix} \Delta\tilde{\mathbf{R}}_{ik}^\mathsf{T} & 0 \\ -\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}^v)^\wedge\Delta t & \mathbf{I} \end{bmatrix}, B = \begin{bmatrix} J_r^k\Delta t & 0 \\ 0 & \Delta\tilde{\mathbf{R}}_{ik}\Delta t \end{bmatrix} \tag{24} $$

where J_r is the right Jacobian of $\mathrm{SO}(3)$ and the other terms are manipulations of $\delta\boldsymbol{\phi}$ and $\delta\mathbf{p}$ from [6] and Eq. (19).

C. Bias Residuals

The biases from Eq. (2) are intended to change slowly and are therefore modeled as a Gaussian random walk. The residual term for the cost function is therefore:

$$ \begin{aligned} \|\mathbf{r}_{\mathbf{b}_{ij}}\|_{\Sigma_\mathbf{b}}^2 \triangleq\ & \|\mathbf{b}_j^g - \mathbf{b}_i^g\|_{\Sigma_{\mathbf{b}^g}}^2 + \|\mathbf{b}_j^a - \mathbf{b}_i^a\|_{\Sigma_{\mathbf{b}^a}}^2 + \\ & + \|\mathbf{b}_j^\omega - \mathbf{b}_i^\omega\|_{\Sigma_{\mathbf{b}^\omega}}^2 + \|\mathbf{b}_j^v - \mathbf{b}_i^v\|_{\Sigma_{\mathbf{b}^v}}^2 \end{aligned} \tag{25} $$

where the covariance matrices are determined by the expected rate of change of these quantities, depending on the IMU specifications or the drift rate of the leg odometry.

D. Stereo Visual Factors

Given a stereo pair of rectified images, the stereo visual odometry residual is the difference between the measured landmark pixel locations $(u^L, v), (u^R, v)$, and the reprojection of the estimated landmark location into image coordinates, $(\pi_u^L, \pi_v), (\pi_u^R, \pi_v)$ using the standard radial-tangential distortion model. The residual at pose i for landmark $\mathbf{m}_\ell$ is:

$$ \mathbf{r}_{i,\mathbf{m}_\ell} = \begin{pmatrix} \pi_u^L(\mathbf{R}_i, \mathbf{p}_i, \mathbf{m}_\ell) - u_{i,\ell}^L \\ \pi_u^R(\mathbf{R}_i, \mathbf{p}_i, \mathbf{m}_\ell) - u_{i,\ell}^R \\ \pi_v(\mathbf{R}_i, \mathbf{p}_i, \mathbf{m}_\ell) - v_{i,\ell} \end{pmatrix} \tag{26} $$

where $\Sigma_\mathbf{m}$ is computed using an uncertainty of 0.5 pixels.

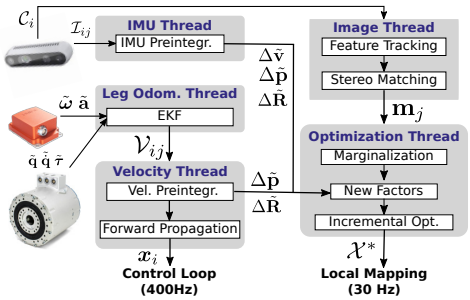


Fig. 5. The VILENS architecture with preintegrated velocity bias estimation.

V. Implementation

The state estimation architecture is shown in Fig. 5. Four parallel threads execute the following operations: preintegration of the IMU factor, preintegration of the velocity factor, stereo feature tracking, and optimization. This approach outputs 400 Hz velocity and pose estimates from the preintegration thread for use by the robot's control system, and a 30 Hz output from the factor-graph optimization thread for use by local mapping. When a new keyframe is processed, the preintegrated measurements and tracked landmarks are collected by the optimization thread, while the other threads process the next set of measurements. The factor graph optimization is implemented using the efficient incremental optimization solver iSAM2 [12], which is part of the GTSAM library [13]. We limit the number of states in the graph to 500 to keep the optimization time approximately constant.

A. Visual Feature Tracking

We detect features using the FAST corner detector, and track them between successive frames using the KLT feature tracker. Outliers are rejected using a RANSAC-based rejection method. Thanks to the parallel architecture and incremental optimization, all frames are used as keyframes, achieving 30 Hz nominal output. In contrast to [7], we use the Dynamic Covariance Scaling (DCS) [14] robust cost function to reduce the effect of landmark correspondence outliers on the optimization.

B. Zero Velocity Update Factors

To limit drift and factor graph growth when the robot is stationary, we enforce zero velocity updates on the different sensor modalities (camera, IMU, and leg odometry). If two out of three modalities report no motion, a zero velocity constraint factor is added to the graph. The IMU and leg odometry threads report zero velocity when position (rotation) is less than 0.1 mm (0.5°) between two keyframes. The image thread reports zero velocity when less than 0.5 pixels displacement of all the features is detected over the same period.

VI. Experimental Results

We have tested our proposed algorithm on a variety of terrain types for a total time of 53 min and 403 m traveled distance. The datasets consist of four scenarios (Fig. 6):



Fig. 6. Onboard camera feed from the four scenarios evaluated. *Top left*: wet concrete (FSC); *Top right:* gravel and grass (SMR1); *Bottom left:* rockbeds (SMR2); *Bottom right:* muddy grass uphill (SMR3).

10 m Relative Pose Error (RPE) $\mu(\sigma)$ [m]				
Data	**TSIF**[9]	**V-VI**	**V-RP**[7]	**V-VB**
FSC	0.49 (0.36)	0.42 (0.32)	0.47 (0.40)	**0.36** (0.30)
SMR1	0.96 (0.44)	**0.36** (0.29)	**0.36** (0.32)	**0.36** (0.33)
SMR2	0.69 (0.23)	**0.24** (0.16)	0.45 (0.10)	**0.24** (0.16)
SMR3	0.87 (0.42)	0.43 (0.48)	0.53 (0.46)	**0.39** (0.48)

TABLE II

- **FSC** a 240 m long trajectory consisting of three loops on wet concrete, standing water/oil, gravel and mud;
- **SMR1** a 106 m straight trot over concrete, gravel and high grass, followed by two loops on short grass alternating between dynamic and static gaits;
- **SMR2** a 22 m straight trot on rockbeds;
- **SMR3** a 35 m trot in a loop uphill with grass, mud and external disturbances applied to the robot to cause slippage events.

The first dataset was collected at Fire Service College (FSC), Moreton-on-Marsh, UK; the other three at the Swiss Military Rescue Center (SMR), Wangen an der Aare, Switzerland. Different copies of the ANYmal robot were used in the experiments. The attached video gives a sense of the conditions.

To generate ground truth, we tracked the robot using the Leica TS16 laser tracker (shown in Fig. 1), which provides millimeter accurate position measurements at 5 Hz. The orientation was reconstructed with an optimization method similar to that used by the EuRoC dataset [15].

We have evaluated the Relative Pose Error (RPE) over a distance of 10 m for the following algorithms:

- **V-VB:** VILENS with our proposed velocity bias factors;
- **V-RP:** VILENS with leg odometry integrated as relative pose factors, as used in our previous work [7];
- **V-VI:** a pure visual inertial navigation system (i.e., VILENS without leg odometry factors);
- **TSIF:** default ANYmal state estimator [9], our baseline.

Note that the same IMU and camera settings have been used for all configurations and datasets. Also, in comparison to our previous work [7], the robustness of visual feature tracking has been improved due to the introduction of stereo factors, higher framerates and robust cost functions.

The results are summarized in Table II. In three of the four datasets (FSC, SMR2 and SMR3), V-VI is able to



Fig. 7. Aerial view of the trajectories for method V-VB and the ground truth on the FSC dataset (240 m traveled). The common start point is indicated with a circle and the two endpoints with a cross. The lower part of the trajectory has no ground truth. Note that V-VB has no loop-closure system.

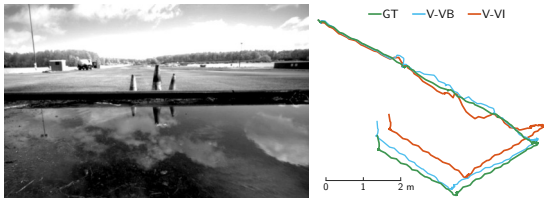


Fig. 8. *Left:* Onboard image of a visually challenging scene containing reflections from a large, oily puddle. *Right:* Top-down comparison of V-VB and V-VI trajectories aligned with ground truth while crossing the puddle.

outperform V-RP. This is because the datasets are designed to be particularly challenging for leg odometry. Therefore, the inclusion of relative pose factors without compensating for leg odometry drift actually degrades the performance compared to a visual-inertial only system. With the preintegrated velocity bias estimation, leg odometry improves the estimate up to 14 % compared to V-VI and 26 % compared to V-RP.

The global performance is shown in Fig. 7, which depicts the estimated and ground truth trajectories on the 240 m FSC dataset. By incorporating visual information to reject drift, the final z position of V-VB is 8.6 cm above ground truth, compared to a drift of 4.02 m from the TSIF kinematic-inertial estimator. Note that since VILENS is an odometry system, no loop closures have been performed.

A. Analysis of Visually Challenging Episodes

Most of the datasets presented favorable conditions for VO (well lit static scenes with texture). However, there were also certain locations where feature tracking struggled.

Fig. 8 shows a situation from FSC dataset where the robot traverses a large puddle. V-VI tracks the features on the water, causing drift in the lateral direction. Instead, V-VB maintains a better pose estimate by relying on leg odometry, whose drift is suppressed using the estimated velocity bias.

B. Velocity Bias Evolution

We have compared the estimated online bias in the z-axis to a lowpass filtered version of the same signal from the Pronto EKF [2] (Fig. 9). The sequence analyzed is the same as the one shown in Fig. 2. Since the z-axis position and average velocity of the robot are zero, the high correlation

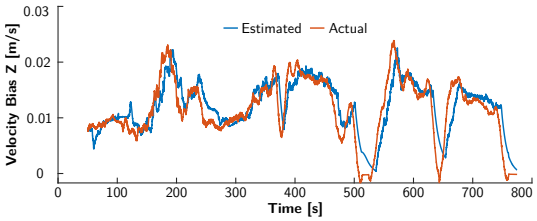


Fig. 9. Using visual information, VILENS is able to accurately and stably estimate the bias of the kinematic-inertial estimate. Experiment: SMR1.

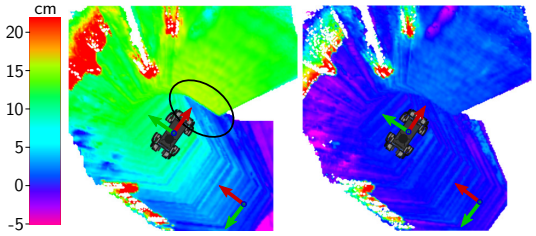


Fig. 10. Terrain reconstruction comparison between TSIF [9] (left) and VILENS (right). The robot walks from the bottom right corner to the center of the image and turns 90° right. VILENS eliminates the 15 cm drift and drastically reduces the number of artifacts on the elevation map.

between the two signals demonstrates the effectiveness of leg odometry drift rejection.

C. Terrain Reconstruction Assessment

We have evaluated the quality of local terrain mapping during a sequence of walking and turning on flat ground from the FSC dataset (Fig. 10). Due to drift in the ANYmal's internal filter [9], the elevation map contains a phantom discontinuity in front of the robot (encircled in black). With VILENS, the drift is reduced for effective footstep planning.

VII. Conclusion

We have presented a novel factor graph formulation for state estimation that estimates preintegrated velocity factors for leg odometry and velocity bias estimation to accommodate for leg odometry drift. These bias effects are difficult to directly model, we instead infer them from vision. The redundancy of our approach is also demonstrated in visual impoverished situations which vision along would struggle. In these situations, our system gracefully relies on leg odometry and the velocity bias estimation compensates for its drift. We have demonstrated the robustness of our method with outdoor experiments which include conditions such as slippery and deformable terrain, reflections, and external disturbances applied to the robot.

VIII. Acknowledgements

This research has been conducted as part of the ANYmal research community. It was part funded by the EU H2020 Projects THING and MEMMO, a Royal Society University Research Fellowship (Fallon) and a Google DeepMind studentship (Wisth). Special thanks to Ruben Grandia and Matthew Jose Pollayil for the support during experiments and the RSL group (ETH) for general support.

REFERENCES

[1] M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, and R. Siegwart, "State estimation for legged robots on unstable and slippery terrain," in *IEEE International Conference on Intelligent Robots and Systems*, 2013, pp. 6058–6064.

[2] M. Camurri, M. Fallon, S. Bazeille, A. Radulescu, V. Barasuol, D. G. Caldwell, and C. Semini, "Probabilistic contact estimation and impact detection for state estimation of quadruped robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1023–1030, 2017.

[3] F. Jenelten, J. Hwangbo, F. Tresoldi, C. D. Bellicoso, and M. Hutter, "Dynamic locomotion on slippery ground," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4170–4176, Oct 2019.

[4] R. Hartley, M. G. Jadidi, L. Gan, J.-K. Huang, J. W. Grizzle, and R. M. Eustice, "Hybrid contact preintegration for visual-inertial-contact state estimation within factor graphs," in *IEEE International Conference on Intelligent Robots and Systems*, 2018.

[5] T. Koolen, S. Bertrand, G. Thomas, T. De Boer, T. Wu, J. Smith, J. Englsberger, and J. Pratt, "Design of a momentum-based control framework and application to the humanoid robot atlas," *International Journal of Humanoid Robotics*, vol. 13, no. 01, p. 1650007, 2016.

[6] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[7] D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robotics and Automation Letters*, 2019.

[8] F. Dellaert, "Factor Graphs and GTSAM: A Hands-on Introduction, Tech. Rep. September, 2012. [Online]. Available: https://research.cc.gatech.edu/borg/sites/edu.borg/files/downloads/gtsam.pdf

[9] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback," *International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[10] J. Ma, M. Bajracharya, S. Susca, L. Matthies, and M. Malchano, "Real-time pose estimation of a dynamic quadruped in GPS-denied environments for 24-hour operation," *International Journal of Robotics Research*, vol. 35, no. 6, pp. 631–653, 2016.

[11] P. Furgale, "Representing Robot Pose: The good, the bad, and the ugly," 2014. [Online]. Available: http://paulfurgale.info/news/2014/6/9/representing-robot-pose-the-good-the-bad-and-the-ugly

[12] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "ISAM2: Incremental smoothing and mapping using the Bayes tree," *International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[13] F. Dellaert and M. Kaess, *Factor Graphs for Robot Perception*, 2017, vol. 6, no. 1-2. [Online]. Available: http://www.nowpublishers.com/article/Details/ROB-043

[14] K. MacTavish and T. D. Barfoot, "At All Costs: A comparison of robust cost functions for camera correspondence outliers," in *Conference on Computer and Robot Vision*, 2015, pp. 62–69.

[15] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	Preintegrated Velocity Bias Estimation to Overcome Contact Nonlinearities in Legged Robot Odometry
Publication Status	Published
Publication Details	D. Wisth, M. Camurri, and M. Fallon, "Preintegrated velocity bias estimation to overcome contact nonlinearities in legged robot odometry," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 392-398.

Student Confirmation

Student Name:	David Wisth		
Contribution to the Paper	My contributions to the paper were: • Developed the idea behind the paper • Implemented the algorithm • Performed the experiments with co-authors • Generated experimental results • Wrote the paper with co-authors		
Signature		Date	30/09/2021

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Maurice Fallon			
Supervisor comments This is accurate			
Signature		Date	4/10/2021

This completed form should be included in the thesis, at the end of the relevant chapter.

4.3 Discussion

This chapter presented two different methods for integrating kinematics into a factor graph framework. These articles were one of the first attempts to apply factor graphs to legged robots. One of the key elements for accurate kinematic sensing is the handling of non-rigid contacts, including slippage or ground deformation. This was addressed by creating a new preintegrated twist factor, which estimated and rejected the kinematic bias online. This system is robust to the failure of either vision or kinematics.

In dark or featureless environments, vision can fail. Without an additional exteroceptive sensor, only kinematic-inertial odometry can be used, which can suffer from higher drift rates. A solution is to incorporate lidar sensors. These complementary sensors are becoming increasingly widespread across robotics, autonomous vehicles, and handheld sensors, and provide metrically accurate 3D information.

Chapter 5

Incorporating Lidar Information

The visual-inertial-kinematic system presented in Chapter 4 performed well in most scenarios. However, in featureless environments, this method reverts to kinematic-inertial inputs only, reducing its accuracy. To overcome this limitation, this chapter presents a novel method of integrating lidar information into a factor graph.

Specifically, this chapter extends the VILENS state estimation with lidar measurements and presents the results of visual-inertial-lidar fusion on both legged robots and handheld sensors. Inspired by the success with visual feature tracking, lightweight geometric primitives from both the lidar and camera sensors are tracked and optimised, along with IMU measurements. These geometric primitives (i.e., points, lines, and planes) are tracked over time, unlike previous works which most commonly use only frame-to-frame feature tracking [48].

A key benefit of this approach is the graceful handling of degenerate states. In a loosely coupled system, if the lidar odometry is degenerate (e.g., in a large open field) then all lidar measurements may need to be discarded. In the worst case, the state estimator may diverge and thus no longer produce sensible output. In contrast, with the proposed method, since all landmarks are fused in a single graph, *partial constraints* from all sensors can be used without modification or hard-switching. Additionally, by choosing a sparse approach, the system is computationally efficient and can run at nominal sensor frequency using only on-board computation.

5.1 Unified Multi-Modal Landmark Tracking for Tightly Coupled Lidar-Visual-Inertial Odometry

This article was published in the IEEE Robotics and Automation Letters (2021) and presented at the IEEE International Conference on Robotics and Automation (ICRA) (2021) [14]. This article was a finalist for the *IEEE ICRA Best Student Paper Award 2021*. A video summary of this publication can be found at: `https://youtu.be/MjXYAHurWe8`. The associated conference presentation can be found at: `https://youtu.be/l3XSGIpQP3Q`.

IEEE ROBOTICS AND AUTOMATION LETTERS. PREPRINT VERSION. ACCEPTED JANUARY, 2021 1

Unified Multi-Modal Landmark Tracking for Tightly Coupled Lidar-Visual-Inertial Odometry

David Wisth[1], Marco Camurri[1], Sandipan Das[2,3], Maurice Fallon[1]

***Abstract*—We present an efficient multi-sensor odometry system for mobile platforms that jointly optimizes visual, lidar, and inertial information within a single integrated factor graph. This runs in real-time at full framerate using fixed lag smoothing. To perform such tight integration, a new method to extract 3D line and planar primitives from lidar point clouds is presented. This approach overcomes the suboptimality of typical frame-to-frame tracking methods by treating the primitives as landmarks and tracking them over multiple scans. True integration of lidar features with standard visual features and IMU is made possible using a subtle passive synchronization of lidar and camera frames. The lightweight formulation of the 3D features allows for real-time execution on a single CPU. Our proposed system has been tested on a variety of platforms and scenarios, including underground exploration with a legged robot and outdoor scanning with a dynamically moving handheld device, for a total duration of 96 min and 2.4 km traveled distance. In these test sequences, using only one exteroceptive sensor leads to failure due to either underconstrained geometry (affecting lidar) or textureless areas caused by aggressive lighting changes (affecting vision). In these conditions, our factor graph naturally uses the best information available from each sensor modality without any hard switches.**

***Index Terms*—Sensor Fusion; Visual-Inertial SLAM; Localization**

I. Introduction

MULTI-MODAL sensor fusion is a key capability required for successful deployment of autonomous navigation systems in real-world scenarios. The spectrum of possible applications is wide, ranging from autonomous underground exploration to outdoor dynamic mapping (see Fig. 1).

Recent advances in lidar hardware have fostered research into lidar-inertial fusion [1], [2], [3], [4]. The wide field-of-view, density, range, and accuracy of lidar sensors makes them suitable for navigation, localization, and mapping tasks.

However, in environments with degenerate geometries, such as long tunnels or wide open spaces, lidar-only approaches can fail. Since the IMU integration alone cannot provide reliable pose estimates for more than a few seconds, the system failure is often unrecoverable. To cope with these situations, fusion with additional sensors, in particular cameras, is also required. While visual-inertial-lidar fusion has already been achieved in the past via loosely coupled methods [5], tightly coupled methods such as incremental smoothing are more desirable because of their superior robustness.

In the domain of smoothing methods, research on Visual-Inertial Navigation Systems (VINS) is now mature and lidar-inertial systems are becoming increasingly popular. However, the tight fusion of all three sensor modalities at once is still an open research problem.



Fig. 1. We tested our multi-sensor odometry algorithm with data from the ANYmal quadruped robot [6] in the DARPA SubT Challenge (top, courtesy RSL/ETH Zurich) and a handheld mapping device in New College, Oxford [7] (bottom). Video: https://youtu.be/MjXYAHurWe8

A. Motivation

The two major challenges associated with the fusion of IMU, lidar and camera sensing are: 1) achieving real-time performance given the limited computational budget of mobile platforms and 2) the appropriate synchronization of three signals running at different frequencies and methods of acquisition.

Prior works have addressed these two problems by adopting loosely coupled approaches [5], [8], [9] or by running two separate systems (one for lidar-inertial and the other for visual-inertial odometry) [10].

Manuscript received: October, 15, 2020; Revised January, 8, 2021; Accepted January, 13, 2021.

This paper was recommended for publication by Editor Eric Marchand upon evaluation of the Associate Editor and Reviewers' comments. This research has been conducted as part of the ANYmal research community. It was part funded by the EU H2020 Projects THING and MEMMO, a Royal Society University Research Fellowship (Fallon) and a Google DeepMind studentship (Wisth). Special thanks to the CERBERUS DARPA SubT Team for providing data from challenge runs.

[1]Oxford Robotics Institute, Department of Engineering Science, University of Oxford, UK {davidw, mcamurri, mfallon}@robots.ox.ac.uk

[2]KTH Royal Institute of Technology, Sweden sandipan@kth.se

[3]Scania AB, Sweden sandipan.das@scania.com

Digital Object Identifier (DOI): see top of this page.

Instead, we are motivated to tackle these problems by: 1) extracting and tracking sparse lightweight primitives and 2) developing a coherent factor graph which leverages IMU pre-integration to transform dynamically dewarped point clouds to the timestamp of nearby camera frames. The former avoids matching entire point clouds (e.g. ICP) or tracking hundreds of feature points (as in LOAM [1]). The latter makes real-time smoothing of all the sensors possible.

B. Contribution

The main contributions of this work are the following:

- A novel factor graph formulation that tightly fuses vision, lidar and IMU measurements within a single consistent optimization process;
- An efficient method for extracting lidar features, which are then optimized as landmarks. Both lidar and visual features share a unified representation, as the landmarks are all treated as n-dimensional parametric manifolds (i.e., points, lines and planes). This compact representation allows us to process all the lidar scans at nominal framerate;
- Extensive experimental evaluation across a range of scenarios demonstrating superior robustness when compared to more typical approaches which struggle when individual sensor modalities fail.

Our work builds upon the VILENS estimation system introduced in our previous works [11], [12] by adding lidar feature tracking and lidar-aided visual tracking. The combination of camera and lidar enables the use on portable devices even when moving aggressively, as it naturally handles degeneracy in the scene (either due to a lack of lidar or visual features).

II. Related Work

Prior works on multi-modal sensor fusion use combinations of lidar, camera and IMU sensing and can be characterised as either loosely or tightly coupled, as summarized in Table I. Loosely coupled systems process the measurements from each sensor separately and fuse them within a filter, where they are marginalized to get the current state estimate. Alternatively, tightly coupled systems jointly optimize both past and current measurements to obtain a complete trajectory estimate.

Another important distinction is between odometry systems and SLAM systems. In the latter, loop closures are performed to keep global consistency of the estimate once the same place is visited twice. Even though some of the works in the table also include a pose-graph SLAM backend, we are mainly interested in high-frequency odometry here.

Sensors	Loosely coupled systems	Tightly coupled systems
Lidar + IMU	LOAM*[1], LEGO-LOAM*[2], SLAM with point and plane†[13]	LIPS[3], LIOM[4], LIO-SAM[14]
Vision + Lidar + IMU	V-LOAM[8], Visual-Inertial-Laser SLAM[9], Multi-modal Sensor Fusion[15]‡, Pronto[5]	LIMO[16], VIL-SLAM[10], INS with Point and Plane feature[17], LIC-Fusion[18], LIC-Fusion 2.0[19]

* *IMU is optional*, † *Lidar only solution*, ‡ *Additional thermal camera.*

TABLE I
Different multi-modal odometry estimation algorithms

A. Loosely Coupled Lidar-Inertial Odometry

Lidar-based odometry has gained popularity thanks to the initial work of Zhang et al. [1], who proposed the LOAM algorithm. One of their key contributions is the definition of edge and planar 3D feature points which are tracked frame-to-frame. The motion between two frames is linearly interpolated using an IMU running at high-frequency. This motion prior is used in the fine matching and registration of the features to achieve high accuracy odometry. Shan et al. [2] proposed LeGO-LOAM, which further improved the real-time performance of LOAM for ground vehicles by optimizing an estimate of the ground plane.

However, these algorithms will struggle to perform robustly in structure-less environments or in degenerate scenarios [20] where constraints cannot be found due to the lidar's limited range and resolution — such as long motorways, tunnels, and open spaces.

B. Loosely Coupled Visual-Inertial-Lidar Odometry

In many of the recent works [8], [9], [15], [5] vision was incorporated along with lidar and IMU for odometry estimation in a loosely coupled manner to provide a complementary sensor modality to both avoid degeneracy and have a smoother estimated trajectory over lidar-inertial systems.

The authors of LOAM extended their algorithm by integrating feature tracking from a monocular camera in V-LOAM [8] along with IMU, thereby generating a visual-inertial odometry prior for lidar scan matching. However, the operation was still performed frame-to-frame and didn't maintain global consistency. To improve consistency, a Visual-Inertial-Lidar SLAM system was introduced by Wang et al. [9] where they used a V-LOAM based approach for odometry estimation and performed a global pose graph optimization by maintaining a keyframe database. Khattak et al. [15] proposed another loosely coupled approach similar to V-LOAM, that uses a visual/thermal inertial prior for lidar scan matching. To overcome degeneracy, the authors used visual and thermal inertial odometry so as to operate in long tunnels with no lighting. In Pronto [5], the authors used visual-inertial-legged odometry as a motion prior for a lidar odometry system and integrated pose corrections from visual and lidar odometry to correct pose drift in a loosely coupled manner.

C. Tightly Coupled Inertial-Lidar Odometry

One of the earlier methods to tightly fuse lidar and IMU was proposed in LIPS [3], a graph-based optimization framework which optimizes the 3D plane factor derived from the closest point-to-plane representation along with preintegrated IMU measurements. In a similar fashion, Ye et al. [4] proposed LIOM, a method to jointly minimize the cost derived from lidar features and pre-integrated IMU measurements. This resulted in better odometry estimates than LOAM in faster moving scenarios. Shan et al. [14] proposed LIO-SAM, which adapted the LOAM framework by introducing scan matching at a local scale instead of global scale. This allowed new keyframes to be registered to a sliding window of prior "sub-keyframes"

merged into a voxel map. The system was extensively tested on a handheld device, ground, and floating vehicles, highlighting the quality of the reconstruction of the SLAM system. For long duration navigation they also used loop-closure and GPS factors for eliminating drift.

Again, due to the absence of vision, the above algorithms may struggle to perform robustly in degenerate scenarios.

D. Tightly Coupled Visual-Inertial-Lidar Odometry

To avoid degeneracy and to make the system more robust, tight integration of multi-modal sensing capabilities (vision, lidar, and IMU) was explored in some more recent works [10], [16], [17], [18], [19]. In LIMO [16] the authors presented a bundle adjustment–based visual odometry system. They combined the depth from lidar measurements by re-projecting them to image space and associating them to the visual features which helped to maintain accurate scale. Shao et al. [10] introduced VIL-SLAM where they combined VIO along with lidar-based odometry as separate sub-systems for combining the different sensor modalities rather than doing a joint optimization.

To perform joint state optimization, many approaches [17], [18], [19] use the Multi-State Constraint Kalman Filter (MSCKF) framework [21]. Yang et al. [17] tightly integrated the plane features from an RGB-D sensor within 3.5 m range and point features from vision and IMU measurements using an MSCKF. To limit the state vector size, most of the point features were treated as MSCKF features and linearly marginalized, while only a few point features enforcing point-on-plane constraints were kept in state vector as SLAM features. LIC-Fusion introduced by Zuo et al. [18] tightly combines the IMU measurements, extracted lidar edge features, as well as sparse visual features, using the MSCKF fusion framework. Whereas, in a recent follow up work, LIC-Fusion 2.0 [19], the authors introduced a sliding window based plane-feature tracking approach for efficiently processing 3D lidar point clouds.

In contrast with previous works, we jointly optimize the three aforementioned sensor modalities within a single, consistent factor graph optimization framework. To process lidar data at real-time, we directly extract and track 3D primitives such as lines and planes from the lidar point clouds, rather than performing "point-to-plane" or "point-to-line" based cost functions. This allows for natural tracking over multiple frames in a similar fashion to visual tracking, and to constrain the motion even in degenerate scenarios.

III. Problem Statement

We aim to estimate the position, orientation, and linear velocity of a mobile platform (in our experiments, a legged robot or a handheld sensor payload) equipped with IMU, lidar and either a mono or stereo camera with low latency and at full sensor rate.

The relevant reference frames are specified in Fig. 2 and include the robot-fixed base frame B, left camera frame C, IMU frame I, and lidar frame L. We wish to estimate the position of the base frame relative to a fixed world frame W.

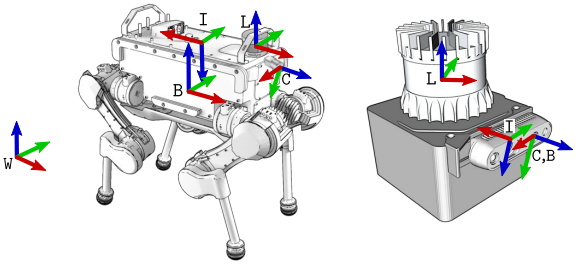


Fig. 2. Reference frames conventions for the ANYmal platform and the handheld device. The world frame W is a fixed frame, while the base frame B, camera optical frame C, IMU frame, I, and lidar frame L are attached to the moving robot's chassis or device. For simplicity, C and B are coincident on the handheld device.

Unless otherwise specified, the position ${}_{\mathtt{W}}\mathbf{p}_{\mathtt{WB}}$ and orientation $\mathbf{R}_{\mathtt{WB}}$ of the base (with ${}_{\mathtt{W}}\mathbf{T}_{\mathtt{WB}} \in \mathrm{SE}(3)$ as the corresponding homogeneous transform) are expressed in world coordinates; velocities ${}_{\mathtt{B}}\mathbf{v}_{\mathtt{WB}}, {}_{\mathtt{B}}\boldsymbol{\omega}_{\mathtt{WB}}$ are in the base frame, and IMU biases ${}_{\mathtt{I}}\mathbf{b}^g$, ${}_{\mathtt{I}}\mathbf{b}^a$ are expressed in the IMU frame.

A. State Definition

The mobile platform state at time t_i is defined as follows:

$$\boldsymbol{x}_i \triangleq [\mathbf{R}_i, \mathbf{p}_i, \mathbf{v}_i, \mathbf{b}_i^g \ \mathbf{b}_i^a] \in \mathrm{SO}(3) \times \mathbb{R}^{12} \tag{1}$$

where: $\mathbf{R}_i$ is the orientation, $\mathbf{p}_i$ is the position, $\mathbf{v}_i$ is the linear velocity, and the last two elements are the usual IMU gyroscope and accelerometer biases.

In addition to the states, we track the parameters of three n-manifolds: points, lines and planes. The point landmarks $\boldsymbol{m}_\ell$ are visual features, while lines $\boldsymbol{l}_\ell$ and planes $\boldsymbol{p}_\ell$ landmarks are extracted from lidar. The objective of our estimation are all states and landmarks visible up to the current time t_k:

$$\mathcal{X}_k \triangleq \left[\bigcup_{\forall i \in \mathsf{X}_k} \{\boldsymbol{x}_i\}, \bigcup_{\forall \ell \in \mathsf{M}_k} \{\boldsymbol{m}_\ell\}, \bigcup_{\forall \ell \in \mathsf{P}_k} \{\boldsymbol{p}_\ell\}, \bigcup_{\forall \ell \in \mathsf{L}_k} \{\boldsymbol{l}_\ell\} \right] \tag{2}$$

where $\mathsf{X}_k, \mathsf{M}_k, \mathsf{P}_k, \mathsf{L}_k$ are the lists of all states and landmarks tracked within a fixed lag smoothing window.

B. Measurements Definition

The measurements from a mono or stereo camera $\mathcal{C}$, IMU $\mathcal{I}$, and lidar $\mathcal{L}$ are received at different times and frequencies. We define $\mathcal{Z}_k$ as the full set of measurements received within the smoothing window. Subsection V-B1 explains how the measurements are integrated within the factor graph, such that the optimization is performed at fixed frequency.

C. Maximum-a-Posteriori Estimation

We maximize the likelihood of the measurements, $\mathcal{Z}_k$, given the history of states, $\mathcal{X}_k$:

$$\mathcal{X}_k^* = \arg\max_{\mathcal{X}_k} p(\mathcal{X}_k|\mathcal{Z}_k) \propto p(\mathcal{X}_0) p(\mathcal{Z}_k|\mathcal{X}_k) \tag{3}$$

The measurements are formulated as conditionally independent and corrupted by white Gaussian noise. Therefore, Eq. (3)

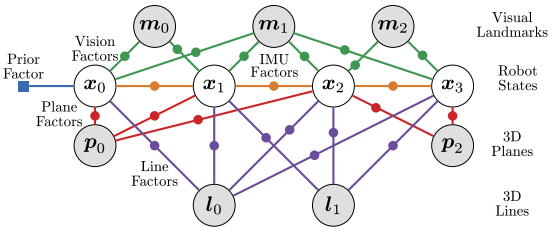


Fig. 3. VILENS sliding-window factor graph structure, showing prior, visual, plane, line, and preintegrated IMU factors. Tracking landmarks over time increases the accuracy of the estimation by allowing new measurements to improve the accuracy of past states.

can be formulated as the following least squares minimization problem [22]:

$$\mathcal{X}_k^* = \arg\min_{\mathcal{X}_k} \sum_{i \in \mathsf{K}_k} \left(\|\mathbf{r}_{\mathcal{I}_{ij}}\|^2_{\Sigma_{\mathcal{I}_{ij}}} + \sum_{\ell \in \mathsf{P}_i} \|\mathbf{r}_{\boldsymbol{x}_i, \boldsymbol{p}_\ell}\|^2_{\Sigma_{\boldsymbol{x}_i, \boldsymbol{p}_\ell}} + \sum_{\ell \in \mathsf{L}_i} \|\mathbf{r}_{\boldsymbol{x}_i, \boldsymbol{l}_\ell}\|^2_{\Sigma_{\boldsymbol{x}_i, \boldsymbol{l}_\ell}} + \sum_{\ell \in \mathsf{M}_i} \|\mathbf{r}_{\boldsymbol{x}_i, \boldsymbol{m}_\ell}\|^2_{\Sigma_{\boldsymbol{x}_i, \boldsymbol{m}_\ell}} \right) + \|\mathbf{r}_0\|^2_{\Sigma_0} \quad (4)$$

where $\mathcal{I}_{ij}$ are the IMU measurements between t_i and t_j and K_k are all keyframe indices before t_k. Each term is the residual associated to a factor type, weighted by the inverse of its covariance matrix. The residuals are: IMU, lidar plane and line features, visual landmarks, and a state prior.

IV. Factor Graph Formulation

We now describe the measurements, residuals, and covariances of the factors in the graph, shown in Fig. 3. For convenience, we summarize the IMU factors in Section IV-A; then, we introduce the visual-lidar landmark factors in Sections IV-B and IV-C, while Section IV-D describes our novel plane and line landmark factors.

A. Preintegrated IMU Factors

We follow the now standard manner of IMU measurement preintegration [23] to constrain the pose, velocity, and biases between two consecutive nodes of the graph, and provide high frequency states updates between nodes. The residual has the form:

$$\mathbf{r}_{\mathcal{I}_{ij}} = \left[\mathbf{r}^\mathsf{T}_{\Delta \mathbf{R}_{ij}}, \mathbf{r}^\mathsf{T}_{\Delta \mathbf{v}_{ij}}, \mathbf{r}^\mathsf{T}_{\Delta \mathbf{p}_{ij}}, \mathbf{r}_{\mathbf{b}^a_{ij}}, \mathbf{r}_{\mathbf{b}^g_{ij}} \right] \quad (5)$$

For the definition of the residuals, see [23].

B. Mono Landmark Factors with Lidar Depth

To take full advantage of the fusion of vision and lidar sensing modalities, we track monocular visual features but use the lidar's overlapping field-of-view to provide depth estimates, as in [16]. To match the measurements from lidar and camera, which operate at 10 Hz and 30 Hz respectively, we use the method described in Section V-B1.

Let $\boldsymbol{m}_\ell \in \mathbb{R}^3$ be a visual landmark in Euclidean space, $\pi : \mathrm{SE}(3) \times \mathbb{R}^3 \mapsto \mathbb{R}^2$ a function that projects a landmark to the image plane given a platform pose $\mathbf{T}_i$ (for simplicity, we omit the fixed transforms between base, lidar and camera), and $(u_\ell, v_\ell) \in \mathbb{R}^2$ a detection of $\boldsymbol{m}_\ell$ on the image plane (yellow dots in Fig. 4, right). We first project all the points $\tilde{\mathbf{x}}_m \in \mathcal{L}_i$ acquired by the lidar between time t_i and t_{i+1} onto the image plane with $\pi(\mathbf{T}_i, \tilde{\mathbf{x}}_m)$ (green dots in Fig. 4, right). Then, we find the projected point $\pi(\tilde{\mathbf{x}}_\ell)$ that is closest to (u_ℓ, v_ℓ) on the image plane within a neighborhood of 3 pixels. Finally, the residual is computed as:

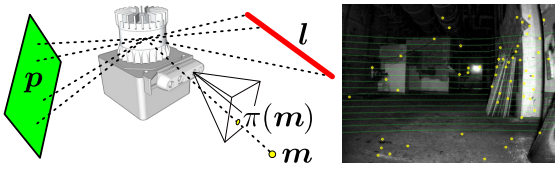


Fig. 4. *Left*: The visual FAST features $\boldsymbol{m}$ (yellow); the lidar line $\boldsymbol{l}$: (red), and plane $\boldsymbol{p}$ primitive (green) are tracked by our method. *Right*: The projection of the lidar data (green) along with the visual features (yellow) into the image frame, which helps to associate depth to the visual features.

$$\mathbf{r}_{\boldsymbol{x}_i, \boldsymbol{m}_\ell} = \mathbf{T}_i^{-1} \boldsymbol{m}_\ell - \tilde{\mathbf{x}}_\ell \quad (6)$$

When we cannot associate lidar depth to a visual feature (due to the different resolution of lidar and camera sensors) or if it is unstable (i.e., when the depth changes $> 0.5\,\mathrm{m}$ between frames due to dynamic obstacles or noise), we revert to stereo matching, as described in the next section.

C. Stereo Landmark Factors

The residual at state $\boldsymbol{x}_i$ for landmark $\boldsymbol{m}_\ell$ is [12]:

$$\mathbf{r}_{\boldsymbol{x}_i, \boldsymbol{m}_\ell} = \begin{pmatrix} \pi^L_u(\mathbf{T}_i, \boldsymbol{m}_\ell) - u^L_{i,\ell} \\ \pi^R_u(\mathbf{T}_i, \boldsymbol{m}_\ell) - u^R_{i,\ell} \\ \pi_v(\mathbf{T}_i, \boldsymbol{m}_\ell) - v_{i,\ell} \end{pmatrix} \quad (7)$$

where $(u^L, v), (u^R, v)$ are the pixel locations of the detected landmark and $\Sigma_{\boldsymbol{m}}$ is computed from an uncertainty of 0.5 pixels. Finally, if only a monocular camera is available, then only the first and last elements in Eq. 7 are used.

D. Plane Landmark Factor

We use the Hessian normal form to parametrize an infinite plane $\boldsymbol{p}$ as a unit normal $\hat{\mathbf{n}} \in \mathbb{R}^3$ and a scalar d representing its distance from the origin:

$$\boldsymbol{p} = \left\{ \langle \hat{\mathbf{n}}, d \rangle \in \mathbb{R}^4 \mid \hat{\mathbf{n}} \cdot (x, y, z) + d = 0 \right\} \quad (8)$$

Let $\otimes$ be the operator that applies a homogeneous transform $\mathbf{T}$ to all the points of a plane $\boldsymbol{p}$, and $\ominus$ the operator that defines the error between two planes $(\boldsymbol{p}_i, \boldsymbol{p}_j)$ as:

$$\boldsymbol{p}_i \ominus \boldsymbol{p}_j = \left(B_p^\mathsf{T} \hat{\xi}, d_i - d_j \right) \in \mathbb{R}^3 \quad (9)$$

where $B_p \in \mathbb{R}^{3 \times 2}$ is a basis for the tangent space of $\hat{\mathbf{n}}_i$ and $\hat{\xi}$ is defined as follows [24]:

$$\hat{\xi} = -\frac{\arccos(\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j)}{1 - (\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j)^2} \left(\hat{\mathbf{n}}_j - (\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j) \hat{\mathbf{n}}_i \right) \in \mathbb{R}^3 \quad (10)$$

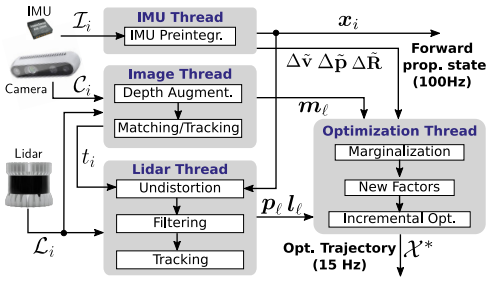


Fig. 5. Overview of the VILENS system architecture. The inputs are handled in separate threads by each front-end measurement handler. The back-end produces both a high frequency forward-propagated output and a lower frequency optimized output. This parallel architecture allows for different measurement inputs depending on the platform.

When a plane $\tilde{\boldsymbol{p}}_i$ is measured at time t_i, the corresponding residual is the difference between $\tilde{\boldsymbol{p}}$ and the estimated plane $\boldsymbol{p}_\ell$ transformed into the local reference frame:

$$\mathbf{r}_{x_i,\boldsymbol{p}_\ell} = \left(\mathbf{T}_i^{-1} \otimes \boldsymbol{p}_\ell\right) \ominus \tilde{\boldsymbol{p}}_i \tag{11}$$

E. Line Landmark Factor

Using the approach from [25], infinite straight lines can be parametrized by a rotation matrix $\mathbf{R} \in \mathrm{SO}(3)$ and two scalars $a,\, b \in \mathbb{R}$, such that $\hat{\mathbf{v}} = \mathbf{R}\hat{\mathbf{z}}$ is the direction of the line and $\mathbf{d} = \mathbf{R}(a\hat{\mathbf{x}} + b\hat{\mathbf{y}})$ is the closest point between the line and the origin. A line $\boldsymbol{l}$ can therefore be defined as:

$$\boldsymbol{l} = \left\{ \langle \mathbf{R}, (a, b) \rangle \in \mathrm{SO}(3) \times \mathbb{R}^2 \right\} \tag{12}$$

Let $\boxtimes$ be the operator that applies a transform $\mathbf{T}_{ij} = (\mathbf{R}_{ij}, \mathbf{p}_{ij})$ to all the points of a line $\boldsymbol{l}_i$ to get $\boldsymbol{l}_j$ such that:

$$\begin{aligned} \mathbf{R}_j &= \mathbf{R}_{ij}\mathbf{R}_i \\ a_j &= a_i - \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \mathbf{R}_{ij}^\mathsf{T} \mathbf{p}_{ij} \\ b_j &= b_i - \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \mathbf{R}_{ij}^\mathsf{T} \mathbf{p}_{ij} \end{aligned} \tag{13}$$

The error operator $\boxminus$ between two lines $\boldsymbol{l}_i, \boldsymbol{l}_j$ is defined as:

$$\boldsymbol{l}_i \boxminus \boldsymbol{l}_j = \left(\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix}^\mathsf{T} \mathrm{Log}\left(\mathbf{R}_i^\mathsf{T}\mathbf{R}_j\right), a_i - a_j, b_i - b_j \right) \in \mathbb{R}^4 \tag{14}$$

Given Eq. 13 and Eq. 14, the residual between a measured line $\tilde{\boldsymbol{l}}_i$ and its prediction is defined as follows:

$$\mathbf{r}_{x_i,\boldsymbol{l}\ell} = \left(\mathbf{T}_i^{-1} \boxtimes \boldsymbol{l}_\ell\right) \boxminus \tilde{\boldsymbol{l}}_i \tag{15}$$

We use the numerical derivatives of Eq. (11) and (15) in the optimization, using the symmetric difference method.

V. Implementation

The system architecture is shown in Fig. 5. Using four parallel threads for the sensor processing and optimization, the system outputs the state estimated by the factor-graph at camera keyframe frequency (typically 15 Hz) and the IMU forward-propagated state at IMU frequency (typically 100 Hz) for use in navigation/mapping and control respectively.

The factor graph is solved using a fixed lag smoothing framework based on the efficient incremental optimization solver iSAM2, using the GTSAM library [22]. For these experiments, we use a lag time of between 5 and 10 s. All visual and lidar factors are added to the graph using the Dynamic Covariance Scaling (DCS) [26] robust cost function to reduce the effect of outliers.

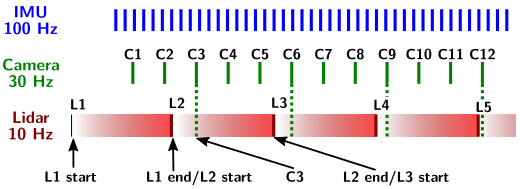


Fig. 6. Example of output frequencies and synchronization between IMU, camera and lidar. IMU and camera signals are captured instantaneously, while lidar points are captured and accumulated for 0.1 s before being sent as a scan. When the scan L2 is received, it is undistorted to the same time as camera frame C3 and added to the same graph node as the camera.

A. Visual Feature Tracking

We detect features using the FAST corner detector, and track them between successive frames using the KLT feature tracker with outliers rejected using RANSAC. Thanks to the parallel architecture and incremental optimization, every second frame is used as a keyframe, achieving 15 Hz nominal output.

B. Lidar Processing and Feature Tracking

A key feature of our algorithm is that we extract feature primitives from the lidar point clouds represented at the same time as a camera frame, such that the optimization can be executed for all the sensors at once. The processing pipeline consists of the following steps: point cloud undistortion and synchronization, filtering, primitive extraction and tracking, and factor creation.

1) Undistortion and Synchronization: Fig. 6 compares the different output frequencies of our sensors. While IMU and camera samples are captured instantaneously, lidars continually capture points while their internal mirror rotates around the z-axis. Once a full rotation is complete, the accumulated laser returns are converted into a point cloud and a new scan starts immediately thereafter.

Since the laser returns are captured while moving, the point cloud needs to be undistorted with a motion prior and associated to a unique arbitrary timestamp – typically the start of the scan [27]. This approach would imply that camera and lidar measurements have different timestamps and thus separate graph nodes.

Instead, we choose to undistort the lidar measurement to the closest camera timestamp after the start of the scan. For example, in Fig. 6, the scan L2 is undistorted to the timestamp of keyframe C3. Given the forward propagated states from the IMU module, the motion prior is linearly extrapolated using the timestamp associated to each point of the cloud (for simplicity, we avoid Gaussian-Process interpolation [28] or state augmentation with time offsets [29]). As the cloud is now associated with C3, the lidar landmarks are connected to the same node as C3 rather than creating a new one.

This subtle detail not only guarantees that a consistent number of new nodes and factors are added to the graph optimization, but it also ensures that the optimization is performed *jointly* between IMU, camera and lidar inputs. This also ensures a fixed output frequency, i.e., the camera framerate or lidar framerate (when cameras are unavailable), but not a mixture of the two.

2) Filtering: Once the point cloud has been undistorted, we perform the segmentation from [30] to separate the points into clusters. Small clusters (less than 5 points) are marked as outliers and discarded as they are likely to be noisy.

Then, the local curvature of each point in the pre-filtered cloud is calculated using the approach of [2]. The points with the lowest and highest curvature are assigned to the set of plane candidates $\mathcal{C}_P$ and line candidates $\mathcal{C}_L$, respectively.

The segmentation and curvature-based filtering typically reduce the number of points in the point cloud by 90%, providing significant computational savings in the subsequent plane and line processing.

3) Plane and Line Extraction and Tracking: Over time, we track planes and lines in the respective candidate sets $\mathcal{C}_P$ and $\mathcal{C}_L$. This is done in a manner analogous to local visual feature tracking methods, where features are tracked within a local vicinity of their predicted location.

First, we take the tracked planes and lines from the previous scan, $\boldsymbol{p}_{i-1}$ and $\boldsymbol{l}_{i-1}$, and use the IMU forward propagation to predict their location in the current scan, $\hat{\boldsymbol{p}}_i$ and $\hat{\boldsymbol{l}}_i$. Then to assist local tracking, we segment $\mathcal{C}_P$ and $\mathcal{C}_L$ around the predicted feature locations using a maximum point-to-model distance. Afterwards, we perform Euclidean clustering (and normal filtering for plane features) to remove outliers. Then, we fit the model to the segmented point cloud using a PROSAC [31] robust fitting algorithm.

Finally, we check that the predicted and detected landmarks are sufficiently similar. Two planes, $\boldsymbol{p}_i$ and $\boldsymbol{p}_j$, are considered a match when difference between their normals and the distance from the origin are smaller than a threshold:

$$\delta_n = \quad \|\arccos(\hat{\mathbf{n}}_i \cdot \hat{\mathbf{n}}_j)\| \quad < \alpha_p \tag{16}$$

$$\delta_d = \quad \|\hat{\mathbf{n}}_i d_i - \hat{\mathbf{n}}_j d_j\| \quad < \beta_p \tag{17}$$

Two lines $\boldsymbol{l}_i$ and $\boldsymbol{l}_j$ are considered a match if their directions and their center distances are smaller than a threshold:

$$\delta_n = \quad \|\arccos(\hat{\mathbf{v}}_i \cdot \hat{\mathbf{v}}_j)\| \quad < \alpha_l \tag{18}$$

$$\delta_d = \quad \|(\mathbf{d}_i - \mathbf{d}_j) - ((\mathbf{d}_i - \mathbf{d}_j) \cdot \hat{\mathbf{v}}_i)\,\hat{\mathbf{v}}_i\| \quad < \beta_l \tag{19}$$

In our case $\alpha_p = \alpha_l = 0.35$ rad, $\beta_p = \beta_l = 0.5$ m.

Once a feature has been tracked, the feature's inliers are removed from the corresponding candidate set, and the process is repeated for the remaining landmarks.

After tracking is complete, we detect new landmarks in the remaining candidate clouds. The point cloud is first divided using Euclidean clustering for lines, and normal-based region growing for planes. We then detect new landmarks in each cluster using the same method as landmark tracking.

Point cloud features are only included in the optimization after they have been tracked for a minimum number of consecutive scans. Note that the oldest features are tracked first, to ensure the longest possible feature tracks.

C. Zero Velocity Update Factors

To limit drift and factor graph growth when the platform is stationary, we add zero velocity constraints to the graph when updates from two out of three modalities (camera, lidar, IMU) report no motion.

VI. Experimental Results

We evaluated our algorithm on a variety of indoor and outdoor environments in two contrasting datasets: the Newer College Dataset [7] and the DARPA SubT Challenge (Urban). An overview of these environments is shown in Fig. 7.

A. Datasets

The Newer College dataset (NC) [7] was collected using a portable device equipped with a Ouster OS1-64 Gen1 lidar sensor, a RealSense D435i stereo IR camera, and an Intel NUC PC. The cellphone-grade IMU embedded in the lidar was used for inertial measurements. The device was carried by a person walking outdoor surrounded by buildings, large open spaces, and dense foliage. The dataset includes challenging sequences where the device was shaken aggressively to test the limits of tracking.

The SubT dataset (ST) consists of two of the most significant runs of the SubT competition (Alpha-2 and Beta-2) collected on two copies of the ANYmal B300 quadruped robot [6] equipped with a Flir BFS-U3-16S2C-CS monocular camera and a industrial-grade Xsens MTi-100 IMU, which were hardware synchronized by a dedicated board [32]. A Velodyne VLP-16 was also available but was synchronized via software. The robots navigated the underground interiors of an unfinished nuclear reactor. This dataset is challenging due to the presence of long straight corridors and extremely dark environments. Note that the leg kinematics from the robot **was not used** in this work.

The specific experiments are named as follows:

- **NC-1:** Walking around an open college environment (1134 m, 17 min).
- **NC-2:** Walking with highly dynamic motion in the presence of strong illumination changes (480 m, 6 min).
- **NC-3:** Shaking the sensor rig at very high angular velocities, up to 3.37 rad/s (91 m, 2 min).
- **ST-A:** Anymal quadruped robot trotting in dark underground reactor facility (167 m, 11 min).
- **ST-B:** A different Anymal robot in a part of the reactor containing a long straight corridor (490 m, 60 min).

To generate ground truth, ICP was used to align the current lidar scan to detailed prior maps, collected using a commercial laser mapping system. For an in-depth discussion on ground truth generation the reader is referred to [7].

B. Results

Table II summarizes the mean Relative Pose Error (RPE) over a distance of 10 m for the following algorithms:

- **VILENS-LI:** VILENS with IMU and lidar only;
- **VILENS-LVI:** VILENS with IMU, visual (with lidar depth), and lidar features;



Fig. 7. *Top:* The Newer College dataset [7] – challenges include large open spaces, dense foliage without clear structure, and large illumination changes from sunlight to shadow. *Bottom:* The DARPA SubT dataset – challenges include long straight corridors and low light conditions (the bottom images were manually enhanced to show the content).

Relative Pose Error (RPE) – Translation [m] / Rotation [°]			
SubT Challenge Urban (ANYmal B300)			
Data	**VILENS-LI**	**VILENS-LVI**	**LOAM**
ST-A	0.28 / 2.65	**0.14 / 1.04**	0.20 / 1.32
ST-B	0.20 / 1.75	**0.12 / 0.79**	0.22 / 0.99
Newer College (Handheld Device [7])			
Data	**VILENS-LI**	**VILENS-LVI**	**LeGO-LOAM**
NC-1	0.50 / 3.23	0.39 / **2.45**	**0.34** / 3.45
NC-2	0.78 / 2.38	**0.43 / 1.85**	12.32 / 55.00
NC-3	**0.18 / 3.12**	0.20 / 3.61	1.85 / 23.67

TABLE II
EXPERIMENTAL RESULTS

- **LOAM:** The output of the LOAM [1] mapping module used during the SubT competition.
- **LeGO-LOAM:** The output of the LeGO-LOAM [2] mapping module.

It should be noted that no loop closures have been performed, and in contrast to both LOAM and LeGO-LOAM methods we do not perform any mapping.

For the SubT datasets, VILENS-LVI outperforms LOAM in translation / rotation by an average of 38% / 21% and VILENS-LI by 46% / 21%. An example of the global performance is shown in Fig. 8, which depicts both the estimated and ground truth trajectories on the ST-A dataset. VILENS-LVI is able to achieve very slow drift rates, even without a mapping system or loop closures.

For the least dynamic NC dataset, NC-1, VILENS-LVI achieves comparable performance to LeGO-LOAM. However, For the more dynamic datasets (up to 3.37 rad/s), NC-2 and NC-3, the VILENS methods significantly outperform LeGO-LOAM. Key to this performance is the undistortion of the lidar cloud to the camera timestamp, allowing accurate visual feature depth-from-lidar, while minimizing computation.

Overall, the best performing algorithm was VILENS-LVI, showing how the tight integration of visual and lidar features allows us to avoid failure modes that may be present in lidar-inertial only methods.

C. Multi-Sensor Fusion

A key benefit arising from the tight fusion of complementary sensor modalities is a natural robustness to sensor degradation.

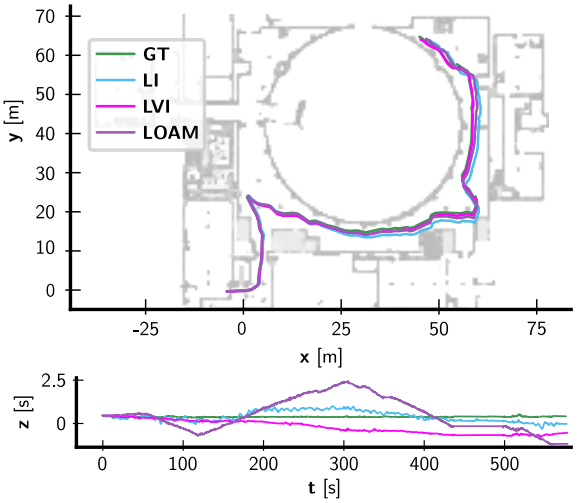


Fig. 8. Aerial view and elevation over time on the **ST-A** dataset, showing the estimated trajectory with Lidar-Inertial (blue), Lidar-Visual-Inertial (magenta), and LOAM [1] against the ground truth (green). Note that there are no loop-closures present in this system.

While a significant proportion of the datasets presented favorable conditions for both lidar and visual feature tracking, there were a number of scenarios where the tight fusion enabled increased robustness to failure modes of individual sensors.

Fig. 9 shows an example from the NC-2 where the camera auto-exposure feature took ∼3 s to adjust when moving out of bright sunlight into shade. During this time the number of visual features drops from around 30 to less than 5 (all clustered in a corner of the image). This would cause instability in the estimator. By tightly fusing the lidar, we are able to use the small number of visual features and the lidar features, without causing any degradation in performance. This is in contrast to methods such as [5], [15] where the use of separate visual-inertial and lidar-inertial subsystems mean that degenerate situations must be explicitly handled.

Similarly, in cases where the lidar landmarks are not sufficient to fully constrain the estimate (or are close to degenerate), the tight fusion of visual features allow the optimisation to take advantage of the lidar constraints while avoiding problems with degeneracy.

D. Analysis

A key benefit from using light-weight point cloud primitives in the optimisation is improved efficiency. The mean computation times for the above datasets are ∼10 ms for visual feature tracking, ∼50 ms for point cloud feature tracking, and ∼20 ms for optimization on a consumer grade laptop. This enables the system to output at 10 Hz (lidar frame rate) when using lidar-inertial only, and 15 Hz (camera keyframe rate) when fusing vision, lidar, and inertial measurements.

VII. CONCLUSION

We have presented a novel factor graph formulation for state estimation that tightly fuses camera, lidar, and IMU

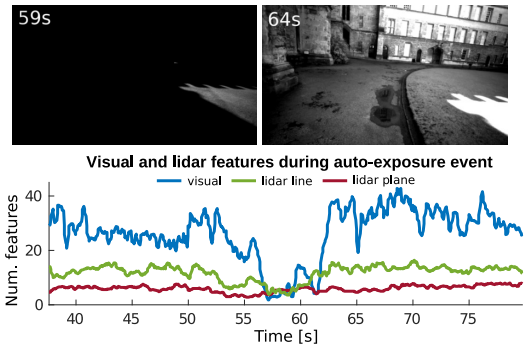


Fig. 9. *Top:* The Newer College (**NC-2**) contains sections with dramatic exposure change from underexposure (left) to more balanced exposure (right). *Bottom:* During this auto-exposure adjustment (57 s to 62 s) the number of visual features (blue) decreases almost to zero, while the lidar plane (red) and line (green) feature count remains relatively constant.

measurements. This fusion enables for graceful handling of degenerate modes – blending between lidar-only feature tracking and visual tracking (with lidar depth), depending on the constraints which each modality can provide in a particular environment. We have demonstrated comparable performance to state-of-the-art lidar-inertial odometry systems in typical conditions and better performance in extreme conditions, such as aggressive motions or abrupt light changes. Our approach also presents a novel method of jointly optimizing lidar and visual features in the same factor graph. This allows for robust estimation in difficult environments such as long corridors, and dark environments.

References

[1] J. Zhang and S. Singh, "Low-drift and real-time lidar odometry and mapping," *Autonomous Robots*, vol. 41, no. 2, pp. 401–416, Feb 2017.

[2] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 4758–4765.

[3] P. Geneva, K. Eckenhoff, Y. Yang, and G. Huang, "LIPS: LiDAR-inertial 3D plane SLAM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 123–130.

[4] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3D lidar inertial odometry and mapping," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 3144–3150.

[5] M. Camurri, M. Ramezani, S. Nobili, and M. Fallon, "Pronto: A multi-sensor state estimator for legged robots in real-world scenarios," *Frontiers in Robotics and AI*, vol. 7, pp. 1–18, 2020.

[6] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. A. Hoepflinger, "ANYmal - A Highly Mobile and Dynamic Quadrupedal Robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 38–44.

[7] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The Newer College Dataset: Handheld LiDAR, inertial and vision with ground truth," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[8] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: low-drift, robust, and fast," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 2174–2181.

[9] Z. Wang, J. Zhang, S. Chen, C. Yuan, J. Zhang, and J. Zhang, "Robust high accuracy visual-inertial-laser slam system," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 6636–6641.

[10] W. Shao, S. Vijayarangan, C. Li, and G. Kantor, "Stereo visual inertial lidar simultaneous localization and mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 370–377.

[11] D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robotics and Automation Letters*, pp. 4507–4514, 2019.

[12] D. Wisth, M. Camurri, and M. Fallon, "Preintegrated velocity bias estimation to overcome contact nonlinearities in legged robot odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 392–398.

[13] W. S. Grant, R. C. Voorhies, and L. Itti, "Efficient Velodyne SLAM with point and plane features," *Autonomous Robots*, vol. 43, no. 5, pp. 1207–1224, 2019.

[14] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[15] S. Khattak, H. Nguyen, F. Mascarich, T. Dang, and K. Alexis, "Complementary multi-modal sensor fusion for resilient robot pose estimation in subterranean environments," in *International Conference on Unmanned Aircraft Systems*, 2020, pp. 1024–1029.

[16] J. Graeter, A. Wilczynski, and M. Lauer, "LIMO: Lidar-monocular visual odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 7872–7879.

[17] Y. Yang, P. Geneva, X. Zuo, K. Eckenhoff, Y. Liu, and G. Huang, "Tightly-coupled aided inertial navigation with point and plane features," in *International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6094–6100.

[18] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, "LIC-Fusion: LiDAR-inertial-camera odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 5848–5854.

[19] X. Zuo, Y. Yang, P. Geneva, J. Lv, Y. Liu, G. Huang, and M. Pollefeys, "LIC-Fusion 2.0: LiDAR-inertial-camera odometry with sliding-window plane-feature tracking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[20] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 809–816.

[21] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 3565–3572.

[22] F. Dellaert and M. Kaess, "Factor graphs for robot perception," *Foundations and Trends in Robotics*, vol. 6, pp. 1–139, Aug. 2017.

[23] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[24] F. Dellaert, "Derivatives and differentials," 2020. [Online]. Available: https://github.com/borglab/gtsam/blob/develop/doc/math.pdf

[25] C. J. Taylor and D. J. Kriegman, "Minimization on the Lie Group SO(3) and Related Manifolds," Yale University, Tech. Rep. 9405, 1994.

[26] K. MacTavish and T. D. Barfoot, "At All Costs: A comparison of robust cost functions for camera correspondence outliers," in *Conference on Computer and Robot Vision*, 2015, pp. 62–69.

[27] F. Neuhaus, T. Koß, R. Kohnen, and D. Paulus, "MC2SLAM: Real-Time Inertial Lidar Odometry Using Two-Scan Motion Compensation," in *German Conference on Pattern Recognition*, T. Brox, A. Bruhn, and M. Fritz, Eds. Springer International Publishing, 2018, pp. 60–72.

[28] C. Le Gentil, T. Vidal-Calleja, and S. Huang, "In2laama: Inertial lidar localization autocalibration and mapping," *IEEE Transactions on Robotics*, vol. 37, no. 1, pp. 275–290, 2 2021.

[29] M. Bosse, R. Zlot, and P. Flick, "Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1104–1119, 2012.

[30] I. Bogoslavskyi and C. Stachniss, "Fast range image-based segmentation of sparse 3D laser scans for online operation," *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 163–169, 2016.

[31] O. Chum and J. Matas, "Matching with PROSAC - progressive sample consensus," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[32] F. Tschopp, M. Riner, M. Fehr, L. Bernreiter, F. Furrer, T. Novkovic, A. Pfrunder, C. Cadena, R. Siegwart, and J. Nieto, "VersaVIS–An Open Versatile Multi-Camera Visual-Inertial Sensor Suite," *Sensors*, vol. 20, no. 5, p. 1439, 2020.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	Unified Multi-Modal Landmark Tracking for Tightly Coupled Lidar-Visual-Inertial Odometry
Publication Status	Published
Publication Details	D. Wisth, M. Camurri, S. Das, and M. Fallon, "Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1004-1011, 2021.

Student Confirmation

Student Name:	David Wisth		
Contribution to the Paper	My contributions to the paper were: • Developed the core idea behind the paper with co-authors • Implemented the algorithm • Assisted Dr Marco Camurri implementing lidar undistortion. • Performed the experiments with co-authors • Generated experimental results and ground truth • Wrote the paper with co-authors		
Signature		Date	30/09/2021

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Maurice Fallon		
Supervisor comments This is an accurate desciption		
Signature 	Date	4/10/2021

This completed form should be included in the thesis, at the end of the relevant chapter.

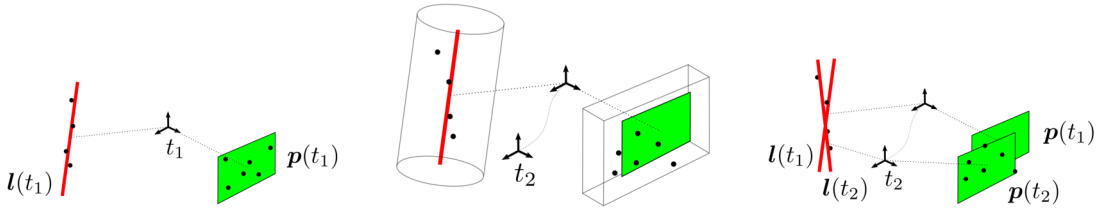
5.2 Extensions

This section will present additional details that were not published in the original paper due to space constraints. Section 5.2.1 provides an in-depth description of the point cloud feature tracking method, and Section 5.2.2 describes an extension to the line and plane factors to support a local linearisation point. This extension is also mentioned in the follow-up work in Chapter 6.

The Pose-to-Point Factor from Eq. (6) of the above article and the Local Plane Factor described below have been open-sourced and are available in the official GTSAM codebase[1] as the `PoseToPointFactor` and `LocalOrientedPlane3Factor`, respectively.

5.2.1 Point Cloud Feature Tracking

This section expands on the point cloud feature tracking framework described in Section V.B of the paper presented in Section 5.1. This information could not be included in the original paper due to page limitations.



(a) Step 1: Geometric features (planes (*green*) and lines (*red*)) are detected using PROSAC [147].

(b) Step 2: In the next scan, a local search is performed around the predicted feature location.

(c) Step 3: If the difference between features is less than a threshold they are considered a match.

Figure 5.1: In contrast to the LOAM approach illustrated in Figure 3.2, VILENS adopts a *sparse* approach, tracking individual point cloud features over time. This is motivated by the success of local visual feature tracking methods.

Most other algorithms in the literature, perform either semi-dense (e.g., scan-to-scan or scan-to-map matching) or dense matching (e.g., ICP). Figure 3.2 shows an example of how LOAM [3] used a frame-to-frame feature matching approach and did not track individual features over time.

[1] `https://github.com/borglab/gtsam`

Algorithm 1: Point Cloud Feature Tracking

Input: Lidar point cloud $\mathcal{C}_{input}$, Previous planes $\boldsymbol{P}_{i-1}$, Previous lines $\boldsymbol{L}_{i-1}$, Predicted frame-to-frame transform $\mathbf{T}_{i-1,i}$

Output: Tracked planes $\boldsymbol{P}_i$, Tracked lines $\boldsymbol{L}_i$

$\mathcal{C}_{filtered} \leftarrow$ segmentationBasedFilter($\mathcal{C}_{input}$)
$\mathcal{C}_P,\ \mathcal{C}_L \leftarrow$ curvatureBasedFilter($\mathcal{C}_{filtered}$)
foreach $(\mathcal{C}_F,\ \mathbf{F}_{i-1},\ \mathbf{F}_i)$ *in* $\{(\mathcal{C}_P,\ \boldsymbol{P}_{i-1},\ \boldsymbol{P}_i),\ (\mathcal{C}_L,\ \boldsymbol{L}_{i-1},\ \boldsymbol{L}_i)\}$ **do**
    $\mathcal{C}_{remaining} \leftarrow C_F$
    foreach $\mathbf{f}_{i-1}$ *in* $\mathbf{F}_{i-1}$ **do**
        $\hat{\mathbf{f}}_i \leftarrow$ predictFeatureLocation($\mathbf{f}_{i-1},\ \mathbf{T}_{i-1,i}$)
        $\mathcal{C}_{segmented} \leftarrow$ segmentCloudAroundFeature($\mathcal{C}_{remaining},\ \hat{\mathbf{f}}_i$)
        $\mathbf{f}_i \leftarrow$ fitFeatureToCloud($\mathcal{C}_{segmented}$)
        if $\|\mathbf{f}_{i-1} \ominus \mathbf{f}_i\| < \beta$ **then**
            $\mathbf{F}_i$.append($\mathbf{f}_i$)
            $\mathcal{C}_{remaining} \leftarrow$ removePointsFromCloud($\mathcal{C}_{segmented},\ \mathcal{C}_{remaining}$)
    $\{\mathcal{C}_{clusters}\} \leftarrow$ euclideanClustering($\mathcal{C}_F$)
    foreach $\mathcal{C}_C$ *in* $\{\mathcal{C}_{clusters}\}$ **do**
        $\mathbf{f}_i \leftarrow$ fitFeatureToCloud($\mathcal{C}_C$)
        if $N_{inliers} > \beta$ **then**
            $\mathbf{F}_i$.append($\mathbf{f}_i$)

Instead, this thesis directly tracks point cloud features using a method that is heavily inspired by local visual feature tracking approaches, such as KLT. The aim is to reduce odometry drift by observing and tracking the same landmarks over time.

Figure 5.1 and Algorithm 1 describe the lidar feature tracking process. First, the lidar point clouds are preprocessed using segmentation and curvature based filters to obtain the candidate line and plane clouds, $\mathcal{C}_L$ and $\mathcal{C}_P$, respectively.

Then planes and lines are tracked separately. To simplify the notation the variable $\mathbf{f}$ will be used to represent a line or plane feature. VILENS first attempts to track previous features $\mathbf{f}_{i-1}$ in the current cloud C_F by searching a local area around the IMU-predicted feature location (see Figure 5.1b). Features are considered a match if the difference between the previous ($\mathbf{f}_{i-1}$) and current ($\mathbf{f}_i$) feature is less than a threshold β. Each successfully matched feature $\mathbf{f}_i$ is added to the list of tracked features $\mathbf{F}_i$. The points used in a successful feature detection $\mathbf{f}_i$ are removed from the cloud to avoid duplicate landmark tracking.

After tracking existing features, new features are detected in the remaining point cloud using a PROSAC-based method [147]. This process is enhanced using Euclidean and normal-based clustering to group spatially similar feature points and remove outliers.

This algorithm is repeated at each timestep to track plane and line landmarks for extended periods of time. This algorithm is efficient and can run in real-time using only on-board computation.

5.2.2 Improving Plane and Line Factors

After the publication of the paper presented in Section 5.1, the plane and line features were tested on much larger datasets (e.g., where the travelled distance was over 1 km). This sometimes created numerical instability which caused the GTSAM optimiser to fail. This section will improve the plane factor (PF) from Section 5.1 by introducing a new local plane factor (LPF). Note that the same process can be followed for the line factor and local line factor which will be presented in Chapter 6.

Background

VILENS uses Gauss-Newton (GN) optimisation to solve the factor graph. Each step of the GN algorithm requires inverting the Hessian matrix, which is defined as,

$$H = J^T \Sigma^{-1} J \tag{5.1}$$

where J is the Jacobian and Σ is the covariance. If the Hessian is full-rank and invertible, the optimiser can converge to a minimum. In comparison, if the Hessian is *ill-conditioned* (i.e., the *condition number* is very far away from 1), there may be issues with numerical stability. The condition number is defined as the ratio of the largest to smallest singular value,

$$C = \frac{\max(V)}{\min(V)} \tag{5.2}$$

where the values V are found using Singular Value Decomposition (SVD) [148].

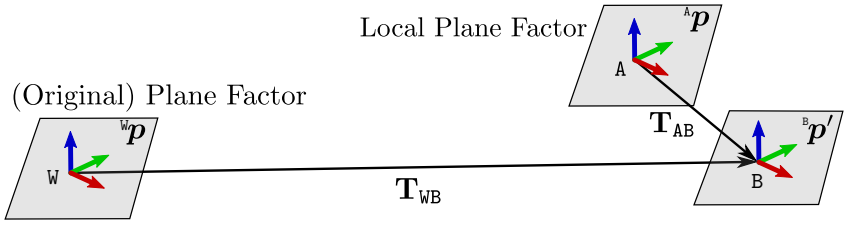


Figure 5.2: A illustration of the difference between the PF and proposed LPF. The PF depends on the global position of the robot $\mathbf{T}_{\mathtt{WB}}$, while the LPF only depends on the relative position to a local frame $\mathtt{A}$, i.e., $\mathbf{T}_{\mathtt{AB}}$.

Analysis of Plane Factors

The PF residual was defined in Equation (11) of the above paper [14] and is restated below for convenience,

$$\mathbf{r}_{x_i,\boldsymbol{p}_\ell} = \left(\mathbf{T}_{\mathtt{WB},i} \otimes {}^{\mathtt{W}}\boldsymbol{p}_\ell\right) \ominus {}^{\mathtt{B}}\tilde{\boldsymbol{p}}_i \tag{5.3}$$

where ${}^{\mathtt{B}}\tilde{\boldsymbol{p}}_i$ is the measured plane in the base frame, ${}^{\mathtt{W}}\boldsymbol{p}_\ell$ is the estimated plane in the world frame, and $\mathbf{T}_{\mathtt{WB},i}$ is the estimated world-to-base transformation at time t_i. $\mathbf{T}_{\mathtt{WB},i}$ is required to transform the estimated plane into the base frame before computing the difference. For brevity, the frame decorators will not be shown in the rest of this section.

