

Inertial Learning and Haptics for Legged Robot State Estimation in Visually Challenging Environments

Russell Buchanan

New College
University of Oxford

*A thesis submitted for the degree of
Doctor of Philosophy*

Hilary 2023

Abstract

Legged robots have enormous potential to automate dangerous or dirty jobs because they are capable of traversing a wide range of difficult terrains such as up stairs or through mud. However, a significant challenge preventing widespread deployment of legged robots is a lack of robust state estimation, particularly in visually challenging conditions such as darkness or smoke.

In this thesis, I address these challenges by exploiting proprioceptive sensing from inertial, kinematic and haptic sensors to provide more accurate state estimation when visual sensors fail. Four different methods are presented, including the use of haptic localisation, terrain semantic localisation, learned inertial odometry, and deep learning to infer the evolution of IMU biases.

The first approach exploits haptics as a source of proprioceptive localisation by comparing geometric information to a prior map. The second method expands on this concept by fusing both semantic and geometric information, allowing for accurate localisation on diverse terrain.

Next, I combine new techniques in inertial learning with classical IMU integration and legged robot kinematics to provide more robust state estimation. This is further developed to use only IMU data, for an application entirely different from robotics: 3D reconstruction of bone with a handheld ultrasound scanner. Finally, I present the novel idea of using deep learning to infer the evolution of IMU biases, improving state estimation in exteroceptive systems where vision fails.

Legged robots have the potential to benefit society by automating dangerous, dull, or dirty jobs and by assisting first responders in emergency situations. However, there remain many unsolved challenges to the real-world deployment of legged robots, including accurate state estimation in vision-denied environments. The work presented in this thesis takes a step towards solving these challenges and enabling the deployment of legged robots in a variety of applications.

Inertial Learning and Haptics for Legged Robot State Estimation in Visually Challenging Environments



Russell Buchanan
New College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy
Hilary 2023

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.

Russell Buchanan, New College

Acknowledgements

First of all I would like to thank my supervisors, Professor Maurice Fallon and Professor Marco Camurri. Their guidance and support empowered me to apply my best self to my work. To Maurice especially, I would like to thank for giving me the opportunity to come to Oxford and for his expert mentorship to which I owe a great deal of my research proficiency. To Marco, I would particularly like to thank for his in-depth technical guidance which has probably saved me hundreds of hours of debugging and re-inventing the wheel.

Next, I would like to thank my many collaborators at ETH Zürich, Poznań University of Technology, Georgia Institute of Technology, Oxford Orthopaedic Engineering Centre and the engineers of the Oxford Robotics Institute. A lot of this work would not have been possible without their contributions. In particular, I would like to thank Professor Frank Dellaert for his valuable advice along the way.

My time at Oxford would not have been the same without my friends and colleagues at ORI: Matías, Mohamed, Lintong and Frank. They brought the fun and light-heartedness to my time at ORI. I would also like to thank all my friends in the New College MCR and the wider New College community for acting as a second home for me in Oxford.

I would like to thank my examiners, Professor Niki Trigoni and Professor Dimitrios Kanoulas for their attention to detail and valuable feedback on my research.

I also gratefully acknowledge the funding bodies that enabled my work: the Royal Society, the European Union Horizon 2020 THING project and New College.

Finally, I must thank my family for their steadfast love and support throughout this process. I want to thank my parents and brother who encouraged me to go to Oxford. And lastly, I want to thank Irene who has been there since the beginning and helped me every step of the way.

Abstract

Legged robots have enormous potential to automate dangerous or dirty jobs because they are capable of traversing a wide range of difficult terrains such as up stairs or through mud. However, a significant challenge preventing widespread deployment of legged robots is a lack of robust state estimation, particularly in visually challenging conditions such as darkness or smoke.

In this thesis, I address these challenges by exploiting proprioceptive sensing from inertial, kinematic and haptic sensors to provide more accurate state estimation when visual sensors fail. Four different methods are presented, including the use of haptic localisation, terrain semantic localisation, learned inertial odometry, and deep learning to infer the evolution of IMU biases.

The first approach exploits haptics as a source of proprioceptive localisation by comparing geometric information to a prior map. The second method expands on this concept by fusing both semantic and geometric information, allowing for accurate localisation on diverse terrain.

Next, I combine new techniques in inertial learning with classical IMU integration and legged robot kinematics to provide more robust state estimation. This is further developed to use only IMU data, for an application entirely different from robotics: 3D reconstruction of bone with a handheld ultrasound scanner. Finally, I present the novel idea of using deep learning to infer the evolution of IMU biases, improving state estimation in exteroceptive systems where vision fails.

Legged robots have the potential to benefit society by automating dangerous, dull, or dirty jobs and by assisting first responders in emergency situations. However, there remain many unsolved challenges to the real-world deployment of legged robots, including accurate state estimation in vision-denied environments. The work presented in this thesis takes a step towards solving these challenges and enabling the deployment of legged robots in a variety of applications.

Contents

Acronyms	viii
----------	------

I Preliminaries	1
------------------------	----------

1 Introduction	2
-----------------------	----------

1.1 Motivation	2
1.2 Objectives	5
1.3 Contributions	7
1.4 Publications and Highlights	7
1.4.1 First-Author Papers	8
1.4.2 Co-Authored Papers	8
1.4.3 Additional Contributions	9
1.5 Thesis Outline	9

2 Background	11
---------------------	-----------

2.1 Notation	11
2.1.1 Mathematical Preliminaries	12
2.2 Terminology	14
2.3 Inertial Measurement Units	16
2.3.1 IMU Technology	16
2.3.2 MEMS IMU Noise Modelling	17
2.3.3 Allan Variance Calibration	19
2.3.4 IMU Mechanisation in $SO(3)$	22
2.4 Other Hardware and Sensors	24
2.4.1 Joint Sensors	24
2.4.2 Cameras	24
2.4.3 Lidar	26
2.5 State Estimation Algorithms	27
2.6 Performance Metrics	31

3	Literature Review	34
3.1	Inertial Navigation Systems	34
3.1.1	Pedestrian Foot Tracking	35
3.1.2	Multi-IMU Tracking	36
3.1.3	Learned Inertial Navigation	37
3.2	Proprioceptive State Estimation	39
3.2.1	Haptic Sensing	39
3.2.2	Haptic Localisation	41
3.2.3	Kinematic-Inertial Odometry	42
3.3	Exteroceptive State Estimation	43
3.3.1	Visual Odometry	44
3.3.2	Visual-Inertial Navigation Systems	45
3.3.3	Fusing Legged Kinematic Information	47
3.3.4	Learning IMU Noise Models	49
3.4	Conclusion	50
II	Haptic Localisation	51
4	Haptic Localisation with Geometric Information	52
4.1	Haptic Sequential Monte Carlo Localization for Quadrupedal Locomotion in Vision-Denied Scenarios	53
4.2	Discussion	63
5	Haptic Localisation with Terrain Semantics	64
5.1	Navigating by touch: haptic Monte Carlo localization via geometric sensing and terrain classification	65
5.2	Unsupervised Learning of Terrain Representations for Haptic Monte Carlo Localization	81
5.3	Discussion	91
III	Inertial Learning	93
6	Learning Inertial Odometry	94
6.1	Learning Inertial Odometry for Dynamic Legged Robot State Estimation	95
6.2	Markerless Handheld Ultrasound 3D Reconstruction of Bone using Visual-Inertial and Deep Inertial Odometry	108
6.3	Discussion	121

7	Learning the Bias Evolution of Inertial Sensors	123
7.1	Deep IMU Bias Inference for Robust Visual-Inertial Odometry with Factor Graphs	124
7.2	Discussion	135
IV	Conclusion	136
8	Future Work and Conclusion	137
8.1	Future Work	137
8.1.1	Haptic SLAM	137
8.1.2	Learning for Legged Robot State Estimation	139
8.1.3	Deep Learning for IMU de-noising	140
8.2	Conclusion	142
Appendices		
A	Derivation of Covariance Propagation of Imu Factors in SO(3)	145
A.1	The ImuFactor	146
A.2	The CombinedImuFactor	148
A.3	Comparison of ImuFactor and CombinedImuFactor	152
References		154

Acronyms

APE	Absolute Pose Error.
CNN	Convolutional Neural Network.
DoF	Degree of Freedom.
EKF	Extended Kalman Filter.
ICP	Iterative Closest Point.
IMU	Inertial Measurement Unit.
INS	Inertial Navigation Systems.
LSTM	Long Short-term Memory.
MAP	Maximum A Posteriori.
MEMS	Micro-electro-mechanical Systems.
ROS	Robotic Operating System.
RPE	Relative Pose Error.
SLAM	Simultaneous Localisation And Mapping.
SMC	Sequential Monte Carlo.
TSIF	Two State Implicit Filter.
VIO	Visual Inertial Odometry.
VO	Visual Odometry.
ZUPT	Zero-Velocity Update.

Part I

Preliminaries

1

Introduction

1.1 Motivation

Legged robots have an enormous potential to benefit society because they are able to navigate our world – which has largely been built for walking people – by walking over steps, gaps and up stairs as well as through unstructured environments such as snow, sand and mud. This can enable legged robots to automate jobs which are dull, dirty or too dangerous for humans to carry out. Additionally, legged robots typically have battery lives of over an hour, can carry payloads of several kilograms and, in some cases, can manipulate objects such as door handles. Some recent industrial applications of legged robots which have been investigated include inspections of sewers [85], mapping construction sites [102] and routine inspection of offshore platforms [52]. In addition to industrial applications, there is significant interest in the capacity for legged robots to assist first responders [18, 32, 82].

In 2018, the United States Defense Advanced Research Projects Agency (DARPA) launched the Subterranean (SubT) Challenge. This three year-long, \$82 million [104] international robotics competition aimed to accelerate the research and development of robotic technologies for operation in underground settings [139]. Although not explicitly required, nearly every team in the final round deployed legged robots and the winning team CERBERUS [140] exclusively used this type of robot. However,

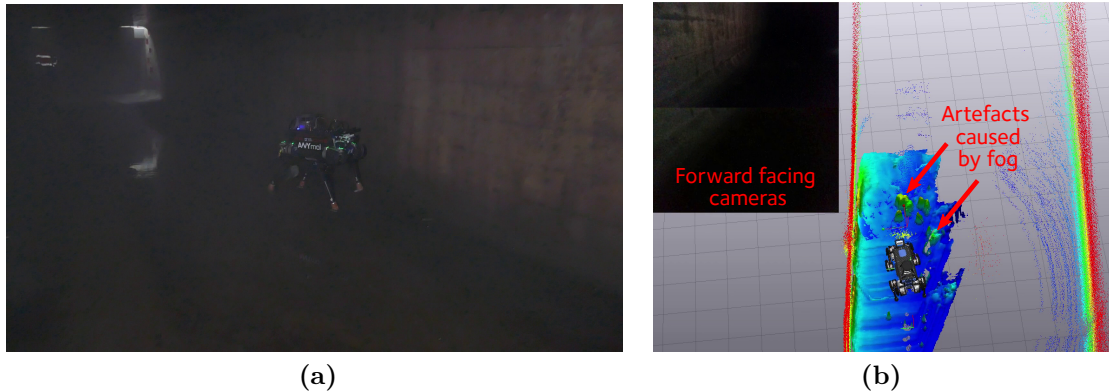


Figure 1.1: There are many environmental factors which degrade exteroceptive sensing. a) An ANYmal B quadruped [69] deployed in a dark and foggy sewer to perform routine inspection. b) The fog results in artefacts in the elevation map which the robot may interpret as obstacles. This can cause delays or even failure for a robot to complete its tasks.

in spite of the enormous interest and potential for societal benefits, we have not yet seen mass deployment of legged robots in industrial, domestic or emergency response applications. This is because there remain many unsolved challenges to the real-world deployment of legged robots [86].

State Estimation is a fundamental requirement for any mobile platform, including legged robots, however it remains an open challenge in robotics research. State estimation is the problem of inferring the position and orientation, jointly referred together as *pose* of the robot, from sensor measurements. Sometimes, additional quantities are included in the state such as the velocity, sensor calibration parameters or foot contact states. State estimation can be divided into two sub-problems: localisation and odometry. *Localisation* is the problem of estimating the robot's pose with respect to a prior known *map* of the world. *Odometry* is the problem of estimating the current pose with respect to the initial pose, having no prior information about the world [136, 152].

In recent years, several state-of-the-art state estimation algorithms have shown impressive performance in both odometry and localisation. This is largely due to the use of *exteroceptive* sensors, such as cameras and lidar which perform well in structured environments; however, these methods can perform poorly in

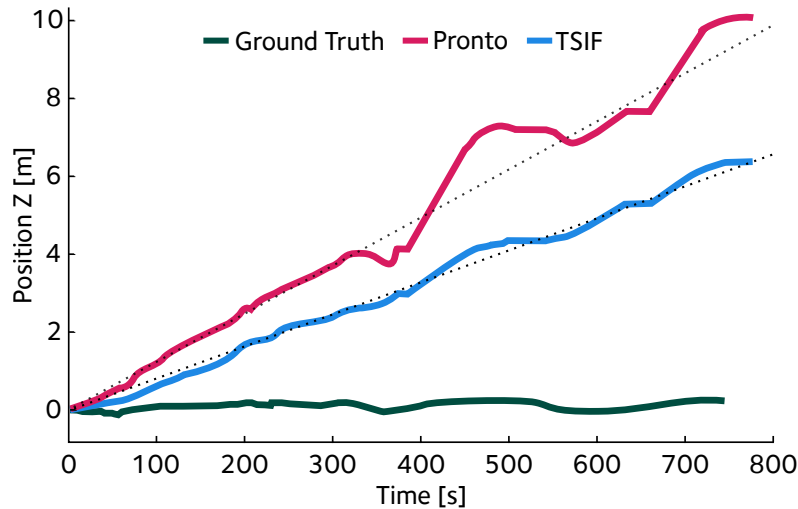


Figure 1.2: Limitation of proprioceptive state estimation for legged robots. This figure from Wisth et al. [154] shows the significant drift, particularly in the upwards direction, for common legged robot proprioceptive methods.

unstructured environments outside the lab. Many of the places where legged robots could be most useful are in fact vision-denied or visually challenging as shown in Fig. 1.1. Darkness, dust, smoke and fog are just a few of the visual challenges that a legged robot might be expected to overcome. This can cause many of the latest state-of-the-art methods to fail when deployed in real-world situations.

Proprioceptive state estimation is well established for legged robots and uses only sensors which measure the internal state of the robot such as Inertial Measurement Units (IMUs), joint encoders/torque sensors and haptics. IMU-only estimation for robots is generally considered impractical due to rapid integration of error and is therefore often aided by additional sensors [44] which, for legged robots, is typically kinematic sensing from joint encoders and torque sensors [7, 21]. However, current methods are still affected by significant drift over long periods of time (see Fig. 1.2) because kinematics are noisy, especially on deformable terrain or when slipping occurs. Additionally, current proprioceptive methods cannot perform localization to correct for drift.

Therefore, a gap remains between the exteroceptive sensing methods which often fail in real-world, vision-denied situations, and the proprioceptive methods which, while being quite robust, are not as accurate and are not capable of localisation.

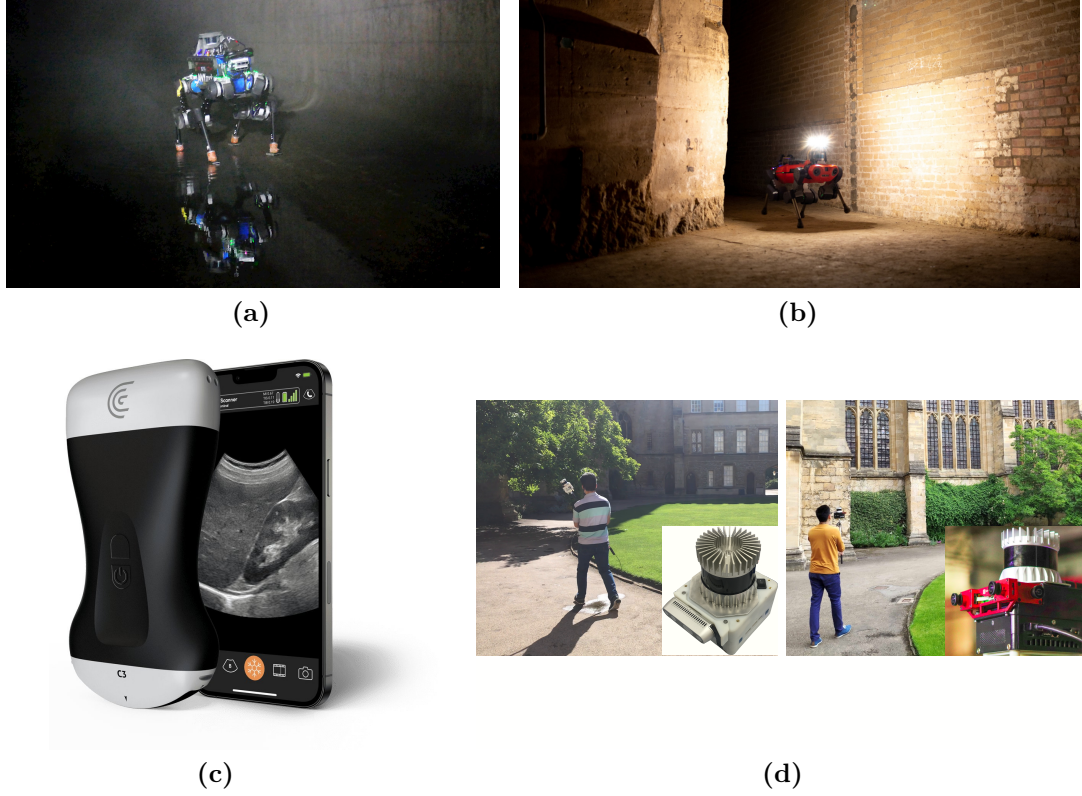


Figure 1.3: Platforms used in this thesis. a) ANYmal B quadruped performing routine inspection of a sewer in Zurich, Switzerland. b) ANYmal C quadruped exploring an underground bunker near Corsham, UK. c) Clarius L7 HD3¹ handheld general-purpose ultrasound scanner. d) Two different custom-made handheld sensing platforms featuring lidar, stereo cameras and IMUs [117, 161].

New methods are needed to: 1) enable proprioceptive methods to localise and thus bound drift, 2) improve proprioceptive methods to reduce the rate of drift, and 3) improve performance of exteroceptive methods in vision-denied environments. Addressing these challenges will improve the usability of legged robots in real-world environments.

1.2 Objectives

The work I present in this thesis addresses this gap in research by developing and deploying new methods of state estimation specifically for vision-denied or visually challenging environments. These methods are primarily developed for

¹Image used with permission Copyright © Clarius 2022. <https://clarius.com/>

legged robots but in many cases can generally be applied to state estimation of any mobile platform. Some of the robotic platforms which I have used are shown in Fig. 1.3 and include legged robots and handheld devices.

This thesis describes four different methods I developed under the theme of robust state estimation for vision-denied environments. Some are capable of purely proprioceptive state estimation (assuming complete lack of vision) whereas others are intended to support vision-based systems which might suffer from intermittent loss of vision or degraded performance.

The first main objective of my work is to exploit haptics as a source of proprioceptive localisation. Legged robots are in constant contact with the ground, which can be a valuable source of information. In a limited study in 2007 by Chitta et al. [29] it was shown that, using a prior geometric map of the environment and contact information, a legged robot could localise proprioceptively. I improve on this idea by: extending the method to 3D, enabling 6 Degree of Freedom (DoF) localisation, online performance and extended experimentation.

Next, I exploit the classification of terrain semantics as a source of localisation. Existing work from Bednarek et al. [5] has demonstrated how haptics can be used to classify different terrain types while walking. I show how proprioceptive localization on diverse terrain is possible by fusing both semantic and geometric information.

In a different domain, recent works on pedestrian tracking which use deep learning for learned inertial motion models [24, 96] show promise for reducing drift in proprioceptive state estimation for legged robots. I built on these works by developing a proprioceptive state estimator which fuses classical IMU integration with learned inertial displacement and legged robot kinematics for more robust state estimation. I then further developed the method to use only IMU data, enabling state estimation of a handheld ultrasound scanner without external tracking cameras. This thesis presents a full pipeline to perform 3D reconstruction of human bone from ultrasound images.

Finally, a major drawback of learning inertial motion models is an inability to generalise to new types of locomotion. Therefore, I explored the novel idea of using

deep learning to infer the evolution model of biases of an IMU for improved state estimation in exteroceptive systems where vision fails. I show that neural networks can learn a prior on the bias evolution and this can reduce drift for Visual Inertial Odometry (VIO) systems in visually challenging situations.

1.3 Contributions

In summary, this thesis describes the following contributions I have made:

- Development of a method for online localisation of a legged robot using haptic sensing in 3D with prior geometric information.
- The first state estimation method enabling haptic localisation using both geometric and terrain class information.
- Development of a neural network which learns the motion model of a legged robot from IMU data. Integration of learned displacements measurements in both filtering and factor graph contexts for state estimation.
- Application of learned inertial motion model to handheld ultrasound scanning enabling 3D reconstruction of human bone using low-cost sensors, without the need for external motion capture. This is integrated with a complete 3D reconstruction pipeline to visualise the position of the patella.
- The first work to investigate learning for IMU bias prediction and demonstrate the use in visually challenging state estimation problems.
- An open sourced C++ package for computing IMU Allan Variance plots.

1.4 Publications and Highlights

Summarized here are peer-reviewed papers accepted for publication during my DPhil. The four first-author publications constitute Chapters 4 – 6 and 8 of this thesis. The co-authored publications cover related topics and are referenced at

relevant points in this thesis. The additional contributions include an open-sourced IMU calibration tool and a draft first-author paper which constitutes Chapter 7.

1.4.1 First-Author Papers

1. **R. Buchanan**, V. Agrawal, M. Camurri, F. Dellaert, and M. Fallon “Deep IMU Bias Inference for Robust Visual-Inertial Odometry with Factor Graphs,” in *IEEE Robotics and Automation Letters*. 2022 [13]
2. **R. Buchanan**, M. Camurri, F. Dellaert, and M. Fallon, “Learning Inertial Odometry for Dynamic Legged Robot State Estimation,” in *Proceedings of the 5th Conference on Robot Learning (CoRL)*, November 2021. [15]
3. **R. Buchanan**, J. Bednarek, M. Camurri, M. R. Nowicki, K. Walas, and M. Fallon, “Navigating by Touch: Haptic Monte Carlo Localization via Geometric Sensing and Terrain Classification,” in *Autonomous Robots*, 2021. [14]
4. **R. Buchanan**, M. Camurri, and M. Fallon, “Haptic Sequential Monte Carlo Localization for Quadrupedal Locomotion in Vision-Denied Scenarios,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2020. [16]

1.4.2 Co-Authored Papers

1. M. Tranzatto, ... **R. Buchanan**, ... K. Alexis, "Team CERBERUS Wins the DARPA Subterranean Challenge: Technical Overview and Lessons Learned". in *Field Robotics* 2022. [140]
2. M. Tranzatto, ... **R. Buchanan**, ... K. Alexis, "CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge". in *Field Robotics* 2021. [139]
3. M. Łysakowski, M. R. Nowicki, **R. Buchanan**, M. Camurri, M. Fallon and K. Walas, “Unsupervised Learning of Terrain Representations for Haptic

Monte Carlo Localization”, in *IEEE International Conference on Robotics and Automation (ICRA)* May 2022. [98]

4. 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,” in *Journal of Field Robotics* 2020 [85]

1.4.3 Additional Contributions

1. The open source package Allan Variance ROS is available online [12].
2. *Draft Paper*: **R. Buchanan**, S. J. Tu, M. Camurri, S. J. Mellon, M. Fallon “Markerless Handheld Ultrasound 3D Reconstruction of Bone using Visual-Inertial and Deep Inertial Odometry,” 2023 [17]

1.5 Thesis Outline

This thesis will be presented in the University of Oxford “integrated thesis” format. Chapters 4 – 8 consist of one or two peer-reviewed publications per chapter with a statement of authorship clearly indicating my contributions to the works. The chapters are grouped into 4 parts to make a distinction between the different threads of research.

The remainder of this thesis is organized in the following chapters:

- **Part I - Preliminaries**

Chapter 2 - Background: Overview of fundamental concepts underlying the work of this thesis. This includes technical terminology, descriptions of hardware and some theoretical background.

Chapter 3 - Related Works: Survey of the relevant literature and recent state-of-the-art methods.

- **Part II - Haptic Localisation**

Chapter 4 - Haptic Localisation with Geometric Information: Introduction to a new method of localisation based only on haptic information.

Chapter 5 - Haptic Localisation with Terrain Semantics: Expansion on haptic localisation using terrain type information.

- **Part III - Inertial Learning**

Chapter 6 - Learning Inertial Odometry: Application of inertial learning to legged robot odometry and to ultrasound imaging to enable 3D reconstruction of bone.

Chapter 7 - Learning the Bias Evolution of Inertial Sensors: A novel method for learning bias evolution of IMUs.

- **Part IV - Conclusion**

Chapter 8 - Future directions and Conclusion: Insights into future areas of research on this topic and concluding discussion.

2

Background

The purpose of this chapter is to summarise important background theory which is necessary to understand the remaining chapters. Firstly, Section 2.1 is a summary of the fundamental notation and mathematical operators used throughout this thesis. In Section 2.2 is a detailed breakdown of key terminologies. Sections 2.3 and 2.4 provide in-depth background information on the various sensors used for this work. The chapter is concluded with Section 2.6, which explains the evaluation metrics used.

2.1 Notation

The notation used throughout this thesis is outlined in Table 2.1. Scalars, vectors and matrices are differentiated by using lowercase italic, lowercase bold and uppercase bold fonts respectively. The state of the robot or platform we are estimating is denoted using lowercase italic bold. When several states or measurements are accumulated over time, the set is denoted using uppercase calligraphic font. Reference frames are written in a typewriter font.

Where relevant, reference frame subscripts are used, such that ${}^W\mathbf{t}_{BI}$ denotes a vector from frame B to frame I, expressed in the W frame.

Table 2.1: Mathematical Notation

Quantity	Description	Example
Scalars	Upper/Lowercase Italic	$a, b, c, \dots$
Vectors	Lowercase Bold	$\mathbf{a}, \mathbf{b}, \mathbf{c}, \dots$
Matrices	Uppercase Bold	$\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$
Single State	Lowercase Italic Bold	$\mathbf{a}, \mathbf{b}, \mathbf{c}, \dots$
Multiple States and Measurements	Uppercase Calligraphic	$\mathcal{A}, \mathcal{B}, \mathcal{C}, \dots$
Reference Frames	Uppercase Typewriter	A, B, C, ...

A transform matrix $\mathbf{T}_{WB} \in \text{SE}(3)$ transforms a point in B into W such that ${}_W\mathbf{p} = \mathbf{T}_{WB} * {}_B\mathbf{p}$. Fig. 2.1 provides an illustration of this convention. In some cases, for brevity and where there is no ambiguity, reference frames are dropped from variables after definition.

2.1.1 Mathematical Preliminaries

To facilitate reading this document, some commonly used variables are defined in Table 2.2. Most importantly, the state $\mathbf{x}$ of a mobile platform (such as a robot or handheld device) is defined as:

$$\mathbf{x} \triangleq \{{}_W\mathbf{p}_{WB}, \mathbf{R}_{WB}, {}_B\mathbf{v}_{WB}, {}_I\mathbf{b}^a, {}_I\mathbf{b}^g\} \in \text{SO}(3) \times \mathbb{R}^{12}, \quad (2.1)$$

where ${}_W\mathbf{p}_{WB} \in \mathbb{R}^3$ is the position from the fixed world frame W to the mobile platform base frame B expressed in W. $\mathbf{R}_{WB} \in \text{SO}(3)$ is the orientation of the base and ${}_B\mathbf{v}_{WB} \in \mathbb{R}^3$ is the linear velocity expressed in B such that ${}_W\mathbf{v}_{WB} = \mathbf{R}_{WB} * {}_B\mathbf{v}_{WB}$. The IMU biases ${}_I\mathbf{b}^a \in \mathbb{R}^3$ and ${}_I\mathbf{b}^g \in \mathbb{R}^3$ which are expressed in the IMU frame I are also estimated as part of the state.

The *Special Orthogonal Group* in 3-dimensions $\text{SO}(3)$ is the group of all rotations about the origin of a 3-dimensional Euclidean space. This can be expressed using rotation matrices in 3D which are formally defined as $\text{SO}(3) \doteq \{\mathbf{R} \in \mathbb{R}^3 : \mathbf{R}^T \mathbf{R} = \mathbf{I}_{3 \times 3}, \det(\mathbf{R}) = 1\}$ where $\mathbf{I}_{3 \times 3}$ is an identity matrix of size 3x3 [46]. This space forms a smoothly differentiable manifold and is therefore a Lie Group. Therefore, there exists everywhere a tangent space to the manifold $\mathfrak{so}(3)$ which locally behaves

as a Euclidean space. To convert between the manifold $SO(3)$ and Euclidean space $\mathbb{R}^3$ we need to define some operators.

Every vector in $\mathbb{R}^3$ can be converted to a skew-symmetric matrix using the *hat* operator $(\cdot)^\wedge$:

$$\boldsymbol{\omega}^\wedge = \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix}^\wedge = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & -\omega_x & 0 \end{bmatrix} \in \mathfrak{so}(3) \quad (2.2)$$

Conversely, we can map a skew symmetric matrix back to a vector using the *vee* operator $(\cdot)^\vee$ such that: $\mathbf{W} = \boldsymbol{\omega}^\wedge$ and $\boldsymbol{\omega} = \mathbf{W}^\vee$.

Now we can define the exponential map $\exp : \mathfrak{so}(3) \rightarrow SO(3)$ which maps an element in the tangent space to the manifold [46]:

$$\begin{aligned} \exp(\phi^\wedge) &= \mathbf{I} + \frac{\sin(\|\phi\|)}{\|\phi\|} \phi^\wedge + \frac{1 - \cos(\|\phi\|)}{\|\phi\|^2} (\phi^\wedge)^2 \\ \exp(\phi^\wedge) &\cong \mathbf{I} + \phi^\wedge \end{aligned} \quad (2.3)$$

The converse of the exponential map is the logarithm map $\log : SO(3) \rightarrow \mathfrak{so}(3)$ which maps a non-identity rotation matrix $\mathbf{R} \neq \mathbf{I}$ to the tangent space [46]:

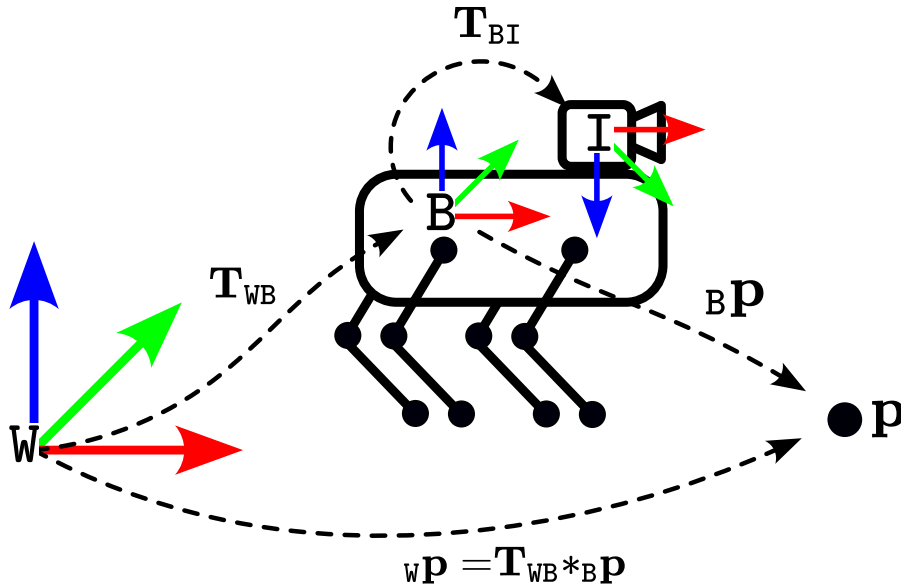


Figure 2.1: Example legged robot with frames to illustrate notation. Typically T_{BI} is a fixed frame denoting the rigid transform from the base frame B to the IMU frame I . The transform T_{WB} is the pose of the robot base frame expressed in world coordinates. A fixed point in the world p can be expressed in B , Bp , or in W , $w p$, where $w p = T_{WB} * B p$.

Table 2.2: Common Variables

Variable	Meaning	Variable	Meaning
$\mathbf{x}$	State	$\mathcal{I}$	Set of IMU Measurements
$\mathbf{p}$	Position	$\tilde{\mathbf{a}}$	Accelerometer Measurement
$\mathbf{R}$	Orientation in $\text{SO}(3)$	$\tilde{\boldsymbol{\omega}}$	Gyroscope Measurement
$\mathbf{v}$	Linear Velocity	$\mathbf{b}^a$	Accelerometer Bias
$\mathcal{X}$	Set of states	$\mathbf{b}^g$	Gyroscope Bias

$$\log(\mathbf{R}) = \frac{\psi (\mathbf{R} - \mathbf{R}^T)}{2 \sin(\psi)} \quad \text{with} \quad \psi = \cos^{-1} \left(\frac{\text{tr}(\mathbf{R}) - 1}{2} \right) \quad (2.4)$$

These operators convert between the manifold $\text{SO}(3)$ and the tangent space $\mathfrak{so}(3)$ but we can simplify the operators to work directly with $\mathbb{R}^3$:

$$\begin{aligned} \text{Exp} : \mathbb{R}^3 &\rightarrow \text{SO}(3) \quad ; \quad \phi \rightarrow \exp(\phi^\wedge) \\ \text{Log} : \text{SO}(3) &\rightarrow \mathbb{R}^3 \quad ; \quad \mathbf{R} \rightarrow \log(\mathbf{R})^\vee \end{aligned} \quad (2.5)$$

2.2 Terminology

In this thesis I make frequent use of terminology common to robotics research. In the following paragraphs some of these relevant terms are defined and explained to improve clarity.

Exteroceptive Sensors

All sensors can be classified as either *exteroceptive* or *proprioceptive* [126, 4]. Exteroceptive sensors measure the environment around the robot. Some examples include Global Navigation Satellite Systems (GNSS), cameras, and laser scanners. These sensors typically record data at relatively low rates (1-30 Hz) compared to some of the dynamic motions which may be encountered e.g. a trotting legged robot, aggressively shaken handheld device or a racing drone.

Proprioceptive Sensors

Proprioceptive sensors measure the internal state of the robot such as temperature, joint angles or forces. Some example sensors include IMUs, joint encoders [126, 4] or haptic sensors¹ and typically sample data at very high (>100 Hz) rates. This makes them more robust to dynamic motions. Additionally, these sensors work independently of external environmental factors, such as lighting conditions.