To simplify the analysis of the PF, this section will examine a toy example where there is a single plane with a normal in the positive z direction, as illustrated in Figure 5.2. First, $\otimes$ is defined as the operator that transforms $\boldsymbol{p}$ by transformation matrix $\mathbf{T}$,

$$\mathbf{T} \otimes \boldsymbol{p} = (\mathbf{T}^{-1})^T \, [a, b, c, d]^T \tag{5.4}$$

$$\boldsymbol{p}' = \begin{bmatrix} r_{11} & r_{12} & r_{13} & x \\ r_{21} & r_{22} & r_{23} & y \\ r_{31} & r_{32} & r_{33} & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^T \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} \tag{5.5}$$

Substituting $\boldsymbol{p} = [0, 0, 1, d]$ (a horizontal plane with displacement d) and assuming that the rotation matrix describes a general YPR rotation (yaw α, pitch β, and roll γ),

$$p' = \mathbf{T} \otimes \boldsymbol{p}_\ell = \begin{bmatrix} -\sin(\beta) \\ \cos(\beta)\sin(\gamma) \\ \cos(\beta)\cos(\gamma) \\ -c_x x - c_y y - c_z z + d \end{bmatrix} \tag{5.6}$$

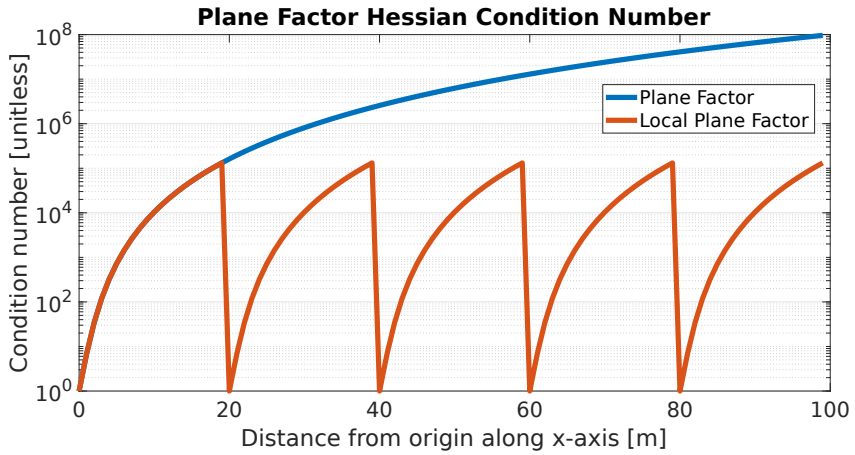


Figure 5.3: The condition number C measures the sensitivity of an output to a change in input (i.e., numerical stability). The ideal value for a perfectly balanced system is $C = 1$. The plot shows that the PF numerical stability *decreases* with distance from the origin (i.e., $C \propto d^4$, where d is the distance from the world to base frame, perpendicular to the plane normal). In contrast, the LPF maintains a bounded C by changing the local frame regularly (e.g., every 20 m), improving numerical stability.

where,

$$c_x = \cos(\alpha)\sin(\beta)\cos(\gamma) + \sin(\alpha)\sin(\gamma) \tag{5.7}$$

$$c_y = \sin(\alpha)\sin(\beta)\cos(\gamma) + \cos(\alpha)\sin(\gamma) \tag{5.8}$$

$$c_z = \cos(\beta)\cos(\gamma) \tag{5.9}$$

Examining the Jacobian of $\boldsymbol{p}'$ with respect to the transformation $\mathbf{T}$ evaluated at $\alpha = \beta = \gamma = 0$,

$$J = \nabla p'|_{\alpha=\beta=\gamma=0} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \tag{5.10}$$

The rows in the Jacobian matrix correspond to the partial derivatives of α, β, γ, x, y, z. It can be seen that some of the partial derivatives are proportional to x and y, i.e., $\left|\frac{\partial \boldsymbol{p}'}{\partial \beta}\right| \propto x$, $\left|\frac{\partial \boldsymbol{p}'}{\partial \gamma}\right| \propto y$. Intuitively, x and y displacements act as a "lever arm" that multiplies the effect of small angle changes in the normal.

Figure 5.3 shows how the condition number varies with the distance from the base to the origin. As the robot gets further from the origin (i.e., as x and y

increase), the Jacobian, and thus the Hessian, becomes more ill-conditioned, until the optimisation becomes numerically unstable.

Intuitively, the residual $\mathbf{r}_{x_i,\boldsymbol{p}_\ell}$ consists of two components: angular error of the normals, and displacement error of the plane. The derivative of the angular error is bounded and independent of the base position. In contrast, the derivative of the displacement error is *position-dependent*, since a small change in the normal in one frame, can cause a large change in displacement in another frame.

Solution

Since the numerical instability is caused by transforming the line or plane over a long distance, the LPF instead uses a local parametrisation, inspired by Kaess [112]. The proposed local plane factor residual is defined as,

$$\mathbf{r}_{x_i,\boldsymbol{p}_\ell} = \left((\mathbf{T}_{\mathtt{WA},i}^{-1}\mathbf{T}_{\mathtt{WB},i}) \otimes {}^{\mathtt{A}}\boldsymbol{p}_\ell\right) \ominus {}^{\mathtt{B}}\tilde{\boldsymbol{p}}_i \tag{5.11}$$

The additional local frame $\mathtt{A}$ is illustrated in Figure 5.2. This local frame reduces the displacement required and thus improves the condition number of the Hessian by a factor of $\sim 10^6$ when the robot is 100 m from the origin (see Figure 5.3).

The local frame $\mathtt{A}$ is tied to a pose in the factor graph. This is arbitrarily set as the pose when the plane is first detected, making the optimisation independent of the absolute location of the robot.

Overall, the local plane and line factors significantly improve numerical stability when the robot is far from the origin by using a local representation.

5.3 Discussion

This chapter presented a lidar-visual-inertial state estimator for mobile platforms, including legged robots. The experimental results demonstrated that visual and lidar feature tracking can outperform loosely coupled methods such as CompSLAM [11] (which switches between the outputs of LOAM [3], ROVIO [32], and TSIF [4]). The difference between approaches was most significant during highly dynamic motions. Additionally, when individual sensor modalities were degraded, the system

maintained accurate state estimation without requiring hard-switching. On the other hand, loosely-coupled approaches may be less sensitive to calibration or synchronisation issues since they process measurement streams separately.

The method presented in this chapter relies on extracting visual and geometric features from the environment, and thus implicitly assumes that the environment is suitably descriptive. It is also sensitive to environmental conditions; for example, lidars and cameras can be negatively affected by heavy rain, fog, or dusty environments [5].

The next chapter will present a combined method for robust estimation in all conditions by combining lidar, camera, kinematic, and inertial sensing.

Chapter 6

Visual, Inertial, Lidar, Kinematic State Estimation

This chapter extends the VILENS framework presented in Chapters 4 and 5. It combines visual, inertial, lidar, and kinematic measurements and presents several significant contributions. The first is a novel factor that fully integrates the kinematics into the cost function, as opposed to coming from an external filter. It also extends the camera model to support fisheye cameras, as well as integrates an ICP-based odometry module to complement lidar feature tracking. Finally, it presents extensive experimental evaluation, including online integration with a dynamic perceptive controller [149] and a local path planner [150]. The experimental results are based on over 2 h / 1.8 km of data from a variety of environments, including the DARPA Subterranean Challenge.

To the best of the author's knowledge, this is the first method to combine lidar, camera, kinematic, and inertial measurements in a factor graph framework.

6.1 VILENS: Visual, Inertial, Lidar, and Leg Odometry for All-Terrain Legged Robots

This manuscript was submitted to the IEEE Transactions on Robotics journal in July 2021 [15]. A video summary of this publication can be found at: `https://youtu.be/2318fiEB2cQ`.

IEEE TRANSACTIONS ON ROBOTICS 2022 1

VILENS: Visual, Inertial, Lidar, and Leg Odometry for All-Terrain Legged Robots

David Wisth, *Student Member, IEEE*, Marco Camurri, *Member, IEEE*, and Maurice Fallon, *Member, IEEE*
Oxford Robotics Institute, University of Oxford, Oxford, UK

***Abstract*—We present VILENS (Visual Inertial Lidar Legged Navigation System), an odometry system for legged robots based on factor graphs. The key novelty is the tight fusion of four different sensor modalities to achieve reliable operation when the individual sensors would otherwise produce degenerate estimation. To minimize leg odometry drift, we extend the robot's state with a linear velocity bias term which is estimated online. This bias is observable because of the tight fusion of this preintegrated velocity factor with vision, lidar, and IMU factors. Extensive experimental validation on different ANYmal quadruped robots is presented, for a total duration of 2 h and 1.8 km traveled. The experiments involved dynamic locomotion over loose rocks, slopes, and mud which caused challenges like slippage and terrain deformation. Perceptual challenges included dark and dusty underground caverns, and open and feature-deprived areas. We show an average improvement of 62 % translational and 51 % rotational errors compared to a state-of-the-art loosely coupled approach. To demonstrate its robustness, VILENS was also integrated with a perceptive controller and a local path planner.**

***Index Terms*—Legged robots, sensor fusion, localization, field robots.**

I. Introduction

THE increased maturity of quadruped robotics has been demonstrated in initial industrial deployments, as well as impressive results achieved by academic research. State estimation plays a key role in field deployment of legged machines: without an accurate estimate of its location and velocity, the robot cannot build up a useful representation of its environment, or plan and execute trajectories to reach desired goal positions.

Most legged robots are equipped with a high frequency (>250 Hz) proprioceptive state estimator for control and local mapping purposes. These are typically implemented as nonlinear filters fusing high frequency signals such as kinematics and IMU [1] [2] [3]. In ideal conditions (i.e., high friction, rigid terrain, slow speeds), these estimators have a limited (yet unavoidable) drift that is acceptable for local mapping and control.

However, deformable terrain, leg flexibility, and foot slippage can degrade estimation performance up to the point where local terrain reconstruction is unusable and multi-step trajectories cannot be executed, even over short ranges. This problem is more evident when a robot is moving dynamically, and can be the limiting factor when crossing rough terrain.

Manuscript received June 28, 2021; revised January 7, 2022. Corresponding author: D. Wisth (email: davidw@robots.ox.ac.uk).



Fig. 1. VILENS has been tested on a variety of platforms. *Top-Left:* ANYmal B300 at the Fire Service College in Moreton-On-Marsh (UK). *Top-Right:* ANYmal B300 modified for the DARPA SubT Challenge (Urban Circuit) in Olympia (Washington, USA) [10]. *Bottom-Left:* ANYmal C100 in a limescale mine in Wiltshire (UK). *Bottom-Right:*ANYmal C100 in an abandoned mine in Seemühle (Switzerland).

This estimate is also affected by modeling errors, such as inaccurate leg lengths or non-zero contact point size.

Recent works have attempted to improve kinematic-inertial estimation accuracy by reducing the convergence time using invariant observer design [3] or by detecting unstable contacts and reducing their influence on the overall estimation [4], [5], [6].

Other approaches have incorporated additional exteroceptive sensing into the estimator to help reduce the pose error. These included either tightly coupled methods fusing camera, IMU, and kinematics [7] or loosely coupled methods combining lidar in addition to the other sensors [8], [9]. These approaches model the contact locations as being fixed and affected only by Gaussian noise. Both assumptions are broken when there is nonrigid terrain, kinematic chain flexibility, or mild but repeated foot slippage. When these occur, fusion with exteroceptive sensors becomes nontrivial.

In our work, we aim to fuse all four sensor modalities (IMU, kinematics, lidar, camera) in a tightly coupled fashion, with particular focus on the proper integration of leg kinematics in presence of non-ideal contacts, when slippage or terrain deformation occurs.

A. Motivation

Our work is motivated by the challenges and limitations of state estimation for the deployment of legged robots in extreme environments, as illustrated in Fig. 1. The DARPA Subterranean (SubT) Challenge [10] involves the deployment of a team of autonomous robots navigating unknown, dark, and unstructured underground environments. In such scenarios, individual sensor modalities can fail (e.g., due to camera blackouts or degenerate geometries for lidar), so robust sensor fusion is paramount. Additional requirements include limited computational budget and the need for high frequency output to update the local footstep planner. For these reasons, we aim to use all of the sensors available on the robot (IMU, kinematics, lidar, and camera) to form constraints for lightweight sliding window optimization, as it can be more accurate than filter-based approaches [11].

When both lidar and camera fail at the same time, IMU integration alone would rapidly lead to divergence. Leg kinematics can prevent this, but special care has to be taken to fuse it with the other sensors in an effective way, especially in the presence of foot impacts and terrain deformation. Fig. 2 shows this effect on the ANYmal B300 robot traveling over various terrains. Two different kinematic-inertial estimators suffer from continuous elevation drift which is locally approximately linear (see dashed lines in Fig. 2). Fahmi *et al.* [12] demonstrated that this drift can be caused by nonrigid, and nonstatic interaction of the legs and terrain during contact events.

An example of a foot contact event on soft gravel is shown in Fig. 3. The contact point was non-static throughout the entire sequence, violating the key assumption of most leg odometry algorithms. This is mainly due to deformation of the ground and rubber foot, as well as the non-zero contact point size (the foot has a hemispherical profile). This can be considered as a systematic modeling error. Drift will be accumulated each time the robot steps on this terrain, leading to a biased estimate.

One approach would be to further model the dynamic properties of the robot, such as torque-dependent bending [13], or to model the terrain directly within the estimator. However, this would be robot specific and terrain dependent – improving performance in one situation but degrading it elsewhere. Additionally, threshold-based methods only reject the most significant slippage or deformation events and ignore the small error accumulated with each footstep.

Inspired by the IMU bias estimation and preintegration methods from [14], we instead propose to extend the state with a velocity bias term to estimate and reject these effects. This bias is observable when doing tight fusion with exteroceptive sensors (see Appendix C). This novel leg odometry factor computes a velocity measurement from kinematic sensing, preintegrates it, and estimates its bias to compensate for the characteristic drift of the leg odometry on slippery or deformable ground.

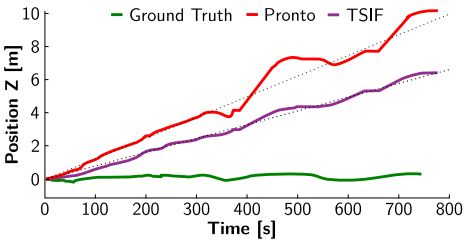


Fig. 2. Comparison between estimated robot elevation by Pronto [4] (red) and TSIF [2] (purple) kinematic-inertial state estimators, against ground truth (green) on the SMR experiment (see Section VI-B). Despite local fluctuations, the drift has a characteristic linear growth for a particular gait and terrain type. For example, between 350 s and 450 s the robot walks over soft gravel, increasing the drift rate.



(a) (b) (c) (d)

Fig. 3. An example of a foot contact sequence while trotting on gravel. After the foot touches the ground (a), both the terrain and robot's rubber foot deform as the controller increases the applied force (b). During the stance phase, the contact point changes as the foot rolls over its hemispherical profile (c) before finally breaking contact (d).

B. Contributions

This paper makes the following contributions, significantly extending our previous work [15]:

- A novel factor that incorporates joint kinematic measurements. In contrast to [15], this factor computes base velocities from the joint states, rather than taking them from an external filter. This avoids potential correlation due to double IMU usage and allows better modeling of the noise associated with the kinematics. A full derivation of the noise propagation is also provided in the Appendix.
- Support and testing on a wider range of sensor modalities, including fisheye cameras and lidar. To the best of our knowledge, this is the first smoothing algorithm to combine inertial, legged, visual, and lidar data into a single factor graph. In particular, in addition to the high frequency lidar feature tracking from [16], we also introduce a lidar odometry factor for low frequency accurate lidar registration. The wide support of complementary sensor modalities is the key to operation in adverse operational conditions, such as the DARPA SubT Challenge.
- Extensive evaluation in a broad set of scenarios including different models of the ANYmal quadruped. In contrast to [15], which was validated only on datasets, we tested the algorithm onboard the robot and integrated it with both a dynamic perceptive controller [17] and an autonomous exploration planner [18].

The remainder of this article is presented as follows: in Section II we review the literature on legged state estimation with a focus on sensor fusion via smoothing methods and

contact modeling; Section III formally defines the problem addressed by the paper and provides the required mathematical background; Section IV describes the factors used in our proposed graph formulation; Section V presents the implementation details of our system; Section VI presents the experimental results and discussion; Section VIII concludes with final remarks.

II. Related Work

We define multi-sensor state estimation as the joint processing of multiple proprioceptive and/or exteroceptive sensors to estimate the pose and velocity of a mobile robot [19]. Multi-sensor estimation can be divided into two broad categories: *filtering* and *smoothing*. Filtering methods such as MSC-KF [20] restrict the inference process to the latest state of the system, while smoothing methods also estimate all or part of the past states. In Section II-A we describe smoothing methods applied to multi-sensor estimation on legged robots. In Section II-B we review the relevant methods for modeling nonlinearities at the contact point during leg odometry.

A. Multi-Sensor Smoothing on Legged Robots

Multi-sensor smoothing for legged robots typically involves the fusion of IMU, leg odometry, and visual tracking within a probabilistic graphical model framework, such as factor graphs. Hartley *et al.* [21] proposed the first method to incorporate leg odometry into a factor graph. They extended the floating base state with the feet contact locations and defined two new factors to incorporate forward kinematics, where zero velocity constraints on the contact point of each foot were imposed. The two new factors were then combined with the preintegrated IMU factor from [14] and relative pose measurements from the SVO visual odometry system [22]. In [7], the same authors extended this work to support multiple footsteps in the same kinematics factor. Both works were demonstrated on the Cassie bipedal robot through experiments in controlled environments lasting $<2\,\mathrm{min}$.

Fourmy *et al.* [23] proposed a factor-graph based state estimation system for the Solo12 quadruped robot [24]. It fused both kinematic and dynamic information to estimate the base frame of the robot and performed online calibration of the offset between the base frame and center of mass. The authors claimed this was important for control since the center of mass of their robot was not known precisely from the CAD model. However, only very limited experimental results were presented, where the robot's torso moved while the feet were stationary. In the related field of motion planning, Xie *et al.* [25] modeled both the kinematics and dynamics of a 3 DoF manipulator using factor graphs. This approach took into account dynamics, contact forces, and joint actuation limits.

In our prior work [26], we proposed a tightly coupled visual-inertial-legged system based on the iSAM2 solver [27] running on the ANYmal robot. The method tracked visual features from a RealSense D435i stereo camera and optimized them as landmarks in a factor graph. Leg odometry was only loosely coupled with relative pose factors formed using the internal state estimator running on the robot [2]. This method was demonstrated through outdoor experiments in urban and industrial scenarios.

All these works were based on the assumption of a stationary point of foot contact. This assumption is violated every time there are slippages, or deformations of the leg and/or the ground. Contact detection methods can help to reject sporadic slippage or deformation events. However, when these occur regularly, they need to be modeled.

B. Modeling Contact Deformation and Slippage

In legged robotics, slippage and/or deformation have typically been addressed by assuming the contact location of a stance foot is entirely static throughout the stance period (yet affected by Gaussian noise). Thus, the main focus has been on detecting and ignoring the feet that are not in fixed contact with the ground. This is a relatively simple task when a foot is equipped with force/torque sensors. In this case, a high vertical component of the measured force would imply that the contact force is within the friction cone and therefore nonslipping. However, residual errors due to model uncertainty or deformation might persist. Fahmi *et al.* [12] have shown that incorrect contact detections on soft ground (i.e., detecting a "rigid" contact while the leg and ground are still deforming) are a key contributor to leg odometry drift.

Bloesch *et al.* [1] proposed an Unscented Kalman Filter design that fused IMU and differential kinematics. The approach used a threshold on the Mahalanobis distance of the filter innovation to infer velocity measurement outliers (caused by misclassified contact legs) which were then ignored. More recently, the idea was extended by Kim *et al.* [6] where the threshold on the Mahalanobis distance was replaced with a tunable threshold on the contact foot velocity estimated from the previous state.

For systems without feet sensors, more sophisticated methods were proposed. Hwangbo *et al.* [28] presented a probabilistic approach where information from kinematics, differential kinematics, and dynamics were fused within a Hidden Markov Model (HMM). This approach was later integrated with a dynamic trotting controller and demonstrated on the ANYmal robot walking on ice [5].

Instead of an HMM, Bledt *et al.* [29] proposed to fuse information from kinematics and dynamics, as well as additional input from the controller's gait cycle within a Kalman filter. Their work was demonstrated on the Cheetah 3 robot walking on rubble. Both [28] and [29] focused more on detecting the contact as early as possible for control purposes, rather than determining the contact periods that would minimize the state estimation error.

Camurri *et al.* [4] proposed a contact detector that learned the optimal force threshold to detect a foot in contact for a specific gait, and an impact detector that adapted the measurement covariance online to reject unreliable measurements.

In our previous work [15], we used this approach to fuse each leg's contribution into a single velocity measurement for our proposed factor graph method. The contact nonlinearities were modeled as a bias term on the linear velocity

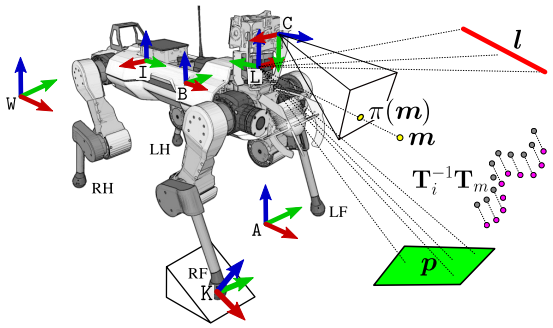


Fig. 4. Reference frames and landmark conventions. The world frame W is fixed to Earth. The base frame B, the camera's optical frame C, the lidar frame L, and the IMU frame I are all rigidly attached to the robot's chassis. The feet are conventionally named: Left Front (LF), Right Front (RF), Left Hind (LH), and Right Hind (RH). When a foot touches the ground (e.g., RF), a contact frame K (perpendicular to the ground and parallel to W's y-axis) is defined. The primitives tracked by the system are points $\boldsymbol{m}$, lines $\boldsymbol{l}$, and planes $\boldsymbol{p}$. To improve numerical stability, when a new plane feature is detected, an additional local fixed anchor frame A is defined. Finally, a relative pose factor (between times t_i and t_m in figure) is created using lidar registration.

measurements from leg odometry. This could reduce the inconsistency between leg and visual odometry and provide a more robust pose and velocity estimate. However, the system was still dependent on an external filter to access the velocity estimates from the kinematics. Furthermore, lidar, which is now a common sensor on quadruped robots, was not used for estimation.

In contrast to [15], in this work we compute the velocity measurements from kinematics internally, instead of receiving them from an external filter. This has several benefits: it eliminates the double usage of the IMU signal (from the filter and from the preintegrated IMU factor) which breaks the independent measurement assumption; it properly models the error propagation from the joints to the feet, thereby estimating the covariance from leg kinematics more accurately; it allows for better integration, as the optimized IMU and kinematics biases are directly accessible from the kinematics module, whereas the external filter did not. Finally, it simplifies the overall estimation architecture, eliminating the dependency on an external component. We also introduce a lidar registration factor with local submapping and tested our algorithm online and onboard the robot in conjunction with a perceptive controller and a local path planner (as opposed to operation in post processing).

III. Problem Statement

Our objective is to estimate the history of poses and velocities of a legged robot equipped with a combination of sensors, including: cameras (mono or stereo), IMUs, lidars, and joint sensors (encoders and torque sensors).

A. Notation

Scalars are lowercase italics $(a, b, c, \dots)$, matrices are uppercase Roman bold $(\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots)$, and vectors are lowercase Roman bold $(\mathbf{a}, \mathbf{b}, \mathbf{c}, \dots)$. Reference frames are indicated in typeface $(\mathtt{A}, \mathtt{B}, \mathtt{C}, \dots)$ and for physical quantities we follow the frame decorator rules from [30]. States and landmarks are bold italics $(\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \dots)$, while sensor measurements are uppercase calligraphics $(\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots)$. Finally, sets of time indices for measurements are uppercase sans $(\mathsf{A}, \mathsf{B}, \mathsf{C}, \dots)$. Where appropriate, a time-dependent quantity will be shortened as $a(t_i) = a_i$.

B. Frame Definitions

The relevant reference frames are specified in Fig. 4 and include: the fixed-world frame W, the base frame B, the IMU frame I, the camera frame C (left camera, when stereo), and the lidar frame L. When a foot is in contact with the ground, a contact frame K, fixed to Earth, is defined. Finally, a local anchor frame A, also fixed to Earth, is defined for lidar feature tracking, as detailed in Section IV-E.

C. State Definition

The robot state at time t_i is defined as follows:

$$\boldsymbol{x}_i \triangleq [\mathbf{R}_i, \mathbf{p}_i, \mathbf{v}_i, \mathbf{b}_i^g, \mathbf{b}_i^a, \mathbf{b}_i^\omega, \mathbf{b}_i^v] \in \mathrm{SO}(3) \times \mathbb{R}^{15} \tag{1}$$

where: $\mathbf{R}_i \in \mathrm{SO}(3)$ is the orientation; $\mathbf{p}_i \in \mathbb{R}^3$ is the position; $\mathbf{v}_i \in \mathbb{R}^3$ is the linear velocity; $\mathbf{b}_i^g, \mathbf{b}_i^a \in \mathbb{R}^3$ are the usual IMU gyroscope and accelerometer biases. We expand the state with angular and linear velocity biases $\mathbf{b}_i^\omega, \mathbf{b}_i^v \in \mathbb{R}^3$ to model slippage, deformations, and other kinematics inaccuracies at the contact point.

Unless otherwise specified, the position ${}_{\mathtt{W}}\mathbf{p}_{\mathtt{WB}}$ and orientation $\mathbf{R}_{\mathtt{WB}}$ of the base are expressed in world coordinates, velocities of the base ${}_{\mathtt{B}}\mathbf{v}_{\mathtt{WB}}, {}_{\mathtt{B}}\boldsymbol{\omega}_{\mathtt{WB}}$ are in base coordinates, IMU biases ${}_{\mathtt{I}}\mathbf{b}^g$, ${}_{\mathtt{I}}\mathbf{b}^a$ are expressed in the IMU frame, and the velocity biases are expressed in the base frame, ${}_{\mathtt{B}}\mathbf{b}_{\mathtt{WB}}{}^{\omega}$, ${}_{\mathtt{B}}\mathbf{b}_{\mathtt{WB}}{}^{v}$.

In addition to the robot state, we also estimate the position of visual and lidar landmarks. Visual landmarks $\boldsymbol{m}$ are parametrized as 3D points in Euclidean space and projected onto the image plane via the function $\pi(\cdot)$ (yellow dot in Fig. 4). Lidar landmarks are parametrized as planar and linear geometric primitives (green plane $\boldsymbol{p}$ and red line $\boldsymbol{l}$ in Fig. 4), as detailed in [16]. For brevity, we will later refer to any of the above landmarks with $\boldsymbol{f}$.

We define the history of states and landmarks, visible up to the current time t_k, as:

$$\mathcal{X}_k \triangleq \{\boldsymbol{x}_i, \boldsymbol{m}_\ell, \boldsymbol{p}_\ell, \boldsymbol{l}_\ell\}_{i \in \mathsf{K}_k, \ell \in \mathsf{F}_k} \tag{2}$$

where K_k and F_k are the sets of all keyframe and landmark indices, respectively.

D. Measurements Definition

We denote with $\mathcal{I}_{ij}$ the IMU measurements received between two consecutive keyframes i and j. Each measurement includes the proper acceleration $\tilde{\mathbf{a}}$ and the rotational velocity $\tilde{\boldsymbol{\omega}}$, both expressed in the IMU frame. Similarly, we define the kinematics measurements $\mathcal{K}_{ij}$ which include the joint positions $\tilde{\boldsymbol{\alpha}}$ and velocities $\tilde{\dot{\boldsymbol{\alpha}}}$. The (mono or stereo) camera images and lidar point clouds collected at time t_i are expressed with $\mathcal{C}_i$ and $\mathcal{L}_i$, respectively. In practice, camera and lidar measurements

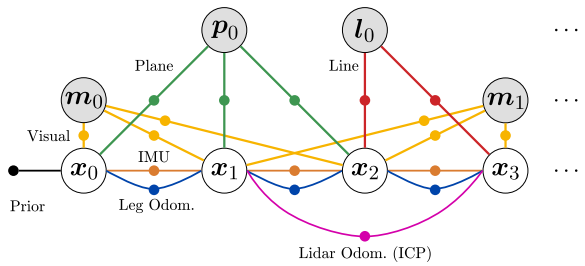


Fig. 5. VILENS factor graph structure. The factors are: prior (black), visual (yellow), lidar planes (green), lidar lines (red), preintegrated IMU (orange), preintegrated velocity (from leg kinematics, blue), and lidar odometry (from ICP registration, magenta). State nodes are white, while landmarks are grey.

are received at different times and frequencies, so they are first synchronized before being integrated into the graph (see Section V-A).

The full set of measurements within the smoothing window is defined as:

$$\mathcal{Z}_k \triangleq \{\mathcal{I}_{ij}, \mathcal{K}_{ij}, \mathcal{C}_i, \mathcal{L}_i\}_{i,j\in\mathsf{K}_k} \tag{3}$$

E. Maximum-A-Posteriori Estimation

We wish to maximize the likelihood of the measurements $\mathcal{Z}_k$ given the history of states and landmarks $\mathcal{X}_k$:

$$\mathcal{X}_k^\star = \arg\max_{\mathcal{X}_k} p(\mathcal{X}_k|\mathcal{Z}_k) \propto p(\mathcal{X}_0)p(\mathcal{Z}_k|\mathcal{X}_k) \tag{4}$$

Given that the measurements are assumed to be conditionally independent and corrupted by white Gaussian noise, Eq. (4) can be formulated as a least squares minimization problem of the following form:

$$\mathcal{X}_k^\star = \arg\min_{\mathcal{X}_k} \|\mathbf{r}_0\|^2_{\Sigma_0} + \sum_{i\in\mathsf{K}_k} \left(\|\mathbf{r}_{\mathcal{I}_{ij}}\|^2_{\Sigma_{\mathcal{I}_{ij}}} + \|\mathbf{r}_{\mathcal{K}_{ij}}\|^2_{\Sigma_{\mathcal{K}_{ij}}} + \|\mathbf{r}_{\mathbf{b}_{ij}}\|^2_{\Sigma_{\mathbf{b}_{ij}}} + \|\mathbf{r}_{\mathcal{L}_i}\|^2_{\Sigma_{\mathcal{L}_i}} + \sum_{\ell\in\mathsf{F}_i} \|\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{f}_\ell}\|^2_{\Sigma_{\boldsymbol{x}_i,\boldsymbol{f}_\ell}} \right) \tag{5}$$

where each term is the squared residual error associated to a factor type, weighted by the inverse of its covariance matrix, which will be detailed in Section IV. All the residuals (except the state prior $\mathbf{r}_0$) are added whenever a new keyframe i is created. These include: preintegrated IMU $\mathbf{r}_{\mathcal{I}_{ij}}$ and kinematic velocity $\mathbf{r}_{\mathcal{K}_{ij}}$, IMU and kinematic velocity biases $\mathbf{r}_{\mathbf{b}_{ij}}$, lidar odometry $\mathbf{r}_{\mathcal{L}_i}$, and landmark primitives $\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{f}_\ell}$. The latter are further divided into point, line, and plane residuals:

$$\sum_{\ell\in\mathsf{F}_i} \|\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{f}_\ell}\|^2_{\Sigma_{\boldsymbol{x}_i,\boldsymbol{f}_\ell}} = \sum_{\ell\in\mathsf{M}_i} \|\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{m}_\ell}\|^2_{\Sigma_{\boldsymbol{x}_i,\boldsymbol{m}_\ell}} + \sum_{\ell\in\mathsf{L}_i} \|\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{l}_\ell}\|^2_{\Sigma_{\boldsymbol{x}_i,\boldsymbol{l}_\ell}} + \sum_{\ell\in\mathsf{P}_i} \|\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{p}_\ell}\|^2_{\Sigma_{\boldsymbol{x}_i,\boldsymbol{p}_\ell}} \tag{6}$$

IV. Factor Graph Formulation

In this section, we describe the measurements, residuals and covariances of the factors which form the graph shown in Fig. 5. For convenience, we summarize the IMU factors from [14] in Section IV-A; our novel velocity factor is detailed in Section IV-B; Sections IV-C and IV-D describe the bias and stereo visual residuals, which are adapted from [14], [15] to include the velocity bias term and support for fisheye cameras, respectively. We also briefly introduce the lidar factor residuals from [16] in Section IV-E. Finally, the novel lidar registration residual is detailed in Section IV-F.

A. Preintegrated IMU Factors

As in [14], the IMU measurements are preintegrated to constrain the pose and velocity between two consecutive nodes of the graph, as well as provide high frequency state updates between them. This uses a residual of the form:

$$\mathbf{r}_{\mathcal{I}_{ij}} = \left[\mathbf{r}^\mathsf{T}_{\Delta\mathbf{R}_{ij}}, \mathbf{r}^\mathsf{T}_{\Delta\mathbf{v}_{ij}}, \mathbf{r}^\mathsf{T}_{\Delta\mathbf{p}_{ij}}\right] \tag{7}$$

where the individual elements of the residual are defined as:

$$\mathbf{r}_{\Delta\mathbf{R}_{ij}} = \mathrm{Log}\left(\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^g)\right)\mathbf{R}_i^\mathsf{T}\mathbf{R}_j \tag{8}$$

$$\mathbf{r}_{\Delta\mathbf{v}_{ij}} = \mathbf{R}_i^\mathsf{T}\left(\mathbf{v}_j - \mathbf{v}_i - \mathbf{g}\Delta t_{ij}\right) - \Delta\tilde{\mathbf{v}}_{ij}(\mathbf{b}_i^g, \mathbf{b}_i^a) \tag{9}$$

$$\mathbf{r}_{\Delta\mathbf{p}_{ij}} = \mathbf{R}_i^\mathsf{T}\left(\mathbf{p}_j - \mathbf{p}_i - \mathbf{v}_i\Delta t_{ij} - \frac{1}{2}\mathbf{g}\Delta t_{ij}^2\right) - \Delta\tilde{\mathbf{p}}_{ij}(\mathbf{b}_i^g, \mathbf{b}_i^a) \tag{10}$$

As explained in more detail in [14], this formulation does not require recomputation of the integration between the two keyframes connected by the factor every time the linearization point changes. This makes the fusion of the high frequency signal from the IMU and lower frequency measurements from camera and lidar computationally feasible.

The Eqs. (8) – (10) depend on the optimized states and preintegrated measurements $\Delta\tilde{\mathbf{R}}_{ij}, \Delta\tilde{\mathbf{v}}_{ij}, \Delta\tilde{\mathbf{p}}_{ij}$. For convenience, we report them here (for the incorporation of the bias update and other details, see [14]):

$$\Delta\tilde{\mathbf{R}}_{ij} = \prod_{k=i}^{j-1} \mathrm{Exp}\left((\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_i^g)\,\Delta t\right) \tag{11}$$

$$\Delta\tilde{\mathbf{v}}_{ij} = \sum_{k=i}^{j-1} \Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{a}}_k - \mathbf{b}_i^a)\,\Delta t \tag{12}$$

$$\Delta\tilde{\mathbf{p}}_{ij} = \sum_{k=i}^{j-1} \left[\Delta\tilde{\mathbf{v}}_{ik}\Delta t + \frac{1}{2}\Delta\tilde{\mathbf{R}}_{ik}\,(\tilde{\mathbf{a}}_k - \mathbf{b}_i^a)\,\Delta t^2\right] \tag{13}$$

These are associated to the corresponding preintegrated noises, which can be expressed in iterative form:

$$\begin{bmatrix}\delta\boldsymbol{\phi}_{i,k+1}\\ \delta\mathbf{v}_{i,k+1}\\ \delta\mathbf{p}_{i,k+1}\end{bmatrix} = \begin{bmatrix}\Delta\tilde{\mathbf{R}}_{k,k+1}^\mathsf{T} & \mathbf{0} & \mathbf{0}\\ -\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{a}}_k - \mathbf{b}_i^a)^\wedge\Delta t & \mathbf{I} & \mathbf{0}\\ -\frac{1}{2}\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{a}}_k - \mathbf{b}_i^a)^\wedge\Delta t^2 & \mathbf{I}\Delta t & \mathbf{I}\end{bmatrix} \begin{bmatrix}\delta\boldsymbol{\phi}_{i,k}\\ \delta\mathbf{v}_{i,k}\\ \delta\mathbf{p}_{i,k}\end{bmatrix} + \begin{bmatrix}\mathbf{J}_R^k\Delta t & \mathbf{0}\\ \mathbf{0} & \Delta\tilde{\mathbf{R}}_{ik}\Delta t\\ \mathbf{0} & \frac{1}{2}\Delta\tilde{\mathbf{R}}_{ik}\Delta t^2\end{bmatrix} \begin{bmatrix}\boldsymbol{\eta}^g\\ \boldsymbol{\eta}^a\end{bmatrix} \tag{14}$$

from which the covariance $\boldsymbol{\Sigma}_{\mathcal{I}_{ij}}$ can be computed.

B. Preintegrated Leg Odometry Factors

Inspired by the preintegrated IMU factor, we define here the preintegrated velocity factor for leg odometry. This factor is used to estimate the position of the robot from high frequency joint kinematics measurements.

To derive a velocity signal from leg kinematics, we first need to estimate the stance legs, i.e., the set of legs in stable contact with the ground (Section IV-B1). Then, we fuse the odometry from these legs as a combined velocity measurement (Section IV-B2). To model the nonlinearities at the contact point, the linear velocity bias and its residuals are introduced in Section IV-B3. The preintegrated measurements and their residuals are described in Sections IV-B4 and IV-B5, respectively. Finally, a special case of this twist bias is outlined in Section V-C.

1) Stance Estimation: The feet of the robot are approximated as points. For simplicity, the contact point is assumed to be on a fixed point at the center of the foot of the robot, which in our case is a soft rubber sphere with a $2\,\mathrm{cm}$ radius. Since no direct force measurement is assumed to be available, we compute the Ground Reaction Force (GRF) at each foot from the dynamics equation of motion:

$$\mathbf{f} = -\left(\mathbf{J}_p(\boldsymbol{\alpha})^\mathsf{T}\right)^\dagger \left(\boldsymbol{\tau} + \mathbf{h}(\boldsymbol{\alpha}) + F^\mathsf{T} \begin{bmatrix} \dot{\boldsymbol{\omega}} \\ \dot{\mathbf{v}} \end{bmatrix}\right) \tag{15}$$

where † indicates the Moore-Penrose pseudo inverse operation, $\boldsymbol{\alpha}$ are the joint positions, $\mathbf{J}_p(\cdot) \in \mathbb{R}^{3\times 3}$ is the Jacobian of the forward kinematics function relative to the point foot (later indicated with $\mathbf{f}_p(\cdot) \in \mathbb{R}^3$), $\boldsymbol{\tau}$ are the joint torques, $\mathbf{h}(\boldsymbol{\alpha})$ are the Coriolis terms, and F is the matrix of spatial forces required at the floating base to support unit accelerations about each joint variable [31]. In practice, both $\dot{\boldsymbol{\omega}}$ and $\dot{\mathbf{v}}$ are obtained from the IMU and projected into the base frame, where the angular acceleration is computed by numerical differentiation and the linear velocity is computed by compensating for the gravity using the latest orientation estimate.

For each leg that is in contact with the ground, we assume *rigid, nonslipping contact*. By thresholding the vertical component of the GRF for each foot, we get the set of binary contact states for all the legs. We indicate the subset of the legs in contact as $S \subseteq L$, with $L = \{\mathrm{LF}, \mathrm{RF}, \mathrm{LH}, \mathrm{RH}\}$ (Fig. 4).

2) Combined Kinematic Velocity Measurement: Given a leg $s \in S$, we can compute the linear velocity of the robot's floating base at time t_i as follows:

$$\mathbf{v}_s = -\mathbf{J}_p(\boldsymbol{\alpha})\dot{\boldsymbol{\alpha}} - \boldsymbol{\omega} \times \mathbf{f}_p(\boldsymbol{\alpha}) \tag{16}$$

Both the joint positions and velocities are measured from encoders and corrupted by additive zero-mean Gaussian noise:

$$\tilde{\boldsymbol{\alpha}} = \boldsymbol{\alpha} + \boldsymbol{\eta}^\alpha \tag{17}$$

$$\tilde{\dot{\boldsymbol{\alpha}}} = \dot{\boldsymbol{\alpha}} + \boldsymbol{\eta}^{\dot{\alpha}} \tag{18}$$

where $\boldsymbol{\eta}^\alpha, \boldsymbol{\eta}^{\dot{\alpha}}$ are available from the sensor specifications. After substituting Eqs. (17) and (18) into Eq. (16), we can formulate a linear velocity measurement [1]:

$$\mathbf{v}_s = -\mathbf{J}_p(\tilde{\boldsymbol{\alpha}} - \boldsymbol{\eta}^\alpha) \cdot (\tilde{\dot{\boldsymbol{\alpha}}} - \boldsymbol{\eta}^{\dot{\alpha}}) - \boldsymbol{\omega} \times \mathbf{f}_p(\tilde{\boldsymbol{\alpha}} - \boldsymbol{\eta}^\alpha) \tag{19}$$

As detailed in Appendix A, the noise terms from Eq. (19) can be separated, so the measurement can be expressed as:

$$\tilde{\mathbf{v}}_s = \mathbf{v}_s + \boldsymbol{\eta}^v_s \tag{20}$$

$$\boldsymbol{\eta}^v_s = -\left(\mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} + \boldsymbol{\omega}^\wedge \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\right)\boldsymbol{\eta}^\alpha - \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\dot{\alpha}} \tag{21}$$

where $\mathbf{H}_p(\cdot) \in \mathbb{R}^{3\times 3\times 3}$ is the Hessian of the forward kinematics function, $\mathbf{f}_p(\boldsymbol{\alpha})$.

Eq. (20) is valid only when the leg s is in contact with the ground. Since foot contacts happen intermittently while the robot walks, multiple legs can be in contact simultaneously. As each velocity measurement from Eq. (20) is associated with a Gaussian noise term $\boldsymbol{\eta}^v_s$, it would be possible to add to the graph one independent velocity measurement per stance leg. This would however unnecessarily increase the graph complexity, as a closed form for fusing the measurements from the stance legs into one can be computed instead [32]. Additionally, treating the leg velocities separately would require explicit handling of contact switching every time a new step is made to ensure all measurements are used [7].

From Eq. (21), since $\boldsymbol{\eta}^v_s$ is a linear combination of zero-mean Gaussians, it is also a zero-mean Gaussian with covariance $\boldsymbol{\Sigma}_s$. Therefore, a combined velocity measurement for all the stance legs can then be computed as a weighted average of the velocity measurements of each stance leg, based on the information matrix $\boldsymbol{\Omega}_s = \boldsymbol{\Sigma}_s^{-1}$:

$$\boldsymbol{\Sigma}_v = \left(\sum_{s\in S} \boldsymbol{\Omega}_s\right)^{-1} \tag{22}$$

$$\tilde{\mathbf{v}} = \boldsymbol{\Sigma}_v \sum_{s\in S} (\boldsymbol{\Omega}_s \tilde{\mathbf{v}}_s) \tag{23}$$

$$\boldsymbol{\eta}^v \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_v) \tag{24}$$

where the compound velocity measurement noise $\boldsymbol{\eta}^v$ is sampled from a zero-mean Gaussian with covariance $\boldsymbol{\Sigma}_v$.

We now have a linear velocity measurement[1] $\tilde{\mathbf{v}}$ and its noise $\boldsymbol{\eta}^v$ that can be used for the preintegrated velocity factor:

$$\tilde{\mathbf{v}} = \mathbf{v} + \boldsymbol{\eta}^v \tag{25}$$

3) Velocity Bias: On slippery or deformable terrains, the constraint from Eq. (16) will not be true, resulting in incorrect leg velocities, and thus drift in the final odometry estimate. In our experience, this velocity drift is locally constant and is gait- and terrain-dependent (see Sections I-A and V-C, and Figs. 2 and 3).

For these reasons, we relax Eq. (16) by adding a slowly varying bias term $\mathbf{b}^v$ to Eq. (25):

$$\tilde{\mathbf{v}} = \mathbf{v} + \mathbf{b}^v + \boldsymbol{\eta}^v \tag{26}$$

This term incorporates the characteristic drift caused by leg or terrain compression, slippage, and impacts occurring at the contact point.

[1]Not to be confused with the IMU preintegrated velocity measurement, $\Delta\tilde{\mathbf{v}}$.

4) Preintegrated Velocity Measurements: We derive the preintegrated linear velocity and noise only, as the preintegrated rotation measurement $\Delta\tilde{\mathbf{\Theta}}_{ij}$ and noise $\delta\boldsymbol{\theta}_{ij}$ have the same form as Eqs. (11) and (14):

$$\Delta\tilde{\mathbf{\Theta}}_{ij} = \prod_{k=i}^{j-1} \mathrm{Exp}\left((\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_i^{\omega})\,\Delta t\right) \tag{27}$$

$$\delta\boldsymbol{\theta}_{ij} = \sum_{k=i}^{j-1} \Delta\tilde{\mathbf{\Theta}}_{k+1,j}^{\mathsf{T}} \mathbf{J}_R^k \boldsymbol{\eta}^{\omega} \Delta t \tag{28}$$

The position at time $t_j = t_i + \Delta t$ is:

$$\mathbf{p}(t_j) = \mathbf{p}(t_i) + \int_{t_i}^{t_j} \mathbf{v}(\tau)\,\mathrm{d}\tau \tag{29}$$

Assuming constant velocity between t_i and t_j, we can iteratively calculate the position in discrete time domain form[2]:

$$\mathbf{p}_j = \mathbf{p}_i + \sum_{k=i}^{j-1} [\mathbf{R}_k(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v - \boldsymbol{\eta}_k^v)\Delta t] \tag{30}$$

From Eq. (30) a relative measurement can be obtained[3]:

$$\Delta\boldsymbol{\kappa}_{ij} = \mathbf{R}_i^{\mathsf{T}}(\mathbf{p}_j - \mathbf{p}_i) = \sum_{k=i}^{j-1} [\Delta\mathbf{R}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v - \boldsymbol{\eta}_k^v)\Delta t)] \tag{31}$$

With the substitution $\Delta\mathbf{R}_{ik} = \Delta\tilde{\mathbf{\Theta}}_{ik}\mathrm{Exp}\left(-\delta\boldsymbol{\theta}_{ik}\right)$ to include the preintegrated rotation measurements (from Eqs. (27) – (28)), and the approximation $\mathrm{Exp}\left(\boldsymbol{\phi}\right) \simeq \mathbf{I} + \boldsymbol{\phi}^{\wedge}$, Eq. (31) becomes:

$$\Delta\boldsymbol{\kappa}_{ij} \simeq \sum_{k=i}^{j-1} \left[\Delta\tilde{\mathbf{\Theta}}_{ik}(I - \delta\boldsymbol{\theta}_{ik}^{\wedge})(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v - \boldsymbol{\eta}_k^v)\Delta t\right] \tag{32}$$

By separating the measurement and noise components of Eq. (32) and ignoring higher order terms, we can define the *preintegrated leg odometry position measurement* $\Delta\tilde{\boldsymbol{\kappa}}_{ij}$ and *noise* $\delta\boldsymbol{\kappa}_{ij}$ as:

$$\Delta\tilde{\boldsymbol{\kappa}}_{ij} \triangleq \sum_{k=i}^{j-1} \left[\Delta\tilde{\mathbf{\Theta}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v)\Delta t\right] \tag{33}$$

$$\delta\boldsymbol{\kappa}_{ij} \triangleq \sum_{k=i}^{j-1} \left[\Delta\tilde{\mathbf{\Theta}}_{ik}\boldsymbol{\eta}_k^v\Delta t - \Delta\tilde{\mathbf{\Theta}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v)^{\wedge}\delta\boldsymbol{\theta}_{ik}\Delta t\right] \tag{34}$$

Note that both quantities still depend on the twist biases $\mathbf{b}^{\omega}, \mathbf{b}^v$. When these change, we would like to avoid the recomputation of Eq. (33). Given a small change $\delta\mathbf{b}$ such that $\mathbf{b} = \bar{\mathbf{b}} + \delta\mathbf{b}$, we use a first order approximation to find the new measurement, as done in [14]:

$$\Delta\tilde{\boldsymbol{\kappa}}_{ij}(\mathbf{b}^{\omega}, \mathbf{b}^v) \simeq \Delta\tilde{\boldsymbol{\kappa}}_{ij}(\bar{\mathbf{b}}^{\omega}, \bar{\mathbf{b}}^v) + \frac{\partial\Delta\tilde{\boldsymbol{\kappa}}_{ij}}{\partial\mathbf{b}^{\omega}}\delta\mathbf{b}^{\omega} + \frac{\partial\Delta\tilde{\boldsymbol{\kappa}}_{ij}}{\partial\mathbf{b}^v}\delta\mathbf{b}^v \tag{35}$$

[2]For simplicity, we keep the symbol $\boldsymbol{\eta}^v$ for the noise in the discrete domain.

[3]The variable name change is to avoid confusion with the position measurements and noise from the IMU factor.

5) Residuals: The factor residuals include rotation and translation:

$$\mathbf{r}_{\mathcal{K}_{ij}} = \left[\mathbf{r}_{\Delta\tilde{\mathbf{\Theta}}_{ij}}^{\mathsf{T}},\ \mathbf{r}_{\Delta\boldsymbol{\kappa}_{ij}}^{\mathsf{T}}\right] \tag{36}$$

$$\mathbf{r}_{\Delta\mathbf{\Theta}_{ij}} = \mathrm{Log}\left(\Delta\tilde{\mathbf{\Theta}}_{ij}(\mathbf{b}_i^{\omega})\right)\mathbf{R}_i^{\mathsf{T}}\mathbf{R}_j \tag{37}$$

$$\mathbf{r}_{\Delta\boldsymbol{\kappa}_{ij}} = \mathbf{R}_i^{\mathsf{T}}(\mathbf{p}_j - \mathbf{p}_i) - \Delta\tilde{\boldsymbol{\kappa}}_{ij}(\mathbf{b}_i^{\omega}, \mathbf{b}_i^v) \tag{38}$$

Since Eq. (37) has the same form as Eq. (8), when the same angular velocity measurements are used for both the IMU and leg odometry factors, the following relations hold:

$$\Delta\tilde{\mathbf{\Theta}}_{ij} = \Delta\tilde{\mathbf{R}}_{ij} \qquad \mathbf{b}^{\omega} = \mathbf{b}^g \qquad \delta\boldsymbol{\theta}_{ij} = \delta\boldsymbol{\phi}_{ij} \tag{39}$$

In this case, we can avoid double counting the IMU signal by setting the rotational residual $\mathbf{r}_{\Delta\mathbf{\Theta}_{ij}}$ to zero. This was not possible in [15], where an external filter was used.

6) Covariance: After simple manipulation of Eq. (34), the covariance of the residual $\mathbf{r}_{\mathcal{K}_{ij}}$ can be computed iteratively:

$$\mathbf{\Sigma}_{i,k+1}^{\mathcal{K}} = \mathbf{A}\mathbf{\Sigma}_{i,k}^{\mathcal{K}}\mathbf{A}^{\mathsf{T}} + \mathbf{B}\mathbf{\Sigma}_{\boldsymbol{\eta}}^{\mathcal{K}}\mathbf{B}^{\mathsf{T}} \tag{40}$$

where the first term evolves from an initial condition of $\mathbf{\Sigma}_{i,i}^{\mathcal{K}} = \mathbf{0}$, while the second term $\mathbf{\Sigma}_{\boldsymbol{\eta}}^{\mathcal{K}}$ is fixed and taken from sensor specifications. The complete derivation of the multiplicative terms $\mathbf{A}$ and $\mathbf{B}$ are detailed in Appendix B.

C. Bias Residuals

The bias terms are intended to change slowly and are therefore modeled as a Gaussian random walk. The residual term for the cost function is as follows:

$$\|\mathbf{r}_{\mathbf{b}_{ij}}\|^2_{\Sigma_{\mathbf{b}_{ij}}} \triangleq \|\mathbf{b}_j^g - \mathbf{b}_i^g\|^2_{\Sigma_{\mathbf{b}_{ij}^g}} + \|\mathbf{b}_j^a - \mathbf{b}_i^a\|^2_{\Sigma_{\mathbf{b}_{ij}^a}} + \|\mathbf{b}_j^v - \mathbf{b}_i^v\|^2_{\Sigma_{\mathbf{b}_{ij}^v}} \tag{41}$$

where the covariance matrices are determined by the expected rate of change of these quantities. In particular, $\mathbf{\Sigma}_{\mathbf{b}_{ij}^g}, \mathbf{\Sigma}_{\mathbf{b}_{ij}^a}$ are available from IMU specifications, while $\mathbf{\Sigma}_{\mathbf{b}_{ij}^v}$ depends on the drift rate of the leg odometry bias, which is found empirically.

D. Visual Factors

We use two main factors related to visual measurements. The first is a traditional reprojection error given by [26]:

$$\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{m}_\ell} = \begin{pmatrix} \pi_u^L(\mathbf{R}_i, \mathbf{p}_i, \boldsymbol{m}_\ell) - u_{i,\ell}^L \\ \pi_u^R(\mathbf{R}_i, \mathbf{p}_i, \boldsymbol{m}_\ell) - u_{i,\ell}^R \\ \pi_v(\mathbf{R}_i, \mathbf{p}_i, \boldsymbol{m}_\ell) - v_{i,\ell} \end{pmatrix} \tag{42}$$

where $(u^L, v), (u^R, v)$ are the pixel locations of the detected landmark. If only a monocular camera is available then the second element of Eq. (42) is not used.

The second factor uses the overlapping fields of view of the lidar and camera sensors (where applicable) to provide depth estimates for visual features, as described in [16].

In addition to the standard camera projective model, in this paper we also introduce support for fisheye cameras with equidistant distortion [33]. As demonstrated in [34], a high camera FoV allows for tracking of features for longer periods of time, but for large open areas the loss of pixel density becomes significant for a diagonal FoV of 180° or more.

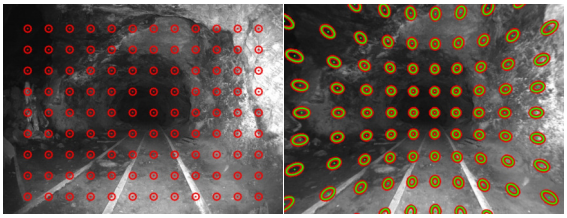


Fig. 6. Example of fisheye covariance warping and fitting. *Left:* Original samples (red points) and 1-σ bound of 9 pixels (red circles) in the distorted image. *Right:* the undistorted points from the 1-σ bound (red) and their ellipse fit (green) in the undistorted image. Notice that the undistorted covariance becomes larger near the edge of the image where there is greater distortion.

To maximize versatility in underground scenarios, our fisheye configurations have a moderately large diagonal FoV of 150–165° (Table I).

To add the landmark to the graph, we first detect and track features in the original, distorted image to avoid costly image undistortion. We then undistort the individual feature locations before adding them to the factor graph using Eq. (42).

As shown in Fig. 6, the distortion far from the focal point can be quite large. This means the angular resolution of the camera varies across the image. To correctly model the visual feature uncertainty, $\mathbf{\Sigma}_{\boldsymbol{x}_i,\boldsymbol{m}_\ell}$, we first select a set of points around the landmark in the distorted image, based on the specified visual feature tracker uncertainty. We then undistort these points and use a least-squares method to fit an ellipse to these points. For ellipse fitting we use at least 6 points since the covariance ellipse is, in general, is not aligned to the x or y axes. This gives us an accurate uncertainty matrix in the undistorted image coordinates. Since the distortion is constant, the undistortion map and covariance ellipses can be precomputed for efficiency, and thus have no impact on runtime performance.

We use $\Sigma_{\boldsymbol{x}_i,\boldsymbol{m}_\ell}$ of between 1 and 2 pixels (adapted by the fisheye covariance warping if applicable).

E. Plane and Line Factors

We extract and track geometric primitives, specifically planes and lines, over successive lidar scans. This is similar to how visual features are tracked. Planes are defined using the Hessian norm form [16]:

$$\boldsymbol{p} = \left\{ \langle \hat{\mathbf{n}}, d \rangle \in \mathbb{R}^4 \mid \hat{\mathbf{n}} \cdot (x, y, z) + d = 0 \right\} \tag{43}$$

while lines use the minimal parametrization from [35]:

$$\boldsymbol{l} = \left\{ \langle \mathbf{R}, (a, b) \rangle \in \mathrm{SO}(3) \times \mathbb{R}^2 \right\} \tag{44}$$

In this paper, we extend the plane and line factors from [16] to support local linearization points. This is important in large scale environments, where the transformation of the plane from world to sensor coordinates can cause numerical instability in the optimization (small changes in the angle of the normal can cause very large changes in the position of the plane or line). This instability increases as the sensor gets further from the origin of the world frame.

Therefore, inspired by [36], we introduce a local linearization frame for planes and lines which we call the anchor frame, $\mathtt{A}$. This both decreases convergence time [36] and increases numerical stability. $\mathtt{A}$ is arbitrarily defined at the first frame in which the landmark is observed.

When a plane $\tilde{\boldsymbol{p}}_\ell$ is measured at time t_i, the corresponding residual is the difference between $\tilde{\boldsymbol{p}}$ and the estimated plane $\boldsymbol{p}_\ell$ is transformed into the local reference frame:

$$\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{p}_\ell} = \left(\left(\mathbf{T}_{\mathtt{WA}}^{-1} \mathbf{T}_{\mathtt{WB}} \right) \otimes \boldsymbol{p}_\ell \right) \ominus \tilde{\boldsymbol{p}}_\ell \tag{45}$$

where $\mathbf{T}_{\mathtt{WA}}$ is the pose of the robot at the time where the plane is first detected, $\mathbf{T}_{\mathtt{WB}} = [\mathbf{p}_i, \mathbf{R}_i]$ is the current pose estimate of the robot.

Similarly, the residual between a measured line $\tilde{\boldsymbol{l}}_i$ and its prediction is defined as follows:

$$\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{l}_\ell} = \left(\left(\mathbf{T}_{\mathtt{WA}}^{-1} \mathbf{T}_{\mathtt{WB}} \right) \boxtimes \boldsymbol{l}_\ell \right) \boxminus \tilde{\boldsymbol{l}}_\ell \tag{46}$$

where $\otimes$, $\boxtimes$ apply a transformation and $\ominus$, $\boxminus$ are difference operators to planes and lines, respectively [16].

The line and plane covariances $\mathbf{\Sigma}_{\boldsymbol{x}_i,\boldsymbol{l}_\ell}$, $\mathbf{\Sigma}_{\boldsymbol{x}_i,\boldsymbol{p}_\ell}$ are determined by analysing the covariance of the inlier points for each feature. This results in typical covariances of 10 cm and 3 deg.

F. Lidar Registration Factor

The lidar feature tracking allows for continuous motion estimation at the full lidar frame rate. Approaches like [37] instead accumulate the features into a local submap for a certain number of frames before integrating them into the factor graph. Local submapping allows for accurate pose estimation refinement, but at a lower frequency. In this work, in addition to the feature tracking, we also integrate fine-grained lidar registration with local submapping. The registration is based on the Iterative Closest Point (ICP) approach by Pomerleau *et al.* [38]. To make the registration more robust, a local submap is maintained with all the scans successfully registered in the past 5 m travelled. The submaps constitutes the reference cloud of the ICP registration process.

Although ICP typically runs at a lower frequency than the other signals, it is able to take advantage of the high quality motion prior provided by the other modules that run at higher frequency (e.g., IMU) to motion correct scans.

ICP odometry measurements are added into the factor graph as relative pose factors between nonconsecutive keyframes i and m (e.g., nodes 1 and 3 in Fig. 5):

$$\mathbf{r}_{\mathcal{L}_i} = \Phi(\widetilde{\mathbf{T}}_i^{-1} \widetilde{\mathbf{T}}_m, \mathbf{T}_i^{-1} \mathbf{T}_m) \tag{47}$$

where $\mathbf{T}$ is the estimated pose, $\widetilde{\mathbf{T}}$ is the estimate from the ICP module, and Φ is the lifting operator defined in [14]. Note that ICP registration is prone to failure in environments with degenerate geometries, such as long tunnels. For this reason, a robust cost function is used to reject the factor in such degenerate situations.

For simplicity, the lidar registration covariance $\mathbf{\Sigma}_{\mathcal{L}}$ is set as a constant value, determined empirically during preliminary experiments and left unchanged for all our experiments, with potential outliers being handled by the robust cost function.

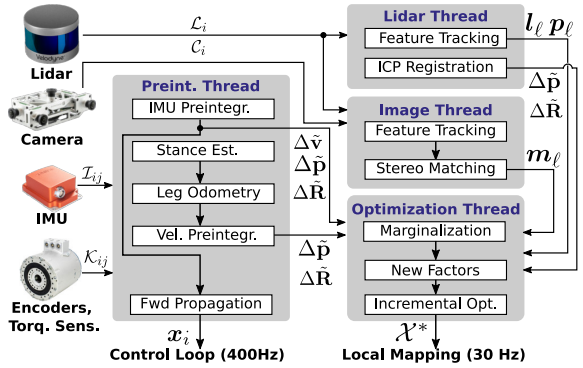


Fig. 7. Block diagram of the VILENS algorithm. Three independent threads (preintegration, camera, and lidar) process the data while a fourth thread performs the optimization on the data already processed by the other threads.

V. Implementation

The architecture of the VILENS state estimator is shown in Fig. 7. Three parallel threads process the preintegration of IMU and kinematics, camera feature tracking, lidar processing (both feature tracking and ICP registration) and a fourth carries out the subsequent optimization. A forward propagated state from the IMU factor is output from the preintegration thread at IMU frequency (i.e., $400\,\mathrm{Hz}$). This is used to motion correct and synchronize the lidar point clouds and is available for high frequency tasks such as control (for simplicity, we assume $\mathcal{L}_i$ is already undistorted in Fig. 7). When a new keyframe is processed, the preintegrated measurements and tracked landmarks are collected by the optimization thread, while the other threads process the next set of measurements. When the optimization step is complete, the optimal set of states is produced at the keyframe rate for use by local mapping and path planning.

The factor graph is solved using a fixed lag smoothing framework based on the efficient incremental optimization solver iSAM2 [27], implemented as part of the GTSAM library [11]. For these experiments, we use a lag time of $5\,\mathrm{s}$. All visual and lidar factors are added to the graph using the Dynamic Covariance Scaling (DCS) [39] robust cost function to reduce the effect of outliers.

A. Sensor Synchronization

To perform joint optimization across all the sensor modalities, the measurements first need to be synchronized. As in [26], the IMU and kinematic measurements, which have high frequency, are interpolated to match the camera timestamps. To synchronize the lidar measurements with the camera, we instead follow the approach described in [16]. When a new point cloud is received, its points are motion compensated using the IMU propagated state (from the preintegrated IMU factor) using the closest camera keyframe timestamp as reference, instead of the start or end of the timestamp, as commonly done. In this way, the minimum number of nodes will be added to the graph. The optimization is performed *jointly* between IMU, kinematics, camera, and lidar inputs. This also ensures a fixed output frequency, i.e., the camera keyframe rate.

B. Forward Kinematics and Dynamics

The forward kinematics and dynamics are implemented in RobCoGen [40], which is a computationally efficient kinematics and dynamics solver, as demonstrated in this comparison paper [41]. RobCoGen takes the current joint configuration and base state to calculate the position, velocity, and force at each of the feet. The covariance values for the kinematics are taken directly from the encoder datasheet.

C. Twist Bias Modes

The introduction of a twist bias is motivated by non-ideal contact events which can cause drift in the estimated velocity. Therefore, if no new contact events are occurring (i.e., the robot is not making any new steps), this velocity bias should not be present.

To account for this, when no footsteps are being made we disable the twist bias and simply use relative pose factors instead. This is important for legged robots since the robot can still move (e.g., roll and pitch) while the feet are stationary. This is in contrast to the zero velocity mode presented in [15] where access to the contact states was not possible due to the use of an external filter to process the kinematics.

In practice, we detect this *low drift* state when the majority of the robot's feet (i.e., 3 or 4 feet) are in constant contact with the ground for an extended period of time (i.e., more than $200\,\mathrm{ms}$). As soon as new footsteps are made we exit this low-drift state and return to normal operation.

D. Visual and Point Cloud Feature Tracking

We detect visual features using the FAST corner detector, and track them between successive frames using the KLT feature tracker. Outliers are rejected using a RANSAC-based fundamental matrix consistency check (similar to [42]). For point cloud features, we use the feature tracking approach based on [16] to extract and track geometric primitives (namely planes $\boldsymbol{p}_\ell$ and lines $\boldsymbol{l}_\ell$) over time.

Keyframes are added when mean visual feature movement between consecutive frames is greater than a threshold ($1\,\mathrm{pixel}$), with minimum ($5\,\mathrm{Hz}$) and maximum ($15\,\mathrm{Hz}$) keyframe frequency bounds.

E. Zero Velocity Update Factors

To limit drift and factor graph growth, we continuously query if the robot is stationary by using a voting mechanism. If the majority of sensor modalities detect no motion then we add a zero velocity constraint to the graph. This method can detect when the robot is stationary, even when one or more sensor inputs are not available (e.g., legs are not in contact with the ground).

For example, the IMU and leg odometry threads report zero velocity when position (rotation) is less than $0.1\,\mathrm{mm}$ ($0.5°$) between two keyframes. The image thread reports zero velocity when average feature movement between frames is

less than 0.5 pixels. The lidar thread reports zero velocity when the motion induced by tracking planes and lines is less than $0.2\,\mathrm{m}$ and $5°$, respectively.

F. Calibration

We use the open source camera and IMU calibration toolbox Kalibr [43] to compute the intrinsic and extrinsic calibrations of the cameras. The IMU, lidar, and kinematics positions are known from accurate CAD models and are rigidly attached to the frame of the robot.

All sensors are hardware-synchronized where possible (using either Precision Time Protocol (PTP), EtherCAT, or hardware triggering). The only exception is the Realsense D435i cameras which are software-synchronized using the manufacturer's driver, before removing any clock drift using continuous-time maximum likelihood estimation [44].

G. Initialization

We initialize the system by averaging the first $1\,\mathrm{s}$ of IMU data at system start-up (assuming the IMU is stationary). This allows us to find the gyroscope bias and initial estimates for orientation of the gravity vector. As they cannot be estimated from stationary data [45], accelerometer biases are initially assumed to be either zero or a known constant value. The accelerometer biases would then converge to their true value once the robot starts moving, thanks to the robot's 3D walking motion. The scale is known from either stereo correspondence or lidar measurements.

VI. Experimental Results

In this section we will describe the platforms and the dataset used in our evaluations (Sections VI-A and VI-B). We will then demonstrate how the tight integration of kinematics into a factor graph framework allows for easy combination of the different sensor modalities (Section VI-C). Finally, we will end with an analysis of specific challenging situations in the dataset, and discuss computational performance (Sections VII and VI-F).

A. Experimental Platforms

The platforms used for our experiments are the ANYbotics ANYmal B300 [46] and C100 quadrupeds (Fig. 1). Both robots have 4 identical legs giving a total of 12 active Degrees-of-Freedom (DoF) and are equipped with an IMU and a Velodyne VLP-16 lidar. Each leg has joint encoders, torque sensors. In some of the experiments, the robots were modified from the stock version to compete in the DARPA SubT Challenge as detailed in [10]. This gives a total of three different sensor configurations (see Fig. 1 and Table I). Note that, even though the experiments were performed using the default leg configuration of ANYmal with inward pointing knees, VILENS is agnostic to the number and configuration of legs used.

Sensor	Model	Hz	Specs
Sensors common to ANYmal B300 and C100			
Encoder	ANYdrive	400	*Res:* $<0.025°$
Torque	ANYdrive	400	*Res:* $<0.1\,\mathrm{N\,m}$
Lidar	Velodyne VLP-16	10	*Res:* $16\,\mathrm{px} \times 1824\,\mathrm{px}$
ANYmal B300 (experiments: SMR, FSC)			
IMU	Xsens MTi-100	400	*Init Bias:* $0.2\,°/\mathrm{s} \mid 5\,\mathrm{mg}$ *Bias Stab:* $10\,°/\mathrm{h} \mid 15\,\mathrm{mg}$
Gray Stereo Camera	RealSense D435i	30	*Res:* $848\,\mathrm{px} \times 480\,\mathrm{px}$ *FoV (Diag.):* $100.6°$
ANYmal B300 (experiment: SUB)			
IMU	Xsens MTi-100	400	*Init Bias:* $0.2\,°/\mathrm{s} \mid 5\,\mathrm{mg}$ *Bias Stab:* $10\,°/\mathrm{h} \mid 15\,\mathrm{mg}$
RGB Mono Camera	FLIR BFS-U3-16S2C-CS	30	*Res:* $1440\,\mathrm{px} \times 1080\,\mathrm{px}$ *FoV (Diag.):* $150°$
ANYmal C100 (experiments: LSM, SMM)			
IMU	Epson G365	400	*Init Bias:* $0.1\,°/\mathrm{s} \mid 3\,\mathrm{mg}$ *Bias Stab:*$1.2\,°/\mathrm{h} \mid 15\,\mathrm{mg}$
Gray Stereo Camera	Sevensense Alphasense	30	*Res:* $720\,\mathrm{px} \times 540\,\mathrm{px}$ *FoV (Diag.):* $165.4°$

TABLE I
Sensor Specifications of the Experimental Platforms

B. Dataset

To evaluate our proposed algorithm, we have collected datasets in a variety of test environments lasting a total of $2\,\mathrm{h}$ and traveling $1.8\,\mathrm{km}$. This includes data from the Urban Circuit of the DARPA SubT Challenge. The dataset is composed of the following experiments:

- **SMR:** Swiss Military Rescue Facility, Wangen, CH. Trotting over concrete & gravel followed by loops on grass with different gaits (ANYmal B300, $106\,\mathrm{m}$, $13\,\mathrm{min}$);
- **FSC:** Fire Service College, Moreton-in-the-Marsh, UK. Three loops of an outdoor industrial environment with standing water, oil residue, gravel and mud (ANYmal B300, $240\,\mathrm{m}$, $34\,\mathrm{min}$);
- **SUB:** DARPA SubT Urban Beta course, Satsop, WA, USA[4]. Autonomous exploration of a dark underground inactive nuclear powerplant (ANYmal B300, $490\,\mathrm{m}$, $60\,\mathrm{min}$);
- **LSM:** A decommissioned limestone mine, Wiltshire, UK. Teleoperated exploration of an unlit mine with several loops (ANYmal C100, $474\,\mathrm{m}$, $20\,\mathrm{min}$);
- **SMM:** Seemühle mine, CH. Autonomous exploration with Cerberus SubT exploration system (ANYmal C100, $522\,\mathrm{m}$, $17\,\mathrm{min}$).

Images from the on-board cameras in each of the experiments are shown in Fig. 8 which illustrate various challenges including slippery ground (affecting kinematics), reflections and darkness (affecting vision), and long corridors (affecting lidar point cloud registration). Different copies of the robots were used in each experiment.

[4]This is the same sequence used in [16], except here we additionally process leg kinematics.



Fig. 8. Onboard camera images showing challenging environments from the experiments. *Top-Left*: wet and oily concrete with reflections and lack of features at the horizon (FSC); *Top-Right:* complete darkness (image manually enhanced in post processing). *Bottom-Left:* low light, overexposure and dust particles (LSM); *Bottom-Right:* long straight corridors and low light (SMM).

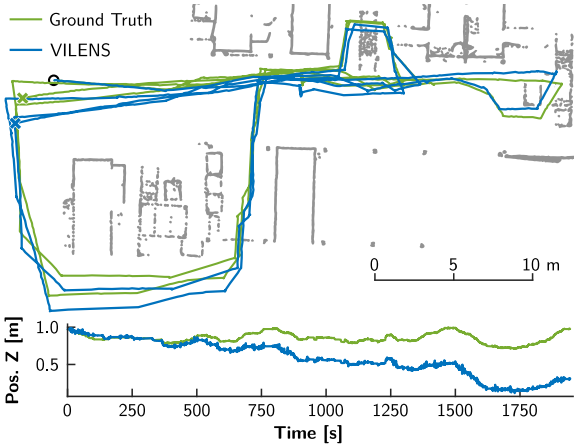


Fig. 9. Aerial view of the estimated and ground truth trajectories on the FSC experiment (240 m traveled). The start of the trajectory is marked with a circle. Ending location is marked with a cross. Note that since VILENS is an odometry system, no loop closures have been performed.

10 m Relative Pose Error (RPE)					
	Translation $\mu(\sigma)$[m]				
Data	**C-S*** [9]	**V-LVI**	**V-LVIK**	**V-IR†**	**VILENS**
SMR	0.28 (0.14)	0.14 (0.12)	0.15 (0.12)	0.24 (0.15)	**0.12** (0.11)
FSC	0.16 (0.08)	0.20 (0.09)	0.24 (0.17)	0.15 (0.09)	**0.15** (0.07)
SUB	0.20 (0.14)	0.11 (0.07)	0.11 (0.08)	0.10 (0.08)	**0.05** (0.03)
LSM	0.27 (0.15)	0.34 (0.30)	0.29 (0.33)	0.10 (0.08)	**0.04** (0.04)
SMM	0.36 (0.14)	0.74 (0.99)	0.74 (0.90)	0.27 (0.22)	**0.12** (0.08)
Mean	0.25 (0.13)	0.31 (0.31)	0.31 (0.32)	0.17 (0.12)	**0.10** (0.07)
	Rotation $\mu(\sigma)$[°]				
Data	**C-S*** [9]	**V-LVI**	**V-LVIK**	**V-IR†**	**VILENS**
SMR	3.14 (1.83)	**1.18** (0.97)	1.38 (1.05)	1.69 (0.93)	1.30 (1.07)
FSC	2.36 (1.07)	1.30 (0.90)	1.17 (0.94)	2.03 (0.86)	**1.14** (0.78)
SUB	0.99 (0.62)	0.75 (0.40)	0.74 (0.43)	1.88 (1.13)	**0.56** (0.34)
LSM	1.80 (0.94)	2.32 (1.73)	1.92 (1.51)	1.44 (0.71)	**0.59** (0.39)
SMM	1.42 (0.73)	4.73 (6.60)	4.38 (5.26)	2.73 (1.73)	**1.19** (0.61)
Mean	1.94 (1.04)	2.06 (2.12)	1.92 (1.84)	1.95 (1.07)	**0.96** (0.64)

TABLE II
EXPERIMENTAL RESULTS. "C-S" = COMPSLAM, "V-" = VILENS-, *OUTPUT AT 5 Hz, †OUTPUT AT 2 Hz

In the SMR and FSC experiments, ground truth was generated by tracking the robot using a Leica TS16 laser tracker and then estimating its orientation using a optimization-based method (as described in [15]). In LSM, SUB, and SMM experiments, an ICP-based method was to align the current lidar scan to an accurate prior map made with survey-grade lidar scanners (as described in [44]).

C. Multi-sensor Fusion Comparison

To evaluate the performance of the system we compute the mean Relative Pose Error (RPE) over distances of 10 m traveled for different module combinations of VILENS (at 15 Hz). We use a letter to indicate the sensor modalities used as follows: lidar features (**L**), visual features (**V**), IMU preintegration (**I**), preintegrated leg kinematics (**K**), ICP registration (**R**).

The set of combinations tested are as follows:

- **VILENS-LVI:** lidar and visual feature tracking with preintegrated IMU but no leg kinematics. This is the same configuration as [16].
- **VILENS-LVIK:** lidar and visual feature tracking, with preintegrated IMU and including leg kinematics.
- **VILENS-IR:** only ICP registration and preintegrated IMU. As a fair comparison, we use the IMU to undistort point clouds. In this configuration, the output is limited by the frequency of ICP and thus can only run at 2 Hz.
- **VILENS:** lidar and visual feature tracking, preintegrated IMU, leg kinematics, and ICP registration. This is the full algorithm proposed in this paper. (Where no letters are appended to the word VILENS we mean this version.)

Note that the same settings have been used for all configurations, with just different modules activated for each experiment.

For comparison we also tested CompSLAM [9] which is a loosely coupled filter combining lidar-inertial odometry (LOAM [47]), visual-inertial odometry (ROVIO [48]), and kinematic-inertial odometry (TSIF [2]). It uses heuristics to switch between modalities. LOAM is its primary modality and has a frequency of 5 Hz. CompSLAM was also tested and deployed on the ANYmal robots of the winning team of the DARPA Subterranean Challenge [10].

Figs. 9 and 10 illustrate the high level performance of VILENS. It can be seen that the estimated trajectory closely matches the ground truth, showing the accuracy of this approach. Note that VILENS is an odometry system — no loop closures are performed in this system. However VILENS can be integrated with an external SLAM system such as [49].

Quantitative results are summarized in Table II. Across the entire dataset, the best performing algorithm was the complete VILENS system, with an overall mean RPE error of just 0.96 % in translation and 0.0956 deg /m in rotation.

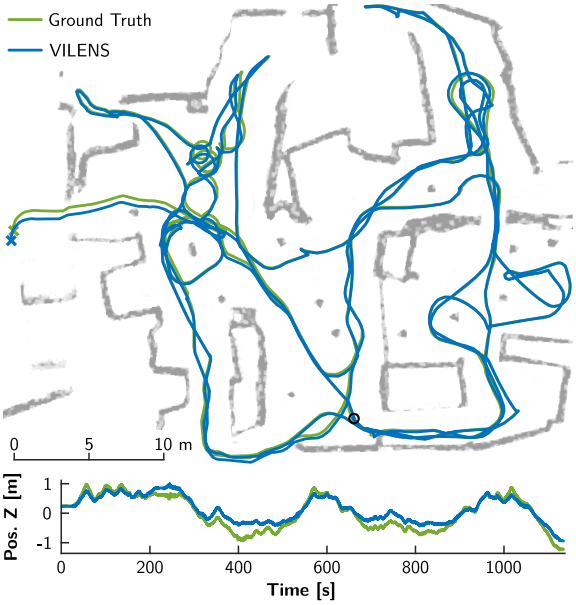


Fig. 10. Top-down view of the trajectory estimate by VILENS (blue) and the ground truth (green) on the 474 m LSM experiment. The start of the trajectory is indicated by a black circle, while the end is indicated with a cross. Note that since VILENS is an odometry system, no loop closures have been performed.

10 m Relative Pose Error (RPE)				
	Translation $\mu(\sigma)$[m]		Rotation $\mu(\sigma)$[°]	
Data	**VILENS-NO-BIAS**	**VILENS**	**VILENS-NO-BIAS**	**VILENS**
SMR	0.12 (0.12)	**0.12** (0.11)	1.30 (0.94)	**1.30** (1.07)
FSC	0.21 (0.15)	**0.15** (0.07)	1.24 (0.79)	**1.14** (0.78)
SUB	0.06 (0.05)	**0.05** (0.03)	0.64 (0.41)	**0.56** (0.34)
LSM	0.05 (0.04)	**0.04** (0.04)	0.64 (0.41)	**0.56** (0.34)
SMM	0.13 (0.09)	**0.12** (0.08)	**1.12** (0.62)	1.19 (0.61)
Mean	0.11 (0.09)	**0.10** (0.07)	1.02 (0.65)	**0.96** (0.64)

TABLE III
ABLATION STUDY OF ONLINE VELOCITY BIAS ESTIMATION

Compared to the ICP and IMU only (VILENS-IR) solution VILENS provides higher frequency and higher accuracy estimation, as well as robustness against degenerate scenarios for lidar, including long tunnels. The improvement in performance is due to the incorporation of different sensor modalities at higher frequency. This created a smoother state estimate which improved lidar motion correction, provided a better prior for ICP, and allowed for inaccurate ICP estimates to be rejected using robust cost functions.

Additionally, VILENS outperforms the loosely coupled approach, CompSLAM, by an average of 62 % in translation and 51 % in rotation.

Accurate, high frequency state estimation is also a requirement for terrain mapping, as discussed in Section VI-F.

D. Twist Bias Ablation Study

In this section we quantitatively evaluate the effect of the velocity bias estimation. As shown in Table III, even when using all available sensor modalities (including cameras and lidar), the biased signal from kinematics can still negatively effect the outcome. On average, adding online bias estimation reduces the RPE on average by 9.0 % / 5.9 % in translation / rotation. However, the effect of bias estimation is most obvious when exteroceptive sensors are degraded (see Section VII-B).

E. Internal Kinematics Comparison

In this section we evaluate quantitatively the benefit of incorporating the kinematics inside the factor graph, rather than relying on an external filter. Fig. 11 shows a comparison between the estimated velocity when using internal kinematics and receiving the output from an external kinematic-inertial filter [2]. The ground truth was obtained by differentiation of the 200 Hz motion capture data. Note that in this configuration we are only using kinematic-inertial inputs (no exteroceptive sensors) to highlight the difference between the two approaches. It can be seen that using internal kinematics improves the velocity tracking, and in turn reduces position drift by 45 %.

F. Integration with Local Planner and Perceptive Controller

This section will provide a qualitative demonstration of VILENS running onboard the ANYmal robot. The limited on-board computation is shared with other processes running on the robot, including control, terrain mapping, and sensor drivers.

Constructing an accurate local terrain map around the robot is crucial for perceptive locomotion and path planning. Locomotion controllers plan footstep placements on these maps such as [50].

The current approach for the ANYmal robot is to use the kinematic-inertial estimator, TSIF [2], to feed the local elevation mapping system [51]. TSIF can suffer from significant drift, which creates “phantom obstacles” in the terrain map. The effect of this is that the robot exhibits undesirable behavior such as poorly placed foot landings or planning suboptimal paths. In the worst case, the robot may fall and get irreversibly damaged or become stuck because the local path planner is unable to find a feasible solution.

Fig. 12 shows the pipeline used for the terrain mapping with VILENS. The inputs to the terrain mapping module [51] are the VILENS state estimate, and the point clouds from several downward-facing depth cameras on the robot’s body. These cameras are not the same as the ones used for state estimation and are not triggered at the same time. Therefore, accurate terrain mapping depends on a smooth, accurate, and high frequency state estimate to avoid interpolation errors.

From our experience, terrain mapping frequencies of $\geq$15 Hz are required for dynamic locomotion over rough terrain. This means that registration-based algorithms with a low frequency output (such as ICP or LOAM) are not suitable for this purpose.

1) Terrain Map Integration 1: Local Path Planning: In the first experiment, we evaluate the quality of online local terrain mapping for local path planning. During the LSM experiment the terrain map was successfully used in the loop by local path

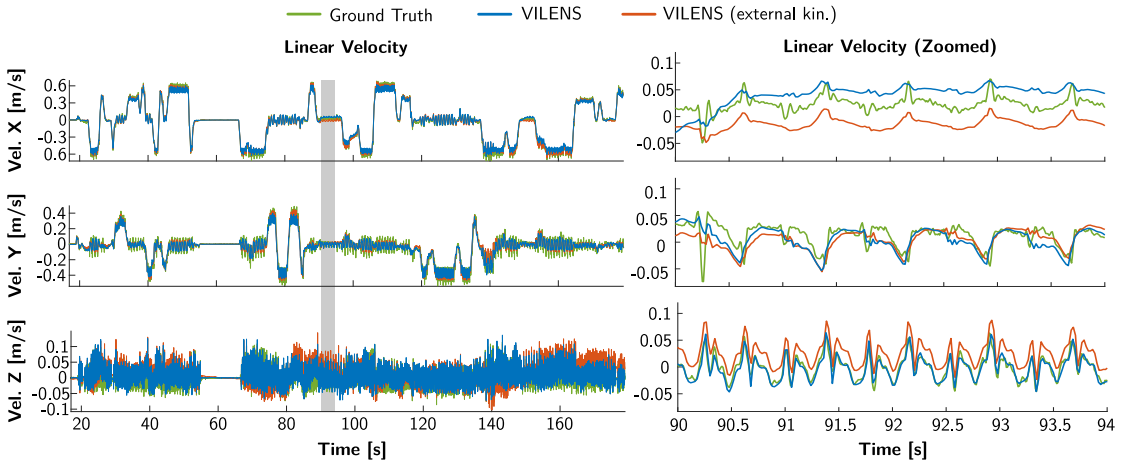


Fig. 11. *Left:* Linear velocity comparison between VILENS fusing IMU and leg kinematics using the external filter from TSIF [2] (orange) and fusing the kinematics inside the optimizer (blue) against ground truth (green). *Right:* zoomed in plot corresponding to the gray area of the complete experiment. The better velocity tracking allows for a reduction in position drift.

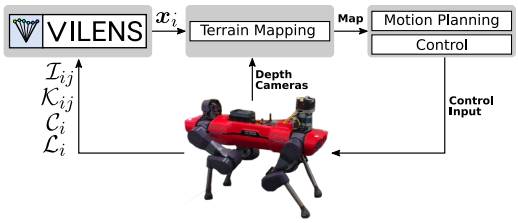


Fig. 12. *Terrain Mapping Pipeline:* The VILENS state estimator produces a high frequency (400 Hz), low drift state estimate, $\boldsymbol{x}$, for the terrain mapping module [51]. This local terrain map can then be used by other modules, such as perceptive motion planning and control [17], or local path planning for obstacle avoidance [18].

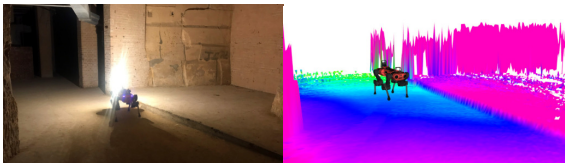


Fig. 13. The accurate terrain map generated with the VILENS state estimate can be fed into the local path planner [18] for autonomous exploration. The ledge near the robot is clearly defined in the elevation map, allowing the local planner to create routes on this obstacle.

planning algorithm, GBPlanner [18], for over 30 min without failure. Fig. 13-*Left* shows an example of the accuracy of the terrain map compared to the ground truth.

2) Terrain Map Integration 2: Perceptive Control: In the second experiment, we demonstrate terrain mapping for perceptive control using RLOC [17] during a 5 min outdoor experiment over grass, gravel, curbs, and slopes. Fig. 13 shows the accuracy of the terrain map, which allowed the controller to plan and execute precise steps on the terrain. By reducing the amount drift in the state estimate, the quality of terrain map improves, allowing for more accurate motion planning.

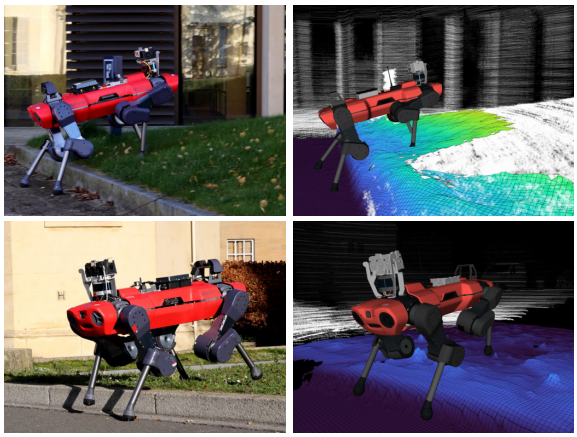


Fig. 14. The ANYmal robot walking over curbs, slopes, and other rough terrain using the RLOC perceptive controller [17]. The accurate terrain map produced by using the VILENS estimate enabled successful execution of the trajectory.

Future work could involve tighter integration of local terrain mapping and contact point locations estimated by VILENS.

VII. Discussion

A. Evolution of Velocity Bias

In Fig. 15 we compare the actual and the estimated velocity error from kinematics in the z-axis, which is where most of the drift occurs for this particular sequence due to the presence of gravel. The orange line indicates the error between the true robot velocity and the one perceived by the leg odometry, while the blue line shows the VILENS estimate of the same error, which is modeled as a velocity bias term of the leg odometry factor, $\mathbf{b}^v$. The sequence analyzed is the same as the one shown in Fig. 2. The high degree of correlation between

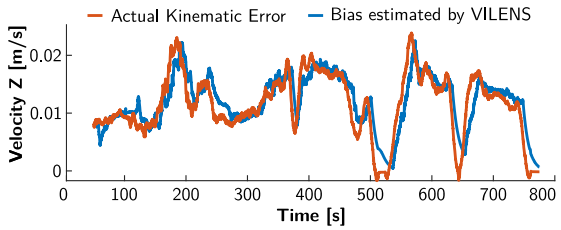


Fig. 15. Comparison between the actual velocity error (from the kinematic odometry in the z-axis inferred using a Leica tracker) and the bias estimated by VILENS during the SMR experiment. Using exteroceptive sensing, VILENS is able to accurately and stably track this effect by modeling the bias term $\mathbf{b}^v$.

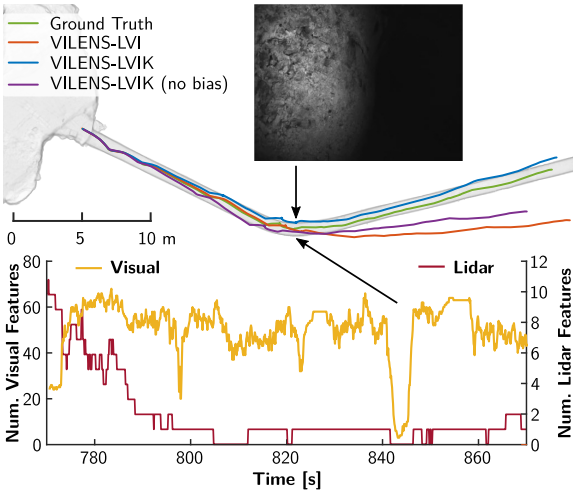


Fig. 16. *Top:* Comparison between the trajectory estimates for different VILENS configurations and ground truth (in green) for the tunnel sequence of the SMM experiment. The lack of visual features causes a large rotation error when not using leg odometry (orange line). Adding leg odometry improves the estimate but still accumulates drift (purple line). With the online velocity bias estimation, the drift is reduced even further (blue line). *Bottom:* number of lidar (in red) and visual (in yellow) features tracked over time. Note that the lidar features plot has been slightly smoothed over a 1 s window for clarity.

the two signals demonstrates the effectiveness of leg odometry velocity bias estimation.

B. Performance in Underconstrained Environments

In contrast to other recent loosely-coupled approaches to multi-sensor state estimation [9], [52], VILENS naturally handles degenerate scenarios without requiring hard switches which are typically hand engineered.

An example is shown in Fig. 16, where the ANYmal robot walks very close to a wall in a long, straight tunnel. At time 843 s, the robot turns towards the straight part of the tunnel. Due to the degenerate geometry, the number of lidar features gradually drops as the robot goes deeper into the tunnel (maroon line, bottom plot). While the lidar feature tracking can still provide partial constraints to the system, ICP is unstable and close to divergence, so it was not included into the analysis. At the same time, the scene changes abruptly

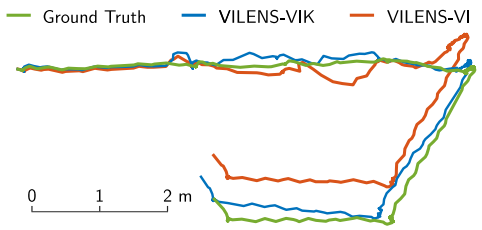


Fig. 17. Top-down comparison of VILENS-VIK and VILENS-VI trajectories aligned with ground truth while crossing the puddle shown in Fig. 8 Top-Left.

causing a period of underexposure (picture at the top), with a dramatic drop in the number of visual features (yellow line, bottom plot). Because of this, the VILENS-LVI configuration (i.e., VILENS without leg odometry and ICP, as in [16]) accumulates a large angular error (orange line). Instead, the addition of leg odometry factor improves the estimate (VILENS-LVIK, blue). To show the benefit of velocity bias estimation, we also show the same VILENS-LVIK configuration but without online bias estimation. Because the terrain is loose, the drift is accumulated faster (purple line).

C. Leg Odometry in Visually Degraded Scenarios

Fig. 17 shows a situation from FSC experiment where the robot crosses a large puddle. VILENS-VI tracks the features on the standing water, causing drift on the xy-plane. Instead, VILENS-VIK maintains a better pose estimate by relying on leg odometry with the bias estimation.

D. Timing Analysis

An important consideration in multi-sensor fusion algorithm is computation time, which generally increases with the number of input sensors.

Table IV shows a summary of the computation time for the components of VILENS. The timing tests were performed on a laptop equipped with an Intel E-2186M processor (6 cores/12 threads @ 2.9 GHz base frequency) and 16 GB of RAM. A key benefit of using this type of factor-graph based approach is that the lightweight visual and point cloud features allow for accurate, low latency state estimation at relatively low computational expense. This also allows more expensive modalities, such as ICP, to run at lower frequency. This saves computation and preserves the benefits of low-drift ICP state estimation.

Table V highlights the three different types of outputs from the VILENS system, with differing levels of latency and accuracy for different purposes. Fig. 18 shows how the high frequency forward propagated estimate closely matches Vicon ground truth. This is important in some applications to capture the high frequency behavior of the robot between optimizations.

VIII. Conclusion

This paper presented VILENS, a multi-sensor fusion algorithm which can seamlessly fuse inertial, legged, lidar, and

Module	Timing $\mu(\sigma)$ [ms]	Freq. [Hz]
IMU	0.05 (0.12)	400
Leg kinematics	0.07 (0.30)	400
Visual features with lidar depth	9.48 (7.69)	10
Lidar point cloud features	19.72 (6.24)	10
Lidar ICP	149.75 (59.23)	2
Optimization	8.65 (3.25)	10

TABLE IV
ANALYSIS OF TIMING FOR DIFFERENT PARTS OF VILENS

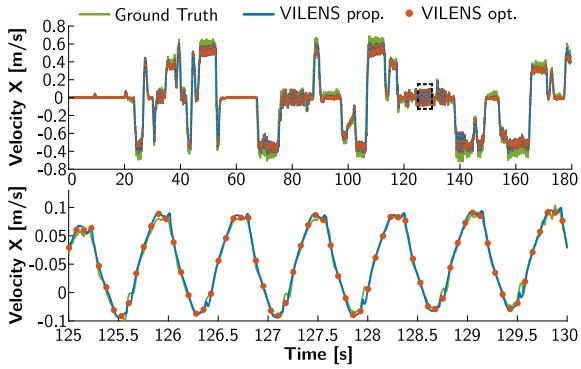


Fig. 18. *Top:* Comparison between the optimized (10 Hz, orange dots) and IMU forward propagated (400 Hz, blue line) velocity estimates against the ground truth from motion capture (green line). *Bottom:* zoomed in view corresponding to the dashed box at time interval 125–130 s. The close tracking of the ground truth shows that IMU forward propagation provides an accurate, high frequency, and low latency prediction of the state between optimizations.

visual sensing within the same factor graph. The tight fusion of all these sensor modalities allows the algorithm to overcome the challenging operational conditions of autonomous navigation in underground environments as well as large open areas. In these conditions, individual sensor modalities such as visual-inertial or lidar-inertial odometry would fail.

A particular contribution of the paper was the integration of the leg kinematic measurements as a dedicated residual for the factor graph, rather than relying on an external filter. This enables tighter integration and better noise modeling. The leg odometry drift, typically occurring due to leg compression at the impact, was modeled as a bias term of the velocity measurement and was proven to be effective in challenging episodes where both the lidar and camera sensors are deprived. In these situations, our system falls back to fusing IMU and leg odometry with no hard switching required.

We demonstrated the robustness of our method in field experiments with several quadruped platforms in challenging scenarios including slippery and deformable terrain, water reflections, complete darkness, and degenerate geometries, such as long corridors lasting longer than two hours.

The high frequency as well as the smoothness of the output was further demonstrated by running the algorithm online and onboard the robot to enable local elevation mapping, which was used by a perceptive controller [17] and a local path planner [18] to cross an obstacle course and to autonomously explore a mine, respectively.

Output type	Mean latency [ms]	Freq. [Hz]
IMU forward-propagated	2.3	400
Factor graph optimized	95.2	10
ICP optimized	395.2	2

TABLE V
FREQUENCY AND LATENCY OF OUTPUTS. LATENCY IS COMPARED TO THE LOWEST LATENCY INPUT SIGNAL (IMU)

APPENDIX
PREINTEGRATED LEG ODOMETRY FACTOR

In this appendix we provide a derivation of the measurements and covariance for the preintegrated leg odometry factor described in Section IV-B. Without loss of generality, we will focus on cases when only one leg is in contact with the ground. As explained in Section IV-B2, when there are multiple stance legs, the compound measurement and noise can be computed independently from each stance leg and the fused via a weighted average.

A. Velocity Measurement and Noise

When a leg is in rigid, nonslipping contact with the ground, the robot's linear velocity can be computed from the position and velocity of the feet:

$$\mathbf{v} = -\mathbf{J}_p(\boldsymbol{\alpha})\dot{\boldsymbol{\alpha}} - \boldsymbol{\omega} \times \mathbf{f}_p(\boldsymbol{\alpha}) \tag{48}$$

where $\mathbf{f}_p(\cdot)$ is the forward kinematics function and $\mathbf{J}_p(\cdot)$ is its Jacobian. To compute a velocity measurement for the factor graph, we need to linearly separate the noise from the rest of Eq. (48). Remembering from Eqs. (17) – (18) that the joint states are corrupted by zero-mean Gaussian noise, such that $\boldsymbol{\alpha} = \tilde{\boldsymbol{\alpha}} - \boldsymbol{\eta}^{\alpha}, \dot{\boldsymbol{\alpha}} = \tilde{\dot{\boldsymbol{\alpha}}} - \boldsymbol{\eta}^{\dot{\alpha}}$, we first start by separating the noise from the differential kinematics term:

$$\begin{aligned}\mathbf{J}_p(\boldsymbol{\alpha})\dot{\boldsymbol{\alpha}} &= \mathbf{J}_p(\tilde{\boldsymbol{\alpha}} - \boldsymbol{\eta}^{\alpha})(\tilde{\dot{\boldsymbol{\alpha}}} - \boldsymbol{\eta}^{\dot{\alpha}}) \\ &= \mathbf{J}_p(\tilde{\boldsymbol{\alpha}} - \boldsymbol{\eta}^{\alpha})\tilde{\dot{\boldsymbol{\alpha}}} - \mathbf{J}_p(\tilde{\boldsymbol{\alpha}} - \boldsymbol{\eta}^{\alpha})\boldsymbol{\eta}^{\dot{\alpha}} \\ &\simeq \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} - \frac{\partial}{\partial \tilde{\boldsymbol{\alpha}}}\left(\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}}\right)\boldsymbol{\eta}^{\alpha} - \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\dot{\alpha}}\end{aligned} \tag{49}$$

where we applied the Taylor expansion of the product $\mathbf{J}_p(\boldsymbol{\alpha})\dot{\boldsymbol{\alpha}}$, ignoring the second order terms. Since $\boldsymbol{\alpha}$ and $\dot{\boldsymbol{\alpha}}$ are independent, $\dot{\boldsymbol{\alpha}}$ can be taken out of the derivative in Eq. (49):

$$\frac{\partial}{\partial \tilde{\boldsymbol{\alpha}}}\left(\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}}\right)\boldsymbol{\eta}^{\alpha} = \frac{\partial}{\partial \tilde{\boldsymbol{\alpha}}}\left(\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\right)\tilde{\dot{\boldsymbol{\alpha}}}\boldsymbol{\eta}^{\alpha} = \mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}}\boldsymbol{\eta}^{\alpha} \tag{50}$$

where $\mathbf{H}_p(\cdot) \in \mathbb{R}^{3\times3\times3}$ is the Hessian of the forward kinematics function $\mathbf{f}_p(\boldsymbol{\alpha})$. Note that, since the forward kinematics is a vector function, the Hessian becomes a tensor of rank 3, so the product with $\mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} = \mathbf{H}_p(\tilde{\boldsymbol{\alpha}}) \bar{\times}_2 \tilde{\dot{\boldsymbol{\alpha}}}$ is a 2-mode vector product [53]. In our implementation, the calculation of Eq. (50) is performed numerically.

Substituting Eqs. (49) – (50) into Eq. (48) leads to:

$$\begin{aligned}\mathbf{v} = -\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} - \mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}}\boldsymbol{\eta}^{\alpha} - \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\dot{\alpha}} \\ - \boldsymbol{\omega} \times \mathbf{f}_p(\tilde{\boldsymbol{\alpha}} - \boldsymbol{\eta}^{\alpha})\end{aligned} \tag{51}$$

It has been demonstrated in [7] that the noise from the last term of Eq. (51) can be separated as follows:

$$\mathbf{f}_p(\tilde{\boldsymbol{\alpha}} - \boldsymbol{\eta}^{\alpha}) \approx \mathbf{f}_p(\tilde{\boldsymbol{\alpha}}) - \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\alpha} \tag{52}$$

From the above relations, Eq. (48) becomes:

$$\begin{aligned}
\mathbf{v} &= -(\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} - \mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}}\boldsymbol{\eta}^{\alpha} - \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\dot{\alpha}}) \\
&\qquad - \boldsymbol{\omega}^{\wedge}\mathbf{f}_p(\tilde{\boldsymbol{\alpha}} - \boldsymbol{\eta}^{\alpha}) \\
&= -\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} + \mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}}\boldsymbol{\eta}^{\alpha} + \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\dot{\alpha}} \\
&\qquad - \boldsymbol{\omega}^{\wedge}(\mathbf{f}_p(\tilde{\boldsymbol{\alpha}}) - \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\alpha}) \\
&= -\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} + \mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}}\boldsymbol{\eta}^{\alpha} + \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\dot{\alpha}} \\
&\qquad - \boldsymbol{\omega}^{\wedge}\mathbf{f}_p(\tilde{\boldsymbol{\alpha}}) + \boldsymbol{\omega}^{\wedge}\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\alpha} \\
&= \underbrace{-\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} - \boldsymbol{\omega}^{\wedge}\mathbf{f}_p(\tilde{\boldsymbol{\alpha}})}_{\tilde{\mathbf{v}}} - \\
&\qquad \underbrace{\left(-\left(\mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} + \boldsymbol{\omega}^{\wedge}\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\right)\boldsymbol{\eta}^{\alpha} - \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\dot{\alpha}}\right)}_{\boldsymbol{\eta}^{v}} \\
&\triangleq \tilde{\mathbf{v}} - \boldsymbol{\eta}^{v}
\end{aligned} \tag{53}$$

Eq. (53) still depends on the rotational velocity, which is measured by the IMU and is affected by measurement noise and bias. If we replace the angular velocity with the measured values, we have:

$$\begin{aligned}
&\mathbf{v} = -\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} - (\tilde{\boldsymbol{\omega}} - \mathbf{b}^{\omega} - \boldsymbol{\eta}^{\omega})^{\wedge}\mathbf{f}_p(\tilde{\boldsymbol{\alpha}}) + \\
&\left(\mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} + (\tilde{\boldsymbol{\omega}} - \mathbf{b}^{\omega} - \boldsymbol{\eta}^{\omega})^{\wedge}\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\right)\boldsymbol{\eta}^{\alpha} + \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{v}
\end{aligned} \tag{54}$$

By applying the inversion rule for the cross product and the distributive property we have:

$$\begin{aligned}
&\mathbf{v} = -\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} + \mathbf{f}_p(\tilde{\boldsymbol{\alpha}})^{\wedge}\tilde{\boldsymbol{\omega}} - \mathbf{f}_p(\tilde{\boldsymbol{\alpha}})^{\wedge}\mathbf{b}^{\omega} - \mathbf{f}_p(\tilde{\boldsymbol{\alpha}})^{\wedge}\boldsymbol{\eta}^{\omega} + \\
&\left(\mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} - \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})^{\wedge}\tilde{\boldsymbol{\omega}} + \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})^{\wedge}\mathbf{b}^{\omega} + \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})^{\wedge}\boldsymbol{\eta}^{\omega}\right)\boldsymbol{\eta}^{\alpha} + \\
&\qquad \mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\dot{\alpha}}
\end{aligned} \tag{55}$$

After rearrangement and removal of the second order terms, the noise can be again separated as follows:

$$\begin{aligned}
\mathbf{v} &\simeq \underbrace{-\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} - \tilde{\boldsymbol{\omega}}^{\wedge}\mathbf{f}_p(\tilde{\boldsymbol{\alpha}}) + \mathbf{b}^{\omega\wedge}\mathbf{f}_p(\tilde{\boldsymbol{\alpha}})}_{\tilde{\mathbf{v}}} - \\
&\quad \underbrace{\left(-\left(\mathbf{H}_p(\tilde{\boldsymbol{\alpha}})\tilde{\dot{\boldsymbol{\alpha}}} + \tilde{\boldsymbol{\omega}}^{\wedge}\mathbf{J}_p(\tilde{\boldsymbol{\alpha}}) - \mathbf{b}^{\omega\wedge}\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\right)\boldsymbol{\eta}^{\alpha}\right.}_{\cdots} \\
&\quad \underbrace{\left. -\mathbf{J}_p(\tilde{\boldsymbol{\alpha}})\boldsymbol{\eta}^{\dot{\alpha}} + \mathbf{f}_p(\tilde{\boldsymbol{\alpha}})^{\wedge}\boldsymbol{\eta}^{\omega}\right)}_{\boldsymbol{\eta}^{v}} \\
&\simeq \tilde{\mathbf{v}} - \boldsymbol{\eta}^{v}
\end{aligned} \tag{56}$$

B. Iterative Noise Propagation

By substitution of $\boldsymbol{\eta}^{v}$ from Eq. (56) into Eq. (34), we can express the preintegrated noises $\delta\boldsymbol{\theta}, \delta\boldsymbol{\kappa}$ in iterative form:

$$\begin{aligned}
\begin{bmatrix}\delta\boldsymbol{\theta}_{i,k+1} \\ \delta\boldsymbol{\kappa}_{i,k+1}\end{bmatrix} &= \underbrace{\begin{bmatrix}\Delta\tilde{\boldsymbol{\Theta}}_{k,k+1}^{\mathsf{T}} & \mathbf{0} \\ -\Delta\tilde{\boldsymbol{\Theta}}_{i,k}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^{v})^{\wedge}\Delta t & \mathbf{I}\end{bmatrix}}_{\mathbf{A}}\begin{bmatrix}\delta\boldsymbol{\theta}_{i,k} \\ \delta\boldsymbol{\kappa}_{i,k}\end{bmatrix} + \\
&\quad \underbrace{\begin{bmatrix}\mathbf{J}_R^k\Delta t & \mathbf{0} & \mathbf{0} \\ -\mathbf{f}_p(\tilde{\boldsymbol{\alpha}}_k)^{\wedge} & \boldsymbol{\chi}_k & \mathbf{J}_p(\tilde{\boldsymbol{\alpha}}_k)\end{bmatrix}}_{\mathbf{B}}\begin{bmatrix}\boldsymbol{\eta}^{\omega} \\ \boldsymbol{\eta}^{\alpha} \\ \boldsymbol{\eta}^{\dot{\alpha}}\end{bmatrix}
\end{aligned} \tag{57}$$

where:

$$\boldsymbol{\chi}_k = \mathbf{H}_p(\tilde{\boldsymbol{\alpha}}_k)\tilde{\dot{\boldsymbol{\alpha}}}_k + \tilde{\boldsymbol{\omega}}_k^{\wedge}\mathbf{f}_p(\tilde{\boldsymbol{\alpha}}_k) - (\mathbf{b}^{\omega})^{\wedge}\mathbf{J}_p(\tilde{\boldsymbol{\alpha}}_k) \tag{58}$$

Eq. (57) can be expressed more compactly as:

$$\boldsymbol{\eta}_{i,k+1}^{\Delta} = \mathbf{A}\boldsymbol{\eta}_{i,k}^{\Delta} + \mathbf{B}\boldsymbol{\eta}_k \tag{59}$$

where $\boldsymbol{\eta}_k = [\boldsymbol{\eta}^{\omega}\ \boldsymbol{\eta}^{\alpha}\ \boldsymbol{\eta}^{\dot{\alpha}}]^{\mathsf{T}}$.

From the linear expression (59) and given the covariance $\boldsymbol{\Sigma}_{\boldsymbol{\eta}} \in \mathbb{R}^{9\times 9}$ of the raw gyro and joint states noises $\boldsymbol{\eta}_k$, the covariance for the factor can be computed iteratively:

$$\boldsymbol{\Sigma}_{i,k+1} = \mathbf{A}\boldsymbol{\Sigma}_{i,k}\mathbf{A}^{\mathsf{T}} + \mathbf{B}\boldsymbol{\Sigma}_{\boldsymbol{\eta}}\mathbf{B}^{\mathsf{T}} \tag{60}$$

starting from the initial condition $\boldsymbol{\Sigma}_{i,i} = \mathbf{0}$.

C. Observability of the Linear Velocity Bias

In Eq. (26) we have introduced the velocity bias term $\mathbf{b}^{v}$ to relax the non-slip relation from Eq. (16). Intuitively, $\mathbf{b}^{v}$ represents the velocity of the contact point. If we rewrite Eq. (26) and ignore Gaussian noise terms, the kinematic velocity bias $\mathbf{b}^{v}$ can be expressed as:

$$\mathbf{b}^{v} = \tilde{\mathbf{v}}_{kin} - \mathbf{v} \tag{61}$$

where $\tilde{\mathbf{v}}_{kin}$ is the estimated velocity from kinematics and $\mathbf{v}$ is the true velocity.

Bloesch *et al.* [1] proved that, if the contact point is stationary (i.e., $\mathbf{b}^{v} = 0$), absolute position and yaw are the only unobservable states (excluding degenerate cases). Therefore, kinematic-inertial measurements alone can only estimate $\tilde{\mathbf{v}}_{kin}$. The same unobservable states have been proved by Hesch *et al.* [54] for a typical visual-inertial system. Therefore, the robot's linear velocity $\mathbf{v}$ can be observed from IMU and camera measurements only.

Since VILENS combines kinematic, inertial, and camera measurements, we speculate that the linear velocity bias $\mathbf{b}^{v}$ should be observable, as confirmed empirically in our experiments. An analytic proof of observability and convergence is left to future work.

ACKNOWLEDGMENT

This research was part funded by the EU H2020 Projects THING and MEMMO the UKRI-funded ORCA Robotics Hub (EP/R026173/1). Fallon was supported by a Royal Society University Research Fellowship and Wisth by a Google DeepMind studentship. This research has been conducted as part of the ANYmal research community. Special thanks to the CERBERUS DARPA SubT Team for providing the data from the challenge runs and the hardware support.

REFERENCES

[1] M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, and R. Siegwart, "State estimation for legged robots on unstable and slippery terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 6058–6064.

[2] M. Bloesch, M. Burri, H. Sommer, R. Siegwart, and M. Hutter, "The two-state implicit filter recursive estimation for mobile robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 573–580, 2018.

[3] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended kalman filtering for robot state estimation," *The International Journal of Robotics Research*, vol. 39, no. 4, pp. 402–430, 2020.
[4] M. Camurri *et al.*, "Probabilistic contact estimation and impact detection for state estimation of quadruped robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1023–1030, 2017.
[5] F. Jenelten, J. Hwangbo, F. Tresoldi, C. D. Bellicoso, and M. Hutter, "Dynamic locomotion on slippery ground," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4170–4176, 2019.
[6] J.-H. Kim *et al.*, "Legged robot state estimation with dynamic contact event information," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6733–6740, 2021.
[7] R. Hartley, M. G. Jadidi, L. Gan, J. Huang, J. W. Grizzle, and R. M. Eustice, "Hybrid contact preintegration for visual-inertial-contact state estimation using factor graphs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 3783–3790.
[8] M. Camurri, M. Ramezani, S. Nobili, and M. Fallon, "Pronto: A multi-sensor state estimator for legged robots in real-world scenarios," *Frontiers in Robotics and AI*, vol. 7, pp. 1–18, 2020.
[9] S. Khattak, H. Nguyen, F. Mascarich, T. Dang, and K. Alexis, "Complementary multi–modal sensor fusion for resilient robot pose estimation in subterranean environments," in *International Conference on Unmanned Aircraft Systems*, 2020, pp. 1024–1029.
[10] M. Tranzatto *et al.*, "CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge," *Field Robotics*, vol. 2, pp. 274–324, 2022.
[11] F. Dellaert and M. Kaess, "Factor graphs for robot perception," *Foundations and Trends in Robotics*, vol. 6, pp. 1–139, 2017.
[12] S. Fahmi, G. Fink, and C. Semini, "On state estimation for legged locomotion over soft terrain," *IEEE Sensors Letters*, vol. 5, no. 1, pp. 1–4, 2021.
[13] T. Koolen *et al.*, "Design of a momentum-based control framework and application to the humanoid robot atlas," *International Journal of Humanoid Robotics*, vol. 13, no. 1, pp. 1–34, 2016.
[14] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual–inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.
[15] D. Wisth, M. Camurri, and M. Fallon, "Preintegrated velocity bias estimation to overcome contact nonlinearities in legged robot odometry," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 392–398.
[16] D. Wisth, M. Camurri, S. Das, and M. Fallon, "Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1004–1011, 2021.
[17] S. Gangapurwala, M. Geisert, R. Orsolino, M. Fallon, and I. Havoutis, "RLOC: Terrain-aware legged locomotion using reinforcement learning and optimal control," *IEEE Transactions on Robotics*, 2022, (to appear).
[18] T. Dang, M. Tranzatto, S. Khattak, F. Mascarich, K. Alexis, and M. Hutter, "Graph-based subterranean exploration path planning using aerial and legged robots," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1363–1388, 2020.
[19] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.
[20] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, pp. 3565–3572.
[21] R. Hartley *et al.*, "Legged robot state-estimation through combined forward kinematic and preintegrated contact factors," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 4422–4429.
[22] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast semi-direct monocular visual odometry," in *IEEE International Conference on Robotics and Automation*, 2014, pp. 15–22.
[23] M. Fourmy, T. Flayols, N. Mansard, and J. Solà, "Contact forces pre-integration for the whole body estimation of legged robots," in *IEEE International Conference on Robotics and Automation*, 2021.
[24] F. Grimminger *et al.*, "An open torque-controlled modular robot architecture for legged locomotion research," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3650–3657, 2020.
[25] M. Xie, A. Escontrela, and F. Dellaert, "A factor-graph approach for optimization problems with dynamics constraints," *arXiv*, 2020.
[26] D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robotics and Automation Letters*, pp. 4507–4514, 2019.
[27] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "ISAM2: Incremental smoothing and mapping using the Bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.
[28] J. Hwangbo, C. D. Bellicoso, P. Fankhauser, and M. Hutter, "Probabilistic foot contact estimation by fusing information from dynamics and differential/forward kinematics," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 3872–3878.
[29] G. Bledt, P. M. Wensing, S. Ingersoll, and S. Kim, "Contact model fusion for event-based locomotion in unstructured terrains," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 4399–4406.
[30] P. Furgale, "Representing Robot Pose: The good, the bad, and the ugly," 2014. [Online]. Available: http://paulfurgale.info/news/2014/6/9/representing-robot-pose-the-good-the-bad-and-the-ugly
[31] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer, 2008.
[32] T. D. Barfoot and P. T. Furgale, "Associating uncertainty with three-dimensional poses for use in estimation problems," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 679–693, 2014.
[33] J. Kannala and S. S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 8, pp. 1335–1340, 2006.
[34] Z. Zhang, H. Rebecq, C. Forster, and D. Scaramuzza, "Benefit of large field-of-view cameras for visual odometry," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 801–808.
[35] C. J. Taylor and D. J. Kriegman, "Minimization on the lie group SO(3) and related manifolds," Yale University, Tech. Rep. 9405, 1994.
[36] M. Kaess, "Simultaneous localization and mapping with infinite planes," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 4605–4611.
[37] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and R. Daniela, "LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 5135–5142.
[38] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP Variants on Real-World Data Sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.
[39] K. MacTavish and T. D. Barfoot, "At all Costs: A comparison of robust cost functions for camera correspondence outliers," in *Conference on Computer and Robot Vision*, 2015, pp. 62–69.
[40] M. Frigerio, J. Buchli, D. G. Caldwell, and C. Semini, "RobCoGen: A code generator for efficient kinematics and dynamics of articulated robots, based on Domain Specific Languages," *Journal of Software Engineering for Robotics*, vol. 7, no. 1, pp. 36–54, 2016.
[41] S. M. Neuman, T. Koolen, J. Drean, J. E. Miller, and S. Devadas, "Benchmarking and workload analysis of robot dynamics algorithms," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 5235–5242.
[42] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.
[43] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1280–1286.
[44] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The Newer College Dataset: Handheld lidar, inertial and vision with ground truth," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 4353–4360.
[45] A. Martinelli, "Closed-form solution of visual-inertial structure from motion," *International Journal of Computer Vision*, vol. 106, no. 2, pp. 138–152, Jan 2014.
[46] M. Hutter *et al.*, "ANYmal – a highly mobile and dynamic quadrupedal robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 38–44.
[47] J. Zhang and S. Singh, "Low-drift and real-time lidar odometry and mapping," *Autonomous Robots*, vol. 41, no. 2, pp. 401–416, 2017.
[48] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback," *International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.
[49] M. Ramezani, G. Tinchev, E. Iuganov, and M. Fallon, "Online lidar-slam for legged robots with robust registration and deep-learned loop closure," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 4158–4164.

[50] O. Melon, R. Orsolino, D. Surovik, M. Geisert, I. Havoutis, and M. Fallon, "Receding-Horizon Perceptive Trajectory Optimization for Dynamic Legged Locomotion with Learned Initialization," in *IEEE International Conference on Robotics and Automation*, June 2021.
[51] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3019–3026, 2018.
[52] M. Palieri *et al.*, "LOCUS: A multi-sensor lidar-centric solution for high-precision odometry and 3d mapping in real-time," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 421–428, 2021.
[53] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Review*, vol. 51, no. 3, pp. 455–500, 2009.
[54] J. A. Hesch, D. G. Kottas, S. L. Bowman, and S. I. Roumeliotis, "Camera-imu-based localization: Observability analysis and consistency improvement," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 182–201, 2014.



David Wisth (Student Member, IEEE) received the B.S. degree in electrical engineering in 2012 and the M.Eng. degree in mechatronics engineering in 2014 from the University of Melbourne, Australia. He is currently pursuing the DPhil. degree in engineering science from the University of Oxford, UK.

From 2015 to 2017, he worked as a Project Engineer at Siemens, Australia. His research interests include multi-sensor fusion, factor graphs, and legged robots.

Mr Wisth's awards and honors include graduating as valedictorian from his M.Eng. degree, and Best Student Paper Award (Finalist) at the IEEE International Conference on Robotics and Automation in 2021.



Marco Camurri (Member, IEEE) received the B.Eng. and M.Eng. degrees in Computer Engineering from the University of Modena and Reggio Emilia (Modena, Italy) in 2009 and 2012, respectively. He received his PhD degree in Advanced Robotics in 2017 from the Istituto Italiano di Tecnologia (IIT) in Genoa, Italy.

From 2017 to 2020 he was postdoc at IIT (one year) and University of Oxford (two years).

Since 2021, he is Senior Research Associate at the Oxford Robotics Institute, University of Oxford. His research interests include legged state estimation, mobile robot perception, sensor fusion, mapping, and mobile autonomous navigation.



Maurice Fallon (Member, IEEE) received his B.Eng (electrical engineering) from University College Dublin, Ireland and his PhD (acoustic source tracking) from University of Cambridge, UK.

From 2008–2012 he was a post-doc and research scientist in MIT Marine Robotics Group working on SLAM. Later, he was the perception lead of MIT's team in the DARPA Robotics Challenge. From 2017 he has been a Royal Society University Research Fellow and Associate Professor at University of Oxford, UK. He leads the Dynamic Robot Systems Group in Oxford Robotics Institute. His research focuses on probabilistic methods for localization, mapping, dynamic motion planning, and multi-sensor fusion.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	VILENS: Visual, Inertial, Lidar, and Leg Odometry for All-Terrain Legged Robots
Publication Status	Submitted for Publication
Publication Details	D. Wisth, M. Camurri, and M. Fallon, "VILENS: Visual, inertial, lidar, and leg odometry for all-terrain legged robots," *IEEE Transactions on Robotics*, 2021.

Student Confirmation

Student Name:	David Wisth		
Contribution to the Paper	My contributions to the paper were: • Developed the idea behind the paper with co-authors • Implemented the algorithm • Assisted Dr Marco Camurri implementing kinematics module • Generated experimental results and ground truth • Performed the experiments with co-authors • Wrote the paper with co-authors		
Signature		Date	30/09/2021

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Maurice Fallon			
Supervisor comments This is an accurate desciption			
Signature		Date	4/10/2021

This completed form should be included in the thesis, at the end of the relevant chapter.

6.2 Discussion

As mentioned in Chapter 4, the full derivation of the preintegrated twist factors is provided in Appendix A.

This chapter presented a comprehensive multi-sensor fusion algorithm, capable of fusing lidar, camera, kinematic, and inertial measurements across a wide range of environments. The breadth of functionality allows the algorithm to adapt to different platforms with a variety of sensor configurations. Additionally, it has been shown to run on-board the ANYmal robot and integrate with terrain mapping and path planning software. It also has the potential to significantly improve performance during highly dynamic motion or while walking on deformable terrain.

The next chapter will detail other real-world deployments of VILENS, including on drones, as well as describe extensions of the algorithm that were not mentioned in the above chapters.

Chapter 7

Real-World Algorithm Deployment

This chapter describes the real-world deployment of the VILENS algorithm. It has been used on a wide variety of different platforms for state estimation and ground truth generation.

Section 7.1 will first present an overview of the VILENS software architecture, highlighting its modular design which makes it suitable for many different platforms and environments. Section 7.2 details ground truth generation using VILENS, and finally real-world deployments of the algorithm are presented in Section 7.3. The articles referenced during this chapter are attached in Appendix B.

7.1 Software Architecture

The VILENS software package is designed to be versatile and easily reconfigurable for new platforms. It is also specifically designed to easily allow new functionality to be added, including new sensor inputs and new front-end processing algorithms. The logo for the VILENS algorithm is shown in Figure 1.2.

Figure 7.1 shows the VILENS software architecture, which is based on modular measurement processing units called *handlers*. The handlers process different inputs, in different threads, to produce factors for the back-end optimisation. This is enabled by the modular factor graph optimisation library GTSAM [36], which allows different factors to be added without modification of the back-end code.

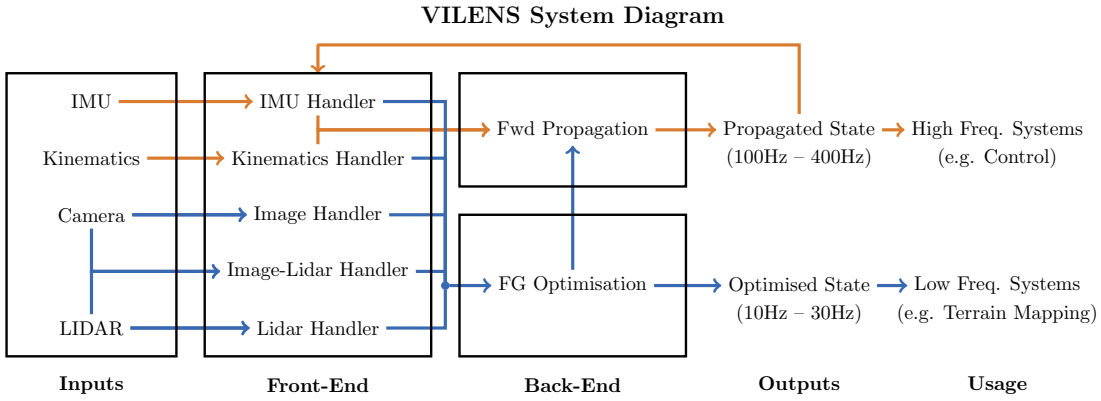


Figure 7.1: VILENS takes input from a number of different sensors, depending on the robot configuration. These measurements are processed in separate threads, then the output is fed into a sliding window optimisation to produce the state estimate. Note: Orange represents high frequency signals (100–400 Hz), and blue represents low frequency signals (10–30 Hz).

This flexibility has allowed VILENS to be easily extended for different applications. This includes Buchanan et al. [151] who added support for deep-learned IMU factors, and Zhang et al. [8] who extended VILENS for multi-camera VIO (see Section 7.3.4). Adding this functionality only required a new handler child class, and a new ROS subscriber if using a different measurement type. The rest of the algorithm is generic and did not require modification. Specific settings for each platform are controlled via a YAML configuration file, allowing sensor modalities, parameters, and calibrations to be easily updated without recompilation.

The next section will present how VILENS can be used for ground truth generation, which has not been described in previous chapters.

7.2 Ground Truth Generation using VILENS

A significant use case of VILENS is ground truth generation. Common methods of ground truth generation include laser tracking (e.g., from a stationary laser), motion capture systems, and offline alignment to a ground truth map.

Laser tracking is highly accurate but limited to datasets where there is line of sight visibility to a stationary laser tracker [13]. Additionally, laser scanners only measure position (not orientation) at low frequency which limits the usefulness of

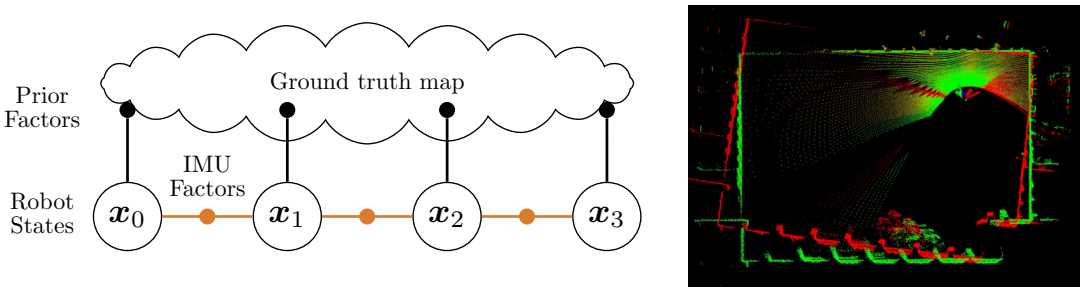


Figure 7.2: *Left:* In localisation mode, VILENS uses prior factors and ICP alignment to connect the poses to the map, and preintegrated IMU factors between successive nodes for smoothness and gravity alignment. *Right:* An example of the original distorted lidar cloud (*red*), compared to the undistorted lidar cloud (*green*). In this example, the sensor rig is being rotated with high angular velocity (over $3\,\mathrm{rad\,s^{-1}}$) in the New College quadrangle (shown in Figure 7.4).

the ground truth. On the other hand, motion capture systems (e.g., Vicon) provide high frequency, high accuracy ground truth but are usually limited to small lab environments where the system can be permanently set up.

Instead, this thesis focuses on ground truth generation through offline alignment to a prior map. In this work, an accurate gravity-aligned 3D map of the environment is built using a commercial laser scanner, such as the Leica BLK360. By matching lidar scans to a ground truth map, VILENS can generate ground truth poses across a broad range of environments. This is important for challenging indoor and outdoor scenarios, which may not be suitable for hardware-based ground truth capture.

Ground truth generation using ICP to a prior map has been presented previously by Ramezani et al. [7]. This method was highly effective in slow-moving datasets, but frequently failed when the motion was too dynamic. This was mainly due to a lack of lidar motion undistortion and poor quality motion prediction. This is particularly important for ICP-based methods which can have a very small basin of convergence.

This thesis extends this approach to generate ground truth using both lidar and IMU measurements. Figure 7.2 (left) shows the factor graph structure, including prior factors from ICP to link the poses and ground truth map, and IMU measurements for smoothness and gravity alignment. The high-frequency IMU measurements allow for motion correction of lidar point clouds (see Figure 7.2 (right)), and to improve the initial estimate for ICP matching. The accuracy of

this method is maximised by smoothing all poses inside a global *batch optimisation* to ensure a smooth and consistent trajectory.

This method was successfully used to create new, highly dynamic sequences for the Newer College Dataset[1] (including angular velocities of over $3\,\mathrm{rad\,s^{-1}}$). This sequence caused ICP alignment alone [7] to quickly diverge due to point cloud distortion and inaccurate prior motion estimates. This method has also been used to generate ground truth for Chapters 5 and 6, Section 7.3.4, as well as Buchanan et al. [151] and Tao et al. [152].

7.3 Practical Deployments

The work in this thesis has been motivated by, and implemented in, many significant experiments. This section will describe some highlights of the experimental contributions of this thesis. This includes sewer inspection (Section 7.3.1), 3D reconstruction (Section 7.3.2), the DARPA Subterranean Challenge (Section 7.3.3), multi-camera VIO (Section 7.3.4), and integration with lidar SLAM (Section 7.3.5).

7.3.1 Underground Sewer Inspection with Legged Robots

Sewer inspection is a challenging and dangerous job for humans to perform. This work aimed to demonstrate autonomous sewer inspection using the ANYmal legged robot. This work was conducted in Zürich, Switzerland as part of the THING H2020 EU project.

An important part of sewer inspection is assessing the condition of the sewer's concrete walls – any cracks or abrasions are a sign that preventive maintenance needs to be performed. This deterioration is often due to Microbial Induced Corrosion (MIC) by bacteria found in the wastewater [153]. This bacteria forms a thick, opaque biofilm on the concrete's surface, meaning that visual methods cannot be used for inspection.

The solution involved "scratching" the surface of the sewer using specially designed waterproof feet containing an IMU. The idea was that moving the foot over

[1] Available here: `https://ori-drs.github.io/newer-college-dataset/`

a rough surface would induce vibrations which could be detected in the IMU data. A Support Vector Machine (SVM) was trained to classify the concrete condition into five different categories, based on labelled data from human experts. Figure 7.3 shows the reconstructed 3D map of the sewer with the concrete inspection points marked by coloured circles.

This paper extended the previous work of Kolvenbach et al. [154] by adding an autonomy system to the ANYmal robot to demonstrate a complete semi-autonomous sewer inspection mission. The autonomy front-end was based on the Director interface [155]. This allowed an operator on street-level to simply set waypoints in the 3D map and the robot would walk to the location, automatically inspecting the sewer along the way.

This work did not contribute to the core body of this thesis, but instead served as an important motivation for this work.

One of the main challenges in sewer inspection is state estimation. The long, repetitive geometry of the sewer tunnels made lidar odometry difficult. Also, the thick fog, which was often present in the sewer, reduced the visibility of the cameras to $\sim$1.5 m which made visual odometry very difficult. This was a motivator for the full VILENS system presented in Chapter 6 which can use all available sensor information, even in degenerate scenarios such as sewers.

The accompanying paper [16] is attached in Appendix B.1.

7.3.2 Outdoor Mapping using a Handheld Sensor Rig

An important part of the development of state estimation algorithms is a high quality dataset for performance evaluation. Part of this thesis involved the creation of a new, high-quality dataset for visual-lidar-inertial SLAM and 3D reconstruction.

The dataset (shown in Figure 7.4) was recorded using the “Rooster” sensor rig which contained an Ouster lidar, Realsense D435i stereo-inertial camera, and on-board Intel NUC computer. These were housed in a 3D-printed case suitable for handheld operation. Importantly, a high-quality 3D map of the environment was created using a commercial laser scanner. This map was used to generate

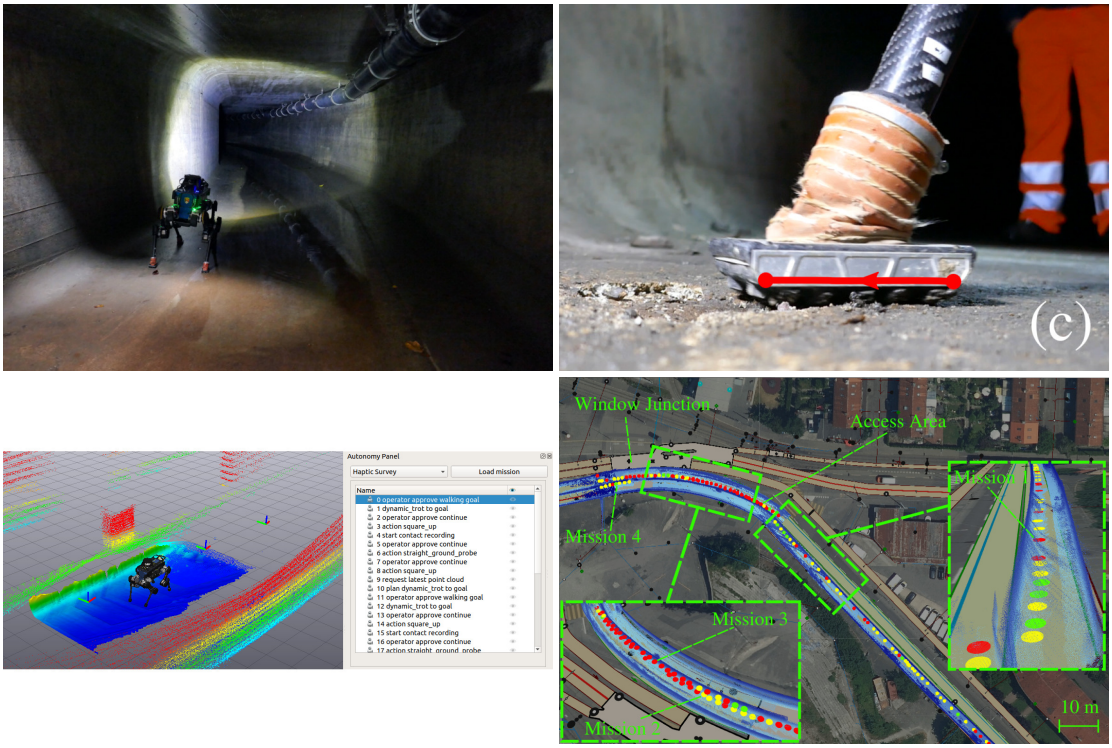


Figure 7.3: Summary of Kolvenbach et al. [16] where the ANYmal robot performed semi-autonomous sewer inspection. *Top-left:* Exploration of a long, underground sewer. *Top-right:* Specially designed feet inspected the concrete using a scratching motion. *Bottom-left:* Supervised autonomy was developed using the Director interface [155]. *Bottom-right:* Over three different missions an accurate 3D map of the sewer was created. Circles represent inspection points which are classified as: Fair (*Red*), Satisfactory (*Yellow*), and Good (*Green*) condition.

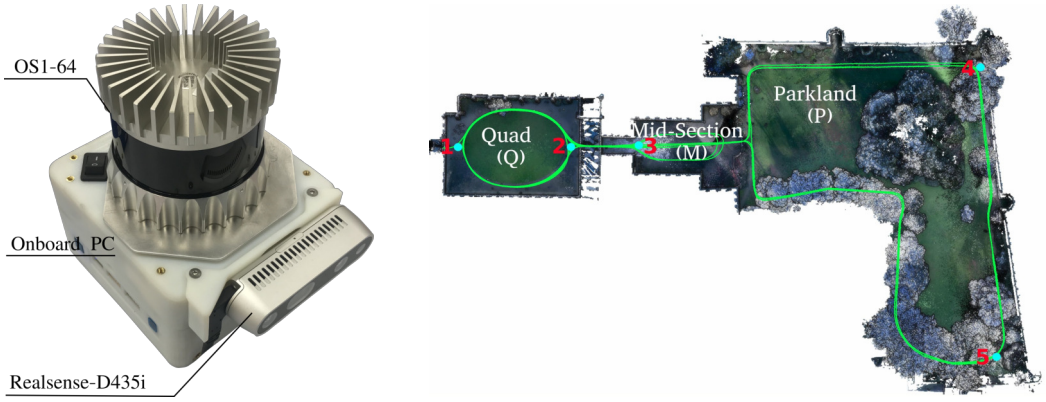


Figure 7.4: *Left:* The Newer College dataset [7] was recorded using the Rooster handheld sensor, which contains a lidar, stereo camera, and IMU. *Right:* An accurate ground truth map was generated using a commercial Leica BLK-360 laser scanner.

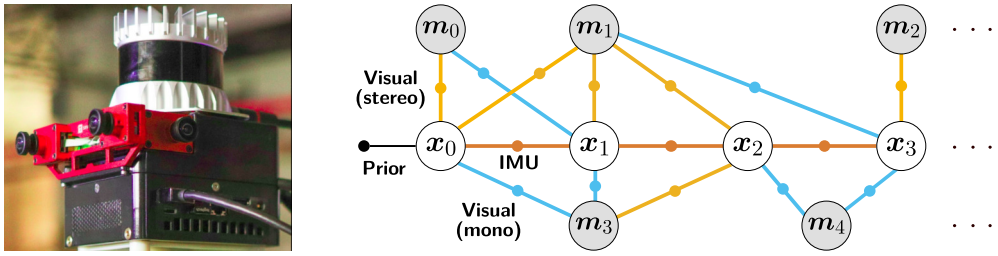


Figure 7.5: *Left:* The multi-camera rig used for experiments [8] contained four cameras (frontal stereo and two lateral) and a lidar. Note that the lidar was only used for ground truth generation. *Right:* Features from all cameras and an IMU are tightly fused in a single factor graph.

ground truth for both SLAM and 3D reconstruction algorithms. VILENS was evaluated on this dataset in Chapter 5.

The accompanying paper [7] is attached in Appendix B.2. As of September 2021, the Newer College dataset has been downloaded 675 times since it was released in Spring 2020.

7.3.3 DARPA Subterranean Challenge

The DARPA Subterranean Challenge [156] is an international robotics competition to motivate the development of the multi-robot field exploration for disaster response. The teams have 60 min to send a team of semi-autonomous robots to find artefacts in an underground environment and report their location.

All robots needed to be operated remotely from a fixed base station with a single human operator. This required significant autonomy, relying on the robot for most tasks including state estimation, object detection, and mapping.

Motivated by this competition, Chapters 5 and 6 presented results from datasets collected during the DARPA Subterranean Challenge's Urban Circuit.

The accompanying paper describing the CERBERUS team's approach to Phase I and II of the challenge [17] is attached in Appendix B.3. In September 2021, the CERBERUS team won the Systems Track of the DARPA Subterranean Challenge finals.

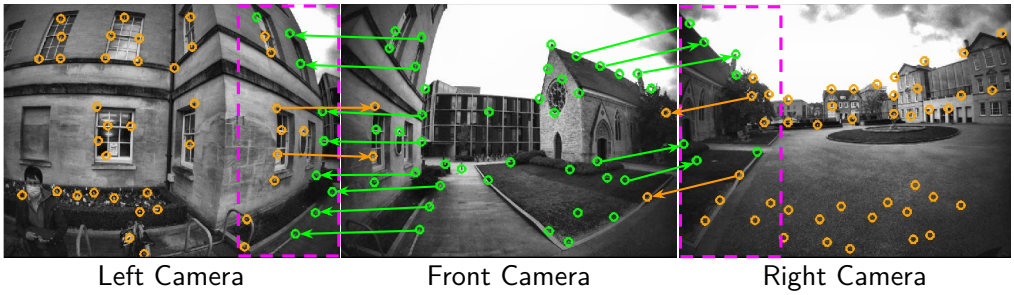


Figure 7.6: VILENS-MC tracks features across cameras, taking advantage of overlapping image regions. This both improves accuracy and reduces duplicate landmark tracking.

7.3.4 Multi-Camera Visual Inertial Odometry

The most significant problem for traditional mono/stereo visual odometry (or VIO) systems is a single point of failure. If the camera cannot track features then motion estimation relies only on IMU measurements, and can quickly diverge. This situation is common in practice, especially in robotics, when it is not always easy to control what the camera is looking at. For instance, most path planning algorithms used in autonomous exploration do not consider state estimation performance when planning camera viewpoints.

To avoid this single point of failure, and to increase the robustness and accuracy of VILENS we introduced VILENS Multi-Camera (VILENS-MC). Natural robustness to individual camera failures was handled by tightly fusing all visual features into the same factor graph (see Figure 7.5). This avoided the need for hard-switching between cameras and allowed each camera to contribute partial constraints. This was important if many cameras are obscured at once. Figure 7.6 shows how VILENS-MC could track features across cameras, which avoided duplicate landmark tracking and improved accuracy.

Adding multiple cameras can significantly increase robustness and reduce drift. In our challenging datasets, we reduced mean RPE by 45 % / 50 % in translation/rotation compared to the baseline (VILENS with only stereo camera and IMU inputs). We also outperformed state-of-the-art stereo VIO approaches (such as OpenVINS [46]) by up to 80 % / 39 % in translational/rotational RPE, and provided accurate state estimation when stereo ORB-SLAM3 [47] failed.

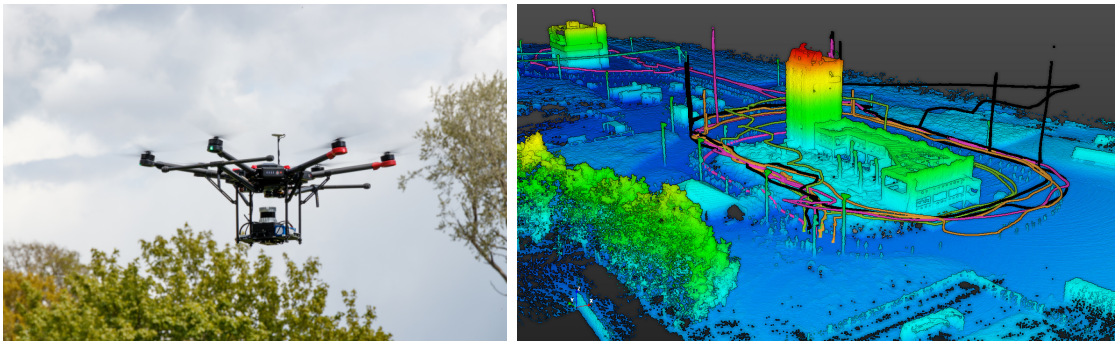


Figure 7.7: *Left:* The DJI drone platform is equipped with a wide field of view lidar (Ouster OS1-64), downward facing stereo fisheye camera (Realsense T265), and forward facing stereo camera (Realsense D435i). *Right:* The estimated trajectories from *multi-session mapping* – where the first run is used to create a map, and subsequent runs are localised to the map. A video summary of this figure is available at: `https://youtu.be/jeMzQRKKAnI`.

The accompanying paper [8] is attached in Appendix B.4.

7.3.5 Integration with Lidar SLAM

This section demonstrates how odometry and motion-corrected point clouds from VILENS can be used as an input to a global lidar SLAM system [7]. These globally accurate pose estimates are useful for many tasks, including mapping and navigation.

An example of VILENS and SLAM deployed on an aerial vehicle is shown in Figure 7.7, where lidar loop closures were able to correct any long-term odometry drift. The aim is to use this system for 3D reconstruction of large outdoor areas, such as disaster sites and forests.

Similarly, VILENS and SLAM have been used for a forest inspection system that automatically measures the diameter of trees for ecological surveys [157].

HILTI SLAM Challenge

The combination of VILENS and SLAM has also been tested on the publicly available HILTI SLAM Challenge dataset [158]. This dataset contained two lidars (a rotational and solid state lidar), five cameras, and an IMU. The setup, shown in Figure 7.8, is similar to the sensor setup that was used for the VILENS multi-camera publication (see Figure 7.5). The dataset included 12 sequences from a variety of indoor and outdoor urban environments.

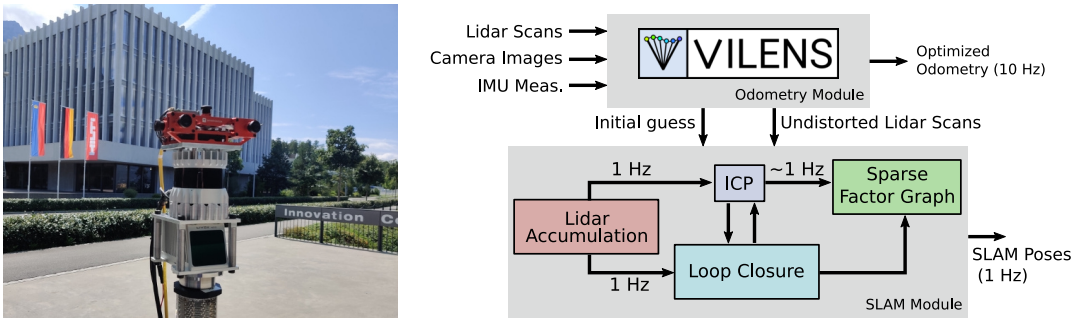


Figure 7.8: The HILTI SLAM Challenge was based on data from a portable sensor rig containing two lidars, five fisheye cameras, and an IMU (*left*). This information was processed by VILENS combined with an external SLAM system [7] (*right)*).

VILENS with lidar SLAM [159] won the academic track of the HILTI SLAM Challenge 2021 by achieving the most accurate state estimate. It achieved lower Absolute Trajectory Error (ATE) than other lidar-visual-inertial approaches such as VIRAL-SLAM [160] and Maplab [161] while maintaining real-time performance.

7.4 Discussion

This chapter has presented many significant extensions of VILENS including localisation to a prior map for ground truth generation, and VILENS-MC, a multi-camera VIO system. These extensions were possible through the efficient and modular software architecture that allows sensor modalities to be easily changed.

VILENS has been deployed in many significant real-world applications, including handheld mapping, forest inspection, and drone localisation. It has been motivated by challenging experimental conditions in legged robot sewer inspection, and evaluated on data from the DARPA Subterranean Challenge and the open-source Newer College Dataset.

Chapter 8

Conclusion and Future Work

8.1 Future Work

VILENS is a versatile and modular multi-sensor fusion algorithm and more research could be conducted on any of its components. Several promising directions to improve VILENS in future work are listed below.

8.1.1 Automate Configuration for New Platforms

I. Online intrinsic and extrinsic calibration: Given the rate of technological change, and the wide variety of platforms that VILENS can be deployed on, a common and time-consuming task is creating new configuration files for each new setup. This can sometimes be a limiting factor in deploying advanced multi-sensor fusion methods on new platforms.

The main challenge is determining the sensors' intrinsic and extrinsic calibration, especially when an accurate CAD model does not exist. We currently perform offline calibration using a combination of Kalibr for camera-IMU calibration [24] and CAD modelling.

Future work could, instead, involve estimating the calibration online in the optimisation. Similar online calibration tools are present in other VIO algorithms [44, 46], but there are only a few examples of this applied to multiple sensors simultaneously [162]. This would require extending existing factors to also optimise the calibration.

II. Automatic parameter tuning: Tuning parameters is an important but time-consuming step in state estimation. Parameters are usually found by hand-tuning which is not optimal and may lead to decreased performance.

These parameters could instead be found using offline Bayesian hyperparameter tuning tools, such as those originally developed for deep learning [163]. Ground truth and evaluation metrics are already available for many different VILENS datasets.

8.1.2 Extend Lidar Odometry

I. Lidar Submapping A common approach in many state-of-the-art lidar SLAM systems is submapping [3,54]. This can significantly reduce drift, especially when the robot stays within the submap region. Future work could involve a tight integration of submapping into the VILENS pipeline. This may require carefully managing the computation to reduce drift, while keeping the high frequency of state estimation.

II. Improve Lidar Feature Tracking Tracking geometric features in point clouds is a challenging task. Most lidar odometry methods only perform frame-to-frame feature tracking, and do not track features over time [3, 48]. Additionally, the geometric plane and line features currently tracked in VILENS do not generalise well to unstructured environments. Future work could use deep learning-based feature tracking [73], or draw inspiration from the 3D lidar object tracking literature [164].

8.1.3 Exploit Correlation Between Sensors

I. Integration with Terrain Mapping A promising direction of future work is the combination of state estimation and terrain mapping. Currently, ANYmal's terrain mapping pipeline [165] fuses point clouds in a probabilistic manner, using the latest state estimate. This method is very sensitive to state estimator drift, which directly impacts the quality of the map.

In contrast, future work could involve using the optimised state estimate from the entire factor graph's sliding window to calculate the terrain map. This would ensure the poses are smooth and self-consistent. It could also improve state

estimation by tying lidar, vision, and kinematic measurements to a terrain map. For example, the foot contact location could be linked to the terrain map to form an additional constraint.

II. Camera and Lidar Joint Feature Tracking Cameras and lidars are common complementary exteroceptive sensing modalities. Future work may explore how features can be jointly tracked across both camera and lidar. For example, planes and lines could be jointly tracked across both sensor modalities [166].

8.2 Conclusion

This thesis has investigated the fusion of lidar, visual, inertial, and kinematic sensors for robust and accurate state estimation. The proposed algorithm, VILENS, has been tested on a wide variety of platforms across many different challenging environments. The approach taken in this thesis of tightly coupling sensor measurements presents a flexible and intuitive method of using all available sensor information, even in degenerate scenarios. It requires no hard switching or hand-crafted rules which makes it adaptable to new platforms and environments.

The thesis presents a body of research that extends the state of the art in factor graph-based state estimation. Chapters 4 and 5 presented a VIO system along with novel methods of integrating lidar and kinematic measurements tightly into a factor graph. Chapter 6 presented, to the best of the author's knowledge, the first method to tightly fuse lidar, visual, inertial, and kinematic measurements into a factor graph framework. Chapter 7 took a deeper look into the VILENS algorithm and some of its key features and notable deployments.

As mobile platforms (including autonomous robots) become more commonplace they will need to deal with complex and challenging environments on a regular basis. A key enabler is accurate and reliable state estimation, even in challenging scenarios. For example, being robust to hardware or software failures, poor lighting conditions, or highly dynamic motions. Overall, this thesis and the VILENS algorithm represents a step towards enabling robots to venture out into the wide world.

Appendix A

Derivation of Preintegrated Twist Factors

This appendix will provide a detailed derivation of the preintegrated twist factors presented in Chapters 4 and 6.

Online estimation of the biases is important since the kinematic bias depends on variety of factors, including the terrain and gait. Since the precise terrain and gait properties are very difficult to both observe and model, an offline bias estimation is likely to improve estimation in some circumstances, but decrease accuracy in others.

Section A.1 will first define the measurement model, then the preintegrated measurement and noise functions will be derived in Section A.2. Section A.3 will show how the preintegrated measurements can be updated with small changes in bias, avoiding the need for costly recomputation if the bias changes. Finally, Section Section A.4 will outline the residual and Jacobians for the factor.

The following section will use the mathematical conventions defined in Section 2.1 and present the true value as x and the measured value (corrupted by noise and bias) as $\tilde{x}$.

A.1 Measurement Model

Based on extensive experimental evaluation [13, 15], the preintegrated twist factor uses the following measurement model for leg odometry, consisting of twist

measurements with a systematic bias,

$$\tilde{\mathbf{v}} = \mathbf{v} + \mathbf{b}^v + \boldsymbol{\eta}^v, \tag{A.1}$$
$$\tilde{\boldsymbol{\omega}} = \boldsymbol{\omega} + \mathbf{b}^\omega + \boldsymbol{\eta}^\omega \tag{A.2}$$

This is similar to a commonly used model for IMU measurements ([20] Eq. (27), (28)). The estimated position $\mathbf{p}_j$ and rotation $\mathbf{R}_j$ at time t_j can be simply found by integrating Equations (A.1) and (A.2),

$$\mathbf{R}_j = \mathbf{R}_i \mathrm{Exp}\left(\int_{t_i}^{t_j} \omega(\tau) d\tau\right) \tag{A.3}$$
$$\mathbf{p}_j = \mathbf{p}_i + \int_{t_i}^{t_j} \mathbf{v}(\tau) d\tau \tag{A.4}$$

where $\mathbf{p}_i$ and $\mathbf{R}_i$ are the position and rotation at a previous time t_i. *Assuming* constant twist between leg odometry measurements (a reasonable assumption if the measurement frequency is high enough to capture the motion of the robot),

$$\mathbf{R}_j \approx \mathbf{R}_i \prod_{k=i}^{j-1} \left[\mathrm{Exp}\left(\omega_k \Delta t\right)\right] \tag{A.5}$$
$$\mathbf{p}_j \approx \mathbf{p}_i + \sum_{k=i}^{j-1} \left[\mathbf{R}_{ik} \mathbf{v}_k \Delta t\right] \tag{A.6}$$

where Δt is the time between measurements. Substituting in the measurement model from Equations (A.1) and (A.2) and defining the discrete time version of the Gaussian noise η^d such that $Cov(\eta^d) = \frac{1}{\Delta t} Cov(\eta)$,

$$\mathbf{R}_j = \mathbf{R}_i \prod_{k=i}^{j-1} \left[\mathrm{Exp}\left((\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_k^\omega - \boldsymbol{\eta}_k^{\omega d})\Delta t\right)\right]$$
$$\Delta \mathbf{R}_{ij} = \prod_{k=i}^{j-1} \left[\mathrm{Exp}\left((\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_k^\omega - \boldsymbol{\eta}_k^{\omega d})\Delta t\right)\right] \tag{A.7}$$

where $\Delta \mathbf{R}_{ij} = \mathbf{R}_i^T \mathbf{R}_j$. Similarly for position,

$$p_j = p_i + \sum_{k=i}^{j-1} \left[R_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_k^v - \boldsymbol{\eta}_k^{vd})\Delta t\right]$$
$$\Delta \mathbf{p}_{ij} = \sum_{k=i}^{j-1} \left[\Delta R_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_k^v - \boldsymbol{\eta}_k^{vd})\Delta t\right] \tag{A.8}$$

where $\Delta \mathbf{p}_{ij} = \mathbf{R}_i^T(\mathbf{p}_j - \mathbf{p}_i)$.