Localisation

Localisation is a problem in which the robot must answer: where am I? [126] Answering this question requires a map of the world which is defined in a fixed coordinate frame $\mathbf{W}$. This map contains information about the world such as obstacles, elevation or different terrain types. This means that localisation is not simply about determining the absolute position of the robot in some fixed frame, but also about estimating with respect to important elements in the world e.g. hazards or goals. Importantly, this can also enable extending the map when exploring, or receiving new information [126, 19]. Accuracy is computed in absolute terms i.e. how far is the estimated state from the true state at any point in time.

Odometry

Odometry is a problem in which the robot must answer: where have I moved to? This does not require a map of the world and instead estimates state in a fixed frame $\mathbf{W}$ defined by the initial pose. Odometry is an estimation of incremental motion over time with no prior information about the world and no capacity to interpret the topology of the environment. For example, a perfect odometry system could walk in a figure of 8 but would not “close the loop” at the intersection and would instead interpret the path as a single long corridor [19]. Odometry is sometimes also known as dead-reckoning [126]. Accuracy is computed by examining relative errors in trajectory over a fixed distance i.e. how much is the method drifting over time.

¹There is some debate over whether haptics are exteroceptive [125] or proprioceptive [29] however for the purpose of this work, I refer to haptics as proprioceptive.

2.3 Inertial Measurement Units

Inertial Measurement Units (IMUs) are multi-sensor devices which typically contain three single-axis accelerometers, three single-axis gyroscopes and three single-axis magnetometers. Accelerometers measure *proper acceleration* $\tilde{\mathbf{a}} \in \mathbb{R}^3$ which, when at rest, will have equal magnitude to Earth's acceleration at point of sampling e.g. ${}^w\mathbf{g} = \{0, 0, 9.81\}$. Gyroscopes measure rotational rate $\tilde{\boldsymbol{\omega}} \in \mathbb{R}^3$. All measurements are expressed in a single, unified IMU frame $\mathbf{I}$. The magnetometers measure the Earth's magnetic field, however this sensor is typically not used in robotics due to magnetic interference from the metal and electronics of the robot. Therefore, the magnetometer is disregarded for the purposes of this work.

2.3.1 IMU Technology

There exist many different types of accelerometers and gyroscopes based on a wide range of physics, such as Nuclear Magnetic Resonance, fibre optics, vibrations and springs [137, 157]. These methods have been used to construct highly accurate sensors, from tactical grade IMUs used in missiles to navigational grade IMUs used on aircraft and marine vessels [55]. These IMUs are so accurate that they can be used for state estimation over long periods of time with limited drift; however, they are prohibitively expensive and large, making them unsuitable for robots.

Instead, consumer grade Micro-electro-mechanical Systems (MEMS) IMUs are built using silicon micro-machining techniques, significantly reducing part count, size and cost. Over the years, MEMS IMUs have become ubiquitous in smart phones, wearable technology and robotics. MEMS gyroscopes typically make use of a vibrating mass (see Fig 2.2a) and the Coriolis force induced from a rotation [119]. This relationship can be expressed as:

$$\mathbf{f}_c = -2m(\boldsymbol{\omega} \times \mathbf{v}) \quad (2.6)$$

where $\mathbf{f}_c$ is the measured Coriolis force, m is a known mass, $\mathbf{v}$ is a known vibration velocity and $\boldsymbol{\omega}$ is the rotation rate.

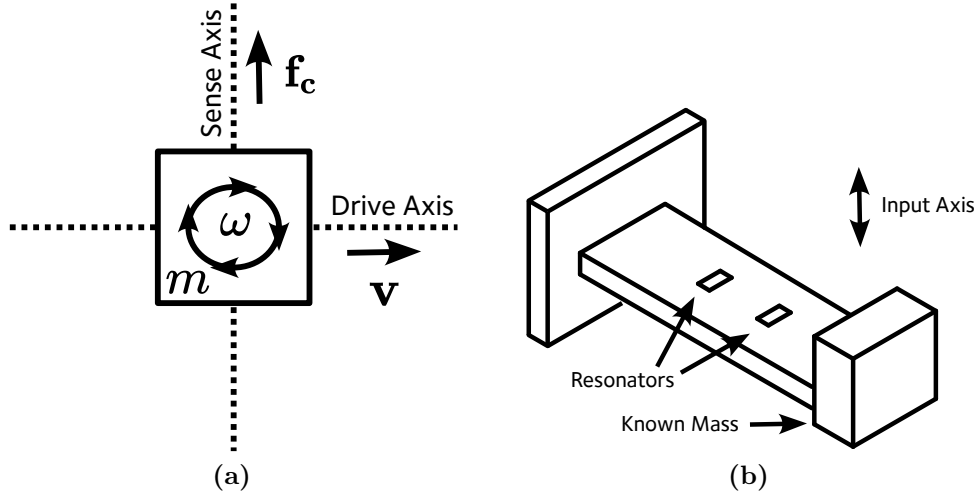


Figure 2.2: a) A simplified MEMS gyroscope design from [157], b) A simplified MEMS accelerometer design from [157]

MEMS accelerometers typically also make use of vibration as shown in Fig. 2.2b [157]. A mass is placed at the end of a cantilever beam which is actively vibrated. As the sensor is accelerated along the input axis, the force from the acceleration bends the beam which proportionally changes the frequency of the vibration. From measuring the change in frequency, the acceleration along the input axis can be inferred.

Typically for MEMS IMUs, three accelerometers and three gyroscopes (one of each, for each axis) are placed on a fixed platform. This is known as a Strap-Down IMU, as opposed to Stable Platform IMUs which use a moving platform on gimbals [137].

2.3.2 MEMS IMU Noise Modelling

While less expensive and smaller than higher grade IMUs, MEMS IMUs also have lower signal-to-noise ratios with significant noise from error sources such as bias, non-linear regions, dead-zones, quantization error and scale-factor [55]. Accounting for these noise sources require careful modelling. For example, let us consider Titterton and Weston's model for a gyroscope measurement along the x axis only: [137]:

$$\begin{aligned} \tilde{\omega}_x = & (1 + \mathbf{S}_x)\omega_x + \mathbf{M}_y\omega_y + \mathbf{M}_z\omega_z + \mathbf{B}_f \\ & + \mathbf{B}_{gx}\mathbf{a}_x + \mathbf{B}_{gz}\mathbf{a}_z + \mathbf{B}_{axz}\mathbf{a}_y\mathbf{a}_z + n_x \end{aligned} \quad (2.7)$$

Where $\tilde{\boldsymbol{\omega}}_x$ is the measurement, $\boldsymbol{\omega}_x$ is the true rotation rate and the rest of the terms represent different error sources. Equation (2.7) is complex, with many terms and correlations between axes and sensors. Therefore, simplified models are more typical in robotics applications and have been used very successfully for state estimation [44, 46].

Using one of the most common noise models, the true linear acceleration $\mathbf{a}$ (which is zero at rest) and the rotation rate $\boldsymbol{\omega}$ of a body frame $\mathbf{B}$ expressed in an Earth-fixed coordinate frame $\mathbf{W}$ can be recovered from the IMU measurements using the following equation:

$$\begin{aligned} {}_{\mathbf{B}}\tilde{\boldsymbol{\omega}}_{\mathbf{WB}}(t) &= {}_{\mathbf{B}}\boldsymbol{\omega}_{\mathbf{WB}}(t) + \mathbf{b}^g(t) + \boldsymbol{\eta}^g(t) \\ {}_{\mathbf{B}}\tilde{\mathbf{a}}(t) &= \mathbf{R}_{\mathbf{WB}}^{\mathbf{T}}(t)({}_{\mathbf{W}}\mathbf{a}(t) - {}_{\mathbf{W}}\mathbf{g}) + \mathbf{b}^a(t) + \boldsymbol{\eta}^a(t) \end{aligned} \quad (2.8)$$

Where $\mathbf{R}_{\mathbf{WB}} \in \text{SO}(3)$ is the orientation of the IMU. The quantities $\boldsymbol{\eta}^a$ and $\boldsymbol{\eta}^g$ represent additive white noise and are modelled as zero-mean Gaussian distributions. The IMU biases $\mathbf{b}^a$ and $\mathbf{b}^g$ are due to physical properties of the IMU. They change with each power cycle and continue to change slowly during operation [137]. The continuous time noise $\boldsymbol{\eta}^a$ and $\boldsymbol{\eta}^g$ can be converted to their discrete-time form by $\boldsymbol{\eta}^g = \frac{1}{\Delta t}\boldsymbol{\eta}^{gd}$, $\boldsymbol{\eta}^a = \frac{1}{\Delta t}\boldsymbol{\eta}^{ad}$ where Δt is the sampling time of the IMU [127].

Typically, the bias evolution is modelled as a Brownian noise process, where the derivatives are sampled from zero-mean Gaussian distributions:

$$\dot{\mathbf{b}}^a(t) = \boldsymbol{\eta}^{ba} \quad \dot{\mathbf{b}}^g(t) = \boldsymbol{\eta}^{bg} \quad (2.9)$$

This can be re-written in discrete time by integrating between two time steps $[t_i, t_j]$:

$$\mathbf{b}_j^a = \mathbf{b}_i^a + \boldsymbol{\eta}^{bad} \quad \mathbf{b}_j^g = \mathbf{b}_i^g + \boldsymbol{\eta}^{bgd} \quad (2.10)$$

Where $\boldsymbol{\eta}^{bad}$ and $\boldsymbol{\eta}^{bgd}$ are in the discrete-time form bias random walk parameters $\boldsymbol{\eta}^{bg} = \Delta t \boldsymbol{\eta}^{bgd}$, $\boldsymbol{\eta}^{ba} = \Delta t \boldsymbol{\eta}^{bad}$ [127]. This model of the evolution of IMU biases is an approximation and does not hold for long periods of time. In reality, the IMU biases change according to highly non-linear functions based on IMU temperature, vibrations, manufacturing defects, impact and time.

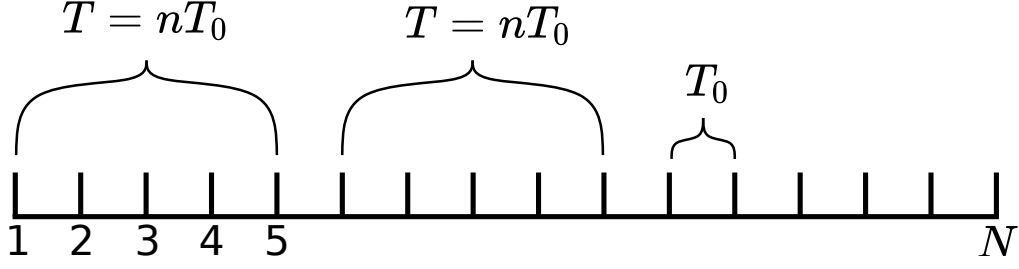


Figure 2.3: Illustrative diagram of computation of Allan Variance averages. Two consecutive, non-overlapping bins are shown each of size $n = 5$.

2.3.3 Allan Variance Calibration

The standard deviations of the Gaussian distributions for white noise $\boldsymbol{\eta}^g, \boldsymbol{\eta}^a$ and for bias evolution $\boldsymbol{\eta}^{bg}, \boldsymbol{\eta}^{ba}$ are extremely important for state estimation and must be determined from calibration of the individual IMU. This calibration can be done using the Allan Variance method as described in the IEEE Standard 952-1997 [72].

To calibrate an IMU using the Allan Variance method, N IMU measurements with sampling time of T_0 are recorded while the IMU is stationary. These measurements $\mathcal{I}(t)$ are then placed in bins of n measurements where $n < N/2$ and total bin period is $T = nT_0$ (See Fig.2.3).

The average value of each bin is computed as:

$$\bar{\mathcal{I}}_k(T) = \frac{1}{T} \sum_{t=t_k}^{t_k+T} \mathcal{I}(t) \quad (2.11)$$

Where $\bar{\mathcal{I}}_k(T)$ is the average value of the k -th bin. For a given bin, the average of the next consecutive bin is denoted as:

$$\bar{\mathcal{I}}_{k+1}(T) = \frac{1}{T} \sum_{t=t_k+T}^{t_k+2T} \mathcal{I}(t) \quad (2.12)$$

The Allan Variance (AV) for the given sampling period T is then computed from the mean squared error of each consecutive pair of bins:

$$AV(T) = \frac{1}{2(K-1)} \sum_{k=1}^{K-1} [\bar{\mathcal{I}}_k(T) - \bar{\mathcal{I}}_{k+1}(T)]^2 \quad (2.13)$$

Where K is the total number of bins. If there is no overlap in bins then $K = N/n$ and equation (2.13) is used. However the more bins that are used,

the more accurate the calibration. Therefore, in practice, the bins are shifted by only one measurement resulting in significant overlap [45]. In the overlapping case the following is used instead:

$$AV(T) = \frac{1}{2(K-2n)} \sum_{k=1}^{K-2n} [\bar{\mathcal{I}}_k(T) - \bar{\mathcal{I}}_{k+1}(T)]^2. \quad (2.14)$$

The next step in the process is to compute Eq. 2.13 for as many sampling periods as possible. However, this is limited by a lower bound of T_0 and upper bound of the total length of the data log. In practice, I have found for a 3 h log sampled at 200 Hz, a sampling range of 0.1 s - 1000 s (with 0.1 s increments) to give accurate results on simulated noise to 2 significant figures.

The Allan Deviation $AD(T) = \sqrt{AV(T)}$ can now be plotted on a log-log graph as in Fig. 2.4. The different regions in the Allan Deviation plot correspond to different IMU noise sources which are dominant at the respective sampling time T . This is because the Allan Variance is related to the Power Spectral Density (PSD) of the different noise sources in the dataset (see the IEEE Standard 952-1997 Annex C for more information [72]). The 1997 standard applies only to gyroscopes but, first in 1998 the IEEE Standard 1293-1998 was introduced applying Allan Variance analysis to accelerometers as well (see [70] Annex I). This was recently updated in the latest standard in 2018 [71].

From the Allan Deviation plot, we can read off the relevant noise parameters of the IMU. The second region in Fig. 2.4 is a result of white noise which in industry, for gyroscopes, is known as the “Angle Random Walk” (ARW)². Fitting a line with slope of -1/2 and measuring the intercept at $T = 1$ will give the ARW σ_g of a gyroscope or the equivalent “White Noise” σ_a of an accelerometer. These variables should be used as standard deviation for $\boldsymbol{\eta}^g$ and $\boldsymbol{\eta}^a$ respectively in Eq. 2.9. They can be converted to discrete time using the sampling time Δt as $\sigma_{gd} = \sigma_g \frac{1}{\sqrt{\Delta t}}$ and $\sigma_{ad} = \sigma_a \frac{1}{\sqrt{\Delta t}}$.

Fitting a line of +1/2 and reading off the intercept at $T = 3$ will give the “Rate Random Walk” σ_{bg} of a gyroscope and the “Acceleration Random Walk” σ_{ba} of

²The reason for this name is that integrating white noise results in a Brownian random walk. Therefore integrating white noise in the rotation rate will cause the angle estimate to have a random walk.

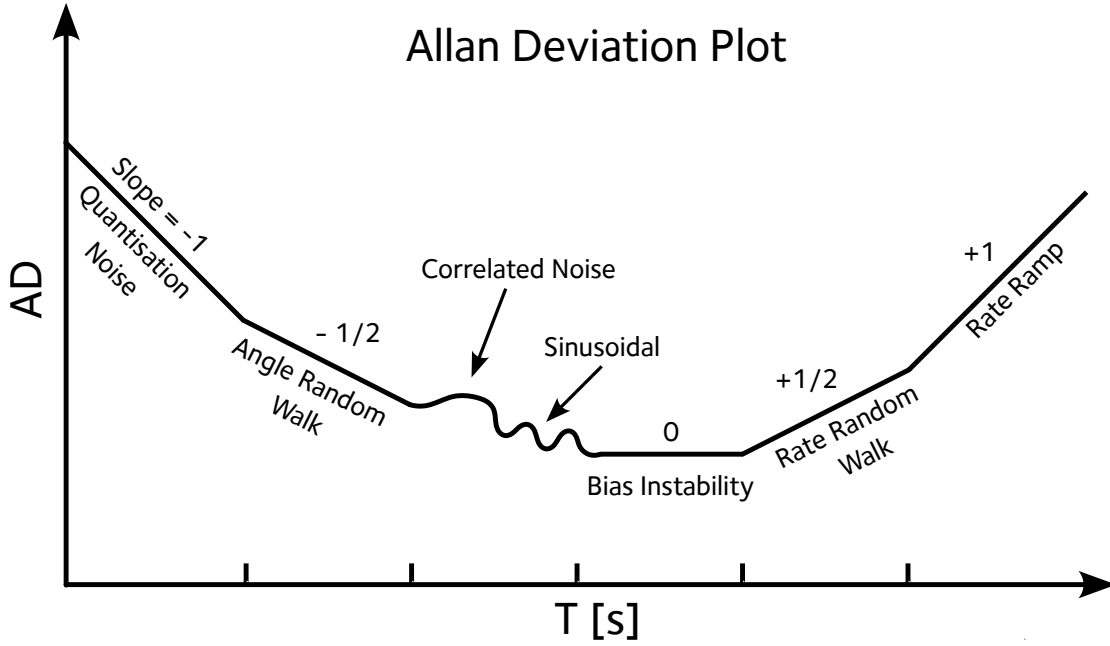


Figure 2.4: Theoretical Allan Deviation plot over sampling period T . The plot shows regions where different noise sources dominate [72]. Gyroscope names are used for the noise sources. An Accelerometer would have the same noise sources with different names as explained in Table 2.3.

Table 2.3: Allan Variance Variables

Sensor	Name	Variable	SI Units
Gyroscope	Angle Random Walk	σ_g	$\frac{rad}{s} \frac{1}{\sqrt{Hz}}$
	Rate Random Walk	σ_{bg}	$\frac{rad}{s^2} \frac{1}{\sqrt{Hz}}$
Accelerometer	Acceleration White Noise	σ_a	$\frac{m}{s^2} \frac{1}{\sqrt{Hz}}$
	Acceleration Random Walk	σ_{ba}	$\frac{m}{s^3} \frac{1}{\sqrt{Hz}}$

an accelerometer. These variables should be used as standard deviation for η^{bg} and η^{ba} respectively in Eq. 2.9. Similarly these numbers can also be converted to discrete time as $\sigma_{bgd} = \sigma_{bg}\sqrt{\Delta t}$ and $\sigma_{bad} = \sigma_{ba}\sqrt{\Delta t}$. A summary of these parameters is provided in Table 2.3.

The first contribution of this thesis is an open-sourced tool named “Allan Variance ROS,” which computes AD plots and automatically extracts the IMU noise parameters. As far as I am aware, this is the first Allan Variance tool written for the Robotic Operating System (ROS) and written in C++ which makes it faster to execute. The tool is also simple to understand and well documented making

it useful to the wider robotics research community. The code is hosted at https://github.com/ori-drs/allan_variance_ros and currently has over 300 stars and 50 forks. I have used this tool extensively in this work for the calibration of IMUs.

Figure 2.5 is an example AD plot of a Bosch BMI085 MEMS IMU which was generated using Allan Variance ROS from 3 hours of data recorded at 100 Hz. There is a clear $-1/2$ slope line around the sampling period of 1 s which is where white noise dominates. Around 100 s sampling time the slope is approximately $1/2$ allowing us to fit a line and to determine the random walk parameters.

2.3.4 IMU Mechanisation in $SO(3)$

IMU data can be integrated over time to estimate position and orientation as illustrated in Fig. 2.6. This process is known as *mechanisation* and begins with the integration equations:

$$\dot{\mathbf{R}}_{WB} = \mathbf{R}_{WB} \hat{\boldsymbol{\omega}}_{WB}^{\wedge}, \quad {}_W\dot{\mathbf{v}} = {}_W\mathbf{a}, \quad {}_W\dot{\mathbf{p}} = {}_W\mathbf{v} \quad (2.15)$$

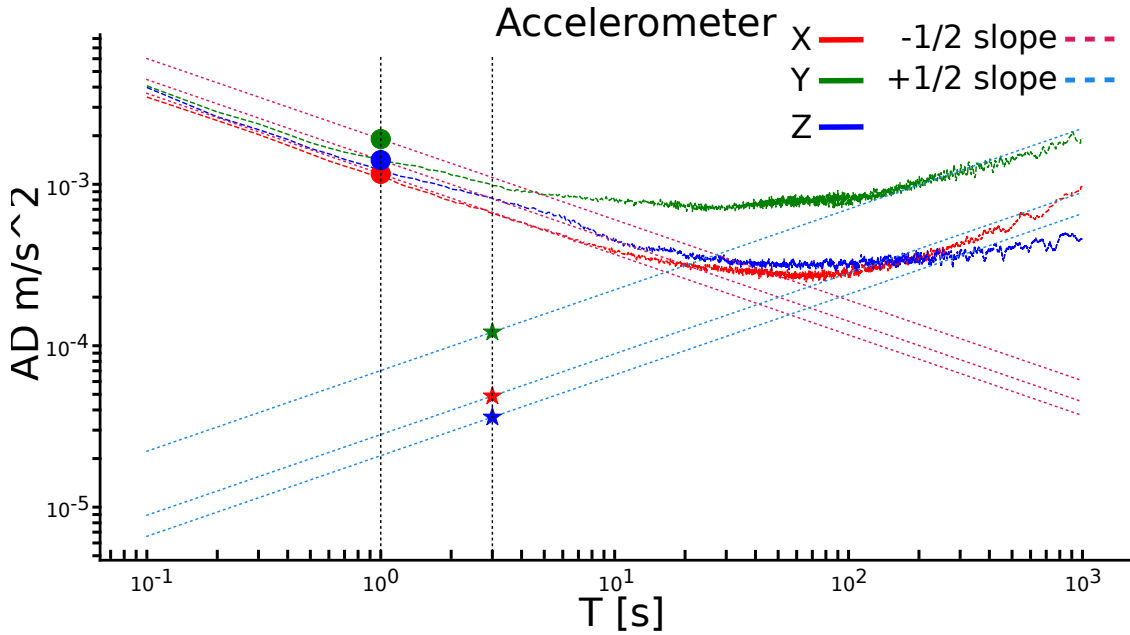


Figure 2.5: Allan Deviation plot for a Bosch BMI085 IMU. Generated using Allan Variance ROS [12]. Circles indicate the $T = 1$ intercept and stars indicate the $T = 3$ intercept. Different parameters are computed for each axis with the worst value used for estimation.

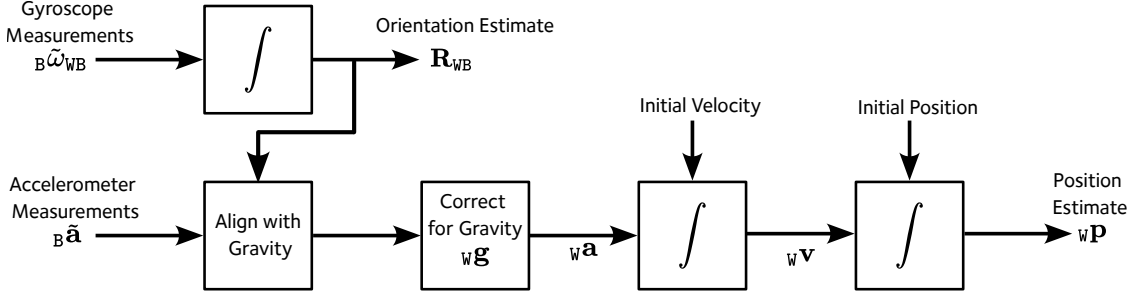


Figure 2.6: Flow diagram showing the process for IMU mechanisation [157].

For an IMU with sampling time Δt and integrating a single IMU measurement over the time interval $[t, t + \Delta t]$, we can estimate the state at $t + \Delta t$ [46]:

$$\begin{aligned}
 \mathbf{R}_{WB}(t + \Delta t) &= \mathbf{R}_{WB}(t) \text{Exp} \left(\int_t^{t+\Delta t} {}_B\boldsymbol{\omega}_{WB}(\tau) d\tau \right) \\
 {}_W\mathbf{v}(t + \Delta t) &= {}_W\mathbf{v}(t) + \int_t^{t+\Delta t} {}_W\mathbf{a}(\tau) d\tau \\
 {}_W\mathbf{p}(t + \Delta t) &= {}_W\mathbf{p}(t) + \int_t^{t+\Delta t} {}_W\mathbf{v}(\tau) d\tau + \int_t^{t+\Delta t} \int_t^{\tau} {}_W\mathbf{a}(\tau) d\tau^2
 \end{aligned} \tag{2.16}$$

If we assume that in the interval $[t, t + \Delta t]$ the acceleration and rotation rate ${}_W\mathbf{a}$ ${}_W\boldsymbol{\omega}$ are constant, the integration is straightforward [46]:

$$\begin{aligned}
 \mathbf{R}_{WB}(t + \Delta t) &= \mathbf{R}_{WB}(t) \text{Exp} (\boldsymbol{\omega}_{WB}(t) \Delta t) \\
 {}_W\mathbf{v}(t + \Delta t) &= {}_W\mathbf{v}(t) + {}_W\mathbf{a}(t) \Delta t \\
 {}_W\mathbf{p}(t + \Delta t) &= {}_W\mathbf{p}(t) + {}_W\mathbf{v}(t) \Delta t + \frac{1}{2} {}_W\mathbf{a}(t) \Delta t^2
 \end{aligned} \tag{2.17}$$

Plugging in the IMU measurement model in (2.8) (and using discrete-time noise) gives the IMU mechanisation equations. We also drop reference frames for simplicity as the orientation estimate $\mathbf{R}_{WB}$ is used to rotate the IMU measurements so that the state is estimated in W :

$$\begin{aligned}
 \mathbf{R}(t + \Delta t) &= \mathbf{R}(t) \text{Exp} \left((\tilde{\boldsymbol{\omega}}(t) - \mathbf{b}^g(t) - \boldsymbol{\eta}^{gd}(t)) \Delta t \right) \\
 \mathbf{v}(t + \Delta t) &= \mathbf{v}(t) + \mathbf{g} \Delta t + \mathbf{R}(t) (\tilde{\mathbf{a}}(t) - \mathbf{b}^a(t) - \boldsymbol{\eta}^{ad}(t)) \Delta t \\
 \mathbf{p}(t + \Delta t) &= \mathbf{p}(t) + \mathbf{v}(t) \Delta t + \frac{1}{2} \mathbf{g} \Delta t^2 + \frac{1}{2} \mathbf{R}(t) (\tilde{\mathbf{a}}(t) - \mathbf{b}^a(t) - \boldsymbol{\eta}^{ad}(t)) \Delta t^2
 \end{aligned} \tag{2.18}$$

These equations have been widely used in many applications for state estimation with IMUs [44].

2.4 Other Hardware and Sensors

The following sections provide brief explanations of several other sensors commonly used in legged robot state estimation.

2.4.1 Joint Sensors

Joint sensors are proprioceptive sensors which are inside motors such as encoders, which measure position (and through integration can be used to estimate velocity) or torque sensors. Typically, quadrupedal robots will have twelve sets of joint sensors, three in each leg, which may or may not measure torque. For less expensive quadrupeds, such as the Unitree Robotics A1 quadruped, torque can be estimated from the electrical current pulled by the motors. Additionally, some legged robots will have force sensors installed on the feet for estimating contacts.

Typically these sensors are high frequency (100 - 400 Hz), have low latency and limited noise, although torque/torque sensors may be noisier. However, they only measure relative motion between the links in the robot's body. Therefore, the robot state must be inferred from the relative measurements and the interaction of the robot with its environment. This relies on an assumption of zero velocity between the feet and the ground during contact. However, if there is slipping or if the robot walks on soft, deformable terrain (e.g. Fig. 2.7c), this assumption breaks down and can lead to significant error in state estimation. This is especially evident on soft terrain as the robot foot will initially make contact with the ground but, while raising the other feet, the robot will sink down into the terrain. The state estimator will interpret this as an upward drift which can grow significantly over time [15].

2.4.2 Cameras

Cameras are passive exteroceptive sensors because they measure the intensity of the light reflected off surfaces in the environment and collected on a photosensitive surface. The intensity is proportional to the number of photons collected during a fixed period of time (exposure time). Measurements are typically at a slower rate (10 - 30 Hz) but contain a significant amount of information — a single HD

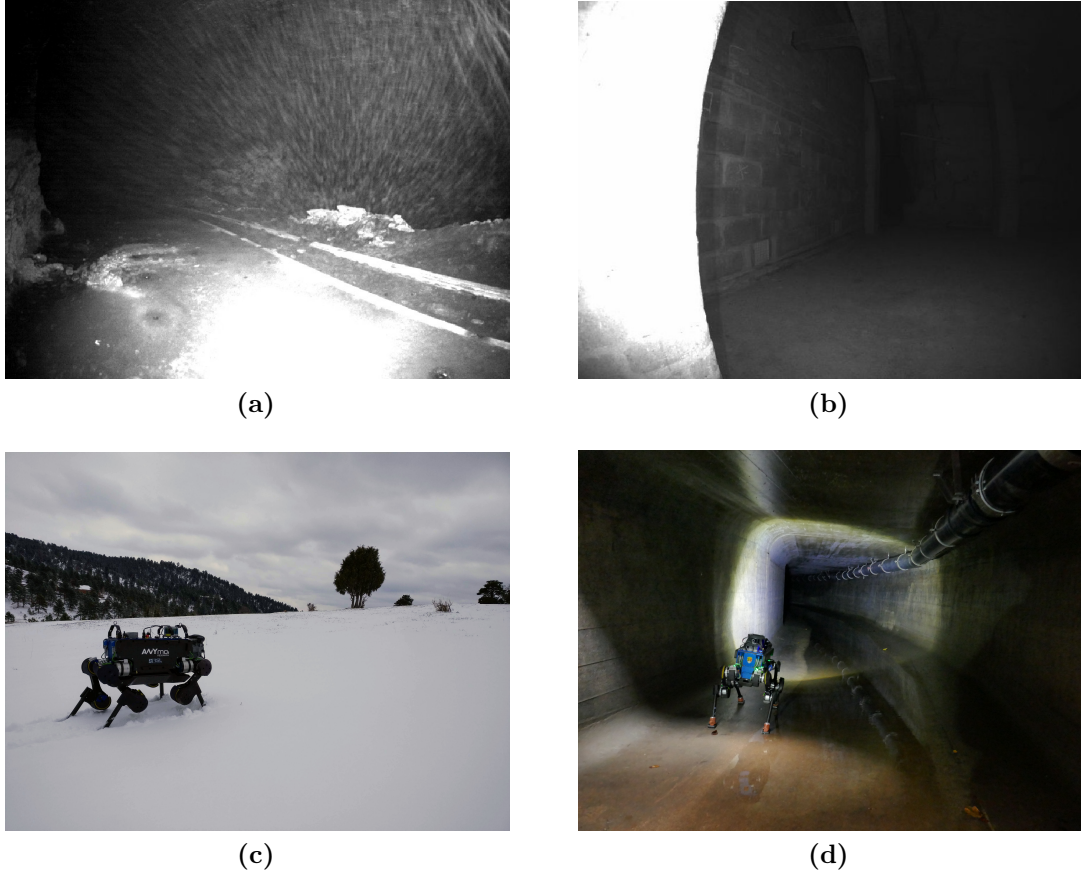


Figure 2.7: Challenges to different sensing sources. a) Dust particles in a mine in Switzerland [139]. b) A robot explores a pitch dark bunker with a mounted light to provide illumination. The light reflects off a nearby wall creating a region of over saturation [15]. c) ANYmal B robot walking in soft snow which causes the state estimator to drift upwards. d) ANYmal B in a sewer in Zurich. The highly self-similar geometry causes lidar-based state estimation to fail [85].

colour image (3 channels for red, green, blue) contains $1920 \text{ B} \times 1080 \text{ B} \times 3 = 6.2 \text{ MB}$. Carefully designed algorithms are required to be capable of processing this amount of data in real time.

To project a 3D point $[x, y, z]$ in $\mathbf{W}$ into a 2D pixel $[u, v]$ in the camera frame $\mathbf{C}$ the following equation is used:

$$\lambda \begin{bmatrix} {}^{\mathbf{C}}u \\ {}^{\mathbf{C}}v \\ 1 \end{bmatrix} = \mathbf{K}[\mathbf{R}|\mathbf{t}] \begin{bmatrix} {}^{\mathbf{W}}x \\ {}^{\mathbf{W}}y \\ {}^{\mathbf{W}}z \\ 1 \end{bmatrix} \quad (2.19)$$

Where λ is a scale factor which can be solved using a stereo pair of cameras,

$\mathbf{K} \in \mathbb{R}^4$ is the calibration matrix and $\mathbf{R} \in \text{SO}(3)$ and $\mathbf{t} \in \mathbb{R}^3$ are the orientation and position of the camera (together, $[\mathbf{R}|\mathbf{t}] \in \text{SE}(3)$) in $\mathcal{W}$. Additionally, different lenses introduce a distortion which must be corrected. Common distortion models include the radial-tangential model and the equidistant models and have distortion parameters $\mathbf{d} \in \mathbb{R}^4$.

Cameras require careful calibration to determine the *intrinsic* $\mathbf{K}$ and $\mathbf{d}$, and *extrinsic* parameters $\mathbf{R}$ and $\mathbf{t}$. These parameters can be solved by using multiple views of a known planar pattern. A common tool used frequently in this thesis is the Kalibr Calibration Toolbox [51, 118]. Once calibrated, these parameters are assumed to be fixed; however, they can change over long enough periods of time due to vibrations, impacts or thermal effects.

While visual sensors can be very accurate, an obvious limitation is in environments where visual conditions are poor. This can be due to darkness or obfuscating particles such as dust and fog such as in Fig. 2.7a and 2.7b. Additionally, cameras can also be subject to motion blur when used in highly dynamic situations such as mounted on a trotting robot.

2.4.3 Lidar

Light Detection and Ranging (LiDAR), or simply “lidar,” is an active exteroceptive sensor which projects a laser beam into the environment and then measures the reflection, using the time-of-flight to compute the distance to an object. With several lasers (typically 16 - 64) rotated by a motor, many points are accumulated together from a 360° scan into a sparse point cloud of the environment.

The change in pose between two consecutive point clouds can be estimated using the Iterative Closest Point (ICP) algorithm [114] which solves for the homogeneous transform that minimizes the distance between points. This has been used very successfully for mapping and reconstruction [111, 117]. However, this can be computationally expensive; therefore, some odometry algorithms instead extract geometric features from the point clouds and track these instead [156, 160].

The lidar range measurements are very accurate over large distances (up to 100 m) and can work in the dark. However, like cameras, lidar are susceptible to dust and motion effects. Additionally, in geometrically self-similar environments such as the sewer tunnel in Fig. 2.7, point clouds measured at different points in the tunnel can appear identical making point cloud matching for odometry difficult.