A.2 Preintegrated Measurement and Noise

To simplify the analysis, the preintegrated twist factor *assumes* constant bias between time t_i and t_j. This is valid if the bias changes much slower than the optimisation frequency ($\sim$30 Hz), which is generally true in practice.

Using the first order SO(3) approximation ([20] Eq. (7)),

$$\mathrm{Exp}\left(\phi + \delta\boldsymbol{\phi}\right) \approx \mathrm{Exp}\left(\phi\right)\mathrm{Exp}\left(J_r(\phi)\delta\boldsymbol{\phi}\right) \tag{A.9}$$

and the right Jacobian definition from [20] Eq. (8),

$$J_r(\phi) = I - \frac{1-\cos(|\phi|)}{|\phi|^2}\phi^\wedge + \frac{|\phi| - \sin(|\phi|)}{|\phi|^3}(\phi^\wedge)^2 \tag{A.10}$$

Equation (A.7) can be simplified to,

$$\begin{aligned}
\Delta\mathbf{R}_{ij} &\approx \prod_{k=i}^{j-1}\left[\mathrm{Exp}\left((\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_i^\omega)\Delta t\right)\mathrm{Exp}\left(-J_r^k\boldsymbol{\eta}_k^{\omega d}\Delta t\right)\right] \\
\Delta\mathbf{R}_{ij} &= \prod_{k=i}^{j-1}\left[\mathrm{Exp}\left((\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_i^\omega)\Delta t\right)\right] \times \prod_{k=i}^{j-1}\left[\mathrm{Exp}\left(-\tilde{\mathbf{R}}_{k+1,j}^T J_r^k\boldsymbol{\eta}_k^{\omega d}\Delta t\right)\right] \\
\Delta\mathbf{R}_{ij} &= \Delta\tilde{\mathbf{R}}_{ij}\mathrm{Exp}\left(-\delta\boldsymbol{\phi}_{ij}\right)
\end{aligned} \tag{A.11}$$

where $J_r^k = J_r\left((\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_k^\omega)\Delta t\right)$. Therefore, the *preintegrated rotation measurement and noise* terms are,

$$\Delta\mathbf{R}_{ij} \doteq \prod_{k=i}^{j-1}\left[\mathrm{Exp}\left((\tilde{\boldsymbol{\omega}}_k - \mathbf{b}_i^\omega)\Delta t\right)\right] \tag{A.12}$$

$$\mathrm{Exp}\left(-\delta\boldsymbol{\phi}_{ij}\right) \doteq \prod_{k=i}^{j-1}\left[\mathrm{Exp}\left(-\tilde{\mathbf{R}}_{k+1,j}^T J_r^k\boldsymbol{\eta}_k^{\omega d}\Delta t\right)\right] \tag{A.13}$$

Similarly for the position estimate, if Equation (A.11) is substituted into Equation (A.8),

$$\Delta\mathbf{p}_{ij} \approx \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}\mathrm{Exp}\left(-\delta\boldsymbol{\phi}_{ik}\right)\left(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v - \boldsymbol{\eta}_k^{vd}\right)\Delta t\right] \tag{A.14}$$

Then using the first order approximation of the exponential map $\mathrm{Exp}\left(\phi\right) \approx I + \phi^\wedge$

([20] Eq. (4)) and ignoring higher order terms,

$$\Delta\mathbf{p}_{ij} \approx \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(I-\delta\boldsymbol{\phi}_{ik}^{\wedge})(\tilde{\mathbf{v}}_k-\mathbf{b}_i^v-\boldsymbol{\eta}_k^{vd})\Delta t\right]$$

$$\begin{aligned}\Delta\mathbf{p}_{ij} = &\sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k-\mathbf{b}_i^v)\Delta t\right]-\sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}\boldsymbol{\eta}_k^{vd}\Delta t\right]\\ &-\sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}\delta\boldsymbol{\phi}_{ik}^{\wedge}(\tilde{\mathbf{v}}_k-\mathbf{b}_i^v)\Delta t\right]+\sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}\delta\boldsymbol{\phi}_{ik}^{\wedge}\boldsymbol{\eta}_k^{vd}\Delta t\right]\end{aligned}$$

$$\Delta\mathbf{p}_{ij} \approx \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k-\mathbf{b}_i^v)\Delta t\right]-\sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}\boldsymbol{\eta}_k^{vd}\Delta t+\Delta\tilde{\mathbf{R}}_{ik}\delta\boldsymbol{\phi}_{ik}^{\wedge}(\tilde{\mathbf{v}}_k-\mathbf{b}_i^v)\Delta t\right]$$

Then using the matrix identity $a^{\wedge}b=-b^{\wedge}a$ ([20] Eq. (2)),

$$\begin{aligned}\Delta\mathbf{p}_{ij} &= \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k-\mathbf{b}_i^v)\Delta t\right]+\sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k-\mathbf{b}_i^v)^{\wedge}\delta\boldsymbol{\phi}_{ij}\Delta t-\Delta\tilde{\mathbf{R}}_{ik}\boldsymbol{\eta}_k^{vd}\Delta t\right]\\ \Delta\mathbf{p}_{ij} &= \Delta\tilde{\mathbf{p}}_{ij}-\delta p_{ij}\end{aligned} \tag{A.15}$$

Therefore, the *preintegrated position measurement and noise* terms are,

$$\Delta\tilde{\mathbf{p}}_{ij} \doteq \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k-\mathbf{b}_i^v)\Delta t\right] \tag{A.16}$$

$$\delta p_{ij} \doteq \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}\boldsymbol{\eta}_k^{vd}\Delta t-\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k-\mathbf{b}_i^v)^{\wedge}\delta\boldsymbol{\phi}_{ij}\Delta t\right] \tag{A.17}$$

Substituting Equations (A.11) and (A.15) into Equations (A.7) and (A.8) and using the identity $\mathrm{Exp}\,(-\phi)^T=\mathrm{Exp}\,(\phi)$ the *preintegrated noise model* is,

$$\Delta\tilde{\mathbf{R}}_{ij} = R_i^T R_j \mathrm{Exp}\left(\delta\boldsymbol{\phi}_{ij}\right) \tag{A.18}$$

$$\Delta\tilde{\mathbf{p}}_{ij} = R_i^T(p_j-p_i)+\Delta\mathbf{p}_{ij} \tag{A.19}$$

A.2.1 Iterative Noise Model Update

For real-time operation it is desired to iterative update the noise model when new measurements are received. The *preintegrated noise vector* is defined as,

$$\eta_{ij}^{\Delta} \doteq \left[\delta\boldsymbol{\phi}_{ij}^T,\delta\mathbf{p}_{ij}^T\right]^T/\sim\mathcal{N}(0_{6\times1},\Sigma_{ij}) \tag{A.20}$$

Taking the log of both sides of Equation (A.13) and multiplying by -1,

$$\delta\boldsymbol{\phi}_{ij} = -\mathrm{Log}\left(\prod_{k=i}^{j-1}\left[\mathrm{Exp}\left(-\Delta\tilde{\mathbf{R}}_{k+1,j}^T J_r^k\boldsymbol{\eta}_k^{\omega d}\Delta t\right)\right]\right) \tag{A.21}$$

Then using the approximation from ([20] Eq. (9)),

$$\text{Log}\left(\text{Exp}\left(\phi\right)\text{Exp}\left(\delta\boldsymbol{\phi}\right)\right) \approx \phi + J_r^{-1}(\phi)\delta\boldsymbol{\phi} \tag{A.22}$$

ignore higher order terms, and note that if $\eta \approx \delta\boldsymbol{\phi} \approx 0$ then $J_r^{-1}(\phi) \approx I$, then,

$$\delta\boldsymbol{\phi}_{ij} \approx \sum_{k=i}^{j-1} \left[\Delta\tilde{\mathbf{R}}_{k+1,j}^T J_r^k \boldsymbol{\eta}_k^{\omega d} \Delta t\right] \tag{A.23}$$

Therefore, $\delta\boldsymbol{\phi}_{ij}$ is approximately a linear combination of Gaussian terms and hence is Gaussian itself. To allow for *iterative calculation* online as new measurements are received Equation (A.23) can be rewritten as,

$$\begin{aligned}
\delta\boldsymbol{\phi}_{ij} &= \sum_{k=i}^{j-2} \left[(\Delta\tilde{\mathbf{R}}_{k+1,j-1}\Delta\tilde{\mathbf{R}}_{j-1,j})^T J_r^k \boldsymbol{\eta}_k^{\omega d}\Delta t\right] + \Delta\tilde{\mathbf{R}}_{j,j}^T J_r^{j-1}\boldsymbol{\eta}_{j-1}^{\omega d}\Delta t \\
\delta\boldsymbol{\phi}_{ij} &= \Delta\tilde{\mathbf{R}}_{j-1,j}^T \delta\boldsymbol{\phi}_{i,j-1} + J_r^{j-1}\boldsymbol{\eta}_{j-1}^{\omega d}\Delta t
\end{aligned} \tag{A.24}$$

Similarly, trivially rearranging (A.17) into *iterative form*,

$$\delta\mathbf{p}_{ij} = \delta\mathbf{p}_{i,j-1} - \Delta\tilde{\mathbf{R}}_{i,j-1}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v)^\wedge \delta\boldsymbol{\phi}_{ij}\Delta t + \Delta\tilde{\mathbf{R}}_{i,j-1}\boldsymbol{\eta}_{j-1}^{vd}\Delta t \tag{A.25}$$

For readability, the notation can be converted from j to $k+1$ to obtain,

$$\delta\boldsymbol{\phi}_{ik+1} = \Delta\tilde{\mathbf{R}}_{kk+1}^T\delta\boldsymbol{\phi}_{ik} + J_r^k\boldsymbol{\eta}_k^{\omega d}\Delta t \tag{A.26}$$

$$\delta\mathbf{p}_{ik+1} = \delta p_{ik} - \Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v)^\wedge\delta\boldsymbol{\phi}_{ik}\Delta t + \Delta\tilde{\mathbf{R}}_{ik}\boldsymbol{\eta}_k^{vd}\Delta t \tag{A.27}$$

Rewriting this in matrix form,

$$\begin{aligned}
\begin{bmatrix}\delta\boldsymbol{\phi}_{ik+1}\\ \delta\mathbf{p}_{ik+1}\end{bmatrix} &= \begin{bmatrix}\Delta\tilde{\mathbf{R}}_{kk+1}^T & 0\\ -\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v)^\wedge\Delta t & I\end{bmatrix}\begin{bmatrix}\delta\boldsymbol{\phi}_{ik}\\ \delta\mathbf{p}_{ik}\end{bmatrix} + \begin{bmatrix}J_r^k\Delta t & 0\\ 0 & \Delta\tilde{\mathbf{R}}_{ik}\Delta t\end{bmatrix}\begin{bmatrix}\boldsymbol{\eta}_k^{\omega d}\\ \boldsymbol{\eta}_k^{vd}\end{bmatrix} \\
\eta_{ik+1}^\Delta &= A\eta_{ik}^\Delta + B\eta_k^d
\end{aligned} \tag{A.28}$$

Therefore the measurement covariance can be expressed in the following iterative format,

$$\Sigma_{ik+1} = A\Sigma_{ik}A^T + B\Sigma_\eta B^T \tag{A.29}$$

A.3 Updating Under Small Bias Changes

To avoid costly re-integration every time the bias changes, this section derives how $\Delta\tilde{\mathbf{R}}_{ij}$ and $\Delta\tilde{\mathbf{p}}_{ij}$ change under a small bias update. First, a small change in the bias is applied to see the effect on $\Delta\tilde{\mathbf{R}}_{ij}$,

$$\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega + \delta\mathbf{b}_i^\omega) = \prod_{k=i}^{j-1} \left[\mathrm{Exp}\left((\omega_k - \mathbf{b}_i^\omega - \delta\mathbf{b}_i^\omega)\Delta t\right)\right]$$

Using the small angle approximation ([20] Eq. (7)),

$$\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega + \delta\mathbf{b}_i^\omega) \approx \prod_{k=i}^{j-1} \left[\mathrm{Exp}\left((\omega_k - \mathbf{b}_i^\omega)\Delta t\right) \mathrm{Exp}\left(-J_r^k \delta\mathbf{b}_i^\omega \Delta t\right)\right]$$

rearranging using [20] Eq. (11),

$$\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega + \delta\mathbf{b}_i^\omega) = \Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega) \prod_{k=i}^{j-1} \left[\mathrm{Exp}\left(-\Delta\tilde{\mathbf{R}}_{k+1,j}^T J_r^k \delta\mathbf{b}_i^\omega \Delta t\right)\right]$$

using the first order approximation ([20] Eq. (40) – (42)), and neglecting higher order terms,

$$\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega + \delta\mathbf{b}_i^\omega) \approx \Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega)\mathrm{Exp}\left(\sum_{k=i}^{j-1} \left[-\Delta\tilde{\mathbf{R}}_{k+1,j}^T J_r^k \Delta t\right] \delta\mathbf{b}_i^\omega\right) \tag{A.30}$$

$$\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega + \delta\mathbf{b}_i^\omega) = \Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega)\mathrm{Exp}\left(\frac{\partial\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega)}{\partial\mathbf{b}^\omega}\delta\mathbf{b}_i^\omega\right)$$

$$\therefore \frac{\partial\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega)}{\partial\mathbf{b}^\omega} = \sum_{k=i}^{j-1} \left[-\Delta\tilde{\mathbf{R}}_{k+1,j}^T J_r^k \Delta t\right] \tag{A.31}$$

Using almost exactly the same derivation as used for $\delta\boldsymbol{\phi}_{ij}$ in Section A.2, and replacing j with $k+1$ then,

$$\frac{\partial\Delta\tilde{\mathbf{R}}_{i,k+1}(\mathbf{b}_i^\omega)}{\partial\mathbf{b}^\omega} = \Delta\tilde{\mathbf{R}}_{k,k+1}^T \frac{\partial\Delta\tilde{\mathbf{R}}_{i,k}(\mathbf{b}_i^\omega)}{\partial\mathbf{b}^\omega} - J_r^k \Delta t \tag{A.32}$$

where the initial value $\frac{\partial\Delta\tilde{\mathbf{R}}_{i,i}(\mathbf{b}_i^\omega)}{\partial\mathbf{b}^\omega} = 0$. This equation can be used to calculate a updated $\Delta\tilde{\mathbf{R}}_{ij}$ when $\mathbf{b}_i^\omega$ changes.

Similarly, for the position, a small change in bias is applied to see the effect on $\Delta\tilde{\mathbf{p}}_{ij}$,

$$\Delta\tilde{\mathbf{p}}_{ij}(\mathbf{b}_i + \delta\mathbf{b}_i) = \sum_{k=i}^{j-1} \left[\Delta\tilde{\mathbf{R}}_{ik}(\mathbf{b}_i^\omega + \delta\mathbf{b}_i^\omega)(\tilde{\mathbf{v}}_k - \mathbf{b}_i^v - \delta\mathbf{b}_i^v)\Delta t\right]$$

Substituting in the rotational component from Equation (A.30),

$$\Delta\tilde{\mathbf{p}}_{ij}(\mathbf{b}_i + \delta\mathbf{b}_i) \approx \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(\mathbf{b}_i^{\omega})\mathrm{Exp}\left(\frac{\partial\Delta\tilde{\mathbf{R}}_{ik}(\mathbf{b}_i^{\omega})}{\partial\mathbf{b}^{\omega}}\delta\mathbf{b}_i^{\omega}\right)(\tilde{\mathbf{v}}_k - \mathbf{b}_i^{v} - \delta\mathbf{b}_i^{v})\Delta t\right]$$

Using the small angle approximation ([20] Eq. (4)), the skew symmetric matrix property ([20] Eq. (2)), and removing higher order terms,

$$\begin{aligned}
&\approx \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(\mathbf{b}_i^{\omega})\left(I + \left(\frac{\partial\Delta\tilde{\mathbf{R}}_{ik}(\mathbf{b}_i^{\omega})}{\partial\mathbf{b}^{\omega}}\delta\mathbf{b}_i^{\omega}\right)^{\wedge}\right)(\tilde{\mathbf{v}}_k - \mathbf{b}_i^{v} - \delta\mathbf{b}_i^{v})\Delta t\right] \\
&\approx \Delta\tilde{\mathbf{p}}_{ij}(b_i) - \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}\Delta t\right]\delta\mathbf{b}_i^{v} - \sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^{v})^{\wedge}\frac{\partial\Delta\tilde{\mathbf{R}}_{ik}(\mathbf{b}_i^{\omega})}{\partial\mathbf{b}^{\omega}}\Delta t\right]\delta\mathbf{b}_i^{\omega}
\end{aligned}$$

Therefore,

$$\Delta\tilde{\mathbf{p}}_{ij}(\mathbf{b}_i + \delta\mathbf{b}_i) \approx \Delta\tilde{\mathbf{p}}_{ij}(b_i) + \frac{\partial\Delta\tilde{\mathbf{p}}_{ij}(b_i)}{\partial\mathbf{b}^{v}}\delta\mathbf{b}_i^{v} + \frac{\partial\Delta\tilde{\mathbf{p}}_{ij}(b_i)}{\partial\mathbf{b}^{\omega}}\delta\mathbf{b}_i^{\omega} \tag{A.33}$$

$$\frac{\partial\Delta\tilde{\mathbf{p}}_{ij}(b_i)}{\partial\mathbf{b}^{\omega}} = -\sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^{v})^{\wedge}\frac{\partial\Delta\tilde{\mathbf{R}}_{ik}(\mathbf{b}_i^{\omega})}{\partial\mathbf{b}^{\omega}}\Delta t\right] \tag{A.34}$$

$$\frac{\partial\Delta\tilde{\mathbf{p}}_{ij}(b_i)}{\partial\mathbf{b}^{v}} = -\sum_{k=i}^{j-1}\left[\Delta\tilde{\mathbf{R}}_{ik}\Delta t\right] \tag{A.35}$$

These derivatives can be precomputed and used to calculate $\Delta\tilde{\mathbf{p}}_{ij}$ for new values of $\mathbf{b}_i^{\omega}$ and $\mathbf{b}_i^{v}$. The above equations can be trivially converted into matrix form for iterative calculation,

$$\begin{aligned}
\begin{bmatrix} \frac{\partial\Delta\tilde{\mathbf{R}}_{ik+1}}{\partial\mathbf{b}^{\omega}} & \frac{\partial\Delta\tilde{\mathbf{R}}_{ik+1}}{\partial\mathbf{b}^{v}} \\ \frac{\partial\Delta\tilde{\mathbf{p}}_{ik+1}}{\partial\mathbf{b}^{\omega}} & \frac{\partial\Delta\tilde{\mathbf{p}}_{ik+1}}{\partial\mathbf{b}^{v}} \end{bmatrix} &= \begin{bmatrix} \Delta\tilde{\mathbf{R}}_{k,k+1}^{T} & 0 \\ -\Delta\tilde{\mathbf{R}}_{ik}(\tilde{\mathbf{v}}_k - \mathbf{b}_i^{v})^{\wedge}\Delta t & I \end{bmatrix}\begin{bmatrix} \frac{\partial\Delta\tilde{\mathbf{R}}_{ik}}{\partial\mathbf{b}^{\omega}} & \frac{\partial\Delta\tilde{\mathbf{R}}_{ik}}{\partial\mathbf{b}^{v}} \\ \frac{\partial\Delta\tilde{\mathbf{p}}_{ik}}{\partial\mathbf{b}^{\omega}} & \frac{\partial\Delta\tilde{\mathbf{p}}_{ik}}{\partial\mathbf{b}^{v}} \end{bmatrix} \\
&\quad + \begin{bmatrix} -J_r^k\Delta t & 0 \\ 0 & -\Delta\tilde{\mathbf{R}}_{ik}\Delta t \end{bmatrix}
\end{aligned} \tag{A.36}$$

$$J_{k+1} = AJ_k + B \tag{A.37}$$

A.4 Residual and Jacobians

The complete residual is defined as,

$$\mathbf{r} = \begin{bmatrix} \mathbf{r}_{\Delta\mathbf{R}_{ij}} \\ \mathbf{r}_{\Delta\mathbf{p}_{ij}} \\ \mathbf{r}_{\Delta\mathbf{b}_{ij}^{\omega}} \\ \mathbf{r}_{\Delta\mathbf{b}_{ij}^{v}} \end{bmatrix} = \begin{bmatrix} \mathrm{Log}\left(\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^{\omega})^{T}\mathbf{R}_i^{T}\mathbf{R}_j\right) \\ \mathbf{R}_i^{T}\left(\mathbf{p}_j - \mathbf{p}_i\right) - \Delta\tilde{\mathbf{p}}_{ij}(\mathbf{b}_i^{\omega}, \mathbf{b}_i^{v}) \\ \mathbf{b}_j^{\omega} - \mathbf{b}_i^{\omega} \\ \mathbf{b}_j^{v} - \mathbf{b}_i^{v} \end{bmatrix} \tag{A.38}$$

where expressions $\Delta\tilde{\mathbf{R}}_{ij}(\mathbf{b}_i^\omega)$ and $\Delta\tilde{\mathbf{p}}_{ij}(\mathbf{b}_i^\omega, \mathbf{b}_i^v)$ can be simply computed using the first order approximations derived in the previous section. Note that these equations assume the bias terms evolve in a Gaussian random walk. The speed of bias change can be tuned by varying the bias noise covariance. These equations simplify to a relative pose constraint if $\mathbf{b} = 0$.

Following Equation (A.38), the Jacobian is defined as (using the derivation from ([20] Eq. (28) – (32)),

$$\nabla\mathbf{r} = \begin{array}{c} \begin{matrix} \mathbf{R}_i & \mathbf{R}_j & \mathbf{p}_i & \mathbf{p}_j & \mathbf{b}_i^v & \mathbf{b}_j^v & \mathbf{b}_i^\omega & \mathbf{b}_j^\omega \end{matrix} \\ \begin{pmatrix} -\alpha\mathbf{R}_j^T\mathbf{R}_i & \alpha & 0 & 0 & \beta & 0 & 0 & 0 \\ (\mathbf{R}_i^T(\mathbf{p}_j - \mathbf{p}_i))^\wedge & 0 & -\mathbf{R}_i^T & \mathbf{R}_i^T & -\frac{\partial\Delta\tilde{\mathbf{p}}_{ij}}{\partial\mathbf{b}^\omega} & 0 & -\frac{\partial\Delta\tilde{\mathbf{p}}_{ij}}{\partial\mathbf{b}^v} & 0 \\ 0 & 0 & 0 & 0 & -I & I & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -I & I \end{pmatrix} \end{array} \begin{matrix} \mathbf{r}_{\Delta\mathbf{R}_{ij}} \\ \mathbf{r}_{\Delta\mathbf{p}_{ij}} \\ \mathbf{r}_{\Delta\mathbf{b}_{ij}^\omega} \\ \mathbf{r}_{\Delta\mathbf{b}_{ij}^v} \end{matrix}$$

where,

$$\alpha = J_r^{-1}(\mathbf{r}_{\Delta\mathbf{R}_{ij}}) \tag{A.39}$$

$$\beta = -\alpha\mathrm{Exp}\left(r_{\Delta\mathbf{R}_{ij}}\right)^T J_r\left(\frac{\partial\Delta\mathbf{R}_{ij}}{\partial\mathbf{b}^\omega}(\mathbf{b}_i^\omega - \hat{\mathbf{b}_i^\omega})\right)\frac{\partial\Delta\mathbf{R}_{ij}}{\partial\mathbf{b}^\omega} \tag{A.40}$$

$$J_r^{-1}(x) = I + \frac{x^\wedge}{2} + \left(\frac{1}{\|x\|^2} - \frac{1+\cos(\|x\|)}{2\|x\|\sin(\|x\|)}\right)(x^\wedge)^2 \tag{A.41}$$

and $\hat{\mathbf{b}_i^\omega}$ refers to the bias value used for the original preintegration.

A.5 Summary

The preintegrated twist factor is an efficient method to calculate the kinematic bias online in a factor graph framework. In practice, at preintegration time, every time a new measurement is received, the pose, noise, and partial derivative terms are iteratively updated. This means the optimiser only needs to compute the residual and Jacobians at runtime.

Appendix B

Additional Articles

B.1 Towards Autonomous Inspection of Concrete Deterioration in Sewers with Legged Robots

This article was published in the Journal of Field Robotics in May 2020 [16]. A video summary of this publication can be found at: `https://youtu.be/oZRb4s1UrFM`.

Towards Autonomous Inspection of Concrete Deterioration in Sewers with Legged Robots

Hendrik Kolvenbach[1], David Wisth[2], Russell Buchanan[2], Giorgio Valsecchi[1], Ruben Grandia[1], Maurice Fallon[2], Marco Hutter[1]

[1] Robotic Systems Lab, ETH Zurich
Leonhardstr. 21
8092 Zurich, Switzerland
(hendrikk, vgiorgio, rgrandia, mahutter)@ethz.ch

[2] Oxford Robotics Institute, University of Oxford
17 Parks Road
Oxford OX1 3PJ, United Kingdom
(davidw, russell, mfallon)@robots.ox.ac.uk

Abstract

The regular inspection of sewer systems is essential to assess the level of degradation and to plan maintenance work. Currently, human inspectors must walk through sewers and use their sense of touch to inspect the roughness of the floor and check for cracks. The sense of touch is used since the floor is often covered by (waste) water and biofilm, which renders visual inspection very challenging. In this paper, we demonstrate a robotic inspection system which evaluates concrete deterioration using tactile interaction. We deployed the quadruped robot ANYmal in the sewers of Zurich and commanded it using shared autonomy for several such missions. The inspection itself is realized via a well-defined scratching motion using one of the limbs on the sewer floor. Inertial and force/torque sensors embedded within specially designed feet captured the resulting vibrations. A pre-trained support vector machine is evaluated to assess the state of the concrete. The results of the classification are then displayed in a 3D map recorded by the robot for easy visualization and assessment. To train the SVM we recorded 625 samples with ground truth labels provided by professional sewer inspectors. We make this dataset publicly available. We achieved deterioration level estimates within three classes of more than 92% accuracy. During the four deployment missions, we covered a total distance of 300 m and acquired 130 inspection samples.

1 Introduction

The Swiss sewage systems have a total length of over 130,000 km and represent a significant communal investment. Maintaining the system is crucial for public health, but also imposes a considerable cost upon the various municipalities. Reports from industrialized countries indicate that most sewage systems have reached half of the average expected lifetime of 80 years [Berger and Falk, 2009]. Thus, the focus is shifting towards maintenance and renovation. As a first step towards maintenance, cost-effective and accurate inspection is essential to assess the state of the sewer system. Sewers have been built successively, with parts of the system being over 100 years old. A large variety materials, components, and standards have been used across this time.

Categorization can be made between medium and large sewers, with an inner diameter of more than 800 mm and small sewers, with a diameter down to 100–150mm. Large sewers are typically made of concrete or masonry and account for around 10% of the network [Dyk and Lohaus, 1997].

Small sewers are typically made of concrete, stoneware or plastics and account for the remaining 90%. Concrete is used in almost 40% of all sewers [Berger et al., 2016]. Although the vast majority of small sewers are of a circular or oval shape, most of the larger ones are jaw-shaped, rectangular or of irregular shape.

The rate of deterioration varies and depends on the design and usage of the sewers. Relevant factors are flow rate, slope, wastewater composition, cleaning intervals, and more [Parande et al., 2006]. An omnipresent disintegration mechanism acting on concrete in sewers is microbial induced corrosion [Hudon et al., 2011]. This type of corrosion occurs when sulfate-reducing bacteria found in the biofilm produce hydrogen sulfide from wastewater, which is absorbed on the moist surfaces of the sewers and creates sulfuric acids. These acids react with the alkaline minerals in the concrete, which leads to the creation of large, expansive minerals and ultimately the loss of structural integrity [Wells and Melchers, 2015].

Current inspection approaches vary depending on the diameter, material, shape and expected damage to the sewer. For example, small pipes are more prone to experience clogging or leaking than large sewers. Thus, visual inspection is performed by tethered pipe-inspection robots, which crawl through the sewer and sometimes carry tools for removing blockages. Many types of small-scale robots are now commercially available [Mirats Tur and Garthwaite, 2010].

Up to now, medium to large sewers are inspected by humans with the goal of manually assessing the deterioration level. Inspectors check the roughness of the concrete visually and tactilely with their hands and feet. As the highest deterioration occurs in the center of the sewer, which is often covered by a biofilm and wastewater, purely visual assessment fails to predict the deterioration reliably. Humans are also able to adapt to irregular sewer shapes and can move through pipe diameters down to 800 mm. However, inspection tasks in typical environmental conditions (slippery ground, flowing water, dirty, damp, occasionally narrow spaces) are monotonous, dangerous and carry health-risks [Berger et al., 2016]. Additionally, extensive safety precautions are necessary such as gas sensors and safety harnesses. It is common to close off large areas of a sewage system before inspections, which potentially disrupts the network. Overall, sewer inspection presents an excellent opportunity for versatile service robots.

Only a few inspection robots for medium to large diameter sewers have been developed. *SVM-RS* from the Fraunhofer Institute for Factory Operation and Automation is a combined cleaning and inspection robot [Walter et al., 2012]. The robot weighs 2000 kg, has a size of 3500 x 1500 x 1500 mm (L x W x H) and a reach of 1200 m. Its payload includes cameras, ultrasonic sensors, structured light line scanners, and temperature sensors. Redzone Robotics' *Responder*[1] is a 300 kg, tracked platform which can be deployed in sewers with a diameter from 915-6000 mm and has a reach of up to 1000 m. The robot is equipped with cameras, laser sensors and ultrasonic sensors. A robot developed by Nanyang Technological University can be passed through tunnels with a minimum diameter of 3000 mm and has explored distances of up to 400 m [Seet et al., 2018]. The robot uses cameras and a laser profiler.

The typical method for detecting concrete damage with a robot is through an operator interpreting the acquired camera images and ultrasonic or laser sensor data. All robots are use a power cable to increase their operational range. For some of the systems, deployment through a common utility hole would be difficult. An interesting robot in this context is Pure Technology's *SmartBall*[2], which is a free-swimming robot that can be deployed in a water stream and scans the sewer with ultrasonic sensors to find leaks. However, there is no means of controlling its path and depending on the sewer shape the deployment and recovery might be difficult.

We propose the usage of autonomous legged robots to inspect and map large and medium-sized sewers. Legged robots are relatively small, have high mobility in complex, human-made environments and can adapt their posture to inspect areas of interest [Hutter et al., 2017a] [Bellicoso et al., 2018].

[1] Available at: https://www.redzone.com/technology/responder, Accessed: 2020-03-25

[2] Available at: https://puretechltd.com/technology/smartball-leak-detection/, Accessed: 2020-03-25

Similar to humans, legged robots can probe their environment tactilely by using their limbs [Hoepflinger et al., 2010] [Kolvenbach et al., 2019]. We have successfully deployed *ANYmal* [Hutter et al., 2017b], an autonomous quadruped robot, in the sewers of Zurich as shown in previous work [Kolvenbach et al., 2019]. With the help of specially designed sensor-equipped feet, we collected a large dataset by performing an inspection motion with one limb. Later on, we trained a model to assess the level of concrete deterioration using supervised machine learning techniques. In this work, we show how the trained model is used to assess the state of the concrete while performing subsequent inspection missions in the sewer[3]. During the missions, the robot was commanded by an operator outside the sewer using a shared autonomy framework. With this, the robot would map its environment, walk, and perform the inspections while the operator sets the next waypoints and actions, respectively. A 3D map of the sewer including the inspection location and state of the concrete are the outcome of the missions. The map can be used by humans to return for repair work or monitor the deterioration rate over time.

This paper is structured as follows. First, we describe the robot with a focus on the foot design in Sec. 2. Then we present the robot deployment in the sewer (Sec. 3.1), the tactile inspection motion (Sec. 3.2), the creation of the dataset (Sec. 3.3), the classification approach (Sec. 3.4) and the mission planning system (Sec. 3.5). The outcome of the inspection missions are presented and discussed in Sec. 4. Finally, we conclude the work in Sec. 5.

2 Hardware description

2.1 ANYmal

ANYmal (Figure 1) is a 30 kg quadruped robot driven by twelve series elastic actuators mounted at the joints. The dimensions of the robot are 800 mm x 600 mm x 700 mm when standing. With its legs tucked up, it is 800 mm x 600 mm x 400 mm which allows it to be deployed through a common utility hole. The kinematic structure of the robot is designed to achieve an extensive range of motion, allowing it to overcome obstacles and manipulate the environment. The structure is made primarily of aluminum and carbon fiber to minimize weight. Shock absorbers and

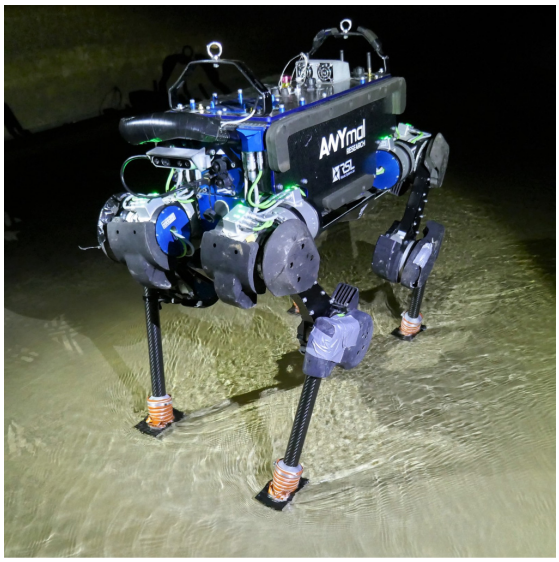


Figure 1: Depiction of *ANYmal* with custom feet deployed in the sewers of Zurich.

[3] https://youtu.be/oZRb4s1UrFM

Kevlar plates protect exposed components from impacts. *ANYmal* is designed in a modular fashion: each actuator unit consists of a motor, gears, and drive electronics and is connected over the EtherCAT bus. The connection between sub-assemblies such as the legs are easily accessible, and water-proof quick connectors are used for power and communication lines. This allows component-level ingress protection as well as fast and simple maintenance in case of hardware failure.

For localization and navigation, the machine is equipped with a LIDAR (Velodyne VLP16) sensor mounted on top of the torso. A front-mounted stereo depth camera (Intel RealSense D435) provides a local map of the terrain to assess traversability and assist foothold planning. Additional sensors modules, such as RGB and thermal cameras, can be mounted for inspection tasks. Powerful LEDs provide sufficient illumination in dark environments. *ANYmal* can operate partially or fully autonomously with on-board batteries. The battery allows the system to traverse up to 3.6 km with a trotting gait at 0.5 m/s on a single charge. Optionally, the robot can be recharged without human interaction via a docking station in case long-term autonomy is required [Kolvenbach and Hutter, 2018].

2.2 Sensor-equipped, adaptive feet

We designed sensor-equipped, adaptive feet to enhance locomotion on the rough and slippery terrains encountered in sewers while measuring local ground inclination and surface properties. Similar to the rest of the robot, the feet have to be sufficiently robust to operate continuously in a challenging environment. The design is based on the adaptive foot proposed in previous work of our group [Käslin et al., 2018]. The foot consists of a large flat contact surface that can comply to the local ground inclination without interfering with the locomotion of the robot. With a possible inclination of the terrain of up to 25°, the range of motion for ground compliance is set to 50° around the pitch axis and 30° around the roll axis. Since each leg only allows for hip abduction/adduction, hip flexion/extension and knee flexion/extension, the foot compliance around yaw prevents slipping while turning. Weighing 314 g (including cabling and connectors), it is lighter than both the original point foot and the previous adaptive foot. Figure 2a illustrates the sub-assemblies of the foot, which are described in the following.

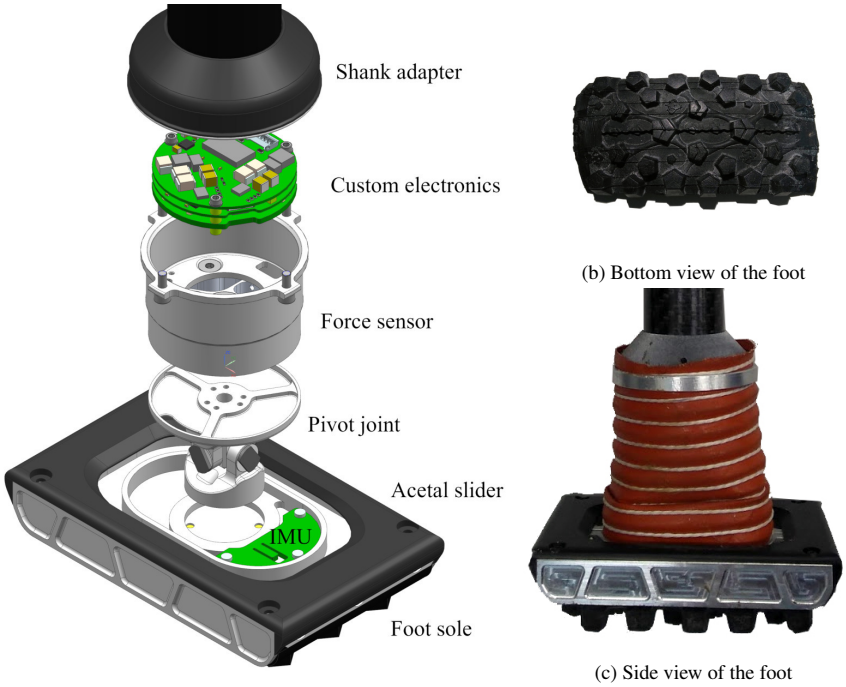
1) Foot sole: The sole has a surface area of 60 cm^2 (100 mm x 60 mm) and is made from an off-road rubber tire featuring 5 mm studs for increased traction (Figure 2b). The sole is connected to a metal rim by clamping, which avoids peeling and gluing issues. Damping foam is placed between the rubber sole and the metal structure to reduce the peak loads resulting from impact forces during walking. An acetal slider avoids the foot getting stuck on overhanging edges and retains the metal rim.

2) Pivot joint: The pivot joint features a lightweight universal joint with integrated end stops to provide the required ground compliance. It is surrounded by an Ester Polyurethane rubber tube of Shore A70 that provides the retaining force to reset the foot to its initial position after deflection.

3) Force sensor: A custom, in-house developed six-axis force/torque sensor is placed above the pivot joint to measure the forces acting on the foot. It consists of a force sensing element with strain gauges. The sensor is lightweight and robust and allows sensing up to 1000 N along the z-axis and 400 N along the x and y-axis. The maximum torque the sensor can sense is specified as 10 Nm. The accuracy lies within 1.5% of the measured value while the repeatability lies below 0.05% [4]. The measurements are filtered at 800 Hz by an integrated sinc filter with cut-off frequency of 1255 Hz. The sensor is temperature compensated to minimize drift during operation.

4) Custom electronics: The electronics of the foot consists of two IMUs (MPU-9250), a force sensor and a microcontroller board. One of the IMUs is located in the sole, while the other is integrated with the PCB in the shank. The IMU's linear acceleration is filtered with a cut-off frequency of 460 Hz while the angular velocity is filtered with a cut-off frequency of 184 Hz. Both the IMUs and the force sensor are connected to the microcontroller via the serial peripheral interface bus (SPI). The IMUs are read out at 1 kHz and force measurements are obtained at 400 Hz. The microcontroller board is connected to the robot via EtherCAT and powered through the auxiliary 12 V power line. A PCB with analog-to-digital converters and a microcontroller processes analog signals coming from strain gauges to compute the forces and torques. Sensor data is recorded on the high-level side at 400 Hz.

[4] Available at: https://www.botasystems.com/, Accessed: 2020-03-25



(a) Sub-assemblies of the foot

(b) Bottom view of the foot

(c) Side view of the foot

Figure 2: Overview of the newly developed foot for sewer inspection.

5) Shank: The carbon fiber shank connects the foot to the knee of the robot. The shank is sealed and features a conical slider for protection of the force sensor.

6) Sealing: The joint is protected by thick bellows (visible in Figure 2c), mechanically clamped to the structure and sealed, which improves the ingress protection rating compared to previous work. O-rings and sealants have been used for all the matching surfaces. Water-proof cable glands and connector have been used for the cables.

3 Sewer inspection with quadruped robots

3.1 Deployment

We conducted multiple field test campaigns in the sewage system of Zurich to iterate the hardware, practice operations, collect datasets, and execute inspection missions. The first test campaign with the robot took place during two days in February 2019 [Kolvenbach et al., 2019] and the second campaign took place on three days in December 2019. Generally, the robot can be easily deployed through a common utility hole (d = 0.8 m). To do so, the robot's legs are unpowered, tucked up, and the system is lowered into the sewer with a tether (Figure 3). During our tests, the robot was operated from a base station which was located outside the sewer, nearby the utility hole. The base station consists of the operator PC, an additional screen, and outlets for electricity and communication links to the robot. The base station is easy to transport and quick to set up. We installed a gazebo to shelter the robot operator and desk during rainy testing days. We deployed directional Wi-Fi antennas (Ubiquiti airMAX, HyperLink HG4958DP-19P) into the utility hole via a tripod with a reversed column to enable a reliable communication link to the robot. During the test, a safety operator stayed in the sewers with a professional inspector and communicated with the robot operator outside via two-way radios (Figure 4).



Figure 3: Depiction of the field test setup and the deployment of *ANYmal* into the sewer. The operator base station and warnings signs are visible in the background.

The sewer inspector had to always accompany the robot for safety reasons. Potential risks include rapid changes in water levels during heavy rainfall and decreased oxygen concentration. The sewers were humid and relatively warm compared to the outside temperatures during our trials because warm wastewater was flowing in the sewer. In the morning, fog was observed, which degraded visibility. Additionally, water levels would vary during the day by a few centimeters as rainfall occurred in the surrounding areas.

3.2 Tactile inspection motion

The scratching motion we use to inspect the floor must be repeatable and reliable across the entire range of possible surface areas. The motion, therefore, needs to be specified in such a way that it can adapt to local terrain geometry and surface roughness.

We implemented a Cartesian-space impedance controller [Khatib, 1987], which allows a motion design on both force and position level. Specifying and executing these motions was done by extending the Free Gait framework [Fankhauser et al., 2016].

The full sequence of the inspection motion, shown in Figure 5, can be split into several phases. A predefined position relative to the three stance legs is approached in (a) and contact is established in (b). In (c), a straight line trajectory is followed until a target location (d). In (e) and (f), the foot is re-positioned to return to a nominal stance on four feet. For the part (c) of the inspection motion, where data is collected, the desired end-effector force is computed as seen in the following equation.

$$\boldsymbol{f}_{des} = \boldsymbol{f}_{ref} + \boldsymbol{\Lambda}(\boldsymbol{q})\ddot{\boldsymbol{x}}_{ref} + \boldsymbol{K}_p(\boldsymbol{x} - \boldsymbol{x}_{ref}) + \boldsymbol{K}_d(\dot{\boldsymbol{x}} - \dot{\boldsymbol{x}}_{ref}) + c_\mu \frac{\dot{\boldsymbol{x}}_{ref}}{\|\dot{\boldsymbol{x}}_{ref}\|} |\boldsymbol{n}^T \boldsymbol{f}_{ref}|, \tag{1}$$

where $\boldsymbol{f}_{ref}$ is the designed Cartesian end-effector force reference, and $\boldsymbol{x}_{ref}, \dot{\boldsymbol{x}}_{ref}, \ddot{\boldsymbol{x}}_{ref}$ are the designed position, velocity and acceleration references. $\boldsymbol{\Lambda}(\boldsymbol{q})$ is the reflected inertia matrix, which depends on the generalized coordinates $\boldsymbol{q}$. $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$ are the position and velocity gain matrices. Finally, c_μ is a scalar value used to provide a feedforward friction compensation in the direction of the motion, scaled by the reference force along the surface normal, $\boldsymbol{n}$.



(a) *ANYmal* setup in the sewer

(b) Walking on biofilm

(c) Walking through wastewater

Figure 4: Depictions of *ANYmal* walking in the sewer. A professional sewer inspector accompanies the testing and assesses the state of the concrete.

For the main part of the inspection motion, we apply a force of either 10 or 15 N on the surface. The force was experimentally tuned to keep the feet on the ground while allowing for a smooth scratching motion. The target location is set 100 mm forward and 50 mm sideways from the start location. Together with a total duration of 2 s and an initial and final velocity of zero, a quintic spline interpolation is defined between the start and end location to generate the motion reference. Impedance gains are set with stiffness $\boldsymbol{K}_p = \mathrm{diag}(200, 200, 0)$ in $\mathrm{N/m}$, and damping $\boldsymbol{K}_d = \mathrm{diag}(20, 20, 20)$ in $\mathrm{Ns/m}$.

Friction compensation c_μ was set to 1.0 after experimental tuning in the field. We found the high friction compensation to be important for successful motion execution on the rougher surfaces. The value of 1.0 served as a safe upper bound for the roughest terrain encountered. For more slippery surfaces the compensation is too high, but this does not pose a problem as the damping terms quickly regulate the velocity and stabilize the motion.

3.3 Dataset collection

A dataset was collected over different times in two rectangular shaped sewers. Both sewers were accessible through a utility hole and large enough to be traversable by humans. A slight inclination towards the center and towards the direction of flow, resulted in a higher accumulation of water in the center. To avoid over-fitting during classification later on, we moved and reoriented the robot frequently to capture different poses of the robot with respect to the sewer floor. In total, we investigated 20 locations and areas of interest to capture a broad set of surface conditions.

Typically, the condition detection in sewers is captured by filling in evaluation sheets in which deterioration levels are grouped in classes [DIN EN 13508-2:2011, 2011]. Similar scales although more general are found in the inspection of other concrete structures, such as bridges [Everett et al., 2008].

In order to construct the dataset, we defined a scale of five condition ratings for the sewers. The scale was developed with professional sewer inspectors and is inspired by similar standards [Everett et al., 2008].

- **Good:** Smooth concrete, no problems noticeable
- **Satisfactory:** Minor signs of deterioration, increased roughness
- **Fair:** Medium signs of deterioration, increased roughness and scratches/spalling
- **Critical:** Major deterioration noticeable, large cracks, imminent failure
- **Failure:** Loss of structural integrity, leakage

The condition of the concrete we encountered in the sewers ranged from *good* to *fair*, while *critical* or extremely bad structural *failures* were not encountered. In total, we were able to collect a total of 625 samples during the two field test campaigns (*good*: 183 samples, *satisfactory*: 183 samples, *fair*: 259 samples). The samples were taken in different parts of the sewers, and were classified together with a professional sewer inspector who provided the ground truth. The dataset, named *STINK* (Sewer Terrain Inspection Knowledge), is openly available[5].

3.4 Classifying concrete deterioration

We chose a machine learning approach to capture and classify the diverse appearance of concrete deterioration together with the varying environmental conditions. As mentioned, the surface condition is not only expressed by the roughness of the concrete but also by macroscopic features such as holes, scratches, or cracks. At the same time, the surface can be dry, wet, submerged and/or covered by a biofilm (Figure 6). We extracted the raw sensor data acquired while performing the inspection motion and normalized the signals. No further cleanup or filtering was performed in post-processing.

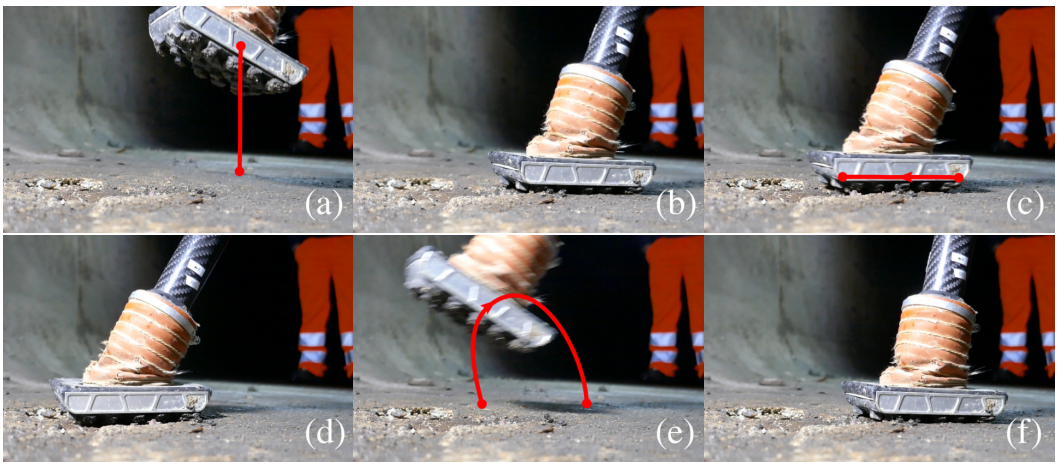


Figure 5: Picture sequence of the tactile inspection motion with the foot placement phase (a), inspection motion start location (b), main inspection phase (c), motion target (d), re-positioning movement (e), and final position (f).

[5]DOI: 10.3929/ethz-b-000336822

The acquired data consists of 18 signals (Force/Torque, IMU shank (linear acceleration/angular velocity), IMU sole (linear acceleration/angular velocity)). The data is cropped to 2 seconds, starting at the point of the scratching motion (Figure 5a). The raw sensor signals show a correlation between signal magnitudes and concrete deterioration level, which we exploited in the following for classification (Figure 7). The cropped sensor signals are standardized using z-scores to obtain zero mean and unit standard deviation. The dataset itself is split between randomly chosen 75% of the data for the training set (468 Samples) and 25% for the validation set (157 Samples).

Next, we decomposed the sensor signals using a fast Fourier transform (FFT) and selected the magnitude of the frequency components as features. Last, we reduced the dimensionality of the training set with a principal component analysis (PCA), which explained at least 95% of the variability. This PCA transformation of the training set was then applied to the validation set.

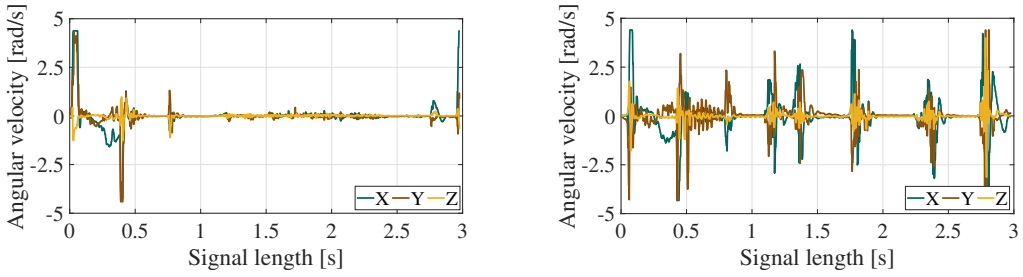
We utilized a support vector machine (SVM) with linear Kernel (LIBSVM for Matlab [Chang and Lin, 2011]) to train the model. The linear Kernel was chosen since it resulted in high overall accuracy. More complex, non-linear, or radial basis Kernels were omitted since they increase the chance of over-fitting while only supplying a marginal increase in performance.

We performed training on the data using five-fold cross-validation combined with a grid search to find an optimal C-setting. The multi-class problem is solved by training three binary one-vs-all classifiers. The accuracy was determined by predicting the degradation level of the validation set, which was left out during training.



(a) good state, water (b) satisfactory state, biofilm (c) fair state, wet

Figure 6: An exemplary set of pictures illustrating the various surface conditions encountered in the sewers.



(a) Concrete in good condition (b) Concrete in fair condition

Figure 7: Comparing the angular velocities of two samples recorded by the foot sole IMU show a correlation between signal magnitudes and concrete deterioration (Complete sequence of the inspection motion).

Table 1: Classification accuracy related to selected sensor signals (average performance over ten evaluations).

	Features	**Precision, Recall [%]**			**Accuracy**
Sensor Signal	[#]	**Fair**	**Satisfactory**	**Good**	**[%]**
Force	144	62.9, 73.3	68.2, 56.1	75.3, 69.8	**67.0**
Torque	171	67.5, 79.0	71.0, 64.7	77.3, 65.8	**70.8**
Force + Torque	214	70.9, 79.4	78.4, 69.4	79.6, 76.3	**75.4**
IMU Sole (linear acceleration)	297	77.1, 80.2	74.8, 70.8	86.8, 86.1	**79.4**
IMU Sole (angular velocity)	269	81.6, 84.0	83.1, 83.5	91.0, 85.7	**84.5**
IMU Sole	332	87.6, 86.5	83.4, 84.9	92.8, 92.7	**87.9**
IMU Shank (linear acceleration)	246	79.0, 81.7	76.1, 72.3	93.0, 92.3	**82.0**
IMU Shank (angular velocity)	138	72.3, 79.4	73.4, 71.0	88.2, 80.1	**77.1**
IMU Shank	252	85.2, 85.3	83.3, 82.1	94.4, 96.0	**86.9**
IMU Sole + IMU Shank	335	87.8, 90.3	88.8, 86.2	97.0, 96.1	**90.7**
F/T + IMU Sole + IMU Shank	341	91.6, 93.0	91.6, 89.5	95.5, 95.5	**92.6**

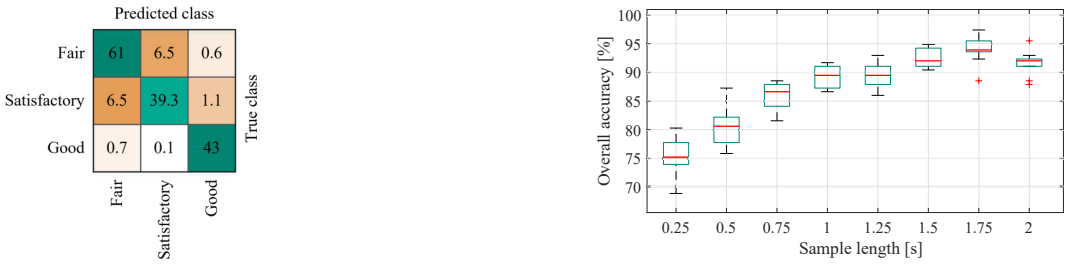
We were able to classify the deterioration levels with an overall classification accuracy of more than 92% and precision & recall of at least 89% on the three assigned classes. Comparing the individual sensor contributions to the classification performance shows that the combined set of sensors provides the highest accuracy (Table 1). The worst performance was found to be on the Force/Torque data, which includes the axis that is actively controlled by the robot. Nevertheless, also a smaller setup consisting of only IMU's achieves a generally good performance. This matches previous findings on the smaller sewer inspection dataset [Kolvenbach et al., 2019] and a dataset used for planetary soil classification [Kolvenbach et al., 2019].

Performing the classification with all sensors resulted in the highest accuracy, and further investigations were performed with this selection. A confusion matrix showing the truly and wrongly classified samples was derived by training and evaluating the model for ten successive trials (Figure 8). Misclassification occurs mostly between the *fair* and *satisfactory* class, which are also fairly similar in terms of roughness compared to the smooth, *good* class. More samples were available for the *fair* class compared to the *satisfactory* and *good* class, which might explain the generally lower classification performance on the *satisfactory* class of the validation set.

Investigating the sampling time shows that the signals should be captured for at least for one second for high classification accuracy. A small drop in performance is seen when taking the full sample length, which might be explained by small deviations in the duration of the scratching motion. Generally, an increase in performance might be realized by increasing the size of the dataset, further optimizing the feet for vibration response, or utilizing alternative classifiers, such as neural networks [Bednarek et al., 2019].

3.5 Inspection mission

We utilized the trained SVM model to perform complete inspection missions in the sewer. The missions were planned and executed using the *Director* user interface (Figure 9) and system architecture, initially designed by MIT during the DARPA Robotics Challenge [Marion et al., 2017]. It features a shared autonomy system where the user can provide high-level commands to the robot, which are then executed autonomously. For the missions, the robot operator simply provides multiple waypoints in the online-generated map for inspection. The robot will then walk to these waypoints and perform the inspection motion autonomously.



(a) Confusion matrix

(b) Accuracy increase over signal duration

Figure 8: Few misclassification events occur between the *Satisfactory* and *Fair* class. Correlation between classification accuracy and signal length indicates that samples should be collected for at least a second after starting the scratching motion (Average performance over ten evaluations).

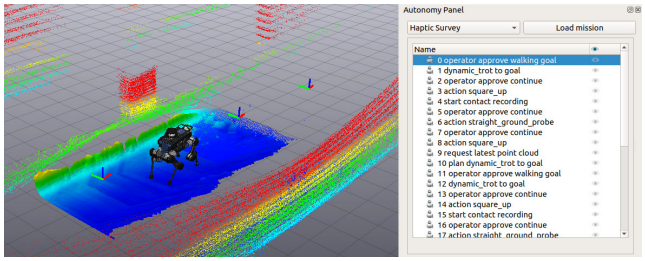


Figure 9: Screenshot of the Director mission interface [Marion et al., 2017], showing three inspection waypoints and the associated task queue.

The robot's odometry is provided by the kinematic-inertial Two-State Implicit Filter [Bloesch et al., 2018]. This is used as a prior to a LIDAR based Iterative Closest Point (ICP) SLAM system for accurate localization and map building. During the missions, the robot also recorded a local elevation map using the forward-mounted depth sensors [Fankhauser et al., 2018]. For each inspection motion, the position of the robot contacts in the map frame, the elevation map of the terrain around the robot, and the timestamp were recorded for post-processing.

The missions consisted of inspection walks between utility holes and tunnel crossing, which is a scenario equally performed by human inspectors. The distances between two utility holes typically range from 100–150m, and crossing between sewers requires the robot to overcome small obstacles such as steps [Fankhauser et al., 2018]. During testing, we chose a set of different mission environments, such as a straight, featureless tunnel, a turn, and a crossing between two sewers to capture this diversity. Inspection motions were performed approximately every 2–3m along the sewer or as commanded by the robot operator. Before each inspection motion, the robot would perform a "square up" maneuver, which would return the robot to a stable stance and consistent start position before performing the inspection.

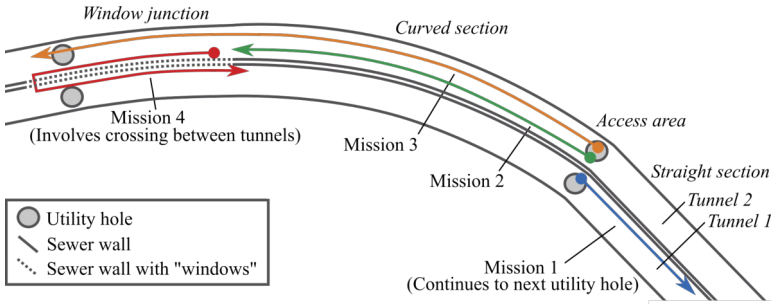
4 Results

4.1 Mission results

Using the shared autonomy framework, we completed four missions as described in Figure 10a and Table 2. For missions 1-3, the robot was deployed through a utility hole and commanded to inspect a section of the tunnel (of length 145 m). The mission ended if either the final goal was reached or the robot slipped and could not recover. A safety operator was present at all times during the missions but did not intervene unless the robot fell. Mission 4 was designed to show the capabilities of using legged robots specifically.

For this mission, the robot was placed near the window relief junction and directed by the robot operator outside the sewer to cross tunnels. A detailed breakdown of each mission is given below:

- **Mission 1**
  The robot started in the first tunnel sewer access area and then walked down a straight section of the tunnel on the left side of the stream. This mission began in the morning when the above surface air was much cooler than the air in the tunnel. This temperature difference created fog in the sewer, as shown in Figure 10b, which was often detected as an obstacle by the LIDAR. The fog and featureless straight tunnel resulted in several artifacts in the LIDAR map (see the highlighted section of the straight tunnel in Figure 11) as well as more drift than in other missions. Since the robot walked from one utility hole to another, this mission imitates a typical inspection excursion for a human. Once the mission was complete, the robot operator directed the robot to cross the stream and walk back to the start.



(a) Top-down diagram of the missions completed in the sewer.



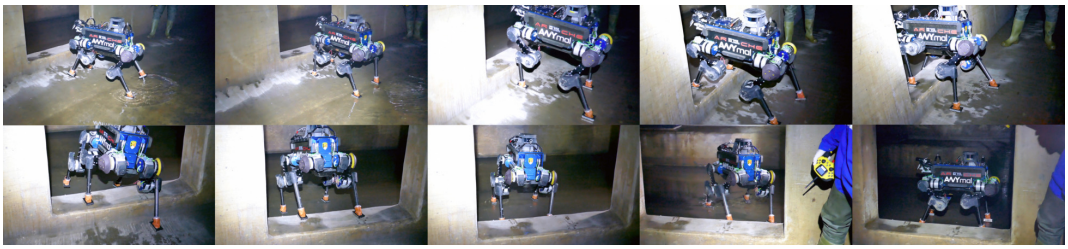
(b) The robot during mission 1 as it walks down the straight featureless tunnel. Fog can be seen in front of the robot.



(c) Mission 2 in the second tunnel. Here the sewer is more curved and there is a large pipe suspended from the ceiling.



(d) View of second tunnel in direction of the window junction during mission 3. Two large windows are visible to the robot's left.



(e) The robot performing the crossing maneuver during mission 4. The robot has climbed onto the threshold and is transiting from tunnel two to tunnel one to continue the inspection.

Figure 10: Photos taken during each robot deployment.

- **Mission 2**
  The robot started in the second tunnel sewer access area and then walked towards the curved section of the tunnel and the window junction. In this tunnel, a pipe was suspended from the ceiling, which, along with the curvature of the tunnel in this section, provided good features for LIDAR SLAM. After reaching the window junction, the robot operator commanded the robot to cross the stream and return to the start. After crossing the stream, the robot slipped and fell into the stream, which required operator intervention to reset the robot.

- **Mission 3**
  The robot started in the second tunnel sewer access area then walked towards the curved section of the tunnel and the window junction, similar to mission 2. As the robot entered the window junction, it slipped on a smooth part of concrete. The robot's controller responded fast enough and automatically froze the joints so that the robot remained standing in place.

- **Mission 4**
  One key motivation to use legged robots for sewer inspection is their versatility. We demonstrated this by placing the robot in the second tunnel and having it walk into the window junction. The robot operator then placed waypoints such that the robot climbed over a 15 cm step from tunnel two into tunnel one and the performed eight inspections. Because a legged robot can transit from one tunnel to another, they are much more useful than a robot confined to a single tunnel per mission.

Table 2: Description of the missions performed in the sewer.

Mission	**Distance** [m]	**No. Inspections**	**Duration** [min]
Mission 1	145	55	31
Mission 2	74	41	25
Mission 3	43	26	13
Mission 4	32	8	10
Total	**294**	**130**	**79**

4.2 ICP Mapping

The 3D point cloud shown in Figure 11 combines the ICP maps generated during each of the four missions, stitched together in post-processing, and overlayed on the sewer construction drawings. It shows that the point cloud reconstruction of the sewer is both internally consistent and matches the scale and geometry of the plans.

The ICP map was most accurate around the window junction due to the strong geometric features and missions 2 and 3 generated almost identical maps in independent runs. The poorest performance was during mission 1 in the long, straight portion of the tunnel. As described in Figure 10, this portion of the tunnel is almost perfectly symmetric along the direction of travel, and there were significant artifacts in the point clouds caused by dense fog, leading to an overestimate of tunnel length by 42.8 %. To analyse the quality of the ICP mapping further we compared it to a ground truth point cloud collected by tripod-based commercial 3D laser scanner (Leica BLK-360). The majority of the points in the ICP map lie within 20 cm of the ground truth as shown in Fig. 12, showing the validity of this approach.

Overall, apart from scale drift in one of the missions, the point cloud is both metrically accurate and consistent with the building plans, allowing a sewer professional to easily diagnose and localize areas of concern within the sewer system.

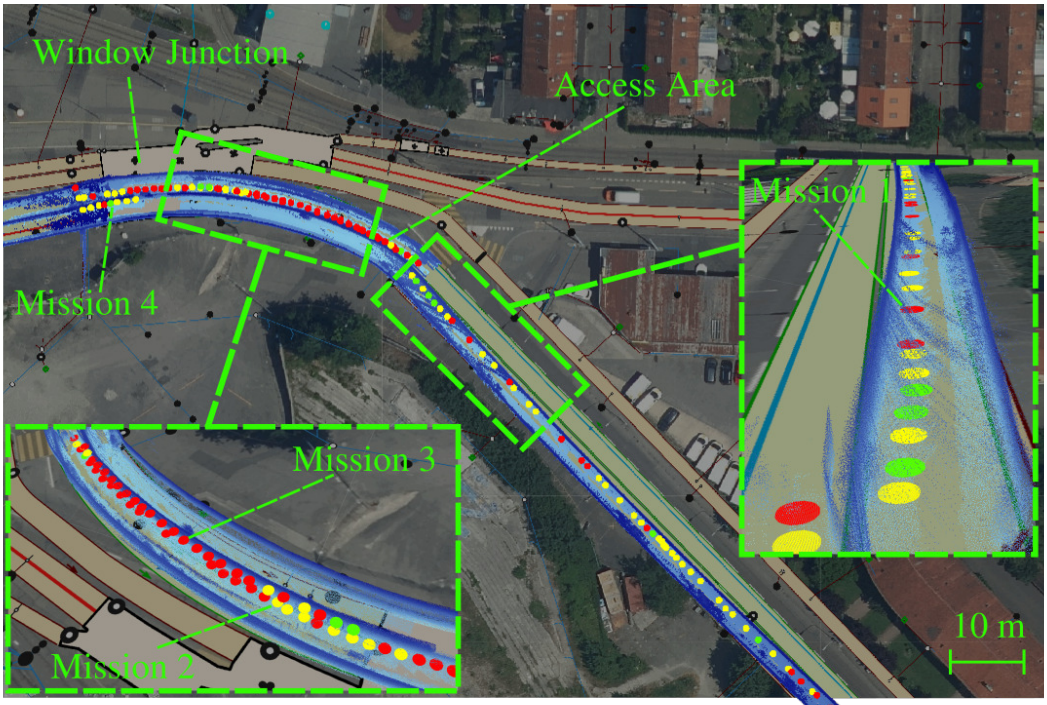


Figure 11: Overview of whole pointcloud map overlaid on sewer plans and satellite image. Blue points represent the map generated by the robot's LIDAR and the color intensity indicates height with dark blue being the sewer floor. Concrete classifications are colorized as: Red (*Fair*), Yellow (*Satisfactory*) and Green (*Good*). In green boxes we highlight two areas of interest: the curved section where missions 2 & 3 occurred and the straight section of mission 1. In the curved section we have raised the mission 3 contact maps above the ground to improve visibility.

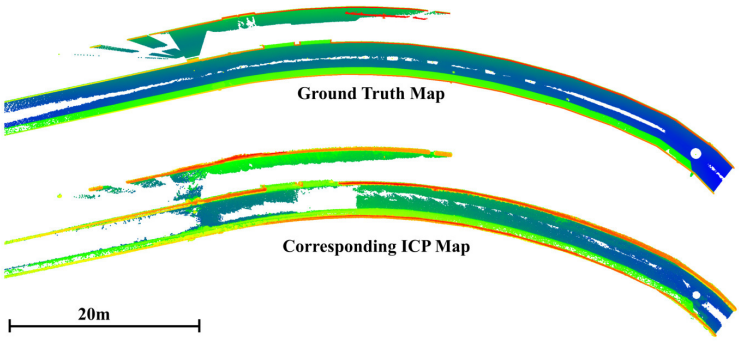


Figure 12: Comparison of the ground truth point cloud and the ICP map generated by the robot. The ICP map shown contains only the points that lie within 20 cm of the ground truth, showing the accuracy of the ICP map. The point clouds are colorized by height for clarity.

4.3 Degradation estimates

The inspection mission results are displayed in Figure 11. In total, we were able to take 130 samples over 294 m to which we assigned a deterioration class in post-processing. There are several sections of the sewer with different levels of concrete degradation, which matched our qualitative impression. For example, the worst conditions were found in the second tunnel between the access area and window junction, while the straight part of the first tunnel was less degraded.

The robot generally inspected areas close to the center of the sewer where smooth terrain is not likely to be found. Mission 2 and mission 3 covered the same area in the sewer, and the classification results reflect this. Areas that have been investigated twice show a high level of consistency in the degradation estimate. Out of 45 samples that have been collected over the same 43 m section of sewer, just five samples show a difference in their classification result. Of the five outliers, four showed a deviation by one class and one sample by two classes. Due to the small investigated footprint of 0.005 m^2 during the inspection, it might be that the areas are not necessarily misclassified but are rather capturing a general change in deterioration of the concrete. The high spatial resolution of the measurements combined with the location in the map make it easy to visualise clusters of good or worse concrete deterioration. Generally, several inspection walks over the same areas have increased the statistical power of this approach. A possible extension of this work would be to perform the evaluation online and increase the amount of samples taken in an area with noticeable change in degradation estimates.

An example of the raw data for contact classifications is provided in Figure 13, showing the vertical ground reaction force over three inspection motions. There is a clear pattern to the data, making it easy to distinguish between the three phases of the inspection task – walking, "square up", and inspection motion. This shows the repeatability of the robot inspection, even during complex real-world missions.

Overall, the total mission consisted of 130 inspections of which there were a total of 9 *Good* classifications, 59 *Satisfactory*, and 62 *Fair* which were colorized green, yellow and red respectively in the contact maps.

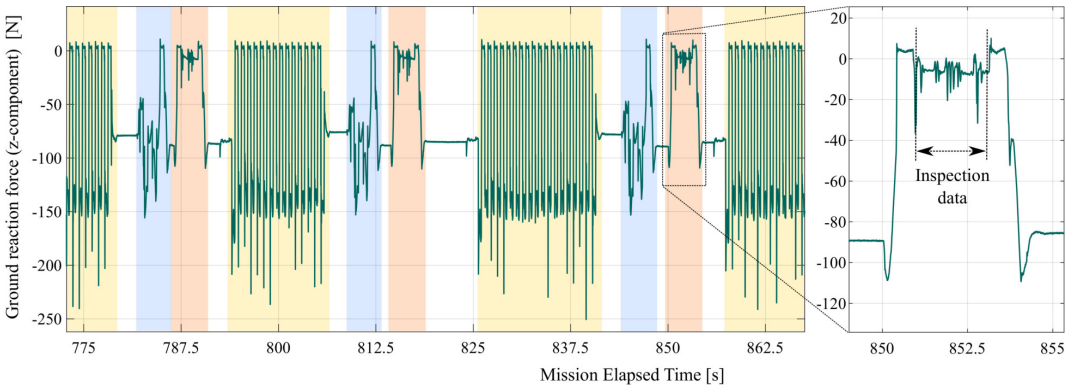


Figure 13: The vertical component of the ground reaction force of the front right foot as measured during mission 2. Walking episodes (yellow) can be easily distinguished from the "square up" reorientation maneuver (blue) and the tactile inspection motion (red). The inspection data is extracted from the sensor stream for classification.

5 Conclusion

We have shown how a legged robot can inspect concrete deterioration in medium and large sewers. Through multiple field trials, we developed a method that allows the quadruped robot *ANYmal* to walk through the sewer system of Zurich, perform tactile inspection of the concrete, and classify the deterioration level.

To cope with the wet and slippery environment in the sewer, we have developed adaptive planar feet with integrated inertial and force/torque sensors. Using an impedance-controlled scratching motion, the robot can probe the terrain with one of its limbs while maintaining balance with the others. We acquired and open-sourced a dataset with 625 samples consisting of the sensor data and ground truth labels supplied by professional sewer inspectors. Training a support vector machine on the dataset allowed us to predict the current state of the concrete deterioration within three classes with overall more than 92% accuracy, as well as more than 89% precision and recall. Analysis of the data shows that even a small set of sensors is sufficient for high classification accuracy.

We planned and executed several inspection missions with the robot, which mimic the approach of human inspectors. Using a shared autonomy framework, we were able to command the robot from outside the sewer in individual missions of up to 145 m while performing the concrete inspection action at regular intervals. While walking, the robot would map the sewer using on-board sensors and record the points of inspection. In post-processing, the state of the concrete was inspected using the pre-trained SVM model. We executed four missions in different areas of the sewer, including climbing between sewers, which demonstrates the advantage of using a legged system in this environment. Within the four missions, we covered a total distance of 294 m in 80 min and performed 130 inspection motions.

While the inspection approach worked well, further improvements are required to increase robustness and reliability. First, the system needs to be tested in narrow, partially flooded, and highly slippery sewers to demonstrate a similar capability to a human inspector. This also includes further robustifying the locomotion system against slip-induced falling. Secondly, more data needs to be collected to confirm the robustness on a more diverse set of sewers, to improve classification accuracy, and eventually, the precision of the prediction. Lastly, the mapping could be improved to cope with straight, featureless and foggy tunnels. Overall, we believe that these shortcomings will be overcome and that legged robots will become a valuable partner in the inspection of sewers.

Acknowledgments

The authors want to thank Roman Weiss and his colleagues from *ERZ - Entsorgung + Recycling Zurich* for their support during the field tests. Klajd Lika and Ilias Patsiaouras from *Bota Systems* helped to integrate the force/torque sensors into the feet and Marco Camurri built the ground truth map. This work has been supported by the European Union's Horizon 2020 research and innovation programme under grant agreement No 780883, by the Swiss National Science Foundation through the National Centre of Competence in Research Robotics (NCCR Robotics), and by the European Space Agency (ESA) and Airbus DS in the framework of the Network Partnering Initiative 481-2016. This work has been conducted as part of ANYmal Research, a community to advance legged robotics.

References

[Bednarek et al., 2019] Bednarek, J., Bednarek, M., Wellhausen, L., Hutter, M., and Walas, K. (2019). What am i touching? learning to classify terrain via haptic sensing. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7187–7193.

[Bellicoso et al., 2018] Bellicoso, C. D., Bjelonic, M., Wellhausen, L., Holtmann, K., Günther, F., Tranzatto, M., Fankhauser, P., and Hutter, M. (2018). Advances in real-world applications for legged robots. *Journal of Field Robotics*, 35(8):1311–1326.

[Berger and Falk, 2009] Berger, C. and Falk, C. (2009). *Zustand der Kanalisation in Deutschland - Ergebnisse der DWA-Umfrage 2009*. Deutsche Vereinigung für Wasserwirtschaft, Abwasser und Abfall e. V. (DWA), Hennef.

[Berger et al., 2016] Berger, C., Falk, C., Hetzel, F., Pinnekamp, J., Roder, S., and Ruppelt, J. P. (2016). *Zustand der Kanalisation in Deutschland - Ergebnisse der DWA-Umfrage 2015*. Deutsche Vereinigung für Wasserwirtschaft, Abwasser und Abfall e. V. (DWA), Hennef.

[Bloesch et al., 2018] Bloesch, M., Burri, M., Sommer, H., Siegwart, R., and Hutter, M. (2018). The two-state implicit filter recursive estimation for mobile robots. *IEEE Robotics and Automation Letters*, 3(1):573–580.

[Chang and Lin, 2011] Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[DIN EN 13508-2:2011, 2011] DIN EN 13508-2:2011 (2011). Investigation and assessment of drain and sewer systems outside buildings - part 2: visual inspection coding system.

[Dyk and Lohaus, 1997] Dyk, C. and Lohaus, J. (1997). *Der Zustand der Kanalisation in Deutschland - Ergebnisse der ATV-Umfrage 1997*. ATV (Abwassertechnische Vereinigung e.V.), Hennef.

[Everett et al., 2008] Everett, T. D., Weykamp, P., Capers, H. A., Cox, W. R., Drda, T. S., Hummel, L., Jensén, P., Juntunen, D. A., Kimball, T., and Washer, G. (2008). Bridge evaluation quality assurance in europe.

[Fankhauser et al., 2016] Fankhauser, P., Bellicoso, C. D., Gehring, C., Dubé, R., Gawel, A., and Hutter, M. (2016). Free gait: An architecture for the versatile control of legged robots. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 1052–1058.

[Fankhauser et al., 2018] Fankhauser, P., Bjelonic, M., Dario Bellicoso, C., Miki, T., and Hutter, M. (2018). Robust rough-terrain locomotion with a quadrupedal robot. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5761–5768.

[Fankhauser et al., 2018] Fankhauser, P., Bloesch, M., and Hutter, M. (2018). Probabilistic terrain mapping for mobile robots with uncertain localization. *IEEE Robotics and Automation Letters (RA-L)*, 3(4):3019–3026.