2.5 State Estimation Algorithms

State estimation is the broad problem of inferring the state $\mathbf{x}$ of a robot platform, as defined in (2.1), from a set of measurements $\mathcal{Z}$. Algorithms for solving these problems can be divided into two groups. *Filtering* methods estimate the current robot state from past robot states and measurements. In contrast, *smoothing* methods estimate a history of several states in batch [122, 4].

Filtering

The most general algorithm for estimating state beliefs is the Bayes Filter [136]. Following from the notation of Thrun et al. [136], the belief, or estimate, of the current state $\mathbf{x}_t$ can be denoted as $bel(\mathbf{x}_t)$. It is represented as a probability distribution computed from all past control and measurement data ($\mathcal{U} = \mathbf{u}_{0:t}$ and $\mathcal{Z} = \mathbf{z}_{0:t}$ respectively) as shown in

$$bel(\mathbf{x}_t) = p(\mathbf{x}_t | \mathcal{Z}, \mathcal{U}). \quad (2.20)$$

The Kalman Filter [80] is the most widely used implementation of the Bayes filter which models linear, Gaussian probabilistic systems. While this method will provide the *best linear unbiased estimate (BLUE)* [4], assuming linear and unbiased systems, most robotic systems are in fact highly nonlinear. Therefore, the strict assumptions of the Kalman filter limits its ability to track robot states [4].

This fundamental problem has motivated an abundance of adapted filters which relax these assumptions. The Extended Kalman Filter (EKF) [103] and Unscented Kalman Filter (UKF) account for nonlinear process models by means of Taylor Expansion and Sigma Points, respectively. These methods are very effective for

uni-modal distribution estimation and the vast majority of state estimation filters are based on one of these two adapted filters. Another major limitation of Kalman filters is the need for an explicit process model. This has been addressed in the Two State Implicit Filter (TSIF) [7] which has been used extensively on legged robots.

There exists another class of Bayes filtering algorithms designed for highly nonlinear and multi-modal systems which are known as *nonparametric* filters. Nonparametric filters do not rely on a fixed form of the belief distribution (e.g. Gaussian) and instead estimate the posterior with a finite number of samples. The most widely used nonparametric method for robot state estimation is the Particle Filter [48].

Particle Filters are a subset of Sequential Monte Carlo (SMC) methods which use random samples – *particles* – to estimate a probability distribution. If the number of particles is large enough and they are well distributed over the state space, the true state distribution can be accurately approximated [136]. Sensor measurements are used to compute the *importance weight* of each particle which determines how well they contribute to the distribution estimate. Since these algorithms are sensitive to the dimension of the state space, particles are occasionally re-sampled with probability given by their weight. This is known as *importance sampling* and allows particles to increase their density in areas where the state is more likely to be. Particle filters are considered to be one of the best methods for estimating highly nonlinear, systems although they are the most computationally demanding [136].

Smoothing

Smoothing is a group of state estimation algorithms which estimate a history of states in batch. This can either be over the whole trajectory, or over a fixed interval of states [122, 4]. This often results in a smoothing effect on the trajectory as the past states are updated when new information is received.

With these methods, we aim to find the Maximum A Posteriori (MAP) estimate of all states, $\mathcal{X}^* = \mathbf{x}_{0:t}$, given the measurements, $\mathcal{Z} = z_{0:t}$. This can be formalised as:

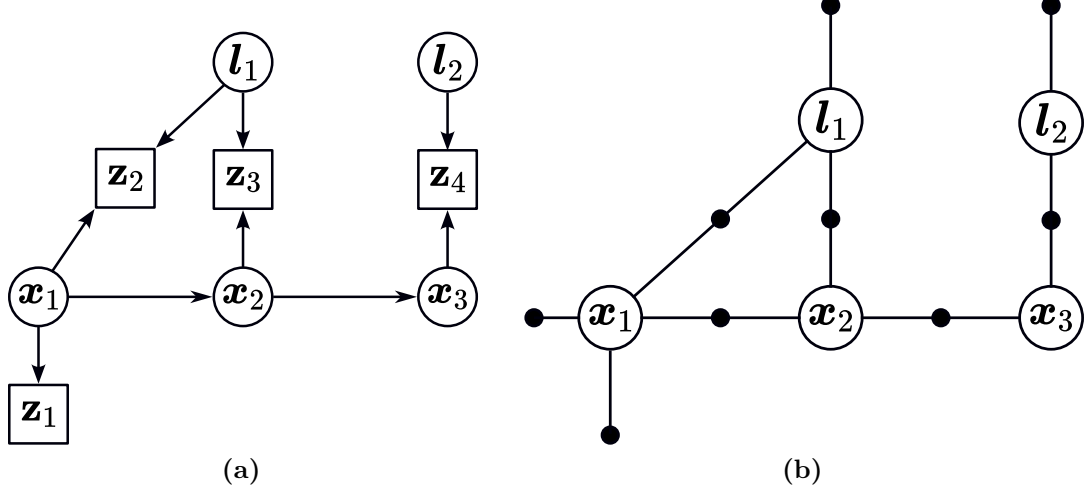


Figure 2.8: a) A Bayes Net graphical representation of a SLAM problem. b) The same SLAM problem formulated as a factor graph. The graph consists of three different types of measurements: (1) Prior information on the initial state, x_1 , and the landmark locations, l_1 and l_2 , (2) Landmark measurements from x to l , and (3) Odometry measurements relating consecutive states of x . Figures from [37].

$$\mathcal{X}^* = \arg \max_{\mathcal{X}} p(\mathcal{X} | \mathcal{Z}) \quad (2.21)$$

Extending the Bayesian probabilistic inference of (2.20) to include past states gives us a problem involving multiple states and measurements which can be represented as nodes in an acyclic graph, as in Fig. 2.8a. This is known as a Bayes net and it can be mathematically described as:

$$p(\mathcal{X}, \mathcal{Z}) = \prod_i p(\mathbf{x}_i, \mathbf{z}_i | \boldsymbol{\pi}_i) \quad (2.22)$$

where $p(\mathbf{x}_i, \mathbf{z}_i | \boldsymbol{\pi}_i)$ is the conditional probability of each state and measurement node on its parent nodes. Computing (2.22) is directly related to the structure of the graph [37].

While useful for modelling problems, Bayes nets are not the best tool for inference problems because each node in the graph must be a probability density, including the measurements which are known. Instead, *Factor Graphs* are more general in that they can be used to specify any factored function over $\mathcal{X}$ [37]. A factor

graph is a bipartite graph with two types of nodes: factors $\phi_i \in \mathcal{U}$, and variables $x_j \in \mathcal{V}$. Edges, $e_{i,j} \in \mathcal{E}$, always connect a factor node ϕ_i to a variable node x_j . The global function $\phi(\mathcal{X})$ of the factor graph is a product likelihood functions formed by factors ϕ_i acting on their adjacent variables x_j so that:

$$\phi(\mathcal{X}) = \prod_i \phi_i(\mathcal{X}_i) \quad (2.23)$$

where the edges $e_{i,j}$ encode the independence relationships between nodes. An example of a factor graph is shown in Fig. 2.8b. It is trivial to convert between Bayes net and factor graph formats [37].

The MAP estimate $\mathcal{X}^*$ can be calculated by maximizing the product of factors in (2.23) [4]. However, if we assume all factors are affected only by zero-mean Gaussian noise, this is equivalent to solving a nonlinear least-squares problem:

$$\mathcal{X}^* = \arg \min_{\mathcal{X}} \left(\sum_i \|\mathbf{h}_i(\mathcal{X}) \boxminus \mathbf{z}_i\|_{\Sigma_i}^2 \right) \quad (2.24)$$

Where $\mathbf{h}_i$ is typically a nonlinear function representing the measurement model, $\mathbf{z}_i$ is a measurement and Σ_i is the measurement covariance. $\boxminus$ represents a general subtraction operator between two points which could be on a manifold such as $SO(3)$ [46]. This subtraction is often referred to as a *residual*.

Equation 2.24 can be solved efficiently using standard nonlinear optimisation methods – such as Gauss-Newton – however a good initial estimate is needed.

The majority of optimisation-based state estimation algorithms in the literature use an external optimisation library for the back-end processing. For example, the mapping and localisation framework *maplab* [124] uses the Google Ceres solver [1] to solve the optimisation problems. Other notable solvers include General Graph Optimisation (g²o) [88] used by ORB-SLAM3 [20] and the Georgia Tech Smoothing and Mapping (GTSAM) framework [36, 34] used by VILENS [156].

So far we have discussed the general case of batch optimisation of all states and measurements. However, for large problems this can become intractable. It is possible to marginalise out older states keeping the factor graph size constant [34];

however, it is still computationally expensive to perform full optimisation each time a measurement is received.

To account for this, Incremental Smoothing and Mapping (iSAM) [79] periodically updates a QR factorisation of the factor graph information matrix, performing full optimisation only periodically. iSAM2 [78] was extended to use a more advanced graphical model called *Bayes Trees* as well as using Cholesky factorisation.

The GTSAM framework now includes incremental solving via iSAM2, making it a useful tool for developing factor graphs. As a result, all smoothing-based state estimation algorithms used in this thesis are built using GTSAM.

Factor graphs can also be used for inference over discrete or hybrid systems [76, 39]. This can be useful for estimating problems such as switching systems in which discrete states affect the process of continuous states, for example the contact state of legged robot feet affect estimate of the torso [60].

2.6 Performance Metrics

An important criterion when developing algorithms for state estimation is quantifying error. While there are many different ways to compute error, not all methods are correct for a given problem.

For localisation methods, robot states are estimated in a global reference frame, so a common error metric is the Absolute Pose Error (APE) [131]. First, we compute the absolute error for each sample $\mathbf{E}_i^A$:

$$\mathbf{E}_i^A = \mathbf{T}_{est,i} \ominus \mathbf{T}_{gt,i} \quad (2.25)$$

Where $\mathbf{T}_{est,i}$ is the estimated pose at time i , $\mathbf{T}_{gt,i}$ is the ground truth pose at the same time. Typically, the translation and rotation components are then separated to compute the instantaneous errors $_{\text{trans}}\text{APE}_i$ and $_{\text{rot}}\text{APE}_i$ respectively:

$$_{\text{trans}}\text{APE}_i = \left\| \text{trans}(\mathbf{E}_i^A) \right\|_2 \quad (2.26)$$

$$\text{rotAPE}_i = \left\| \text{Log}(\text{rot}(\mathbf{E}_i^A)) \right\|_2 \quad (2.27)$$

where (2.27) uses the logarithm map defined in (2.5). The RMSE for translation and rotation can then be computed from the instantaneous error values over the whole trajectory:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N \text{APE}_i^2} \quad (2.28)$$

This gives a sense of the magnitude of error; however, it may not be the most useful quantity in all situations. For example, as illustrated in Fig. 2.9, consider a trajectory that is perfect, but is initialized with a translation error. This results in a high APE, even though locally, between any two time steps, the trajectory is exactly the same as the ground truth. For this reason, it is often important to examine the relative error over a period of time [89].

For odometry problems, estimates are with respect to the initial frame. Therefore, a metric which quantifies the rate of drift over distance travelled is necessary. Usually, Relative Pose Error (RPE) is used for which we must first compute the relative error $\mathbf{E}_i^R$:

$$\mathbf{E}_i^R = \mathbf{T}_{est,i}^{-1} \mathbf{T}_{est,j} \boxminus \mathbf{T}_{gt,i}^{-1} \mathbf{T}_{gt,j} \quad (2.29)$$

where $\mathbf{T}_{est,i}$, $\mathbf{T}_{est,j}$ are estimated poses at time i and j , while $\mathbf{T}_{gt,i}$ and $\mathbf{T}_{gt,j}$ are ground truth poses at the same corresponding times. The choice of the times i and j is typically based on a $\delta_{i,j}$ distance travelled between the two times.

The relative errors transRPE_i and rotRPE_i can be computed from the translation and rotation components of $\mathbf{E}_i^R$ similarly to (2.26) and (2.27). Using relative error means that RPE gives a sense of drift for every distance travelled $\delta_{i,j}$. Typically, the mean and median values over the whole trajectory are computed, however analysing the instantaneous RPE over time can also be informative and show times where there is significant drift during the trajectory. For computing APE and RPE I use the tool *evo* [56].

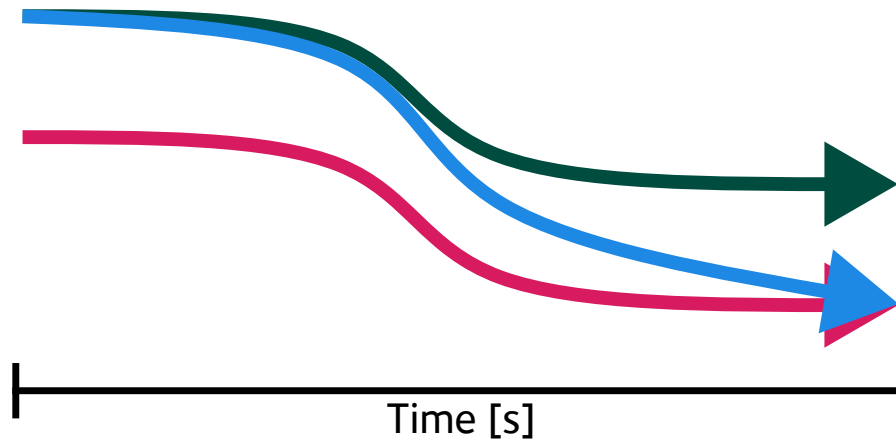


Figure 2.9: Example of three different trajectories to illustrate differences in error. The green arrow is ground truth and the red and blue are two estimated trajectories. The red is identical to green but with an initial bias. This means it has large APE but zero RPE as it has no drift. The blue arrow on the other hand, is constantly drifting and therefore large a RPE; however its APE is smaller than the red arrow at every point in time.

3

Literature Review

This chapter contains a detailed review of the relevant literature in state estimation, particularly for vision-denied environments. There are 3 Sections divided by sensing system: Section 3.1 discusses IMU only methods, Section 3.2 discusses methods which incorporate IMUs with other proprioceptive sensors such as kinematics and haptics. Finally, Section 3.3 covers vision-based estimation methods with a specific focus on legged robots and methods developed for visually challenging environments.

3.1 Inertial Navigation Systems

Inertial Navigation Systems (INS) are state estimation methods which use only sensing from IMUs. Long-term INS which rely only the mechanisation equations of (2.18) are not practical with consumer grade MEMS IMUs [137, 44] due to rapid accumulation of error. Double integration of noisy acceleration measurements and the dependence on orientation from noisy gyroscope measurements will cause the integrated position error to grow with t^3 [49, 157].

While careful filtering and calibration can reduce the accumulation of error, the IMU biases change over time and, generally, this cannot be observed using the IMU alone. As a result, nearly all practical MEMS INS methods exploit prior knowledge of the system to correct for drift. In the following sections I discuss the

different methods which each exploit prior information in some way, for example prior knowledge of the IMU motion (Section 3.1.1), using multiple IMUs together (Section 3.1.2) or using deep neural networks (Section 3.1.3)

3.1.1 Pedestrian Foot Tracking

There is a large field of research in tracking pedestrians using MEMS IMUs. This was first explored by Foxlin [49] who noted the potential to use human walking patterns to correct the data integration. When a human (or robot) walks, each foot alternates between a stationary *stance* phase and a moving *swing* phase. Foxlin showed how measurements from a shoe mounted IMU (for example see Fig. 3.1a) can be integrated during the flight phase as a position update within the filter. Once the foot touches down on the ground, and therefore has zero velocity with respect to the ground, the velocity estimate can be set to zero. This Zero-Velocity Update (ZUPT) can be fused into a state estimation filter such as an EKF which can update the full IMU state and biases [49].

The main challenge with this method is in detecting when the foot is in contact with the ground. Skog et al. [128] performed a thorough comparison of various threshold-based ZUPT detectors and found the Stance Hypothesis Optimal Detector (SHOE) detector gave the highest performance. This detector evaluates a window of IMU measurements, weighting acceleration and rotation rate separately, and detects contact based on a threshold. This method requires significant tuning which has motivated several learning-based methods for zero velocity detection [147, 146].

Another challenge to this type of INS is the unobservability of yaw rotation. As a result, the majority of drift comes from heading error [49].

Angermann et al. [3] instead proposed their FootSLAM algorithm which represents the world as a 2D hexagonal grid map. They use a probabilistic motion model for pedestrians with a Rao-Blackwellized particle filter to enable re-localisation when the pedestrian revisits a location within the map. Wahlstrom et al. [148] expanded on this work by using an adaptive threshold for ZUPT detection.

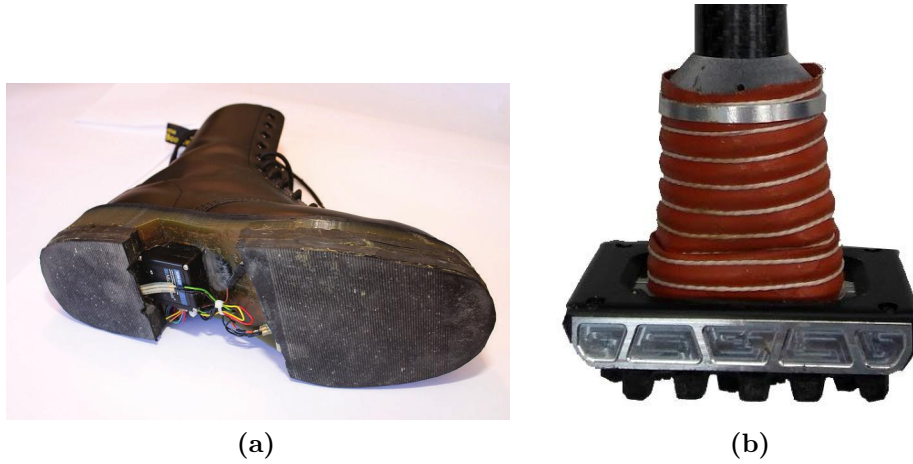


Figure 3.1: a) shoe mounted IMU from Skog [128]. b) sensorized robot foot from [141]

3.1.2 Multi-IMU Tracking

Another approach to correcting INS error accumulation is to use several IMUs and exploit their known relative positions to correct for drift. This can be done by placing several MEMS IMUs in an array on a circuit board to make a single sensor package [149]; however, this generally requires custom hardware which is not widely commercially available.

Alternatively, several discrete IMUs can be distributed about a rigid body. Vigne et al. [145] used several separate IMUs distributed in the legs of an exoskeleton. This significantly improved the orientation estimate of each IMU allowing them to estimate the flexibility in the motor joints.

A similar approach can also be applied to a human body; however, without kinematic sensors the problem is more difficult. For example, Schepers et al. of Xsens Technologies B.V. [123] presented a human motion tracking suit capable of accurate estimation of the full body pose of a human. However, they assume flat ground and only estimate the body's pose with respect to the torso.

Some learning methods also estimate body pose from multiple IMUs distributed about a suit [68, 159]. Especially notable is Physical Inertial Poser (PIP) [159] which is also capable of estimating odometry of the person; however, there has not been extensive investigation into the drift rate of their method.

In any methods using multiple IMUs, careful attention must be paid to synchronization of the data collection. This is especially difficult if separate, discrete IMUs are used which are not connected to the same circuit board. Because of these challenges, and because often multiple IMUs are not available, I focus on methods using only a single IMU in this work.

3.1.3 Learned Inertial Navigation

Recently, there has been significant research applying deep learning to inertial navigation. These methods are based on the insight that a neural network can learn the motion model of the walking subject (typically a pedestrian). From many labelled examples of pedestrians walking while carrying an IMU (for example in a smartphone), a neural network can be trained to estimate velocity or displacement.

Chen et al. introduced IONet [24], the first data driven INS which infers 2D displacement and orientation change from buffered IMU data. Their method outperformed both pure INS and ZUPT-based pedestrian tracking methods and they demonstrated their method with non-periodic motion – a wheeled trolley. Herath et al. [64] proposed RoNIN, a similar network which inferred 2D velocity and orientation from raw IMU data and compared different neural network architectures finding that a Convolutional Neural Network (CNN) based on ResNet [62] to give the most reliable performance.

Liu et al. [96] proposed TLIO, the first 3D approach and the first to use a head mounted IMU. An EKF is used to estimate the full 6 DoF state and the process update is propagated by classical integration of the IMU measurements. IMU measurements are buffered and then aligned with gravity before being passed to the ResNet CNN which infers displacement measurements. The learned displacement measurements are then used as measurement updates for the filter; thereby allowing bias estimation.

More recently, some work has sought to incorporate Simultaneous Localisation And Mapping (SLAM) concepts into learning-based INS. Herath et al. [63] developed Neural Inertial Localization (NILoc) which stores a history of velocity estimates

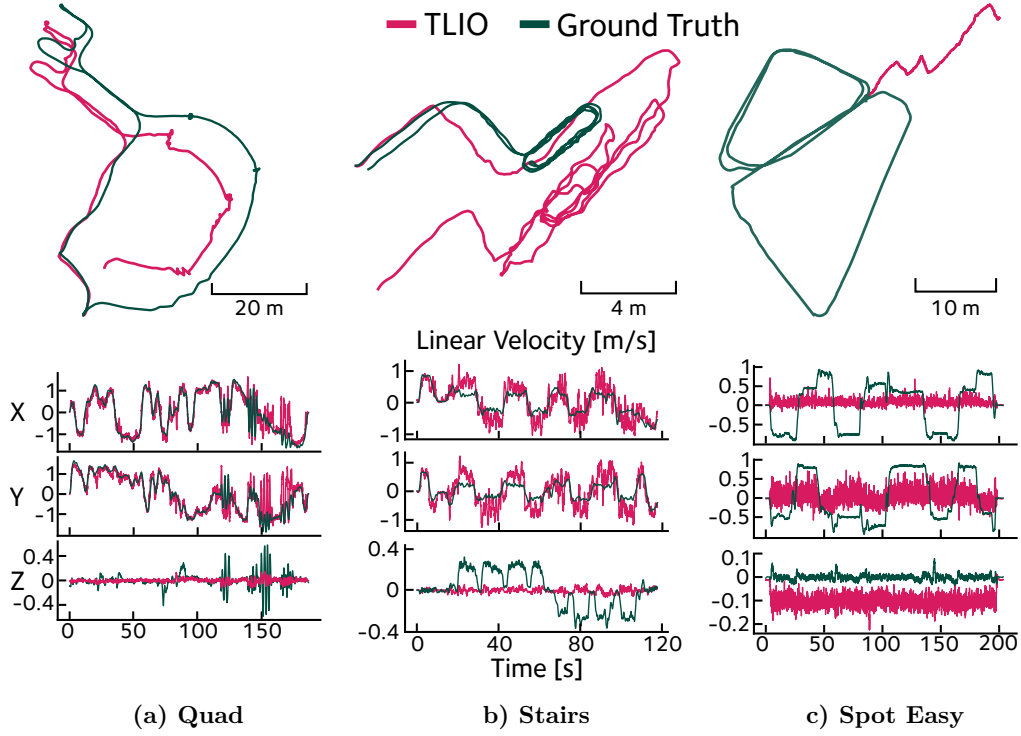


Figure 3.2: Figure from Buchanan et al. [13] (see Chapter ??). a) TLIO [96] trained on handheld data on flat ground and tested on flat ground. b) the same network is tested on a sequence in which the person walks up and down stairs. Because stair climbing was not present in the training set, inference fails significantly (n.b. z velocity estimates). c) when applied to a quadrupedal robot, because of the completely different locomotion modality, TLIO is unable to estimate any displacement. This demonstrates a limitation of all motion model-based learning methods.

and uses this as information for localisation. This is based on the assumption of constrained passageways in a structured environment such as an office. Therefore, a pedestrian might re-visit an intersection several times, making the same movements while turning down hallways; therefore enabling localisation.

While these inertial learning works have had success with tracking pedestrians, their application to robotics has so far been limited. Brossard et al. [10] developed an IMU-only state estimation algorithm specifically for wheeled vehicles. However, their method heavily relied on exploiting the constrained motion model of the vehicle and is not suitable for other types of locomotion. Very recently, inertial learning of displacement measurements was applied to Micro Aerial Vehicles (MAVs) [30]. They were able to overcome the lack of periodic motion by including the velocity commands as an input to the network.

These methods have shown impressive performance compared to pure integration INS and ZUPT-based pedestrian tracking methods. However, they become significantly less accurate for motions not present in the training data – for example a network trained on pedestrian data will not generalise to riding a bicycle. This is shown in Fig. 3.2 where TLIO has been trained on walking data on flat ground. The network cannot generalise to stair climbing or to a robot walking on flat ground. Some works have explored applying domain adaptation methods to improve transferring to different motions [26, 9] and have been able to reduce the drop in accuracy.

3.2 Proprioceptive State Estimation

In this section I review the literature of estimation methods using proprioceptive sensing such as haptics and kinematics. These are often combined with IMUs to enable proprioceptive state estimation.

3.2.1 Haptic Sensing

Haptic perception is the sense of touch which, in robotics, is a problem of estimating the magnitude and direction of force vectors which act on the robot’s body. There has been extensive research in estimating these forces, usually for collision detection [58]. Typically, this is done with a combination of encoders and torque sensors in the joints and occasionally force sensors in the feet. Assuming a rigid-body model of the robot, it is then possible to compute external forces exerted anywhere on the body [100]. Alternatively, small force sensors can be placed on and all around the robot to simulate a robotic skin [99]. In robotic arm manipulation force sensor arrays are often used in the end effectors [67].

While these methods are largely limited to fixed base robots, legged robots are continuously interacting with the ground which makes contact measurements vital. The contact state of the robot’s feet are often included in the state vector and can be measured with dedicated contact sensors in the feet, although these are liable to break [43]. Contact can also be inferred through sensors in the leg

joints [22, 73] by estimating the ground reaction force and foot velocity. However, this can require significant tuning and will not work reliably for all terrain (for example soft, deformable terrain).

Deep learning has been applied to this problem as well [95, 113]. In an approach from Lin et al. [95] their deep neural network could predict binary contact state with high accuracy, but their method could not generalise to new robots or controllers without new ground truth data labels. Piperakis et al. [113] solved this problem by developing a method trained in simulation and deployed on real, different robot platforms.

Other works have focused on estimating terrain surface normals, an important quantity to know for locomotion on rough terrain. Valsecchi et al. [141] used a passive flat foot design (See Fig. 3.1b) with two IMUs to estimate the surface normal.

Some works have sought to use the robot’s legs not just for locomotion but to also infer information about the terrain. Early work by Krotkov [87] proposed methods to measure terrain friction, stiffness and geometry from each footstep using a force/torque sensor located in the foot. The first haptic terrain classification method was introduced by Hoepflinger et al. [66]. They used a single leg detached from the robot to classify several different terrain types using only joint motor currents and contact forces. Later, Kolvenbach et al. [84] classified terrain using motor torque sensors. This method was extended to classify different qualities of concrete and was used for autonomous inspection of a sewer [85].

While the previous works required dedicated probing motions from the robot, some recent work has shown terrain classification while the robot walks naturally. Bednarek et al. [5] achieved this by using a sensorized foot with force/torque sensors (see Fig. 3.1b). Alternatively, Wellhausen et al. [151] fused vision with joint torques to assign costs to terrain based on difficulty to negotiate. Finally, Zürn et al. [165] used contrastive learning with acoustics and vision to train a self supervised labelling system based on the assumption that terrain which looks similar will sound similar when driven over.

3.2.2 Haptic Localisation

While there is a significant body of work on haptics in robotics, there are few methods which use the information obtained from this sensing for state estimation. Some works have sought to use contact states as information to estimate the pose of a known object in a robot hand. This in-hand haptic localisation typically uses particle filters as the estimation algorithm.

In tactile manipulation, some work has focused on localising an object in a robotic hand using force sensors. Vezzani et al. [144] proposed an online algorithm for tactile localisation using the Unscented Particle Filter (UPF) [142] and localised four different objects in the robot’s reference frame assuming a static base. This assumption allows them to overcome the sparsity of the measurements from the fingertips by using of a window of past measurements. Another in-hand haptic localisation method was proposed by Chalon et al. [23]. They also used a particle filter with the particles representing the object’s pose and particle weights computed according to a function which penalised both finger/object co-penetration and the distance between the object and the fingertip that detected the contact.

For legged robots the problem is instead to localise with respect to a known terrain. The robot’s feet can be used to gather information about the terrain and can be thought of as “grasping” the ground. Given a pose estimate (for example from a filter-based state estimator, see Section 3.2.3), the joint positions and the foot contact states, a legged robot is able to identify its most likely location within a prior map. This problem is analogous to a blindfolded person in a room with which they are familiar, using their hands to localise within the room through touch.

Chitta et al. [29] presented the earliest example of legged haptic localisation. In their work, they presented a proprioceptive localisation algorithm based on a particle filter for the LittleDog [109], a small electric quadruped. The robot was controlled to walk over known irregular terrain with a statically stable gait. A motion capture system was used for controlling the robot and to record a ground truth pose trajectory. In post-processing, their algorithm approximates the probability distribution of the linear position on the xy -plane and yaw of the robot base with a

set of particles. Each particle is sampled from the uncertainty of the state estimation filter, while the weights are updated using the L^2 norm of the height error between the map and the contact location of the feet.

Schwendner et al. [125] also demonstrated haptic localization on a wheeled robot with protruding spikes. Contacts were estimated between the spikes and the ground and compared to a prior digital elevation map. Each wheel enabled multiple contact measurements which they used to perform plane fitting as an additional measurement on larger-scale flat terrain. While able to function over large distances and flat ground, their system was highly specific to the hardware.

3.2.3 Kinematic-Inertial Odometry

Assuming a rigid robot body, it is possible to compute the position of the feet relative to the base frame using forward kinematics informed by the encoder measurements. Then, assuming a rigid environment and zero-velocity feet when in contact, the pose of the robot base can be estimated while walking. This, in combination with inertial sensing has enabled many legged robot state estimators.

The earliest work to develop a dead-reckoning algorithm using this method was Roston and Krotkov [121]. They directly solved for leg kinematics as a least squares problem with Singular Value Decomposition (SVD). They could solve this directly by exploiting the fact that their hexapedal robot constantly maintained at least three feet in contact with the ground.

Lin et al. [94] presented the first work which combined information from both kinematics and an IMU in an EKF. They also used a hexapod and assumed three feet in contact although adding the inertial data enabled them to execute a more dynamic tripedal running gait. Bloesch et al [8] introduced an observably-constrained EKF which tracked the position of the robot's feet in addition to its body pose.

This state estimator was later improved with a new filtering approach called TSIF [7]. This method employs a residual-based modelling of the available information, eliminating the need for an explicit process model. This allows for greater flexibility of the approach and they demonstrated experiments on an MAV

and legged robot. TSIF is also the current, default state estimator installed on the ANYmal quadruped robot [69]. However, this state estimator, like all purely proprioceptive methods, suffers from continual incremental drift over extended periods of time. This makes it unsuitable for long-term navigation purposes.

In contrast to the previous filtering-based methods, kinematic-inertial odometry can also be solved using smoothing (see Section 2.5). Kin et al. [81] formulated the problem as solving the MAP and performed non-linear optimisation on $SO(3)$. They solved for the optimal solution balancing costs from the IMU and the kinematics while using slip detection to reject kinematic information when slipping.

Recently factor graphs have become a more important method for legged robot state estimation. First, in a proprioceptive-only method, Hartley et. al. [61] introduced the contact factor to connect two robot states. For each step, a node is added to the graph at lift-off and touch-down with forward kinematics connecting the states. Hartley et al [59] also, later, presented an invariant EKF for kinematic-inertial state estimation providing additional derivations, experiments under different initial conditions and an open source C++ implementation.

Some learning-based approaches also exist [92, 74] however these works focus more on developing robust controllers. State estimation is a component of the learning process but these works do not isolate the module to specifically examine accuracy or drift.

All of these methods are odometry and therefore subject to increasing drift over time. Additionally, while some methods assume noise on the contact [7, 61, 59], it is often insufficient to account for significant slipping which can be common in real-world situations e.g. ice or wet surfaces.

3.3 Exteroceptive State Estimation

In this section, I briefly cover the most recent literature on Visual Odometry (VO) and VIO. This is an extremely broad topic which has been extensively researched. Therefore, I focus on methods and systems which have been developed specifically for legged robots or for visually challenging environments.

3.3.1 Visual Odometry

Estimation of the ego motion of a camera from a series of images is known as Visual Odometry. In the seminal work PTAM [83] this was done by detecting corner features in each images and matching them between images. Then, the poses and landmark locations were jointly optimised using a sliding window bundle adjustment. The reprojection error was minimised using sliding window bundle adjustment. ORB-SLAM [107] is based on this workflow, adding SLAM capabilities. This workflow has proven to be very efficient and has been used in many other works.

This method of matching features is known as an “indirect” method. This is in contract to the “direct” methods such as LSD-SLAM [42] and DSO [41] which estimate motion directly from the pixel intensities. This is done by minimizing the photometric error of, usually, a set of pixels with a large image gradient e.g. edges and corners. SVO [47] is a “semi-direct” method which operates between these concepts by using direct methods for frame-to-frame estimation but maintains a local map with associated keyframes and features for pose refinement.

Deep learning is becoming increasingly important for VO. Several works have used deep neural networks for feature detection and matching [38, 54]. These works have shown that learned features are generally more robust to lighting changes than hand tuned features.

Recently, new learning methods are capable of predicting motion directly from a sequence of images, bypassing any need to detect features or minimise photometric error. DeepVO [150] was the first end-to-end learning-based VO method. They used a combination of convolutional layers which learn latent image features that are then passed to an RNN for pose estimation. DROID-SLAM [134] used a similar workflow but instead estimated optical flow. In a separate thread, global bundle adjustment is done over the full history or poses, keyframes and flow.

In iMAP [133], the authors used a Multi-layer Perceptron (MLP) as an implicit neural representation of the SLAM map. Their method required no pre-training. The network is randomly initialised at start and learns online to build the map of the world by minimising re-projection error as a learning loss.

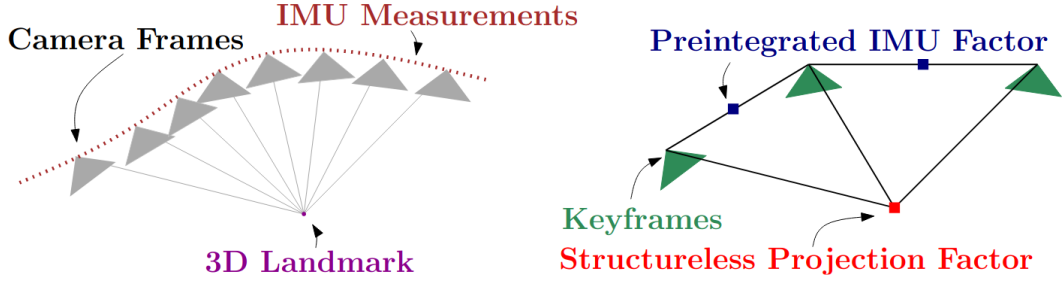


Figure 3.3: Example of preintegration from Forster et al. [46]. A subset of camera frames are selected as keyframes and the IMU measurements are integrated between them. This integration can occur while waiting for the next keyframe so that the IMU factor is already computed once the image arrives. This process is known as preintegration.

3.3.2 Visual-Inertial Navigation Systems

VIO systems allow the strengths of each sensor to counteract the weaknesses of the other. Whereas classical INS will drift rapidly due to noise, vision enables the IMU biases to become observable if there is sufficient movement [101]. Conversely, in visually challenging situations such as low-light, particulates or during dynamic motions (causing motion blur) IMUs can provide accurate estimation for short periods of time.

IMU Preintegration

Early VIO systems fused IMU and vision in EKFs using the IMU measurements as predictions which were then corrected by the vision measurement update [105]. More recently, with the advent of graph-based SLAM different works have sought to compress IMU measurements into a single measurement which can be jointly optimised with the camera measurement.

Lupton et al. [97] first introduced “preintegration” theory in which IMU measurements are integrated between two consecutive keyframes as they arrive and then are added to the graph as a single constraint. As IMU measurements arrive at a faster rate than camera images, they can be preintegrated as soon as they are available and, once the next keyframe is available, they can then be jointly optimised with the image data to solve for the state [46] (See Fig. 3.3). Preintegration relies on several

assumptions such as a constant acceleration between accelerometer measurements and constant bias during the preintegration.

Eckenhoff et al. [40] later developed a closed-form preintegration method which assumed constant local acceleration between measurements rather than global, reducing errors. Le Gentil et al. [91, 90] used Gaussian Processes (GPs) to fit acceleration data enabling preintegration in continuous time. Other work added components for time offsets between camera and IMU [158].

The GTSAM library [36] provides two different implementations of an IMU preintegration factor based on [46] which differ in how noise is propagated during preintegration. I use GTSAM for a significant portion of this work and, therefore, investigated the difference between the implementations. In Appendix A I prove they are similar but not identical under certain conditions. However, in practice I find estimation results are extremely similar regardless of which factor is used.

VIO systems

Some notable VIO works include OKVIS [93] and VINS-MONO [115] which both use factor graphs to optimise a sliding window of visual landmarks with IMU measurements. Filtering has also been used successfully such as ROVIO [6] which tightly integrates visual and inertial constraints in an EKF. Open-VINS is another open sourced EKF-based implementation developed to aid the research community in rapid development of systems [53]. VIO systems can also be used for SLAM such as ORB-SLAM2 and ORB-SLAM3 [108, 20].

Deep learning has also been applied to VIO. Clark et al. [31] introduced VINet, the first end-to-end VIO method. They used an LSTM to process the IMU data and a CNN to process the images. The feature vectors were then concatenated together and passed to another LSTM. Almalioglu et al. [2] implemented a self-supervised VIO system in which they also trained separate networks for vision and IMU measurements as well as predicting depth. An attention mechanism weights the two odometry sources and then an LSTM outputs the fused estimate. Other works focused on deep learning for sensor fusion [27, 25], using attention mechanisms to

either weight or explicitly select learned features in vision, depth or inertial data. This can effectively reject fully or partially corrupted data.

While these systems have shown impressive performance, if the vision is failing for too long, then the system reverts to a mechanisation-based INS. This will cause significant drift from which the system might not be able to recover if vision returns. Therefore, some research has explored ways to make VIO more robust.

Zhao et al. [164] proposed an IMU-centric system which frames the VIO problem around estimating the IMU biases. This is done by using vision and lidar odometry measurement as priors with the IMU preintegration to estimate biases in a factor graph. Wisth et al. [153] also developed a system fusing vision and lidar but tracked planes and lines in the point cloud as features rather than using ICP. While lidar can be useful in low lighting, in narrow spaces the lidar may not be able to observe the ceiling and floor, resulting in degeneracy along the vertical direction.

Sturmberg et al. [130] proposed an initialisation method using visual and inertial factors in a second factor graph with a delayed marginalization. In cases where there was vision failure for several seconds, the second factor graph can be used for re-initialisation.

Other attempts to improve VIO have focused on the camera hardware, for example using multiple cameras with different viewpoints [162] or omnidirectional cameras [116]. Alternatively, event cameras are a different type of vision sensor which asynchronously updates pixels based on changes in intensity [106]. However these methods are complex to integrate and not widely available.

3.3.3 Fusing Legged Kinematic Information

Another option for VIO on legged robots is the fusion of kinematic information with vision and inertial data. As kinematic information is usually sampled at rates much faster than the camera, several measurements are typically accumulated between keyframes.

Filtering-based methods have been used successfully for this purpose. Fallon et al. [43] used a separate EKF for inertial-kinematic estimation and a Gaussian

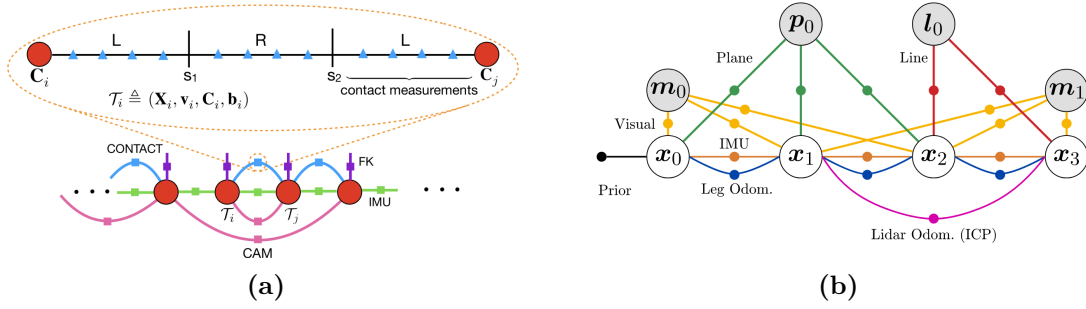


Figure 3.4: a) Factor graph design from Hartley et al. [60]. On top is shown how three footsteps for a bipedal robot (alternating left, right, left) are merged into a single contact factor. b) factor graph design from Wisth et al [156]. Individual vision and lidar features locations are jointly optimised with IMU and kinematic factors in contrast to [60] which uses visual relative pose factors.

Particle Filter as a localisation module to incorporate the lidar measurements. The Pronto state estimator [21] uses only a single EKF but maintains a history of past states. When a keyframe or lidar update arrives, Pronto can retroactively update the past state and chain of states up to the current time. Piperakis et al. [112] presented an approach using probabilistic outlier detection to reject erroneous measurements from vision and lidar.

Factor graphs have also been used extensively in this area. Hartley et al. [60] expanded on their proprioceptive work by adding visual landmarks to the graph. To account for the slower rate of keyframes, they preintegrate several kinematic measurements – similar to IMU – between the keyframes. An example factor graph of this work is shown in Fig. 3.4a.

Wisth et al. [155] first proposed a similar factor graph framework, expressing the kinematic factors as between factors where the relative pose was estimated by a stand-alone EKF [7]. Later, in [154] they improved on the factor graph formulation by introducing a velocity bias into the robot’s state which represents model uncertainties, for example from slipping or leg flexibility. A novel preintegrated velocity factor accurately estimates a velocity bias using data from the IMU and stereo vision. This system was fully matured in [156] with added capability for incorporating lidar features or relative pose measurements from ICP as shown in Fig. 3.4b.

3.3.4 Learning IMU Noise Models

A different approach to improving VIO performance in visually challenging environments is to improve the IMU estimation. The vision information helps to correct for drift caused by noise in the IMU. If the noise could be reduced or estimated without vision, there would be less dependence on this sensing modality.

Several recent works have used deep learning methods to learn IMU noise models. In this case, the IMU measurements are passed through a network trained to “denoise” them and output an estimate of the perfect IMU measurements. These denoised signals can then be used directly as input to a VIO pipeline. This is in contrast to the methods described in Section 3.1.3 where the goal is to learn the motion model of the subject.

Brossard et al. [11] used a Convolutional Neural Network (CNN) to denoise a gyroscope using the output to correct for the true angular velocity. The CNN used dilated convolutions to increase the temporal coverage over longer sequences. Zhang et al. [163] trained a Recurrent Neural Network (RNN) to denoise both gyroscope and accelerometer measurements. Their network used IMU measurements as input to estimate a corrected measurement which, when integrated, reduced pose error. This eliminated the need for the model to learn the underlying mechanization equations. Similarly, Steinbrener et al. [129] performed denoising on IMU measurements, comparing Long Short-term Memory (LSTM) [65] and Transformer [143] architectures, finding the LSTM to be more effective.

The main drawback of these methods is that they do not distinguish between different noise sources and, as a result, it is not clear what the network has learned to remove from the IMU measurements. For example, most results were primarily demonstrated on drones which, when flying, introduce high frequency vibrations affecting the IMU measurements. It is unclear if the noise characteristics being estimated was caused by the vibrations, the internal state of the IMU (i.e. the true bias), the motion of the drone, or a combination of the above.

3.4 Conclusion

State estimation is a fundamental problem for any mobile platform. Exteroceptive sensors, such as cameras and lidar, have been used with great success in a wide variety of applications. However, these methods can fail in common, visually challenging environments. Adding sensing from legged robot joints or from an IMU can improve robustness but they introduce new challenges such as detecting contact states and IMU biases.

Separately, there has been recent success in deep learning for inertial odometry of pedestrians. These methods exhibit impressive performance using only proprioceptive sensing. However, current methods for proprioceptive sensing are nearly all incapable of localisation. This means they will drift without bound, limiting their usefulness for long-term operation.

So a gap exists between highly accurate exteroceptive methods which may struggle in vision-denied environments, and the robust proprioceptive methods which are less accurate and cannot localise. In this thesis I lay out several new methods for state estimation that works towards closing this gap.

Part II

Haptic Localisation

4

Haptic Localisation with Geometric Information

This chapter introduces a new haptic localisation method capable of localising a legged robot in 3-dimensions using only contact information. This could be very useful for routine inspection tasks in known environments such as a factory or power plant. If vision is lost due to sudden poor lighting, smoke or sensor failure, the robot would be able to blindly re-localise and navigate to safety.

Localising purely from sparse contact information can lead to multiple hypotheses of the robot's pose. Because of the multimodal nature of this problem, I use Sequential Monte Carlo methods to propagate particles as hypotheses of the robot pose. Contact measurements are used with a prior geometric model the environment to weight the particles based on how well they explain current contact state. The limiting factors for this method are a prior geometric map of the environment and an assumption of geometrically varied terrain (this method will not work for extended periods of time on flat ground). This work also demonstrates 3D probing of walls using one of the robot's legs. This enables the robot to perform wall following and re-localise by "feeling its way."

4.1 Haptic Sequential Monte Carlo Localization for Quadrupedal Locomotion in Vision-Denied Scenarios

The following article was presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) 2020 [16]. An accompanying video is available online at: <https://www.youtube.com/watch?v=VfA8DAyr8t0>.

Haptic Sequential Monte Carlo Localization for Quadrupedal Locomotion in Vision-Denied Scenarios

Russell Buchanan, Marco Camurri, Maurice Fallon

Abstract—Continuous robot operation in extreme scenarios such as underground mines or sewers is difficult because exteroceptive sensors may fail due to fog, darkness, dirt or malfunction. So as to enable autonomous navigation in these kinds of situations, we have developed a type of proprioceptive localization which exploits the foot contacts made by a quadruped robot to localize against a prior map of an environment, without the help of any camera or LIDAR sensor. The proposed method enables the robot to accurately re-localize itself after making a sequence of contact events over a terrain feature. The method is based on Sequential Monte Carlo and can support both 2.5D and 3D prior map representations. We have tested the approach online and onboard the ANYmal quadruped robot in two different scenarios: the traversal of a custom built wooden terrain course and a wall probing and following task. In both scenarios, the robot is able to effectively achieve a localization match and to execute a desired pre-planned path. The method keeps the localization error down to 10 cm on feature rich terrain by only using its feet, kinematic and inertial sensing.

I. INTRODUCTION

Perception is an essential prerequisite for legged robot navigation on challenging terrain. Several works have demonstrated the potential of such machines when they are endowed with effective state estimation [1], [2] and visual perception capabilities [3], [4], [5]. However, vision-denied areas such as dusty underground mines or foggy sewers are places where legged robots are likely to be deployed [6] (see Fig. 1). Additionally, unrecoverable sensor failures are to be expected when a robot is deployed for long-term autonomous missions. Performance should degrade gracefully in such conditions by taking advantage of all proprioceptive sensor measurements and prior information available to the robot.

In this work, we explore haptics as the main source of robot localization information on non-degenerate (i.e., non flat) ground. We present an algorithm for 6 Degrees of Freedom (DoF) localization of a quadrupedal robot based on Sequential Monte Carlo methods.

Given the pose (estimated by a proprioceptive state estimator [1]), the joint kinematics and the foot contact states, the robot is able to identify its most likely location within a prior map and its past trajectory. This is done using only proprioceptive sensors, such as Inertial Measurement Unit (IMU), encoders and torque sensors. To this end, the robot uses its legs to probe the terrain and uses the contact events to estimate its trajectory relative to the terrain map. Unlike previous works, our method can localize in 6-DoF which

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



Fig. 1. ANYmal using its foot to measure the vibrations of a piece of mining equipment. Tasks such as this might require localization even when vision cannot be used. By exploiting prior knowledge of the 3D environment, the robot could use contact events to localize itself. Source: ETH Zurich.

we demonstrate with an experiment using a foot to probe in 3-dimensions. This problem is analogous to a blindfolded person in a room they know well, using their hands to localize within the room through touch.

We tested our algorithm using an ANYbotics ANYmal quadruped robot in two different scenarios. First, we constructed a diverse terrain course with features such as ramps, chevrons and steps. The robot traversed several loops of this course, comparing contact points to the prior map (represented as a 2.5D digital elevation map). In the second scenario, the robot probed against two perpendicular walls to improve its estimated pose, with the map represented as a 3D point cloud.

The remainder of this document is structured as follows: Section II summarizes the recent developments in the field of legged haptic localization and the related problem of in-hand tactile localization; Section III defines the mathematical background of the legged haptic localization problem; Section IV describes our proposed haptic localization algorithm; Section VI presents the experimental results collected using the ANYmal robot; Section VII provides an interpretation of the results and discusses the limitations of the approach; finally, Section VIII concludes the paper with final remarks.

II. RELATED WORKS

Pioneering work on how to use robot's legs not just for locomotion but also to infer terrain information such as friction, stiffness and geometry has been presented by Krotkov [7]. This is a useful capability when legged robots have to operate in vision-denied areas. The idea was recently

revisited by Bednarek et al. [8] to classify different terrain types in order to improve locomotion parameters and by Kolvenbach et al. [6] to inspect concrete in sewage systems by vibration analysis of the sewer floor.

While these works were more focused on using haptic information to improve locomotion or to monitor the terrain, little research has been conducted on how to use haptic information to localize the robot in the environment. However, a similar problem has been explored in grasping: tactile localization.

Tactile localization involves the estimation of the 6-DoF pose of an object (of known shape) in the robot's base frame by means of kinematics of the robot's fingers and its tactile sensors. The pose of the object (in robot base coordinates) is inferred by matching the set of all positions of the sensed contacts (also in base coordinates) with the 3D model of the object being grasped.

Intuitively, this problem is analogous to legged robot haptic localization, if we imagine the entire robot as a hand trying to "grasp" the ground. The objective of legged haptic localization is estimating the pose of the robot relative to a fixed object (in this case the terrain) by means of its "fingers" (in this case the robot's feet), instead of the pose of the object being grasped relative to the robot.

A. Sequential Monte Carlo Methods

Tactile localization has typically been addressed using Sequential Monte Carlo (SMC) methods, a subfamily of which is called *particle filters* [9]. Since the object can have any shape, the probability distribution of its pose given tactile measurements can be multimodal. This excludes all Gaussian estimators such as Kalman filters. In contrast, SMC methods can maintain a discrete approximation of an arbitrary distribution by generating many hypotheses (also called particles) from a known *proposal distribution*. If the number of particles is large enough and the proposal distribution covers the state space well (i.e., it captures the areas of high density of the true distribution), the target distribution is well approximated by a list of state particles and their associated importance weights.

SMC is sensitive to the dimension of the state space, which should be low enough to avoid combinatorial explosions or particle depletion. State-of-the-art methods aim to reduce this dimensionality and also to sample the state space in an efficient manner. In the SLAM community, this has been typically achieved by careful design of the proposal distribution and adaptive importance resampling to avoid particle depletion [10].

B. Tactile Localization in Manipulation

Vezzani et al. [11] proposed an algorithm for tactile localization using the Unscented Particle Filter (UPF) [12] and tested it on the iCub robot (equipped with contact sensors at the fingertips) to localize four different objects in the robot's reference frame. The algorithm is recursive and can process data in real-time as new measurements become available. The object and the robot's base are assumed to be static, therefore

the pose to be estimated is a fixed parameter. This assumption allows the use of a window of past measurements to better address the sparsity of the measurements, which consist of a series of single finger touches. For legged haptic localization, the assumption of both a static robot and terrain does not hold in general and more general methods are required.

Chalon et al. [13] proposed a particle filtering method for online in-hand object localization that tracks the pose of an object while it is being manipulated by a fixed base arm. The pose is subsequently used to improve the performance of pick and place tasks (e.g., by placing a bottle in an upright position). The particles representing the object's pose are initialized by a Gaussian distribution around the true initial pose of the object, acquired by a vision system (used only at start). The particle weights are taken from the measurements according to a function which penalizes both finger/object co-penetration and the distance between the object and the fingertip that detected the contact. The best estimate is the one with highest weight.

This work, as with many others in the field of grasping, assumes the hand can envelop the object, since they have comparable sizes. This is not the case with legged robots touching the ground, which makes the problem generally harder and involves more uncertainty. For example, a hand touching a very large object has fewer clues as to the object's pose, because the contact points would not be equally spread across the object but concentrated in a small area of it.

C. Haptic Localization of Walking Robots

To the best of our knowledge, Chitta et al. [14] is the only significant prior example of legged haptic localization to date. In their work, they presented a proprioceptive localization algorithm based on a particle filter for the LittleDog, a small electric quadruped. The robot was commanded to perform a statically stable gait over a known irregular terrain course, using a motion capture system to feed the controller. While walking, the algorithm approximated the probability distribution of the base state with a set of particles. The state was reduced from six to three dimensions: linear position on the xy -plane and yaw. Each particle was sampled from the uncertainty of the odometry, while the weight of a particle was determined by the L2 norm of the height error between the map and the contact location of the feet. The algorithm was run offline on eight logged trials of 50 s each.

In our proposed work, we improve upon [14] both theoretically and experimentally. First, we estimate the full past trajectory at every step instead of the most recent pose. This allows us to estimate poses which are globally consistent over the entire trajectory of the map. Second, we perform localization in the full 6-DoF of the robot, instead of limiting the estimation to the x , y and yaw dimensions. Third, we experimentally demonstrate the effectiveness of the algorithm by running it online on our quadruped robot. This is the first time such an algorithm has been successfully executed online on real hardware and used in a closed loop navigation system to successfully execute a planned path.

Finally, we demonstrate the algorithm in three dimensions in online experiments.

III. PROBLEM STATEMENT

Let $\mathbf{x}_k = [\mathbf{p}_k, \boldsymbol{\theta}_k]$ be a robot's pose at time k , composed of the base position in the world frame $\mathbf{p} \in \mathbb{R}^3$ and orientation $\boldsymbol{\theta} \in SO(3)$. With a slight abuse of notation, we will use the same symbol for its homogeneous matrix form $\mathbf{x}_k \in SE(3)$. We assume that for each timestep k , an estimate of the robot pose $\tilde{\mathbf{x}}_k$ and its covariance $\Sigma_k \in \mathbb{R}^{6 \times 6}$ is available from an odometric estimator¹. We also assume that the location of the robot's end effectors in the base frame $(\mathbf{d}_{LF}, \mathbf{d}_{RF}, \mathbf{d}_{LH}, \mathbf{d}_{RH}) \in \mathbb{R}^{3 \times 4}$ are known from forward kinematics and their binary contact states $C \in \mathbb{B}^4$ from inverse dynamics.

For simplicity, we neglect any uncertainties due to inaccuracy in joint encoder readings or limb flexibility. Therefore, the propagation of the uncertainty from the base to the end effectors is straightforward to compute. For brevity, the union of the aforementioned states (pose and contacts) at time k will be referred as the *quadruped state* $\mathcal{Q}_k = \{\tilde{\mathbf{p}}_k, \tilde{\boldsymbol{\theta}}_k, \Sigma_k, \mathbf{d}_{LF}, \mathbf{d}_{RF}, \mathbf{d}_{LH}, \mathbf{d}_{RH}, C_k\}$.

Our approach can localize against both 2.5D terrain elevation maps and 3D point clouds. These prior maps are denoted with $\mathcal{M}$.

Our goal is to use a sequence of quadruped states, and their corresponding uncertainties Σ_k , to estimate the most likely sequence of robot states up to time k as:

$$\mathcal{X}_{0:k}^* = [\mathbf{x}_0^*, \mathbf{x}_1^*, \dots, \mathbf{x}_k^*] \quad (1)$$

such that the likelihood of the contact points to be on the map is maximized.

IV. PROPOSED METHOD

To perform localization, we sample a predefined number of particles at regular intervals from the pose distribution provided by the odometry, as described in Section IV-B, and we compute the likelihoods of the measurements by comparing each particle to the prior map so as to update the weight of the particle filter (see Section IV-C). For convenience, we give a brief summary of particle filter theory in Section IV-A.

A. Particle Filter Theory

In a particle filtering framework, the objective is to approximate the posterior distribution of the state $\mathbf{x}_k$ given a history of measurements $\mathbf{z}_0, \dots, \mathbf{z}_k = \mathbf{z}_{0:k}$ as follows:

$$p(\mathbf{x}_k | \mathbf{z}_{0:k}) = \sum_i w_{k-1}^i p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{x}_{k-1}^i) \quad (2)$$

where w^i is the *importance weight* of the i -th particle; $p(\mathbf{z}_k | \mathbf{x}_k)$ is the measurement likelihood function and $p(\mathbf{x}_k | \mathbf{x}_{k-1}^i)$ is the dynamic model for the i -th particle state.

¹uncertainties for the rotation manifold are maintained in the Lie tangent space, as in [15]

Since $p(\mathbf{x}_k | \mathbf{z}_{0:k})$ is normally unknown, the state $\mathbf{x}_k$ is typically sampled from $p(\mathbf{x}_k | \mathbf{x}_{k-1}^i)$, yielding:

$$p(\mathbf{x}_k | \mathbf{z}_{0:k}) = \sum_i w_{k-1}^i p(\mathbf{z}_k | \mathbf{x}_{k-1}^i) \delta(\mathbf{x}_k - \mathbf{x}_{k-1}^i) \quad (3)$$

where $\delta(\cdot)$ is the Dirac delta function.

B. Locomotion Control and Sampling Strategy

In the considered situation, the robot would be locomoting blindly. As a result, it is likely to do so with a statically stable gait while using the terrain to achieve a position fix.

Statically stable gaits alternate between three and four points of support by moving a single foot. When a new four-point support phase occurs at time k , the estimated robot state $\mathcal{Q}_k$ from the onboard state estimator is collected. Similarly to [14], particles are sampled from the following proposal distribution:

$$p(\mathbf{x}_k | \mathbf{x}_{k-1}^i) = \mathcal{N}(\mathbf{x}_k, \Delta \tilde{\mathbf{x}}_k \mathbf{x}_{k-1}^i, \Sigma_k) \quad (4)$$

where $\Delta \tilde{\mathbf{x}}_k = \tilde{\mathbf{x}}_{k-1}^{-1} \tilde{\mathbf{x}}_k$ is the relative pose between the estimated states at times $k-1$ and k .

The new samples $\mathbf{x}_k^i$ are drawn from a Gaussian centered at the pose of the previous iteration composed of the relative pose from the odometry, with the covariance of the odometry at time k . Since the estimator [1] fuses both IMU and leg odometry, the covariance Σ_k has very low uncertainty on roll and pitch because these two quantities are observable due to gravity. This would result in all particles having virtually the same roll and pitch which permits reduction of the sample space to the four other dimensions. This allows us to localize in the full 6-DoF space without incurring problems due to high dimensionality.

C. 2.5-Dimensional Measurement Likelihood Model

To account for uncertainties in the prior map, we model the measurement likelihood as a univariate Gaussian centered at the local elevation of each cell, with a manually selected variance σ_z (1 cm in our case). In contrast with [14], we evaluate each contact point individually. Given a particle state $\mathbf{x}_k^i$, the estimated position of a contact in world coordinates for foot f is defined as the concatenation of the estimated robot base pose and the location of the end effector in base coordinates (Fig. 2 left):

$$\mathbf{d}_f^i = (d_{xf}^i, d_{yf}^i, d_{zf}^i) = \mathbf{x}_k^i \mathbf{d}_f \quad (5)$$

Thus, the measurements and their relative likelihood functions for the i -th particle and a specific foot f are:

$$z_k = d_{zf}^i - \mathcal{M}(d_{xf}^i, d_{yf}^i) \quad (6)$$

$$p(z_k | \mathbf{x}_k^i) = \mathcal{N}(z_k, 0, \sigma_z) \quad (7)$$

where: d_{zf}^i is the vertical component of the estimated contact point location in world coordinates of foot f , according to the i -th particle; $\mathcal{M}(d_{xf}^i, d_{yf}^i)$ is the corresponding map elevation at the xy coordinates of $\mathbf{d}_f^i$.

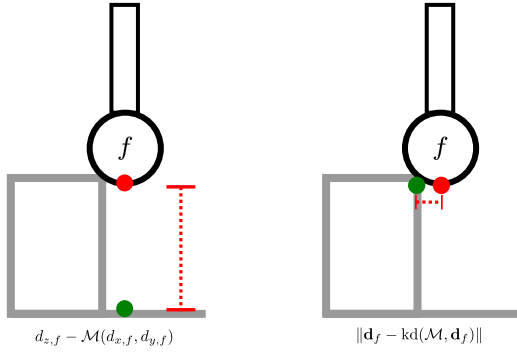


Fig. 2. Comparison between contact measurements for 2.5D (left) and 3D (right) map representations. The red dots indicate the contact point as sensed by the robot, while the green dots indicate the corresponding location returned by the map. The red line shows the magnitude of the measurement. Note how the same contact interaction generates a much lower likelihood in the 2.5D map than the 3D map, which is more representative. This is a practical limitation of using a 2.5D map.

D. 3-Dimensional Measurement Likelihood Model

Our method can incorporate contact events from 3D probing. This is useful for areas where the floor does not provide enough information to localize. Instead, the robot can probe walls and 3D objects. In this case, we instead represent the prior map $\mathcal{M} \in \mathbb{R}^{3 \times N}$ by a 3D point cloud with N points. The likelihood of a particular contact point is computed using the Euclidean distance between the foot and the nearest point in the map. This likelihood is again modeled as a zero-mean Gaussian evaluated at the Euclidean distance between the estimated contact point $\mathbf{d}_f^i$ and its nearest neighbor on the map, with variance σ_z :

$$z_k = \|\mathbf{d}_f^i - \text{kd}(\mathcal{M}, \mathbf{d}_f^i)\| \quad (8)$$

$$p(z_k | \mathbf{x}_k^i) = \mathcal{N}(z_k, 0, \sigma_z) \quad (9)$$

where $\text{kd}(\mathcal{M}, \mathbf{d}_f^i)$ is the function that returns the nearest neighbor of $\mathbf{d}_f^i$ on the map $\mathcal{M}$, computed from its k-d tree.

V. IMPLEMENTATION

Pseudocode for our method is listed in Algorithm 1. We update the filter when the robot enters a new four-support configuration. At this time k , the estimated robot pose $\tilde{\mathbf{x}}_k$ is collected and used to compute the relative motion $\Delta \tilde{\mathbf{x}}_{k-1:k}$. This is used to propagate forward the state of each particle $\mathbf{x}_k^i$ and draw a sample from the distribution centered in $\Delta \tilde{\mathbf{x}}_k \mathbf{x}_{k-1}^i$ with covariance $\Sigma_k = (\sigma_{x,k}, \sigma_{y,k}, \sigma_{z,k})$.

The weight of a particle w^i is then updated by multiplying it by the likelihood that each foot is in contact with the map. In our implementation, we modify the likelihood functions from Equations 7, 9 as:

$$p(z_k | \mathbf{x}_k^i) = \min(\rho, \mathcal{N}(z_k, 0, \sigma_z)) \quad (10)$$

where ρ is a minimum weight threshold, so that outlier contact measurements do not immediately lead to degeneracy. Resampling is triggered when the variance of the weights rises above a certain threshold. This was manually tuned once and used for all experiments. Resampling is

```

 $\mathbf{x}_0^i \sim \mathcal{N}(\mathbf{x}_0, \tilde{\mathbf{x}}_0, \Sigma_0) \quad \forall i \in N$ 
foreach four-support phase  $k$  do
   $\Delta \tilde{\mathbf{x}}_k \leftarrow \tilde{\mathbf{x}}_{k-1}^{-1} \tilde{\mathbf{x}}_k$ 
  foreach particle  $i \in N$  do
     $\mathbf{x}_k^i \sim \mathcal{N}(\mathbf{x}_k, \Delta \tilde{\mathbf{x}}_k \mathbf{x}_{k-1}^i, \Sigma_k)$ 
     $w_k^i \leftarrow w_{k-1}^i$ 
    foreach foot  $f$  do
      if 2.5D then
         $z_k \leftarrow d_{zf}^i$ 
      else
         $z_k \leftarrow \|\text{kd}(\mathcal{M}, \mathbf{d}_f^i) - \mathbf{d}_f^i\|$ 
      end
       $w_k^i \leftarrow w_k^i p(z_k | \mathbf{x}_k^i)$ 
    end
  end
   $\mathbf{x}_k^* \leftarrow \text{WeightedMean}(\mathbf{x}_k^0 \dots \mathbf{x}_k^N, w_k^0, \dots, w_k^N)$ 
   $\mathcal{X}_k^* \leftarrow [\mathbf{x}_k^0, \dots, \mathbf{x}_k^N]$ 
  if  $\text{variance}(w_k^i) > \text{threshold}$  then
     $\text{resample}(\mathbf{x}_k^i)$ 
  end
end

```

Algorithm 1: Haptic Sequential Monte Carlo Localization

necessary otherwise the particle set would disperse across the state space and many particles would have low weight. By triggering this process when the variance of the weights increases, the particles can first track the dominant modes of the underlying distribution.

A. Particle Statistics

The optimal value for the k -th iteration, $\mathbf{x}_k^*$, is computed from the weighted average of all the particles, as in [14]. However, this method does not take into account the multimodality of the particle distribution. There can be several hypotheses resulting in multiple clusters of particles whose weighted average is not a reliable estimate of the robot's location. To account for this, we check the variance of the particle positions in the x and y axes. If they are low (i.e., $\ll \sigma_{x,k}, \sigma_{y,k}$), we assume a good estimate and update the robot's full pose. However, if they are high, we update only the z component of the robot's location (the main drifting dimension). The other dimensions drift at the same rate as the state estimator, but the upward drift is corrected since the robot is always in contact with the ground.

VI. EXPERIMENTAL RESULTS

Our localization algorithm has been tested in two different scenarios: a walking task crossing a custom built terrain course and a task involving probing and following of a wall. The prior maps were captured with a Leica BLK-360 laser scanner, which provides registered point clouds with sub-centimeter accuracy.

Before testing on real hardware, the 3D model of the terrain course was imported as a simulated environment (in Gazebo) allowing us to validate the localization algorithm

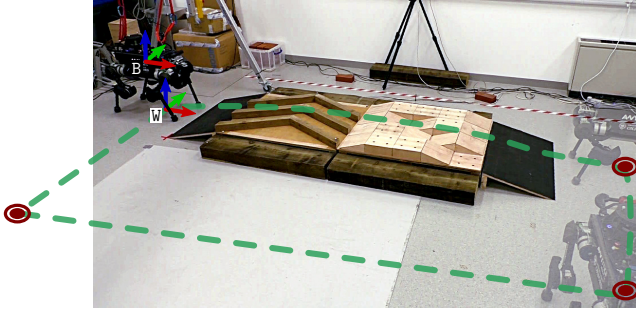


Fig. 3. ANYmal haptic localization experiments. The robot traverses the terrain, turns 90 deg right and comes back to the initial position passing through the flat area. The goals given to the planner are marked by the dark red disks, while the planned route is a dashed green line (one goal is out of the camera field of view). The world frame W is fixed to the ground, while the base frame B is rigidly attached to the robot's chassis. The mutual pose between the robot and the terrain course is bootstrapped with the Vicon.

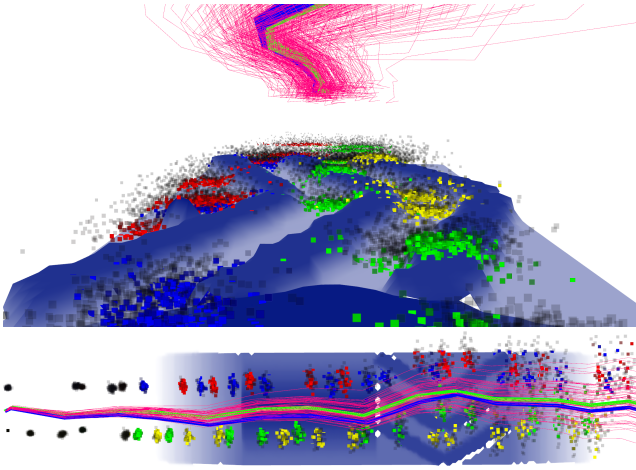


Fig. 4. *Top*: Perspective view of a simulated traversal of similar terrain to Fig. 3. At the top is the 1000 sampled trajectories shown by thin magenta lines, while the ground truth and estimated trajectory are shown with a green and blue line, respectively. *Bottom*: Top down view of the same scene. The top perspective is from the center the terrain facing right. Each squared dot represents one of 4000 contacts, colored by leg type (LF: red, RF: green, LH: blue, RH: yellow). The tone and alpha of each square is associated to the likelihood: darker and more transparent relates to a lower likelihood. The grid map is colored by height, from white to dark blue.

with different values of noise and particle set dimensions. Fig. 4 shows an example with 1000 particles depicted as candidate trajectories (in magenta) with the associated contacts colored by their likelihood (brighter/more solid color = higher likelihood). The most likely trajectory is highlighted in blue and is the closest to the ground truth (in green).