[Hoepflinger et al., 2010] Hoepflinger, M. A., Remy, C. D., Hutter, M., Spinello, L., and Siegwart, R. (2010). Haptic terrain classification for legged robots. In *2010 IEEE International Conference on Robotics and Automation*, pages 2828–2833.

[Hudon et al., 2011] Hudon, E., Mirza, S., and Frigon, D. (2011). Biodeterioration of concrete sewer pipes: State of the art and research needs. *Journal of Pipeline Systems Engineering and Practice*, 2(2):42–52.

[Hutter et al., 2017a] Hutter, M., Diethelm, R., Bachmann, S., Fankhauser, P., Gehring, C., Tsounis, V., Lauber, A., Guenther, F., Bjelonic, M., Isler, L., Kolvenbach, H., Meyer, K., and Hoepflinger, M. (2017a). Towards a Generic Solution for Inspection of Industrial Sites. In *Field and Service Robots (FSR)*.

[Hutter et al., 2017b] Hutter, M., Gehring, C., Lauber, A., Gunther, F., Bellicoso, C. D., Tsounis, V., Fankhauser, P., Diethelm, R., Bachmann, S., Bloesch, M., Kolvenbach, H., Bjelonic, M., Isler, L., and Meyer, K. (2017b). Anymal - toward legged robots for harsh environments. *Advanced Robotics*, 31(17):918–931.

[Käslin et al., 2018] Käslin, R., Kolvenbach, H., Paez, L., Lika, K., and Hutter, M. (2018). Towards a passive adaptive planar foot with ground orientation and contact force sensing for legged robots. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2018)*.

[Khatib, 1987] Khatib, O. (1987). A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53.

[Kolvenbach et al., 2019] Kolvenbach, H., Bärtschi, C., Wellhausen, L., Grandia, R., and Hutter, M. (2019). Haptic inspection of planetary soils with legged robots. *IEEE Robotics and Automation Letters*, 4(2):1626–1632.

[Kolvenbach and Hutter, 2018] Kolvenbach, H. and Hutter, M. (2018). Life extension: An autonomous docking station for recharging quadrupedal robots. In *Field and Service Robotics*, pages 545–557, Cham. Springer International Publishing.

[Kolvenbach et al., 2019] Kolvenbach, H., Valsecchi, G., Grandia, R., Ruiz, A., Jenelten, F., and Hutter, M. (2019). Tactile inspection of concrete deterioration in sewers with legged robots. In *Field and Service Robotics (FSR)*.

[Marion et al., 2017] Marion, P., Fallon, M., Deits, R., Valenzuela, A., D'Arpino, C. P., Izatt, G., Manuelli, L., Antone, M., Dai, H., Koolen, T., Carter, J., Kuindersma, S., and Tedrake, R. (2017). Director: A user interface designed for robot operation with shared autonomy. *Journal of Field Robotics*, 34:225–426.

[Mirats Tur and Garthwaite, 2010] Mirats Tur, J. M. and Garthwaite, W. (2010). Robotic devices for water main in-pipe inspection: A survey. *Journal of Field Robotics*, 27(4):491–508.

[Parande et al., 2006] Parande, A. K., Ramsamy, P. L., Ethirajan, S., Rao, C. R. K., and Palanisamy, N. (2006). Deterioration of reinforced concrete in sewer environments. *Proceedings of the Institution of Civil Engineers - Municipal Engineer*, 159(1):11–20.

[Seet et al., 2018] Seet, G., Yeo, S., Law, W., Burhan, Wong, C., Sapari, S., and Liau, K. (2018). Design of tunnel inspection robot for large diameter sewers. *Procedia Computer Science*, 133:984 – 990. International Conference on Robotics and Smart Manufacturing (RoSMa2018).

[Walter et al., 2012] Walter, C., Saenz, J., Elkmann, N., Althoff, H., Kutzner, S., and Stuerze, T. (2012). Design considerations of robotic system for cleaning and inspection of large-diameter sewers. *Journal of Field Robotics*, 29(1):186–214.

[Wells and Melchers, 2015] Wells, T. and Melchers, R. (2015). Modelling concrete deterioration in sewers using theory and field observations. *Cement and Concrete Research*, 77:82 – 96.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	Towards Autonomous Inspection of Concrete Deterioration in Sewers with Legged Robots
Publication Status	Published
Publication Details	H. Kolvenbach, D. Wisth, R. Buchanan, G. Valsecchi, R. Grandia, M. Fallon, and M. Hutter, "Towards autonomous inspection of concrete deterioration in sewers with legged robots," *Journal of Field Robotics*, pp. 1-14, 2020.

Student Confirmation

Student Name:	David Wisth		
Contribution to the Paper	My contributions to the paper were: • Modified the Director interface to perform semi-autonomous sewer inspection, including writing scripts to interface with the motion controller. • Traveled to Zurich for integration and the sewer experiments (Dec 2019). • Operated the robot during all field trials in the experiments. • Assisted in paper writing, including producing the 3D maps with sewer inspection points.		
Signature		Date	30/09/2021

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Maurice Fallon		
Supervisor comments This is an accurate desciption		
Signature 	Date	4/10/2021

This completed form should be included in the thesis, at the end of the relevant chapter.

B.2 The Newer College Dataset: Handheld LiDAR, Inertial and Vision with Ground Truth

This paper was presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (2020) [7]. A video summary of this publication can be found at: `https://youtu.be/aIeMPeHDUgs`. The dataset is available to the public at: `https://ori-drs.github.io/newer-college-dataset/`.

The Newer College Dataset: Handheld LiDAR, Inertial and Vision with Ground Truth

Milad Ramezani, Yiduo Wang, Marco Camurri, David Wisth, Matias Mattamala and Maurice Fallon

***Abstract*— In this paper, we present a large dataset with a variety of mobile mapping sensors collected using a handheld device carried at typical walking speeds for nearly 2.2 km around New College, Oxford as well as a series of supplementary datasets with much more aggressive motion and lighting contrast. The datasets include data from two commercially available devices - a stereoscopic-inertial camera and a multi-beam 3D LiDAR, which also provides inertial measurements. Additionally, we used a tripod-mounted survey grade LiDAR scanner to capture a detailed millimeter-accurate 3D map of the test location (containing ∼290 million points). Using the map, we generated a 6 Degrees of Freedom (DoF) ground truth pose for each LiDAR scan (with approximately 3 cm accuracy) to enable better benchmarking of LiDAR and vision localisation, mapping and reconstruction systems. This ground truth is the particular novel contribution of this dataset and we believe that it will enable systematic evaluation which many similar datasets have lacked. The large dataset combines both built environments, open spaces and vegetated areas so as to test localisation and mapping systems such as vision-based navigation, visual and LiDAR SLAM, 3D LiDAR reconstruction and appearance-based place recognition, while the supplementary datasets contain very dynamic motions to introduce more challenges for visual-inertial odometry systems. The datasets are available at:**
`ori.ox.ac.uk/datasets/newer-college-dataset`

I. Introduction

Research in robotics and autonomous navigation has benefited significantly from the public availability of standard datasets which enable systematic testing and validation of algorithms. Over the past 10 years, datasets such as KITTI [1], the New College [2] and EuRoC MAV [3] have been released and provided a transparent benchmark of performance. These datasets were collected on a variety of platforms (UGVs, UAVs and Autonomous Vehicles) with reference poses based upon GPS/INS [1], dead reckoning [2], and laser trackers/motion capture systems [3].

Many vehicular benchmarks use a form of GPS/INS fusion for ground truth but do not provide precise local accuracy. This is specially noted as a shortcoming of the KITTI odometry benchmark which does not use length scales of less than 100 m for this reason[1]. Tripod-mounted laser trackers follow a prism placed on the robot to achieve precise localisation. However, they cannot maintain line of sight to the robot for large experiments. Motion capture systems provide accurate 6 DoF ground truth but are limited to small indoor facilities.

The authors are with the Oxford Robotics Institute, University of Oxford, UK. {`milad, ywang, mcamurri, davidw, matias, mfallon`}`@robots.ox.ac.uk`

[1]http://www.cvlibs.net/datasets/kitti/eval_odometry.php

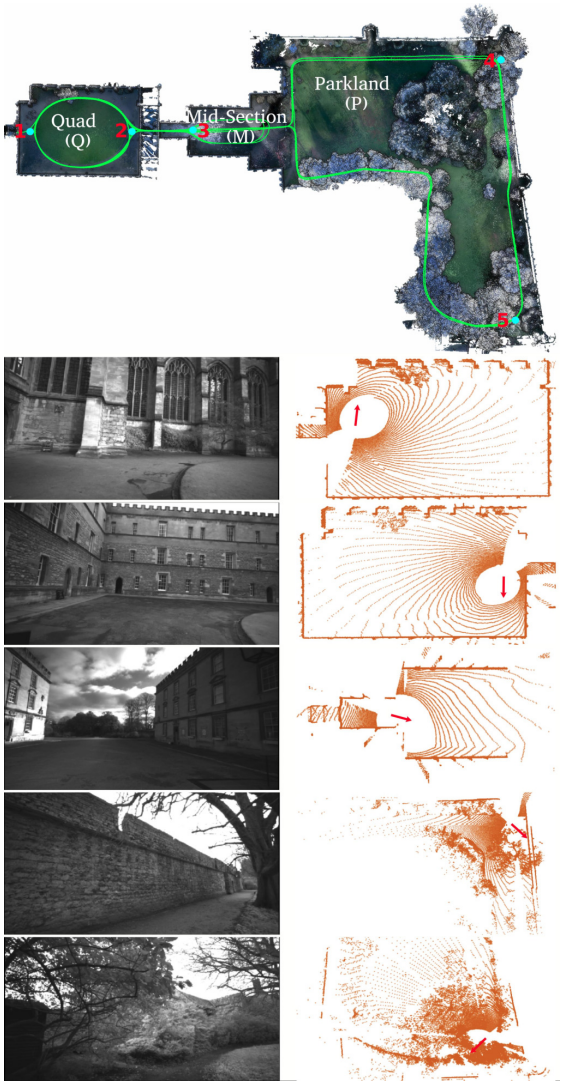


Fig. 1: **Top**: A bird's eye view of the 3D model of New College generated with a Leica BLK360 scanner and the ground truth trajectory obtained by the proposed approach. **Bottom**: Image samples and corresponding 3D LiDAR scans from the Quad (locations 1 and 2), the Mid-Section (location 3) and the Parkland (location 4 and 5) show varied environments suited for assessment of localisation and mapping algorithms in autonomous robot navigation.

We aim to provide a large-scale dataset which provides centimeter accuracy so as to enable evaluation of short length-scale odometry, as well as large scale drift. The raw data files are accompanied by a precise 3D model constructed using a survey-grade LiDAR scanner. Utilizing the 3D model, we inferred the location of the device using LiDAR ICP at 10 Hz across the entire location. Fig. 1 shows a plan view of the 3D model with vision and corresponding LiDAR samples of the data from different locations.

The data was collected by a handheld device carried by a person at walking speed, unlike the majority of the released datasets acquired from a robotics platforms. The handheld device comprises a 3D LiDAR and a stereo camera each with a self-contained IMU. In particular, we use commercially available low-cost sensors — the widely used Intel Realsense D435i and a 64-beam Ouster OS-1 LiDAR scanner. The walking motion can, to a degree, replicate the jerky motion of a flying drone or a quadruped.

This dataset revisits New College, Oxford — the location of the original Smith *et al.* [2] dataset. This dataset is particularly useful due to its variety of scenarios, including structured buildings, vegetation, and open space areas with lack of texture. We replicate the sequences of the original dataset and then go on to add more aggressive, faster sequences designed to test algorithms for visual navigation, LiDAR SLAM, reconstruction and place recognition.

The remainder of the paper is structured as follows: Sec. II reviews related work followed by a description of the device in Sec. III. Sec. IV details our dataset. Sec. V explains how we determined the ground truth. Sec. VI demonstrates example usage of our dataset for a set of navigation research topics in mobile robotics before a summary is presented in Sec. VII.

II. Related Work

Dataset papers can be divided into two subsections based on the platform carrying the sensor; a (self-driving) car is often used for outdoor, large-scale datasets while robots or humans-carried datasets are typically much smaller and often indoors. Focusing on vision, IMU and LiDAR modules, Tab. I, provides the basic details of the datasets discussed in the following.

A. Vehicle-based Datasets

There is a large body of localisation and mapping data targeting autonomous navigation for ground vehicles.

The MIT DARPA Urban Challenge dataset [4] is one of the first major UGV datasets. It was collected using MIT's Talos vehicle over the course of a 90 km traverse spanning 7 hours of self-driving. The ground truth was provided by the integration of a high-precision GPS and an INS.

The Marulan multi-modal datasets [5] were gathered by an unmanned ground vehicle, Argo, in which artificial dust, smoke and rain added extra challenges for on-board perception sensors.

Similarly to the MIT dataset, the Ford Campus dataset [6] was obtained along an almost 6 km traverse of a mock-up urban environment. The Malaga urban dataset [7] has the distinctive feature of high-resolution pair images captured for over an almost 37 km trajectory in urban scenarios.

The KITTI dataset [1] was collected on a car driving around the streets of Karlsruhe and it has significantly catalyzed autonomous vehicle navigation research. The dataset consists of stereoscopic image pairs for the sequences of 39.2 km length and has been used for a variety of SLAM/odometry and object detection tasks. KITTI provides 6 DoF ground truth trajectory for all the traversals using RTK-GPS/INS with accuracy below 10 cm. However, this accuracy is not guaranteed in GPS-deprived areas such as urban canyons. Furthermore, the IMU readings and images are not synchronized which effects the performance of many visual-inertial odometry algorithms.

The longest autonomous driving dataset we are familiar with is the Oxford RobotCar dataset [8] collected in all natural weather conditions over the course of 1000 km driven through central Oxford. Recently, the Complex Urban LiDAR dataset [9] was gathered and is targeted at multiple challenges in complex urban areas including GPS loss, multi-lane highways and dynamic entities such as pedestrians, bikes and cars. This multi-faceted dataset was acquired over the course of approximately 180 km of travel.

Nonetheless, the ground truth of the aforementioned driverless-car datasets is highly dependent on GPS observations and therefore, as noted in [8] and [9], the usage of ground truth is not recommended in GPS-deprived areas for the evaluation of localisation and mapping algorithms.

B. Mobile Robot or Human-carried Datasets

The New College Vision and LiDAR dataset [2], which is a motivation for this dataset, provides carefully timestamped laser range data, stereo and omnidirectional imagery along with 5 DoF odometry (2D position and roll, pitch, heading). The data was collected using a wheeled robot, a Segway, over a 2.2 km traverse of the college's grounds and the adjoining garden area. No ground truth is available for this dataset.

Similar to the New College, the North Campus Long-Term (NCLT) dataset [11] was gathered across a college campus, indoor and outdoor, over 147.4 km traverse and 15 months, again with a Segway. The significant difference is the provision of the ground truth using LiDAR scan matching and high-accuracy RTK-GPS. Although this approach potentially provides centimeter accuracy, it is susceptible to drift indoors or near buildings which cause GPS multi-path errors.

Recent datasets such as EuRoC MAV [3], Zurich Urban MAV [14] and PennCOSYVIO [13] specifically focused on visual-inertial odometry and visual SLAM. The data in EuRoC and Zurich was gathered using a micro aerial vehicle flying indoor and outdoor for 0.9 km and 2 km, respectively, while the data in PennCOSYVIO was obtained from a handheld device, similar to our platform, along 0.6 km trajectory outdoor.

To provide accurate ground truth at a millimeter level, EuRoC MAV employed a laser tracker and a motion capture system. However, a laser tracker only provides measurement

Dataset	Year	Environment	Ground Truth	Sensors			Platform
				IMUs	LiDAR	Cameras	
Rawseeds [10]	2009	Structured	2D+Yaw Visual Markers/Laser	accel/gyro @128Hz	2 2D-Hokuyo @10Hz 2 2D-SICK @75Hz	Trinocular Vision: 3×640×480 @30Hz RGB: 640×480 @30Hz Fisheye RGB: 640×640 @15Hz	Wheeled Robot
New College [2]	2009	Structured Vegetated	N/A	gyro @28Hz	2 2D-SICK @75Hz	BumbleBee: 2×512×384 @20Hz LadyBug 2: 5×384×512 @3Hz	Wheeled Robot
DARPA [4]	2010	Structured Urban	GPS/INS	N/A	12 2D-SICK @ 75Hz 3D-Velodyne HDL-64E @15Hz	4 Point Grey: 4×376×240 @10Hz Point Grey: 752×480 @22.8Hz, Wider FOV	Car
Marulan [5]	2010	Open Area	DGPS/INS	accel/gyro @50Hz	4 2D-SICK @18Hz	Mono Prosilica: 1360×1024 @10Hz Infrared Raytheon: 640×480 @12.5Hz	Wheeled Robot
Ford Campus [6]	2011	Urban	GPS/INS	accel/gyro @100Hz	3D-Velodyne HDL-64E @10Hz 2 2D-Riegl LMS @	LadyBug 3: 6×1600×600 @8Hz	Car
KITTI [1]	2013	Structured Urban	RTK GPS/INS	accel/gyro @10Hz	3D-Velodyne HDL-64E @10Hz	2 Point Grey(gray): 2×1392×512 @10Hz 2 Point Grey(color): 2×1392×512 @10Hz	Car
Malaga [7]	2014	Structured Urban	N/A	accel/gyro @100Hz	3 2D-Hokuyo @40Hz 2 2D-SICK @75Hz	BumbleBee: 2×1024×768 @20Hz	Car
NCLT [11]	2015	Structured Urban	RTK-GPS LiDAR-SLAM	accel/gyro @100Hz	3D-Velodyne HDL-32E @10Hz 2 2D-Hokuyo @10/40HZ	LadyBug 3: 6×1600×1200 @5Hz	Wheeled Robot
EuRoC MAV [12]	2016	Structured	6DOF Vicon 3D Laser Tracker	accel/gyro @200Hz	N/A	2 MT9V034: 2×752×480 @20Hz	UAV
PennCOSYVIO [13]	2017	Structured	Visual Tags	ADIS accel/gyro @200Hz 2 Tango accel @128Hz 2 Tango gyro @100Hz	N/A	3 GoPro (color): 3×1920×1080 @30Hz 2 MT9V034 (gray): 2×752×480 @20Hz	Handheld
Zurich Urban MAV [14]	2017	Structured Urban	Aerial Photogrammetry	accel/gyro @10Hz	N/A	GoPro (color): 1920×1080 @30Hz	UAV
Oxford RobotCar [8]	2017	Structured Urban	GPS/INS Not Recommended	accel/gyro @50Hz	2 2D-SICK @50Hz 3D-SICK @12.5Hz	BumbleBee: 2×1280×960 @16Hz 3 Grasshoper2: 3×1024×1024 @11.1Hz	Car
TUM VI [15]	2018	Structured	6DOF MoCap Available at Start/End	accel/gyro @200Hz	N/A	IDS (gray): 2×1024×1024 @20Hz	Handheld
Complex Urban [9]	2019	Structured Urban	SLAM Not Recommended	accel/gyro @200Hz FOG @1000Hz	2 3D-Velodyne-16 @10Hz 2 2D-SICK @100Hz	FLIR (color): 2×1280×560 @10Hz	Car
Our Dataset	2020	Structured Vegetated	6DOF ICP Localisation	accel @250Hz gyro @400Hz	3D-Ouster-64 @10Hz	D435i (Infrared): 2×848×480 @30Hz	Handheld

TABLE I: Comparison of related datasets used in robotics and autonomous systems research.

of position but not orientation. Additionally, tracking is lost if the robot travels beyond the line of sight of the tracker. Motion capture systems are limited to the experiments within small areas and indoors. The ground truth in Zurich Urban and PennCOSYVIO was obtained using aerial photogrammetry and close-range photogrammetry, respectively. Nonetheless, because photogrammetric techniques rely on image observations, it is hard to achieve an accuracy below 10 cm, as reported in [13], if the observations are not within a few meters from the camera.

The Rawseeds dataset [10] was used to develop vision and LiDAR-based techniques for indoor navigation. By deploying multiple pre-calibrated cameras or laser scanners in the operating environments, an external network is formed from which the trajectory of the robot was estimated. However, these techniques require continuous line of sight limiting the scale of experiments.

We use a unique approach for determining ground truth that, to the best of our knowledge, has not been used in the published datasets. Our approach is based upon the registration of individual LiDAR scans with an accurate prior map, utilizing ICP. The method is properly explained in Sec. V.

III. The Handheld Device

Our device is shown in Fig. 2 (top-left). The sensors are rigidly attached to a precisely 3D-printed base. The top-right figure shows a 3D model of the device from front view. A complete URDF model of the device is available as open source ROS package[2] and it is shown in Fig. 2 (middle). The Ouster LiDAR is mounted on the top, with a clockwise rotation of 45 degrees for cable routing. Tab. II overviews the sensors used in our handheld device.

[2]https://github.com/ori-drs/rooster_description

Sensor	Type	Rate	Characteristics
LiDAR	Ouster, OS1-64	10 Hz	64 Channels, 120 m Range 45° Vertical FOV 1024 Horizontal Resolution
Cameras	Intel Realsense-D435i	30 Hz	Global shutter (Infrared) 848×480
LiDAR IMU	ICM-20948	100 Hz	3-axis Gyroscope 3-axis Accelerometer
Camera IMU	Bosch BMI055	400 Hz 250 Hz	3-axis Gyroscope 3-axis Accelerometer

TABLE II: Overview of the sensors in our handheld device.

The Intel Realsense is a commodity-grade stereo-inertial camera while the Ouster LiDAR has 64 beams, which provides much denser data than many other LiDAR datasets. Both sensors have become commonly used in mobile robotics in the last 2 years, for example the ongoing DARPA Subterranean Challenge.

To distinguish the sensor frames, we use the following abbreviations:

- OS_I: The IMU coordinate system in the LiDAR.
- OS_L: The LiDAR coordinate system with respect to which the point clouds are read.
- RS_C1: The left camera coordinate system which is considered as the base frame.
- RS_I: The IMU coordinate system in the stereo setup.
- RS_C2: The right camera coordinate system.

We use the open source camera and IMU calibration toolbox Kalibr [16], [17] to compute the intrinsic calibration of the Realsense cameras as well as their extrinsics. As our device is not hardware synchronized, it is crucial to leverage as much software/network synchronization as possible. We perform spatio-temporal calibration between the cameras and the two IMUs embedded in the Realsense and the Ouster sensor. As described in [16], the temporal offsets between measurements of the Realsense IMU and Ouster IMU with respect to the Realsense cameras are estimated using batch,

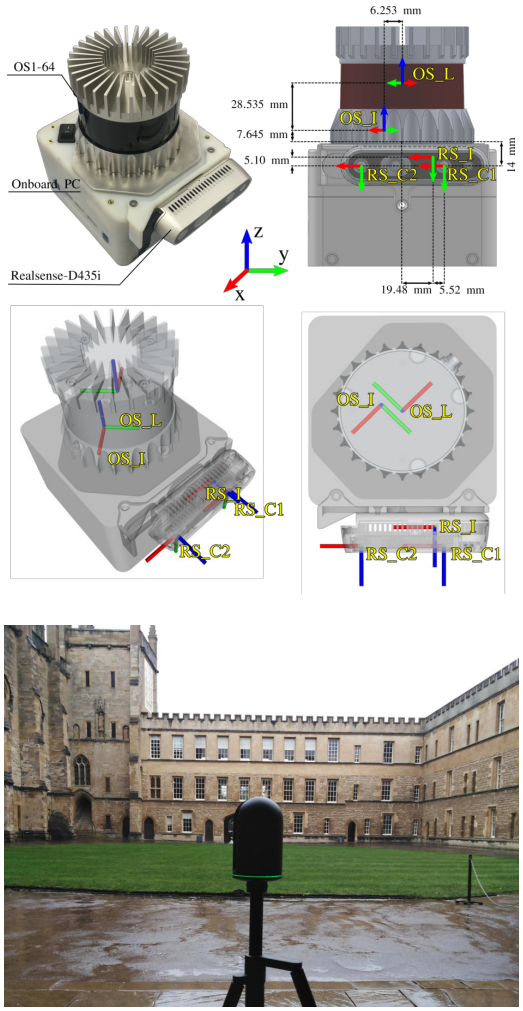


Fig. 2: Devices used in our dataset. **Top left:** Our custom built handheld device; **Top right:** front view of the 3D CAD model with reference frames their relative distances; **Middle (left to right):** Isometric and top down views of the URDF model with reference frames; **Bottom:** Leica BLK360 for creating the ground truth map.

continuous-time, maximum-likelihood estimation. In addition, by comparing the angular velocities of both IMUs, we found out the sensors drift relative to one another at about 58 ms per hour. We studied it and details are presented on the dataset website. We also provide the calibration dataset along with the main dataset. The Ouster LiDAR synchronizes with the recording computer using the Precision Time Protocol (PTP), which achieves sub-microsecond accuracy [18].

Our handheld device is equipped with an onboard Intel Core i7 NUC computer kit. To get the correct timestamps for IMU messages of the Realsense D435i, we use Ubuntu 18.04 with the Linux kernel of 4.15.0-74-generic and version 2.32.1.0 for Realsense libraries[3]. The firmware installed on the sensor was version 0.5.10.13.00. In our experience, we found the Realsense to be sensitive to the right combination of kernel, driver and firmwares to be used. In particular for the IMU messages, special care needs to be taken to achieve a reliable configuration. The device was powered by an 8000 mAh LI-PO battery.

[3]https://github.com/IntelRealSense/librealsense

IV. Data Collection

We have collected a variety of datasets with different speeds of walking and turning. They are organised by the aggressiveness of the motion and described in more detail on our website.

Since this paper is motivated by the New College dataset [2], the longest experiment carefully followed the same path that the original data collection followed in 2009 in New College, Oxford, UK. Borrowing the terminology from [2], we break the dataset into 3 main sections: Quad (**Q**), Mid-Section (**M**) and Parkland (**P**). Quad has an oval lawn area at the center and is surrounded with medieval buildings with repeating architecture. The Mid-Section includes a short tunnel where illumination changes quickly, leading to an open area which is flanked by buildings on the northern and southern sides. Parkland is a garden area connected to the Mid-Section through a wrought iron gate from west.

The data was gathered during early February 2020 from morning to noon. The handheld device was held by a person walking at constant pace about 1 m/s. To reduce the number of blocked laser beams, the device was held above the shoulder throughout the dataset. It is worth mentioning that the movement was not intended to be highly dynamic. However, natural vibration caused by human walking and hand motion is inevitable. The motion induced by this walking gait would make the dataset similar to a flying UAV.

Following the same path as the original New College dataset, the data collection began from the west of the Quad. As illustrated in Fig. 3, after three and a half loops, clockwise, around the Quad with the duration of about 390 seconds (Q1), the Quad and the Mid-Section were traversed back and forth twice (M1-Q2 and M2-Q3), counter-clockwise in periods M1, M2 and Q2 while clockwise in period Q3. These sections took until second 820 of the data collection, followed by a straight traversal in period M3 which took 60 seconds.

The data collection continued by entering the Parkland at second 1240 and it was circumnavigated twice clockwise in 610 seconds. Unlike the paved path in the Quad and the Mid-Section, the path in the Parkland was gravel and was muddy in parts due to the time of data collection. Since the path is adjacent to a vegetated border and partly passes along dense foliage, it is hard to see any building structure, posing a challenge to vision-based localisation techniques.

The data continued to be captured by walking straight back to the Quad (M4 with the same duration as M3) and this time the sensor was carried counter-clockwise for about 105 seconds, followed by walking straight back to the Parkland and taking in an extra loop in this area (but

counter-clockwise). The traversal in the P3 ended at Second 2180. Finally, the data collection ended by walking back to starting point, i.e. passing through the Mid-Section (M6) and half circumnavigating the Quad (Q5) counter-clockwise. Altogether, the time duration of the entire dataset is 2300 seconds.

To extend the usefulness of the dataset, we carried out further experiments with faster walking and aggressive motions of the device. These datasets are listed as below:

- *Shorter experiment* (1500 seconds): A slightly shorter version of the main dataset, at the same walking speed.
- *Quad with dynamics* (398 seconds): 4 clockwise loops around the Quad at a faster walking speed (1.5msec/): for 2 loops the device was held flat; 1 loop had the device swinging from side to side; and 1 loop where the swinging speed varied.
- *Dynamic spinning* (120 seconds): Aggressive angular motions of the device, rotation rates of 2.5 rad/sec while pointing at a corner of the Quad. Useful for testing motion distortion.
- *Parkland mound* (500 seconds): Walking a few laps in the grassy area of the Parkland, up and down the Mound's stairs and finally a loop around the Mound at a fast walking speed.

For more details regarding these datasets, we refer the readers to the dataset website.

V. Ground Truth

The ground truth poses of this dataset are obtained with an approach whose core uses Iterative Closest Point (ICP), a well-known method for registration of 3D shapes [19]. When provided with a prior pose from which the point clouds are captured, ICP minimizes the Euclidean distance between closest points, namely correspondences, to iteratively compute the relative displacement between the reading cloud and the reference cloud. The former refers to the cloud which is intended to be registered with the latter.

To provide the prior map of our dataset, we use a survey-grade 3D imaging laser scanner, Leica BLK360[4] (Fig. 2 (bottom)). For the New College environment with approximately the size of 135×225 m^2, 47 static point clouds were captured to fully map the area. This took over 8 hours. The capture locations were decided to be closer together in the Parkland due to foliage and less structured features. All the point clouds were matched with over 90% of inliers. The fully merged map is seen in Fig. 1 (top). According to the Leica BLK360 datasheet[4]], the achievable accuracy for 3D points in the map is 6 mm at range 10 m and 8 mm at range 20 m. Hence, we can conclude that the accuracy of the majority of points of the entire map is better than 1 cm since the range of the points (in the Quad, Mid-Section and the perimeter of the Parkland) in the map is no more than 20 meters from at least one of the scanning stations. The final map consists of about 290 million points.

[4] https://leica-geosystems.com/en-gb/products/laser-scanners/scanners/blk360

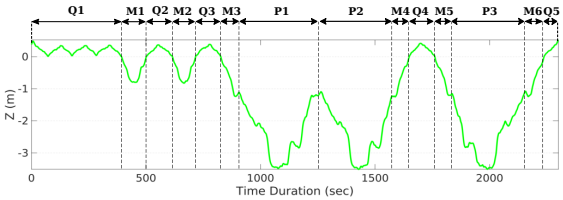


Fig. 3: Ground truth elevation in the entire dataset.

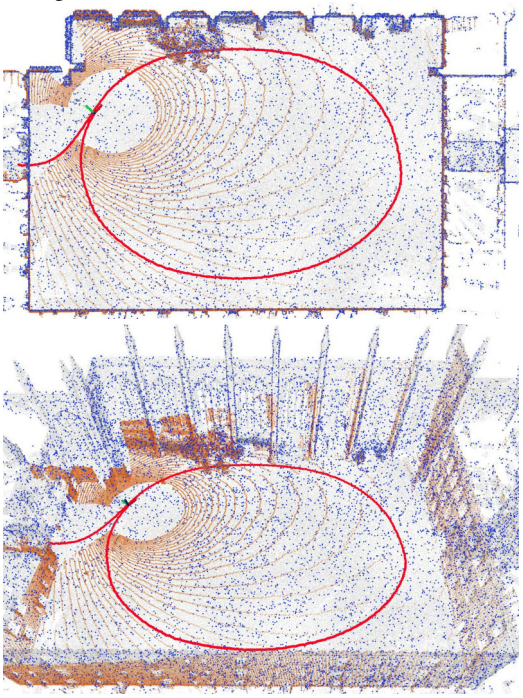


Fig. 4: Plan and perspective view of the Quad when current laser scan (in maroon) is registered against the reference cloud (in blue). The reference cloud is part of the prior map (in gray) which is cropped around the pose.

Having generated the accurate prior map, we first downsample the map cloud to 1 cm resolution. This way, we reduce the map to about 17 million points enabling us to use it in our localisation approach without an observable drop in registration accuracy. Further, we dynamically crop the pointcloud to create a reference cloud in the area of 100 m by 100 m around the sensor's pose. To localise individual scans, we use a libpointmatcher filter chain [20] to remove the outliers in the clouds and finally register the scans against the map. Fig. 4 demonstrates the ground truth registration procedure with a single Ouster scan. It is worth noting that the ground truth poses are with respect to the base frame which is the center of the left camera, as described in Sec. III. This facilitates the estimation of the camera poses at 30 Hz through interpolation.

Evaluation of ground truth accuracy is challenging for certain scenarios in the dataset – such as the confined areas in

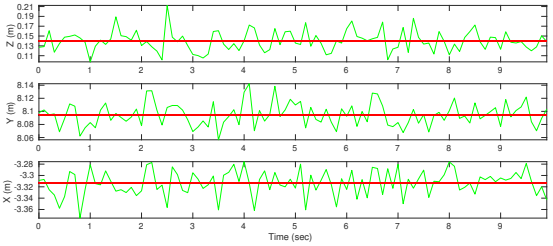


Fig. 5: Ground truth position values for the first 10 seconds of the dataset when the device was stationary. Red lines show the mean values over this period of time.

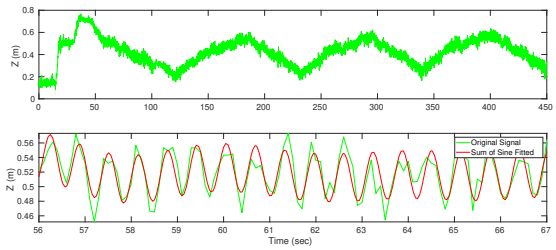


Fig. 6: **Top**: Ground truth z values over the course of 450 seconds of the dataset when walking around the quad (**Q1**). **Bottom**: A closer analysis over 12 seconds. The fitted red curve shows the pattern of the walking gait.

the Mid-section and the natural environment in the Parkland. Fig. 5 and Fig. 6 illustrate the accuracy of the ground truth. Fig. 5 shows the behaviour of the ICP ground truth procedure along x, y and z for the first 10 seconds of the dataset when the device was stationary. The standard deviations along x, y and z are 2 cm, 1.6 cm and 2 cm or approximately 3 cm overall. Evaluating the localisation performance when the device is in motion is more challenging. It involves dynamic effects such as LiDAR motion distortion and it's accuracy likely to be reduced lower. Note Fig. 6 demonstrates the periodic behaviour of the trajectory over the course of the three Quad loops (**Q1**). A closer analysis of 12 seconds of the trajectory (Fig. 6, bottom) shows periodicity of the trajectory due to the walking motion — an indication of the accuracy.

VI. Example Dataset Usage

This section presents example results for a series of potential uses of our dataset in localisation and mapping research areas: LiDAR-SLAM, Appearance-Based Loop-Closure, 3D Reconstruction and Visual (Inertial) Odometry.

A. LiDAR SLAM

We use our LiDAR SLAM system detailed in [21] to estimate ego-motion at 2 Hz and find loop-closures geometrically. Fig. 9 (middle) shows the SLAM trajectory for the entire dataset. This demonstrates that our dataset is useful for LiDAR-based localisation systems.

B. Visual Appearance-Based Loop-Closure

As an illustration of our dataset used in visual place recognition, we used DBoW2 [22] with ORB features [23]. We computed the similarity score against all the poses (spaced at 2 m intervals) traveled in the past and applied a threshold to obtain loop candidates. Fig. 7 shows the similarity matrix, a square matrix indicates whether the nodes in a pose graph are similar or not, and some examples of the most similar views captured at different times are shown.

C. LiDAR 3D Reconstruction

Our dataset can also be used for 3D reconstruction, as demonstrated in Fig. 8. The top row represents the surface mesh generated from the Leica ground truth map using Poisson surface reconstruction [24]. The bottom row shows the mesh created by using the Ouster laser scans registered against the prior map (i.e. leveraging the ground truth localisation). We applied the filtering and mesh generation tools provided by PCL [25] including Moving Least Squares (MLS) smoothing [26], a voxel grid filter of 5 cm resolution and the greedy triangulation projection [27] for the reconstruction.

D. Visual (Inertial) Odometry

We provide visual and inertial measurements which are software time synchronized, in our dataset, as well as providing calibration files to test on two visual odometry methods. We tested ORB-SLAM2 [28] for basic stereo odometry with loop closures disabled. Fig. 9 (bottom) shows the trajectory estimated by this approach using the image pairs for the first 1483 seconds of the dataset. We also used our visual-inertial odometry approach, VILENS [29], which carries out windowed smoothing of the two measurements sources for the same period.

VII. Conclusion and Future Work

In this paper, we presented the Newer College Vision and LiDAR dataset. By leveraging a highly accurate and detailed prior map, we determined accurate 6 DoF ground truth for the entire dataset, which distinguishes our dataset from many others. We used a modern visual-inertial camera and a dense 3D LiDAR sensor and provide the dataset in both the original ROSbags and individual files (such as png images and csv files).

We also demonstrated the use of the dataset for different subproblems in mobile robotics and navigation. It is our intent to further extend our dataset with additional difficult sequences including aggressive motions and make them available publicly in the near future. A demonstration video is available at: `https://youtu.be/aIeMPeHDUgs`

VIII. Acknowledgment

The authors would like to thank the members of the Oxford Robotics Institute (ORI) who helped with the creation of this dataset release, especially Chris Prahacs, Simon Venn and Benoit Casseau. We also thank the personnel of New College for facilitating our data collection.

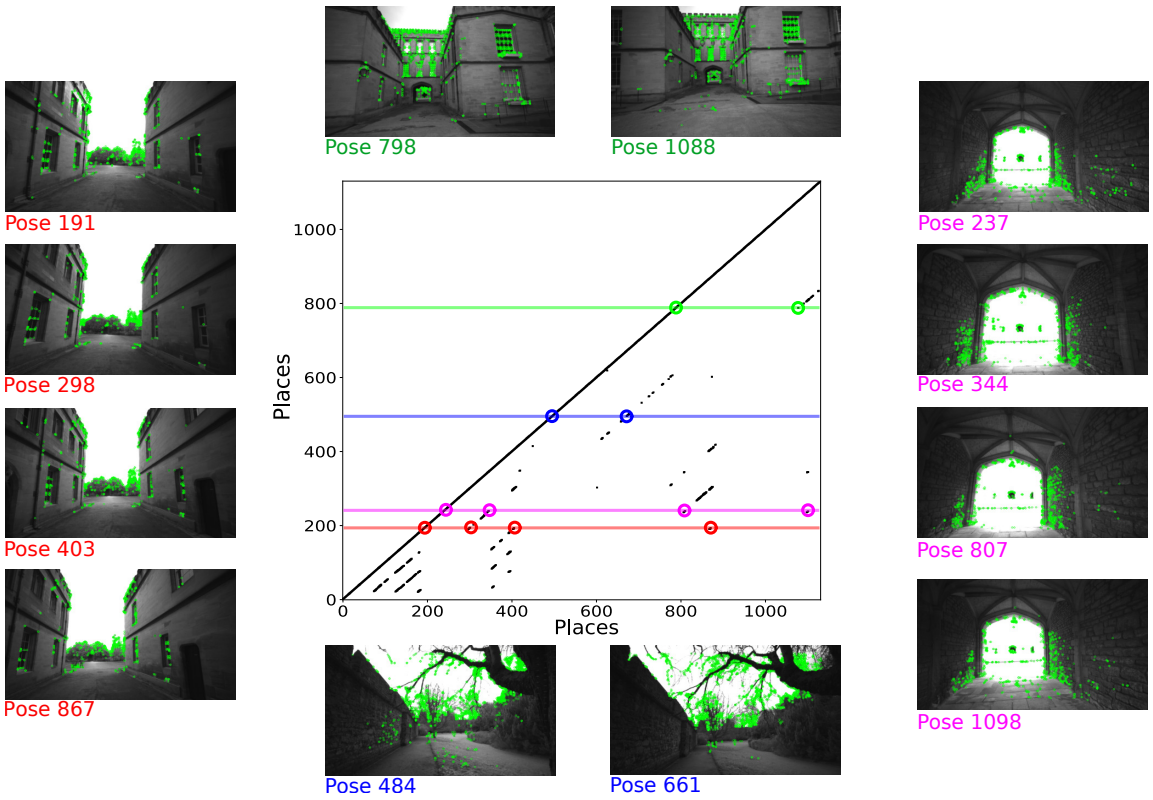


Fig. 7: An example of dataset usage for appearance-based loop-closure detection.

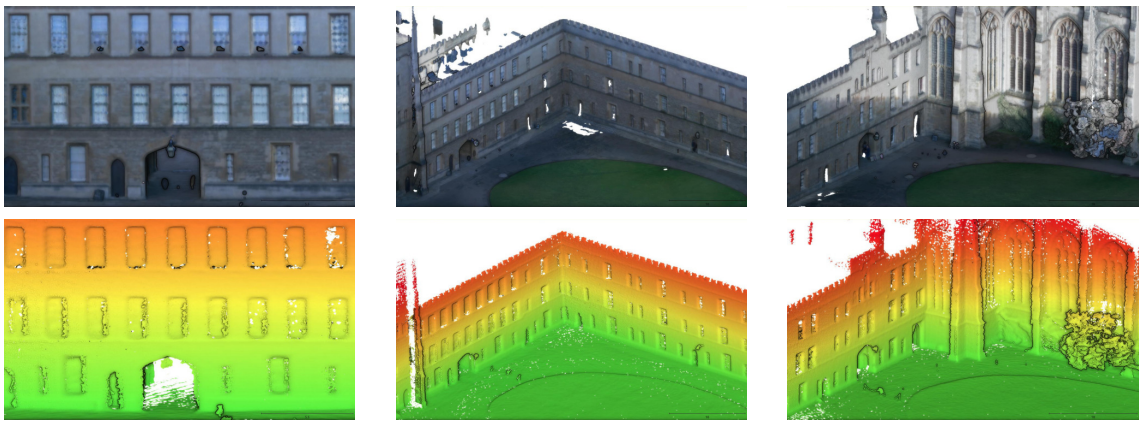


Fig. 8: 3D reconstruction of the Quad from different view points using Ouster laser scans (bottom row) and using the ground truth map (top row). The heat map displays the height from the ground along z axis w.r.t. map frame.

This research was supported by the Innovate UK-funded ORCA Robotics Hub (EP/R026173/1) and the EU H2020 Project THING. Maurice Fallon is supported by a Royal Society University Research Fellowship. Matias Mattamala is supported by the National Agency for Research and Development (ANID) / Scholarship Program / DOCTORADO BECAS CHILE/2019 - 72200291

References

[1] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics: The KITTI Dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[2] M. Smith, I. Baldwin, W. Churchill, R. Paul, and P. Newman, "The New College Vision and Laser Data Set," *The International Journal of Robotics Research*, vol. 28, no. 5, pp. 595–599, 2009.

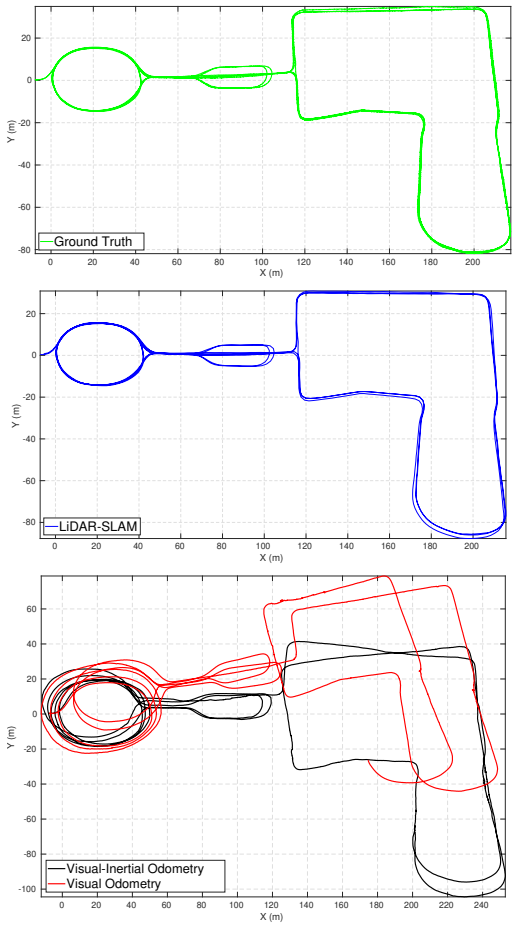


Fig. 9: Ground truth trajectory (top) with our LiDAR-SLAM trajectory (middle) throughout the data. Bottom are the trajectory of vision-based localisation systems, ORB-SLAM2 (red) with loop-closure disabled and VILENS (black).

[3] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, 2016. [Online]. Available: http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.abstract

[4] A. S. Huang, M. Antone, E. Olson, L. Fletcher, D. Moore, S. Teller, and J. Leonard, "A High-rate, Heterogeneous Data Set From The DARPA Urban Challenge," *The International Journal of Robotics Research*, vol. 29, no. 13, pp. 1595–1601, 2010.

[5] T. Peynot, S. Scheding, and S. Terho, "The Marulan Data Sets: Multi-sensor Perception in a Natural Environment with Challenging Conditions," *The International Journal of Robotics Research*, vol. 29, no. 13, pp. 1602–1607, 2010.

[6] G. Pandey, J. R. McBride, and R. M. Eustice, "Ford Campus Vision and Lidar Data Set," *The International Journal of Robotics Research*, vol. 30, no. 13, pp. 1543–1552, 2011.

[7] J.-L. Blanco-Claraco, F.-Á. Moreno-Dueñas, and J. González-Jiménez, "The Málaga urban dataset: High-rate stereo and LiDAR in a realistic urban scenario," *The International Journal of Robotics Research*, vol. 33, no. 2, pp. 207–214, 2014.

[8] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The Oxford RobotCar dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.

[9] J. Jeong, Y. Cho, Y.-S. Shin, H. Roh, and A. Kim, "Complex urban dataset with multi-level sensors from highly diverse urban environments," *The International Journal of Robotics Research*, vol. 38, no. 6, pp. 642–657, 2019.

[10] S. Ceriani, G. Fontana, A. Giusti, D. Marzorati, M. Matteucci, D. Migliore, D. Rizzi, D. G. Sorrenti, and P. Taddei, "Rawseeds ground truth collection systems for indoor self-localization and mapping," *Autonomous Robots*, vol. 27, no. 4, p. 353, 2009.

[11] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of Michigan North Campus Long-Term Vision and Lidar Dataset," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2016.

[12] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[13] B. Pfrommer, N. Sanket, K. Daniilidis, and J. Cleveland, "Penncosyvio: A Challenging Visual Inertial Odometry Benchmark," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3847–3854.

[14] A. L. Majdik, C. Till, and D. Scaramuzza, "The Zurich Urban Micro Aerial Vehicle Dataset," *The International Journal of Robotics Research*, vol. 36, no. 3, pp. 269–273, 2017.

[15] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The TUM VI Benchmark for Evaluating Visual-Inertial Odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 1680–1687.

[16] P. Furgale, J. Rehder, and R. Siegwart, "Unified Temporal and Spatial Calibration for Multi-Sensor Systems," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 1280–1286.

[17] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending Kalibr: Calibrating the Extrinsics of Multiple IMUs and of Individual Axes," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 4304–4311.

[18] W. . Chin and S. . Chen, "Ieee 1588 clock synchronization using dual slave clocks in a slave," *IEEE Communications Letters*, vol. 13, no. 6, pp. 456–458, June 2009.

[19] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," in *Sensor fusion IV: control paradigms and data structures*, vol. 1611. International Society for Optics and Photonics, 1992, pp. 586–606.

[20] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP Variants on Real-World Data Sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, Feb. 2013.

[21] M. Ramezani, G. Tinchev, E. Iuganov, and M. Fallon, "Online LiDAR-SLAM for Legged Robots with Robust Registration and Deep-Learned Loop Closure," *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[22] D. Gálvez-López and J. D. Tardos, "Bags of Binary Words for Fast Place Recognition in Image Sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, oct 2012. [Online]. Available: http://ieeexplore.ieee.org/document/6202705/

[23] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proceedings of the IEEE International Conference on Computer Vision*, 2011, pp. 2564–2571.

[24] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson Surface Reconstruction," in *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, ser. SGP '06. Goslar, DEU: Eurographics Association, 2006, p. 61–70.

[25] R. B. Rusu and S. Cousins, "3D is here: Point Cloud Library (PCL)," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[26] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and Rendering Point Set Surfaces," *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 1, pp. 3–15, Jan 2003.

[27] Z. C. Marton, R. B. Rusu, and M. Beetz, "On Fast Surface Reconstruction Methods for Large and Noisy Point Clouds," in *2009 IEEE International Conference on Robotics and Automation*, May 2009, pp. 3218–3223.

[28] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[29] D. Wisth, M. Camurri, and M. Fallon, "Robust Legged Robot State Estimation Using Factor Graph Optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4507–4514, 2019.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	The Newer College Dataset: Handheld LiDAR, Inertial and Vision with Ground Truth
Publication Status	Published
Publication Details	M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The Newer College Dataset: Handheld LiDAR, Inertial and Vision with Ground Truth," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

Student Confirmation

Student Name:	David Wisth		
Contribution to the Paper	My contributions to the paper were: • Assisted with the development and testing of the mobile sensor rig. • Helped refine data collection procedures to ensure high quality data. • Tested the dataset with VILENS. • Added camera-lidar time calibration, as well as ground truth for new dynamic datasets.		
Signature		Date	30/09/2021

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Maurice Fallon		
Supervisor comments This is an accurate desciption		
Signature 	Date	4/10/2021

This completed form should be included in the thesis, at the end of the relevant chapter.

B.3 CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge

This article was published in the Field Robotics journal in 2022 [17]. A video summary of this publication can be found at: `http://bit.ly/cerberus2020`.

CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge

Marco Tranzatto[1] * **Frank Mascarich**[2] **Lukas Bernreiter**[3] **Carolina Godinho**[4] **Marco Camurri**[5] **Shehryar Khattak**[1] **Tung Dang**[2] **Victor Reijgwart**[3] **Johannes Löje**[4] **David Wisth**[5] **Samuel Zimmermann**[1] **Huan Nguyen**[2] **Marius Fehr**[3] **Lukas Solanka**[4] **Russell Buchanan**[5] **Marko Bjelonic**[1] **Nikhil Khedekar**[2] **Mathieu Valceschini**[4] **Fabian Jenelten**[1] **Mihir Dharmadhikari**[2] **Timon Homberger**[1] **Paolo De Petris**[2] **Lorenz Wellhausen**[1] **Mihir Kulkarni**[2] **Takahiro Miki**[1] **Satchel Hirsch**[2] **Markus Montenegro**[1] **Christos Papachristos**[2] **Fabian Tresoldi**[1] **Jan Carius**[1] **Giorgio Valsecchi**[1] **Joonho Lee**[1] **Konrad Meyer**[1] **Xiangyu Wu**[6] **Juan Nieto**[3] **Andy Smith**[7] **Marco Hutter**[1] **Roland Siegwart**[3] **Mark Mueller**[6] **Maurice Fallon**[5] **Kostas Alexis**[2]

Abstract

Autonomous exploration of subterranean environments constitutes a major frontier for robotic systems as underground settings present key challenges that can render robot autonomy hard to achieve. This has motivated the DARPA Subterranean Challenge, where teams of robots search for objects of interest in various underground environments. In response, the CERBERUS system-of-systems is presented as a unified strategy towards subterranean exploration using legged and flying robots. As primary robots, ANYmal quadruped systems are deployed considering their endurance and potential to traverse challenging terrain. For aerial robots, both conventional and collision-tolerant multirotors are utilized to explore spaces too narrow or otherwise unreachable by ground systems. Anticipating degraded sensing conditions, a complementary multi-modal sensor fusion approach utilizing camera, LiDAR, and inertial data for resilient robot pose estimation is proposed. Individual robot pose estimates are refined by a centralized multi-robot map optimization approach to improve the reported location accuracy of detected objects of interest in the DARPA-defined

*[1] Robotic Systems Lab, ETH Zurich, [2] Autonomous Robots Lab, University of Nevada, Reno — Norwegian University of Science and Technology, [3] Autonomous Systems Lab, ETH Zurich, [4] Flyability, 5 Oxford Robotics Institute, University of Oxford [6] University of California, Berkeley, [7] Sierra Nevada Corporation, Direct correspondence to Marco Tranzatto `marcot@ethz.ch`

coordinate frame. Furthermore, a unified exploration path planning policy is presented to facilitate the autonomous operation of both legged and aerial robots in complex underground networks. Finally, to enable communication between the robots and the base station, CERBERUS utilizes a ground rover with a high-gain antenna and an optical fiber connection to the base station, alongside breadcrumbing of wireless nodes by our legged robots. We report results from the CERBERUS system-of-systems deployment at the DARPA Subterranean Challenge Tunnel and Urban Circuits, along with the current limitations and the lessons learned for the benefit of the community.

1 Introduction

This paper reports the technological progress made by team "CERBERUS" participating in the DARPA Subterranean Challenge (SubT Challenge). The SubT Challenge is an international robotics competition organized and coordinated by the Defense Advanced Research Projects Agency (DARPA) and calls on teams to compete with respect to exploring, mapping and searching large-scale underground environments such as tunnels or mines, urban subterranean infrastructure and cave networks. Responding to the requirements of this challenge, this paper outlines the vision and technological developments of team CERBERUS, short for "CollaborativE walking & flying RoBots for autonomous ExploRation in Underground Settings". Team CERBERUS corresponds to a combined system of legged and aerial robots equipped with multi-modal perception capabilities, localization and mapping autonomy, as well as robot-deployable communications that enables resilient navigation, exploration, mapping and object search inside complex, large-scale, sensor-degraded and communications-denied subterranean environments. CERBERUS is developed based on an international partnership, which up to the time of writing involves the Autonomous Robots Lab (University of Nevada, Reno), Robotic Systems Lab (ETH Zurich), Autonomous Systems Lab (ETH Zurich), HiPeR Lab (University of California, Berkeley), the Dynamic Robot Systems Group (University of Oxford), Flyability and Sierra Nevada Corporation. This paper presents both the technological elements of CERBERUS and detailed results of our team's performance in the Tunnel and Urban Circuits, which were held in an underground mine and a never-commissioned nuclear facility, respectively.

During the Tunnel and Urban Circuits, CERBERUS' core robots were the "ANYbotics ANYmal B" quadruped, and two classes of "Aerial Scouts", namely multirotors with conventional airframes, as well as the "Gagarin" class of collision-tolerant aerial robots. We have further added a roving robot called "Armadillo". ANYmal has the most extended endurance compared to all our robots and integrates our multi-modal perception and autonomy solutions. At the same time, ANYmal can also deploy wireless communication nodes. The Aerial Scout, with conventional airframe, is a traditional quadrotor design and acts as a rapid explorer. The Gagarin collision-tolerant Aerial Scout is of smaller size and its mechanical structure enables it to survive contact with surfaces and to make its way through challenging, narrow, multi-branched, and possibly multi-level underground environments. Both aerial platforms integrate the CERBERUS multi-modal perception and autonomy pipelines and operate without any human supervision after take-off. Lastly, the Armadillo roving platform primarily deploys a high-gain wireless communications antenna deeper in the underground environment, while maintaining an optical fiber connection to the Base Station. As a fall-back role, this roving robot also integrates multi-modal perception capabilities to allow it to detect artifacts. Figure 1 presents the CERBERUS robots deployed at the Tunnel and Urban Circuits of the SubT Challenge.

To enable autonomous exploration of subterranean environments, team CERBERUS focuses on multi-modal and multi-robot perception, as well as path planning for exploration and area search. A unified complementary onboard perception solution is developed that fuses LiDAR, cameras, and inertial sensor cues. This enables resilient localization and situational awareness even in conditions of perceptual degradation such as geometric self-similarity, darkness and the presence of obscurants. The individual multi-modal perception capabilities are further exploited at the level of multi-robot collaboration with map sharing and alignment across the robotic systems of CERBERUS using centralized map optimization at the Base Station. In brief, each robot can detect objects of interest in the field, and with map optimization, refine their position esti-

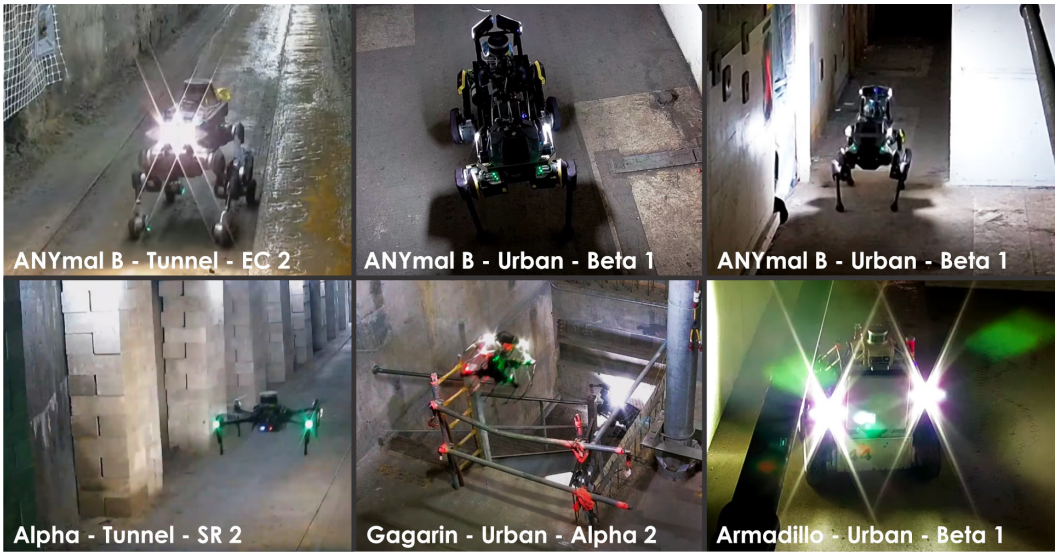


Figure 1: Instances of deployments of the CERBERUS robotic system-of-systems in the DARPA Subterranean Challenge Tunnel and Urban Circuits. Upper row: ANYmal B on wheels and two standard ANYmal B robots exploring sections of the Tunnel Circuit's "Experimental" Course and Urban Circuit's "Alpha" and "Beta" Courses. Lower row: the Alpha and Gagarin Aerial Scouts in the Tunnel Circuit's "Safety Research" Course and the Urban Circuit's "Alpha" Course respectively, alongside the Armadillo rover in the Urban Circuit's "Beta" Course. Relevant video results are available at `http://bit.ly/cerberus2020`

mates back at the Base Station. Building on top of this core capacity, CERBERUS uses a unified exploration path planning solution on both legged and aerial robotic systems. The method relies on a graph-based search and an information gain formulation related to volumetric exploration, avoids collisions, and accounts for traversability limitations. While it enables efficient local search for optimized paths, it also enables the ability for automated homing and to travel to frontiers of the exploration space detected at earlier steps of the robot's mission. It is thus tailored to the large-scale, multi-level, and often narrow character of underground environments.

As CERBERUS develops autonomous exploration and situational awareness through a team of robotic systems overseen by a Human Supervisor located outside of the underground environment, the final major development priority relates to a robot-deployable communications solution. Maintaining a connection with sufficient bandwidth as the robots proceed deeper into the subterranean environment is challenging due to the size of the environments, along with their geometric and material properties. In response to this challenge, CERBERUS utilizes customized 5.8GHz WiFi modules with each robot integrating its own antennas, the ANYmal robots ferrying and being able to deploy communication breadcrumbs, and the Armadillo rover integrating a high-gain antenna connected to the Base Station. This network allows for bidirectional communication between the robots and to the Base Station, enabling both high-level control of the robot behavior by the supervisor, as well as the reporting of detected objects of interest.

Alongside detailing the technological elements, we present how the robots operated in the underground mine environments of the Tunnel Circuit and the abandoned nuclear facility of the Urban Circuit and discuss technological adjustments in our solution. We compare our mapping results with ground-truth provided by DARPA, specifically present the results of both the onboard and the multi-robot SLAM solutions, detail and discuss the performance of our autonomy pipeline and when the single-operator exerted control, and finally report the detected and missed objects of interest. Finally, we present an assessment of our team

performance both at the system level and for each technological component and focus on the challenges faced and the lessons learned with the hope of informing other researchers contributing to this exciting field. As significant components of our research have been released as open-source contributions, a selected list of relevant software packages is also provided.

The remainder of this paper is structured as follows: Section 2 presents related work, while a high-level overview of the CERBERUS vision is outlined in Section 3. The details about the individual robotic systems are presented in Section 4. The two autonomy-enabling functionalities for exploration (path planning and multi-modal multi-robot perception) are presented in Sections 5 and 6 respectively, followed by our object detection and reporting solution in Section 7. The robot-deployable communications solution is outlined in Section 8, while the Human Supervisor interface is discussed in Section 9. Experimental results primarily from the DARPA Subterranean Challenge field deployments are presented in Section 10, followed by lessons learned in Section 11. Finally, we present our plans for future work in Section 12, followed by conclusions in Section 13. An appendix is also provided containing links to selected open-source contributions of our team, alongside links from deployment results at the Tunnel and Urban Circuit events.

2 Related Work

The domain of subterranean exploration has attracted the interest of a niche community. The works in (Silver et al., 2006; Baker et al., 2004) present methods for topological exploration of underground environments based on edge exploration and the detection of intersections. This work has been evaluated using ground platforms with the Groundhog system demonstrating pioneering levels of exploration capacity. The contribution in (Rogers III et al., 2017) presents methods for underground exploration using a team of UAVs. The works in (Mansouri et al., 2019a; Mansouri et al., 2019b) outline methods for underground robotic exploration - using aerial robots - via contour and junction detection. Similarly, the work in (Kanellakis et al., 2019) utilizes methods based on open space attraction. The paper in (Jacobson et al., 2020) focuses on the localization challenges in subterranean settings. From a systems perspective, the set of works in (Morris et al., 2006; Novák et al., 2015; Maity et al., 2013) overview innovations in the domain of ground and submersible subterranean robotics. An overview of the utilization of ground robotic systems in underground environments is provided in (Tardioli et al., 2019). Furthermore, the methods in (Bakambu and Polotski, 2007; Lösch et al., 2018) outline navigation solutions for robotic rovers underground. It should be noted that the research challenges faced in such robotic missions relate to a breadth of research in the domains of perception, planning, control, communications and more. This is reflected in the relevance of the challenges faced in other competitions, including - but not limited to - the earlier "Multi Autonomous Ground-robotic International Challenge (MAGIC)" as reported in papers from several of the participating teams (Olson et al., 2012; Lacaze et al., 2012; Boeing et al., 2012; Butzke et al., 2012).

Accelerated by the kick-off of the DARPA SubT Challenge in September 2018, the domain of underground robotic autonomy is currently experiencing rapid growth. The works in (Palieri et al., 2020) and (Tabib and Michael, 2019) present accurate multi-sensor techniques for subterranean localization and mapping, while the work in (Bouman et al., 2020) presents progress in the ability of legged robots to search such environments autonomously. At the same time, the contribution in (Miller et al., 2020) uses multiple legged systems, while a relevant dataset of artifact classification is released by the same team (Shivakumar et al., 2020). The work in (Steindl et al., 2020) focuses on a hexapod design that has benefits in negotiating the challenging terrain often encountered in subterranean applications. The work in (Hines et al., 2021) takes a different approach using virtual surfaces to propose a cohesive mapping and planning solution for ground robots. Aiming to explore scenes of interest better, the contribution in (Wang et al., 2020) helps in identifying areas of visual interest through online learning, while the work in (Carlbaum et al., 2020) proposes a deep Convolutional Neural Network (CNN) architecture to establish a novel feature tracking scheme for robust localization. Focusing on the multi-robot operations aspect of underground autonomy, the authors in (Rouček et al., 2019; Zoula et al., 2021; Goel et al., 2021; Corah et al., 2019) present methods for multi-robot exploration, map sharing and communications. Contributing a largely different idea, the work in (Huang et al., 2019) proposes

the use of blimp system for long-endurance in underground exploration missions. Our team contributions are aligned with these community efforts and aim to support the development of subterranean robotic autonomy.

3 CERBERUS System-of-Systems

The Defense Advanced Research Projects Agency (DARPA) created the DARPA Subterranean Challenge (SubT Challenge) to incentivize research on novel approaches to explore, navigate, map, and search underground environments with teams of robots. The underground domain features key challenges for mobile autonomous systems including dynamic and rough terrain, limited visibility, poor communication, and hard access. Extensive research in the fields of autonomy, perception, network, and mobility is therefore needed to allow robots to explore these settings autonomously.

The SubT Challenge is structured in two parallel tracks, namely the "Systems" and the "Virtual" track. In the former, teams choose to develop their own hardware and software to compete in a physical course. In contrast, in the latter, teams choose to develop software-only solutions that are evaluated in simulation. The proposed underground scenarios for the Systems track include man-made tunnel networks extending for several kilometers and presenting vertical openings ("Tunnel Circuit" - August 2019), multi-level urban underground structures with complex layouts ("Urban Circuit" - February 2020), and natural cave environments with complex morphology and constrained passages ("Cave Circuit" - August 2020, cancelled event). Finally, the challenges of the previous circuits are combined in a comprehensive benchmark mission ("Final Event" - September 2021).

The primary goal is to provide rapid situational awareness to small teams preparing to enter unknown and dynamic subterranean environments. Therefore the competing teams are requested to explore and map unknown underground scenarios, with the additional requirement that only one team member, the Human Supervisor, can manage and interact with the deployed systems. Teams are scored based on their ability to localize some specified objects, called "artifacts", that represent features or items of interest that could be reasonably found in subterranean contexts. The classes of the artifacts are known prior to each Circuit: *Survivor*, *Cell Phone*, and *Backpacks* are common objects for all the Events; *Drill* and *Fire Extinguisher* are specific to the Tunnel Circuit whereas *Gas* and *Vent* can be found at the Urban Circuit. A total of 20 such objects are distributed along the Competition Course. Their positions are initially unknown to the teams. If an artifact is localized and its position is reported within 5 m of its surveyed position, the team scores one point. The artifacts' surveyed positions are expressed with respect to a DARPA-defined coordinate frame.

The Tunnel and the Urban Circuit - which relate to the focus of this paper - each featured two different mission courses in their respective environments. All teams were given two "scored runs" per course, scheduled on different days. The total final score for each Circuit was computed as the sum of a team's best run on each course. Before the beginning of each run, the team's Pit Crew (composed of up to nine people) had 30 minutes to complete the necessary setup of their Base Station and of the robotic systems before the actual mission would start. The Base Station is a collection of one or more PCs used by the Human Supervisor to interact with the deployed robotics agents. Moreover, it serves as an interface for forwarding artifact reports and map updates to the DARPA Command Post, which evaluates the validity of the reports and provides score updates. Each scored run lasted for 60 minutes and was the only time available for the teams to explore and report the artifacts' positions.

Motivated by and responding to these challenges, the CERBERUS collaborative walking and flying robotic system-of-systems envisions the autonomous exploration of subterranean environments through the synergistic operation of diverse systems with unique navigation capabilities. The principal idea behind CERBERUS' vision is that the combination of legged and aerial systems offers unique advantages tailored to the needs of complex underground settings. Walking robots present potential benefits for overcoming complex terrain, both in terms of the size of obstacles that can be negotiated for a certain overall robot size, and robust traversal of dynamic terrain. Aerial robots, especially when equipped with collision-tolerance capabilities,

offer the advantage of seamless navigation that is not bound to the extreme terrain challenges often present in environments such as caves. CERBERUS focuses on the co-deployment of such legged and aerial robots that are further equipped with resilient multi-modal and multi-robot localization and mapping, artifact detection and universal exploration path planning capabilities. A robot-deployable communications network solution is being realized which allows the CERBERUS system-of-systems to set up its own communications network underground. Figure 2 depicts an illustration of a CERBERUS deployment. At the time of writing, a subset of the technological components, methods and solutions have been developed and deployed incrementally at the Tunnel and Urban Circuit events of the competition. The highlights of these developments are presented in the rest of the paper. An appendix is provided that lists the open-source repositories containing our research to facilitate its reproducibility and verifiability, which in turn can help accelerate the developments of the community as a whole.

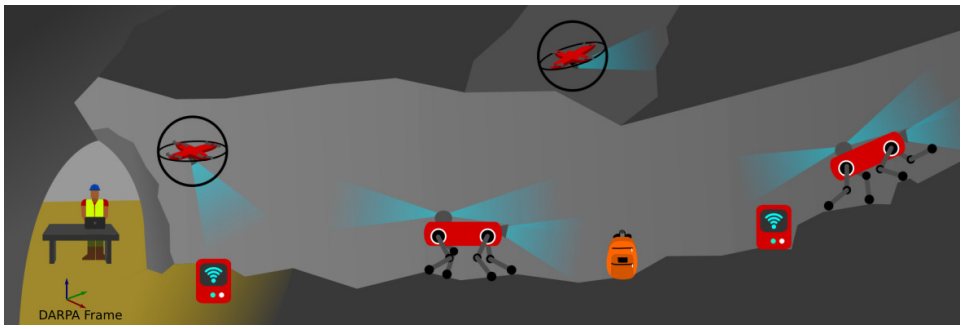


Figure 2: Graphical overview of the CERBERUS system: A single Human Supervisor outside the facility orchestrates the deployment of a heterogeneous team of quadrupeds and aerial robots. Our ANYmal quadrupeds can traverse mobility hazards, while flying systems and especially the collision-tolerant robots can act as scouts or used to detect artifacts in chimneys and shafts. ANYmal deploys communication nodes to extend communication into the tunnel. The robots collaborate to reconstruct a consistent map, while artifacts detected are reported in a global frame defined by DARPA.

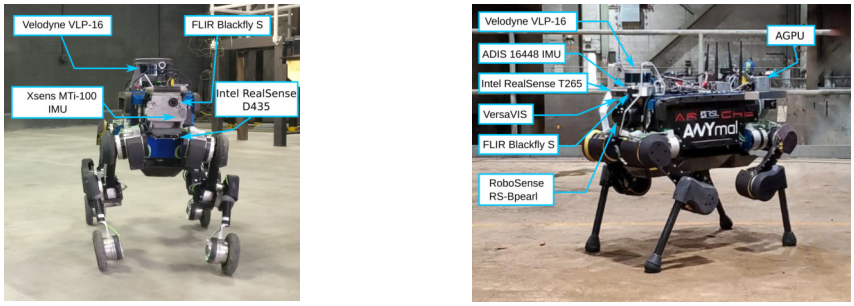
4 CERBERUS Robots

This section outlines the CERBERUS robots, their system design and specifics with respect to subterranean exploration functionality.

4.1 Subterranean Walking Robots - ANYmal B SubT

Team CERBERUS developed and deployed a team of legged robots specialized to operate in underground environments. These robots are based on the ANYbotics ANYmal B300 (Hutter et al., 2016) series quadruped, and feature several hardware and software modifications to create a specialized subterranean version: the ANYmal B "SubT" platform.

The ANYbotics ANYmal B300 series quadrupedal robot is a robust and agile legged system that is tailored for autonomous, long endurance tasks under challenging environments. Each leg has three compliant joints with integrated electronics which provide torque controllability and resilience against impact. This series of quadrupeds have been developed for over 10 years and have achieved significant robustness, dexterity and mobility, while maintaining a small size and weight with an endurance of 1 to 2 hours. In contrast to wheeled or tracked systems, legged robots have the potential to adapt to terrain challenges by stepping over obstacles (Jenelten et al., 2020) or adjusting their posture to crawl under obstacles (Buchanan et al.,



(a) Tunnel Circuit (b) Urban Circuit

Figure 3: ANYmal B SubT robot with specialized hardware and computation payload for deployment in underground settings.

2021). They also allow for omnidirectional locomotion, which is advantageous inside narrow environments. Additionally, legged machines can be combined with wheels to achieve high speed and energy efficiency, combining the best of both worlds (Bjelonic et al., 2020). For this reason, legged robots are increasingly used in industrial inspection applications (Bellicoso et al., 2018) and in the underground domain (Kolvenbach et al., 2020). Consequently, other DARPA Subterranean Challenge competitors also deployed quadrupedal robots during the Tunnel and Urban Circuits. For example, team *PLUTO* deployed four *Ghost Robotics Vision 60* during the Tunnel Circuit (Miller et al., 2020), and team *CoSTAR*, which used wheeled systems during the Tunnel Circuit, later deployed two *Boston Dynamics Spot* robots (Bouman et al., 2020).

Hardware Modifications
Sensors - For the Urban Circuit, the onboard sensors were placed in two main locations atop the robot. A front facing aluminum case included an Intel RealSense T265 Camera, a FLIR Blackfly S color camera, an IMU, a VersaVIS synchronization board (Tschopp et al., 2020), a Velodyne Puck VLP 16 LiDAR, and a Robosense BPearl LiDAR. A rear 3D-printed housing enclosed a Sensirion SDC30 gas sensor and two FLIR Blackfly S cameras. The location of each component on the robot can be seen in Figures 3 and Figure 4. All components in the front aluminum case are protected by a sheet metal structure. In order to have a minimally obstructed field of view, the metal around both LiDARs is parallel to the pulsed laser beams of the respective LiDAR. This setup is an improved version of the one deployed for the Tunnel Circuit, where an Intel RealSense D435 Camera was used instead of the Robosense BPearl LiDAR.

Terrain Mapping Sensors - It is essential to ensure accurate terrain perception for safe navigation and locomotion during autonomous missions. Therefore terrain mapping sensors should be accurate and reliable in challenging environmental features such as darkness or reflective surfaces. Three different sensors were compared: the Intel RealSense D435, the Pico Monstar and the Robosense BPearl. The RealSense is an active stereo sensor that uses stereo matching with two IR cameras and complements it with active depth sensing based on the projection of an IR pattern. The Pico Monstar is a time-of-flight camera that is devised for long ranges and a wide Field-Of-View (FOV). The Bpearl is a dome-LiDAR that measures the reflection times of emitted laser pulses using a hemispherical scanning pattern. An overview of the comparison results is given in Table 1. The Pico Monstar and the RealSense were found to be prone to reflections (showing missing data on wet ground) and other conditions of the surrounding environments (puddles, absorbent material). Therefore the Bpearl was chosen as the terrain mapping sensor unit for the Urban Circuit, given its accuracy and robustness in various conditions and its large 90° FOV.

Parallel Computing - The terrain mapping algorithms can be efficiently parallelized using a graphics processing unit (GPU), while visual object detection also benefits from running neural-networks on a GPU. A Jetson AGX Xavier was installed as an Auxiliary Graphics Processing Unit (AGPU). The unit is enclosed in a 3D-printed PA-6 housing for protection against dust and water. Figure 4 shows the content of the AGPU.

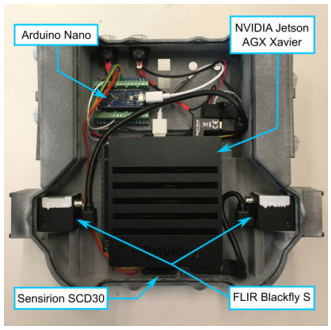


Figure 4: Auxiliary Graphics Processing Unit (AGPU) package enclosing a Jetson AGX Xavier, an Arduino Nano microcontroller, a Sensirion SDC30 gas sensor and two FLIR Blackfly S cameras.

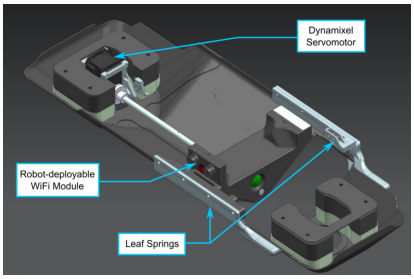
Foot variants - To augment terrain specific locomotion capabilities, the legged robots were equipped with three different types of feet. The standard version of ANYmal comes with point feet which adapt to any inclination angle but are prone to slip. The flat feet developed in (Valsecchi et al., 2020) have a larger contact surface and therefore provide more robust traction. To clear a considerable distance in a short time, ANYmal was also equipped with actuated wheels (Bjelonic et al., 2020).