In real world experiments, the ground truth robot poses were collected at 100 Hz with millimetric accuracy using a Vicon motion capture system. At start of the experiment, the relative position of the robot within the map was measured using Vicon and used for initialization only. Thereafter, the pose of the robot was estimated using the particle filter and the Vicon was used only to compute error statistics. To account for initial errors, particles at the start were sampled from a Gaussian centered at the initial robot pose with a covariance of 20 cm.

Exp.	Dist. [m]	Time [s]	Loops	ATE TSIF	ATE HL
1	66.42	525	2	0.63	0.13
2	145.31	1097	4	2.57	0.40
3	55.67	557	2	0.52	0.19
4	68.71	604	2	0.65	0.32
5	172.65	1606	4	2.00	0.61

TABLE I

ESTIMATION PERFORMANCE ON THE TERRAIN COURSE DATASETS. ATE = ABSOLUTE TRANSLATION ERROR; TSIF = TWO-STATE IMPLICIT FILTER [1]; HL = HAPTIC LOCALIZATION.

A. Terrain Course Experiments

In the first scenario, the robot was commanded to navigate between four walking goals at the corners of a rectangle. One of the edges required crossing a 4.2 m terrain course composed of a 12° ascending ramp, a 13 cm high chevron pattern, an asymmetric composition of uneven square blocks and a 12° descending ramp (Fig. 3). After crossing the wooden course, the robot returned to the starting position across a portion of flat ground, which tests how the system behaves in feature-deprived conditions.

While impressive, blind reactive locomotion has been developed by a number of groups including [16], [4], unfortunately our haptic controller was not sufficiently reliable to cross the terrain course so we resorted to use of the statically stable gait from [3] which used a depth camera to aid footstep planning. Using our approach, localization was performed without access to any camera information.

To demonstrate repeatability, we have performed five experiments of this type, for a total distance traveled of more than 0.5 km and 1 h 13 min duration. A summary of the experiments is presented in Table I, where the haptic localization shows an overall improvement between 50% to 85% in the Absolute Translation Error (ATE). With the haptic localization, the ATE is 33 cm on average, which reduces to 10 cm when evaluating only the feature-rich portion of the experiments (i.e., the terrain course traversal).

For Experiments 1 and 2, the robot was manually operated to traverse the terrain course, completing two and four loops, respectively. In Experiments 3–5, the robot was commanded to follow the rectangular path autonomously. In these experiments, the haptic localization algorithm was run online and in closed-loop and effectively guided the robot towards the goals (Fig. 5). Using only the prior knowledge of the terrain geometry and the contact events, the robot stayed localized in all the runs and successfully tracked the planned goals. This can be seen in Fig. 5, where the estimated trajectory (in blue) drifts on the xy -plane when the robot is walking on the flat ground, but it becomes re-aligned with the ground truth when the robot crosses the terrain course again.

Fig. 6 shows in detail the estimator performance for each of the three linear dimensions and yaw. Since position and yaw are unobservable, the drift on these states is unbounded.

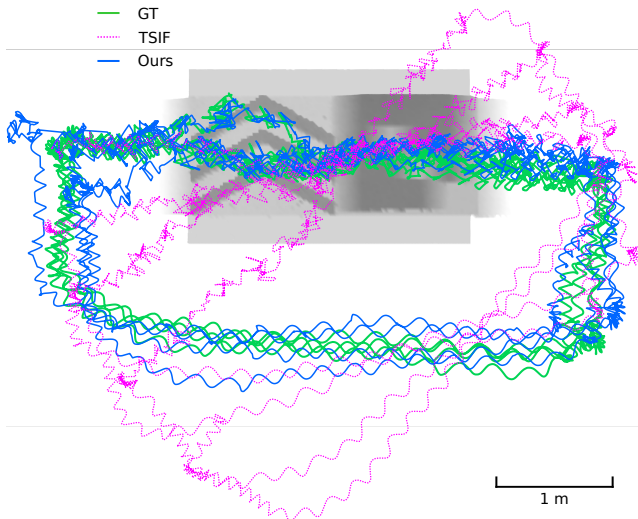


Fig. 5. Top view of the estimated trajectories from TSIF (dashed magenta), haptic localization (blue) and ground truth (green) for Experiment 2.

In particular, the error on the odometry filter (TSIF [1], magenta dashed line) is dominated by upward drift (due to kinematics errors and impact nonlinearities, see third plot) and yaw drift (due to IMU bias, see bottom plot). This drift is estimated and compensated for by the haptic localization (blue solid line), allowing accurate tracking of the ground truth (green solid line) in all dimensions. This can be noted particularly at the four peaks in the z -axis plot, where the estimated trajectory and ground truth overlap. These times coincide with the robot's summit of the terrain course.

B. Online Haptic Exploration on Vertical Surfaces

The second scenario involved a haptic wall following task: the robot started in front of a wall with an uncertain location. The particles were again initialized with 20 cm position covariance. To test the capability to recover from an initial error, we applied a 10 cm offset in both x and y from the robot's true position in the map. At start, the robot was commanded to walk 1 m to the right (negative y direction) and press a button on the wall. To accomplish the task, the robot needed to "feel its way" by alternating probing motions with its right front foot and walking laterally to localize inside the room. The fixed number of probing motions was pre-scripted so with each step to the right, the robot probed both in front and to its right. In this scenario, the prior map is represented as a 3D point cloud with a minimum distance of 1 cm between its points. The whole experiment was executed blindly with the static controller from [3].

As shown in Fig. 7 the robot was able to correct its localization and complete the task of touching the button. The initial probe to the front reduced uncertainty in the robot's x and z directions, which reduces the particles to an ellipsoidal elongated along y . As the robot moves, uncertainty in the x direction increases slightly. By touching the wall on the right, the robot re-localized in all three dimensions in much the same way as a human following a wall in the dark would. The re-localization allows the robot to press the button,

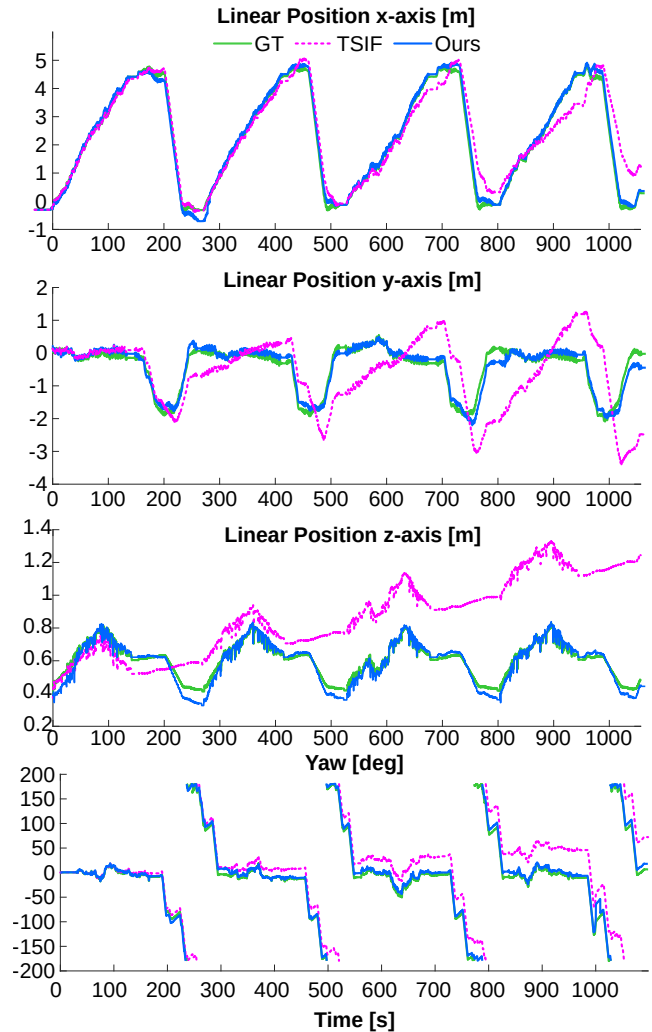


Fig. 6. Comparison between the estimated position from TSIF (dashed magenta) and haptic localization (blue) against Ground Truth (green) for Experiment 2. After 200 s, the estimation error in TSIF has drifted significantly upward and in yaw. In particular, the upward drift is noticeable in the third plot, where the values grow linearly. The drift is eliminated by the re-localization against the prior map.

demonstrating the generalization of our algorithm to 3D. The final position error was: $[7.7, -3.7, -0.2]$ centimeters in the x , y and z directions².

VII. DISCUSSION

The results presented in Section VI demonstrate that a terrain with moderate complexity such as the one shown in Fig. 3 already provides enough information to bound the uncertainty of the robot's location. The effectiveness is obviously limited by the actual terrain morphology in a real world situation, which would need to contain enough features such that all the DoF of the robot are constrained once the robot has touched them. Fig. 8 shows the evolution of the particles up to the first half of the terrain course for Experiment 2. As the robot walks through, the particle swarm

²A video showing these experiments is available online: <https://youtu.be/VfA8DAyr8t0>

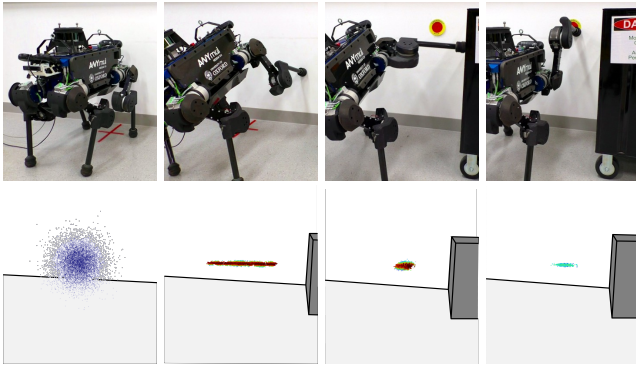


Fig. 7. 3-dimensional haptic probing experiment. The top row shows the robot performing the experiment while the bottom row shows the particle distribution and localization estimate. The particle set is colored by normalized weight according to the jet colormap (i.e., dark blue = lowest weight, dark red = highest weight). First, on the bottom left, the robot has an initial distribution with poses equally weighed. The robot makes a forward probe and then moves to the right. Now the particles are distributed as an ellipse with high uncertainty to the left and right of the robot. Then, the robot makes a probe to the right and touches an obstacle; the particle cloud collapses into a tight cluster. Since the robot is now localized, it is able to complete the task of pressing the button on the wall.

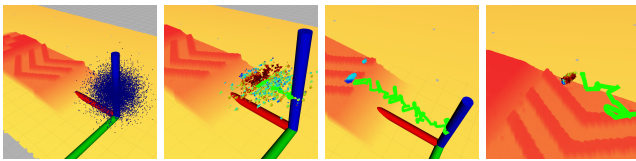


Fig. 8. Evolution of the particle set during a trial on the terrain course. The particle set is colored by normalized weight according to the jet colormap (i.e., dark blue = lowest weight, dark red = highest weight). The green line indicates the ground truth trajectory. A) At start, all the particles have the same weight and are normally distributed at the starting position. B) After a few steps on the ramp, the robot pose is well estimated on x and z directions, but there is uncertainty on y . C) When the robot approaches the chevron the particle set divides into two clusters, indicating two strong hypotheses as to the robot pose. D) After a few more steps on the chevron, the robot is fully localized and the particles are tightly clustered.

gets tighter, indicating good convergence to the most likely robot pose.

In the third subfigure, it can be noted how the probability distribution over the robot's pose follows a bimodal distribution which is visible as two distinct clusters of particles. This situation justifies the use of particle filters, as they are able to model non-Gaussian distributions which can arise from a particular terrain morphology. In this case, the bimodal distribution is caused by the two identical gaps in between the chevrons. In such situations, a weighted average of the particle swarm would lead to a poor approximation of the true pose distribution. Therefore, the particle analysis described in Section V-A is crucial to reject such an update.

VIII. CONCLUSION

We have presented a haptic localization algorithm for quadruped robots based on Sequential Monte Carlo methods. The algorithm can reduce the uncertainty of the estimated robot's trajectory walking over a non-degenerate terrain course, with an average pose estimation accuracy up to 10 cm. This is fundamental for repetitive autonomous tasks

in vision-denied conditions such as inspections in sewage systems. With routine inspection tasks in mind, we have also implemented a haptic exploration motion to localize against walls. In future work, we are planning to extend the localization algorithm to incorporate more features into the measurement model, such as estimated slope and friction, to further improve the localization performance as well as active planning for direct the probing actions.

IX. ACKNOWLEDGMENTS

This research has been conducted as part of the ANYbotics research community. It was part funded by the EU H2020 Project THING and a Royal Society University Research Fellowship (Fallon).

REFERENCES

- [1] 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.
- [2] 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 (RSS)*, Cambridge, Massachusetts, 2017.
- [3] P. Fankhauser, M. Bjelonic, C. Dario Bellicoso, T. Miki, and M. Hutter, "Robust rough-terrain locomotion with a quadrupedal robot," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2018, pp. 1–8.
- [4] M. Focchi, R. Orsolino, M. Camurri, V. Barasuol, C. Mastalli, D. G. Caldwell, and C. Semini, *Heuristic Planning for Rough Terrain Locomotion in Presence of External Disturbances and Variable Perception Quality*. Cham: Springer International Publishing, 2020, pp. 165–209.
- [5] M. F. Fallon, P. Marion, R. Deits, T. Whelan, M. Antone, J. McDonald, and R. Tedrake, "Continuous humanoid locomotion over uneven terrain using stereo fusion," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, 2015, pp. 881–888.
- [6] 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," *J. of Field Robotics*.
- [7] E. Krotkov, "Active perception for legged locomotion: every step is an experiment," in *Proc. 5th IEEE International Symposium on Intelligent Control*, 1990, pp. 227–232 vol.1.
- [8] J. Bednarek, M. Bednarek, L. Wellhausen, M. Hutter, and K. Walas, "What am I touching? Learning to classify terrain via haptic sensing," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2019, pp. 7187–7193.
- [9] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, *Particle Filters for Mobile Robot Localization*. New York, NY: Springer New York, 2001, pp. 401–428.
- [10] G. Grisetti, C. Stachniss, and W. Burgard, "Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, pp. 2432–2437.
- [11] G. Vezzani, U. Pattacini, G. Battistelli, L. Chisci, and L. Natale, "Memory unscented particle filter for 6-DOF tactile localization," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1139–1155, 2017.
- [12] R. Van Der Merwe, A. Doucet, N. De Freitas, and E. Wan, "The unscented particle filter," in *Proc. of the 13th Intl. Conf. on Neural Information Processing Systems (NIPS)*, 2000, pp. 563–569.
- [13] M. Chalon, J. Reinecke, and M. Pfanne, "Online in-hand object localization," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2013, pp. 2977–2984.
- [14] S. Chitta, P. Vernaza, R. Geykhman, and D. Lee, "Proprioceptive localization for a quadrupedal robot on known terrain," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2007, pp. 4582–4587.
- [15] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE Trans. Robotics*, vol. 33, no. 1, pp. 1–21, 2017.
- [16] J. Di Carlo, P. M. Wensing, B. Katz, G. Bledt, and S. Kim, "Dynamic locomotion in the MIT cheetah 3 through convex model-predictive control," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018, pp. 1–9.

©2020 IEEE. Reprinted, with permission, from Russell Buchanan, Marco Camurri and Maurice Fallon, “Haptic Sequential Monte Carlo Localization for Quadrupedal Locomotion in Vision-Denied Scenarios,” in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), October 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	Haptic Sequential Monte Carlo Localization for Quadrupedal Locomotion in Vision-Denied Scenarios
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Russell Buchanan, Marco Camurri, and Maurice Fallon. "Haptic Sequential Monte Carlo Localization for Quadrupedal Locomotion in Vision-Denied Scenarios". In: <i>IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)</i> . 2020.

Student Confirmation

Student Name:	Russell Buchanan		
Contribution to the Paper	• Wrote a majority of the code • Designed and performed experiments • Analyzed data • Wrote large portions of the paper		
Signature		Date	February 24, 2023

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 contributions listed above are an accurate representation of the work Russell contributed			
Signature		Date	February 24, 2023

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

4.2 Discussion

In this chapter, I introduce a method for legged robot localisation using information from contact with the ground. The contributions compared to prior work is the estimation of a full 6 DoF pose over a trajectory. This enabled significantly more experiments over longer distances including use in a closed loop with the navigation system and localisation in 3D for wall following.

In the implementation, I did not restrict the movement of the particles according to terrain feasibility. For example, the particles are able to move through walls. However, this could be another source of information for weighting the particles and improve the method further. Adding collision checking or reachability checking for each contact of each particle would be able to eliminate many impossible hypotheses but would also increase computational requirements.

The covariance of the particle propagation Σ_k comes from the EKF-based state estimator [7]. For both the 2.5D and 3D cases, the contact measurement error variance σ_z is manually selected to be 1 cm. This is to account for the elastic compression of the robot's rubber feet when making contact with the terrain as well as uncertainty in the prior map.

Finally, the need for morphologically varied terrain is an important limitation for this work. If the ground were mostly flat, there would not be enough information in the contacts to localise the robot. So while this work is an important contribution towards proprioceptive localisation, this method remains limited in its application. This limitation motivated my work in the next chapter to use semantic information for localisation even on flat ground.

5

Haptic Localisation with Terrain Semantics

In the method described in the previous chapter, the robot used only contact information to localise. While this was suitable for non-degenerate terrain, it would not work for long term walking on flat ground. Other recent works have shown how a robot can classify different terrain types while walking naturally (without the need for a dedicated probing motion) [5].

Two papers are provided in this chapter. The first seeks to exploit terrain semantic information and fuse it with the geometric information to enable haptic localisation over a wider range of terrains. This is done using the sensorised feet from [141] which have force/torque sensors as well as IMUs. The proprioceptive signals are then used to classify the terrain into one of 8 different terrains using a method based on [5]. The localisation algorithm is based on work from the previous chapter [16] and, like the previous work, requires a prior geometric and semantic map of the environment.

A major limitation of this method is the need to pre define a discrete number of terrain classes. This is not always feasible, either because the number of different terrain types changes or because the terrain cannot be easily differentiated into distinct classes – for example sand scattered over concrete. Therefore, in the

second paper of this chapter, a novel method is presented which uses unsupervised learning to generate compressed representations of the terrain which is stored in a map. This allows the robot to build a haptic map of the environment during an initial walk (encoding semantic information), and then use this information to re-localise during subsequent runs. This is analogous to “Visual Teach & Repeat” [50] methods but using haptic information.

5.1 Navigating by touch: haptic Monte Carlo localization via geometric sensing and terrain classification

The following article was published in the journal *Autonomous Robots* [16]. An accompanying video is available online at: <https://www.youtube.com/watch?v=Pr-u2oy6UZY>.



Navigating by touch: haptic Monte Carlo localization via geometric sensing and terrain classification

Russell Buchanan¹ · Jakub Bednarek² · Marco Camurri¹ · Michał R. Nowicki² · Krzysztof Walas² · Maurice Fallon¹

Received: 17 November 2020 / Accepted: 23 July 2021 / Published online: 27 August 2021
© The Author(s) 2021

Abstract

Legged robot navigation in extreme environments can hinder the use of cameras and lidar due to darkness, air obfuscation or sensor damage, whereas proprioceptive sensing will continue to work reliably. In this paper, we propose a purely proprioceptive localization algorithm which fuses information from both geometry and terrain type to localize a legged robot within a prior map. First, a terrain classifier computes the probability that a foot has stepped on a particular terrain class from sensed foot forces. Then, a Monte Carlo-based estimator fuses this terrain probability with the geometric information of the foot contact points. Results demonstrate this approach operating online and onboard an ANYmal B300 quadruped robot traversing several terrain courses with different geometries and terrain types over more than 1.2 km. The method keeps pose estimation error below 20 cm using a prior map with trained network and using sensing only from the feet, leg joints and IMU.

Keywords Legged robots · Proprioceptive localization · Terrain classification · Tactile sensing

1 Introduction

Recent advances in the maturity and robustness of quadrupedal robots have made them appealing for dull and dirty industrial operations, such as routine inspection and monitoring. Automating these operations in underground mines and sewers is particularly challenging due to darkness,

in-air dust, dirt, and water vapor, which can significantly impair a robot's vision system (Kolvenbach et al. 2020). Additionally, camera or laser sensor failure may leave only proprioceptive sensors (i.e., IMU and joint encoders) at the robot's disposal (Fig. 1).

Blind quadrupedal locomotion has achieved impressive levels of reactive robustness without requiring vision sensors (Focchi et al. 2020; Lee et al. 2020). However, without the ability to also *localize* proprioceptively, a robot would still be incapable of completing missions or inspections.

1.1 Contributions

In this paper, we significantly extend our prior work on proprioceptive localization (Buchanan et al. 2020) with the following contributions:

- A novel proprioceptive legged robot localization system that, in contrast to Buchanan et al. (2020), fuses both terrain geometry and semantic information. To the best of our knowledge, this is the first localization system using semantics when completely blind.
- A terrain classification method employing signal masking in the 1D convolutional modules, making it possible to process variable length signals from footsteps without

Russell Buchanan
russell@robots.ox.ac.uk

Jakub Bednarek
jakub.bednarek@put.poznan.pl

Marco Camurri
mcamurri@robots.ox.ac.uk

Michał R. Nowicki
michal.nowicki@put.poznan.pl

Krzysztof Walas
krzysztof.walas@put.poznan.pl

Maurice Fallon
mfallon@robots.ox.ac.uk

¹ Oxford Robotics Institute, University of Oxford, Oxford, UK

² Institute of Robotics and Machine Intelligence, Poznan University of Technology, Poznan, Poland

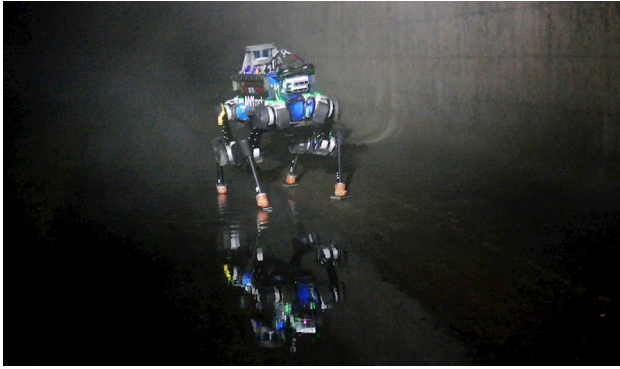


Fig. 1 An ANYmal robot (Hutter et al. 2016) in a sewer with two feet in a slippery, wet depression and two feet in a dry, elevated area. With prior information about terrain type and geometry, it is possible for the robot to localized in the world using only touch. This would be extremely useful in dark and foggy environments (Image courtesy of RSL/ETH and from Kolvenbach et al. (2020))

the need to truncate or pad them. This enables our method to work on uneven terrain and at differing walking speeds, unlike in Bednarek et al. (2019b).

- Extensive additional testing on an ANYmal B300 quadruped robot (Hutter et al. 2016) including different geometries and terrain types, for a total duration of 2.5 h and more than 1.2 km of traveled distance. The pose estimation error is kept down to 10 cm on geometrically feature rich terrain and on average below 20 cm on all terrain while exploiting terrain semantics. We also demonstrate convergence after only five steps from an unknown initial pose.

The remainder of this document is structured as follows: Sect. 2 summarizes relevant research in the fields of terrain classification, in-hand tactile localization and legged haptic localization; Sect. 3 defines the mathematical background of the legged haptic localization problem; Sect. 4 describes our proposed haptic localization algorithm; Sect. 5 describes the implementation details to deploy our algorithm on a quadruped platform; Sect. 6 presents the experimental results collected using the ANYmal robot; Sect. 7 provides an interpretation of the results and discusses the limitations of the approach; finally, Section 8 concludes with final remarks.

2 Related works

Pioneering work which exploits a robot's legs, not just for locomotion, but also to infer terrain information such as friction, stiffness and geometry has been presented by Krotkov (1990). This idea has recently been revisited to perform terrain vibration analysis or improve locomotion parameter selection via terrain classification. Since we are interested in

using terrain classification for localization, we cover the most relevant works applied to legged robots in Sect. 2.1. Works on proprioceptive localization in manipulation and legged robots are described in Sects. 2.2 and 2.3, respectively.

2.1 Tactile terrain classification

The first tactile terrain classification method for walking robots was presented by Hoepflinger et al. (2010) and concerned experiments with a single leg detached from a robot's body. In this work, force measurements and motor currents were used to successfully distinguish between four terrain types. These results paved the way for the application of terrain classification methods on complete legged robots.

More recently, Kolvenbach et al. (2019) used a single leg on a real, standing ANYmal robot to differentiate between four different types of soil. Two types of feet (point and planar) were used to collect force, torque and IMU measurements that were processed by an SVM classifier. The system showed that the tactile information could be used to differentiate between visually similar soils. However, their method required a pre-determined probing action which is impractical as it forces the robot to stop walking.

A terrain classification system that could operate during locomotion was presented by Wellhausen et al. (2019). At start, their legged robot was trained to assign a terrain negotiation cost based on force/torque sensors. Once operating, their system assigned a terrain negotiation cost from images based on previous feet-to-image correspondences and terrain classification based on proprioceptive sensors. The ability to predict the terrain negotiation based on images was then used to plan the robot's motion and avoid high cost terrains.

In complete darkness, which is our intended domain, vision-based sensors are of limited use for terrain classification, so we focus on purely proprioceptive sensing. In our previous work (Bednarek et al. 2019a), we showed how deep learning models can be used to increase the terrain classification accuracy. The system showed 98% classification accuracy from force-torque measurements against six terrain classes during a statically stable walk. However, this approach was limited to fixed-length input signals and thus could not generalize to aperiodic gaits, different speeds, or uneven terrain. In our work, we overcome these limitations with a novel masking mechanism in the convolutional layer, which allows us to process variable length signals.

More recently, Lee et al. (2020) showed an end-to-end approach to terrain classification for locomotion. Their deep learning controller was based on proprioceptive signals to adapt the gait to rough terrains. Even though tactile terrain classification was not explicitly performed, an internal representation of the terrain type was implicitly stored inside the network's memory. In our work, we opt for a modular

approach that explicitly returns a terrain class, which can then be used by the localization estimator.

2.2 Tactile localization in manipulation

Tactile localization involves the estimation of the 6-Degree of Freedom (DoF) pose of an object (of known shape) in the robot's base frame by means of kinematics of the robot's fingers and its tactile sensors. Since the object can have any shape, the probability distribution of its pose given tactile measurements can be multimodal. For this reason, tactile localization has typically been addressed using Sequential Monte Carlo (SMC) methods, a subfamily of which are called *particle filters* (Fox et al. 2001). SMC methods are sensitive to the dimension of the state space, which should be low enough to avoid combinatorial explosions or particle depletion. State-of-the-art methods aim to reduce this dimensionality and also to sample the state space in an efficient manner. For example, Koval et al. (2016) reduced the state space of the pose of an in-hand object to the observable contact manifold.

Chalon et al. (2013) proposed a particle filtering method for online in-hand object localization that tracks the pose of an object while it is being manipulated by a fixed base arm. The estimated pose was subsequently used to improve the performance of pick and place tasks. The particle weights were updated by penalizing finger/object co-penetration and the distance between the object and the fingertip in contact.

Manuelli and Tedrake (2016) approached a slightly different problem, using a particle filter to estimate external contacts on a rigid body robot using only force/torque sensors in the joints of the robot. Particles were distributed around the robot's body and particle weights were computed from how well the contact point explained an external torque.

Vezzani et al. (2017) proposed an algorithm for tactile localization using the Unscented Particle Filter (UPF) on a iCub robot with sensorized fingertips to localize four different objects in the robot's reference frame. The algorithm was recursive and could process data in real-time. The object and the robot's base were assumed to be static, allowing the pose to be estimated as a fixed parameter. For legged haptic localization, the assumption of both a static robot and terrain does not always hold and more general methods are required.

2.3 Haptic localization of legged robots

The first example of haptic localization applied to legged robots is from Chitta et al. (2007). In their work, they presented a proprioceptive localization algorithm based on a particle filter for LittleDog, a small electric quadruped. The robot was commanded to perform a statically stable gait over a known irregular terrain course, using a motion capture system to feed the controller. While walking, the algorithm approximated the probability distribution of the base state

with a set of particles. The state included three DoF: linear position on the xy -plane and yaw. Each particle was sampled from the uncertainty of the odometry, while the weight of a particle was determined by the L2 norm of the height error between the map and the feet contact location. The algorithm was run offline on eight logged trials of 50 s each.

Schwendner et al. (2014) demonstrated haptic localization on a wheeled robot with protruding spikes. The spikes detected contacts with the ground, which were compared to a prior 2.5D elevation map. Each wheel enabled multiple contact measurements, which they used to perform plane fitting against the prior map and improve localization over larger, flatter terrain. They also performed terrain classification, but with a camera, which we do not require in our proposed work. They demonstrated an average position error 39 cm in five experiments, of approximately 100 m each.

In Buchanan et al. (2020), we presented an SMC method that estimated the past trajectory (instead of the latest pose) at every step. Furthermore, the localization was performed for the full 6-DoF of the robot, instead of just the x , y and yaw dimensions as in Chitta et al. (2007) and Schwendner et al. (2014). The localization system was experimentally demonstrated online and onboard an ANYmal robot and used in a closed loop navigation system. When walking on flat areas, the localization uncertainty increased due to the lack of constraints on the xy -plane. We are therefore motivated to use terrain classification techniques described in Sect. 2.1 to incorporate more information into the SMC.

3 Problem statement

Let $\mathbf{x}_k \in SE(3)$ be a robot's pose at time k . We use the notation $\tilde{\mathbf{x}}$ to represent a pose estimate from an external estimator, and $\mathbf{x}^*$ to represent its most likely estimate from our SMC filter.

3.1 Quadruped state definition

We assume that for each timestep k , an estimate of the robot pose $\tilde{\mathbf{x}}_k$ and its covariance $\Sigma_k \in \mathbb{R}^{6 \times 6}$ are available from an inertial-legged odometric estimator, such as Bloesch et al. (2018), Fink and Semini (2020), Hartley et al. (2020). The uncertainties for the rotation manifold are maintained in the Lie tangent space, as in Forster et al. (2017). We also assume that the location of the robot's end effectors in the base frame $\mathcal{D}_k = (\mathbf{d}_{LF}, \mathbf{d}_{RF}, \mathbf{d}_{LH}, \mathbf{d}_{RH}) \in \mathbb{R}^{3 \times 4}$ are known from forward kinematics. The forces acting on each foot $\mathcal{F}_k = (\mathbf{f}_{LF}, \mathbf{f}_{RF}, \mathbf{f}_{LH}, \mathbf{f}_{RH}) \in \mathbb{R}^{3 \times 4}$ are measured by foot sensors (when available) or inferred from inverse dynamics. Finally, the binary contact states $\mathcal{K}_k = (\kappa_{LF}, \kappa_{RF}, \kappa_{LH}, \kappa_{RH}) \in \mathbb{B}^4$ are inferred from $\mathcal{F}_k$.

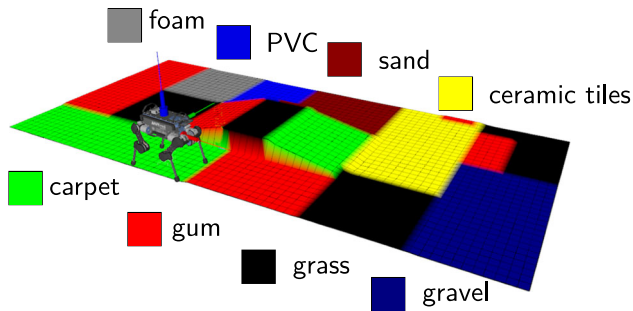


Fig. 2 Terrain map used in Experiment 3, showing terrain categories

For simplicity, we assume errors due to joint encoder noise or limb flexibility to be negligible. Therefore, the propagation of the uncertainty from the base to the end effectors is straightforward to compute. For brevity, the union of the aforementioned states (pose, contacts, and forces) at time k will be referred as the *quadruped state* $\mathcal{Q}_k = \{\tilde{\mathbf{x}}_k, \Sigma_k, \mathcal{D}_k, \mathcal{K}_k, \mathcal{F}_k\}$.

3.2 Prior map

Our approach can localize against 2.5D terrain elevation maps as well as full 3D maps. Terrain classification is meant to be carried out while the robot is walking, with no dedicated probing actions, therefore 2.5D maps are augmented with a terrain class category for each cell. This enables our method to overcome the degeneracy caused by featureless geometries (e.g., flat grounds, which are uninformative about the robot position on the xy -plane). Point clouds are used for 3D maps and only contain geometric information. To distinguish between the three types of map, we will refer to $\mathcal{M}$ when 2.5D only, $\mathcal{M}_3$ when 3D, and $\mathcal{M}_c$ when 2.5D augmented with class information, respectively. An example of an $\mathcal{M}_c$ map colorized by terrain class is shown in Fig. 2.

3.3 Estimation objective

Our goal is to use a sequence of quadruped states to estimate the most likely sequence of robot poses up to time k :

$$\mathcal{X}_k^* = [\mathbf{x}_0^*, \mathbf{x}_1^*, \dots, \mathbf{x}_k^*] \quad (1)$$

such that the likelihood of the contact points to be on the map is maximized. Additionally, we assume $\mathbf{x}_0^*$ is known.

4 Proposed method

To perform localization, we sample a predefined number of particles at regular intervals from the pose distribution provided by the odometry (as described in Sect. 4.2) and we

compute the likelihoods of the measurements by comparing each particle to the prior map, so as to update the weights of the particle estimator (see Sect. 4.3). For convenience, we give a brief summary of SMC theory in Sect. 4.1.

4.1 Sequential Monte Carlo localization

In SMC Localization, the objective is to approximate the posterior distribution of the state $\mathbf{x}_k$ given a history of measurements $\mathbf{z}_0, \dots, \mathbf{z}_k = \mathbf{z}_{0:k}$ as follows:

$$p(\mathbf{x}_k | \mathbf{z}_{0:k}) = \sum_i w_{k-1}^i p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{x}_{k-1}^i) \quad (2)$$

where w^i is the *importance weight* of the i -th particle; $p(\mathbf{z}_k | \mathbf{x}_k)$ is the measurement likelihood function and $p(\mathbf{x}_k | \mathbf{x}_{k-1}^i)$ is the motion model for the i -th particle state. Since $p(\mathbf{x}_k | \mathbf{z}_{0:k})$ is typically unknown, the state $\mathbf{x}_k$ is typically sampled from $p(\mathbf{x}_k | \mathbf{x}_{k-1}^i)$, yielding:

$$p(\mathbf{x}_k | \mathbf{z}_{0:k}) = \sum_i w_{k-1}^i p(\mathbf{z}_k | \mathbf{x}_{k-1}^i) \delta(\mathbf{x}_k - \mathbf{x}_{k-1}^i) \quad (3)$$

where $\delta(\cdot)$ is the Dirac delta function. Over repeated sampling steps, the particles will spread out of the whole state space with weights approaching zero. To avoid this “impoverishment” of the particles, an additional re-sampling from the mostly likely state is used. When the re-sampling is done at every step, the method is known as particle filtering. We use a different strategy that merges likelihoods from a history of states and so refer to our method with the more general term SMC.

4.2 Locomotion control and sampling strategy

Without a very robust reactive controller, blind locomotion requires conservative footstep placement, hence we opt for a statically stable gait, which guarantees stability at all times even when the motion is stopped mid flight phase. Since only one leg can be moved at the time, as soon as the swing leg touches the ground the robot enters into a four-support phase; at this time, the quadruped state estimate $\mathcal{Q}_k$ and the estimated terrain class $\tilde{c}$ for a given foot position are collected. Then, a new set of particles is sampled in a manner similar to Chitta et al. (2007):

$$p(\mathbf{x}_k | \mathbf{x}_{k-1}^i) = \mathcal{N}(\mathbf{x}_k, \Delta \tilde{\mathbf{x}}_k \mathbf{x}_{k-1}^i, \Sigma_k) \quad (4)$$

where $\Delta \tilde{\mathbf{x}}_k = \tilde{\mathbf{x}}_{k-1}^{-1} \tilde{\mathbf{x}}_k$ is the pose increment measured by the onboard state estimator at times $k-1$ and k .

At time k , the new particles $\mathbf{x}_k^i$ are thus sampled from a normal distribution centered at the pose estimated from the

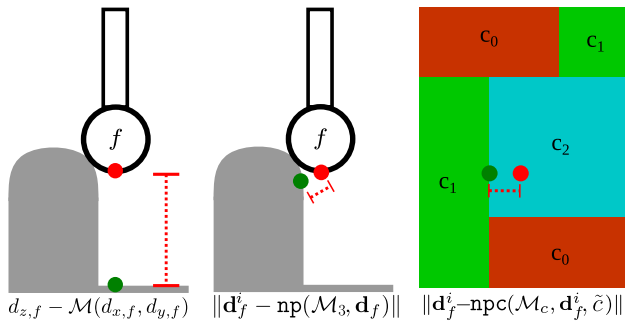


Fig. 3 Comparison between contact measurements for 2.5D (left), 3D (middle) and terrain class (right) map representations. The red dots indicate the contact point as sensed by the robot, while the green dots indicate the corresponding location returned by the map. The red line shows the magnitude of the measurement. In the terrain classification case, the view is from a top down perspective. The robot's foot sensed a contact in the c_2 region but a classification of c_1 was detected. The nearest c_1 point was returned by the map (Color figure online)

odometry with its corresponding covariance. Since roll and pitch angles are observable from inertial-legged estimators, their estimates have low uncertainty. In practice, this allows us to reduce the number of necessary particles along these two dimensions, while still retaining the ability to compensate for the errors of the state estimator due to IMU biases, which are observable by exteroceptive sensors only.

4.3 Measurement likelihood model for 2.5D data

The measurement likelihood is modeled as a univariate Gaussian centered at the local elevation of each cell, as done in Buchanan et al. (2020). The variance σ_z was manually set to 1 cm. Given a particle state $\mathbf{x}_k^i$, the estimated position of a contact in world coordinates for an individual foot f is defined as the concatenation of the estimated robot base pose and the location of the end effector, in base coordinates:

$$\mathbf{d}_f^i = (d_{x_f}^i, d_{y_f}^i, d_{z_f}^i) = \mathbf{x}_k^i \mathbf{d}_f \quad (5)$$

Thus, the measurements and their relative likelihood functions for the i -th particle and a specific foot f are (Fig. 3, left):

$$z_k = d_{z_f}^i - \mathcal{M}(d_{x_f}^i, d_{y_f}^i) \quad (6)$$

$$p(z_k | \mathbf{x}_k^i) = \mathcal{N}(z_k, 0, \sigma_z) \quad (7)$$

where $d_{z_f}^i$ is the vertical component of the estimated contact point location in world coordinates of foot f , according to the i -th particle; $\mathcal{M}(d_{x_f}^i, d_{y_f}^i)$ is the corresponding map elevation at the xy coordinates of $\mathbf{d}_f^i$.

4.4 Measurement likelihood model for 3D data

Our method can incorporate contact events from 3D probing. This is useful for areas where the floor does not provide enough information to localize. In this case, the robot can probe walls and 3D objects with its feet. To better model this situation, we represent the prior map $\mathcal{M}_3 \in \mathbb{R}^{3 \times N}$ by a 3D point cloud with N points. The likelihood of a particular contact point is computed using the Euclidean distance between the foot and the nearest point in the map. This likelihood is again modeled as a zero-mean Gaussian evaluated at the Euclidean distance between the estimated contact point $\mathbf{d}_f^i$ and its nearest neighbor on the map, with variance σ_z :

$$z_k = \|\mathbf{d}_f^i - \text{np}(\mathcal{M}_3, \mathbf{d}_f^i)\| \quad (8)$$

$$p(z_k | \mathbf{x}_k^i) = \mathcal{N}(z_k, 0, \sigma_z) \quad (9)$$

where $\text{np}(\mathcal{M}_3, \mathbf{d}_f^i)$ is the function that returns the nearest point of $\mathbf{d}_f^i$ on the map $\mathcal{M}_3$, computed from its k -d tree (Fig. 3, middle). In our tests, point clouds were sufficiently small to make the k -d tree search time negligible. For larger scale environments, more compact representations based on Truncated Sign Distance Fields (Oleynikova et al. 2017) or Octrees (Vespa et al. 2018) could be used to allow for faster search and less memory usage.

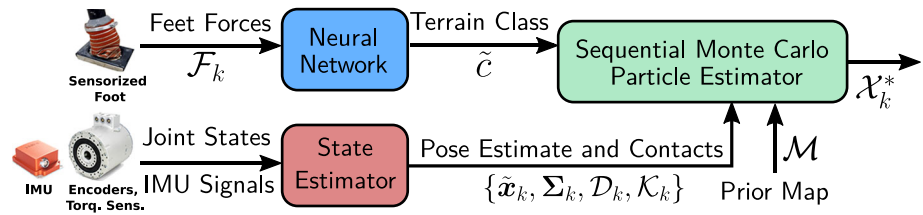
4.5 Terrain classification

Let $f: \mathcal{S} \mapsto \mathcal{C}$ be the haptic terrain classification function that associates an element from the signal domain $\mathcal{S}$ to an integer from the class counter-domain $\mathcal{C}$. The set $\mathcal{S} : \{s \in \mathbb{R}^{l(s) \times 6}\}$ includes sequences of variable length force and torque signals s (of length $l(s)$) generated by a foot touchdown event. The set $\mathcal{C}$ is defined as the integers from 0 to $n - 1$, where n is the total number of terrain classes that the robot is expected to be walking on (in our case, $n = 8$). The terrain classes were chosen to have a different haptic response, ensuring the existence of the function f .

Given the problem definition above, we introduce the following:

- a classification method $f': \mathcal{S} \mapsto \mathcal{C}$, which approximates the function f . As an implementation of f' we used a neural network;
- a dataset consisting of a list of pairs $d: [(s, c)]$, where $s \in \mathcal{S}, c \in \mathcal{C}$. Such a dataset was divided into two subsets, training and validation, with a ratio of 80:20;
- a training process formulated as an approximation of the function f using the function f' by the minimization of cross-entropy between the probability distributions generated by these functions.

Fig. 4 Information flow diagram of the overall system



4.6 Terrain class measurement likelihood model

The prior map is represented as a 2-dimensional grid whose cells are associated with a terrain class (an example is provided by Fig. 2). Measurements are represented as a piece-wise cost function. If the estimated terrain class $\tilde{c}$ for a given foot position $\mathbf{d}_f^i$ does match the class c , at that location in the prior map $c = \mathcal{M}_c(d_{xf}^i, d_{yf}^i)$ the probability is a constant value corresponding to the maximum value of a zero-mean univariate Gaussian with manually selected variance σ_c of 5 cm (which is the width of the foot).

If the estimated class does not match the expected class in the map, the same Gaussian distribution is used to model the likelihood, where the input z_k^c is the distance to the closest point in the map with the expected class. This function is shown as

$$p(z_k | \mathbf{x}_k^i) = \begin{cases} \frac{1}{\sigma_c \sqrt{2\pi}} & \tilde{c} = c \\ \mathcal{N}(z_k, 0, \sigma) & \tilde{c} \neq c, \end{cases} \quad (10)$$

where

$$z_k^c = \|\mathbf{d}_{xyf}^i - \text{npC}(\mathcal{M}_c, \mathbf{d}_{xyf}^i, \tilde{c})\|_2. \quad (11)$$

The function $\text{npC}(\mathcal{M}_c, \mathbf{d}_{xyf}^i, \tilde{c})$ returns the nearest 2D point with class $\tilde{c}$ to the 2D foot position $\mathbf{d}_{xyf}^i$ in the map $\mathcal{M}_c$. This last case is shown in Fig. 3, right.

We assume elevation and terrain class measurements (z_k, z_k^c) are conditionally independent. Therefore, their joint probability can be computed as

$$p(z_k, z_k^c | \mathbf{x}_k^i) = p(z_k | \mathbf{x}_k^i) p(z_k^c | \mathbf{x}_k^i). \quad (12)$$

5 Implementation

The block diagram of our system is shown in Fig. 4. The internal estimator on the ANYmal robot, TSIF (Bloesch et al. 2018), provides the odometry for the particle estimator, while the neural network estimates the class. This information is compared against the prior map to provide an estimate of the robot's trajectory, $\mathcal{X}_k^*$.

Pseudocode for the particle estimator (green block in Fig. 4) is listed in Algorithm 1. At time k , the estimates of the

```

 $\mathbf{x}_0^i \sim \mathcal{N}(\mathbf{x}_0, \tilde{\mathbf{x}}_0, \Sigma_0) \quad \forall i \in N$ 
foreach four-support phase  $k$  do
     $\Delta \tilde{\mathbf{x}}_k \leftarrow \tilde{\mathbf{x}}_{k-1}^{-1} \tilde{\mathbf{x}}_k$ 
    foreach particle  $i \in N$  do
         $\mathbf{x}_k^i \sim \mathcal{N}(\mathbf{x}_k, \Delta \tilde{\mathbf{x}}_k \mathbf{x}_{k-1}^i, \Sigma_k)$ 
         $w_k^i \leftarrow w_{k-1}^i$ 
        foreach foot  $f$  do
            if 3D then
                 $z_k \leftarrow \|\text{np}(\mathcal{M}_3, \mathbf{d}_f^i) - \mathbf{d}_f^i\|$ 
                 $w_k^i \leftarrow w_k^i p(z_k | \mathbf{x}_k^i)$ 
            else
                 $z_k \leftarrow d_{zf}^i - \mathcal{M}(d_{xf}^i, d_{yf}^i)$ 
                 $z_k^c \leftarrow \|\mathbf{d}_{xyf}^i - \text{npC}(\mathcal{M}_c, \mathbf{d}_{xyf}^i, \tilde{c})\|$ 
                 $w_k^i \leftarrow w_k^i p(z_k, z_k^c | \mathbf{x}_k^i)$ 
            end
        end
         $\mathbf{x}_k^* \leftarrow \text{WeightedMean}(\mathbf{x}_k^0 \dots \mathbf{x}_k^N, w_k^0, \dots, w_k^N)$ 
         $\mathcal{X}_k^* \leftarrow [\mathbf{x}_0^j, \dots, \mathbf{x}_k^j]$ 
        if  $\text{Var}(w_k^i) > \text{threshold}$  then
             $\text{resample}(\mathbf{x}_k^i)$ 
        end
    end
end

```

Algorithm 1: Haptic Sequential Monte Carlo Localization

terrain class $\tilde{c}$ and the robot pose $\tilde{\mathbf{x}}_k$ are collected. The pose estimate is used to compute the relative motion $\Delta \tilde{\mathbf{x}}_{k-1:k}$, propagate forward the state of each particle $\mathbf{x}_k^i$, and draw a sample from the distribution centered in $\Delta \tilde{\mathbf{x}}_k \mathbf{x}_{k-1}^i$ with covariance $\Sigma_k = (\sigma_{x,k}, \sigma_{y,k}, \sigma_{z,k})$.

The weight of a particle w^i is then updated by multiplying it by the likelihood that each foot is in contact with the map and the terrain class. In our implementation, we modify the likelihood functions from Eq. 7 as:

$$p(z_k | \mathbf{x}_k^i) = \min(\rho, \mathcal{N}(z_k, 0, \sigma_z)) \quad (13)$$

where ρ is a minimum weight threshold, so that outlier contact measurements do not immediately lead to degeneracy.

Re-sampling is triggered when the variance of the weights rises above a certain threshold. This is necessary to avoid dispersion of the particle set across the state space, with many particles with low weight. By triggering this process when the variance of the weights increases, the particles can first track the dominant modes of the underlying distribution.

5.1 Particle statistics

The pose estimate for the k -th iteration, $\mathbf{x}_k^*$, is computed from the weighted mean of all the particle poses. However, as we showed in our previous work, the particle distribution is often multi-modal. This motivated us to selectively update different dimensions of the robot pose. If the variance of the particle positions in the x and y axes are low (i.e., $\ll \sigma_{x,k}, \sigma_{y,k}$), we assume a well defined estimate and update the robot's full pose. However, if they are high, we update only the z component of the robot's location, which is always low as the robot keeps contact with the ground.

In practice, with the terrain classification we found the terrain course was sufficiently detailed to keep particle position variance in the xy -plane low, therefore z -only updates were rare. The threshold we used was a standard deviation of 10 cm, which corresponds to the typical uncertainty in our experiments.

To avoid particle degeneracy, importance sampling can be done in areas with higher likelihood. For example, if a grass terrain is detected, some particles could be injected in every grass terrain in the map. Further investigation on the benefits of importance sampling are left for future work.

5.2 Terrain classifier network

Our neural network architecture is shown in Fig. 5 and consists of three components: convolutional, recurrent, and predictive. Both the convolutional and recurrent components must process variable-length data. Therefore, for the convolution part, masking of the signal is required to prevent padded values from affecting the forward-pass result.

The first component of our network consists of two residual layers (ResLay). The ResLay used in our work is an adaptation of the one by He et al. (2016) with 2D convolutions replaced by 1D and support for masking. The recurrent component uses two bidirectional layers (Bidir) with two Gated Recurrent Units (GRU) (Cho et al. 2014) in each. The output of the recurrent component is an average of two resulting hidden states of the last Bidir. The final output of the neural network is produced by the predictive component, which takes the recurrent component's output, and using two fully connected (FC) layers produces a probability distribution from which the terrain class is inferred.

The number of convolutional layers and their sizes were chosen empirically to produce the best possible features before applying the compressed signal to the recursive part. To ensure better error propagation for the convolution part, we employed so-called skip-connections between layers thus reducing dimensionality.

The consecutive layers of the model are presented in Fig. 5. Each convolution block executes the following operations: batch normalization (Ioffe Szegedy 2015), dropout

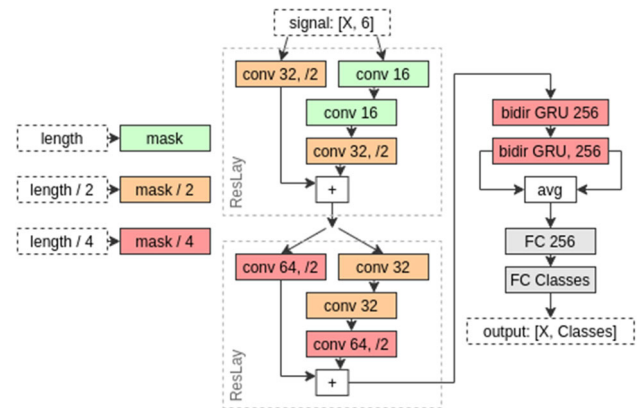


Fig. 5 Neural Network Structure used for Terrain Classification. The main blocks of neural architecture are convolutional (conv), recursive (GRU bidir), and fully connected (FC) blocks. The masking mechanism used in variable-length signal processing blocks takes appropriate masks, marked in green, orange, and red. These masks correspond to the individual signal lengths, taking into account the initial length and the reduction of size by convolution with stride equal to 2 (Color figure online)

(Srivastava et al. 2014), and Exponential Linear Unit (ELU) activation function (Clevert et al. 2016). All are modified to support masking. We used kernel size of 5 in each convolution layer. The output from each ResLay block is two times smaller as a result of stride in convolutions (marked as /2). Dropout is also used in every Bidir and FC (which also use batch normalization).

The model uses a dropout rate of 0.3 and a batch normalization momentum of 0.6. The proposed neural network consists of 1,374,920 trainable parameters.

5.2.1 Training

The learning process was carried out using the k -fold cross-validation methodology, with $k = 5$. The AdamW optimizer from Loshchilov and Hutter (2019) was used to minimize the loss function, with the following parameters:

- learning rate: $5e-4$ with exponential decay,
- weight decay: $1e-4$ with cosine decay.

Training was performed for 1000 epochs after which the training continued until no progress was made for 100 consecutive epochs. The size of each mini-batch was 256.

6 Experimental results

We extensively evaluated the performance of our algorithm in three different experiments, each one targeting a different type of localization. These are described in more detail in Sects. 6.2, 6.3, and 6.4.

6.1 Evaluation protocol

There are three different modalities in our algorithm: HL-G (Haptic Localization Geometric), which uses only geometric information, HL-C (Haptic Localization Class), using only class information, and HL-GC (Haptic Localization Geometric and Class), which uses both geometric and terrain class information. HL-G was tested in Experiments 1 and 2 using 2.5D and 3D prior maps, respectively. HL-G, HL-C and HL-GC were tested in Experiment 3 with 2.5D maps augmented with terrain class information.

6.1.1 Evaluation metrics

We quantitatively evaluated localization performance by computing the mean of the Absolute Translation Error (ATE) as described by Sturm et al. (2012):

$$\frac{1}{n} \sum_{i=1}^n \|\mathbf{T}_i^{-1} \hat{\mathbf{T}}_i\| \quad (14)$$

where $\mathbf{T}_i$ and $\hat{\mathbf{T}}_i$ are the robot's ground truth and estimated poses, respectively. In contrast to Sturm et al. (2012), we do not perform the alignment of trajectories, as ground truth and estimated poses are represented in the same coordinate system.

A qualitative evaluation was also performed for Experiment 1 and 2 by assessing the ability of the robot to reach its planned goals or end effector targets while using the localization online. This demonstrated the benefit of the localization when used in the loop with the onboard motion planner.

6.1.2 Ground truth and prior map

The ground truth trajectories were collected by motion capture systems at 100 Hz. The pose of both the robot and the terrain course reference frame were accurately measured with mm accuracy via reflective markers installed on them.

At start of the experiment, the relative position of the robot within the map was measured using ground truth and used for initialization only. Thereafter, the pose of the robot was estimated using the particle filter. To account for initial errors, particles at the start were sampled from a Gaussian centered at the initial robot pose with a covariance of 20 cm.

The prior maps were captured with survey grade laser scanners (Leica BLK-360 and SURPHASER 100HSX) which provided point clouds with sub-centimeter accuracy.

6.2 Experiment 1: 2.5D terrain course

In this experiment, the robot was commanded to navigate between four walking goals at the corners of a rectangle. One

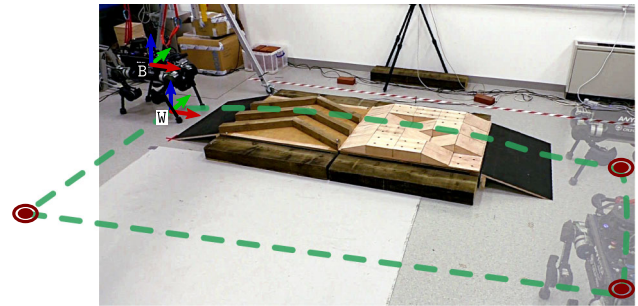


Fig. 6 Experiment 1: ANYmal haptic localization experiments. The robot traverses the terrain, turns 90 deg right and comes back to the initial position passing through the flat area. The goals given to the planner are marked by the dark red disks, while the planned route is a dashed green line (one goal is out of the camera field of view). The world frame W is fixed to the ground, while the base frame B is rigidly attached to the robot's chassis. The mutual pose between the robot and the terrain course is bootstrapped with the motion capture (Color figure online)

Table 1 Experiment 1: Estimation performance

Mean Absolute Translation Error (ATE)				
Trial	Dist. (m)	Time (s)	TSIF (m)	HL-G (m)
1	66.42	525	0.63	0.13
2	145.31	1097	2.57	0.40
3	55.67	557	0.52	0.19
4	68.71	604	0.65	0.32
5	172.65	1606	2.00	0.61

TSIF, Two-State Implicit Filter (Bloesch et al. 2018); HL-G, Haptic Localization with Geometry only

of the edges required crossing a 4.2 m terrain course composed of a 12° ascending ramp, a 13 cm high chevron pattern, an asymmetric composition of uneven square blocks and a 12° descending ramp (Fig. 6). After crossing the wooden course, the robot returned to the starting position across a portion of flat ground, so as to test the system behavior in feature-deprived conditions.

While blind reactive locomotion has been developed by a number of groups including (Di Carlo et al. 2018; Focchi et al. 2020), unfortunately our blind controller was not sufficiently reliable to cross this terrain course, so we resorted to use of the statically stable gait from Fankhauser et al. (2018) which used a depth camera to aid footstep planning. However, the pose estimation was performed without access to any camera information.

To demonstrate repeatability, we performed five trials of this experiment, for a total distance traveled of more than 0.5 km and 1 h 13 min duration. A summary of the results is presented in Table 1, where HL-G shows an overall improvement between 50 and 85% in the ATE compared to the onboard state estimator. ATE is 33 cm on average, which reduces to 10 cm when evaluating only the feature-rich portion of the experiments (i.e., the terrain course traversal).

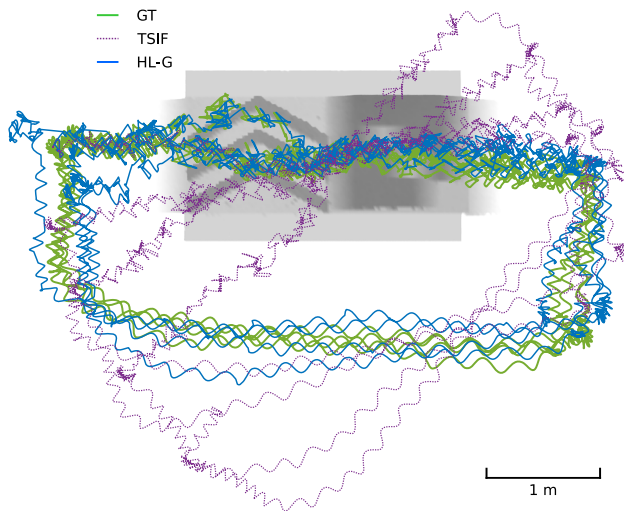


Fig. 7 Experiment 1, Trial 2: Top view of the estimated trajectories from TSIF (dashed purple), haptic localization (blue) and ground truth (green) (Color figure online)

For trials 1 and 2, the robot was manually operated to traverse the terrain course, completing two and four loops, respectively. In trials 3–5, the robot was commanded to follow the rectangular path autonomously. In these trials, the haptic localization algorithm was run online in closed-loop and effectively guided the robot towards the goals (Fig. 7). Using only the prior map and the contact events only, the robot stayed localized in all the runs and successfully tracked the planned goals. This can be seen in Fig. 7, where the estimated trajectory (in blue) diverges from ground truth on the xy -plane when the robot is walking on the flat ground. This is due to growing uncertainty from lack of geometric information, however the covariance reduces significantly and the cluster mean re-aligns with the ground truth when the robot reaches the terrain.

Figure 8 shows in detail the estimator performance for each of the three linear dimensions and yaw. Since position and yaw are unobservable, the drift on these states is unbounded. In particular, the error on the odometry filter [TSIF (Bloesch et al. 2018), purple dashed line] is dominated by upward drift (due to kinematics errors and impact nonlinearities, see third plot) and yaw drift (due to IMU gyro bias, see bottom plot). This drift is estimated and compensated for by the haptic localization (blue solid line), allowing accurate tracking of the ground truth (green solid line) in all dimensions. This can be noted particularly at the four peaks in the z -axis plot, where the estimated trajectory and ground truth overlap. These times coincide with the robot is at the top of the terrain course.

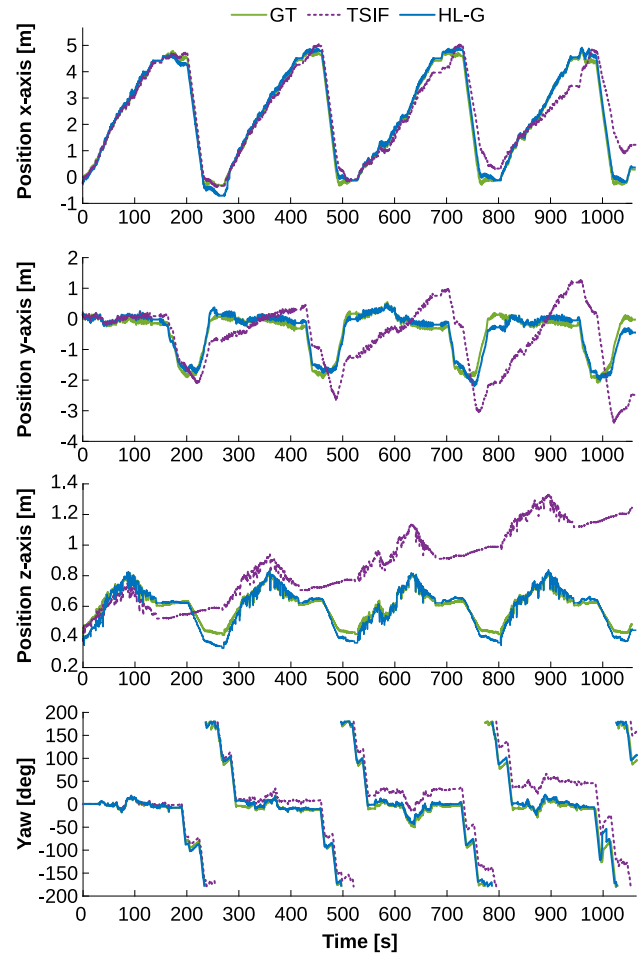


Fig. 8 Experiment 1, Trial 2: Comparison between the estimated position from TSIF (dashed purple) and haptic localization (blue) against ground truth (green). After 200 s, the estimation error in TSIF has drifted significantly upward and in yaw. In particular, the upward drift is noticeable in the third plot, where the values grow linearly. The drift is eliminated by the re-localization against the prior map (Color figure online)

6.3 Experiment 2: online haptic exploration on vertical surfaces

The second experiment involved a haptic wall following task with the robot starting in front of a wall but with an uncertain location. The particles were again initialized with 20 cm position covariance. To test the capability to recover from an initial error, we applied a 10 cm offset in both x and y from the robot's true position in the map. At start, the robot was commanded to walk 1 m to the right (negative y direction) and press a button on the wall, whose location in the prior map was known. To accomplish the task, the robot needed to “feel its way” by alternating probing motions with its right front foot and walking laterally to localize inside the room. The fixed number of probing motions was pre-scripted so with each step to the right, the robot probed both in front and

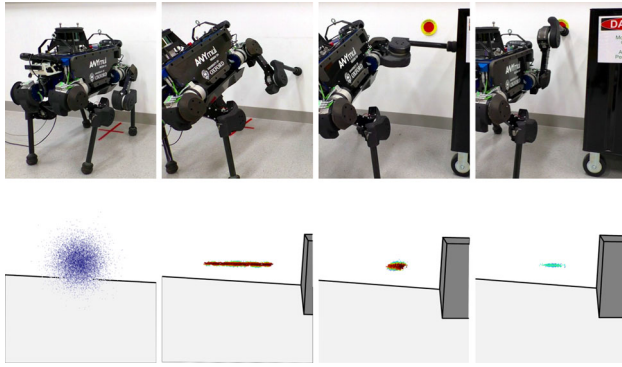


Fig. 9 Experiment 2: Haptic probing experiment in 3-dimensions. The top row shows the robot performing the experiment while the bottom row shows the particle distribution of position estimates. The particle set is colorized by normalized weight according to the jet colormap (i.e., dark blue = lowest weight, dark red = highest weight). First, on the bottom left, the robot has an initial distribution with poses equally weighed. The robot makes a forward probe and then moves to the right. Now the particles are distributed as an ellipse with high uncertainty to the left and right of the robot. Then, the robot makes a probe to the right and touches an obstacle; the particle cloud collapses into a tight cluster. Since the robot is now localized, it is able to complete the task of pressing the button on the wall (Color figure online)

to its right. The whole experiment was executed blindly with the static controller from Fankhauser et al. (2018).

As shown in Fig. 9 the robot was able to correct its localization and complete the task of touching the button. The initial probe to the front reduced uncertainty in the robot's x and z directions, which reduces the particles to an ellipsoidal elongated along y . As the robot moves, uncertainty in the x direction increases slightly. By touching the wall on the right, the robot re-localized in all three dimensions in much the same way as a human following a wall in the dark would. The re-localization allows the robot to press the button, demonstrating the generalization of our algorithm to 3D. This would enable a robot to localize by probing a known piece of mining machinery, allowing it to perform maintenance tasks. The final position error was: $[7.7, -3.7, -0.2]$ centimeters in the x , y and z directions.

6.4 Experiment 3: terrain classification

In the third experiment, we demonstrate the localization using terrain semantic information, which has been tested on a custom designed terrain course. Multiple $1 \times 1 \text{ m}^2$ tiles of different terrain materials were placed on a $3.5 \times 7 \text{ m}^2$ area. The course includes a 20 cm high platform with two ramps with different terrain materials, as shown in Fig. 10. The different terrain types used were: gum, carpet, PVC, foam, sand, artificial grass, ceramic and gravel.

For training, we gathered an additional dataset of the robot walking on the different patches consisting of 8773 steps with a quasi-static walk gait. Examples of data collection is shown



Fig. 10 Top: Examples of foot positioning when collecting data on different terrains under controlled laboratory conditions. Bottom: Experiment 3: ANYmal with sensorized flat feet standing on the multiple terrain type course. Close up of the foot is provided. An IMU and fore/torque sensor is located in the sole with coordinate frame shown

in Fig. 10 Top. During the data collection, the maximum base displacement and rotation were enforced to 0.21 m and 0.23 rad, respectively. These limits ensured a stable walk at all times. The dataset was split into 7018 training and 1775 testing samples. To minimize the impact of unbalanced data on the learning process (e.g., more steps on a specific class), the loss function was weighted based on the number of steps taken on each terrain. The network was trained as described in Sect. 5.2.1 and the mean and standard deviation of the accuracy was estimated from k-fold cross-validation to be 94% and 0.09, respectively.

The robot was equipped with sensorized feet which feature high quality 6-axis force/torque sensors (Valsecchi et al. 2020). These feet are necessary to provide the signals for terrain classification as in Bednarek et al. (2019b). The robot autonomously walked between pre-programmed waypoints placed over the entire course, including several passes over the ramp. Large sections of the trajectory were only on the flat terrain tiles, forcing the algorithm to rely mostly on terrain classification for localization. Unlike Experiment 1, the robot was able to walk completely blind and no exteroception was used for footstep planning. A statically stable gait was used such that one foot was in the air at a time.

To demonstrate repeatability, we have performed three trials of this type, for a total distance traveled of more than 0.7 km and 1 h 7 min duration. We compare results produced using HL-G (Geometry), HL-C (Terrain Class) and HL-GC (Geometry and Terrain Classification). As the majority of the terrain course is flat, there is not enough information for geometry only localization to be continuously accurate. Only when using terrain class information as well as geometry can the robot localize in all parts of the terrain course.

A summary of the experiments is presented in Table 2, where HL-GC shows an overall improvement between 14 and 56% in the ATE compared to HL-G. Using only the prior knowledge of the terrain geometry and class, the robot stayed localized in all the runs and bounded the linearly growing drift of the state estimator. This can be seen in Fig. 11, where the estimated trajectory (in red) is able to stay near the ground truth trajectory (green). In areas where there are large patches of the same material, such as the gravel (dark blue) and ceramic (yellow), there is not enough information to localize in the x - y -plane and the pose estimate drifts. When the robot crosses the boundary into a new terrain type the localization is able to correct.

Figure 13 shows in detail the estimator performance for each of the three linear dimensions and yaw. As in Experiment 1, the error on the odometry filter (TSIF, purple dashed line) of the robot is dominated by upward and yaw drift. This drift is estimated and compensated for by the haptic localization (red solid line), allowing accurate tracking of the ground truth (green solid line) in all dimensions.¹

7 Discussion

The results presented in Sects. 6.2 and 6.3 demonstrate that terrain with a moderate degree of geometric complexity (such as Fig. 6) already provides enough information to bound the uncertainty of the robot's location. The effectiveness of a purely geometric approach is obviously limited by the actual terrain morphology in a real world situation, which would need to contain enough features such that all the DoF of the robot are constrained once the robot has touched them.

In the case where there is not enough geometric information, we have shown in Sect. 6.4 that terrain semantics can be used to localize. With sufficiently diverse terrain types (as shown in Fig. 10), boundary crossings from one terrain to another provide enough information to correct for drift in the x - y -plane.

¹ A video showing all of these experiments is attached as supplementary material.

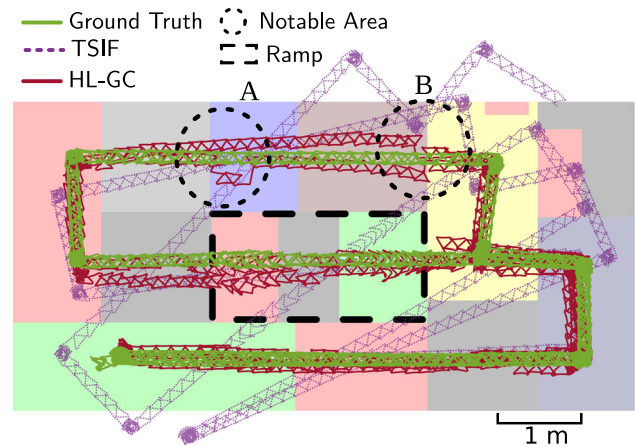


Fig. 11 Experiment 3, Trial 2: Top down view of the state estimator (dashed purple), HL-GC estimated trajectory (red) and ground truth (green). Trajectories are overlaid on terrain map with ramp shown in a dashed box. Two dotted circles show notable areas in the trajectory. **A** At a boundary crossing, the particle mean diverges from the ground truth. However, as the particle cloud nears the ramp, the geometric information gives higher likelihood to the particles in the center of the terrain. **B** Another boundary crossing which in two separated crossings triggers good localization updates (Color figure online)

7.1 Analysis of particle distribution on geometric terrain

Figure 14 Top shows the evolution of the particles up to the first half of the terrain course for Experiment 1, Trial 2. As the robot walks through, the particle cluster becomes concentrated, indicating good convergence to the most likely robot pose.