Payload Release Mechanism - The original belly plate of the robot, which protects the main body from impacts, was modified to carry and release specialized WiFi Breadcrumbs (described in Sec. 8.3), as depicted in Figure 5. A Dynamixel servomotor drives a slide screw which converts the rotary motion into linear movement of a pusher plate. The modules are pushed towards an opening in the belly plate where leaf springs prevent accidental release.

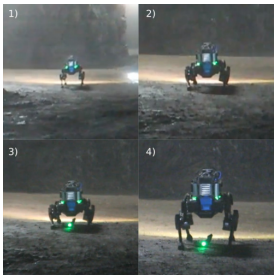
Software Modifications
Locomotion - Two different locomotion controllers were deployed on the ANYmal robots. During the Tunnel Circuit, a model-based perceptive controller was utilized (Jenelten et al., 2020). The footholds and base motion are sequentially optimized using a simplified description of the robot dynamics and while accounting for the local terrain properties in a batch search. Additionally, a slip reflex (Jenelten et al., 2019) stabilizes the robot in case of major slip events. In the Urban Circuit, a neural-network controller was deployed. It was trained in simulation using reinforcement learning (Lee et al., 2020). The controller relies solely on proprioceptive sensor information and uses a history of joint and IMU measurements to implicitly infer information about the terrain and external disturbances without depending on the local terrain map. While the model-based controller performs motion planning with anticipation of terrain changes, the reinforcement learning-based controller has significantly increased locomotion robustness in various challenging environments. During the Tunnel Circuit, ANYmal experienced corrupted local terrain representations leading to incorrect locomotion controller actions that prevented the robot from continuing the mission. Therefore at

Sensor	**Type**	**Note**	**Size** (mm)	**Weight** (g)
RealSense	Active Stereo	Less accurate but functional under sunlight	90x25x25	72
Pico Monstar	Time-of-Flight	Accurate but weak to sunlight or absorbent material	66x62x29	142
BPearl	LiDAR	Accurate, robust against sunlight or absorbent material but sparse	100x111x100	920

Table 1: Comparison of sensors for terrain mapping.



(a) Release mechanism.



(b) WiFi beacon deployment.

Figure 5: ANYmal's payload release mechanism and one deployable module (a). ANYmal deploying a WiFi beacon on the ground (b).

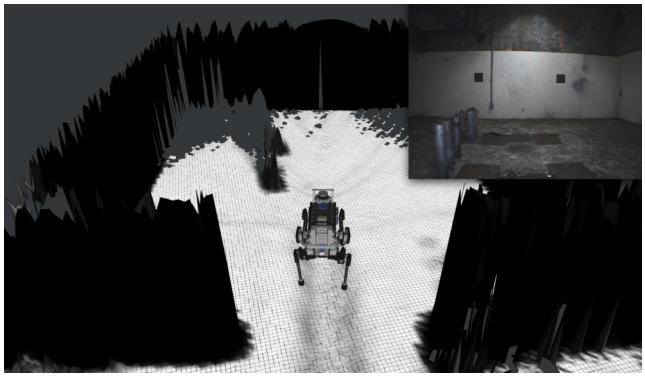


Figure 6: Elevation map built by ANYmal during the Urban Circuit. Black cells indicate non-traversable regions as classified by the learning-based traversability estimation.

the Urban Circuit the learning-based locomotion controller was employed due to its robust performance, as well as its characteristic ability to plan footholds without relying on an explicit representation of the local terrain.

Terrain Perception - As legged robots are constrained to move on the ground plane, a terrain representation in the form of a 2.5D elevation map is commonly used. It is built in a robot-centric fashion from depth sensor readings and robot odometry (Fankhauser et al., 2014). The map has dimensions equal to 12x12 m, with a resolution of 4 cm. These settings reflect the sensor limitations and represent a trade-off between capturing the surface of the terrain accurately and keeping the computation cost to a reasonable level for onboard processing. In addition to the height value, a traversability value is computed for each grid cell. The onboard state machine uses this information to prevent the robot walking into "non-traversable" areas. Traversability values are computed with a two-layer CNN where the first layer applies multiple learnable traversability templates to the height map and the second layer computes a weighted sum of template activations (Wellhausen and Hutter, 2021). Since this network has only 120 learnable parameters, it can be trained using only a dozen hand-labeled elevation maps and allows for rapid inference as it operates directly on the elevation maps in GPU memory. A visual representation of the elevation map with associated traversability estimation is shown in Figure 6.

Path Tracking - The path tracking module accepts paths from the autonomy module that are created either by the the high-level exploration planner or by the Human Supervisor. This module is organized in a hierarchical manner: a path manager converts the received paths (defined in an inertial frame) into a drifting

frame in which the robot's locomotion controller operates. Subsequently it triggers the path following module which computes twist commands for the robot to follow the given path.

State Machine - The capabilities for coordinating different software modules to execute autonomous missions are provided by a multi-faceted state machine, whose interaction with the other main modules is depicted in Figure 7. Its primary purpose is to minimize Human Supervisor interactions by automatically coordinating between a high-level exploration planner, providing way-points where the robot should go, and a path tracking controller that steers the agent to the desired location. The state machine features a safety layer that monitors and responds to events such as non-traversable terrain, unexpected obstacles or loss of communication. This software module operates in two different modes depending on the operator's requirements: (a) supervised-autonomous and (b) fully-autonomous mode. The former case is used when the operator has a data connection to the robot, can check the high-level exploration planner's goals, and can decide to overrule the given way-points. During this mode, if communication to the Base Station is unexpectedly lost, the robot is commanded to trace back its path until a connection is re-established and subsequently wait for the supervisor to make a decision. The latter mode is used to explore regions without WiFi coverage. In this case, the state machine does not take any action in case communication to the Base Station is lost. This mode can be activated by the operator, alongside setting a time limit for autonomous exploration. Once this time has elapsed, a homing request to the high-level exploration planner is issued, which will compute a trajectory back to the Base Station. It is critical to get the robot back into the region with WiFi coverage because the generated map and the identified artifacts need to be streamed back to the operator in order to score points.

Despite the current operating mode of the state machine, if obstacles are detected along the robot's direction of motion, a traversability safety module is triggered. This module ensures that the robot reverses a fixed distance and informs the high-level exploration planner about the non-traversable area (called a "geofence zone"). The exploration planner will then re-plan a new path that avoids the obstacle. The state machine uses the current robot's direction to check if the agent is moving towards a non-traversable area. The direction of motion is identified by performing a look-ahead projection of the robot's future poses (up to 30 cm ahead) using its estimated velocity. Non-traversable obstacles are identified with the help of the traversability estimation layer of the elevation map. The estimated look-ahead poses are used to anticipate obstacles and to stop the robot sufficiently early. The state machine checks for the current robot's direction of motion instead of the high-level exploration planner's path for a set of reasons. Firstly, the robot can react to previously undetected non-traversable regions. Secondly, the portion of the elevation map closer to the agent has higher accuracy. Lastly, checking the traversability of a path of several meters in a high resolution map is computationally expensive. Therefore, checking for a relatively short range in front of the robot is preferred.

4.2 Subterranean Aerial Robots

The SubT Challenge provides several unique challenges to autonomous robotic operation. Mines, caves, and tunnels frequently contain many challenging terrain features and multi-level settings. Terrain obstacles, including stairs, shafts both vertical and horizontal, boulders, cliffs, and bodies of water all impose significant hazards to ground robots. Simultaneously, the elevated perspective inherent to aerial platforms provides additional capabilities to observe, localize, and report artifacts. Despite limited endurance, aerial platforms can cover more area in less time in many environments. Furthermore, they may correspond to the only option for certain subsets of subterranean settings. Therefore, CERBERUS developed and deployed a team of "Aerial Scouts" to provide rapid exploration capabilities, especially in regions of the course which were inaccessible by ground robots.

System Design: Three robots, named Alpha, Bravo, and Charlie, utilize the DJI Matrice 100 quadrotor as their airframe. For high-level processing, including sensor data collection, localization, mapping, and planning, an Intel NUC Core-i7 computer (NUC7i7BNH) is mounted on each robot, while an Intel Movidius Neural Compute Stick is used for artifact detection. As shown in Table 2, all three robots fuse LiDAR, visual

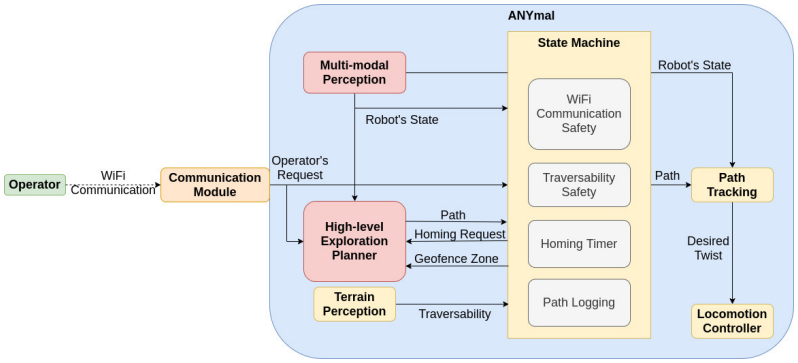


Figure 7: Schematic system overview of the main software modules on ANYmal, specifically tailored for underground exploration missions. The components in the blue shaded area are executed on the onboard computer. The yellow rectangles are ANYmal specific modules, whereas red rectangles indicate software modules used also by the other CERBERUS' robots.

and inertial data, while Charlie further integrates thermal vision and thus provides unique localization capabilities in visually-degraded environments, as well as thermal object detection (e.g., for the survivor artifact class). To enable perception in low-light environments, all three M100 Aerial Scouts are equipped with Cree LEDs, which are synchronized with their visible light camera's shutter producing a power-efficient flashing behavior.

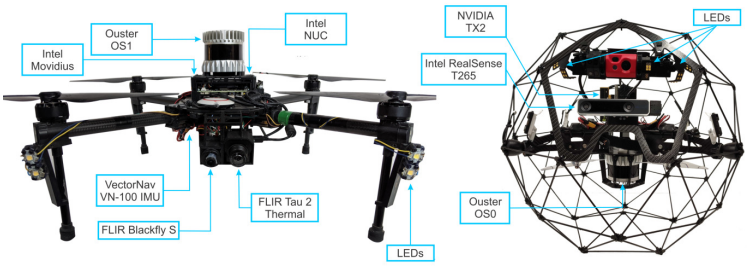


Figure 8: The Charlie (left) and Gagarin (right) Aerial Scouts.

A second - and particularly important for our team - class of Aerial Scouts, named Gagarin, also provides a number of unique capabilities. This robot is designed to traverse narrow and visually degraded environments. Gagarin is a drone surrounded by a collision tolerant cage that allows it to sustain collisions with obstacles

	Alpha / Bravo	Charlie	Gagarin
Platform	DJI M100	DJI M100	Collision-tolerant platform 1-3
Camera	FLIR Blackfly S	FLIR Blackfly S	Intel RealSense T265
Thermal	N/A	FLIR Tau2	N/A
LiDAR	Velodyne Puck LITE	Ouster OS1	Ouster OS0 / OS1
IMU	VectorNav VN-100	VectorNav VN-100	Integrated in autopilot

Table 2: Sensor suites used in different Aerial Scouts.

at speeds exceeding 2m/s. In order to thrive in narrow spaces, this drone has a compact design with a tightly integrated power-efficient NVIDIA TX2 board with 6 ARM CPU-cores and a 256-core NVIDIA Pascal GPU, an Ouster OS0 or OS1 (depending on the version of the Gagarin robot) LiDAR having a wide vertical FOV, an IMU and two fisheye cameras, rendering it suitable for going through staircases and vertical shafts. Additionally, to traverse visually-degraded environments, Gagarin has a custom lighting system involving LEDs at different angles designed to reduce the effect of dust on the image, increasing the visual range and enhancing the environment texture.

All Aerial Scouts communicate with the Base Station using their onboard 5.8GHz WiFi radios to connect directly to the Base Station antenna or through the breadcrumbs deployed by our legged robots. Importantly, as reliable high-bandwidth communication becomes increasingly infeasible in most underground environments, the Aerial Scouts are able to operate completely autonomously with little to no input from the Human Supervisor, and return to their home position before their batteries become depleted. The M100 Aerial Scouts are capable of flight times of 8 to 10 minutes, while the Gagarin collision-tolerant aerial scouts present an endurance of 5 to 6 minutes.

Software Architecture: The Aerial Scouts all share the same software architecture, whose basic functionalities are illustrated in Figure 9. Similar to other robots in the CERBERUS team, the Aerial Scouts use a high-level exploration path planner, described in Section 5, to plan feasible paths for the robots. The path is then tracked by a linear model predictive controller as in (Kamel et al., 2017) and the low-level command is executed by the onboard autopilot inside each robot. The odometry feedback of the robots and the map of the environment are provided by the multi-modal perception module presented in Section 6.

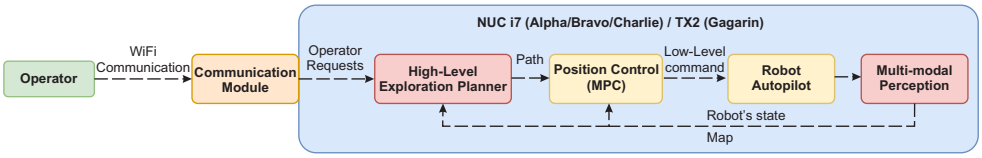


Figure 9: Schematic system overview of the core functional modules of the aerial scouts. All the components inside the blue shaded area are executed onboard the robot. Yellow color depicts robot-specific functionalities, while red is used for software components that are unified across the CERBERUS robots.

4.3 Roving Robots

Alongside the legged and aerial platforms, team CERBERUS also brought a roving robot called Armadillo into the field. This ground vehicle is complementary to the walking and flying systems presented above.

System Design: Armadillo is based on the Inspectorbots Super Mega Bot (SMB) robot platform and was chosen to complement the performance of the core CERBERUS team, primarily as a means to deploy a high-gain communications antenna and due to its capability to carry large payloads and sensors. A custom-designed frame, made of lightweight aluminum extrusions, was used to house all the components on the robot. This frame was fixed atop Armadillo's chassis and an onboard computer, namely a Zotac VR GO Backpack PC, was placed inside the protected cage-like structure. A sensor-head, attached at the highest part of the frame, houses various sensors to provide an unobstructed view of the environment.

The sensor-head on the Armadillo consists of a two level design having mounting points for all the sensors of the robot. A FLIR Boson LWIR Camera and a FLIR Blackfly Color camera are mounted in the front facing configuration alongside a VectorNav VN-100 IMU. Two identical FLIR Blackfly S cameras are mounted on the sides of the sensor looking at either side of the robot. Additional FLIR Boson and Blackfly cameras are mounted in a top-facing configuration for detecting artifacts on the ceiling.

A frame extension on the right side of the robot is made to mount a Robosense BPearl LiDAR Sensor that

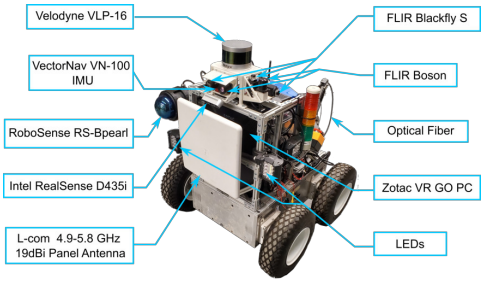


Figure 10: The Armadillo roving robot - a modified Inspectorbots Super Mega Bot.

provides a dome-like view of the environment. At the back of the robot, a 300m-long optical fiber reel is mounted with the capability of automatic unrolling while moving. This allows wired connectivity from the Base Station to the robot. An additional 5.8GHz directional antenna is mounted in the front of Armadillo, which allows the robot to act as a network range extender, providing high-bandwidth connectivity to the Base Station.

Software Architecture: The Armadillo shared the same perception architecture as the flying robots. However one significant difference is that - given its constant connectivity with the base-station through the optical fiber connection and design choices of our team - it was manually controlled by the Human Supervisor. The front facing Blackfly camera, the IMU and the Velodyne LiDAR are used for the multimodal perception as described in Section 6. The remaining sensors provide a live feed to the operator at the Base Station.

5 Exploration Autonomy

The CERBERUS system-of-systems performs autonomous exploration and search for artifacts in the subterranean environment based on a Graph-based exploration path planner (GBPlanner) (Dang et al., 2020) that is universally designed and deployed across both legged and aerial robots. The method exploits a volumetric representation of the incrementally updated and explored 3D occupancy map $\mathbb{M}$ reconstructed based on measurements from a set of onboard depth sensors $\{\mathbb{S}\}$, as well as robot poses derived from each robot localization system $\{\mathbb{O}\}$. In the Tunnel Circuit $\mathbb{M}$ was based on the work in (Hornung et al., 2013), while in the Urban Circuit the method in (Oleynikova et al., 2017) was employed. In both cases, $\mathbb{M}$ is using a fixed resolution, thus each voxel has an edge size r_V (throughout this work this value was set to $r_V = 0.2$m). The map consists of voxels m of three categories, namely $m \in \mathbb{M}_{free}$, $m \in \mathbb{M}_{occupied}$, or $m \in \mathbb{M}_{unknown}$ representing free, occupied, and unknown space respectively, while certain (generally disconnected) subsets of the map may correspond to "no-go" zones $\mathbb{M}_{NG}$ (hereafter referred to as "geofences") representing possible traversability constraints or other imposed limits. Let $d_{\max}$ be the effective range, and $[F_H, F_V]$ be the FOV in horizontal and vertical directions of each of the depth sensors $\mathbb{S}$. In addition, let the robot's configuration at time t be defined as the 3D position and heading $\xi_t = [x_t, y_t, z_t, \psi_t]$. Notably, since for most range sensors perception stops at surfaces, hollow spaces or narrow pockets cannot always be explored fully. This leads to a residual map $\mathbb{M}_{*,res} \subset \mathbb{M}_{unknown}$ with volume $V_{*,res}$ which is infeasible to explore given the robot's constraints. As a result, given a volume V_*, the potential volume to be explored is $V_{*,explored} = V_* \setminus V_{*,res}$.

Given this representation, the planning problem is organized based on a local and a global planning bifurcation. The overall planner architecture can be seen in Figure 11. In the local exploration planning mode, given the occupancy map $\mathbb{M}$ and a local subset of it $\mathbb{M}_L$ centered around the current robot configuration ξ_0, the goal is to find an admissible path $\sigma_L = \{\xi_i\}$ in order to guide the robot towards unmapped areas and thus maximize an exploration gain $\mathbf{\Gamma}_E^L(\sigma_i), \sigma_i \in \Sigma_L$(set of candidate paths), primarily relating to the volume expected to be mapped if the robot traverses along σ_L with a sensor $\mathbb{S}$. A path is admissible if it is

collision-free in the sense of not colliding with 3D obstacles in the map and not going through geofences.

Given the subspace $\mathbb{M}_L$ the local planning problem may report that no admissible solution is interesting for the robot (zero-gain or below a set threshold). When this event, called "local completion", occurs, the global planning mode uses the latest $\mathbb{M}$ representation and the current robot configuration ξ_0 to identify a collision-free path σ_G that leads the robot towards the frontiers of the unmapped areas. Feasible paths of this global problem take into account the remaining endurance of the robot. When the environment is explored completely ("global completion") or the battery approaches its limits, the method identifies an admissible path σ_H that returns the robot to its home position ξ_{home}. The functional specifics of these two modes are outlined below.

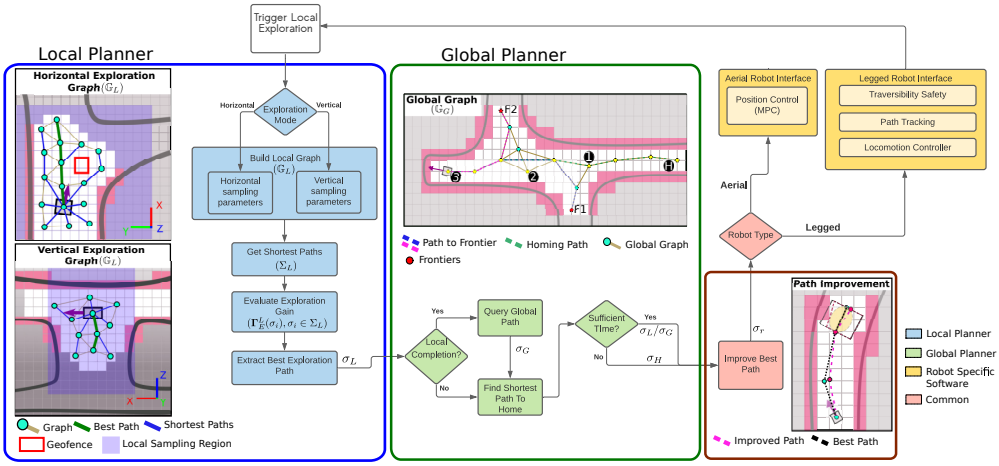


Figure 11: Functional diagram of the key functionalities of GBPlanner. This method bifurcates around a local and global planning mode with the first responsible for efficient exploration around the current robot location and the second responsible for re-positioning to frontiers of the explored volume in distant areas of the map, as well as auto-homing. Special functionality is provided for multi-level exploration, while the admissible paths account for collisions with 3D obstacles as well as "no-go" zones (geofences) that mark the non-traversable regions detected.

5.1 Multi-Level Local Exploration Path Planning

The local planning mode of GBPlanner first iteratively builds an undirected graph $\mathbb{G}_L$ inside the local map space $\mathbb{M}_L$ around the current robot configuration ξ_0, which will be used to derive candidate exploration paths. In the implementation of the planner, up to the Urban Circuit, the local map subset $\mathbb{M}_L$ had a fixed cuboid size set at the beginning of the mission (details for each robot are provided in Section 10). The graph $\mathbb{G}_L$ is built in the following manner. Given the updated occupancy map $\mathbb{M}$, a bounded "local" volume V_{D_L} with dimensions D_L centered around ξ_0 and the respective map subset $\mathbb{M}_L$ are first derived. Then considering a bounding box D_R representing the robot's physical size (alongside the respective map subset $\mathbb{M}_R(\xi_*)$ for a robot state ξ_*), the planner samples a set of random configurations ξ_{rand} inside V_{D_L} and after checking which local connections (within a defined radius δ) are collision-free and outside of the geofences, builds the graph $\mathbb{G}_L$ with its vertex and edge sets $\mathbb{V}, \mathbb{E}$ respectively. Subsequently, using Dijkstra's shortest path algorithm (Dijkstra et al., 1959), it derives paths Σ_L in the graph starting from the root vertex at ξ_0 and leading to all destination vertices. Given a path $\sigma_i \in \Sigma_L, i = 1...\mu$ with a set of vertices $\nu_j^i \in \sigma_i, j = 1...m_i$ along the path (corresponding to a set of robot ξ configurations), an information gain, called **VolumeGain**, relating to the new volume anticipated to be explored is computed for each vertex ν_j^i. To compute the **VolumeGain** at a vertex we count the number of unknown voxels inside a model of the sensor frustum (accounting for the sensor range and field-of-view) centered at the vertex. This is done by ray casting into

the map with rays starting at the vertex. A fixed resolution of ray casting is employed. The exploration gain $\mathbf{\Gamma}_E^L(\sigma_i)$ for the path σ_i is then calculated as follows:

$$\mathbf{\Gamma}_E^L(\sigma_i) = e^{-\zeta \mathcal{Z}(\sigma_i, \sigma_e)} \sum_{j=1}^{m_i} \mathbf{VolumeGain}(\nu_j^i) e^{-\delta \mathcal{D}(\nu_1^i, \nu_j^i)} \tag{1}$$

where $\zeta, \delta > 0$ are tunable factors, $\mathcal{D}(\nu_1^i, \nu_j^i)$ is the cumulative Euclidean distance from vertex ν_j^i to the root ν_1^i along the path σ_i, and $\mathcal{Z}(\sigma_i, \sigma_e)$ is a similarity distance metric between the planned path as compared to a pseudo straight path σ_e with the same length along the currently estimated exploration direction. The heading of every planned vertex is generally aligned with the direction of the respective path segment. While legged systems are allowed to turn in place to ensure that the desired heading is reached, the aerial robots are subject to a maximum yaw rate constraint. Therefore, a reachable heading as close as possible to that of the direction of the path segment is assigned given the length of the path segment, the fixed reference velocity, and the yaw rate constraint. Importantly, this local planning stage methodology implements a specialized sub-bifurcation in its architecture to handle the case of multi-level/multi-storey environments as, for example, encountered in the SubT Challenge Urban Circuit. In this mode, the method biases its sampling of the local graph $\mathbb{G}_L$ in order to sample densely in the vertical direction. Considering that a depth sensor unveiling relevant space is available, this offers an artificial reward of exploration gain when the robot changes "level" and thus identifies connections from the current level to the next. This motivates the robot to take a staircase or other vertical passageway connecting two storeys of the environment. In the implementation of the method as deployed up to the Urban Circuit event, it is the responsibility of the Human Supervisor to select the appropriate local exploration mode.

In the case of legged robots, the path should not only be obstacle-free but also traversable. To that end, an elevation map (see Section 4.1) is utilized to check for traversability along the robot's direction of motion, while following the planned path. The density of the elevation map is a trade-off between capturing the surface of the terrain accurately and keeping the computation cost to a reasonable level for onboard processing. Hence, the elevation map is calculated in a significantly smaller area than the size of the local map subset $\mathbb{M}_L$ which is used for exploration planning. However, planning only within short distances can lead to inefficient exploration behavior. Therefore, the exploration path is calculated based on the occupancy map (specifically $\mathbb{M}_L$), assuming that the areas outside the elevation map are traversable, and the traversability check is done during path execution. If an area where the robot is walking towards is found to be non-traversable, the path execution is stopped, that area is marked with a geofence, and the local planner is re-triggered. This process, however, can lead to sub-optimalities as it may cause the robot to stop and re-plan when an initially unknown non-traversable region is detected which in turn can slow down the exploration. To better address this problem, longer range elevation mapping would be essential which in turn poses a set of computational and sensing challenges as discussed in Section 4.1.

5.2 Global Exploration Path Planning

The global planning mode enables two critical behaviors in the exploration mission. First, upon local completion of unexplored areas, the global planning mode enables the re-positioning of the platform to unexplored areas of the map. Second, it permits autonomous homing upon reaching a critical point in the time budget of the robot. To execute these behaviors, the planner maintains a computationally lightweight global graph $\mathbb{G}_G$ that contains vertices which potentially lead to unexplored parts of the environment called "frontier vertices", or simply "frontiers". A vertex is marked as a frontier if the **VolumeGain** of that vertex is found to be larger than a set threshold. At each local planning step, the frontier vertices from the local graph $\mathbb{G}_L$ are extracted and the frontiers within a radius λ of any existing frontiers in the global graph are removed. The paths from the remaining frontier vertices to the root vertex of $\mathbb{G}_L$ are extracted, clustered together according to the Dynamic Time Warping (DTW) similarity metric (Bachrach, 2013; Dang et al., 2020) and the principal path corresponding to the longest path from each cluster is retained. Finally, all the principal paths are added to the global graph, as well as admissible edges connecting them to nearby global vertices. In order to maintain a reduced list of frontiers in $\mathbb{G}_G$, vertices which no longer offer high gain are

eliminated by periodically re-evaluating each frontier vertex. Additionally, at every local planning step, the planner needs to ensure that the robot has enough endurance to execute the path and return to the home location. Hence, utilizing Dijkstra's algorithm on the global graph, the planner identifies a return-to-home path at each local planner iteration, and commands the robot to follow this path upon reaching the time budget. It is noted that the homing path queries are fast due to the sparse nature of the global graph.

Choosing the best frontier to visit upon local completion is a difficult problem due to the size of the map over which the search is executed, the uncertain knowledge of these frontier vertices, as well as their true potential for exploration. Given these challenges, the selection of the frontier to visit depends on a) the time budget remaining upon reaching that frontier and b) the volumetric gain for that frontier given the time to reach it. Formally, let $\nu_{G,cur}$ be a vertex in the global graph representing the current robot configuration and $\mathcal{F} = \{\nu^{\mathcal{F}}_{G,i}\}, i = 1...m$ be the set of updated frontiers. A Global Exploration Gain $\mathbf{\Gamma}^{G}_{E}(\nu^{\mathcal{F}}_{G,i})$ is evaluated for each frontier as follows:

$$\mathbf{\Gamma}^{G}_{E}(\nu^{\mathcal{F}}_{G,i}) = \mathcal{T}(\nu_{G,cur}, \nu^{\mathcal{F}}_{G,i})\mathbf{VolumeGain}(\nu^{\mathcal{F}}_{G,i})e^{-\epsilon_{\mathcal{D}}\mathcal{D}(\nu_{G,cur},\nu^{\mathcal{F}}_{G,i})} \tag{2}$$

where $\mathcal{T}(\nu_{G,cur}, \nu^{\mathcal{F}}_{G,i})$ is the estimated exploration time remaining if the planner chooses to visit the frontier $\nu^{\mathcal{F}}_{G,i}$. This term is used to favor frontiers which lead to higher exploration times once they are reached. It is approximately calculated using the remaining exploration time $\mathrm{T}_e(t)$ (based on the robot's endurance or user-imposed mission time limit) at current instance t and the estimated time required to travel from the current vertex to that frontier, $\Upsilon(\nu_{G,cur}, \nu^{\mathcal{F}}_{G,i})$, and from there to the home location, $\Upsilon(\nu^{\mathcal{F}}_{G,i}, \nu_{G,home})$, as:

$$\mathcal{T}(\nu_{G,cur}, \nu^{\mathcal{F}}_{G,i}) = \mathrm{T}_e(t) - \Upsilon(\nu_{G,cur}, \nu^{\mathcal{F}}_{G,i}) - \Upsilon(\nu^{\mathcal{F}}_{G,i}, \nu_{G,home}) \tag{3}$$

The term $\mathcal{D}(\nu_{G,cur}, \nu^{\mathcal{F}}_{G,i})$ is the shortest path length from the current vertex to the frontier, which along with the tunable parameter $\epsilon_{\mathcal{D}}$ penalizes longer paths.

Finally, the exploration path, local or global, is further improved by modifying all vertices in the path to be further away from obstacles generating a safer path σ_r. The planner paths are provided to the robot-specific local planners and controllers which are responsible for their execution.

6 Multi-Modal Perception for Multi-Robot Mapping

To enable robot autonomy in challenging subterranean environments, onboard perception plays a crucial role in enabling reliable robot localization and navigation. Resilient robot perception, both in terms of sensors and methods, is of particular importance for operation during the SubT Challenge as, in addition to being GPS-denied, underground environments can impose various conditions of sensor degradation due to their structural nature and operational conditions. Complete darkness, low-texture, geometric self-similarity, thermal-flatness, and the presence of obscurants (smoke, fog, dust) are some of the challenging conditions that can be encountered during the competition. Relying on any single sensing modality is unsuitable for operation in such environments, and hence, the robots of team CERBERUS utilize multi-modal sensor data from visual/thermal cameras, LiDARs, and IMUs in a complementary manner to provide reliability, robustness, and redundancy for underground robot operations. This section details the processing of the multi-modal sensor data to estimate the robot state and create a globally consistent multi-robot map. First, sensor time-synchronization and calibration are discussed. Second, the utilized complementary multi-modal fusion approach for onboard localization and mapping for individual robots is described. Finally, the deployed global multi-robot mapping approach is detailed, along with the alignment of individual robot frames $\mathbb{B}$ with the DARPA defined coordinate frame $\mathbb{D}$, and transmission of maps to the Base Station. Figure 12 depicts the used notation with coordinate frames and provides a general overview of the deployed localization and mapping scheme. During the deployment, each robot builds and maintains an onboard map of its traversed environment. These maps are periodically sent to the Base Station mapping server,

which combines individual robot maps and produces a globally optimized map. The self-contained onboard localization and mapping approach enables the robots to explore autonomously, i.e., a connection to the Base Station is optional. However, when connected, the global map allows, in particular, the Human Supervisor to have a combined and optimized overview of the explored environment.

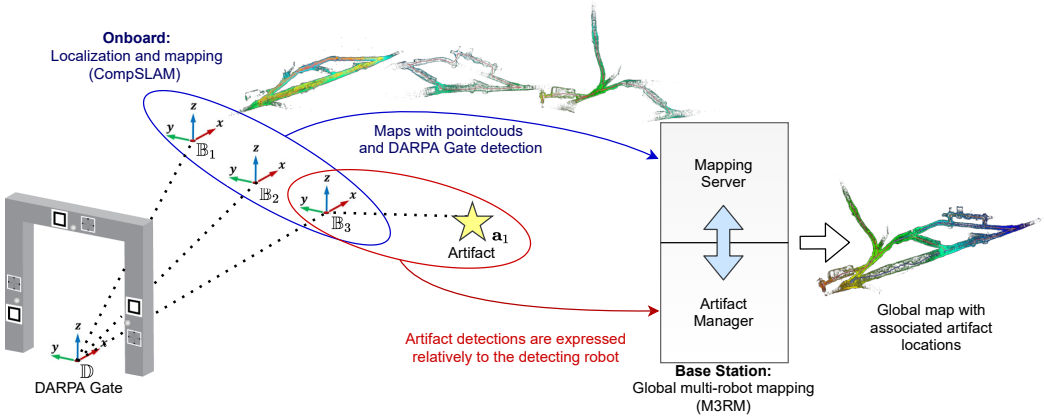


Figure 12: Illustration of the overall localization and mapping setup with coordinate frames for each robot $\mathbb{B}_1$, $\mathbb{B}_2$ and $\mathbb{B}_3$. All robots align themselves with respect to a common DARPA-defined coordinate frame $\mathbb{D}$ using fiducial markers on the starting DARPA gate. The mapping server on the Base Station receives maps from the individual robots (blue) and combines them into a globally optimized map. Artifact detections, such as $\mathbf{a}_1$, are recorded relative to the robot detecting them ($\mathbb{B}_3$) and are sent to the artifact manager (red) for storing, additional inspection, and reporting them in coordinate frame $\mathbb{D}$ using the globally optimized map to the DARPA Command Post for scoring.

6.1 Time Synchronization and Calibration

Precise time synchronization of the employed sensory systems is a crucial task for mapping and localization. Although the robots only move at moderate velocities, the range at which they detect visual landmarks and LiDAR points can be large. Given that small rotation changes can lead to large translations for points that lie far away from the sensor frame's origin, small timestamp inconsistencies between angular and linear measurements can lead to significant systematic errors when distant points are observed.

The Armadillo robot (during the Tunnel Circuit) and the ANYmal robots (during the Urban Circuit) employed a camera trigger board (Tschopp et al., 2020) capable of onboard synchronizing with a host computer using time translation. An integrated Kalman Filter estimates the clock-skew and offset between all sensors and the host computer. Additionally, since multi-camera systems inherently yield different exposure times for each camera, the trigger board performs exposure compensation using the mid-exposure times.

Moreover, a tightly-coupled illumination board was used to trigger a set of LEDs according to the cameras' exposure times for the flying robots and Armadillo. The LEDs are only turned on between the start of the first camera and the end of the last camera's exposure. This makes it possible to run the LEDs at a high brightness while minimizing heat and energy consumption.

For the purposes of artifact localization and detection, as well as robot localization and mapping, a number of calibrations are required. The color camera intrinsics and camera-to-IMU extrinsics were identified based on (Furgale et al., 2013). The intrinsic calibration parameters of the thermal camera were calculated using our custom designed thermal checkerboard pattern (Papachristos et al., 2018). The camera-to-LiDAR calibration

was derived based on an implementation of the work in (Zhou et al., 2018).

The Aerial Scouts operate with hardware-assisted software camera-to-IMU synchronization. While the camera and the IMU are free-running, the internal functionalities of the VectorNav VN-100 IMU are used to receive a synchronization pulse from the FLIR Blackfly S camera at the start of each exposure. Contained within every IMU message is the time difference between the rising edge of the synchronization pulse and the sample time of the current IMU message. IMU messages are assumed to be received in near real-time, and therefore the image timestamps are corrected based on the time difference reported by the IMU subtracted from the IMU message receive time.

6.2 Complementary Multi-Modal Localization and Mapping

To enable reliable and independent robot operation in challenging environments, the robots of team CERBERUS individually deploy an onboard complementary multi-modal localization and mapping (CompSLAM) approach (Khattak et al., 2020). This approach fuses visual and thermal imagery, LiDAR depth data, and inertial cues in a hierarchical manner and provides redundancy against cases of sensor data degradation. A schematic overview of the proposed approach is shown in Figure 13. Camera images from both visual and thermal cameras, as well as IMU measurements are fused using an Extended Kalman Filter (EKF) estimator. The deployed EKF-estimator builds on top of the work presented in (Bloesch et al., 2017), due to its low computational complexity, and extends it to concurrently operate on both visual and full-radiometric thermal images, as described in (Khattak et al., 2019; Khattak et al., 2020).

To keep computational costs tractable while utilizing multi-spectral camera inputs, the Visual-Thermal-Inertial Odometry (VTIO) estimator tracks a fixed number of features in its state by balancing the share of tracked features from each image, subject to image quality using a spatial and temporal entropy evaluation (Khattak et al., 2019). The health of the odometry estimate is evaluated by measuring covariance growth using the D-Optimality criterion (Carrillo et al., 2012). If the VTIO estimate is healthy, it is passed to the Laser Odometry (LO) and Laser Mapping (LM) modules. The LO module first independently ensures that the input estimate's translational and rotation components are within robot motion bounds, set by the controller, before utilizing it as a full or partial motion prior for the alignment of consecutive LiDAR scans.

Scan-to-scan alignment is performed by minimizing point-to-line and point-to-plane distances between extracted line and plane features, similar to the work of (Zhang and Singh, 2014). The quality of scan alignment is checked by evaluating the minimum eigenvalues of the optimization Hessian matrix (Zhang et al., 2016), and ill-conditioned dimensions are replaced with VTIO estimates before propagating the LO estimate and providing it to the LM module, along with extracted scan features. The VTIO and LO modules provide estimates at the update rates of images and point clouds, respectively; in contrast, the LM module operates at half the LO module rate to remain computationally tractable. For this reason, the LM module also independently evaluates input motion estimates from both modules before using them to estimate the refined robot pose in the local map using scan-to-submap alignment. Once a refined estimate is obtained, it is checked using similar health metrics in the LO step, and the final estimate is utilized for the propagation of the local robot map.

6.3 Global Multi-Robot Mapping

In addition to the onboard CompSLAM, robots with an established connection to the Base Station send maps back to the mapping server, as well as artifact detections to the artifact manager. A centralized mapping server allows the Human Supervisor to view and inspect artifact detections along with a combined multi-robot representation of the environment. Generally, the global multi-modal, multi-robot mapping (M3RM) approach is based on the existing framework Maplab (Schneider et al., 2018) and divided into two parts. In the first part, each robot converts its raw sensor measurements into submaps that span consecutive time intervals. Each submap consists of a factor graph, visual descriptors, and compressed LiDAR pointclouds.

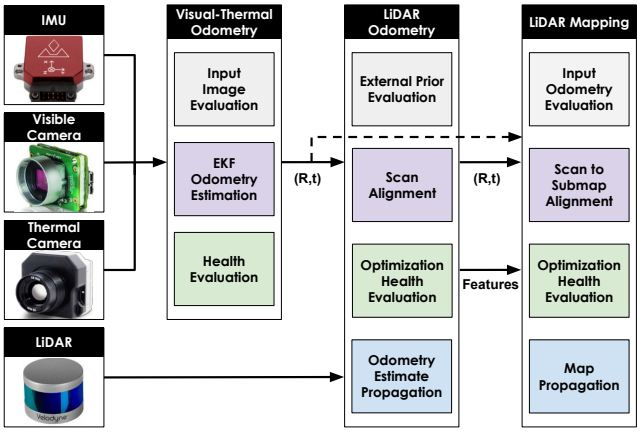


Figure 13: Overview of the deployed complementary multi-modal localization approach (CompSLAM). The Visual-Thermal-Inertial Odometry module receives camera and IMU data and provides a motion prior to both LiDAR Odometry and LiDAR Mapping modules to perform scan-to-scan and scan-to-submap alignments respectively. Motion estimates between modules are only propagated if they pass individual health checks.

These submaps are then transferred to the mapping server. Building the submaps on the robots reduces the required robot-to-mapping-server bandwidth, while only using a modest amount of computing power on each robot. The second part takes place on the mapping server and consists of combining the individual submaps into an optimized, globally consistent multi-robot map. The following subsections start by covering the submap creation and transmission approach in more detail, followed by the submap merging and global optimization strategy.

Submapping and Map Transmission over Network - Each robot builds an on-robot pose graph comprising IMU, visual landmarks, and relative transformation constraints from the odometry. Furthermore, the on-robot pose graphs are divided into chunks (i.e. submaps) and are periodically sent to the Base Station to build a multi-robot map. The composed system is generally independent of each robot's odometry estimation, i.e., the mapping system solely requires a consistent estimate of the robot's state and the IMU biases. Specifically, during the on-robot submap building, BRISK features are detected and tracked, and at submap completion, triangulated to construct a local map using the odometry information. Consequently, this approach requires motions within a submap such that uninitialized landmarks can be triangulated. In the case of motionless submaps, which commonly occur at startup, a stationary edge connects the first and the last vertex in the pose graph to avoid optimization issues.

In addition to visual landmarks, LiDAR scans are attached to the pose graph as resources using their timestamp and extrinsic calibration. The attached LiDAR scans are associated with a specific location in the pose graph and are not used for any additional computation on the robot itself but instead only on the Base Station.

Submaps are created by splitting the local maps at a fixed time interval and are uniquely identified using a hash of the timestamp and robot name. To ensure consistency in the submaps when combining them at the Base Station, the first and last vertex of consecutive submaps overlap. The server additionally ensures that the last and first vertex of consecutive submaps observe the same landmarks during the combination and merges them accordingly. Before transmission, each submap is keyframed (Dymczyk et al., 2015) and serialized to disk. A synchronization service then transfers the submaps to the Base Station in a granular fashion (file-based), such that transfers can efficiently be resumed after sudden connection losses. Whenever a submap transfer is completed, the synchronization service notifies the mapping server, which in turn processes and merges the submap into the global map.

Globally Consistent Multi-Robot Mapping - The mapping server on the Base Station periodically checks for completed submap transmissions and maintains a global map comprising all individual maps per robot. During the optimization of the global map, the mapping server utilizes different types of constraints as depicted in Figure 14. Maps from individual robots are connected by either merging visual landmarks (global visual loop closure) or performing a dense alignment between spatially close poses. Moreover, each robot

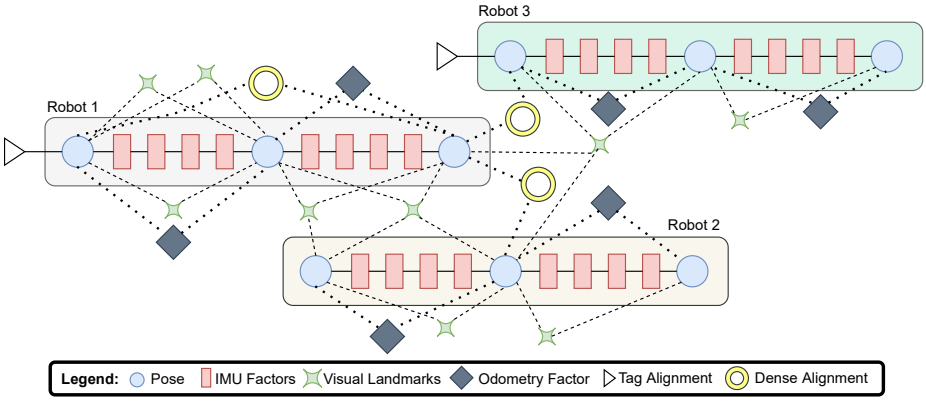


Figure 14: Illustration of three multi-modal pose graphs in the Base Station's mapping server. The graphs from individual robots are combined by co-observed landmarks or aligned pointclouds. Robot 1 and 3 detected the DARPA gate, whereas robot 2 was merged using shared landmarks.

performs an alignment to the DARPA frame $\mathbb{D}$ by detecting AprilTags on the Starting Gate with known 3D coordinates. The camera position is found in DARPA frame coordinates by solving the perspective-n-point problem of the 2D tag detections and its result is added to the pose graph as a prior factor. Incorporating the DARPA frame alignment into the optimization has the advantage of including detection uncertainties, being more robust against outliers, and being independent of the utilized sensor system. Additionally, this facilitates the global map merging since the base frame of the robot's map is already known.

Before merging a robot's submap into the global map, each submap is processed by itself. The submap processing performs a fixed number of operations, i.e., (i) a visual landmark quality check (ii) visual loop closure (iii) LiDAR registrations, and (iv) optimization. Since the individual submap processing is independent of other submaps, the mapping server processes up to four submaps concurrently. Furthermore, the optimizations are limited to a maximum of two minutes to ensure fast iterations.

All new visual landmarks are first evaluated in terms of their distance, angle, and the number of observations. Landmarks that do not fall within the thresholds for these criteria are marked as disabled during the optimization. Next, visual-based loop closures (Lynen et al., 2015) are found using the tracked visual BRISK features and an inverted multi-index. Loop closures are handled by merging the visual landmarks in the pose graph, enabling the optimization to close loops solely using the reprojection error.

Furthermore, additional LiDAR constraints are extracted by registering consecutive scans within a submap using Iterative Closest Point (ICP). The current states of the poses in the submap graph are used to determine a transformation prior for the registration. Registration results that moderately contradict the prior transformation are rejected as outliers to avoid constraints across floors or through walls. The remaining registrations are added as relative 6DoF constraints between each scan's closest pose in the submap where the IMU is additionally utilized for interpolation between the timestamp of the scan and the pose. The optimization assumes a fixed uncertainty for each registration constraint.

After the optimization, the submaps are merged into the current global multi-robot map. Similar to the submap processing, the mapping server continuously performs a set of operations on the global map, i.e., (i)

global visual loop closure, (ii) global LiDAR registrations, and (iii) optimization. In contrast to the previous submap processing, each operation utilizes all information from each robot's currently explored areas. Hence, the LiDAR constraints are found based on a proximity search within the global map. For each pointcloud associated with the global pose graph, a radius search is performed to retrieve spatially close registration candidates, including poses to other robots. Furthermore, all constraints resulting from LiDAR registrations are implemented as switchable constraints (Sunderhauf and Protzel, 2012).

The global multi-robot map provides an overview of the explored areas to the Human Supervisor and allows the operator to command robots to unexplored regions. Last but not least, artifact detections are utilized together with the reconstructed map to query their location. When only the map onboard the robot is available, the artifact locations are directly acquired, while when the M3RM-based map is also available, then it yields optimized artifact locations.

7 Artifacts

In the SubT Challenge, teams are scored by reporting the class and location of a set of artifacts to the DARPA Command Post. A fixed number of report attempts are given to each team for each Competition Course. A point is earned for each report which correctly identifies the class of an artifact and its position is within 5m of the object's ground-truth location. Throughout both competition circuits, survivors, cell phones, and backpacks were placed alongside drills and fire extinguishers in the Tunnel Circuit, followed by CO_2 (gas) sources and vents in the Urban Circuit. In this section, multi-modal artifact detection and classification are described, including the initial detection onboard the robot, as well as how the detected artifacts are attached to the global map presented in Section 6.3 such that their estimated positions can be improved using the combined measurements of all robots.

7.1 Detection and Classification

Artifact detection and classification initially takes place onboard each robot. For the visible artifacts, survivors, cell phones, backpacks, drills, fire extinguishers, and vents, visual artifact detection using deep learning was employed. Specifically, we used YOLOv3 (Redmon and Farhadi, 2018) and its ROS porting (Bjelonic, 2018), trained on datasets featuring all DARPA-specified artifacts except CO_2 gas. The training data consists of images from both the aerial and ground platforms, in laboratory settings against backgrounds of varying color, as well as in subterranean environments in varying lighting conditions. In addition, handheld versions of robot sensor configurations along with the on-board lighting solutions were re-created to facilitate rapid data collection and to improve viewpoint variation in the dataset. The number of images used to train each class varies from a minimum of 1915 for the cell phone, to a maximum of 3519 for the fire extinguisher. Table 3 details the number of training images used for each artifact class.

Artifact	Survivor	Fire Extinguisher	Cell Phone	Drill	Backpack	Vent
No. Images	2660	3519	1915	3084	3388	2301

Table 3: Number of images used to train YOLOv3 for each class of visible artifact.

In addition to automatic detections, compressed image streams are sent back from the robots to the Base Station, where the Human Supervisor is located. Using the Operator Interface described in Section 9, the supervisor can pause and rewind these streams and manually select artifacts by clicking on the images. Given the bounding box of an automatic detection or a click from the supervisor, the interface further allows to manually add, translate, relabel and remove artifacts attached to the global map that can be reported to the DARPA Command Post.

Due to the low precision-recall for visual cell phone detection, each robot could also detect the phone's Bluetooth signal, and report its presence to the Human Supervisor. In the Urban Circuit, all ground systems additionally integrated a Sensirion SCD30 CO_2 sensor to detect the gas artifact, reporting the position above which the CO_2 concentration had passed some threshold.

7.2 Map Attachment

The attachment of artifacts to the global map is performed in two steps. The first step takes place onboard and consists of filtering the artifact detections to determine the artifact's position in the robot's local coordinate frame. Referring to Figure 12, we denote the estimated position of the jth artifact in the body frame of the robot which detected it with ${}^{\mathbb{B}}\mathbf{a}_j$. In the second step, an artifact report is transferred to the mapping server, where it is attached to the pose graph and its optimized position in the DARPA frame ${}^{\mathbb{D}}\mathbf{a}_{j,opt}$ can be computed.

Due to the possibility of false positives and to prevent continuous artifact detections from saturating the robot's network connection to the Base Station, detections are first cast into the robot's map, and then filtered before being sent to the Base Station, providing the Human Supervisor with only high confidence detections. Specifically, the bounding box containing the detected object in the image plane is divided into a grid of pixels. These pixels are converted into rays using the camera model and the robot's extrinsic calibration parameters. Casting these rays into $\mathbb{M}$ results in a set of points that include points on the object, as well as other surfaces contained within the bounding box. To approximate the object's true position, the median point is selected as the measured artifact position. To filter both multiple class detections and refine the estimate of the artifact's position $\mathbf{a}_j$, a sphere with radius R_a is drawn at the initial estimate of $\mathbf{a}_j$. Additional detections whose projected positions lie within this sphere are considered to be detections of the same object. For each sphere, separate binary Bayes filters for each artifact class are recursively updated to estimate the probability that the sphere contains an object of a given class, and the sphere's center $\mathbf{a}_j$ is further updated as the average position of all detections projected into the sphere. Once the probability of any one class exceeds a predefined threshold, the artifact's detection is considered "confirmed". A summarized detection report is sent to the Human Supervisor alongside a representative image of the artifact, and the corresponding sphere is frozen to prevent any additional detections and subsequent duplicate reports. This process is depicted in Figure 15.

Despite effort for automated localization of the cell phone's position via a sequence of pose-annotated Bluetooth-based detections and tracking of their RSSI values, it was determined that the most reliable method would be to notify the Human Supervisor of the robot pose at which RSSI values of a cell phone were detected. The supervisor could then improve the cell phone's position by moving a marker on the map or by drawing a bounding box around the cell phone in the camera images. In case the latter option was used, the bounding box would be sent back to the robot, which would then convert it to a 3D artifact position $\mathbf{a}_j$ using the method described in the previous paragraph. In the Urban Circuit, this same approach was used to determine the gas artifact positions.

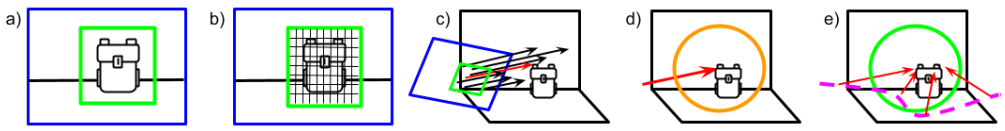


Figure 15: Illustration of the process of casting automatic detections into the map. a) Shows the detection of a backpack artifact with the image frame in blue, and the bounding box in green. b) The bounding box is divided into a fixed grid. c) A ray from each grid cell is cast from the image plane until it intersects a surface. d) The median ray, in red, is selected and a sphere of radius R_a, in orange, is placed in the map. e) Subsequent rays from additional detections along the path shown in pink, which land within the sphere, increase the confidence of the artifact category until reaching a threshold.

The artifact reports sent from the robot to the mapping server consist of the artifact's type, the timestamp t_i at which it was "confirmed" and the artifact's filtered position in the robot's local frame ${}^{\mathbb{B}_{t_i}}\mathbf{a}_j$. The annotated camera image in which the artifact was detected at time t_i is also included to provide additional context to the Human Supervisor. Once received by the mapping server, the artifact report is attached to the pose graph node $T_{\mathbb{D}\mathbb{B}_{t_i}}$ that corresponds to the pose of the robot's body in the DARPA frame $\mathbb{D}$ at timestamp t_i. The artifact's most recent optimized position can then be computed as ${}^{\mathbb{D}}\mathbf{a}_{j,opt} = T_{\mathbb{D}\mathbb{B}_{t_i},opt}\ {}^{\mathbb{B}_{t_i}}\mathbf{a}_j$. When desired, the artifact's unoptimized position can also be recovered with ${}^{\mathbb{D}}\mathbf{a}_{j,unopt} = T_{\mathbb{D}\mathbb{B}_{t_i},unopt}\ {}^{\mathbb{B}_{t_i}}\mathbf{a}_j$.

8 Networking

During mission deployments, all robots communicate with the Base Station via an ad hoc wireless network to share mapping data, artifact locations, and other information. This network operates in the 5.8GHz range and is progressively built by the robots that can communicate with each other and extend the network by dropping additional WiFi nodes. This section describes the hardware and software configuration that enables remote networking of the agents in the underground. Figure 16 shows an overview of the communication architecture used by team CERBERUS.

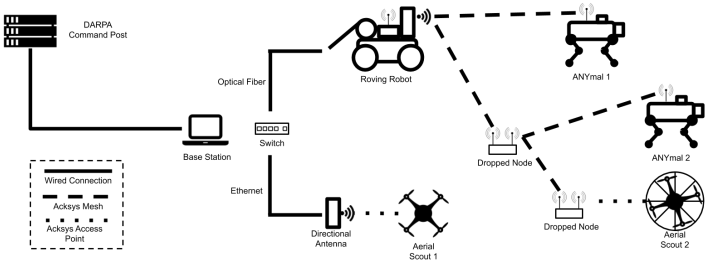


Figure 16: Block-diagram view of the networking infrastructure used during the Tunnel and Urban Circuits. Roving and quadrupedal robots, as well as dropped communication nodes, create an ad hoc mesh network, which establishes the backbone communication infrastructure. The roving robot was also equipped with a fiber optical cable to maintain connection to the Base Station. Aerial scouts used a standard (802.11) WiFi network, created either by a directional antenna at the Base Station or by a dropped breadcrumb.

8.1 Mobile Agent Nodes

Agents employ two different hardware methods. Ground robots are equipped with a commercial off-the-shelf device (ACKSYS Communications & Systems - EmbedAir1000) that supports WiFi access point, client, repeater, and MESH point modes. In mesh mode, these devices create a mesh network, that is a local network topology in which the infrastructure nodes connect directly, dynamically and non-hierarchically to as many other nodes as possible and cooperate with one another to efficiently route data from/to clients. In this way, ground robots establish a backbone wireless network shared among the other agents.

Due to their limited payload capability and size constraints, the aerial robots did not carry mesh node-capable devices and acted solely as clients. They connect to the closest existing wireless access point, broadcast either by the Base Station or by nodes deployed by ground robots.

8.2 Base Station Network

In preparation for a wide variety of competition scenarios, the team prepared four solutions to provide communications from the Base Station into the subterranean environment. First, the Base Station uses an ACKSYS EmbedAir1000 WiFi radio and connects to the mesh network using a high gain, directional panel antenna from L-Comm; supplying a high bandwidth connection both within the Staging Area as well as deep into the competition area. Second, Armadillo was deployed with a 300m-long optical fiber cable, a panel antenna, as well as a multi-modal perception system that further allowed it to be remotely controlled by the Human Supervisor while extending the WiFi network around corners and down corridors with ease in relative proximity to the Base Station. Third, a panel antenna was attached to a PVC pole and angled to provide a network down a stairway very near the competition gate. The pole was manually extended and oriented down the stairway by a team member. Finally, an EmbedAir1000 WiFi radio and its battery were placed in a foam ball which was manually thrown by a team member down a stairway to provide a network connection to the lower level of the circuit. The Gagarin Aerial Scout utilized both the ball and pole antennas while exploring the staircase.

8.3 Deployable Mesh Nodes

ANYmal robots were able to carry additional WiFi nodes and deploy them on the ground to expand the reach of the wireless mesh network incrementally. Each robot was able to store four modules inside its belly plate and release them by lowering the torso height and activating the payload release mechanism described in Section 4.1. Each node is a water and dustproof self contained unit, consisting of a battery pack (allowing approximately two hours of operation), a battery management system, a magnetic switch (to switch the module on after releasing it), an LED to aid the recovery of the module after the mission, two WiFi patch antennas, and an EmbedAir1000 module (see Figure 5). In as ideal as possible underground conditions, such as a straight tunnel, the modules can reliably communicate within a range of 100 m when dropped on the ground. However, due to obstructed lines of sight from solid material (e.g., in a mine corner), multipath and reflections in underground settings, performance may vary considerably. As evaluated via experimental observations, close to the first communication module a maximum available bandwidth of approximately 200Mbps was observed. In team deployments, the legged robots and the droppable communication nodes had both radios of the EmbedAir1000 module enabled to maximize the communication performance and maintain as much of the initial bandwidth as possible across the deployed communication topology.

The Human Supervisor was left in charge of deciding the deployment location, as wireless signal propagation is often unpredictable in subterranean scenarios. (Rizzo et al., 2013) show that a deviation in robot trajectory of a few centimeters can cause a significant difference in terms of received power on its onboard mesh receiver. In a more recent analysis, (Tardioli et al., 2019) show that due to destructive self-interference of signals reflecting off tunnel walls, there is a highly varying fading effect which leads to Received Signal Strength Indication (RSSI) having a variance of up to $30 - 40\%$. As a result, RSSI can be a misleading indicator to check the quality of a wireless connection inside a tunnel. Moreover, this variance can result in a link between two nearby mesh nodes reporting a worse signal strength than between nodes further apart. These peculiar characteristics, therefore, make an automatic decision of where to deploy mesh nodes a non-trivial problem. As a result, the deployment location was decided by the Human Supervisor. This decision was based on information about signal quality (such as the output of *ping* utility) between the Base Station and the agent, as well as considerations about the geometry of the environment surrounding the robot.

8.4 Nimbro - Multi-Master Network

The Robot Operating System (ROS) was used as middleware for message passing to enable communication between processes running on each robot, as well as across robots on the same network. However, ROS requires a *rosmaster* process which monitors all communication connections. There can only be one *rosmaster* on a given network and typically, this process is run onboard a robot's computer in case there is

communication loss with the Base Station. This presents a challenge in scenarios with multiple robots, each with their own *rosmaster*, communicating on the same network to the same Base Station.

The Nimbro Network[1] software package allows the generic transport of ROS topics and services over unreliable network connections and has been deployed in several robotic challenges (Stückler et al., 2016; Schwarz et al., 2017). Since it avoids any configuration/discovery handshake, it is suited for situations like the SubT Challenge where wireless connections can drop and recover unexpectedly. Its primary purposes are to decouple multiple *rosmasters* and select which data are transmitted between agents and the Base Station. A hardcoded list of data to be exchanged may be cumbersome to maintain and adapt, especially when tools such as *multimaster_fkie* (Tiderko et al., 2016) offer the possibility to automatically establish and manage a multi-master network with little or no configuration. However, these automatic tools require extra customization to be adapted to the SubT Challenge, as done by team CoSTAR (Queralta et al., 2020) and may be error-prone, transmitting unnecessary data and using limited bandwidth. Therefore, given that an ANYmal robot had more than 800 topics and 400 services in its ROS system, the alternative to clearly identify which data streams should be transmitted and not rely on any automatic decision was preferred. A key factor in a system with multiple *rosmasters* is time synchronization between the agents and the Base Station. This was achieved through the usage of the software tool *chrony*[2]. *chrony* is an implementation of the Network Time Protocol (NTP) that can synchronize the system clock with a predefined server. In this use case scenario, each robot acted as a client of the Base Station, which was the main NTP server.

9 Operator Interface

A crucial aspect of the SubT Challenge is the single Human Supervisor at the Base Station. This person is the only individual allowed to coordinate, manage, and monitor the deployed systems. The single operator requirement is intended to minimize the amount of human intervention for this type of autonomous mission and forces the robots to have a high level of autonomy, while remotely mapping and/or navigating complex environments.

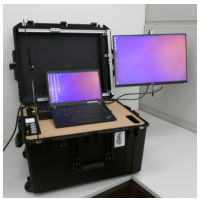
During the Tunnel and Urban Circuits, the Human Supervisor used a custom and integrated solution for the Base Station setup (see Figure 17). This solution was composed of a sturdy box (Peli Air Case - Pelican) containing electronic components (WiFi router, auxiliary screen, etc.). It ensured fast deployment and setup of the Base Station. The supervisor used two main software modules to interact with the deployed robots: a Mission Control Interface (MCI), and an Artifacts Reporting Interface (ARI), exposed through a Graphical User Interface (GUI) (see Figure 18). The MCI was used to (a) check the status of each agent, (b) give high-level commands to override the current robot's behavior and (c) request specific sensor data to evaluate the current situation better. The ARI was used by the Human Supervisor to (a) have a top-down view of the global map created during the mission, (b) visualize throttled and compressed image streams from the agents, to manually report artifacts, and to (c) check, and approve or discard artifacts that were automatically detected by the robots.

Team CERBERUS adapted its approach towards the single Human Supervisor requirement between the Tunnel and Urban Circuits, improving the GUIs to control multiple agents, as well as the amount of exchanged data between robots and the Base Station. Major changes to the communication architecture were introduced. These are highlighted in the following sections.

9.1 Robot Control

During the Tunnel Circuit, each robot received commands from the Human Supervisor via a dedicated Operator's PC (OPC). An OPC was specifically set up to interact with one or more robots. It used the

[1] `https://github.com/AIS-Bonn/nimbro_network`
[2] `https://chrony.tuxfamily.org/`



(a) Base Station.



(b) Tunnel Circuit setup.



(c) Urban Circuit setup.

Figure 17: Setup of the Base Station for the Human Supervisor. Operator's box with external screen and docking station (a). Setup for Tunnel Circuit: one PC for aerial robots, two PCs to control two ANYmal robots, one PC for the Armadillo rover, and the main ARI running on the screen of another PC (b). Setup for Urban Circuit: one PC for aerial robots, one PC to control all ground robots, and the main ARI running on the screen of another PC (c).

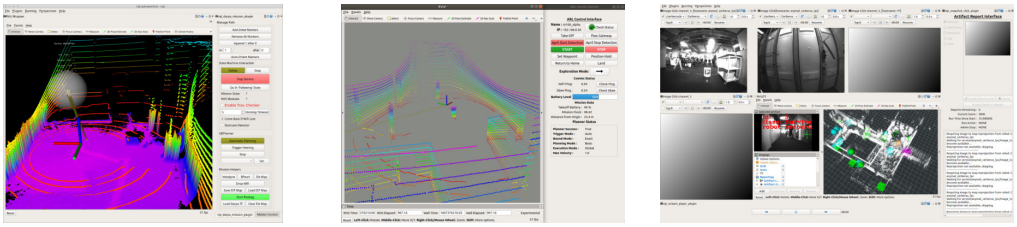
rosmaster of the corresponding robot and was used to initialize the agent (launching procedure, detection of the DARPA gate, etc.). Every ground robot operated in conjunction with its own OPC, and could stream all requested topics to the supervisor. On the other hand, all the aerial scouts could be controlled - one at a time - by another OPC. Each of these OPCs run its own instance of MCI, customized based on the type of robot. Only Aerial Scouts took advantage of the multi-master architecture with the Nimbro Network software package (described in Section 8.4), whereas ground robots used it only to report artifacts and map updates. Once each robot was set up by the Pit Crew and ready to start the mission, its OPC was passed to the Human Supervisor that could then use it to deploy the robot. The described setup resulted in an extra burden for the single Human Supervisor who had to deal with multiple PCs to execute the mission. For example, to control two ANYmals, one wheeled ground robot and aerial robots, the supervisor had to deal with four different OPCs, plus the additional PC running ARI - see example setup in Figure 17b. This approach was not sustainable, and therefore specific modifications were applied for the Urban Circuit.

A different setup of the Base Station and for controlling the robots was employed during the Urban Circuit. Every robot leveraged the multi-master architecture with the Nimbro Network package and exported only a subset of the ROS topics and services to its MCI. This prevented the wireless network from being cluttered with data that was not strictly needed for the mission execution. Moreover, the concept of "on-demand" topics was introduced: sensor data such as LiDAR scans and elevation maps require a considerable amount of bandwidth to be streamed over WiFi, and are most of the time not necessary to be continuously sent to the Human Supervisor. Therefore a single message of these data streams was sent only upon request of the supervisor via the MCI. Each robot still had its dedicated OPC, but it was used solely for agent setup by the Pit Crew prior to the deployment. The overall startup procedure included one PC to run the instances of MCIs for all ground robots, and one other for the MCI for aerial robots. This meant that to control two ANYmals, the Armadillo rover and the aerial scouts, the Human Supervisor had to deal with two computers, located at the Base Station (see Figure 17c).

In both Circuits, the Human Supervisor was able to supervise the decisions taken by each agent and, if required, decide to re-define the exploration bounding box in which each robot was allowed to explore. In every assigned area as defined by the planner bounding box, each robot was exploring the unknown space autonomously without relying on external information. If necessary, the Human Supervisor was able to overrule the autonomous planner and provide way-points to any robot.

9.2 Artifact Reporting

The artifact reporting interface (see Figure 18c) is designed to reduce the strain on the Human Supervisor while improving situational awareness and reducing the chance of reporting mistakes. It is composed of panels showing video streams from the selected robots, a panel notifying the supervisor of all new automatic



(a) ANYmal MCI. (b) Aerial Scout MCI. (c) ARI

Figure 18: Mission Control Interfaces (MCI) for ANYmal (a) and aerial scouts (b). They have a similar layout but feature some customization based on the specific type of robot. The minimal amount of data displayed to the Human Supervisor reflects the need to reduce streamed data from each robot to the essential ones: a frame representing the robot's pose in the map, requested LiDAR scans and/or elevation map snapshot, information about automatically detected artifacts and their location. The Artifact Reporting Interface (ARI) combines video streams from all robots, a panel to accept and modify artifact detections, and an overview of the reported global dense map overlaid with the state of all detected artifacts (c).

artifact detections, an RViz window showing all detected artifacts in the global dense map, and a text window for important system errors. The video streams can be paused and rewound, allowing the supervisor to better understand the areas explored by the robots. In case some artifacts were not detected automatically by the robots, a bounding box can be drawn on the image to attach the artifact to the map as discussed in section 7.2. When the Human Supervisor is notified of a new automatic artifact detection, the associated camera image, annotated with the artifact's bounding box and class, is shown for context. Under normal operation, the supervisor would accept the artifact, which would then be added to the multi-robot global map. As a fallback, the supervisor is also able to bypass the automatic artifact type detection and mapping server and file artifact reports directly to the DARPA Command Post.

Once an artifact has been added to the multi-robot global map, it appears in an RViz panel together with the optimized global dense map. For additional context, the artifact's original position, optimized position, and the robot pose from which it was observed are shown. Interactive markers are used to allow the Human Supervisor to report, modify, and delete artifacts through a right-click menu. Whenever an artifact is selected, its associated annotated camera image is shown in a dock next to the map. This proved very helpful for the supervisor to recall and distinguish artifacts. Once an artifact is reported, its color changes from cyan (not yet reported) to green (valid) or red (invalid) depending on the response from the DARPA Command Post. As the robots explore the environment further, the global map improves due to additional measurements, including loop closures. When significant improvements occur, the Human Supervisor can submit new reports for artifacts that were initially rejected.

9.3 Fallback Solutions

Given the centralized nature of the reporting architecture, special care is taken to minimize the damage to the team's competition score in case of system malfunctions on the Human Supervisor's PC at the Base Station. Starting at the hardware level, a custom-built server is used as the Operator PC. To provide robustness to temporary power losses, it is powered through an Uninterruptible Power Supply. At the software level, all packages are designed such that they can be started independently and at any point in time. The packages that cannot be implemented in a stateless fashion continuously save backups of their internal state to persistent storage. As an example, all the artifact positions received at the Base Station are periodically saved to disk. All packages can, therefore, safely be relaunched at any time, either by the Human Supervisor or automatically if crashes are detected.

The primary reliability concept for the multi-robot global map is to combine measurements from comple-

mentary sensor modalities and to perform robust outlier rejection using switchable constraints (see section 6.3). To account for cases where areas in the global map appear incorrect, or the global map is delayed, a bypass is implemented for artifact and global map reporting. As discussed in section 7.2, the optimized position for an artifact $\mathbf{a}_j$ expressed in DARPA frame $\mathbb{D}$ is computed as ${}^{\mathbb{D}}\mathbf{a}_j = T_{\mathbb{DB}_{t_i},opt}\ {}^{\mathbb{B}_{t_i}}\mathbf{a}_j$, where $T_{\mathbb{DB}_{t_i},opt}$ corresponds to the optimized pose graph node of the robot that detected the artifact at timestamp t_i. In addition to $T_{\mathbb{DB}_{t_i},opt}$, the pose $T_{\mathbb{DB}_{t_i},unopt}$ given by the robot's own odometry at time t_i is also stored. Using the artifact reporting GUI, the Human Supervisor can report alternative, unoptimized artifact poses computed with ${}^{\mathbb{D}}\mathbf{a}_j = T_{\mathbb{DB}_{t_i},unopt}\ {}^{\mathbb{B}_{t_i}}\mathbf{a}_j$. In a similar fashion, the global pointcloud map is typically computed by transforming the robot's pointclouds into the DARPA frame based on the robot's optimized poses $T_{\mathbb{DB}_{t_i},opt}$. If the Human Supervisor flags the global map as unreliable, or if the time since the optimized map's last update exceeds a threshold, the map reporting system temporarily falls back to the unoptimized global pointcloud map computed by transforming the pointclouds into the DARPA frame using the unoptimized poses $T_{\mathbb{DB}_{t_i},unopt}$.

10 Experimental Evaluation

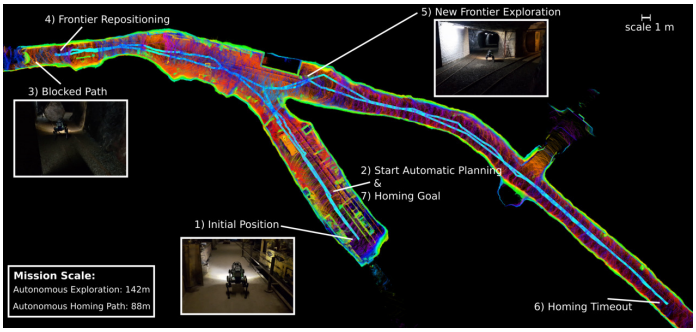
This section focuses on selected results from the deployment of team CERBERUS in the DARPA Subterranean Challenge Tunnel and Urban Circuit competition events.

10.1 Field Tests prior to the Circuit events

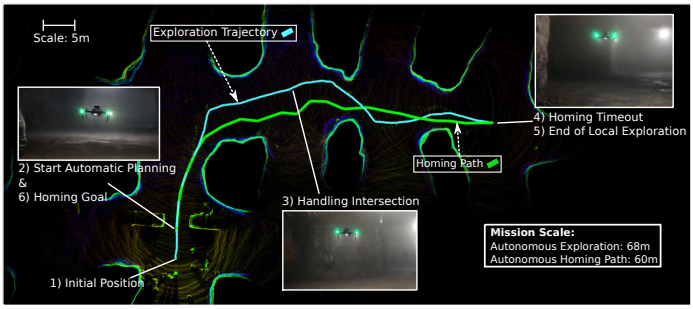
To verify and evaluate the performance of the CERBERUS walking and flying robots system-of-systems and prepare for the participation of our team in the DARPA Subterranean Challenge Circuit events, a set of field tests took place. Reflecting the geographical organization of our team, field evaluation took place both in the USA and in Switzerland which in turn allowed us to explore different types of underground mines, subterranean urban infrastructure, and lately cave environments.

In further detail, our field testing activities involved the following main sites: a) the TRJV underground mine (NV, USA), b) the Comstock underground mine (NV, USA), c) the Wampum underground room-and-pillar mine (PA, USA), d) several buildings of the University of Nevada, Reno (NV, USA), f) a railroad tunnel and a city tunnel (NV, USA), g) the Gonzen underground mine (Sargans, Switzerland), h) the Menznau Bunker (Luzern, Switzerland), i) the ARCHE test facilities (Wangen an der Aare, Switzerland) and other settings. More recently, we have tested in the Moaning Caverns (CA, USA), the Subway Cave (CA, USA), and other cave environments. Among those, only a subset have been utilized for testing with the entire team – both in terms of members of each of our groups and of our robots – as this required special travel organization. This included the Gonzen mine, the Wampum mine (shown in Figure 19), the Menznau Bunker, as well as the facilities of UNR. Another important testing event was the SubT Integration Exercise (STIX), which took place in April 2019 at the Edgar Experimental Mine (CO, USA). STIX was organized by DARPA so that qualified teams could test in a representative environment, with the same rules as the actual runs.

During such team-wide integration activities, the main effort has been on the integration and mission-like deployment of our system-of-systems solution involving robot configurations with significant differences, all being largely ongoing research prototypes instead of acquired and rather more mature products. This in turn also implied certain challenges relating to integration largely experienced in the Tunnel Circuit. To enable smoother integration and interoperability of our robotics team, we have since emphasized on unifying components as much as possible, such as sensing, perception algorithms, planning strategies, interfaces, and more. This approach paid off significantly in our Urban Circuit deployment and has since been improved continuously.



(a) ANYmal B deployed in the Gonzen underground mine in Switzerland.



(b) Alpha Aerial Scout deployed in Wampum mine.

Figure 19: Indicative field deployments of team CERBERUS' robots prior to official mission events. The shown missions were executed autonomously by the agents.

10.2 DARPA Subterranean Challenge – Tunnel Circuit

The Tunnel Circuit took place in August 2019 at the National Institute for Occupational Safety and Health (NIOSH) Mine in Pittsburgh, Pennsylvania, USA. This facility has the characteristics of a "room-and-pillar" mine, with horizontal arrays of rooms where pillars are composed of untouched material, left to support the roof. Teams were scored on two different courses: the Safety Research Course (SR) and the Experimental Course (EC).

Team CERBERUS deployed a total of six different types of robots across the four runs, correctly reporting six artifacts and scoring five points in total. CERBERUS finished sixth out of eleven teams in the overall ranking. The deployed robots were: two ANYmal B robots (called Bear and Badger), Armadillo rover, Alpha Aerial Scout, and Gagarin Aerial Scout. The ANYmal B robots, and the Alpha and Gagarin Aerial Scouts operated autonomously running the exploration planner, for which the local map subset $\mathbb{M}_L$ was set as a cuboid of dimensions $30 \times 30 \times 3.0$m, $16 \times 16 \times 1.5$m and $20 \times 20 \times 1.0$m for the ANYmals, Alpha and Gagarin respectively. The two ANYmal robots were deployed in every run, featuring different feet configurations (point feet, flat feet, wheels) but could score only one point. Armadillo scored the other four points during the last run; after both ANYmals could not proceed further and the rover was driven remotely by the Human Supervisor using a joystick. This was possible by viewing the live camera image streams, taking advantage of the extended connection provided by the fiber optic cable.

To report artifacts during the Tunnel Circuit, robots relied on their onboard multi-modal odometry estimation and their initial pose with respect to the DARPA global frame $\mathbb{D}$ (Figure 12). To provide a quantitative evaluation of the robot localization, ground-truth trajectories were later created by registering recorded robot

DARPA Tunnel Circuit - Robot Localization Results		
Robot / Run	**Relative Pose Error**	**Absolute Pose Error**
Alpha Aerial Scout / SR2	0.11m / 0.86°	0.48m / 1.05°
Armadillo Rover / SR2	0.03m / 0.37°	3.39m / 2.87°

Table 4: Results of the onboard CompSLAM approach deployed on the Alpha Aerial Scout and Armadillo rover robots during the DARPA Tunnel Circuit.

LiDAR scans to the provided DARPA ground-truth maps (Ramezani et al., 2020) using the ICP algorithm. Table 4 presents the Relative Pose Errors (RPE), calculated every 10 m, and Absolute Pose Errors (APE) for the Alpha Aerial Scout and the Armadillo rover for the Tunnel Circuit deployments, with Figure 20 presenting a qualitative comparison of each robot's trajectory against the ground-truth trajectory. During the Tunnel Circuit, the Alpha Aerial Scout utilized visual, depth, and inertial data for its odometry estimation which allowed it to autonomously explore and safely returned to its take-off position while traversing a total path of 180 m. As shown on the left in Figure 20, the estimated robot trajectory closely follows the ground-truth trajectory. During this deployment, the Alpha Aerial Scout would have continued its autonomous exploration mission (and would have explored the opposite direction as the Armadillo), but the exploration planner's bounding box, which defines the total area of interest for the planner, was incorrectly set by the Human Supervisor at the Base Station. The Armadillo was manually moved forward during the deployment and traversed a total path of 285 m before being stopped due to the optical fiber cable wrapping around its wheels. Armadillo was initially planned to provide only extended communication during the deployment; however, after initial competition deployments, the team re-purposed the robot to detect and report artifacts by modifying the onboard sensory payload and performing complete calibrations during the break between runs. This in-field modification proved useful, and the rover successfully reported four artifacts during its deployment. However, due to technical issues, the rover only utilized LiDAR data for its odometry estimation without employing any additional sensing data to improve its estimates. As noted in Table 4, even though the RPE remains low, the lack of additional sensing to improve the odometry led to an accumulation of errors, especially during rapid in-spot rotations, causing the robot trajectory to drift from the ground-truth, as seen in Figure 20. Nevertheless, the APE at the end of the robot's run still remained with-in the DARPA's margin of error of ± 5 m for artifact reporting.

10.3 DARPA Subterranean Challenge – Urban Circuit

The Urban Circuit took place in February 2020 at the Satsop Business Park in Elma (WA, USA). The Satsop Park is an unfinished nuclear power plant, featuring an underground labyrinth to be explored, with multiple levels, narrow passages, and different obstacles to traverse. Teams were scored on two different courses: the Alpha Course and the Beta Course. Team CERBERUS deployed a total of five robots across the four sessions, correctly reported eleven artifacts and scored seven points in total, achieving fifth placement (out of ten teams) in the overall ranking. The deployed robots were: two ANYmal B robots (Bear and Badger) with point feet, the Armadillo rover, the Alpha Aerial Scout, and the Gagarin Aerial Scout. Another roving robot for networking support was deployed but was used minimally during the sessions.

To provide detail with respect to the results of the path planning autonomy of our systems, Figures 21 and 22 present the graph-based exploration planner paths in the Alpha (second scored run) and Beta (second scored run) courses, respectively. For the Alpha Course, the paths for the Alpha Aerial Scout and ANYmal robot exploring the first level are presented, alongside the result for the Gagarin collision-tolerant aerial robot traveling down the stairwell to the next level and landing afterwards. The frontier vertices from the global graph of ANYmal are highlighted in the leftmost sub-figure. For the Beta Course, Figure 22 presents the results of the planner for the two ANYmal robots exploring the first floor of the course. The results show autonomous navigation through narrow openings, path refinement to enhance the clearance from obstacles, and instances where the traversability constraints applicable to ANYmal have led to adjustment of the

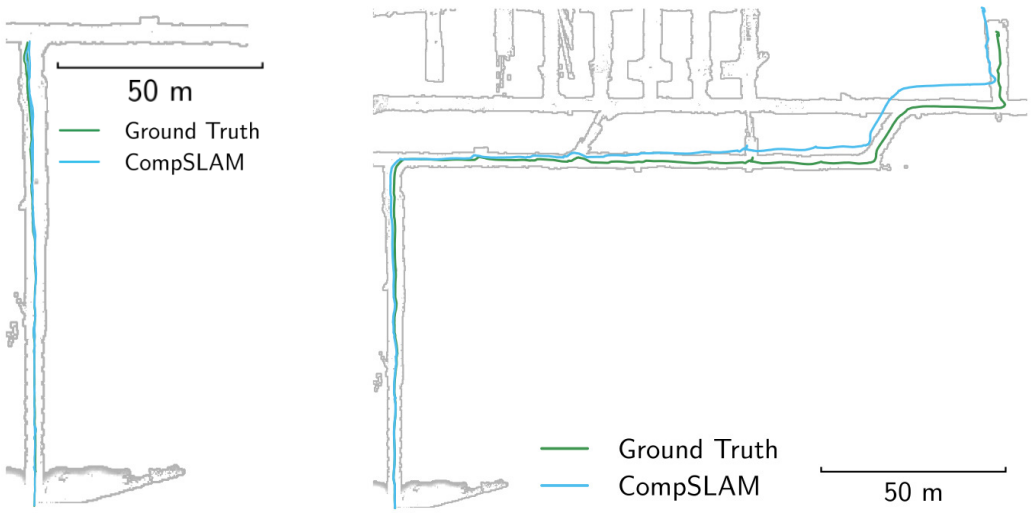


Figure 20: Comparison of onboard (CompSLAM) estimated trajectory (blue) with ground-truth trajectory (green) for the Alpha Aerial Scout (left) and the Armadillo rover (right) during competition runs of the DARPA SubT Tunnel Circuit. The aerial robot utilized visual, depth, and inertial data for its odometry estimation compared to the rover, which relied purely on depth data, hence producing better trajectory estimates when compared against ground-truth. This trajectory comparison demonstrates the importance of utilizing multi-modal sensor data for robot navigation in challenging underground environments. A video of the Aerial Scout's mission can be found at `https://youtu.be/mw0qy05Fo6Q`.

exploration path. The figure further presents the dense local graphs built at different instances of the robot mission, alongside the incrementally built sparse global graph, and the associated detected frontiers of the exploration space for candidate robot re-positioning. In both runs, the aerial and legged robots operated in the horizontal local exploration mode with the local map $\mathbb{M}_L$ set as a cuboid of dimensions $20 \times 20 \times 3$m and $30 \times 30 \times 3$m respectively. The Gagarin collision-tolerant aerial robot also flew autonomously through a stairwell and for this task it was operating in the vertical exploration mode with $\mathbb{M}_L$ set to $8 \times 8 \times 5.5$m.

To provide a quantitative evaluation of the onboard localization and global mapping approaches, ground-truth robot trajectories were created by registering each robot pointcloud to the DARPA provided ground-truth map using the ICP algorithm, following a similar approach as described in Section 10.2. Table 5 provides RPE and APE metrics for the CompSLAM (onboard) localization and the M3RM (global) mapping approaches by comparing against individual ground-truth trajectories of walking and flying robots. During the Urban Circuit deployments, all robots utilized visual, depth, and inertial data for their localization and mapping estimation. It can be noted that both onboard and global approaches report low RPE values, and the measured APE values are within the allowed error margin of $\pm$ 5 m for artifact reporting. These results also demonstrate the benefit of employing a global mapping approach in addition to the onboard estimation as it notably produces lower APE values, which are of particular importance for accurate artifact reporting and scoring. A qualitative comparison of the robot trajectories with respect to ground-truth is shown in Figure 23. Furthermore, to demonstrate global mapping consistency, point clouds from all deployed aerial, wheeled, and legged robots during all scored runs were combined into globally optimized maps of Alpha and Beta Courses. On the left in Figure 24, a globally optimized map created by combining the individual maps of the Alpha Aerial Scout, Armadillo rover, and ANYmal Bear and ANYmal Badger legged robots is shown and overlaid on top of a DARPA-provided ground-truth map of the Alpha Course for comparison.

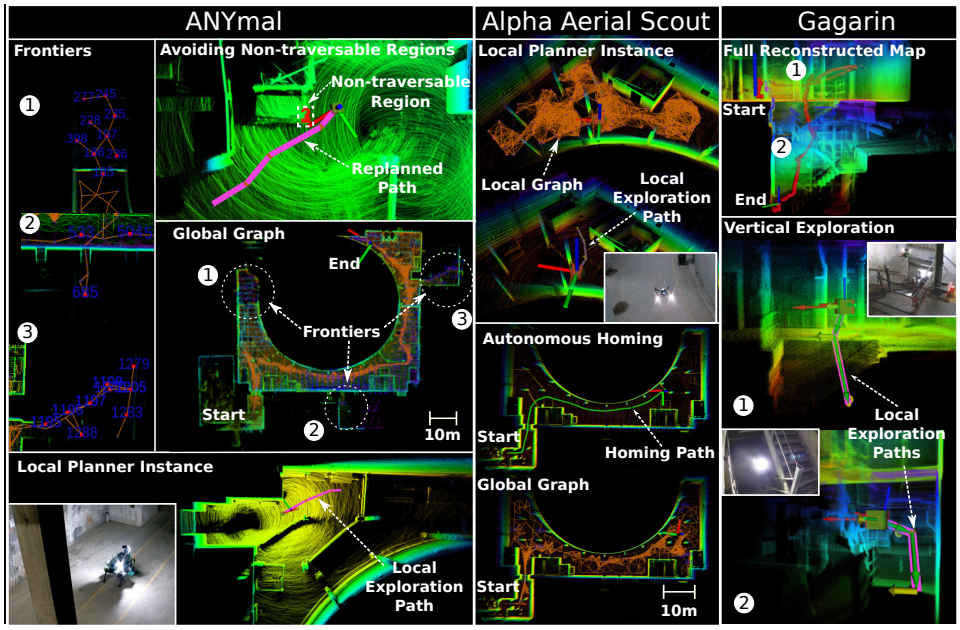


Figure 21: Details of deployment of our exploration planner in the Alpha course of the Urban Circuit. The ANYmal B robot, Alpha Aerial Scout and Gagarin Aerial Scout were deployed running the exploration planner. All robots were deployed at the entrance of the environment and after manual gate passing, the local planner was triggered. The Gagarin was commanded to perform vertical exploration of the staircase near the entrance, whereas the other robots performed horizontal exploration. The Alpha Aerial Scout explored the bottom section of the first floor of the environment and triggered auto-homing after reaching the battery limit. The ANYmal robot explored the area around the main reactor of the first floor. When the robot encountered non-traversable areas, the planner stopped, marked these areas as geofence zones and re-triggered the local planner. The planned paths, explored map and the global graph are shown for the Alpha Aerial Scout and ANYmal along with the frontier vertices detected by the planner in ANYmal's mission. The sub-figure for the Gagarin Aerial Scout shows the robot's path going down the staircase along with two local planner instances. Videos of Alpha Course deployments can be found at `https://youtu.be/160jJqJPKdo` (ANYmal), `https://youtu.be/Idmq_5hhMic` (Alpha Aerial Scout), and `https://youtu.be/iJqPAy0_tGM` (Gagarin Aerial Scout).

A similar comparison is provided for the Beta Course on the right in Figure 24, with a globally optimized map created by combining individual maps of the legged robots Bear and Badger. The presented maps are further annotated with locations of reported, detected (but not reported), and missed (unobserved despite in the area covered by the robot) artifacts as well as the locations of the dropped communication nodes to provide a holistic overview of each mission.

An overview of two scored runs (Alpha and Beta second run) is depicted in Figure 25. In the Alpha 2 run, the Armadillo rover was driven by the Human Supervisor into the mission area to provide connectivity to the other deployed robots. The rover had to stop after a short period of time because its fiber optical cable got tangled in one of the back wheels. Then ANYmal Badger was deployed and operated in fully autonomous mode. The robot explored the main area of the upper floor of the Alpha Course, but due to a bug in its onboard state machine it neither came back after it lost WiFi connection nor kept on exploring a different area once it reached a dead-end. Afterwards ANYmal Bear was deployed and teleoperated to descend a first ramp of stairs but could not proceed further. At this point, the Alpha Aerial Scout was sent in for

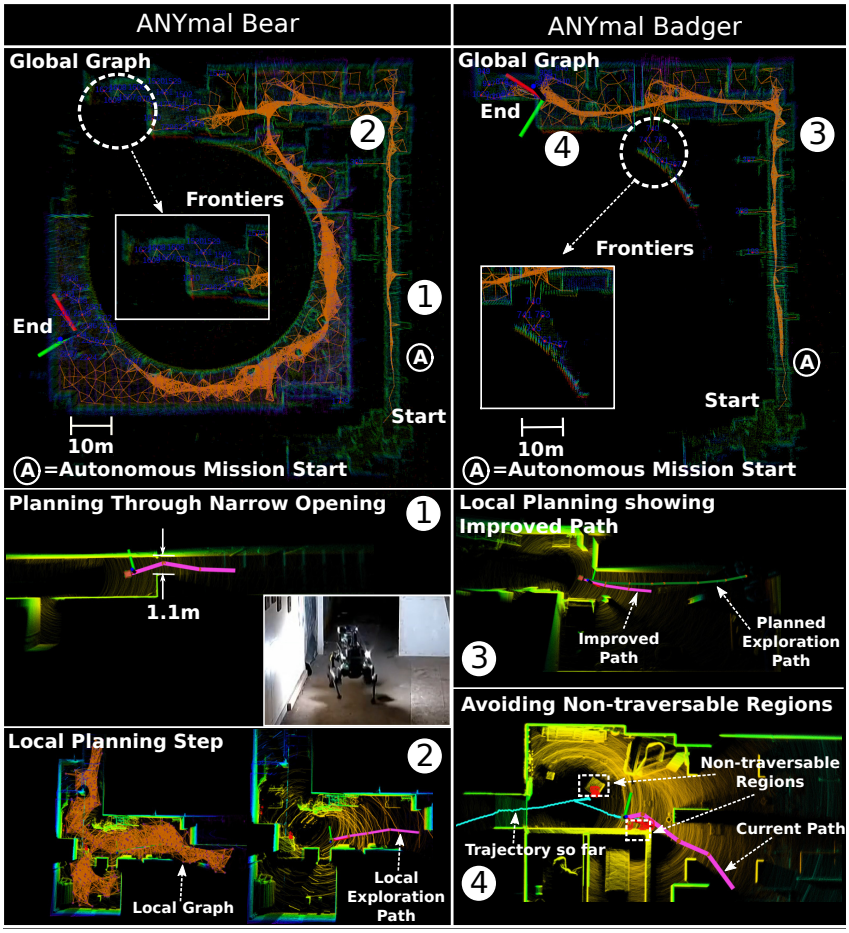


Figure 22: Details of deployment of our exploration planner in the Beta course of the Urban Circuit. Both the ANYmal B robots were deployed running the exploration planner. The figures on the top show the area explored along with the global graph and its frontier vertices built during the exploration mission. In the bottom half, various local exploration planning instances are shown. The planner's performance in the presence of narrow passages, non-traversable areas and an intersection are shown. The center-right sub-figure shows the planned path and path refined for safety. A video of the mission can be found at `https://youtu.be/0-GbFrrWcL4`.

another completely autonomous exploration of the upper floor, after which it came home successfully and was retrieved by the Pit Crew. Finally, the Gagarin Aerial Scout could plan its path through a series of staircases completely autonomously. In the Beta 2 run ANYmal Bear was deployed first, followed at a certain distance by the Armadillo rover. Once the first legged robot arrived at a point far enough from the starting area then ANYmal Badger was deployed and started its autonomous mission. The robot could explore a significant part of the upper floor of the Beta Course until it got stuck on an obstacle and could not recover properly. Moreover, since in that spot there was no WiFi connectivity, the Human Supervisor could not intervene at all. The Aerial Scouts were not deployed in this run due to the presence of two consecutive narrow doors at the entrance of the course, that rendered their deployment less safe, while significant area forward was already explored by the ANYmal robots. After the first half of the mission the Armadillo rover was parked

DARPA Urban Circuit - Robot Localization Results				
Robot / Run	**Relative Pose Error**		**Absolute Pose Error**	
	CompSLAM	**M3RM**	**CompSLAM**	**M3RM**
ANYmal Badger / Alpha 2	0.19m / 0.94°	0.11m / 0.82°	1.07m / 2.05°	0.26m / 0.89°
ANYmal Bear / Beta 2	0.22m / 0.99°	0.12m / 0.94°	0.58m / 0.86°	0.49m / 0.83°
Alpha Aerial Scout / Alpha 2	0.07m / 0.66°	0.13 m / 1.37 °	0.30m / 0.77°	0.25m / 1.23°

Table 5: Comparison of the mean Relative Pose Error and Absolute Pose Error from the DARPA Urban Circuit between for the onboard (CompSLAM) and global (M3RM) localization and mapping approaches.



Figure 23: CompSLAM (blue) and M3RM (magenta) trajectories executed onboard the CERBERUS robots against ground truth (green). *Left:* ANYmal Badger, Urban Alpha 2 mission. *Middle:* ANYmal Bear, Urban Beta 2 mission. *Right:* Alpha Aerial Scout, Urban Alpha 2 mission.

and ANYmal Bear continued the exploration mission. During this last part, the robot successfully deployed two WiFi beacons on the ground to extend the reach of the wireless network. It is noted that the further the robot explored the more the supervisor intervened in steering the robot (roughly after 38 min inside the mission). This was due to a bug in the onboard state machine that was triggering the homing command too often, even if the robot was commanded to keep on exploring. Therefore the Human Supervisor decided to stop the autonomous exploration (54 min inside the mission) and proceed solely with manual way-points. Three artifacts were correctly detected and reported to the DARPA Command Post.

10.4 Circuits Evaluation

The main technology evaluations are highlighted for the Tunnel and Urban Circuits. Team CERBERUS successfully deployed a set of diverse robotic systems and finished sixth (out of eleven teams) in the Tunnel Circuit and fifth (out of ten teams) at the Urban Circuit. At the same time, a set of limitations and challenges were faced due to the complexity of the underground environments and the challenges of the competition. Importantly, we describe the relative progress of the CERBERUS system at the Urban Circuit as compared to the Tunnel. This section is followed by a summary of lessons learned and prioritized future activities which reflect how the overall experience is digested and how the plan forward is organized towards long-term resilient autonomy in subterranean environments.

10.4.1 Tunnel Circuit

Legged robots: At this Circuit the legged robots faced a set of challenges primarily relating to locomotion issues. ANYmal experienced a corrupted elevation map leading to incorrect locomotion controller actions

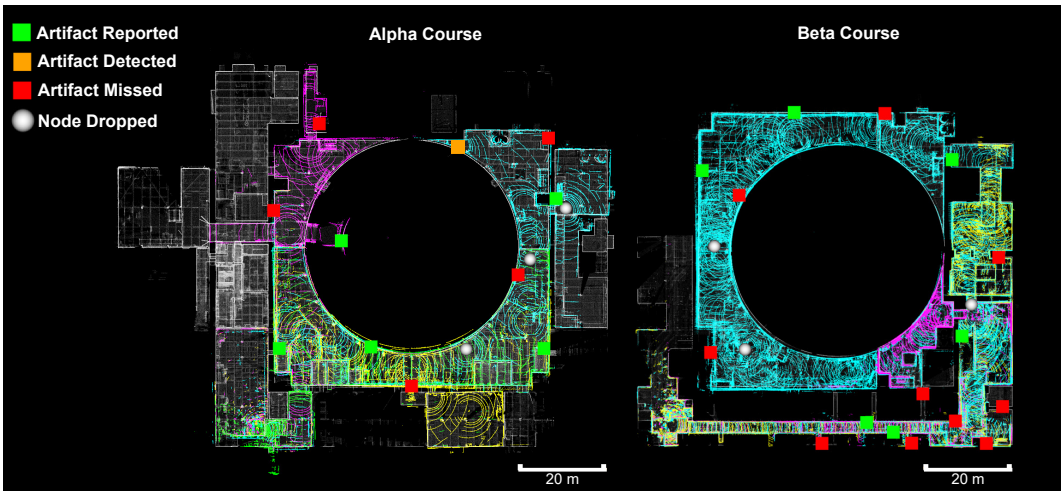


Figure 24: Global multi-robot map of all four sessions of the Urban Alpha and Beta Course using the M3RM approach. The global maps consist of the individual robot maps from ANYmal Bear (cyan), ANYmal Badger (yellow), Armadillo (magenta), and Alpha Aerial Scout (green) if available. Gagarin's map navigating to the next level is not merged. The ground truth map is denoted in white. The color squares denote successfully reported, detected, and missed artifact locations. The spheres illustrate WiFi beacons dropped by ANYmal Bear.

that prevented the robot from continuing the mission. The issues in the elevation map were caused either by foot slippage or introduced by corrupted data produced by the RealSense depth sensor (for example due to the presence of puddles of water). These challenges were corrected prior to the Urban Circuit especially through the integration of a different sensor and deployment of a new locomotion neural-network based controller, to decouple terrain perception and locomotion modules. ANYmal on wheels on the other hand, executed a fixed trotting gait which did not result in optimal motions and was prone to slippage between the wheels and the mud, which becomes even more critical at a higher speed. The lessons learned at this circuit guided our work for the Urban Circuit and further helped the improvements of the next generation of ANYmal on wheels (Bjelonic et al., 2021).

Aerial Scouts: The Alpha and Gagarin Aerial Scouts were the flying robots of our team that demonstrated the potential for fast exploration of underground tunnel environments. Especially the Alpha robot presented full autonomous exploration capacity - with no human inputs after initialization - and accurate localization and mapping. At the same time, the Gagarin robot showcased the importance of collision-tolerance when it accidentally hit the mine ceiling but was unaffected. Human mistakes in the definition of the bounding box for the exploration mission of the Alpha flying robot did not allow the robot to go to areas that were not explored previously by ground robots.

Armadillo rover: Armadillo was developed for the Tunnel Circuit to support the other robots by extending the WiFi range with its directional antenna and fiber optic cable connection to the Base Station. Given the results of the first three runs - where only one point was scored - the team decided to adapt CERBERUS' CompSLAM (a subset of its functionalities) specifically for the rover and made sure it could report artifacts with the onboard cameras. These modifications, which took place the night before the last run, made the robot tele-operable, capable of LiDAR-based localization and mapping alongside vision-based artifact detection/detection and resulted in four additional scored points. As such, the Armadillo rover played a larger role in the Tunnel Circuit than originally intended. This also led to the team's decision to use it as a - fallback - perception platform in the subsequent phases of the competition.

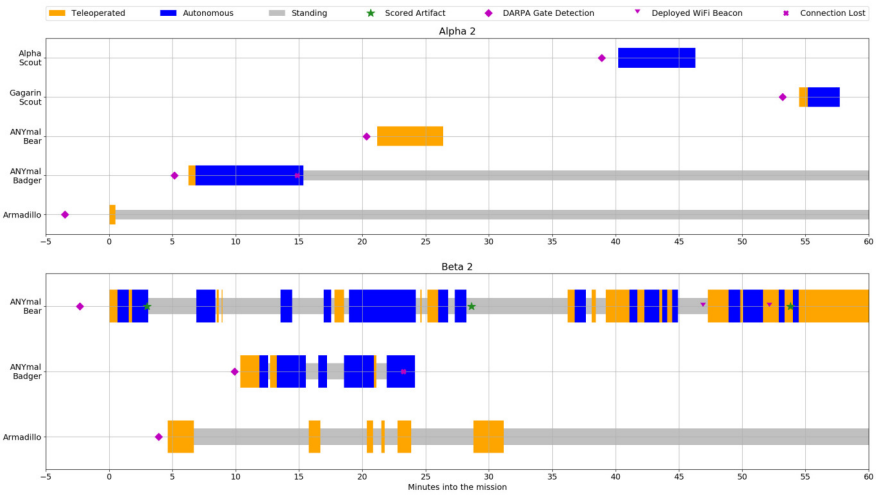


Figure 25: Mission analysis of Alpha 2 and Beta 2 runs. The plots show in which mode the robots were operating and when major mission events occurred. During the Alpha 2 run three robots were deployed in autonomous mode: The ANYmal Badger and the Alpha Aerial Scout could explore a significant part of the upper floor of the competition area, whereas the Gagarin Scout could plan its path through a series of staircases. ANYmal Bear was teleoperated downstairs a first set of staircases but could not proceed afterwards. At the Beta 2 run Aerial Scouts were not deployed due to narrow openings at the beginning of the course. The Armadillo rover was used to support the autonomous exploration of ANYmal Bear, while ANYmal Badger was sent to explore another area but got stuck on an obstacle and could not recover anymore. The last part of ANYmal Bear's mission was supervised by the Human Supervisor due to an issue in its onboard state machine. ANYmal Bear scored three points by detecting and reporting artifacts to the DARPA Command Post. A video of the Beta 2 mission can be found at `https://youtu.be/0-GbFrrWcL4`.

10.4.2 Urban Circuit

Legged robots: In the Urban Circuit our legged robots were reliably deployed in all four stages and scored the majority of the points. With the robots now presenting reliable locomotion capabilities, the robots' speed was increased from 0.2 m/s to 0.45 m/s. The main challenges faced during this Circuit related primarily to their autonomy behavior and especially to the decoupling of the terrain-aware traversability analysis from the exploration planner. Even though the legged systems could avoid walking into obstacles, the planned paths did not take into considerations any traversability analysis for these systems. Without a local traversability-aware planner, the quadrupedal robots were sometimes commanded to cross non-traversable regions and therefore had to stop, backup, and trigger the exploration planner again (with the non-traversable region now marked as geofence). This behavior slowed down the overall mission execution and sometimes also required the intervention of the Human Supervisor. Moreover, some corner cases, such as the robot becoming stuck on an obstacle or the concurrent triggering of safety events, were not handled correctly by the onboard state machine and therefore the robot could not recover properly from these events. These aspects are precious findings and are taken into account for the development of the new state machine for the Final Event.

Aerial Scouts: The team deployed two types of flying robots (Alpha Aerial Scout, Gagarin Aerial Scout) fully autonomously. The Alpha Aerial Scout performed autonomous local exploration and safely returned to the home location without a crash or need for human intervention demonstrating the reliable autonomy, path planning efficiency and accurate localization and mapping. It conducted its mission with an average

speed of 0.75 m/s. The Gagarin Aerial Scout demonstrated the vertical exploration capability of the planner by going down the staircase - presenting an average vertical velocity of 0.24 m/s - with the only human input being the initial selection of vertical exploration mode (the system exits this mode automatically once a floor is traversed). The Gagarin robot even collided with a metal bar of the staircase it traversed but due to its collision-tolerant design it survived and seamlessly continued its path. However, three problems persist. First, the aerial robots did not score points primarily because they started from the DARPA entrance gate and thus mainly went through areas already covered by ground robots. Second, the robots operate fully autonomously but within some general bounds provided at take-off time. If these bounds are set wrong - primarily due to the not very intuitive interface - the mission may not be productive. Third, we did not use flying robots extensively in this competition event and we deployed them after ground robots as opposed to an early scouting operation. Resolutions for these limitations are discussed in the lessons learned and future work plans.

Evaluation of Path Planning Autonomy: The exploration planner was able to provide safe and efficient local exploration paths in real-time. Being designed to work with both flying and legged robots, the planner worked as-is for all the deployed autonomous robots, with the only changes being robot-specific parameters. An observed drawback was the lack of shared information between robots, which resulted in two robots exploring the same area rendering the second robot's mission inefficient. Another observed issue was the lack of the possibility to bias the planner to change the exploration direction. If the Human Supervisor wanted to relocate the robot to a different area, a sequence of way-points towards it had to be created, forcing the supervisor to devote full attention to a single robot.

Evaluation of the Multi-modal and Multi-Robot Perception System: The onboard complementary multi-modal localization and mapping approach deployed on the aerial and wheeled robots during the Tunnel Circuit enabled reliable and robust robot navigation. The approach was extended and unified among all robots for the Urban Circuit and demonstrated its efficacy by producing low relative pose errors, as well as absolute pose errors within the margin of error for artifact reporting. In addition, the multi-robot global mapping solution which was still under development during the Tunnel Circuit was deployed during the Urban Circuit. It improved upon the estimates of the onboard localization and mapping solution and considerably improved the absolute pose errors for accurate artifact reporting. While exploring the Urban Circuit, the robots were sending submaps to the mapping server at the Base Station every 30 s. During the Urban Circuit, each submap was approximately 4 to 7 MB large. Although exact bandwidth usages were not recorded during the competition, internal bandwidth experiments indicated a constant use of 2 MBps for teleoperation (e.g. camera streams) and fallback information (cf. Section 9.3). Together with the submap transmission, the used bandwidth can reach 6 to 9 MBps. This bandwidth demand is considered to be low enough and compatible with the capabilities of the WiFi breadcrumbs, with the caveat being the ability of the robots to deploy enough WiFi nodes across the competition area.

Nevertheless, the onboard localization and global mapping solutions' performance was limited by a lack of standardization across the robots in terms of sensing setups. During the competition, the non-availability of sensors, such as thermal cameras, on some robots limited their suitability to operate in challenging obscurant filled sections of the environment. This did not correspond to a practical problem for our team during the deployment, but observing the complete courses after the competition it is important to report that it might become critical in the future. Similarly, robots equipped with similar sensing modalities such as visual cameras; differed in sensor types and lens setups, producing images of varying quality in poorly illuminated environments and limiting the potential for loop closure detection for multi-robot mapping. Furthermore, combining all the robots' information allowed for accurate mapping and improved pose estimation but increased the size of the underlying optimization problem, thus requiring longer processing times. During the processing of globally optimized maps, the Human Supervisor could not view a sufficiently recent global multi-robot map and relied on onboard maps of individual robots for mission assessment.

Evaluation of the Deployed Communications Solution: The combination of a tethered rover and deployable WiFi modules considerably helped the team's overall performance during the mission execution by extending the wireless network's reach. The main faced challenges were due to the quality of our com-

munication breadcrumbs, the difficulties in driving the tethered rover in an environment featuring several turns, the decision of how many WiFi modules to deploy from one ANYmal versus moving the rover further, and how many breadcrumbs ANYmal could integrate in relation to the effective range provided by each of them.

Optical fiber solution - The usage of an optical fiber cable offered a considerable advantage, especially in urban scenarios where thick walls and multiple turns played an important role in reducing the WiFi signal strength. On the other hand, while teleoperating the Armadillo rover, the Human Supervisor was completely focused on this task, while also making sure that the tether itself would not tangle or become tangled in the back wheels.

WiFi Range and Quality - Our team faced a set of challenges with respect to the WiFi modules utilized. Efforts for in-house development were not successful which led to the use of an off-the-shelf solution for which we had limited time to adjust. Observing the quality reached by other competing teams we identified this as a key limitation that we are currently prioritizing for our current field experiments and the next DARPA Circuit deployment.

WiFi Breadcrumbing - The decision of where to deploy a WiFi breadcrumb is not a trivial task. Both signal strength and geometric characteristics of the surrounding environments had to be taken into account. Two commonly occurring problems were detected: wrong decisions on where to deploy, and tilted modules after deployment. In the former case, it could happen that a deployed node was released too far from the previous breadcrumb and therefore could only connect to the mesh network through the robot which deployed it. It would then lose connection as soon as the agent proceeded further. For the latter case, the modules could tilt or flip up after the deployment (for example due to releases on slopes or uneven terrains in general), causing the internal antennas to face against the ground and therefore considerably reducing the wireless range of the node.

11 Main Lessons Learned and Discussion

Every field testing experience and especially every circuit event of the DARPA Subterranean Challenge provides large amount of information and insights with respect to how our research and development efforts towards efficient subterranean exploration should progress. CERBERUS represents an approach to the problem of robotic subterranean exploration that aims for one unifying solution across all types of underground environments that, once developed, can be scalable and versatile. This is the key motivation behind the robot platforms utilized, primarily legged and (collision-tolerant) flying platforms, and the methods designed. We believe that such a system-of-systems solution offers the resourcefulness required to navigate any subterranean setting. However, this approach also comes with a certain development overhead. Most of our deployed platforms are primarily research prototypes with significant technical challenges that had to be addressed during the circuit event preparations. At the same time, our focus on autonomous operation also implied that we initially had to dedicate a disproportionate amount of effort towards reaching a behavior that would have been rather easy to achieve with a more trained teleoperation process. However, observing the technological progress and the way our systems performed at every new field testing activity, we are confident that this solution is one of the most appropriate and we are thus particularly optimistic for new field tests, the subsequent circuit events, and the overall future of the technology. This section, provides overall conclusions for lessons learned that we hope can inform other teams, research groups, technology developers and system operators that share similar goals and interests.

Multi-Modal Perception, Localization and Mapping: The underground environments present a variety of challenges for robot perception. Proper utilization of multi-modal sensor data is critical for enabling robust and resilient, single- and multi-robot localization and mapping underground.

Unifying Onboard Sensing Solution - During the competition deployments, robots varied in their onboard

sensing payloads. Such heterogeneous sensor setups can impose significant challenges for deploying the same multi-modal estimation and multi-robot mapping approaches between robots. In addition to utilizing different sensors, small changes such as different camera types, lenses, and placements of illumination sources on the robot, play a critical role in the overall system performance, especially for loop closure and map merging. In the future, we aim to unify the employed sensor setup among robots. First, given robot payload capacity, we aim to equip the robots with the same set of multi-modal sensing capabilities to allow resilient operation of each robot in sensory degraded environments. Second, we aim to equip them with analogous sensors, lenses, synchronization hardware, and illumination. Such unification would help to guarantee an expected baseline performance from each robot and simplify the deployment of onboard multi-modal localization across the robots. Furthermore, it would also facilitate building globally consistent multi-robot maps.

Localization in Subterranean Environments - Localization in underground environments can be particularly difficult due to their complex nature and perceptually-degraded operating conditions. To improve the onboard localization performance, the CompSLAM approach will be extended to utilize external odometry estimates that are unique to robots, such as legged and wheeled odometry. In addition, to improve each estimation module's performance, data augmentation across sensing modalities, such as LiDAR depth fusion for visual and thermal feature tracking, would be incorporated. This can contribute both in accuracy but also the resilience of the system in obscurant-filled settings. Furthermore, to incorporate data from multiple sensors of the same type (e.g., multiple visual camera images), data level quality evaluation (Khattak et al., 2019) will be performed to make use of all available sensor data while keeping computational costs low. Finally, to facilitate tight constraint-based fusion across sensing modalities and external inputs, we aim to exploit factor-graph-based fusion (Wisth et al., 2021) to provide robustness and resilience in dark, dusty, and degenerate underground environments.

Consistent Onboard and Multi-Robot Perception - The integration between the CompSLAM and the M3RM client module running on the robot was still limited during the Urban Circuit. In future missions, the computed onboard features will also be shared for tighter integration with the global map building process and to avoid redundant computations.

Map Sharing - Bi-directional map sharing between robots and the Base Station, when in communication range, will provide individual robots with insight into unexplored areas of the environment. Such consistent map sharing between individual robots and the Base Station will improve each robot's global planner by explicitly knowing other robots' explored areas and allowing for collaborative planning. Furthermore, maps will be shared between robots to facilitate the launching of aerial robots from ground robots deeper into the circuit, allowing them to report artifacts without requiring an independent initial DARPA frame alignment.

Artifacts - Accurate artifact detection and reporting turned out to be a challenging problem. Our robots experienced cases where they passed by artifacts without detecting them. One reason was the limited and inconsistent illumination provided by each robot. Another reason related to limitations in our implementation. Although we initially attempted to determine the cell phone artifact positions by trilaterating the received signal strength of their Bluetooth signal, we eventually found that just notifying the Human Supervisor of Bluetooth signal presence and letting them determine the cell phone position yielded better results. A second requirement is that of accurate location reporting. The results given by the onboard localization and mapping solutions were satisfactory, but the centralized M3RM approach with map optimization provides notable improvements. M3RM decreases the individual robot APE values, resulting in more precise map and artifact position estimates even in the presence of drifting onboard estimation, whose errors grow in proportion to the distance travelled in the deployed implementation. Considering that artifact reports are only scored when the reported and true artifact positions differ by less than 5m, these improved estimates become indispensable as the exploration range of the robots and the scale of the environments increases.

Exploration and System-level Autonomy - Due to the scale and nature of subterranean environments and the number of robots controlled by one Human Supervisor, the deployed machines have to be fully autonomous with respect to their system functionalities and as autonomous as possible concerning their

exploration and information gathering behavior.

Exploration Path Planning and Artifact Search - GBPlanner has proven to be an efficient planner for volumetric exploration but presents two essential limitations with respect to the needs and challenges of the SubT Challenge. First, this relates to the fact that the method only accounts for the frustum of a depth sensor and derives paths based on the maximization of an associated information gain reflecting how much new volume will be explored. However, mapping new areas with a depth sensor such as a LiDAR does not automatically imply that the artifacts in this area will be detected. Artifact detection primarily relies on multiple sensors, such as cameras with different frustums or CO_2 and Bluetooth detection. In fact, a multi-hypothesis/multi-objective exploration and artifact search path planning pipeline is required. The second key limitation of our exploration planner is that for legged robots in particular it only has access to a short-range traversability analysis map which in turn does not allow it to securely plan long-term exploration paths. This can be addressed in two main ways, extending the range of the traversability map or, at a minimum, re-formulate the planner to run in a receding horizon fashion, thus maximizing robustness against wrong assumptions for the future terrain traversability. The latter is a direction we already consider, while the former corresponds to a more challenging problem in robotic perception. Last but not least, as multi-robot map sharing is facilitated then the extension of our methods to autonomous multi-robot exploration will take place.

State Machine - To maintain resilient autonomy and thus detect more artifacts, it is of major importance that each robot returns into communication range reliably before the end of the mission. To this end, the onboard state machine handles all the different cases. Until the Urban Circuit event, the team did not test long missions often enough which resulted in corner cases that were not handled properly during the circuit events. One of the major issues faced during the Urban Circuit by the legged robots was the inability to handle unforeseen events, such as getting stuck on an obstacle, with the predefined recovery behaviors. Whenever such events occurred, especially in areas with no WiFi connection with the Base Station, the robots could not recover and continue with the mission execution. To increase robustness in terms of autonomy, a behavior-based autonomy pipeline (Colledanchise and Ögren, 2016; Colledanchise and Ögren, 2018) is under consideration, in contrast to a more classical state machine framework. Furthermore, there is a clear need for a "watchdog" module with minimal interaction with other parts of the software that can recover the robot in case of partial software failures e.g., the planning pipeline.

Use of Simulation - In our developments up to the Tunnel Circuit, we were only making minimal use of simulation tools to test our path planners extensively. We had not invested in developing the world models that would allow us to test a set of planning corner cases. Instead, we prioritized field testing. On the way to the Urban Circuit, we employed simulation significantly more, and we are currently investing in its utilization even further. Based on our experience, simulation utilization is particularly important as a team progresses to investigate more complex exploration behaviors or multi-robot teaming.

Locomotion Controllers: A set of different controllers were deployed for the three versions of ANYmals (point feet, flat feet, wheels) during the two circuits. During the Tunnel Circuit, a perceptive model-based controller was used for the point and flat feet versions of the robot, whereas a specialized, non perceptive, model-based controller was employed for ANYmal on wheels. At the Urban Circuit, a neural-network-based controller was used for the flat feet robots.

Point Feet - The main driver for the change in the control strategy between the Tunnel and Urban Circuits lay in the interactions between the model-based controller and the terrain perception modules. The robustness of our locomotion controller, in the end, was limited by the quality of the perception part, which turned out to be challenging in the diverse subterranean environments, making the model-based controller more prone to failure. Therefore, during the Urban Circuit, the perception part was used only for traversability analysis purposes. Moreover, we observed very resilient locomotion in diverse environments with our blind neural-network-based controller. The steps towards the following runs include the investigation of a perceptive neural-network based controller to exploit the full locomotion capabilities of the legged robots while improving the detected issues in the terrain perception modules.

Wheels - ANYmal on wheels and its locomotion controller were not deployed in underground environments often enough before the Tunnel Circuit, which resulted in locomotion respectively slippage issues over the muddy terrain. During the Tunnel event, the robot executed a fixed trotting gait, which did not result in optimal motions. The experience and lessons learned during this circuit contributed to the development of a new model predictive controller (Bjelonic et al., 2021), where the robot dynamically chooses an optimal hybrid gait that is a fusion of powered rolling and legged stepping. In the Urban Circuit, although ANYmal on wheels could have brought competitive advantages (such as higher navigation speed), the team decided to favor a more standardized and tested approach and therefore employed two legged robots with the same point foot configuration.

Communication: The reliability and performance of underground communications have corresponded to a critical challenge in our experience so far. On the contrary, we have a positive experience concerning the networking solution and the particular implementation via the software package "Nimbro Network".

WiFi Breadcrumbs - Due to changes in the development process our team has faced difficulty to a) deploy miniaturized communication breadcrumbs with the appropriate balance of size/weight and range such that they can correspond to a solution tailored to our ground robotic systems, and b) reliably test the deployed solution well in advance of the competition. Despite focusing on autonomy, ensuring reliable communications persists as an important technical challenge.

Tethered Rover - The Armadillo rover was a valuable resource to extend the reach of the WiFi network and also act as a fallback perception platform. However, too much of the Human Supervisor's time was used to remotely control it. This solution should have been employed only for the very first part of the mission in the Urban Circuit, i.e., the rover should have been driven through the first main turns from the starting point and be parked there to support the other robots.

Interfacing with Multiple Robots: Supervising a multi-robot team while operating at the same time in unknown environments requires customized tools whose development was tackled only before the Urban Circuit. The main observed limitations and associated necessities with respect to this are: a) the lack of a unified and easy to use single user interface to control multiple robots; b) a user interface too complex to gather information about the updated global map, camera streams from the robots and status of detected artifacts; c) the absence of the possibility to both re-locate robots as well as bias their exploration direction without manually specifying intermediate way-points to the new zone that should be explored. These points will be addressed for the next runs.

Software and Hardware Unification: CERBERUS incorporates research prototypes of systems implementing novel features, either relating to legged robots or collision-tolerant flying systems. In our team - especially due to our geographic organization - we have identified the importance of unified software among as many of the robot functionalities as possible. Examples include the CompSLAM, M3RM and GBPlanner software packages which at the Urban Circuit (and subsequently) are uniformly deployed across all robots. This was not the case in the Tunnel Circuit and thus, we faced many integration challenges. The same principle holds for sensing hardware which was not fully unified in the Urban Circuit and is currently under improvement. Different hardware leads to different behaviors, the need for additional training data for artifact detection and other challenges.

Team Preparation: In the first period of the DARPA Subterranean Challenge, our team faced two important challenges specific to our configuration: a) our technological solutions existed across different groups and laboratories, and b) we were established in different geographic locations (USA and Switzerland). Integration activities across both sides of the Atlantic were scheduled and took place, but this was not enough for truly detailed system integration and testing. Since then we have achieved and established two major features, namely a) that our software solutions are being unified across different robotic configurations in all aspects that do not relate to robot-specific functionality, and b) we can execute missions that represent CERBERUS robots as a whole in Switzerland where the ANYmal robot is based and is rather more difficult to transfer between continents for frequent field tests. The know-how for the latter largely benefited from

the exchange of research personnel among our teams and from the approach to make aerial robots operate fully autonomously after take-off.

Research vs. Competition: Different tools and best practices were developed during the first two DARPA SubT events, which contributed to increasing the robustness of the overall systems. The main points of which are outlined here.

Logging - Extensive logging is key for robotic system development as issues can often not be detected during field-testing directly due to the complete system's complexity. Additionally, data can be used to analyze performance after field-testing in more detail. These findings were implemented between the Tunnel and the Urban Circuits and resulted in logging essential data in chunks (created every 5 min) to preserve as much data as possible in case of a complete system failure.

System Launch - To ensure a stable and robust launching of software, all ssh-shell sessions of the onboard PCs were run as *screen* sessions. This allowed to detach and reattach to existing sessions and avoided issues in the event of connection drops. To enhance remote monitoring and usability of the different applications (ROS nodes) running on the robots, *rosmon*[3] was used during the Urban Circuit. The next step before the final event is to switch from *screen* to a different terminal multiplexer with some extra built-in functionalities such as *tmux*[4] which also offers the possibility of saving the complete output history to file to further improve post processing analysis and debugging of every mission.

Periodic Mission Shakeouts - Weekly mission shakeouts were introduced after the Tunnel Circuit, to help the continuous integration tests of the deployed software. The goal was to test a mission scenario in a smaller scale environment with easy access, such as a garage or an underground hallway, and help the team spot software defects. They were carried out even if no major software changes were introduced and also helped the Pit Crew in standardizing the setup procedure of each robot. Though, these weekly shakeouts did not replace proper field deployments in underground scenarios. These were carried out regularly across the months proceeding the Urban Circuit.

Relevant and Challenging Field Testing - Eventually, a key component relates to real field testing our robotic systems in relevant environments and scenarios. We have learned that this process not only reveals the current status and allows for team training but also greatly helps in guiding the research required to enable resilient autonomy in such challenging robotic missions. For example, our CompSLAM and exploration planning pipelines are largely developed based on observations acquired during such deployments. Thus relevant field testing is of paramount importance both for competition-readiness and informed guidance of the research efforts.

Open Sourcing: Our experience in the SubT Challenge indicates that major progress will have been achieved by its completion, but corner cases will likely still exist. For the research and innovation community to proceed on top of the prior experience our team strongly considers and performs open-sourcing activities. We believe this is essential as such packages lead to research reproducibility, ability to compare, continuity, community development and collaborative exploitation. A list of packages open-sourced by the team is presented in Appendix A.

12 Priorities for Future Work

CERBERUS has outlined a plan of essential improvements that are currently taking place. These relate primarily to the following:

Subterranean Legged and Flying robots - One of our key priorities for the future is the introduction and

[3] `http://wiki.ros.org/rosmon`
[4] `https://github.com/tmux/tmux`

utilization of the new generation of the ANYbotics ANYmal robot: ANYmal C, which will fully replace the previous ANYmal B version. The new ANYmal C robots are now in use and provide the advantages of a new stronger, larger and more dexterous robot that further implements the latest version of the learning-based locomotion controller and seamlessly integrates the complete multi-modal sensor and processing suite of CERBERUS. Furthermore, another essential priority relates to high-speed aerial robot exploration, primarily via the collision-tolerant flying robots, and the integration of the flying robots onboard the ground systems so that their critical flight time is best used by being deployed where the ground robots cannot reach. These are - according to CERBERUS' vision - prerequisites for successful utilization of flying robots for underground exploration.

Resilient Multi-modal Robot Perception - The employed localization and mapping approaches will be improved to integrate available external estimates unique to robots, such as leg odometry for walking robots, to improve the overall estimation quality. Furthermore, to improve the robot operation's resilience and robustness in self-similar and obscurant-filled environments, detection of ill-conditioning and degradation will be performed on both method and data levels, respectively. Finally, tighter integration between multi-modal sensors will be exploited to improve the overall localization and mapping performance.

Multi-robot Mapping - To improve the collaborative localization, mapping, and planning between robots, the onboard and global localization and mapping approaches will be updated to communicate in a bi-directional manner with knowledge of the areas covered. Such development would facilitate the synergistic deployment of the communications breadcrumbs and detection of artifacts.

Autonomous Exploration and Visual Search - Fully autonomous exploration and artifact search capability at the level of every robot by co-optimizing both in terms of volumetric gain and artifacts' coverage will be developed, alongside sampling of paths in a manner that accounts for all the particularities of every platform, while remaining unified, and incorporating the recovery behaviors that facilitate resilient and unobstructed mission execution without need for human intervention other than true high-level decisions.

Communications - New hardware for subterranean communications based on a new baseline WiFi module (Rajant DX2) with improved range and reliability is being developed, alongside optimized breadcrumbing capabilities on the ANYmal C robot. This is combined with the Armadillo rover deploying its high-gain antenna while maintaining optical fiber connection to the ground.

Frequency and Character of Field Testing - A key aspect to meaningfully progress towards the SubT Challenge Finals and help in enabling resilient autonomous robotic operations underground relates to the frequency and the character of our field testing. The goal is to conduct large-scale field testing with increased frequency (e.g., monthly) and with a "Circuits-character" (i.e., by deploying the robots as per the rules of the challenge).

These priorities with respect to future work are part of a broader plan to enhance the operational capabilities and technological novelty of the CERBERUS system-of-systems as guided by the lessons learned and renewed via the continuous research contributions of our team laboratories.

13 Conclusion

This paper presents a comprehensive report of the technological progress made by team "CERBERUS" participating in the DARPA Subterranean Challenge up to the Urban Circuit event. A detailed explanation of the CERBERUS vision and motivation for the team's overall "System-of-Systems" methodology is given, followed by an extensive presentation of the employed robotic systems. At the core of the CERBERUS robotic system is a team of legged and flying robots (including collision-tolerant designs), further enhanced with a roving platform. The implemented autonomy components, including multi-modal perception and path planning, as well as the automated artifact detection and scoring systems, are detailed alongside the

robot-deployable communications solution and the operator interfaces for exerting high-level control. The contributed research allowed to facilitate certain levels of resilience in the CERBERUS robots operating inside visually-degraded, large-scale, narrow and broadly complex underground settings. Finally, the work presents the team's performance in both the Tunnel and Urban Circuits of the challenge, followed by the critical lessons learned from these experiences and a brief overview of future work to be performed leading up to the final Subterranean Challenge event. We hope that the presented work, released open-source code, and future activities can contribute towards the overall community goal of enabling resilient robotic autonomy in extreme underground environments.

Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Agreement No. HR00111820045. The presented content and ideas are solely those of the authors.

The authors would also like to thank ANYmal Bear, ANYmal Badger, SMB Armadillo, Aerial Scouts Alpha, Bravo, Charlie, and Gagarins who were not harmed during the DARPA SubT Challenge Events, even though the same might not be said for the prior field deployments. In any case, they made it home safely after each circuit and were able to keep up their hard working attitude and support to the robotics communities of their research labs. Even more importantly, we want to thank and express our appreciation for all members of the other teams participating in the SubT Challenge, the organization committee and the overall community involved in this exciting competition.

A List of Open Source Packages

With the goal to support the overall community around subterranean robotics, our team simultaneously focuses on open-sourcing significant components of our technological solution. A selected - but not exhaustive - list of relevant open-source packages of our team are outlined below.

GBPlanner - a graph-based exploration planner for subterranean environments: `https://github.com/ntnu-arl/gbplanner_ros`

MBPlanner - a motion primitives-based agile exploration planner for subterranean environments: `https://github.com/ntnu-arl/mbplanner_ros`

Gird Map - a universal grid map library for mobile robotic mapping: `https://github.com/ANYbotics/grid_map`

Kalibr - a visual-inertial calibration toolbox: `https://github.com/ethz-asl/kalibr`

Lidar Align - a package to calibrate LiDAR extrinsics: `https://github.com/ethz-asl/lidar_align`

Maplab - an open visual-inertial mapping framework, which includes the M3RM centralized multi-modal, multi-robot mapping extension: `https://github.com/ethz-asl/maplab`

Darknet ROS - Real-Time Object Detection for ROS: `https://github.com/leggedrobotics/darknet_ros`

Elevation Mapping - a robot-centric elevation mapping framework for rough terrain navigation: `https://github.com/ANYbotics/elevation_mapping`

ROVIO - a Robust Visual Inertial Odometry framework: `https://github.com/ethz-asl/rovio`

Traversability Estimation - a traversability mapping framework for mobile rough terrain navigation: `https:`

`//github.com/leggedrobotics/traversability_estimation`

Voxblox - a library for flexible voxel-based mapping: `https://github.com/ethz-asl/voxblox`

Voxgraph - a globally consistent volumetric mapping framework: `https://github.com/ethz-asl/voxgraph`

Simulation Models

The SubT Challenge Systems and Virtual Competitions are interconnected with the Systems' teams delivering models of their robots for the Virtual Competition, while progress achieved in the Virtual Competition can benefit the autonomy functionalities in the Systems Competition. Models of most of the robots fielded by our team in the Tunnel and Urban Circuits have already been released.

Virtual Robot Models - CERBERUS robots have been released for use in the the official "SubT Virtual Track Challenge" simulation environment. Robots models released so far include the ANYmal B "SubT" (`https://github.com/osrf/subt/tree/master/submitted_models/cerberus_anymal_b_sensor_config_1`), the Alpha Aerial Scout (`https://github.com/osrf/subt/tree/master/submitted_models/cerberus_gagarin_sensor_config_1`), and the Gagarin Aerial Scout (`https://github.com/osrf/subt/tree/master/submitted_models/cerberus_m100_sensor_config_1`).

B Videos from the Tunnel and Urban Circuits

We have assembled a set of videos presenting highlights of the deployment of the CERBERUS system-of-systems in the Tunnel and Urban Circuit events of the DARPA Subterranean Challenge.

- CERBERUS deployment at the DARPA SubT Challenge Urban Circuit *(combination from different courses)*: `https://youtu.be/pkgcsdAgxVc`
- Detailed overview of the CERBERUS deployment at the Beta Course of the DARPA SubT Challenge Urban Circuit: `https://youtu.be/0-GbFrrWcL4`
- The Alpha Aerial Scout exploring the first level of the Alpha Course of the DARPA SubT Challenge Urban Circuit *(video previously released)*: `https://youtu.be/Idmq_5hhMic`
- The Gagarin Aerial Scout proceeding to the lower level of the Alpha Course of the DARPA SubT Challenge Urban Circuit *(video previously released)*: `https://youtu.be/iJqPAy0_tGM`
- The ANYmal B exploring the first level of the Alpha Course of the DARPA SubT Challenge Urban Circuit *(video previously released)*: `https://youtu.be/160jJqJPKdo`
- The Alpha Aerial Scout in the Safety Research Course of the DARPA SubT Challenge Tunnel Circuit *(video previously released)*: `https://youtu.be/mw0qy05Fo6Q`

References

Bachrach, A. G. (2013). *Trajectory bundle estimation For perception-driven planning.* PhD thesis, Massachusetts Institute of Technology.

Bakambu, J. N. and Polotski, V. (2007). Autonomous system for navigation and surveying in underground mines. *Journal of Field Robotics*, 24(10):829–847. Wiley Online Library.

Baker, C., Morris, A., Ferguson, D., Thayer, S., Whittaker, C., Omohundro, Z., Reverte, C., Whittaker, W., Hahnel, D., and Thrun, S. (2004). A campaign in autonomous mine mapping. In *2004 IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 2004–2009. IEEE.

Bellicoso, C. D., Bjelonic, M., Wellhausen, L., Holtmann, K., Günther, F., Tranzatto, M., Fankhauser, P., and Hutter, M. (2018). Advances in real-world applications for legged robots. *Journal of Field Robotics*, 35(8):1311–1326. Wiley Online Library.

Bjelonic, M. (2016–2018). YOLO ROS: Real-time object detection for ROS. `https://github.com/leggedrobotics/darknet_ros`.

Bjelonic, M., Grandia, R., Harley, O., Galliard, C., Zimmermann, S., and Hutter, M. (2021). Whole-body mpc and online gait sequence generation for wheeled-legged robots. *under review for IEEE Robotics and Automation Letters*.

Bjelonic, M., Sankar, P. K., Bellicoso, C. D., Vallery, H., and Hutter, M. (2020). Rolling in the deep – hybrid locomotion for wheeled-legged robots using online trajectory optimization. *IEEE Robotics and Automation Letters*, 5(2):3626–3633. IEEE.

Bloesch, M., Burri, M., Omari, S., Hutter, M., and Siegwart, R. (2017). Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback. *The International Journal of Robotics Research*, 36(10):1053–1072. SAGE Publishing.

Boeing, A., Boulton, M., Bräunl, T., Frisch, B., Lopes, S., Morgan, A., Ophelders, F., Pangeni, S., Reid, R., Vinsen, K., et al. (2012). Wambot: Team magician's entry to the multi autonomous ground-robotic international challenge 2010. *Journal of Field Robotics*, 29(5):707–728. Wiley Online Library.

Bouman, A., Ginting, M. F., Alatur, N., Palieri, M., Fan, D. D., Touma, T., Pailevanian, T., Kim, S. K., Otsu, K., Burdick, J., and a. Agha-Mohammadi, A. (2020). Autonomous Spot: Long-Range Autonomous Exploration of Extreme Environments with Legged Locomotion. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2518–2525. IEEE.

Buchanan, R., Wellhausen, L., Bjelonic, M., Bandyopadhyay, T., Kottege, N., and Hutter, M. (2021). Perceptive whole-body planning for multilegged robots in confined spaces. *Journal of Field Robotics*, 38(1):68–84. Wiley Online Library.

Butzke, J., Daniilidis, K., Kushleyev, A., Lee, D. D., Likhachev, M., Phillips, C., and Phillips, M. (2012). The university of pennsylvania magic 2010 multi-robot unmanned vehicle system. *Journal of Field Robotics*, 29(5):745–761. Wiley Online Library.

Carlbaum, E., Mansouri, S. S., Kanellakis, C., Koval, A., and Nikolakopoulos, G. (2020). Towards robust localization deep feature extraction by cnn. In *IECON 2020 The 46th Annual Conference of the IEEE Industrial Electronics Society*, pages 807–812. IEEE.

Carrillo, H., Reid, I., and Castellanos, J. A. (2012). On the comparison of uncertainty criteria for active slam. In *2012 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2080–2087. IEEE.

Colledanchise, M. and Ögren, P. (2016). How behavior trees modularize hybrid control systems and generalize sequential behavior compositions, the subsumption architecture, and decision trees. *IEEE Transactions on robotics*, 33(2):372–389. IEEE.

Colledanchise, M. and Ögren, P. (2018). *Behavior trees in robotics and AI: An introduction*. CRC Press.

Corah, M., O'Meadhra, C., Goel, K., and Michael, N. (2019). Communication-efficient planning and mapping for multi-robot exploration in large environments. *IEEE Robotics and Automation Letters*, 4(2):1715–1721. IEEE.

Dang, T., Tranzatto, M., Khattak, S., Mascarich, F., Alexis, K., and Hutter, M. (2020). Graph-based subterranean exploration path planning using aerial and legged robots. *Journal of Field Robotics*, 37(8):1363–1388. Wiley Online Library.

Dijkstra, E. W. et al. (1959). A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271. Springer.

Dymczyk, M., Lynen, S., Bosse, M., and Siegwart, R. (2015). Keep it brief: Scalable creation of compressed localization maps. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2536–2542. IEEE.

Fankhauser, P., Bloesch, M., Gehring, C., Hutter, M., and Siegwart, R. (2014). Robot-centric elevation mapping with uncertainty estimates. In *Mobile Service Robotics*, pages 433–440. World Scientific.

Furgale, P., Rehder, J., and Siegwart, R. (2013). Unified temporal and spatial calibration for multi-sensor systems. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1280–1286. IEEE.

Goel, K., Tabib, W., and Michael, N. (2021). Rapid and high-fidelity subsurface exploration with multiple aerial robots. In Siciliano, B., Laschi, C., and Khatib, O., editors, *Experimental Robotics*, pages 436–448, Cham. Springer International Publishing.

Hines, T., Stepanas, K., Talbot, F., Sa, I., Lewis, J., Hernandez, E., Kottege, N., and Hudson, N. (2021). Virtual surfaces and attitude aware planning and behaviours for negative obstacle navigation. *IEEE Robotics and Automation Letters*. IEEE.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., and Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*.

Huang, Y.-W., Lu, C.-L., Chen, K.-L., Ser, P.-S., Huang, J.-T., Shen, Y.-C., Chen, P.-W., Chang, P.-K., Lee, S.-C., and Wang, H.-C. (2019). Duckiefloat: a collision-tolerant resource-constrained blimp for long-term autonomy in subterranean environments. *arXiv preprint arXiv:1910.14275*.

Hutter, M., Gehring, C., Jud, D., Lauber, A., Bellicoso, C. D., Tsounis, V., Hwangbo, J., Bodie, K., Fankhauser, P., Bloesch, M., Diethelm, R., Bachmann, S., Melzer, A., and Hoepflinger, M. (2016). ANYmal - A Highly Mobile and Dynamic Quadrupedal Robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 38–44. IEEE.

Jacobson, A., Zeng, F., Smith, D., Boswell, N., Peynot, T., and Milford, M. (2020). What localizes beneath: A metric multisensor localization and mapping system for autonomous underground mining vehicles. *Journal of Field Robotics*. Wiley Online Library.

Jenelten, F., Hwangbo, J., Tresoldi, F., Bellicoso, C. D., and Hutter, M. (2019). Dynamic locomotion on slippery ground. *IEEE Robotics and Automation Letters*, 4(4):4170–4176. IEEE.

Jenelten, F., Miki, T., Vijayan, A. E., Bjelonic, M., and Hutter, M. (2020). Perceptive locomotion in rough terrain – online foothold optimization. *IEEE Robotics and Automation Letters*, 5(4):5370–5376. IEEE.

Kamel, M., Stastny, T., Alexis, K., and Siegwart, R. (2017). *Model Predictive Control for Trajectory Tracking of Unmanned Aerial Vehicles Using Robot Operating System*, pages 3–39. Springer International Publishing.

Kanellakis, C., Karvelis, P., and Nikolakopoulos, G. (2019). Open space attraction based navigation in dark tunnels for mavs. In *International Conference on Computer Vision Systems*, pages 110–119. Springer.

Khattak, S., Mascarich, F., Dang, T., Papachristos, C., and Alexis, K. (2019). Robust thermal-inertial localization for aerial robots: A case for direct methods. In *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1061–1068. IEEE.

Khattak, S., Nguyen, H., Mascarich, F., Dang, T., and Alexis, K. (2020). Complementary multi–modal sensor fusion for resilient robot pose estimation in subterranean environments. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1024–1029. IEEE.

Khattak, S., Papachristos, C., and Alexis, K. (2019). Visual-thermal landmarks and inertial fusion for navigation in degraded visual environments. In *2019 IEEE Aerospace Conference*, pages 1–9. IEEE.

Khattak, S., Papachristos, C., and Alexis, K. (2020). Keyframe-based thermal–inertial odometry. *Journal of Field Robotics*, 37(4):552–579. Wiley Online Library.

Kolvenbach, H., Wisth, D., Buchanan, R., Valsecchi, G., Grandia, R., Fallon, M., and Hutter, M. (2020). Towards autonomous inspection of concrete deterioration in sewers with legged robots. *Journal of Field Robotics*. Wiley Online Library.

Lacaze, A., Murphy, K., Del Giorno, M., and Corley, K. (2012). Reconnaissance and autonomy for small robots (rasr) team: Magic 2010 challenge. *Journal of Field Robotics*, 29(5):729–744. Wiley Online Library.

Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., and Hutter, M. (2020). Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47). Science Robotics.

Lösch, R., Grehl, S., Donner, M., Buhl, C., and Jung, B. (2018). Design of an autonomous robot for mapping, navigation, and manipulation in underground mines. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1407–1412. IEEE.

Lynen, S., Sattler, T., Bosse, M., Hesch, J., Pollefeys, M., and Siegwart, R. (2015). Get Out of My Lab: Large-scale, Real-Time Visual-Inertial Localization. In *Robotics: Science and Systems XI*, volume 11. Robotics: Science and Systems Foundation.

Maity, A., Majumder, S., and Ray, D. N. (2013). Amphibian subterranean robot for mine exploration. In *2013 International Conference on Robotics, Biomimetics, Intelligent Computational Systems*, pages 242–246. IEEE.

Mansouri, S. S., Castaño, M., Kanellakis, C., and Nikolakopoulos, G. (2019a). Autonomous mav navigation in underground mines using darkness contours detection. In *2019 International Conference on Computer Vision Systems (ICVS)*, pages 164–174. Springer.

Mansouri, S. S., Karvelis, P., Kanellakis, C., Koval, A., and Nikolakopoulos, G. (2019b). Visual subterranean junction recognition for mavs based on convolutional neural networks. In *IECON 2019-45th Annual Conference of the IEEE Industrial Electronics Society*, volume 1, pages 192–197. IEEE.

Miller, I. D., Cladera, F., Cowley, A., Shivakumar, S. S., Lee, E. S., Jarin-Lipschitz, L., Bhat, A., Rodrigues, N., Zhou, A., Cohen, A., et al. (2020). Mine tunnel exploration using multiple quadrupedal robots. *IEEE Robotics and Automation Letters*, 5(2):2840–2847. IEEE.

Morris, A., Ferguson, D., Omohundro, Z., Bradley, D., Silver, D., Baker, C., Thayer, S., Whittaker, C., and Whittaker, W. (2006). Recent developments in subterranean robotics. *Journal of Field Robotics*, 23(1):35–57. Wiley Online Library.

Novák, P., Babjak, J., Kot, T., Olivka, P., and Moczulski, W. (2015). Exploration mobile robot for coal mines. In *International Workshop on Modelling and Simulation for Autonomous Systems*, pages 209–215. Springer.

Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., and Nieto, J. (2017). Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

Olson, E., Strom, J., Morton, R., Richardson, A., Ranganathan, P., Goeddel, R., Bulic, M., Crossman, J., and Marinier, B. (2012). Progress toward multi-robot reconnaissance and the magic 2010 competition. *Journal of Field Robotics*, 29(5):762–792. Wiley Online Library.

Palieri, M., Morrell, B., Thakur, A., Ebadi, K., Nash, J., Chatterjee, A., Kanellakis, C., Carlone, L., Guaragnella, C., and Agha-mohammadi, A.-a. (2020). Locus: A multi-sensor lidar-centric solution for high-precision odometry and 3d mapping in real-time. *IEEE Robotics and Automation Letters*, 6(2):421–428. IEEE.

Papachristos, C., Mascarich, F., and Alexis, K. (2018). Thermal-inertial localization for autonomous navigation of aerial robots through obscurants. In *2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 394–399. IEEE.

Queralta, J. P., Taipalmaa, J., Pullinen, B. C., Sarker, V. K., Gia, T. N., Tenhunen, H., Gabbouj, M., Raitoharju, J., and Westerlund, T. (2020). Collaborative multi-robot search and rescue: Planning, coordination, perception, and active vision. *IEEE Access*, 8:191617–191643. IEEE.

Ramezani, M., Wang, Y., Camurri, M., Wisth, D., Mattamala, M., and Fallon, M. (2020). The newer college dataset: Handheld lidar, inertial and vision with ground truth. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4353–4360. IEEE.

Redmon, J. and Farhadi, A. (2018). Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*.

Rizzo, C., Tardioli, D., Sicignano, D., Riazuelo, L., Villarroel, J. L., and Montano, L. (2013). Signal-based deployment planning for robot teams in tunnel-like fading environments. *The International Journal of Robotics Research*, 32(12):1381–1397. SAGE Publications Sage UK: London, England.

Rogers III, J. G., Sherrill, R. E., Schang, A., Meadows, S. L., Cox, E. P., Byrne, B., Baran, D. G., Curtis III, J. W., and Brink, K. M. (2017). Distributed subterranean exploration and mapping with teams of UAVs. In *Ground/Air Multisensor Interoperability, Integration, and Networking for Persistent ISR VIII*, volume 10190, pages 285 – 301. International Society for Optics and Photonics, SPIE.

Rouček, T., Pecka, M., Čížek, P., Petříček, T., Bayer, J., Šalanskỳ, V., Heřt, D., Petrlík, M., Báča, T., Spurnỳ, V., et al. (2019). Darpa subterranean challenge: Multi-robotic exploration of underground environments. In *International Conference on Modelling and Simulation for Autonomous Systesm*, pages 274–290. Springer.

Schneider, T., Dymczyk, M., Fehr, M., Egger, K., Lynen, S., Gilitschenski, I., and Siegwart, R. (2018). Maplab: An Open Framework for Research in Visual-Inertial Mapping and Localization. *IEEE Robotics and Automation Letters*, 3(3):1418–1425. IEEE.

Schwarz, M., Rodehutskors, T., Droeschel, D., Beul, M., Schreiber, M., Araslanov, N., Ivanov, I., Lenz, C., Razlaw, J., Schüller, S., et al. (2017). Nimbro rescue: Solving disaster-response tasks with the mobile manipulation robot momaro. *Journal of Field Robotics*, 34(2):400–425. Wiley Online Library.

Shivakumar, S. S., Rodrigues, N., Zhou, A., Miller, I. D., Kumar, V., and Taylor, C. J. (2020). Pst900: Rgb-thermal calibration, dataset and segmentation network. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9441–9447. IEEE.

Silver, D., Ferguson, D., Morris, A., and Thayer, S. (2006). Topological exploration of subterranean environments. *Journal of Field Robotics*, 23(6-7):395–415. Wiley Online Library.

Steindl, R., Molnar, T., Talbot, F., Hudson, N., Tam, B., Murrell, S., and Kottege, N. (2020). Bruce–design and development of a dynamic hexapod robot. *arXiv preprint arXiv:2011.00523*.

Stückler, J., Schwarz, M., Schadler, M., Topalidou-Kyniazopoulou, A., and Behnke, S. (2016). Nimbro explorer: Semiautonomous exploration and mobile manipulation in rough terrain. *Journal of Field Robotics*, 33(4):411–430. Wiley Online Library.

Sunderhauf, N. and Protzel, P. (2012). Switchable constraints for robust pose graph SLAM. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1879–1884. IEEE.

Tabib, W. and Michael, N. (2019). Simultaneous localization and mapping of subterranean voids with gaussian mixture models. In *2019 Field and Service Robotics (FSR)*, pages 173–187. Springer.

Tardioli, D., Riazuelo, L., Sicignano, D., Rizzo, C., Lera, F., Villarroel, J. L., and Montano, L. (2019). Ground robotics in tunnels: Keys and lessons learned after 10 years of research and experiments. *Journal of Field Robotics*, 36(6):1074–1101. Wiley Online Library.

Tiderko, A., Hoeller, F., and Röhling, T. (2016). *The ROS Multimaster Extension for Simplified Deployment of Multi-Robot Systems*, pages 629–650. Springer International Publishing, Cham.

Tschopp, F., Riner, M., Fehr, M., Bernreiter, L., Furrer, F., Novkovic, T., Pfrunder, A., Cadena, C., Siegwart, R., and Nieto, J. (2020). VersaVIS—An Open Versatile Multi-Camera Visual-Inertial Sensor Suite. *Sensors*, 20(5):1439. Multidisciplinary Digital Publishing Institute.

Valsecchi, G., Grandia, R., and Hutter, M. (2020). Quadrupedal locomotion on uneven terrain with sensorized feet. *IEEE Robotics and Automation Letters*, 5(2):1548–1555. IEEE.

Wang, C., Wang, W., Qiu, Y., Hu, Y., and Scherer, S. (2020). Visual memorability for robotic interestingness via unsupervised online learning. In *Computer Vision – ECCV 2020*, pages 52–68, Cham. Springer International Publishing.

Wellhausen, L. and Hutter, M. (2021). Rough terrain navigation planning for legged robots. *Submitted to IEEE International Conference on Robotics and Automation*. IEEE.

Wisth, D., Camurri, M., Das, S., and Fallon, M. (2021). Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry. *IEEE Robotics and Automation Letters*, 6(2):1004–1011. IEEE.

Zhang, J., Kaess, M., and Singh, S. (2016). On degeneracy of optimization-based state estimation problems. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 809–816. IEEE.

Zhang, J. and Singh, S. (2014). LOAM: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems X*. Robotics: Science and Systems Foundation.

Zhou, L., Li, Z., and Kaess, M. (2018). Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5562–5569. IEEE.

Zoula, M., Prágr, M., and Faigl, J. (2021). On building communication maps in subterranean environments. In *Modelling and Simulation for Autonomous Systems*. Springer International Publishing.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

Title of Paper	CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge
Publication Status	Accepted for publication
Publication Details	M. Tranzatto, F. Mascarich, L. Bernreiter, C. Godinho, M. Camurri, S. Khattak, T. Dang, V. Reijgwart, J. Löje, D. Wisth, S. Zimmermann, H. Nguyen, M. Fehr, L. Solanka, R. Buchanan, M. Bjelonic, N. Khedekar, M. Valceschini, F. Jenelten, M. Dharmadhikari, T. Homberger, P. D. Petris, L. Wellhausen, M. Kulkarni, T. Miki, S. Hirsch, M. Montenegro, C. Papachristos, F. Tresoldi, J. Carius, G. Valsecchi, J. Lee, K. Meyer, X. Wu, J. Nieto, A. Smith, M. Hutter, R. Siegwart, M. Mueller, M. Fallon, and K. Alexis, "CERBERUS: Deployment at the DARPA Subterranean Challenge Urban Circuit," *Field Robotics,* 2021, accepted, to appear.

Student Confirmation

Student Name:	David Wisth		
Contribution to the Paper	My contributions to the paper were: • Participated in preparatory field trials with the CERBERUS team in Switzerland (Jul 2019, Dec 2019) and Reno, NV, USA (Feb 2020). • Member of the CERBERUS team for the DARPA SubT Urban Challenge in Satsop, WA, USA (Feb 2020). • Deployed VILENS within the team's software stack as an alternative state estimation algorithm for the competition. • Evaluated VILENS on data from the SubT challenge for two publications. • Assisted with writing and proof-reading the paper. • Participated in weekly coordination meetings.		
Signature		Date	30/09/2021

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Maurice Fallon			
Supervisor comments	This is an accurate description		
Signature		Date	4/10/2021

This completed form should be included in the thesis, at the end of the relevant chapter.

B.4 Balancing the Budget: Feature Selection and Tracking for Multi-Camera Visual-Inertial Odometry

This article was published in the IEEE Robotics and Automation Letters (2022) and presented at the IEEE International Conference on Robotics and Automation [8]. A video summary of this publication can be found at: `https://youtu.be/cLWeAT72e0U`.

IEEE ROBOTICS AND AUTOMATION LETTERS. PREPRINT VERSION. ACCEPTED DECEMBER, 2021 1

Balancing the Budget: Feature Selection and Tracking for Multi-Camera Visual-Inertial Odometry

Lintong Zhang[1], David Wisth[1], Marco Camurri[1], and Maurice Fallon[1]

***Abstract*—We present a multi-camera visual-inertial odometry system based on factor graph optimization which estimates motion by using all cameras simultaneously while retaining a fixed overall feature budget. We focus on motion tracking in challenging environments, such as narrow corridors, dark spaces with aggressive motions, and abrupt lighting changes. These scenarios cause traditional monocular or stereo odometry to fail. While tracking motion with extra cameras should theoretically prevent failures, it leads to additional complexity and computational burden. To overcome these challenges, we introduce two novel methods to improve multi-camera feature tracking. First, instead of tracking features separately in each camera, we track features continuously as they move from one camera to another. This increases accuracy and achieves a more compact factor graph representation. Second, we select a fixed budget of tracked features across the cameras to reduce back-end optimization time. We have found that using a smaller set of informative features can maintain the same tracking accuracy. Our proposed method was extensively tested using a hardware-synchronized device consisting of an IMU and four cameras (a front stereo pair and two lateral) in scenarios including: an underground mine, large open spaces, and building interiors with narrow stairs and corridors. Compared to stereo-only state-of-the-art visual-inertial odometry methods, our approach reduces the drift rate, relative pose error, by up to 80 % in translation and 39 % in rotation.**

***Index Terms*—Visual-Inertial SLAM; Omnidirectional Vision; Localization**

I. Introduction

STATE estimation is a fundamental capability required for autonomous robot navigation in real-world scenarios. Motion tracking using cameras is very popular due to their low weight, small form factor, and low hardware cost. Specifically, Visual-Inertial Odometry (VIO) methods, which fuse feature tracking from a camera with estimation from an Inertial Measurement Unit (IMU), have now become the standard for odometry on Micro Aerial Vehicles (MAVs) [1]. These systems can also be deployed on a range of platforms, such as handheld rigs, quadrupeds, and vehicles with applications ranging from indoor and outdoor navigation to underground exploration [2] (Fig. 1).

Manuscript received: September, 9, 2021; Revised November, 30, 2021; Accepted December, 6, 2021.

This paper was recommended for publication by Editor Sven Behnke upon evaluation of the Associate Editor and Reviewers' comments. This research was part funded by the EU H2020 Projects THING and Innovate UK-funded ORCA Robotics Hub (EP/R026173/1), a Royal Society University Research Fellowship (Fallon) and a Google DeepMind studentship (Wisth). It has been conducted as part of the ANYmal research community.