In the third subfigure, it can be noted how the probability distribution over the robot's pose follows a bimodal distribution, which is visible as two distinct clusters of particles. This situation justifies the use of particle filters, as they are able to model non-Gaussian distributions which can arise from a particular terrain morphology. In this case, the bimodal distribution is caused by the two identical gaps in between the chevrons. In such situations, a weighted average of the particle cluster would lead to a poor approximation of the true pose distribution. Therefore, the particle evolution illustrated in Sect. 5.1 is crucial to reject such an update.

Figure 14 Middle shows the particle distribution over flat ground. While not transitioning between terrain types, our method must rely on geometric information only and therefore is constrained in z but not x and y . In these cases we only update the drift estimate in the z direction which keeps the particle distribution near the ground but spread out.

7.2 Analysis of particle distribution on terrain class

Figure 12 shows data from Experiment 3, Trial 1. We compare results from HL-G, HL-GC and HL-C. We can see that

Table 2 Experiment 3: HL-G = Haptic Localization with only geometry; HL-C = Haptic Localization with only terrain class; HL-GC = Haptic Localization with both geometry and terrain class

Trial	Dist. (m)	Time (s)	Mean absolute translation error (ATE)			
			TSIF (m)	HL-G (m)	HL-C (m)	HL-GC (m)
1	191	1114	0.64	0.23	0.63	0.14
2	331	1850	1.28	0.25	0.73	0.11
3	193	1090	0.72	0.21	0.61	0.18

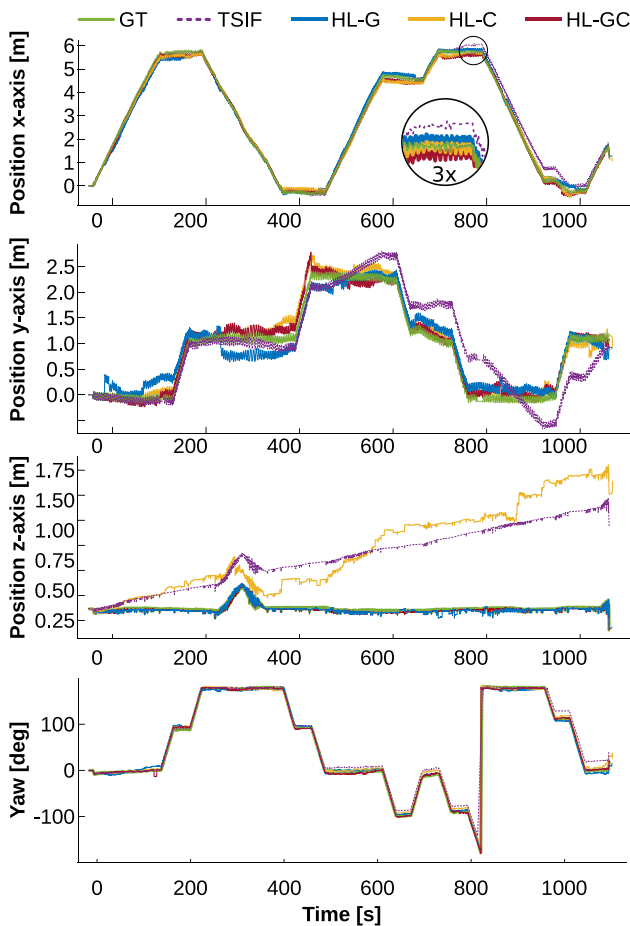


Fig. 12 Experiment 3, Trial 2: Here we show the difference in results when using only geometric information HL-G (blue), only terrain class information HL-C (yellow) and both HL-GC (red). We compare these to the state estimator (dashed purple) and ground truth (green) (Color figure online)

even with only class information, this method is able to keep pose estimation error bounded in the x - y plane (mean ATE for the class only trajectory was 0.63 m). In the third subplot from the top, the class only trajectory drifts upward in a similar way to TSIF in Fig. 13. This is because of the absence of any measurement in the z , hence our method relies on the proprioceptive state estimator.

Further analysis of the effect of terrain class on localization is shown in Fig. 14 middle. Here, we show the evolution of particles from HL-GC in Experiment 3, Trial 3. The particles, which initially are normally distributed around the

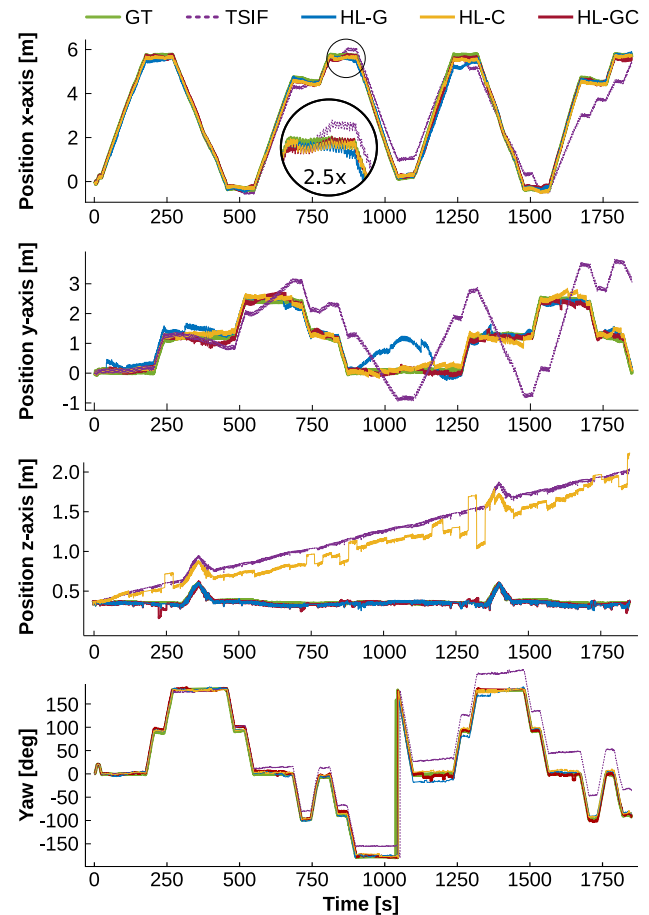


Fig. 13 Experiment 3, Trial 2: Here we show the difference in results when using only geometric information HL-G (blue), only terrain class information HL-C (yellow) and both HL-GC (red). We compare these to the state estimator (dashed purple) and ground truth (green) (Color figure online)

starting position, quickly converge along the z axis as the floor elevation information is used. Once the boundary transition from green to red occurs, the particles correct for drift in the x direction.

Finally, in Fig. 14 bottom we show the behavior of HL-GC when the particles are initialized evenly across the entire map. Within 5 footsteps the distribution has converged to the green section in the bottom left. This is because there are only two green terrains, and the bottom left is the only one on flat ground.

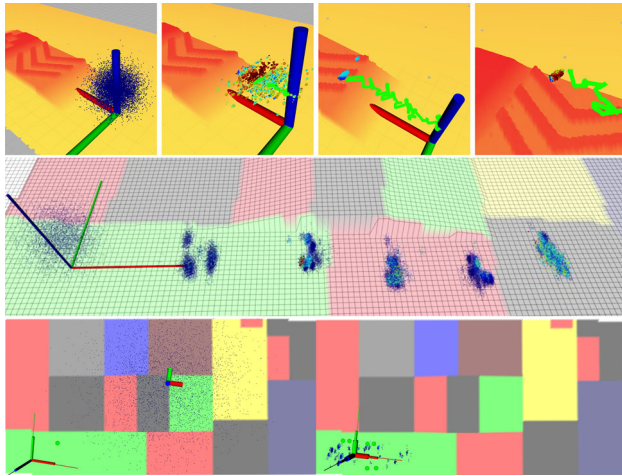


Fig. 14 Evolution of particle distributions during experiments. Particles are colored by normalized weight according to the jet colormap (i.e., dark blue = lowest weight, dark red = highest weight). **Top (Experiment 1):** The green line indicates the ground truth trajectory. **A** At start, all the particles have the same weight and are normally distributed at the starting position. **B** After a few steps on the ramp, the robot pose is well estimated on x and z directions, but there is uncertainty on y . **C** When the robot approaches the chevron the particle set divides into two clusters, indicating two strong hypotheses as to the robot pose. **D** After a few more steps on the chevron, the robot is fully localized and the particles are tightly clustered. **Middle (Experiment 3):** As the robot walks from left to right, the particle cloud makes two terrain class transitions. As the robot crosses the first transition, the cloud becomes more narrow in the x direction as error along this axis is corrected. **Bottom (Experiment 3):** Top-down perspective with the robot estimate initialized in the middle of the map and the initial particle distribution spread out over the whole map. Within 5 steps the particle distribution has re-sampled over the correct green terrain patch. The green circles are the classified footsteps

8 Conclusion

We have presented a haptic localization algorithm for quadrupedal robots based on Sequential Monte Carlo methods. The algorithm can fuse geometric information (in 2.5D or 3D) as well as terrain semantics to localize against a prior map. We have demonstrated that even using only geometric information, walking over a non-degenerate terrain course containing slopes and interested geometry can reduce localization error to 10 cm. Our method also works if the robot probes vertical surfaces, measuring its environment in full 3D. Finally, we have shown how in areas of even sparser geometric information, terrain semantics can be used to augment this geometry.

The proposed approach demonstrated an average of 20 cm position error over all areas of a terrain course with different terrain classes and geometries. The ability to localize purely proprioceptively is valuable for repetitive autonomous tasks in vision-denied conditions, such as inspections of sewage systems. This method could also serve as a backup localiza-

tion system in case of sensor failure—enabling a robot to complete its task and return to base.

8.1 Future work

The main limitation to our method is the need for sufficiently informative terrain. To mitigate this, we intend to incorporate other terrain properties such as slope or friction coefficient. Additionally, incorporating the network uncertainty as a prior on the terrain classification measurement would improve fusion with geometric information. Finally, we intend to generalize to more dynamic gaits and remove the costly dependency on sensorized feet.

Supplementary Information The online version contains supplementary material available at <https://doi.org/10.1007/s10514-021-10013-w>.

Acknowledgements This research has been conducted as part of the ANYbotics research community. It was part funded by the EU H2020 Project THING (Grant ID 780883) and a Royal Society University Research Fellowship (Fallon).

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Bednarek, J., Bednarek, M., Kicki, P., & Walas, K. (2019a). Robotic Touch: Classification of materials for manipulation and walking. In *IEEE international conference on soft robotics (RoboSoft)* (pp. 527–533). <https://doi.org/10.1109/ROBOSOFT.2019.8722819>
- Bednarek, J., Bednarek, M., Wellhausen, L., Hutter, M., & Walas, K. (2019b). What am I touching? Learning to classify terrain via haptic sensing. In *IEEE international conference on robotics and automation (ICRA)* (pp 7187–7193). <https://doi.org/10.1109/ICRA.2019.8794478>
- Bloesch, M., Burri, M., Sommer, H., Siegwart, R., & Hutter, M. (2018). The two-state implicit filter recursive estimation for mobile robots. *IEEE Robotics and Automation Letters*, 3(1), 573–580. <https://doi.org/10.1109/LRA.2017.2776340>.
- Buchanan, R., Camurri, M., & Fallon, M. (2020). Haptic sequential Monte Carlo localization for quadrupedal locomotion in vision-denied scenarios. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)*
- Chalon, M., Reinecke, J., & Pfanne, M. (2013). Online in-hand object localization. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, IEEE (pp. 2977–2984). <https://doi.org/10.1109/IROS.2013.6696778>

- Chitta, S., Vernaza, P., Geykhman, R., & Lee, D. (2007). Proprioceptive localization for a quadrupedal robot on known terrain. In *IEEE international conference on robotics and automation (ICRA)* (pp. 4582–4587). <https://doi.org/10.1109/ROBOT.2007.364185>
- Cho, K., van Merriënboer, B., Gülgehr, Ç., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. *Conference on empirical methods in natural language processing (EMNLP)*. <https://doi.org/10.3115/v1/D14-1179>.
- Clevert, D. A., Unterthiner, T., & Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (ELUs). In *International conference on learning representations (ICLR)*. <http://arxiv.org/abs/1511.07289>
- Di Carlo, J., Wensing, P. M., Katz, B., Bledt, G., & Kim, S. (2018). Dynamic locomotion in the MIT Cheetah 3 through convex model-predictive control. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 1–9). <https://doi.org/10.1109/IROS.2018.8594448>
- Fankhauser, P., Bjelonic, M., Dario Bellicoso, C., Miki, T., & Hutter, M. (2018). Robust rough-terrain locomotion with a quadrupedal robot. In *IEEE international conference on robotics and automation (ICRA)* (pp. 1–8). <https://doi.org/10.1109/ICRA.2018.8460731>
- Fink, G., & Semini, C. (2020). Proprioceptive sensor fusion for quadruped robot state estimation. In *IEEE international conference on intelligent robots and systems (IROS)*
- Focchi, M., Orsolino, R., Camurri, M., Barasuol, V., Mastalli, C., Caldwell, D. G., & Semini, C. (2020). *Heuristic planning for rough terrain locomotion in presence of external disturbances and variable perception quality* (pp. 165–209). Springer. https://doi.org/10.1007/978-3-030-22327-4_9
- Forster, C., Carlone, L., Dellaert, F., & Scaramuzza, D. (2017). On-manifold preintegration for real-time visual-inertial odometry. *IEEE Transactions on Robotics*, 33(1), 1–21. <https://doi.org/10.1109/TRO.2016.2597321>.
- Fox, D., Thrun, S., Burgard, W., & Dellaert, F. (2001). *Particle filters for mobile robot localization* (pp. 401–428). Springer. https://doi.org/10.1007/978-1-4757-3437-9_19
- Hartley, R., Ghaffari, M., Eustice, R. M., & Grizzle, J. W. (2020). Contact-aided invariant extended Kalman filtering for robot state estimation. *The International Journal of Robotics Research*, 39(4), 402–430. <https://doi.org/10.1177/0278364919894385>.
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *IEEE conference on computer vision and pattern recognition (CVPR)* (pp. 770–778). <http://arxiv.org/abs/1512.03385>
- Hoepflinger, M. H., Remy, C. D., Hutter, M., & Siegwart, R. (2010). Haptic terrain classification on natural terrains for legged robots. In *International conference on climbing and walking robots (CLAWAR)* (pp. 785–792). https://doi.org/10.1142/9789814329927_0097
- 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., & Hoepflinger, M. (2016). ANYmal—A Highly mobile and dynamic quadrupedal robot. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)*, IEEE (pp. 38–44). <https://doi.org/10.1109/IROS.2016.7758092>
- Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of machine learning research* (Vol. 37, pp. 448–456). Lille, France: PMLR.
- Kolvenbach, H., Bärtschi, C., Wellhausen, L., Grandia, R., & Hutter, M. (2019). Haptic inspection of planetary soils with legged robots. *IEEE Robotics and Automation Letters*, 4(2), 1626–1632. <https://doi.org/10.1109/LRA.2019.2896732>.
- Kolvenbach, H., Wisth, D., Buchanan, R., Valsecchi, G., Grandia, R., Fallon, M., & Hutter, M. (2020). Towards autonomous inspection of concrete deterioration in sewers with legged robots. *Journal of Field Robotics*, 37, 1314–1327. <https://doi.org/10.1002/rob.21964>.
- Koval, M. C., Pollard, N. S., & Srinivasa, S. S. (2016). *Manifold representations for state estimation in contact manipulation* (pp. 375–391). Springer. https://doi.org/10.1007/978-3-319-28872-7_22
- Krotkov, E. (1990). Active perception for legged locomotion: Every step is an experiment. In *Proceedings of the 5th IEEE international symposium on intelligent control 1990* (Vol. 1, pp. 227–232). <https://doi.org/10.1109/ISIC.1990.128462>
- Lee, J., Hwangbo, J., Wellhausen, L., Koltun, V., & Hutter, M. (2020). Learning quadrupedal locomotion over challenging terrain. *Science Robotics*. <https://doi.org/10.1126/scirobotics.abc5986>.
- Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. In *International conference on learning representations*. <http://arxiv.org/abs/1711.05101>
- Manuelli, L., & Tedrake, R. (2016). Localizing external contact using proprioceptive sensors: The contact particle filter. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 5062–5069).
- Oleynikova, H., Taylor, Z., Fehr, M., Siegwart, R., & 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)*.
- Schwendner, J., Joyeux, S., & Kirchner, F. (2014). Using embodied data for localization and mapping. *Journal of Field Robotics*, 31(2), 263–295. <https://doi.org/10.1002/rob.21489>.
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56), 1929–1958.
- Sturm, J., Engelhard, N., Endres, F., Burgard, W., & Cremers, D. (2012). A Benchmark for the evaluation of RGB-D SLAM systems. In *IEEE/RSJ international conference on intelligent robots and systems (IROS)* (pp. 573–580). <https://doi.org/10.1109/IROS.2012.6385773>
- Valsecchi, G., Grandia, R., & Hutter, M. (2020). Quadrupedal locomotion on uneven terrain with sensorized feet. *IEEE Robotics and Automation Letters*, 5(2), 1548–1555. <https://doi.org/10.1109/LRA.2020.2969160>.
- Vespa, E., Nikolov, N., Grimm, M., Nardi, L., Kelly, P. H. J., & Leutenegger, S. (2018). Efficient octree-based volumetric SLAM supporting signed-distance and occupancy mapping. *IEEE Robotics and Automation Letters*, 3(2), 1144–1151. <https://doi.org/10.1109/LRA.2018.2792537>.
- Vezzani, G., Pattacini, U., Battistelli, G., Chisci, L., & Natale, L. (2017). Memory unscented particle filter for 6-DOF tactile localization. *IEEE Transactions on Robotics*, 33(5), 1139–1155. <https://doi.org/10.1109/TRO.2017.2707092>.
- Wellhausen, L., Dosovitskiy, A., Ranftl, R., Walas, K., Cadena, C., & Hutter, M. (2019). Where should i walk? Predicting terrain properties from images via self-supervised learning. *IEEE Robotics and Automation Letters*, 4(2), 1509–1516. <https://doi.org/10.1109/LRA.2019.2895390>.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

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	Navigating by touch: haptic Monte Carlo localization via geometric sensing and terrain classification
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Russell Buchanan, Jakub Bednarek, Marco Camurri, Michalk R. Nowicki, Krzysztof Walas, and Maurice Fallon, "Navigating by touch: haptic Monte Carlo localization via geometric sensing and terrain classification," <i>Autonomous Robots</i> , 2021.

Student Confirmation

Student Name:	Russell Buchanan		
Contribution to the Paper	- Contributed to development of the idea - Wrote the entirety of the localisation code - Performed all data analysis and processing - Wrote large portions of the paper		
Signature		Date	February 24, 2023

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 contributions listed above are an accurate representation of the work Russell contributed			
Signature		Date	February 24, 2023

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

5.2 Unsupervised Learning of Terrain Representations for Haptic Monte Carlo Localization

The following article was presented at the IEEE International Conference on Robotics and Automation (ICRA) 2022 [98]. An accompanying video is available online at: <https://www.youtube.com/watch?v=T361Eacqo28>.

Unsupervised Learning of Terrain Representations for Haptic Monte Carlo Localization

Mikołaj Łysakowski¹, Michał R. Nowicki¹, Russell Buchanan², Marco Camurri²,
Maurice Fallon² and Krzysztof Walas¹

Abstract—Haptic sensing has recently been used effectively for legged robot localization in extreme scenarios where cameras and LiDAR might fail, such as dusty mines and foggy sewers. However, existing haptic sensing mainly relies on supervised classification, with training and evaluation executed over explicit terrain classes. Defining classes is a significant limitation to real-world applications, where prior labelling and handcrafted classes are often impractical. This paper proposes a novel haptic localization system based on a fully unsupervised terrain representation learned solely from the force/torque sensors located in the quadruped robot's feet. Instead of using the detected terrain class for localization, we propose an improved autoencoder architecture to generate a sparse map of encodings on the first run and to localize against this sparse map during subsequent runs. We compare our approach to a haptic localization system based on supervised terrain classification, showing that the unsupervised method has comparable or better performance than the supervised one for the same trajectories while clearly outperforming the proprioceptive odometry estimator available on the robot. Therefore, the proposed approach is well-suited for a routine maintenance application, increasing the platform's robustness.

I. INTRODUCTION

The increased maturity of legged robots has made them suitable for many industrial applications. Often, the ability to operate despite sensory failure is paramount, especially for long routine tasks in dangerous environments, such as dark and dusty mines (Fig. 1). In this work, we consider an autonomous quadruped robot performing an inspection task. The goal is to reliably localize with only proprioceptive sensors (IMU, joint states, tactile sensors) to complete the task or return to safety even when visual sensors are unavailable.

Tactile measurements can be used to determine the class of the terrain [1], [2], [3], which in turn can be used to localize a legged robot in a prior map [4]. However, existing approaches are limited as they require a dense and terrain-annotated map of the environment which often necessitates external, ground truth sensors. Moreover, the terrain classes are often differentiated by human perception rather than the haptic signals from the robot.

¹ Institute of Robotics and Machine Intelligence, Poznań University of Technology, Poznań, Poland
firstname.lastname@put.poznan.pl

² Oxford Robotics Institute, University of Oxford, UK
{russell,mcamurri,mfallon}@robots.ox.ac.uk

*This work is supported by the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 780883, THING and grant agreement No 780684, MEMMO. M. R. Nowicki is supported by the Foundation for Polish Science (FNP).



Fig. 1: An ANYmal B300 quadruped robot performing an inspection of conveyors 800m underground. The failure of vision or LiDAR-based localization can happen due to hot, humid, and dusty working conditions. Our system can increase the robustness of the robot in these repetitive tasks.

Our objective is a localization solution that does not require prior human input but instead builds a haptic map entirely from the robot's internal sensors and localizes against it. Therefore, the proposed unsupervised method learns a compact, internal representation of the traversed terrains. The terrain representation is provided by an autoencoder network, which encodes the force/torque signals from the robot's sensorized feet into a low-dimensional latent vector representing the terrain. The representation is firstly used to generate a sparse, latent map of the environment when vision or LiDAR-based localization is available. Then, when the robot blindly re-visits the same area, it uses the encoded tactile sensing to localize against the sparse map making it analogous to visual *teach & repeat* algorithms [5].

The contributions of our work can be summarized as:

- A Haptic AutoEncoder (HAE) network for unsupervised, compressed haptic terrain representations for walking robots.
- A novel Haptic Monte Carlo localization method based on the latent vector from the HAE and a sparse latent map built only from onboard sensors.
- Experimental evaluation on the ANYmal B300 quadruped robot walking on different terrain types and comparison with the state-of-the-art haptic localization solutions.

II. RELATED WORK

In this section, we review the relevant literature on haptic terrain classification (Section II-A), unsupervised encoding algorithms (Section II-B), and localization of legged robots (Section II-C).

A. Haptic terrain classification

Supervised haptic terrain classification for walking robots is a well-researched topic [1], [2], [3], [6] with methods achieving accuracy up to 90%. However, these methods require many supervised training samples while offering poor generalization to unseen terrain classes.

Ahmadi et al. [7] explored the idea of using a semi-supervised approach with fewer labelled terrains while achieving similar results to supervised approaches with their Recurrent Neural Network (RNN) based on Gated Recurrent Units (GRU). Similarly, Wellhausen et al. [8] have shown that the correspondence between the feet and vision can be learned to predict the traversal cost based solely on images. This allowed them to reduce the number of required training samples and plan the motion of the robot considering the cost of the terrain. In our most recent work [9], we evaluated the problem of supervised terrain recognition based on the practical aspects of grading the solutions based on their accuracy, model sizes, and inference times.

The classes for terrain classification are typically based on visual clues. As such, the classes might not reflect the physical properties of their corresponding terrains. In [10], the authors presented an end-to-end deep learning approach to gait adaptation based on proprioceptive sensing, avoiding class definition. In this approach, the internal representation of the haptic signals was used to make decisions about the walking gait. Inspired by [10], we are motivated to use the internal terrain representation from force/torque signals, but we avoid relying on end-to-end learning design to achieve explainable solutions.

B. Encoding tactile signals

To use the unsupervised force/torques signals for localization, we need a method to compute the similarity of encoded signals to achieve a scalable solution with generalization capabilities to unseen terrains. While this idea has not yet been explored for walking robots, Dynamic Time Warping (DTW) is a golden standard algorithm used to compare uncompressed, speed-varying signals in the time-domain [11]. In the typical application of finding the best match, the distances between the query signal and each signal from the database must be computed to generate a list of the most similar matches in a method called DTW K-Nearest Neighbors (DTW-KNN).

The most common deep learning technique used for encoding compressed latent representations is the AutoEncoder [12], which has been adopted for many applications [13]. The AutoEncoder consists of an encoder part that reduces the dimensionality of an input signal and a decoder part that reconstructs the original signal based on this reduced latent representation. The AutoEncoder is trained without

supervision, and the resulting latent representations can be compared to determine the similarity of the original signals. A performance comparison between DTW, AutoEncoder, and our proposed Haptic AutoEncoder will be provided in Section V-B.

C. Haptic Localization of Legged Robots

The earliest work on haptic localization applied to legged robots was presented by Chitta et al. [14]. They used a particle filtering-based algorithm to proprioceptively localize a quadrupedal robot on a known terrain. While walking, their method approximated the probability distribution of the pose of the robot's base with a set of particles that were sampled from the uncertainty of the odometry. The particle weights were determined by the L_2 norm of the height error between the prior map of the terrain and the feet' contact location. They demonstrated experiments with a Boston Dynamics' LittleDog robot using a Vicon system to control the robot.

Schwendner et al. [15] demonstrated a similar method on a wheeled robot with protruding spikes. In addition to geometric information, their method also fused information from explicit terrain classifications using a camera, which we do not require in our proposed work.

In [16], we presented a geometric only method based on Sequential Monte Carlo (SMC) Localization that estimated the robot's 6-Degrees of Freedom (DoF) pose trajectory of the robot. Building on this work, in [4] we presented a proprioceptive, supervised terrain classification method to improve localization on terrains lacking distinct geometric features. However, this method required explicit classification of different terrains. In this work, we tackle the localization problem with a new method of unsupervised terrain representation (Section IV-A).

III. PROBLEM STATEMENT

Given a quadruped robot with 12 active DoF, we wish to blindly localize the robot's base against a prior map, given a source of odometry and the force/torque (F/T) signals measured at the feet following the ground contact. Additionally, it is assumed that the prior map was built using a low-drift or drift-free pose estimate of the robot, for example, using LiDAR at a time when there is minimal dust. For our experiments, we use the ANYmal B300 [17] equipped with a version of the sensorized feet from [18] (without IMUs) and the odometry estimator by Bloesch et al. [19]. Because the estimate depends on the shape of the map, the probability distribution of the estimated robot pose may not be Gaussian. Thus, we will use an SMC method to find the optimal pose estimate of the robot in the map.

IV. PROPOSED METHOD

An overview of our SMC system is presented in Fig. 2. Firstly, the encoder determines a terrain representation for each footstep (Section IV-A). The computed encoding of each footstep is associated with an accurate source of robot poses (e.g., from a SLAM system) to build a sparse haptic latent map (Section IV-B).

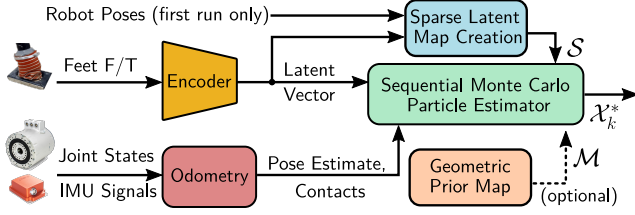


Fig. 2: A system overview of our proposed Haptic Monte Carlo localization. The odometry from the robot is fused with the latent vector from the encoder using a sparse latent map as prior. Optionally, a geometric map can also be used for increased accuracy.

Once the map is built, the localization is performed with a predefined number of particles at regular intervals, as described in Section IV-C. At each new step, the SMC estimator updates the state of its particles using the odometry from the robot, the current tactile signals, and optionally the geometry of the terrain.

A. Learned Terrain Representation

As the sparse map of terrain representation contains steps from previous walks over the same route, the map's size naturally grows linearly with the length of the trajectory. Assuming the systematic inspection over 1 km with steps taken every 10 cm, the approximated latent map size for unsupervised, uncompressed signals can exceed 200 MB. Therefore, we seek an encoding that will generate a reduced tactile description of size b called a latent vector. This would reduce the size of the map while preserving the properties of the original signals while also increasing generalization against terrains that are similar to those previously explored.

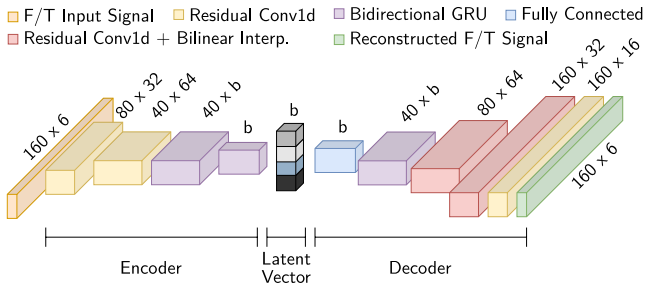


Fig. 3: The scheme of the Haptic AutoEncoder (HAE) architecture was used to generate a b -dimensional latent vector. The main blocks of the network are convolutional layers (CNN), bidirectional GRU layers (Bidir GRU), and fully connected layer (FC).

For the encoding, we propose an architecture called Haptic AutoEncoder (HAE), consisting of three types of blocks: convolutional, recurrent, and fully connected, as shown in Fig. 3. This combination of blocks, inspired by the supervised terrain classification network from [4], extracts high-level feature representation of the time-series signals and can process variable-length data. The HAE model uses two residual layers with 1D convolutions in the CNN module and

two bidirectional layers with two GRU cells in the recurrent component of an encoder. The convolutional part of the encoder is responsible for extracting the features of signals and reducing the spatial dimension of a sequence using a stride of 2. Next, in contrast to a simple convolutional AutoEncoder, the bidirectional GRU layers are used to capture the global and local dependencies between the features. Finally, the RNN block produces the b -dimensional latent vector.

The decoder's recurrent block consists of a fully connected (FC) layer and a bidirectional GRU layer, which is followed by three residual 1D convolution blocks. The FC layer takes as an input the latent vector and produces an initial state for the bidirectional GRU layer. The RNN block is used to construct a sequence of features, which is then upsampled via bilinear interpolation in a signal's length dimension and reduced in terms of a number of features with convolutions. The model uses dropout layers with a rate of 0.4 in every block and a batch normalization with a momentum equal to 0.6 in residual blocks.

The training process of the HAE lasted for approximately 640 epochs (around 4.5 hours on NVIDIA GeForce GTX 1660 Ti) until convergence was observed. We used the AdamW optimizer [20] to minimize Mean Squared Error (MSE) loss with exponential learning rate decay and cosine decay of weight decay. The size of the mini-batch was 64.

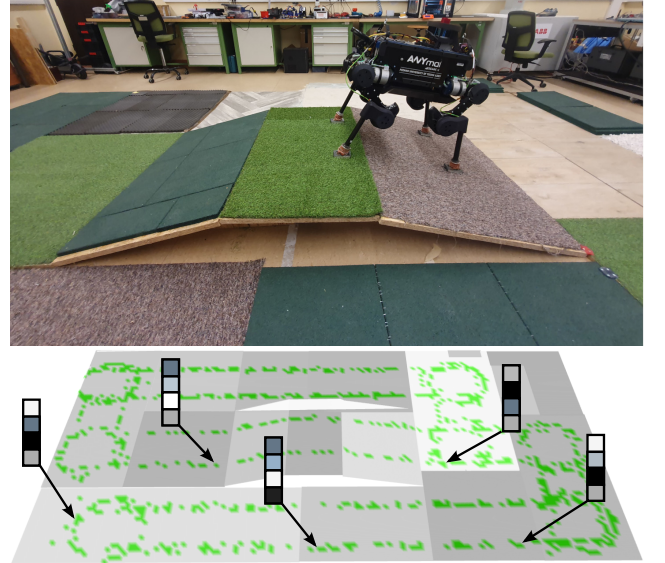


Fig. 4: The ANYmal robot walking over the prepared terrain course (top) and the visualization of the sparse latent map generated from several trials recorded prior to the localization phase (bottom). A green color marks each cell of the latent map containing the compressed tactile signal.

B. Sparse Haptic Map Generation

The sparse map was generated from steps recorded prior to the localization, as shown in Fig. 4. Whenever a contact between the ground and a foot was detected by ANYmal's state estimator, 160 samples of 3-axis force and 3-axis torque signals were collected from a foot. Then, the signals were

normalized and processed by the encoding algorithm resulting in latent vectors. This resulted in a database mapping the global position of each footstep (available from a source of ground truth or a SLAM system) and the corresponding latent vector. To store the encoded signals from this phase, we defined a data structure called a latent map, which is based on the implementation of a grid map [21]. The sparse latent map $\mathcal{S}$ stores the information about the encoded representations of tactile signals and their positions in the Euclidean space for a set of cells visited in prior runs of a robot.

C. Sequential Monte Carlo Localization

The core localization algorithm used in this work was initially proposed in, [4] and it is here summarized for convenience, with some details omitted due to space constraints.

Let $\mathbf{x}_k \in SE(3)$ be a robot's pose at time k . We use the notation $\hat{\mathbf{x}}$ to represent a pose estimate from an external estimator, and $\mathbf{x}^*$ to represent its most likely estimate from our SMC filter. Our objective is to approximate the posterior distribution of the state $\mathbf{x}_k$ given a history of measurements $\mathbf{z}_0, \dots, \mathbf{z}_k = \mathbf{z}_{0:k}$ as follows:

$$p(\mathbf{x}_k | \mathbf{z}_{0:k}) = \sum_i w_{k-1}^i p(\mathbf{z}_k | \mathbf{x}_k) p(\mathbf{x}_k | \mathbf{x}_{k-1}^i) \quad (1)$$

where w^i is the *importance weight* of the i -th particle; $p(\mathbf{z}_k | \mathbf{x}_k)$ is the measurement likelihood function and $p(\mathbf{x}_k | \mathbf{x}_{k-1}^i)$ is the motion model for the i -th particle state. Since $p(\mathbf{x}_k | \mathbf{z}_{0:k})$ is typically unknown, the state $\mathbf{x}_k$ is normally sampled from $p(\mathbf{x}_k | \mathbf{x}_{k-1}^i)$, yielding:

$$p(\mathbf{x}_k | \mathbf{z}_{0:k}) = \sum_i w_{k-1}^i p(\mathbf{z}_k | \mathbf{x}_{k-1}^i) \delta(\mathbf{x}_k - \mathbf{x}_{k-1}^i) \quad (2)$$

where $\delta(\cdot)$ is the Dirac delta function.