[1]Oxford Robotics Institute, Department of Engineering Science, University of Oxford, UK lintong, davidw, mcamurri, mfallon@robots.ox.ac.uk

Digital Object Identifier (DOI): see top of this page.



Fig. 1. We tested our multi-camera odometry algorithm with data from a handheld sensor rig in New College, Oxford (*left*) and the ANYmal quadruped [3] during preparation for the DARPA SubT Challenge [2] (*right*). Video: https://youtu.be/cLWeAT72e0U.

When performing VIO, cameras can track scene information at the frame rate while the IMU can provide high frequency motion estimates between keyframes. Although VIO frameworks are generally effective, they are prone to failure in degenerate scenarios such as over/under exposure, textureless surfaces, and when there is aggressive motion. Another issue with the minimal setup composed of just a monocular camera is that this system results in a single point of failure. Even if IMU measurements can be integrated, the system can quickly diverge within a few seconds of a feature tracking failure.

Fusing information from additional cameras can greatly improve robustness and estimation accuracy at the cost of extra complexity and computational burden. The most obvious benefit of Multi-Camera VIO (MC-VIO) is redundancy, e.g., if one of the cameras suffers from a sudden drop in tracked features, the other cameras can still estimate motion. Furthermore, if the cameras are arranged to have overlapping Fields of View (FoV), a feature could potentially be tracked across the cameras for as long as it is visible in any camera. Multi-camera solutions have been presented in the past [4], [5], [6], however, no proposal has studied cross camera feature tracking.

In this paper, we explore how to effectively track features across all the cameras and how to select the best subset of features to keep the computational time bounded.

A. Contribution

The main contributions of this work are the following:

- A novel factor graph formulation that tightly fuses tracked features from any number of stereo and monocular cameras, along with IMU measurements, in a single consistent optimization process.
- A simple and effective method to track features across cameras with overlapping FoVs to reduce duplicate landmark tracking and improve accuracy.

- A submatrix feature selection (SFS) scheme that selects the best landmarks for optimization with a fixed feature budget. This bounds computational time and achieves the same accuracy compared to using all available features.
- Extensive experimental evaluation across a range of scenarios demonstrating superior robustness, particularly when VIO with an individual camera fails.

The proposed algorithm, VILENS Multi-Camera (VILENS-MC), builds upon our previous VILENS estimation system [7], [8], by fusing multiple cameras and improving front-end feature processing.

II. Related Work

There has been extensive research into monocular and stereo camera VIO. These methods can be categorized into either optimization or filter based approaches. In recent years, optimization based methods such as VINS-Mono [9], ORB-SLAM3 [10], and OKVIS [11] have become popular due to their ability to optimize a trajectory of poses. This is in contrast with methods based on Extended Kalman Filters (EKF) which marginalize all previous states. However, some modern filtering methods have achieved competitive performance via stochastic cloning [12]. For a more detailed review, readers can refer to comparisons in [13] and [14].

MC-VIO has been less extensively studied because of hardware complexity and potentially high computational requirements. Recent work from Jaekel et al. [4] fused two pairs of stereo cameras and accounted for the uncertainty in the extrinsics in both their front- and back-end systems to improve performance. However, they had limited real-world experiments and no ground truth comparisons.

A VIO system by Liu et al. [15], with three pairs of stereo cameras, estimated poses by minimising photometric errors. This work was developed and tested in autonomous driving scenarios where the authors showed the benefit of a multi-camera system when driving at night. However, since a constant velocity motion model was used, their method is not suitable for handheld sensing or robots with highly dynamic motions.

A unique omnidirectional setup was proposed by Seok et al. [5] with four large FoV cameras. The four overlapping image regions were treated as four stereo cameras, but the system did not fully take advantage of the camera setup to track features across the camera pairs.

With a focus on aerial robotics, Müller et al. [16] used two pairs of stereo cameras on an MAV (looking up and down). The images from their wide-angle cameras were split in half and fed into four separate stereo VIO systems running on a Field Programmable Gate Array (FPGA). The VIO outputs and an IMU were then fused together using an EKF in a loosely coupled fashion. However, the experimental results were limited to an indoor office building.

Kuo et al. [17] introduced a more general design for multi-camera Simultaneous Localization and Mapping (SLAM), which involved an adaptive initialization scheme, keyframe selection, and map management. Their system was based on SVO [18] and their approach required minimal parameter tuning. However, their real-world experiments showed little improvement in accuracy when using a multi-camera setup.

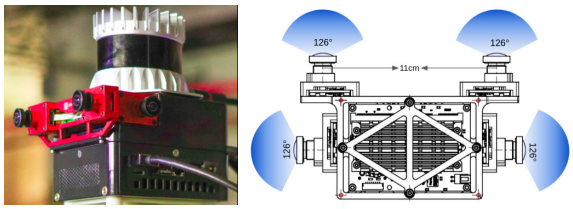


Fig. 2. VILENS-MC is evaluated using a custom built multi-camera handheld device. See details in Section VII-A.

Meanwhile, using a Multi-State Constraint Kalman Filter (MSCKF), Eckenhoff et al. [6] fused six cameras with multiple IMUs. The MSCKF's low computational requirements allowed for asynchronous camera keyframe processing and real-time operation. They presented detailed simulation results, however, real-world experiments were limited to a small lab environment.

We introduce a MC-VIO framework to fuse an arbitrary number of stereo and monocular cameras, with a focus on multi-camera feature selection and tracking.

We are particularly motivated to address common challenges and failure cases in mono and stereo VIO, which have not been a focus of recent studies. Hence, our system is designed and tested in a variety of challenging environments, covering fast and abrupt motions, severe illumination changes, indoor/outdoor scenes, and dark underground environments.

III. Problem Statement

We aim to estimate the position, orientation, and velocity of a mobile platform equipped with an IMU and multiple hardware-synchronized cameras. Fig. 2 shows the multi-camera device used in for the majority of the experimental results.

The relevant reference frames are as follows: the earth-fixed world frame $\mathtt{W}$, the platform-fixed base frame $\mathtt{B}$, the IMU frame $\mathtt{I}$, and n individual camera frames $\mathtt{C}_c$ where $c \in \{1, ..., n\}$.

Unless otherwise specified, the position ${}_{\mathtt{W}}\mathbf{p}_{\mathtt{WB}}$ and orientation $\mathbf{R}_{\mathtt{WB}}$ of the base (with ${}_{\mathtt{W}}\mathbf{T}_{\mathtt{WB}} \in \mathrm{SE}(3)$ as the corresponding homogeneous transform) are expressed in world coordinates; the linear velocity is in the base frame ${}_{\mathtt{B}}\mathbf{v}_{\mathtt{WB}}$, and IMU biases ${}_{\mathtt{I}}\mathbf{b}^g$, ${}_{\mathtt{I}}\mathbf{b}^a$ are expressed in the IMU frame.

A. State and Measurements Definition

The state of the sensor rig at time t_i is defined as follows,

$$\boldsymbol{x}_i \triangleq [\mathbf{R}_i, \mathbf{p}_i, \mathbf{v}_i, \mathbf{b}_i^g \ \mathbf{b}_i^a] \in \mathrm{SO}(3) \times \mathbb{R}^{12} \tag{1}$$

where: $\mathbf{R}_i$ is the orientation, $\mathbf{p}_i$ is the position, $\mathbf{v}_i$ is the linear velocity, and $\mathbf{b}_i^g$, $\mathbf{b}_i^a$ are, respectively, the usual IMU gyroscope and accelerometer biases. In addition to the states, we track point landmarks $\boldsymbol{m}_\ell$ as triangulated visual features.

The objective is to estimate the optimized trajectory $\mathcal{X}_k$ of all states $\boldsymbol{x}_i$ and landmarks $\boldsymbol{m}_\ell$ visible up to the current time t_k within a fixed lag smoothing window.

The measurements from the set of multiple cameras $\mathcal{C}$ and an IMU $\mathcal{I}$ are hardware synchronized but received at different frequencies. We define $\mathcal{Z}_k$ as the full set of measurements received within the smoothing window.

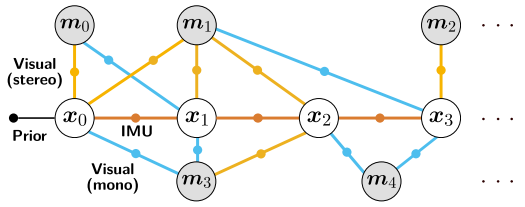


Fig. 3. Sliding window factor graph structure, showing prior, visual, and preintegrated IMU factors. Landmarks can be tracked across both stereo and mono cameras to create longer feature tracks and improve accuracy.

B. Maximum-a-Posteriori Estimation

We maximize the likelihood of the measurements $\mathcal{Z}_k$, given the history of states $\mathcal{X}_k$,

$$\mathcal{X}_k^* = \arg\max_{\mathcal{X}_k} p(\mathcal{X}_k|\mathcal{Z}_k) \propto p(\mathcal{X}_0)p(\mathcal{Z}_k|\mathcal{X}_k) \tag{2}$$

The measurements are formulated as conditionally independent and corrupted by white Gaussian noise. Therefore, Eq. (2) can be expressed as the following least squares minimization [19],

$$\mathcal{X}_k^* = \arg\min_{\mathcal{X}_k} \|\mathbf{r}_0\|^2_{\Sigma_0} + \sum_{i\in\mathsf{K}_k}\left(\|\mathbf{r}_{\mathcal{I}_{ij}}\|^2_{\Sigma_{\mathcal{I}_{ij}}} + \sum_{\ell\in\mathsf{M}_i}\|\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{m}_\ell}\|^2_{\Sigma_{\boldsymbol{x}_i,\boldsymbol{m}_\ell}}\right) \tag{3}$$

where $\mathcal{I}_{ij}$ are the IMU measurements between t_i and t_j and K_k are all the keyframes indices in the sliding window up to t_k. Each term is the residual associated to a factor type, weighted by the inverse of its covariance matrix. The residuals include prior, IMU, and visual landmark factors.

IV. Factor Graph Formulation

In this section, we describe the measurements, residuals, and covariances of the IMU and visual factors shown in Fig. 3.

A. Preintegrated IMU Factors

We follow the standard manner of IMU measurement preintegration from [1] to constrain the pose, velocity, and biases between two consecutive nodes of the graph, providing high frequency state updates between nodes. The residual has the following form,

$$\mathbf{r}_{\mathcal{I}_{ij}} = \left[\mathbf{r}^\mathsf{T}_{\Delta\mathbf{R}_{ij}}, \mathbf{r}^\mathsf{T}_{\Delta\mathbf{v}_{ij}}, \mathbf{r}^\mathsf{T}_{\Delta\mathbf{p}_{ij}}, \mathbf{r}_{\mathbf{b}^a_{ij}}, \mathbf{r}_{\mathbf{b}^g_{ij}}\right] \tag{4}$$

For a detailed definition of the above residuals, see [1].

B. Stereo and Mono Landmark Factors

We define a visual landmark in Euclidean space as $\boldsymbol{m}_\ell \in \mathbb{R}^3$. Given the platform pose $\mathbf{T}_i$ (for simplicity, we omit the fixed transform between base and camera), we can project $\boldsymbol{m}_\ell$ onto the image plane with the function $\pi : \mathrm{SE}(3) \times \mathbb{R}^3 \mapsto \mathbb{R}^2$, resulting in the projected coordinates $(u_\ell, v_\ell) \in \mathbb{R}^2$ on the image plane (orange/green circles in Fig. 4). Thus, the residual at state $\boldsymbol{x}_i$ for landmark $\boldsymbol{m}_\ell$ is defined as [7],

$$\mathbf{r}_{\boldsymbol{x}_i,\boldsymbol{m}_\ell} = \begin{pmatrix} \pi^L_u(\mathbf{T}_i, \boldsymbol{m}_\ell) - u^L_{i,\ell} \\ \pi^R_u(\mathbf{T}_i, \boldsymbol{m}_\ell) - u^R_{i,\ell} \\ \pi_v(\mathbf{T}_i, \boldsymbol{m}_\ell) - v_{i,\ell} \end{pmatrix} \tag{5}$$

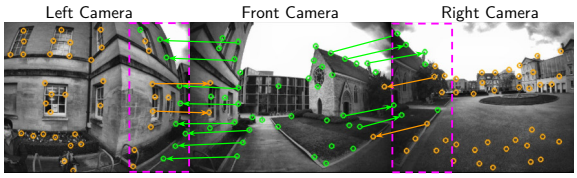


Fig. 4. VILENS-MC takes advantage of any overlapping image regions (purple rectangles) in a multi-camera setup to track features across cameras. This increases feature track length and avoids tracking the same feature independently in different cameras. The arrows indicate features being tracked from one image to another.

where $(u^L, v), (u^R, v)$ are the pixel locations of the detected landmark. $\Sigma_{\boldsymbol{m}}$ is computed using an uncertainty of 0.25 pixels. We also account for lens distortion using the covariance warping method from [8]. For monocular landmarks, only the first and the last rows of Eq. (5) are used.

The location of the landmarks detected by the stereo camera pair is initialized using stereo triangulation. For landmarks detected in monocular cameras, we triangulate the feature location over the last N_{obs} frames using the Direct Linear Transform (DLT) algorithm from [20].

V. Cross Camera Feature Tracking

Most multi-camera systems do not take advantage of the overlapping FoVs of their setups, which produce common image regions that allow for cross-camera feature matching.

We propose a simple and effective method to continuously track features across different cameras which we call Cross Camera Feature Tracking (CCFT). CCFT can avoid the optimization of redundant landmarks by tracking the same feature in different cameras simultaneously. As shown in Fig. 3, we continually add constraints between states $\boldsymbol{x}_i$ and landmarks $\boldsymbol{m}_\ell$ even as they are tracked across cameras. For example, landmark $\boldsymbol{m}_1$ is first tracked in a stereo camera at $\boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2$, then tracked in a monocular camera at $\boldsymbol{x}_3$.

A common approach to perform this feature matching would be via feature descriptors. However, this would be computationally expensive for real-time, high-frequency VIO on a mobile platform. For example, the combination of feature extraction and stereo matching in ORB-SLAM2 [21] takes ~24.8 ms which is significant for a 30 Hz image stream. Instead, we make use of the known camera extrinsics and the preintegrated IMU measurements between states to associate features geometrically.

When feature depth is available at the current image time t_k (e.g., from stereo camera triangulation), then we can directly project the visual landmark $\boldsymbol{m}_\ell$ from camera $\mathtt{C}_i$ into camera $\mathtt{C}_j$ via the extrinsic transformation between cameras $\mathbf{T}_{\mathtt{C}_i\mathtt{C}_j}$,

$$(u, v)_{\ell,c_j,t_k} = \pi_{c_j}\left(\mathbf{T}_{\mathtt{C}_j\mathtt{C}_i}\ {}_{\mathtt{C}_i}\boldsymbol{m}_{\ell,t_k}\right) \tag{6}$$

Alternatively, if the feature depth is not directly available at the current timestamp (e.g., if the feature is only tracked in a monocular camera), then we instead use the preintegrated IMU measurements to estimate the landmark location at the current time,

$$(u, v)_{\ell,c_j,t_k} = \pi_{c_j}\left(\mathbf{T}_{\mathtt{C}_j\mathtt{C}_i}\ \hat{\mathbf{T}}_{t_k,t_{k-1}}\ {}_{\mathtt{C}_i}\boldsymbol{m}_{\ell,t_{k-1}}\right) \tag{7}$$

where $\hat{\mathbf{T}}_{t_k,t_{k-1}}$ is the estimated transform between the previous camera pose at t_{k-1} and the current time t_k.

This process is completed for each feature in the overlapping image regions. The projected feature location is refined by matching it to the closest image feature in camera c_j, using a Euclidean distance metric. This is a highly effective method in practice, assuming there is a good extrinsic calibration. Any incorrect associations are handled by the optimizer using robust cost functions.

Fig. 4 shows an example of the feature matches using CCFT. The purple areas highlight the overlapping image regions where features can be tracked across cameras. The CCFT method typically reduces the number of landmarks added into the optimization back-end and gives improved estimation accuracy. This is discussed further in Section IX-A.

VI. Submatrix Feature Selection

In general, increasing the number of features tracked in a VIO system improves the estimation accuracy [15]. However, it also increases computation, eventually reaching a point where the algorithm can fail due to computational constraints. This is a particular problem in multi-camera systems, where more features can be tracked than the optimizer can handle. Thus, it is important to track and optimize only the best features.

Specifically, our SFS algorithm is based on [22], where the authors presented a feature selection algorithm to reduce the computational cost of active map-to-frame feature matching. We instead apply this algorithm to MC-VIO and implement changes to improve numerical stability. The aim is to maintain accuracy while reducing the optimization time of MC-VIO.

In this section, we first outline the construction of the joint feature Jacobian and covariance matrices. Then, we describe the *Max-logDet* algorithm to select the most representative subset of tracked features to be optimized.

A. Construct Joint Feature Jacobian Matrix

The optimization aims to minimize the cost function defined in Eq. (3). Since we are interested in selecting the best features in each frame k, we focus on improving the conditioning of the landmark residual, where $\boldsymbol{z}_\ell$ is the feature location of (u, v),

$$\mathcal{X}^* = \arg\min_{\mathcal{X}} \sum_{\ell \in \mathsf{M}} \|\mathbf{r}_{\boldsymbol{x},\boldsymbol{m}_\ell}\|^2_{\Sigma_{\boldsymbol{m}_\ell}} \tag{8}$$

$$= \arg\min_{\mathcal{X}} \sum_{\ell \in \mathsf{M}} \|\pi(\mathbf{T}, \boldsymbol{m}_\ell) - \boldsymbol{z}_\ell\|^2_{\Sigma_{\boldsymbol{m}_\ell}} \tag{9}$$

Stacking $\boldsymbol{m}_\ell$, $\boldsymbol{z}_\ell$ into column matrices $\mathbf{M}$, $\mathbf{Z}$ we get,

$$\mathcal{X}^* = \arg\min_{\mathcal{X}} \|\Pi(\mathbf{T}, \mathbf{M}) - \mathbf{Z}\|^2_\Sigma \tag{10}$$

Applying the method from [22], we perform a first order linearization around the initial value $\mathbf{T}_g$,

$$\|\Pi(\mathbf{T}, \mathbf{M}) - \mathbf{Z}\|^2_\Sigma \approx \|\Pi(\mathbf{T}_g, \mathbf{M}) + \mathbf{H}_\mathbf{T}(\mathbf{T} \ominus \mathbf{T}_g) - \mathbf{Z}\|^2_\Sigma \tag{11}$$

where $\mathbf{H}_\mathbf{T}$ is the pose Jacobian linearized about the value $\mathbf{T}_g$, and $\ominus$ is the Lie group difference operator between poses [8].

To minimize Eq. (11), we can use Gauss-Newton optimization to iteratively update the pose estimate,

$$\mathbf{T}_{g+1} = \mathbf{T}_g - \mathbf{H}^+_\mathbf{T} \left(\Pi(\mathbf{T}_g, \mathbf{M}) - \mathbf{Z}\right) \tag{12}$$

where $\mathbf{H}^+_\mathbf{T}$ is the Moore-Penrose pseudo-inverse of $\mathbf{H}_\mathbf{T}$. This method has converged (i.e., reached minimal error) when the update step is approximately zero, i.e.,

$$\boldsymbol{\epsilon}_\mathbf{T} = \mathbf{H}^+_\mathbf{T} \left(\Pi(\mathbf{T}_g, \mathbf{M}) - \mathbf{Z}\right) \approx 0 \tag{13}$$

Performing another first order linearization around $\mathbf{M}_g$,

$$\boldsymbol{\epsilon}_\mathbf{T} \approx \mathbf{H}^+_\mathbf{T} \left(\Pi(\mathbf{T}_g, \mathbf{M}_g) - \mathbf{Z} + \mathbf{H}_\mathbf{M}(\mathbf{M} - \mathbf{M}_g)\right) \tag{14}$$

$$\boldsymbol{\epsilon}_\mathbf{T} = \mathbf{H}^+_\mathbf{T} \left(\boldsymbol{\epsilon}_\boldsymbol{Z} - \mathbf{H}_\mathbf{M} \boldsymbol{\epsilon}_\mathbf{M}\right) \tag{15}$$

where $\mathbf{H}_\mathbf{M}$ is the landmark-to-image projection Jacobian. We define the measurement error as $\boldsymbol{\epsilon}_\boldsymbol{Z} = \Pi(\mathbf{T}_g, \mathbf{M}_g) - \mathbf{Z}$, and the landmark error as $\boldsymbol{\epsilon}_\mathbf{M} = \mathbf{M} - \mathbf{M}_g$.

Without loss of generality, the measurement error can be modeled as $\boldsymbol{\epsilon}_{\boldsymbol{z}_\ell} \sim \mathcal{N}(0, \Sigma_{z_\ell})$ which is influenced by image processing parameters and noise. However, if biased landmark errors exist, then the mean of error distribution is non-zero. This could be caused by batch perturbation of the landmark positions by incorrect camera poses, or a slight scale error. Hence the landmark error is modeled as non-zero-mean i.i.d. Gaussian, $\boldsymbol{\epsilon}_{\boldsymbol{m}_\ell} \sim \mathcal{N}(\mu_{\boldsymbol{m}_\ell}, \Sigma_{\boldsymbol{m}_\ell})$, such that the expected value of $\boldsymbol{\epsilon}_\mathbf{T}$ is,

$$\mathbb{E}[\boldsymbol{\epsilon}_\mathbf{T}] = \mathbb{E}\left[\mathbf{H}^+_\mathbf{T} \left(\boldsymbol{\epsilon}_\boldsymbol{Z} - \mathbf{H}_\mathbf{M} \boldsymbol{\epsilon}_\mathbf{M}\right)\right] \tag{16}$$

$$\mathbb{E}[\boldsymbol{\epsilon}_\mathbf{T}] = \mathbf{H}^+_\mathbf{T} \mathbf{H}_\mathbf{M} \mathbf{1}_n \mu_\mathbf{M} \tag{17}$$

$$\mathbf{H}^+_\mathbf{M} \mathbf{H}_\mathbf{T} \mathbb{E}[\boldsymbol{\epsilon}_\mathbf{T}] = \mathbf{1}_n \mu_\mathbf{M} \tag{18}$$

where $\mathbf{1}_n$ is a tall matrix made of identity matrices to transform the dimension of $\mu_\mathbf{M}$. $\mathbf{H}_\mathbf{M}$ consists of n diagonal blocks made up of individual landmark Jacobians $\mathbf{H}_{\boldsymbol{m}_\ell} \in \mathbb{R}^{2\times 3}$, where n is the number of landmarks with known 3D positions. $\mathbf{H}_\mathbf{T}$ consists of pose Jacobians for each landmark $\mathbf{H}_{\mathbf{T},\boldsymbol{m}_\ell} \in \mathbb{R}^{2\times 6}$.

To obtain a joint Jacobian matrix $\mathbf{H}_J$, we add an additional row of zeros to $\mathbf{H}_{\mathbf{T},\boldsymbol{m}_\ell}$ and a row of $[0, 0, 1]$ to $\mathbf{H}_{\boldsymbol{m}_\ell}$. This makes $\mathbf{H}_{\mathbf{T},\boldsymbol{m}_\ell}$ invertible without changing the structure of the least squares problem [23]. Performing block-wise multiplication results in the matrix $\mathbf{H}_J$,

$$\mathbf{H}_J = \begin{bmatrix} \mathbf{H}^{-1}_{\boldsymbol{m}_0} \mathbf{H}_{\mathbf{T},\boldsymbol{m}_0} & \dots & \mathbf{H}^{-1}_{\boldsymbol{m}_{n-1}} \mathbf{H}_{\mathbf{T},\boldsymbol{m}_{n-1}} \end{bmatrix}^\mathsf{T} \tag{19}$$

from which the simplified pose covariance matrix follows,

$$\Sigma_\mathbf{T} = \mathbf{H}^+_J (\mathbf{H}^+_J)^\mathsf{T} = (\mathbf{H}^\mathsf{T}_J \mathbf{H}_J)^{-1} \tag{20}$$

B. Submatrix Selection

Since the overall pose error depends on the spectral properties of $\mathbf{H}_J$, we aim to find a submatrix $\mathbf{H}_{sub}$ (i.e., subset of features) that best preserves this distribution. This can be understood as finding row blocks in $\mathbf{H}_J$ to maximize the norm of the selected submatrix.

We adapt the stochastic sampling method presented in [22] for feature selection. We choose the *Max-logdet* metric since it best approximates the original full feature set in visual odometry [23], [24]. Additionally, the combination of greedy (deterministic) and stochastic sampling (randomized acceleration) gives the best approximation ratio of any polynomial time algorithm [25].

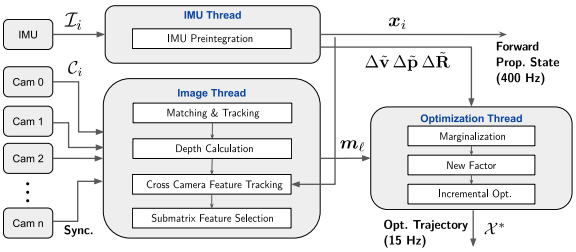


Fig. 5. Overview of the VILENS-MC architecture. The IMU and cameras are handled in separate threads by front-end measurement handlers. The back-end produces both a high frequency forward-propagated output and a lower frequency optimized output.

We note that since $\mathbf{H}_{J_i} \in \mathbb{R}^{2\times 6}$ is not a full rank matrix, the determinant of its square $\mathbf{H}_{J_i}^{\mathsf{T}}\mathbf{H}_{J_i}$ would be zero. In contrast to [22], we add a small diagonal matrix $\lambda\mathbf{I} \in \mathbb{R}^{6\times 6}$ to the $\log\det$ calculation,

$$\mathbf{H}_{J_i} = \arg\max \left(\log\det(\mathbf{H}_{J_i}^{\mathsf{T}}\mathbf{H}_{J_i} + \mathbf{H}_{sub}^{\mathsf{T}}\mathbf{H}_{sub} + \lambda\mathbf{I})\right) \quad (21)$$

This makes the squared matrix full rank with a non-zero determinant while preserving its spectral properties. An alternative method would be projecting $\mathbf{H}_{J_i}^{\mathsf{T}}\mathbf{H}_{J_i}$ onto a full rank subspace, which we will consider in future work.

We apply this algorithm to each camera individually in our multi-camera setup. An ablation study showing how this method maintains high accuracy and improves optimization time is presented in Section IX-B.

VII. Implementation

An overview of the system architecture is shown in Fig. 5. Three parallel threads perform IMU preintegration, camera image processing, and optimization, respectively. The system outputs the optimized state estimate from the factor graph at camera keyframe frequency (typically 15 Hz) for navigation or mapping, while a forward-propagated state at IMU frequency is also produced at 400 Hz for high frequency tasks.

The factor graph is solved using fixed lag smoothing based on the efficient incremental optimization solver iSAM2 from the GTSAM library [19]. For all of our experiments, we used a lag time of 3.5 s. To reduce the effect of outliers, the visual factors are added to the graph using the Dynamic Covariance Scaling (DCS) robust cost function [26].

A. Hardware and Calibration

The sensor used for our experiments is the Alphasense multi-camera development kit from Sevensense Robotics AG, shown in Fig. 2. An onboard FPGA synchronizes the IMU and four grayscale fisheye cameras – a frontal stereo pair with an 11 cm baseline and two lateral cameras. Each camera has a FoV of 126° × 92.4° and a resolution of 720 × 540 px. This configuration produced an overlapping FoV between the front and side cameras of about 36°. The cameras and the embedded cellphone-grade IMU were operated at 30 Hz and 200 Hz, respectively. Camera-IMU extrinsic and intrinsic calibration was conducted offline using the Kalibr [27] toolbox. The Ouster OS0-128 lidar shown in the figure was used only for ground truth generation and all sensors were hardware synchronized with an Intel NUC via Precision Time Protocol (PTP).



Fig. 6. Test datasets: *Top:* Newer College Dataset [28]. *Bottom Left:* Oxford Mathematical Institute. *Bottom Right:* A dark underground limestone mine.

B. Initialization

We initialize the IMU biases by averaging the first 1 s of data at system start up (assuming the IMU is stationary).

To solve the scale initialization problem, which is often present in monocular visual odometry systems, we combine preintegrated IMU measurements and depth from the stereo camera pair. Notably, the CCFT method allows features from the stereo camera to flow into the monocular camera, speeding up the depth initialization process.

C. Visual Feature Tracking

To get an even feature distribution across each image, we split them into 3×3 tiles before applying the FAST feature detector to each segment. These features are tracked between frames using the KLT feature tracker and across cameras using the CCFT method described above. Outliers are rejected using RANSAC-based methods. Thanks to the incremental optimization and multi-threading, every second frame is used as a keyframe, achieving a 15 Hz nominal output.

VIII. Experimental Results

We evaluated our algorithm on a variety of challenging indoor and outdoor datasets, varying from narrow corridors to large open spaces. We compared our algorithm to state-of-the-art methods, evaluating both quantitative and qualitative performance. We also include ablation studies showing how our proposed contributions reduced Relative Pose Error (RPE) and decreased computation time. Finally, we demonstrated the versatility of this approach by applying VILENS-MC to a quadruped robot operating in an unlit mine.

A. Datasets

Fig. 6 gives an overview of the datasets used for evaluation. These include additional multi-camera experiments collected for the Newer College Dataset (NCD) [28], operation in an

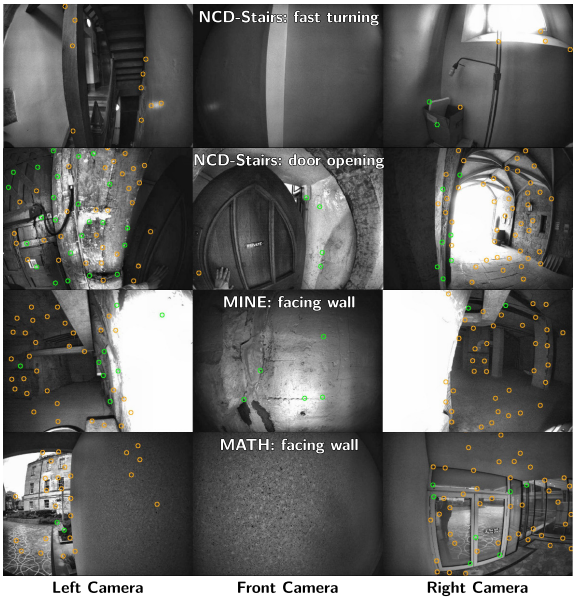


Fig. 7. Examples of challenging scenes where it was difficult to track features. Orange and green circles are features tracked by monocular and stereo cameras, respectively (some have been tracked across images). Note that stereo features were mostly absent or rejected in these examples.

unlit underground limestone mine (MINE), and a large circuit around the Oxford Mathematical Institute (MATH).

These datasets were collected using the handheld multi-camera device described in Section VII-A. They were specifically chosen to test the limits of VIO systems and include challenges such as aggressive shaking of the device (up to 5.5 rad/s), severe illumination changes, and loss of visual feature tracking in one or more cameras (see Fig. 7). All datasets were collected at a fast walking pace of ∼1.5 m/s. The specific experiments are:

- **NCD-Quad:** Moving between narrow corridors and a large open college quadrangle, with strong illumination changes caused by bright sunlight (244 m, 3 min).
- **NCD-Stairs:** Traversing a very narrow staircase, including a sequence where a door is opened directly in front of the camera (59 m, 2 min).
- **MINE:** A medium scale, dark underground environment with regular illumination changes due to onboard lighting (236 m, 3 min).
- **MATH:** A large scale outdoor environment with aggressive shaking of the device up to 5.5 rad/s (329 m, 4 min).

The multi-camera experiments which extend NCD (NCD-Quad, NCD-Stairs) have been publicly released. [1]

To generate the ground truth, Iterative Closest Point (ICP) registration was used to align the current lidar scan to detailed prior maps collected from a commercial laser scanner. The high frequency motion estimate from the IMU was used to carefully remove lidar motion distortion [7]. For an in-depth discussion on ground truth generation, refer to [29].

[1] Available at https://ori-drs.github.io/newer-college-dataset/.

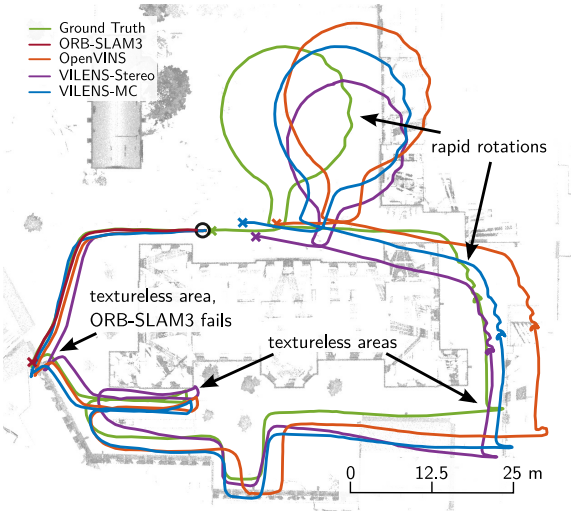


Fig. 8. Top-down view of **MATH** dataset comparing the estimated trajectory of ORB-SLAM3, OpenVINS, and VILENS-MC against the ground truth. A black circle marks the start of the trajectory, while colored crosses indicate the last pose of each trajectory. Textureless locations, where cameras were facing walls, are shown in Fig. 7, row 4.

10 m Relative Pose Error (RPE) – Translation [m] / Rotation [°]				
	OpenVINS	**ORB-SLAM3**	**VILENS-Stereo**	**VILENS-MC**
Cameras	2 (Stereo)	2 (Stereo)	2 (Stereo)	4
NCD-QUAD	1.01 / 1.26	**0.23** / 0.91	0.30 / 1.24	0.31 / **0.82**
NCD-STAIRS	0.33 / 3.05	0.20* / 3.04	0.43 / 5.85	**0.16** / **2.56**
MINE	0.98* / 2.10	Fail	0.41* / 3.57	**0.20** / **1.65**
MATH	0.65* / 1.69	Fail	0.56* / 1.57	**0.26** / **1.03**

TABLE I
PERFORMANCE COMPARISON BETWEEN DIFFERENT ALGORITHMS
* = ONE OR MORE INSTABILITIES OCCURRED

B. Quantitative Analysis

Table I shows the RPE over 10 m for OpenVINS [12], ORB-SLAM3 [10] and VILENS [8] (which our system builds upon). We refer to the configuration of VILENS which uses only stereo and IMU input as *VILENS-Stereo*. Note we were not aware of any open-source MC-VIO algorithm for comparison.

Compared to our baseline (VILENS-Stereo), we reduced mean RPE by 45 % / 50 % in translation/rotation, while tracking fewer features due to SFS. VILENS-MC also outperformed other state-of-the-art stereo VIO algorithms by up to 80 % / 39 % in translation/rotation.

In benign VIO conditions, such as those found in the NCD-Quad dataset, the performance of VILENS-MC was similar to state-of-the-art systems. In particular, ORB-SLAM3 appeared to outperform the other algorithms by taking advantage of its local map matching SLAM system to correct for failure events when the camera revisited a previous location.

However, in the MINE and MATH datasets, where there were few or no stereo features, VILENS-MC performed well, even when the other stereo VIO systems failed or had significant drift. This highlights a key benefit of the tight fusion of multiple cameras, the natural robustness to scene degradation.

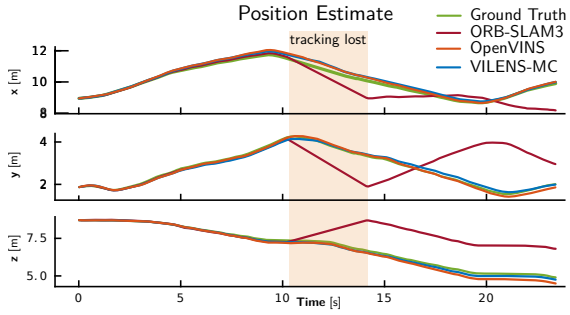


Fig. 9. ORB-SLAM3 lost tracking and failed to estimate odometry for 4 s in NCD-Stairs (scenario shown in Fig. 7 Row 3). Note that elevation (z axis) is properly tracked afterwards.

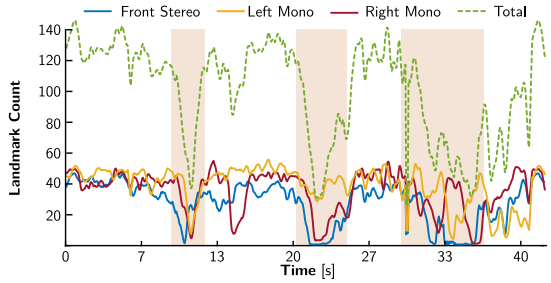


Fig. 10. A 40 s section from the **MINE** dataset showing landmark count per each camera: front stereo pair (solid blue), left mono camera (solid yellow), right mono camera (solid red). The total number of landmarks is dashed green.

C. Degeneracy and Failure Analysis

1) MATH: A top-down view of the trajectories in the MATH dataset is shown in Fig. 8. In challenging scenarios where stereo camera feature tracking failed (marked as "textureless areas"), stereo VIO algorithms continued to operate but relied on the IMU only, causing the RPE to quickly increase, while VILENS-MC maintained robust estimation by utilizing multiple cameras looking in different directions.

2) NCD-Stairs: Fig. 9 shows an extract from the NCD-Stair dataset. Note how ORB-SLAM3 suffered from significant drift when no features were detected in the stereo camera for 4 s until it re-initialized (orange area). Even though the average RPE over the entire NCD-Stair sequence remained low, a gap in estimations would be undesirable for an autonomous robot.

3) MINE: Fig. 10 shows the number of tracked landmarks during a 40 s interval in the MINE dataset. During this time, several instances occurred where feature tracking failed in either one or two cameras. Failures occurred when the device entered a dark room (∼10 s), when it approached a wall and turned on the spot (∼20 s, shown in Fig. 7 row 3), and when it entered a dark corner where only one camera could observe features at a time (∼30 s). By tracking features across multiple cameras, VILENS-MC maintains accurate estimation throughout these events.

	Relative Pose Error – Translation [m] / Rotation [°]				
	Baseline	**CCFT**	**Baseline** (90 feat.)	**Baseline** (150 feat.)	**SFS** (90 feat.)
NCD-QUAD	**0.27** / **0.77**	0.34 / 0.88	0.37 / 1.02	0.34 / 0.88	**0.31** / **0.82**
NCD-STAIR	0.17 / 3.60	**0.16** / **2.85**	0.28 / 5.06	**0.16** / **2.85**	0.18 / 3.03
MINE	0.25 / 1.91	**0.21** / **1.64**	0.34 / 2.70	0.21 / **1.71**	**0.20** / 1.75
MATH	0.42 / 1.01	**0.28** / **1.01**	0.33 / 1.16	0.28 / **1.01**	**0.26** / 1.03

TABLE II
ABLATION STUDIES: CROSS CAMERA FEATURE TRACKING (CCFT) AND SUBMATRIX FEATURE SELECTION (SFS)

IX. Ablation Study

A. Cross Camera Feature Tracking

In this section, we evaluate the benefit of the CCFT method (Section V). CCFT enables continuous feature tracking over longer periods and adds additional constraints to landmarks.

Table II summarizes the effect this method had on mean RPE (over 10 m), showing that on average CCFT decreased both translational and rotational RPE on all datasets except for translation in NCD-QUAD. We attribute this to the system's susceptibility to extrinsic multi-camera calibration errors.

Additionally, CCFT can reduce the number of landmarks in the optimization by ∼10 %. Since our hardware setup (ref. Section VII-A) had only a small overlapping FoV between frontal and lateral cameras, we expect to see a greater benefit with a larger FoV overlap.

B. Submatrix Feature Selection

This ablation study demonstrates the benefit of SFS. The baseline approach simply spreads features evenly across the image based on their FAST score and mutual pixel distance.

A comparison between the baseline and proposed SFS methods is shown in Table II. The first two columns show the baseline with 90 and 150 features tracked and optimized across all cameras, while the SFS method selected only 90 out of 150 tracked features to optimize.

By selecting only the most representative features, SFS outperformed the baseline (90 features) by up to 36 %, even though both methods optimized the same number of features.

Crucially, SFS lowers computation without sacrificing accuracy. Comparing the baseline (150 features) to SFS (90 features), the latter achieves similar accuracy with a 20 % reduction in optimization time (53.7 ms to 43.0 ms). The standard deviation of the computation was also reduced by 24.3 %. This demonstrates that SFS can achieve similar accuracy with lower and more consistent computational requirements.

X. Demonstration on Legged Robots

To demonstrate the versatility of our approach, we tested VILENS-MC on the ANYmal C quadruped robot equipped with an Alphasense for the DARPA Subterranean Challenge [2]. In the trajectory shown in Fig. 11 the robot autonomously explored the same environment as the MINE dataset. VILENS-MC achieved almost the same accuracy as for the handheld dataset (0.27 m, 2.65° RPE), indicating that performance was similar across the different platforms.

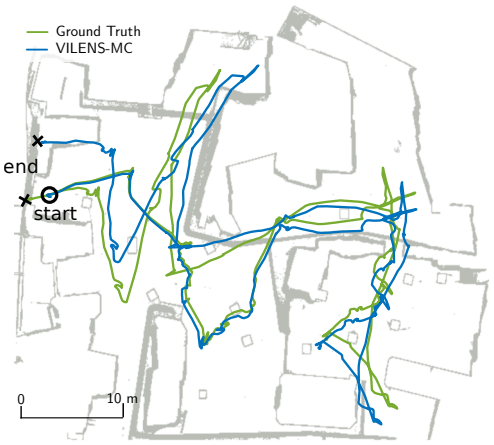


Fig. 11. State estimate on an ANYmal C quadruped robot autonomously exploring an underground mine over ∼10 min. Fig. 6 shows the robot's onboard lighting which caused strong illumination changes in the images.

XI. Conclusion

We presented a novel factor graph formulation for state estimation that tightly fuses an arbitrary number of cameras. The joint optimization of IMU measurements with monocular and camera constraints enables graceful handling of degenerate scenarios without requiring hard switching between cameras. This also simplifies the initialization of the individual monocular cameras. We have demonstrated comparable tracking performance to state-of-the-art VIO systems in benign conditions and better performance in extreme situations, such as in the case of aggressive motions, loss of tracking in one or more cameras, or abrupt light changes.

We proposed two algorithmic components specific to multi-camera odometry: cross camera feature tracking and submatrix feature selection. Ablation studies showed that these components improve accuracy and reduce optimization time.

Overall, we have presented a robust MC-VIO system that is capable of handling various challenging real-world environments while consistently achieving accurate state estimation. Future work would consider including camera extrinsics into the optimization.

References

[1] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-Manifold Preintegration for Real-Time Visual-Inertial Odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[2] M. Tranzatto, *et al.*, "CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge," *Field Robotics*, 2021.

[3] M. Hutter, *et al.*, "ANYmal – A Highly Mobile and Dynamic Quadrupedal Robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 38–44.

[4] J. Jaekel, J. G. Mangelson, S. Scherer, and M. Kaess, "A Robust Multi-Stereo Visual-Inertial Odometry Pipeline," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

[5] H. Seok and J. Lim, "ROVINS: Robust Omnidirectional Visual Inertial Navigation System," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6225–6232, 2020.

[6] K. Eckenhoff, P. Geneva, and G. Huang, "MIMC-VINS: A Versatile and Resilient Multi-IMU Multi-Camera Visual-Inertial Navigation System," *IEEE Transactions on Robotics*, pp. 1–20, 2021.

[7] D. Wisth, M. Camurri, S. Das, and M. Fallon, "Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1004–1011, 2021.

[8] D. Wisth, M. Camurri, and M. Fallon, "VILENS: Visual, inertial, lidar, and leg odometry for all-terrain legged robots," *arXiv preprint arXiv:2107.07243*, 2021.

[9] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[10] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM," *IEEE Transactions on Robotics*, pp. 1–17, 2021.

[11] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[12] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "OpenVINS: A Research Platform for Visual-Inertial Estimation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 4666–4672.

[13] J. Delmerico and D. Scaramuzza, "A Benchmark Comparison of Monocular Visual-Inertial Odometry Algorithms for Flying Robots," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 2502–2509.

[14] M. He, C. Zhu, Q. Huang, B. Ren, and J. Liu, "A review of monocular visual odometry," *The Visual Computer*, vol. 36, no. 5, pp. 1053–1065, 2020.

[15] "Towards Robust Visual Odometry with a Multi-Camera System," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.

[16] M. G. Müller, *et al.*, "Robust visual-inertial state estimation with multiple odometries and efficient mapping on an MAV with ultra-wide FOV stereo vision," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3701–3708, 2018.

[17] J. Kuo, M. Muglikar, Z. Zhang, and D. Scaramuzza, "Redesigning SLAM for Arbitrary Multi-Camera Systems," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 2116–2122.

[18] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[19] F. Dellaert and M. Kaess, "Factor Graphs for Robot Perception," *Foundations and Trends in Robotics*, vol. 6, pp. 1–139, 2017.

[20] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[21] R. Mur-Artal and J. D. Tardos, "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[22] Y. Zhao and P. A. Vela, "Good Feature Matching: Toward Accurate, Robust VO/VSLAM with Low Latency," *IEEE Transactions on Robotics*, vol. 36, no. 3, pp. 657–675, 2020.

[23] ——, "Good Feature Selection for Least Squares Pose Optimization in VO/VSLAM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 1183–1189.

[24] L. Carlone and S. Karaman, "Attention and Anticipation in Fast Visual-Inertial Navigation," *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 1–20, 2019.

[25] B. Mirzasoleiman, A. Badanidiyuru, A. Karbasi, J. Vondrák, and A. Krause, "Lazier than lazy greedy," *Proceedings of the National Conference on Artificial Intelligence*, vol. 3, pp. 1812–1818, 2015.

[26] K. MacTavish and T. D. Barfoot, "At All Costs: A Comparison of Robust Cost Functions for Camera Correspondence Outliers," in *Conference on Computer and Robot Vision*, 2015, pp. 62–69.

[27] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, "Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 4304–4311.

[28] L. Zhang, M. Camurri, and M. Fallon, "Multi-Camera LiDAR Inertial Extension to the Newer College Dataset," *arXiv preprint arXiv:2112.08854*, 2021.

[29] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The Newer College Dataset: Handheld LiDAR, Inertial and Vision with Ground Truth," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

Statement of Authorship for joint/multi-authored papers for PGR thesis

To appear at the end of each thesis chapter submitted as an article/paper

The statement shall describe the candidate's and co-authors' independent research contributions in the thesis publications. For each publication there should exist a complete statement that is to be filled out and signed by the candidate and supervisor **(only required where there isn't already a statement of contribution within the paper itself).**

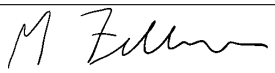
Title of Paper	Balancing the Budget: Feature Selection and Tracking for Multi-Camera Visual-Inertial Odometry
Publication Status	Submitted for publication
Publication Details	L. Zhang, D. Wisth, M. Camurri, and M. Fallon, "Balancing the Budget: Feature Selection and Tracking for Multi-Camera Visual-Inertial Odometry," *IEEE Robotics and Automation Letters (Submitted)*, 2022.

Student Confirmation

Student Name:	David Wisth		
Contribution to the Paper	My contributions to the paper were: • Significant role in developing the novelty for the paper. • Significant role in the algorithm implementation and experimental results, including extending parts of VILENS to support this work. • Providing the code for ground truth generation. • Assisted with writing and proof-reading the paper.		
Signature		Date	30/09/2021

Supervisor Confirmation

By signing the Statement of Authorship, you are certifying that the candidate made a substantial contribution to the publication, and that the description described above is accurate.

Supervisor name and title: Professor Maurice Fallon			
Supervisor comments The description above is correct and accurate			
Signature		Date	4/10/2021

This completed form should be included in the thesis, at the end of the relevant chapter.

Bibliography

[1] F. Dellaert, "Factor graphs and gtsam: A hands-on introduction," Georgia Institute of Technology, Tech. Rep., 2012. [Online]. Available: http://hdl.handle.net/1853/45226

[2] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. J. Leonard, and F. Dellaert, "iSAM2: Incremental smoothing and mapping using the bayes tree," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 216–235, 2012.

[3] J. Zhang and S. Singh, "Low-drift and real-time lidar odometry and mapping," *Autonomous Robots*, vol. 41, no. 2, pp. 401–416, 2017.

[4] M. Bloesch, M. Burri, H. Sommer, R. Siegwart, and M. Hutter, "The two-state implicit filter recursive estimation for mobile robots," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 573–580, 2018.

[5] Y. Li, P. Duthon, M. Colomb, and J. Ibanez-Guzman, "What happens for a tof lidar in fog?" *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[6] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer, and M. A. Hoepflinger, "ANYmal - A Highly Mobile and Dynamic Quadrupedal Robot," in *IEEE International Conference on Intelligent Robots and Systems*, 2016, pp. 38–44.

[7] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon, "The Newer College Dataset: Handheld LiDAR, Inertial and Vision with

Ground Truth," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

[8] L. Zhang, D. Wisth, M. Camurri, and M. Fallon, "Balancing the Budget: Feature Selection and Tracking for Multi-Camera Visual-Inertial Odometry," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1182–1189, 2022.

[9] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer, "Super Odometry: IMU-centric LiDAR-Visual-Inertial Estimator for Challenging Environments," *arXiv preprint arXiv:2104.14938*, 2021.

[10] A. Santamaria-Navarro, R. Thakker, D. Fan, B. Morrell, and A.-a. Aghamohammadi, "Towards resilient autonomous navigation of drones," in *International Symposium on Robotics Research*, 2019.

[11] S. Khattak, H. Nguyen, F. Mascarich, T. Dang, and K. Alexis, "Complementary multi-modal sensor fusion for resilient robot pose estimation in subterranean environments," in *International Conference on Unmanned Aircraft Systems*, 2020, pp. 1024–1029.

[12] D. Wisth, M. Camurri, and M. Fallon, "Robust legged robot state estimation using factor graph optimization," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4507–4514, 2019.

[13] ——, "Preintegrated velocity bias estimation to overcome contact nonlinearities in legged robot odometry," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 392–398.

[14] D. Wisth, M. Camurri, S. Das, and M. Fallon, "Unified Multi-Modal Landmark Tracking for Tightly Coupled Lidar-Visual-Inertial Odometry," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1004–1011, 2021.

[15] D. Wisth, M. Camurri, and M. Fallon, "VILENS: Visual, Inertial, Lidar, and Leg Odometry for All-Terrain Legged Robots," *IEEE Transactions on Robotics*, 2021, submitted.

[16] H. Kolvenbach, D. Wisth, R. Buchanan, G. Valsecchi, R. Grandia, M. Fallon, and M. Hutter, "Towards autonomous inspection of concrete deterioration in sewers with legged robots," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1314–1327, 2020.

[17] M. Tranzatto, F. Mascarich, L. Bernreiter, C. Godinho, M. Camurri, S. Khattak, T. Dang, V. Reijgwart, J. Löje, D. Wisth, S. Zimmermann, H. Nguyen, M. Fehr, L. Solanka, R. Buchanan, M. Bjelonic, N. Khedekar, M. Valceschini, F. Jenelten, M. Dharmadhikari, T. Homberger, P. D. Petris, L. Wellhausen, M. Kulkarni, T. Miki, S. Hirsch, M. Montenegro, C. Papachristos, F. Tresoldi, J. Carius, G. Valsecchi, J. Lee, K. Meyer, X. Wu, J. Nieto, A. Smith, M. Hutter, R. Siegwart, M. Mueller, M. Fallon, and K. Alexis, "CERBERUS: Autonomous legged and aerial robotic exploration in the tunnel and urban circuits of the DARPA subterranean challenge," *Field Robotics*, no. 2, pp. 274–324, 2022.

[18] P. Furgale, "Representing Robot Pose: The good, the bad, and the ugly," 2014. [Online]. Available: http://paulfurgale.info/news/2014/6/9/representing-robot-pose-the-good-the-bad-and-the-ugly

[19] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.

[20] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[21] M. Grewal and A. Andrews, "How Good Is Your Gyro?" *IEEE Control Systems*, vol. 30, no. 1, pp. 12–86, 2010.

[22] P. D. Groves, "Navigation using inertial sensors [tutorial]," *IEEE Aerospace and Electronic Systems Magazine*, vol. 30, no. 2, pp. 42–69, 2015.

[23] J. Kannala and S. Brandt, "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses," *IEEE Transactions on*

Pattern Analysis and Machine Intelligence, vol. 28, no. 8, pp. 1335–1340, 2006.

[24] P. Furgale, J. Rehder, and R. Siegwart, "Unified temporal and spatial calibration for multi-sensor systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1280–1286.

[25] A. Carballo, J. Lambert, A. Monrroy, D. Wong, P. Narksri, Y. Kitsukawa, E. Takeuchi, S. Kato, and K. Takeda, "Libre: The multiple 3d lidar dataset," in *IEEE Intelligent Vehicles Symposium*, 2020, pp. 1094–1101.

[26] F. Tschopp, M. Riner, M. Fehr, L. Bernreiter, F. Furrer, T. Novkovic, A. Pfrunder, C. Cadena, R. Siegwart, and J. Nieto, "VersaVIS—An Open Versatile Multi-Camera Visual-Inertial Sensor Suite," *Sensors*, vol. 20, no. 5, 2020.

[27] J. Zhang, M. Kaess, and S. Singh, "On degeneracy of optimization-based state estimation problems," in *IEEE International Conference on Robotics and Automation*, 2016, pp. 809–816.

[28] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[29] F. Pomerleau, F. Colas, R. Siegwart, and S. Magnenat, "Comparing ICP Variants on Real-World Data Sets," *Autonomous Robots*, vol. 34, no. 3, pp. 133–148, 2013.

[30] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents.* The MIT Press, 2005.

[31] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[32] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback," *The International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[33] R. Raz, "On the complexity of matrix product," *Annual ACM Symposium on Theory of Computing*, pp. 144–151, 2002.

[34] C. Stachniss, J. J. Leonard, and S. Thrun, *Simultaneous Localization and Mapping*. Cham: Springer International Publishing, 2016, pp. 1153–1176.

[35] F. Dellaert, "Factor graphs: Exploiting structure in robotics," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 4, no. 1, 2021.

[36] F. Dellaert and M. Kaess, *Factor Graphs for Robot Perception*. Now Foundations and Trends, 2017.

[37] G. Grisetti, T. Guadagnino, I. Aloise, M. Colosi, B. Della Corte, and D. Schlegel, "Least squares optimization: From theory to practice," *Robotics*, vol. 9, no. 3, 2020.

[38] K. MacTavish and T. D. Barfoot, "At all Costs: A Comparison of Robust Cost Functions for Camera Correspondence Outliers," in *Conference on Computer and Robot Vision*, 2015, pp. 62–69.

[39] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, "Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1127–1134, 2020.

[40] M. Grupp, "evo: Python package for the evaluation of odometry and slam." 2017. [Online]. Available: https://github.com/MichaelGrupp/evo

[41] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Autonomous robots*, vol. 4, no. 4, pp. 333–349, 1997.

[42] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 4909–4916.

[43] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 1689–1696.

[44] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[45] T. Qin, J. Pan, S. Cao, and S. Shen, "A general optimization-based framework for local odometry estimation with multiple sensors," 2019.

[46] P. Geneva, K. Eckenhoff, W. Lee, Y. Yang, and G. Huang, "Openvins: A research platform for visual-inertial estimation," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 4666–4672.

[47] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam," *IEEE Transactions on Robotics*, 2021.

[48] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 4758–4765.

[49] W. S. Grant, R. C. Voorhies, and L. Itti, "Efficient velodyne slam with point and plane features," *Autonomous Robots*, vol. 43, no. 5, pp. 1207–1224, 2019.

[50] M. Yokozuka, K. Koide, S. Oishi, and A. Banno, "Litamin2: Ultra light lidar-based slam using geometric approximation applied with kl-divergence," *arXiv preprint arXiv:2103.00784*, 2021.

[51] L. Zhou, S. Wang, and M. Kaess, "π -LSAM : LiDAR Smoothing and Mapping With Planes," in *IEEE International Conference on Robotics and Automation*, Xi'an, China, 2021, to appear.

[52] P. Geneva, K. Eckenhoff, Y. Yang, and G. Huang, "LIPS: LiDAR-inertial 3D plane SLAM," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 123–130.

[53] H. Ye, Y. Chen, and M. Liu, "Tightly coupled 3D lidar inertial odometry and mapping," in *IEEE International Conference on Robotics and Automation*, 2019, pp. 3144–3150.

[54] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and R. Daniela, "Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 5135–5142.

[55] Y. Yang, P. Geneva, X. Zuo, K. Eckenhoff, Y. Liu, and G. Huang, "Tightly-coupled aided inertial navigation with point and plane features," in *International Conference on Robotics and Automation*, 2019, pp. 6094–6100.

[56] R. Hartley, M. Ghaffari, R. M. Eustice, and J. W. Grizzle, "Contact-aided invariant extended kalman filtering for robot state estimation," *The International Journal of Robotics Research*, vol. 39, no. 4, pp. 402–430, 2020.

[57] M. Camurri, M. Fallon, S. Bazeille, A. Radulescu, V. Barasuol, D. G. Caldwell, and C. Semini, "Probabilistic contact estimation and impact detection for state estimation of quadruped robots," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1023–1030, 2017.

[58] Pei-Chun Lin, H. Komsuoglu, and D. E. Koditschek, "Sensor data fusion for body state estimation in a hexapod robot with dynamical gaits," in *IEEE International Conference on Robotics and Automation*, 2005, pp. 4733–4738.

[59] M. Bloesch, C. Gehring, P. Fankhauser, M. Hutter, M. A. Hoepflinger, and R. Siegwart, "State estimation for legged robots on unstable and slippery terrain," in *IEEE International Conference on Intelligent Robots and Systems*, 2013, pp. 6058–6064.

[60] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: low-drift, robust, and fast," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 2174–2181.

[61] Z. Wang, J. Zhang, S. Chen, C. Yuan, J. Zhang, and J. Zhang, "Robust high accuracy visual-inertial-laser slam system," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 6636–6641.

[62] J. Zhang and S. Singh, "Laser-visual-inertial odometry and mapping with high robustness and low drift," *Journal of Field Robotics*, vol. 35, no. 8, pp. 1242–1264, 2018.

[63] J. Lin, C. Zheng, W. Xu, and F. Zhang, "R2live: A robust, real-time, lidar-inertial-visual tightly-coupled state estimator and mapping," *arXiv preprint arXiv:2102.12400*, 2021.

[64] J. Graeter, A. Wilczynski, and M. Lauer, "LIMO: Lidar-monocular visual odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 7872–7879.

[65] W. Shao, S. Vijayarangan, C. Li, and G. Kantor, "Stereo visual inertial lidar simultaneous localization and mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 370–377.

[66] X. Zuo, Y. Yang, P. Geneva, J. Lv, Y. Liu, G. Huang, and M. Pollefeys, "Lic-fusion 2.0: Lidar-inertial-camera odometry with sliding-window plane-feature tracking," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 5112–5119.

[67] T. Shan, B. Englot, C. Ratti, and R. Daniela, "Lvi-sam: Tightly-coupled lidar-visual-inertial odometry via smoothing and mapping," in *arXiv preprint arXiv:2104.10831*, 2021.

[68] S. Teng, M. W. Mueller, and K. Sreenath, "Legged robot state estimation in slippery environments using invariant extended kalman filter with velocity update," *arXiv preprint arXiv:2104.04238*, 2021.

[69] R. Hartley, M. G. Jadidi, L. Gan, J. Huang, J. W. Grizzle, and R. M. Eustice, "Hybrid contact preintegration for visual-inertial-contact state estimation using factor graphs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 3783–3790.

[70] M. Camurri, M. Ramezani, S. Nobili, and M. Fallon, "Pronto: A multi-sensor state estimator for legged robots in real-world scenarios," *Frontiers in Robotics and AI*, vol. 7, p. 68, 2020.

[71] M. Palieri, B. Morrell, A. Thakur, K. Ebadi, J. Nash, A. Chatterjee, C. Kanellakis, L. Carlone, C. Guaragnella, and A. A. Agha-Mohammadi, "LOCUS: A Multi-Sensor Lidar-Centric Solution for High-Precision Odometry and 3D Mapping in Real-Time," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 421–428, 2021.

[72] J. Ma, M. Bajracharya, S. Susca, L. Matthies, and M. Malchano, "Real-time pose estimation of a dynamic quadruped in gps-denied environments for 24-hour operation," *The International Journal of Robotics Research*, vol. 35, no. 6, pp. 631–653, 2016.

[73] C. Chen, B. Wang, C. X. Lu, N. Trigoni, and A. Markham, "A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence," 2020.

[74] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization

and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[75] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual slam algorithms: A survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, pp. 1–11, 2017.

[76] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *European Conference on Computer Vision*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 430–443.

[77] C. Tomasi and T. Kanade, "Detection and tracking of point features," International Journal of Computer Vision, Tech. Rep., 1991.

[78] J. Shi and C. Tomasi, "Good Features to Track," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1994, pp. 246–253.

[79] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint kalman filter for vision-aided inertial navigation," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 3565–3572.

[80] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 2502–2509.

[81] S. Agarwal, K. Mierle, and Others, "Ceres solver," 2021. [Online]. Available: http://ceres-solver.org

[82] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 International Conference on Computer Vision*, 2011, pp. 2548–2555.

[83] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *IEEE International Conference on Robotics and Automation*, 2014, pp. 15–22.

[84] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G2o: A general framework for graph optimization," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 3607–3613.

[85] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2017.

[86] L. Carlone, Z. Kira, C. Beall, V. Indelman, and F. Dellaert, "Eliminating conditionally independent sets in factor graphs: A unifying perspective based on smart factors," in *IEEE International Conference on Robotics and Automation*, 2014, pp. 4290–4297.

[87] A. Rosinol, T. Sattler, M. Pollefeys, and L. Carlone, "Incremental visual-inertial 3d mesh generation with structural regularities," in *International Conference on Robotics and Automation*, 2019, pp. 8220–8226.

[88] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571.

[89] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[90] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *IEEE International Conference on Robotics and Automation*, 2017, pp. 2043–2050.

[91] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition*, 2012.

[92] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *IEEE Intelligent Vehicles Symposium*, 2011, pp. 963–968.

[93] W. Wang, Y. Hu, and S. Scherer, "Tartanvo: A generalizable learning-based vo," in *Conference on Robot Learning*, 2020.

[94] P. E. Sarlin, D. Detone, T. Malisiewicz, and A. Rabinovich, "SuperGlue: Learning Feature Matching with Graph Neural Networks," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4937–4946.

[95] H. Zhan, C. S. Weerasekera, J. W. Bian, and I. Reid, "Visual Odometry Revisited: What Should Be Learnt?" in *IEEE International Conference on Robotics and Automation*, 2020, pp. 4203–4210.

[96] L. Clement, M. Gridseth, J. Tomasi, and J. Kelly, "Learning matchable image transformations for long-term metric visual localization," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1492–1499, 2020.

[97] N. Yang, L. v. Stumberg, R. Wang, and D. Cremers, "D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1281–1292.

[98] N. Kaygusuz, O. Mendez, and R. Bowden, "Multi-camera sensor fusion for visual odometry using deep uncertainty estimation," in *IEEE Intelligent Transportation Systems Conference*, 2021.

[99] Y. Almalioglu, M. Turan, A. E. Sari, M. R. U. Saputra, P. P. de Gusmão, A. Markham, and N. Trigoni, "Selfvio: Self-supervised deep monocular visual-inertial odometry and depth estimation," *arXiv preprint arXiv:1911.09968*, 2019.

[100] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[101] Z. Teed and J. Deng, "DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras," *arXiv preprint arXiv:2108.10869*, 2021.

[102] M. R. U. Saputra, P. Gusmao, Y. Almalioglu, A. Markham, and N. Trigoni, "Distilling knowledge from a deep pose regressor network," in *IEEE/CVF International Conference on Computer Vision*, 2019, pp. 263–272.

[103] Y. He, J. Zhao, Y. Guo, W. He, and K. Yuan, "Pl-vio: Tightly-coupled monocular visual-inertial odometry using point and line features," *Sensors*, vol. 18, no. 4, 2018.

[104] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, "Lsd: A fast line segment detector with a false detection control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 4, pp. 722–732, 2010.

[105] L. Zhang and R. Koch, "An efficient and robust line segment matching approach based on lbd descriptor and pairwise geometric consistency," *Journal of Visual Communication and Image Representation*, vol. 24, no. 7, pp. 794–805, 2013.

[106] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, 2018.

[107] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *IEEE International Symposium on Mixed and Augmented Reality*, 2011, pp. 127–136.

[108] F. Pomerleau, F. Colas, and R. Siegwart, *A Review of Point Cloud Registration Algorithms for Mobile Robotics*. Now Foundations and Trends, 2015.

[109] C. Park, P. Moghadam, J. L. Williams, S. Kim, S. Sridharan, and C. Fookes, "Elasticity meets continuous-time: Map-centric dense 3d lidar slam," *IEEE Transactions on Robotics*, 2021.

[110] T. Shan, B. Englot, D. Meyers, W. Wang, C. Ratti, and D. Rus, "LIO-SAM: Tightly-coupled lidar inertial odometry via smoothing and mapping," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020.

[111] M. Karimi, M. Oelsch, O. Stengel, E. Babaians, and E. Steinbach, "Lola-slam: Low-latency lidar slam using continuous scan slicing," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2248–2255, 2021.

[112] M. Kaess, "Simultaneous localization and mapping with infinite planes," in *IEEE International Conference on Robotics and Automation*, 2015, pp. 4605–4611.

[113] L. Bernreiter, L. Ott, J. Nieto, R. Siegwart, and C. Cadena, "Phaser: A robust and correspondence-free global pointcloud registration," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 855–862, 2021.

[114] J. Nubert, S. Khattak, and M. Hutter, "Self-supervised learning of lidar odometry for robotic applications," *arXiv preprint arXiv:2011.05418*, 2020.

[115] W. Wang, M. R. U. Saputra, P. Zhao, P. Gusmao, B. Yang, C. Chen, A. Markham, and N. Trigoni, "Deeppco: End-to-end point cloud odometry through deep parallel neural network," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 3248–3254.

[116] Z. Li and N. Wang, "Dmlo: Deep matching lidar odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 6010–6017.

[117] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5540–5549.

[118] W. Liu, D. Caruso, E. Ilg, J. Dong, A. I. Mourikis, K. Daniilidis, V. Kumar, and J. Engel, "Tlio: Tight learned inertial odometry," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5653–5660, 2020.

[119] S. Fahmi, G. Fink, and C. Semini, "On state estimation for legged locomotion over soft terrain," *IEEE Sensors Letters*, vol. 5, no. 1, pp. 1–4, 2021.

[120] G. P. Roston and E. P. Krotkov, "Dead reckoning navigation for walking robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, 1992, pp. 607–612.

[121] M. Reinstein and M. Hoffmann, "Dead reckoning in a dynamic quadruped robot: inertial navigation system aided by a legged odometer," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 617–624.

[122] M. Bloesch, M. Hutter, M. A. Hoepflinger, S. Leutenegger, C. Gehring, C. David Remy, and R. Siegwart, "State Estimation for Legged Robots - Consistent Fusion of Leg Kinematics and IMU," in *Robotics: Science and Systems VIII*, 2012.

[123] M. Ramezani, G. Tinchev, E. Iuganov, and M. Fallon, "Online lidar-slam for legged robots with robust registration and deep-learned loop closure," in *IEEE International Conference on Robotics and Automation*, 2020, pp. 4158–4164.

[124] J. Hwangbo, C. D. Bellicoso, P. Fankhauser, and M. Hutter, "Probabilistic foot contact estimation by fusing information from dynamics and differential/forward kinematics," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 3872–3878.

[125] F. Jenelten, J. Hwangbo, F. Tresoldi, C. D. Bellicoso, and M. Hutter, "Dynamic locomotion on slippery ground," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4170–4176, 2019.

[126] G. Bledt, P. M. Wensing, S. Ingersoll, and S. Kim, "Contact model fusion for event-based locomotion in unstructured terrains," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 4399–4406.

[127] J.-H. Kim, S. Hong, G. Ji, S. Jeon, J. Hwangbo, J. H. Oh, and H.-W. Park, "Legged robot state estimation with dynamic contact event information," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 6733–6740, 2021.

[128] T.-Y. Lin, R. Zhang, J. Yu, and M. Ghaffari, "Deep multi-modal contact estimation for invariant observer design on quadruped robots," *arXiv preprint arXiv:2106.15713*, 2021.

[129] B. Katz, J. D. Carlo, and S. Kim, "Mini cheetah: A platform for pushing the limits of dynamic quadruped control," in *International Conference on Robotics and Automation*, 2019, pp. 6295–6301.

[130] A. Segal, D. Haehnel, and S. Thrun, "Generalized-icp," in *Robotics: Science and Systems*, Seattle, USA, 2009.

[131] A. Reinke, X. Chen, and C. Stachniss, "Simple But Effective Redundant Odometry for Autonomous Vehicles," *arXiv preprint arXiv:2105.11783*, 2021.

[132] M. Valente, C. Joly, and A. d. L. Fortelle, "Deep sensor fusion for real-time odometry estimation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 6679–6685.

[133] X. Zuo, P. Geneva, W. Lee, Y. Liu, and G. Huang, "Lic-fusion: Lidar-inertial-camera odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 5848–5854.

[134] Y. Jia, H. Luo, F. Zhao, G. Jiang, Y. Li, J. Yan, and Z. Jiang, "Lvio-fusion: A self-adaptive multi-sensor fusion slam framework using actor-critic method," 2021.

[135] A. Chilian, H. Hirschmüller, and M. Görner, "Multisensor data fusion for robust pose estimation of a six-legged walking robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 2497–2504.

[136] D. Wooden, M. Malchano, K. Blankespoor, A. Howardy, A. A. Rizzi, and M. Raibert, "Autonomous navigation for BigDog," in *IEEE International Conference on Robotics and Automation*, 2010, pp. 4736–4741.

[137] M. F. Fallón, M. Antone, N. Roy, and S. Teller, "Drift-free humanoid state estimation fusing kinematic, inertial and lidar sensing," in *IEEE International Conference on Humanoid Robots*, 2014, pp. 112–119.

[138] S. Nobili, R. Scona, M. Caravagna, and M. Fallon, "Overlap-based ICP tuning for robust localization of a humanoid robot," in *IEEE International Conference on Robotics and Automation*, 2017, pp. 4721–4728.

[139] C. Semini, N. G. Tsagarakis, E. Guglielmino, M. Focchi, F. Cannella, and D. G. Caldwell, "Design of hyq – a hydraulically and electrically actuated quadruped robot," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, vol. 225, no. 6, pp. 831–849, 2011.

[140] R. Hartley, J. Mangelson, L. Gan, M. Ghaffari Jadidi, J. M. Walls, R. M. Eustice, and J. W. Grizzle, "Legged robot state-estimation through combined forward kinematic and preintegrated contact factors," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 4422–4429.

[141] M. Fourmy, T. Flayols, N. Mansard, and J. Solà, "Contact forces pre-integration for the whole body estimation of legged robots," in *IEEE International Conference on Robotics and Automation*, 2021.

[142] F. Grimminger, A. Meduri, M. Khadiv, J. Viereck, M. Wüthrich, M. Naveau, V. Berenz, S. Heim, F. Widmaier, T. Flayols, J. Fiene, A. Badri-Spröwitz,

and L. Righetti, "An open torque-controlled modular robot architecture for legged locomotion research," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3650–3657, 2020.

[143] M. Xie, A. Escontrela, and F. Dellaert, "A factor-graph approach for optimization problems with dynamics constraints," *arXiv*, 2020.

[144] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Fast incremental smoothing and mapping with efficient data association," in *IEEE International Conference on Robotics and Automation*, 2007, pp. 1670–1677.

[145] X. Wang, R. Marcotte, G. Ferrer, and E. Olson, "AprilSAM: Real-time Smoothing and Mapping," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 2486–2493.

[146] A. Juric, F. Kendes, I. Markovic, and I. Petrovic, "A Comparison of Graph Optimization Approaches for Pose Estimation in SLAM," in *International Convention on Information and Communication Technology, Electronics and Microelectronics*, 2021.

[147] O. Chum and J. Matas, "Matching with PROSAC - progressive sample consensus," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2005.

[148] W. Gander, M. J. Gander, and F. Kwok, *Scientific computing-An introduction using Maple and MATLAB*. Springer Science & Business, 2014, vol. 11.

[149] S. Gangapurwala, M. Geisert, R. Orsolino, M. Fallon, and I. Havoutis, "RLOC: Terrain-aware legged locomotion using reinforcement learning and optimal control," *arXiv*, 2020.

[150] T. Dang, M. Tranzatto, S. Khattak, F. Mascarich, K. Alexis, and M. Hutter, "Graph-based subterranean exploration path planning using aerial and legged robots," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1363–1388, 2020.

[151] R. Buchanan, M. Camurri, F. Dellaert, and M. Fallon, "Learning inertial odometry for dynamic legged robot state estimation," in *Conference on Robot Learning*, 2021, accepted.

[152] Y. Tao, M. Popovic, N. Chebrolu, and M. Fallon, "3D lidar reconstruction with probabilistic depth completion for robotic navigation," 2022.

[153] E. Hudon, S. Mirza, and D. Frigon, "Biodeterioration of concrete sewer pipes: State of the art and research needs," *Journal of Pipeline Systems Engineering and Practice*, vol. 2, no. 2, pp. 42–52, 2011.

[154] H. Kolvenbach, G. Valsecchi, R. Grandia, A. Ruiz, F. Jenelten, and M. Hutter, "Tactile inspection of concrete deterioration in sewers with legged robots," in *Conference on Field and Service Robotics*, 2019.

[155] P. Marion, M. Fallon, R. Deits, A. Valenzuela, C. P. D'Arpino, G. Izatt, L. Manuelli, M. Antone, H. Dai, T. Koolen, J. Carter, S. Kuindersma, and R. Tedrake, "Director: A user interface designed for robot operation with shared autonomy," *Journal of Field Robotics*, vol. 34, pp. 225–426, 2017.

[156] DARPA, *DARPA SubTerranean Challenge*. [Online]. Available: https://www.subtchallenge.com/

[157] A. Proudman, M. Ramezani, and M. Fallon, "Online estimation of diameter at breast height (dbh) of forest trees using a handheld lidar," *arXiv preprint arXiv:2108.01552*, 2021.

[158] M. Helmberger, K. Morin, N. Kumar, D. Wang, Y. Yue, G. Cioffi, and D. Scaramuzza, "The hilti slam challenge dataset," 2021.

[159] M. Camurri, L. Zhang, D. Wisth, and M. Fallon, "HILTI SLAM Challenge Submission: VILENS and SLAM," 2021.

[160] T.-M. Nguyen, S. Yuan, M. Cao, T. H. Nguyen, and L. Xie, "Viral slam: Tightly coupled camera-imu-uwb-lidar slam," *arXiv preprint arXiv:2105.03296*, 2021.

[161] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart, "Maplab: An open framework for research in visual-inertial mapping and localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1418–1425, 2018.

[162] A. Reinke, M. Camurri, and C. Semini, "A factor graph approach to multi-camera extrinsic calibration on legged robots," in *IEEE International Conference on Robotic Computing*, 2019, pp. 391–394.

[163] K. Koide, M. Yokozuka, S. Oishi, and A. Banno, "Adaptive Hyperparameter Tuning for Black-box LiDAR Odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021.

[164] M. Sualeh and G.-W. Kim, "Visual-lidar based 3d object detection and tracking for embedded systems," *IEEE Access*, vol. 8, pp. 156 285–156 298, 2020.

[165] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic terrain mapping for mobile robots with uncertain localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3019–3026, 2018.

[166] L. Zhou, Z. Li, and M. Kaess, "Automatic extrinsic calibration of a camera and a 3d lidar using line and plane correspondences," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018, pp. 5562–5569.