Blind locomotion requires conservative footstep placement. Hence we opt for a statically stable gait, which guarantees stability at all times, even when the motion is stopped mid-flight phase. Since only one leg can be moved at the time, as soon as the swing leg touches the ground, the robot enters into a four-support phase; at this time a new set of particles are sampled:

$$p(\mathbf{x}_k | \mathbf{x}_{k-1}^i) = \mathcal{N}(\mathbf{x}_k, \Delta \tilde{\mathbf{x}}_k \mathbf{x}_{k-1}^i, \Sigma_k) \quad (3)$$

where $\Delta \tilde{\mathbf{x}}_k = \tilde{\mathbf{x}}_{k-1}^{-1} \tilde{\mathbf{x}}_k$ is the pose increment measured by the onboard state estimator at times $k-1$ and k . At time k , the new particles $\mathbf{x}_k^i$ are thus sampled from a normal distribution centered at the pose estimated from the odometry with its corresponding covariance.

1) *Unsupervised Haptic Measurement Model:* The F/T signals recorded from the sensorized feet are processed during the localization phase to generate localization likelihoods, as shown in Fig. 5. Given a particle state $\mathbf{x}_k^i$, the estimated contact position in the world coordinates for each foot f is defined as the concatenation of the estimated robot base pose and the location of the end effector $\mathbf{d}_f$, in base coordinates:

$$\mathbf{d}_f^i = (d_{xf}^i, d_{yf}^i, d_{zf}^i) = \mathbf{x}_k^i \mathbf{d}_f. \quad (4)$$

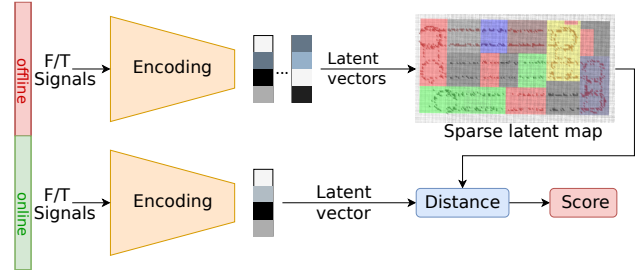


Fig. 5: Comparing encoded measurement with a prior map of latent vectors using a selected distance measure. Note that the latent map is sparse (based on previous steps), a colored segmentation is simply for visualization.

The corresponding measured F/T signal for the foot f is encoded into a latent vector $\mathbf{v} \in \mathbb{R}^b$, where b is the dimension of the latent vector.

During the localization stage, we determine the latent vector $\mathbf{m} = \mathcal{S}(d_{xf}^i, d_{yf}^i)$ from a sparse latent map $\mathcal{S}$ that is the closest to the current foot placement $\mathbf{d}_f^i$ and located in the latent map at global coordinates (s_{xf}^i, s_{yf}^i) . We compute the distance d_u between the measured latent vector $\mathbf{v}$ and the latent vector $\mathbf{m}$ from the latent map. We evaluate two definitions of distance d_u as the Euclidean L_2 norm f_{L_2} and the cosine similarity $f_{\cos}$ (see Section V-B):

$$d_u = f_{L_2}(\mathbf{v}, \mathbf{m}) = \sqrt{\sum_{i=1}^n (v_i - m_i)^2}, \quad (5)$$

$$d_u = f_{\cos}(\mathbf{v}, \mathbf{m}) = \frac{\mathbf{v} \cdot \mathbf{m}}{\|\mathbf{v}\| \|\mathbf{m}\|}, \quad (6)$$

where v_i and m_i indicate the i -th components of the encoded latent representations.

To model the likelihood, we use the univariate Gaussian distribution centered at the cell matched in the sparse latent map:

$$p(\mathbf{z}_k | \mathbf{x}_k^i) = \begin{cases} \frac{1}{\sigma_u \sigma_d 2\pi} & d_{2D} > d_t \\ \mathcal{N}(d_u, 0, \sigma_u) \mathcal{N}(d_{2D}, 0, \sigma_d) & d_{2D} \leq d_t, \end{cases} \quad (7)$$

where $d_{2D} = \sqrt{(d_{xf}^i - s_{xf}^i)^2 + (d_{yf}^i - s_{yf}^i)^2}$ is the Euclidean distance between the foot's position (d_{xf}^i, d_{yf}^i) and the matched, closest step in the sparse latent map (s_{xf}^i, s_{yf}^i) . We reject matches if the Euclidean distance exceeds the threshold of $d_t = 25$ cm, which was set to match the similar Euclidean threshold for terrain classification in [4]. The sigma value of $\sigma_u = 0.4$ determines the standard deviation of the F/T signal, while the sigma value of $\sigma_d = 0.4$ captures the decreased credibility of the sparse map measurement when the foot was placed further away from the known location in the map. The sigma values σ_u and σ_d were chosen empirically.

2) *Geometric Measurement Model:* If a dense geometric prior map is available, we integrate this information into the localization using the geometric measurement likelihood

from [4]. This is modeled as an univariate Gaussian centered at the local elevation of each cell. Given the estimated contact position of the foot f in the world coordinates $\mathbf{d}_f^i$ by a particle state $\mathbf{x}_k^i$, the measurements and their relative likelihood functions for the i -th particle are:

$$z_k = d_{zf}^i - \mathcal{M}(d_{xf}^i, d_{yf}^i) \quad (8)$$

$$p(z_k | \mathbf{x}_k^i) = \mathcal{N}(z_k, 0, \sigma_z) \quad (9)$$

where: d_{zf}^i is the vertical component of the estimated contact point location in world coordinates of foot f , according to the i -th particle; $\mathcal{M}(d_{xf}^i, d_{yf}^i)$ is the corresponding dense map elevation at the xy coordinates of $\mathbf{d}_f^i$ and $\sigma_z = 0.01$ is a measurement variance.

V. EXPERIMENTAL RESULTS

Experimental setup is described first in Section V-A. We evaluate the Haptic AutoEncoder (HAE) and compare it against other state-of-the-art algorithms in Section V-B. Then, we compare the performance of our unsupervised SMC localization system against a supervised approach both with (Section V-C) and without (Section V-D) a prior geometric map.

A. Dataset and Ground Truth

The SMC algorithm was evaluated over a $3.5 \text{ m} \times 7 \text{ m}$ area containing flat terrain with two ramps covered by eight different terrain types (rubber, carpet, PVC, foam, sand, artificial grass, ceramic, and gravel). The ground truth robot movement was recorded from the external motion capture system (OptiTrack), providing poses at 100 Hz. The reference map for the geometric component was captured with a sub-centimeter accurate survey-grade 3D laser scanner (SURPHASER 100HSX).

This setup was used to capture the terrain classification PUTANY dataset [9]. We chose the PUTANY dataset as several supervised terrain classification algorithms based on the F/T signals were already evaluated on this dataset [9]. In our case, we measured the performance of the encoded terrain representations using accuracy (Acc) averaged over 1148 terrain samples in the testing part of PUTANY dataset (Sec. V-B).

The performance of the haptic SMC localization (Sec. V-C) was measured over three trials accumulating to a total length of 715 meters, 4054 seconds, and 6658 steps. The sparse map for these trials was created based on previous, independent runs. This haptic localization dataset is the same as in our previous work [4] and thus, the localization results can be directly compared.

B. Unsupervised Compressed Terrain Representation

The unsupervised compression of the tactile signals aims to find a reduced latent representation that still contains the original information. In our case, we measured the ability to store the original information on the terrain classification problem, arguing that greater accuracy corresponds to better discrimination and thus is preferred for the haptic SMC localization. The terrain classification accuracy of different

encoding algorithms against the length of the latent vector b is presented in Fig. 6. The five best unsupervised results are compared to the state-of-the-art supervised terrain classification algorithms in Table I. Classification with unsupervised methods was performed using the nearest neighbour algorithm, performed on the similarity measure between latent representations. Methods with varying length of latent vector b and distance definitions d_u are abbreviated to pattern method- b - d_u , e.g., HAE-512- L_2 .



Fig. 6: The terrain recognition accuracy for the unsupervised, compressed tactile signals depends on the latent vector's length (representation) used to capture the original properties of the F/T signals

While the supervised methods outperform the unsupervised approaches, the latter still provide reliable accuracy (above 70%). The highest accuracy among the unsupervised methods (74.2%) was achieved with HAE-512- L_2 , which compresses the original signal to half its length. The second-best result was achieved with DTW-KNN [9], which directly compares the signals in the time domain, making it unsuitable for larger terrain maps.

TABLE I: Accuracy comparison of supervised and unsupervised methods.

Method	Supervised	b	d_u	Acc [%]
CNN-RNN [4]	Yes	-	-	93.0
HAPTR [9]	Yes	-	-	91.7
HAE-512- L_2	No	512	L_2	74.2
DTW-KNN [11], [9]	No	960	DTW	74.0
HAE-512-cos	No	512	cos	74.0
HAE-256- L_2	No	256	L_2	73.6
HAE-256-cos	No	256	cos	73.3

This analysis of the different configurations reveals that the L_2 measure outperforms the cosine similarity $\cos$ for AE or HAE regardless of the chosen length of the latent vector. A shorter length of the compressed signal degrades accuracy, but this effect is reduced for AE and HAE as the value nears 256. From Fig. 6, it is evident that HAE better captures the original signal characteristic than AE as it can achieve similar performance with shorter latent vectors. Among the evaluated solutions, the encoding with HAE-256- L_2 was chosen as a reasonable trade-off: it uses approximately four times less memory than DTW-KNN and two times less than HAE-512- L_2 while achieving an accuracy that is only 0.6% worse.

Absolute Pose Error (APE) μ [m]				
Trial	TSIF [19], [4]	HL-G [4]	HL-GC [4]	HL-GU
1	0.64	0.23	0.14	0.15
2	1.28	0.25	0.11	0.18
3	0.72	0.21	0.18	0.13

TABLE II: Comparison of haptic localization performance when prior geometry of the map is used. The usage of unsupervised F/T signals (HL-GU) results in comparable gains when supervised terrain classification is used (HL-G).

C. Haptic localization with geometric and latent map

We directly compared our method to the approaches presented in [4]: Haptic Localization (HL) performed with the geometry of the environment utilizing a dense, prior geometric map (HL-G), with the supervised terrain classifications (HL-C), or with both of these sources of information (HL-GC). We also show the non-localized results from the robot's proprioceptive odometry TSIF [19].

We propose haptic localization with the unsupervised, compressed F/T signals working solely on the sparse map (HL-U) and combining this approach with the geometry of the terrain requiring the dense, geometric map (HL-GU). The localization accuracy obtained over three trials used in [4] is presented in Tab. II.

These results confirm that when using unsupervised tactile signals with geometric information (HL-GU), performance is improved over methods using only geometry (HL-G). Compared to the supervised terrain classification method with geometry (HL-GC), the usage of unsupervised F/T signals (HL-GU) yields comparable results achieving the best result for trial 3. This demonstrates that the unsupervised F/T signals can capture the differences between signals obtained for different classes.

D. Haptic localization with latent map (without geometry)

In some cases, it may not be feasible to create a prior, dense geometric map. We use the same trails as in the previous section to replicate this case, but only the sparse latent map is available for localization purposes. Results obtained without F/T signals (TSIF), with supervised terrain classification (HL-C) and unsupervised, compressed tactile signals (HL-U) are presented in Tab. III.

Absolute Pose Error (APE) μ [m]			
Trial	TSIF [19], [4]	HL-C [4]	HL-U
1	0.64	0.63	0.47
2	1.28	0.73	0.57
3	0.72	0.61	0.5

TABLE III: The comparison between the localization accuracy obtained with a haptic SMC localization utilizing the internal ANYmal sensors (TSIF) combined with terrain classification (HL-C) or proposed, unsupervised terrain representation (HL-U).

The method based on the unsupervised haptic signals (HL-U) outperforms the method utilizing the supervised

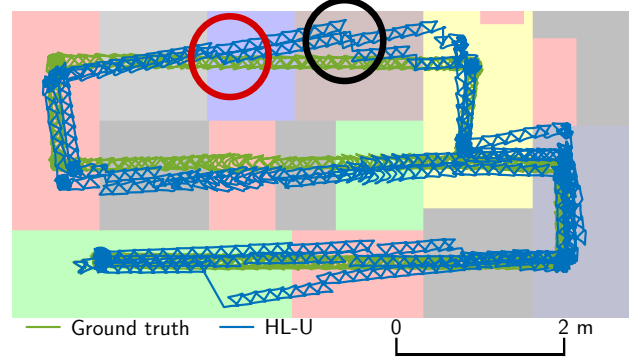


Fig. 7: The trajectory comparison between the ground truth (GT) and haptic SMC with unsupervised, compressed tactile signals (HL-U) for trial 2. The corrections for robot localization for HL-U can be observed once a robot crosses to a different terrain class (red circle) and during localization over the same terrain type (black circle). Different colors mark the patches of different terrain classes.

terrain classification (HL-C) in all considered trials while also not requiring a prior, dense map of the environment. The obtained localization accuracy is also clearly better when compared to the default state estimator TSIF. The use of sparse representation allows for better scalability, e.g., for underground tunnels.

The trajectory obtained with the HL-U for trial 2 is presented in Fig. 7. The visual analysis reveals that the method is effective at decreasing the accumulated drift of the internal odometry of the walking robot. This is true both when crossing boundaries between terrain classes (red circle) and also observed while walking over long stretches of the same terrain class (black circle). The latter benefit is a direct result of using unsupervised terrain representations instead of explicit terrain classes.

VI. CONCLUSIONS

In this work, we propose to utilize unsupervised, compressed force/torque signals directly in a haptic localization framework for routine inspection tasks. Our experiments indicate that the best trade-off between compressed signal length and performance of signal encoding can be obtained with the proposed Haptic AutoEncoder. We employ this method with haptic SMC localization over 3 trials containing external ground truth from a motion capture system. Localization with unsupervised tactile signals performs similarly or even outperforms the method using supervised terrain classification, both with and without geometric information. Additionally, there is no need to specify terrain classes using our method. However, the localization improvement depends on the diversity of the terrain. In future work, we plan to investigate IMU information in the sensorized feet to learn the latent representations. We also intend to deploy our approach outdoors while also extending the geometric component to work with a sparse map created in *teach & repeat* scenarios. Finally, we plan to expand our work using different gaits, i.e., dynamic ones.

REFERENCES

- [1] M. H. Hoepflinger, C. D. Remy, M. Hutter, and R. Siegwart, "Haptic Terrain Classification on Natural Terrains for Legged Robots," in *International Conference on Climbing and Walking Robots (CLAWAR)*, 2010, pp. 785–792.
- [2] J. Bednarek, M. Bednarek, P. Kicki, and K. Walas, "Robotic Touch: Classification of Materials for Manipulation and Walking," in *IEEE International Conference on Soft Robotics (RoboSoft)*, 2019, pp. 527–533.
- [3] H. Kolvenbach, C. Bärtschi, L. Wellhausen, R. Grandia, and M. Hutter, "Haptic Inspection of Planetary Soils With Legged Robots," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1626–1632, 2019.
- [4] R. Buchanan, J. Bednarek, M. Camurri, M. R. Nowicki, K. Walas, and M. Fallon, "Navigating by touch: haptic Monte Carlo localization via geometric sensing and terrain classification," *Autonomous Robots*, 2021.
- [5] P. Furgale and T. D. Barfoot, "Visual teach and repeat for long-range rover autonomy," *Journal of Field Robotics*, vol. 27, no. 5, pp. 534–560, 2010.
- [6] X. A. Wu, T. M. Huh, R. Mukherjee, and M. Cutkosky, "Integrated Ground Reaction Force Sensing and Terrain Classification for Small Legged Robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 1125–1132, 2016.
- [7] A. Ahmadi, T. Nygaard, N. Kottege, D. Howard, and N. Hudson, "Semi-Supervised Gated Recurrent Neural Networks for Robotic Terrain Classification," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1848–1855, 2021.
- [8] L. Wellhausen, A. Dosovitskiy, R. Ranftl, K. Walas, C. Cadena, and M. Hutter, "Where Should I Walk? Predicting Terrain Properties From Images Via Self-Supervised Learning," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1509–1516, 2019.
- [9] M. Bednarek, M. Łysakowski, M. Bednarek, M. R. Nowicki, and K. Walas, "Fast haptic terrain classification for legged robots using transformer," in *2021 European Conference on Mobile Robots (ECMR)*, 2021.
- [10] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, 2020.
- [11] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, p. 565–592, May 2015.
- [12] M. Kramer, "Nonlinear principal component analysis using autoassociative neural networks," *Aiche Journal*, vol. 37, pp. 233–243, 1991.
- [13] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," arXiv:2003.05991, 2020.
- [14] S. Chitta, P. Vernaza, R. Geykhman, and D. Lee, "Proprioceptive localization for a quadrupedal robot on known terrain," in *IEEE International Conference on Robotics and Automation (ICRA)*, April 2007, pp. 4582–4587.
- [15] J. Schwendner, S. Joyeux, and F. Kirchner, "Using embodied data for localization and mapping," *Journal of Field Robotics*, vol. 31, no. 2, pp. 263–295, 2014.
- [16] R. Buchanan, M. Camurri, and M. Fallon, "Haptic Sequential Monte Carlo Localization for Quadrupedal Locomotion in Vision-Denied Scenarios," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, October 2020.
- [17] M. Hutter *et al.*, "Anymal - a highly mobile and dynamic quadrupedal robot," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 38–44.
- [18] G. Valsecchi, R. Grandia, and M. Hutter, "Quadrupedal locomotion on uneven terrain with sensorized feet," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1548–1555, 2020.
- [19] 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, Jan 2018.
- [20] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, 2019.
- [21] 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)*, A. Koubaa, Ed. Springer, 2016, ch. 5.

©2022 IEEE. Reprinted, with permission, from M. Łysakowski, M. R. Nowicki, R. Buchanan, M. Camurri, M. Fallon and K. Walas, “Unsupervised Learning of Terrain Representations for Haptic Monte Carlo Localization,” in IEEE International Conference on Robotics and Automation (ICRA) 2021.

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	Unsupervised Learning of Terrain Representations for Haptic Monte Carlo Localization
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Mikolaj Łysakowski, Michal R. Nowicki, Russell Buchanan, Marco Camurri, Maurice Fallon and Krzysztof Walas, "Unsupervised Learning of Terrain Representations for Haptic Monte Carlo Localization", <i>IEEE International Conference on Robotics and Automation (ICRA)</i> 2021.

Student Confirmation

Student Name:	Russell Buchanan		
Contribution to the Paper	• Contributed to development of the idea • Minor code changes to existing work to facilitate co-authors • Contributed to writing of the paper		
Signature		Date	February 24, 2023

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 contributions listed above are an accurate representation of the work Russell contributed			
Signature		Date	February 24, 2023

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

5.3 Discussion

In this chapter I present two papers which expand on the previous haptic localisation by adding semantic information. In the first paper in Section 5.1, this is from an explicit terrain classifier while the work in Section 5.2 uses an unsupervised encoder-decoder network to learn a compressed representation of the terrain.

For the first paper in Section 5.1, I used a fixed uncertainty σ_c for all terrain classification measurements in the particle filter. It was manually set to 5 cm which was the width of the robot foot. This implicitly assumes that the network classification is correct and any error is positional. A more complete approach would have been to integrate over the full probability distribution of different possible terrain classifications. However, to compute the error, the function $npc(\cdot)$ in (11) is used to find the nearest point on the 2D map which corresponds to the queried class. This involves a search in the map which can be computationally expensive and would not have been feasible to do for every potential terrain class. In practice, the classifier had 94% accuracy which meant it was not necessary to compute probability for every class. Additionally, because with every step, four measurements were available (one for each foot), the system was not very sensitive to misclassification although no systematic analysis was done. In future work it would be interesting to quantitatively analyse the impact of misclassification or errors in the terrain map.

In the second paper in Section 5.2, the map of latent terrain representations was much sparser. In that work, we showed how the robot could localise in a sparse map so long as it stayed close to the path used when building the map. However, we did not systematically test the impact of different map densities. Future work should investigate to what extent the density of the map affects localisation accuracy, both for terrain semantic representations and for geometric information.

Additionally, in Section 5.2, we used an encoder-decoder network to learn a compressed representation of terrain from proprioceptive signals only. Related work has shown how vision can help classify terrain properties [151, 165]. However, these works are underpinned by the assumption that similar looking terrain will have similar physical properties (e.g. produce a similar sound when driven over by a

wheeled robot). In contrast, our approach does not make this assumption and therefore in certain situations our method would likely perform better. For example, there may be two identical carpets except that one is blue and the other red. Our method would not be confused by the different appearance.

Finally, both works relied the assumption that the geometric and terrain type measurements were conditionally independent. However, this may not be the case as certain terrain types, such as long grass, may affect the geometric measurements. Estimation from correlated data remains a challenge in robotics. One potential solution is to train a network to predict the marginal covariances for the two measurements. A similar approach has been used to adapt uncertainty in IMU measurements [10] and point cloud measurements [33]. This could be trained in a self supervised way using a ground truth localization of a robot walking on different terrain examples.

The two papers presented in this chapter pose several interesting possibilities and challenges for future work. In particular, concepts of SLAM or “Visual Teach & Repeat” (VT&R) [50] could be used with the methods in this chapter for more advanced proprioceptive localisation. In Section 8.1.1, I go into more detail on ideas for future work.

Part III

Inertial Learning

6

Learning Inertial Odometry

In the previous part, I developed ideas of haptic localisation, building a map of tactile information while the robot walks and using it to re-localise. However, this system still requires a separate odometry system which is needed to propagate the particles. In the previous works, the source of odometry was the kinematic-inertial EKF from [7] which is affected by large amounts of drift, in particular on soft terrain or when slipping. This means my previous methods could still fail if the terrain were more difficult to traverse.

To improve on the odometry, I investigated recent work applying deep learning to INS. It has been shown that neural networks can be trained to learn a motion model of pedestrians and to then infer displacement from only IMU data [24, 96].

This chapter presents two papers which apply learned inertial odometry to different problems. In the first paper in Section 6.1, I use this to reduce odometry drift for legged robots in different challenging situations like walking on soft ground. I use a neural network based on [96] with a ResNet architecture [62] to learn a motion model of a legged robot as it walks over various terrains. The displacement measurements are then fused with kinematic and visual information and used in both an EKF and factor graph estimator.

In the second paper in Section 6.2, which is a draft to be submitted, I apply this technique as an IMU-only odometry method to estimate the pose of an ultrasound

scanner. Using ultrasound for 3D reconstruction of bones has enormous potential as a useful medical tool because it has lower cost and is radiation free as compared to MRI and X-ray imaging technologies [110]. However, current methods for 3D reconstruction of bones rely heavily on external infrastructure to enable state estimation e.g. motion tracking systems [75].

Inertial learning of motion models needs a repetitive, periodic motion i.e. a pedestrian or legged robot walking. The back and forth motion of scanning a patient with a handheld device is therefore well suited for inertial learning. In this work, I compare VIO and a learning-based IMU-only method for the purpose of estimating the pose of a handheld ultrasound scanner. I also introduce a full pipeline for 3D reconstruction of bone from ultrasound images.

6.1 Learning Inertial Odometry for Dynamic Legged Robot State Estimation

The following article was presented at Conference on Robot Learning (CoRL) 2021 [15]. An accompanying video is available online at: <https://www.youtube.com/watch?v=IMaMq0Uew9I>.

Learning Inertial Odometry for Dynamic Legged Robot State Estimation

Russell Buchanan

Oxford Robotics Institute
University of Oxford
russell@robots.ox.ac.uk

Marco Camurri

Oxford Robotics Institute
University of Oxford
mcamurri@robots.ox.ac.uk

Frank Dellaert

Institute for Robotics
and Intelligent Machines
Georgia Institute of Technology
dellaert@cc.gatech.edu

Maurice Fallon

Oxford Robotics Institute
University of Oxford
mfallon@robots.ox.ac.uk

Abstract: This paper introduces a novel proprioceptive state estimator for legged robots based on a learned displacement measurement from IMU data. Recent research in pedestrian tracking has shown that motion can be inferred from inertial data using convolutional neural networks. A learned inertial displacement measurement can improve state estimation in challenging scenarios where leg odometry is unreliable, such as slipping and compressible terrains. Our work learns to estimate a displacement measurement from IMU data which is then fused with traditional leg odometry. Our approach greatly reduces the drift of proprioceptive state estimation, which is critical for legged robots deployed in vision and lidar denied environments such as foggy sewers or dusty mines. We compared results from an EKF and an incremental fixed-lag factor graph estimator using data from several real robot experiments crossing challenging terrains. Our results show a reduction of relative pose error by 37 % in challenging scenarios when compared to a traditional kinematic-inertial estimator without learned measurement. We also demonstrate a 22 % reduction in error when used with vision systems in visually degraded environments such as an underground mine.

Keywords: Legged Robots, Inertial Navigation, Deep Neural Networks

1 Introduction

Recent advances in legged robot locomotion have motivated the use of quadruped robots in dull and dirty industrial applications. To perform these tasks autonomously, accurate state estimation with limited drift over long periods of time is fundamental. However, operating in challenging environments such as mines and sewers, dust particles and vapor can severely degrade camera and lidar systems [1] (Fig. 1). Proprioceptive methods can work in these environments, as they fuse information only from joint kinematics and Inertial Measurement Unit (IMU) data [2, 3, 4]. However, slip events or deformation of the robot’s foot or terrain introduce non-Gaussian error that accumulates over time and leads to unbounded drift.

Separately, recent advances in pedestrian tracking using only IMU data with machine learning have shown promising results [5, 6]. The key insight of these works is that it is possible to learn a *motion prior* from inertial data which could be robust to changing biases. We propose a deep neural network capable of learning a motion prior on the locomotion of a legged robot for any terrain and in challenging conditions such as slipping. The network takes a buffer of IMU data and outputs a displacement measurement and covariance which is fused with kinematic information. We demonstrate the application of this measurement in both filtering and factor-graph optimization contexts to track the motion of an ANYbotics ANYmal quadrupedal robot and compare results to other learning-based inertial odometry systems [5].



Figure 1: **Left:** The ANYmal C100 quadrupedal robot exploring a limestone mine. **Right:** Images from the on-board camera which demonstrate the vision challenges in such an environment.

The contributions can be listed as follows: 1) First kinematics-inertial estimator based on learned IMU displacement measurements; 2) Integration with both a filter-based estimator and a factor graph, merging concepts from machine learning and model based methods; 3) Extensive testing on a quadruped robot, including application to field experiments of the robot navigating a mine.

2 Related Work

In this section, we briefly summarize relevant work on proprioceptive state estimation for legged robots (Section 2.1) and related research in learned inertial state estimation (Section 2.2).

2.1 Kinematic-Inertial State Estimation in Legged Robots

Lin et al. [7] presented the first state estimator for legged robots which fused IMU and kinematic data in an Extended Kalman Filter (EKF). Their method was highly dependent on the assumption that their hexapedal robot had always three feet in contact with the ground. Bloesch et al. [2] introduced a new filtering approach called the Two-State Implicit Filter (TSIF). This method employs residual-based modeling of the available information, eliminating the need for an explicit process model. The default state estimator for the ANYmal robot, which we use in our experiments, is based on TSIF.

More recently, Pronto [8] has been presented as an EKF-based state estimator with the capacity to fuse pose corrections from exteroceptive sensing. This is done by maintaining a history of states, measurements and covariances, and applying corrections to the past trajectory asynchronously.

Factor graphs have also been used for legged robot state estimation. Hartley et al. [9] proposed a hybrid preintegrated contact factor so that multiple foot steps or measurements could be combined into a single factor, reducing the number of nodes in the graph. Wisth et al. [10] used factor graphs to fuse visual information with inertial and kinematics. In [11], the same authors improved on the factor graph formulation by adding feature tracking-based lidar measurements.

2.2 Learned Inertial Navigation

Recent research in inertial navigation systems (INS) has applied machine learning to infer motion directly from IMU data — typically focused on tracking a handheld mobile phone. These methods have shown impressive performance when compared to traditional inertial-only pedestrian tracking methods. However, these methods are susceptible to failure if applied outside of the training domain (e.g., a network trained with pedestrian motion would not generalize to a person on a bicycle).

Chen et al. [6] introduced IoNet, the first data driven INS which estimates odometry using deep recurrent neural networks (RNNs). The network was trained to relate buffered data from an IMU into 2D displacements and orientation changes. Yan et al. [12] proposed RoNIN, a network architecture which similarly infers 2D velocity and orientation from raw IMU data.

The first approach estimating 3D motion was proposed by Liu et al. [5], who presented the TLIO filtering framework including learned motion estimates from batches of IMU signals. An EKF was used to filter the full 6 DoF state and process updates were performed by traditional IMU integration. They trained a network with 40 h of pedestrian data to output displacement and covariance from the buffered IMU data. These measurements were provided to the EKF as relative position measurements and allowed the filter to correct for biases online. Their approach showed impressive results for IMU only navigation but was unable to integrate other sensors such as kinematics or vision.

Currently, the application of inertial learning to robotics is limited. Brossard et al. [13] demonstrated IMU-only state estimation specifically for wheeled vehicles, heavily exploiting the constrained motion model. Zhang et al. [14] trained a network to learn the noise parameters of the IMU as opposed to the motion of the subject. Their RNN generates rotation, velocity and acceleration terms with bias removed, however this makes integration with other frameworks such as filtering more difficult.

3 Problem Statement

Our objective is to estimate the state of a legged robot equipped with an IMU, joint sensors (encoders, torque sensors) and, optionally, exteroceptive sensors. Following standard conventions, we define a fixed world frame W aligned with gravity, and two frames attached to the robot: the base frame B and the IMU frame I . The transformations between B and I are known. The robot's state at the current time $x(t_k) = x_k$ is defined as follows:

$$x_k = [R_k, p_k, v_k, b_k^a, b_k^g] \quad (1)$$

and includes the rotation $R_{WB} \in SO(3)$ in world coordinates, the position ${}_W p_{WB} \in \mathbb{R}^3$, the linear velocity ${}_B v_{WB}$, and the gyro and accelerometer IMU biases ${}_I b^g, {}_I b^a$. The IMU measures rotational velocity ${}_I \tilde{\omega}_{WI} \in \mathbb{R}^3$ and the proper acceleration ${}_I \tilde{a} \in \mathbb{R}^3$. These are corrupted by zero-mean Gaussian noise η^g, η^a and by slowly changing biases:

$$\tilde{\omega}(t) = \omega(t) + b^g(t) + \eta^g(t) \quad \tilde{a}(t) = R^T({}_W a(t) - {}_W g) + b^a(t) + \eta^a(t) \quad (2)$$

where ${}_W g$ is the gravity vector in world frame. Without constraining measurements, the biases are not observable and simple IMU integration will quickly lead to runaway error.

4 Learned Displacement Measurement

In this section, we describe our learning method including network architecture and loss functions. Let $\mathcal{I}_{ij}$ be a sequence of N consecutive IMU measurements collected from time t_i to t_j and gravity aligned. From $\mathcal{I}_{ij}$, we want to predict the robot's linear position increment $\hat{d}_i \in \mathbb{R}^3$ between t_i and t_j . To this end, we define a nonlinear function $\mathcal{F}_\theta$ that will be approximated by a neural network:

$$\mathcal{F}_\theta : (\mathcal{I}_{ij}) \mapsto (\hat{d}_i, \hat{u}_i) \quad (3)$$

where $\hat{u}_i = (\hat{u}_i^x, \hat{u}_i^y, \hat{u}_i^z)$ contains the logarithms of standard deviations from which the covariance of the displacement $\hat{\Sigma}_i^d \in \mathbb{R}^{3 \times 3}$ can be computed:

$$\hat{\Sigma}_i^d = \text{diag}(\exp(2\hat{u}_i^x), \exp(2\hat{u}_i^y), \exp(2\hat{u}_i^z)) \quad (4)$$

Following an approach similar to TLIO [5], we approximate $\mathcal{F}_\theta$ with a 1D version of the ResNet18 architecture [15]. We also make use of two loss functions, but in contrast to [5] we use a robust Huber loss kernel to account for outliers in the training set.

4.1 Loss Function

For the first 10 epochs, the loss function is defined by the Mean Square Error (MSE):

$$\mathcal{L}(d, \hat{d}) = \frac{1}{n} \sum_{k=1}^n \|d_k - \hat{d}_k\|^2, \quad (5)$$

where n is the number of training samples, $\hat{d}$ is the network output, and d is the ground truth displacement. Once the network has begun to converge, the loss function is changed to a robust modification of the Gaussian Maximum Likelihood (GML), which allows the network to learn uncertainty (as done in [16]) but without being over influenced by outliers. If the MSE loss is not initially used we find the network does not converge. The negative log-loss of the GML is given by:

$$\begin{aligned} \mathcal{L}(d, \hat{d}) &= \frac{1}{n} \sum_{k=1}^n -\log \left(\frac{1}{\sqrt{8\pi \det(\hat{\Sigma}_k^d)}} \exp \left(H(d_k - \hat{d}_k, \hat{\Sigma}_k^d) \right) \right) \\ &= \frac{1}{n} \sum_{k=1}^n \left(\frac{1}{2} \log \det(\hat{\Sigma}_k^d) + H(d_k - \hat{d}_k, \hat{\Sigma}_k^d) + \text{const.} \right) \end{aligned} \quad (6)$$

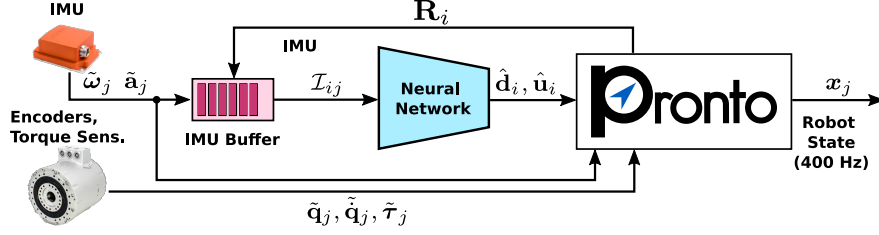


Figure 2: EKF-IKD pipeline: IMU data is integrated in Pronto which provides an estimate of the robot’s orientation. The IMU buffer is aligned with gravity and provided as input to the neural network. Displacement measurements and covariance are fused back into Pronto.

where $H(\epsilon_k, \hat{\Sigma}_k^d)$ is the sum of the Huber loss $h(\cdot, \cdot)$ applied to each axis of $\epsilon_k = \mathbf{d}_k - \hat{\mathbf{d}}_k$:

$$H(\epsilon_k, \Sigma_k^d) = h(\epsilon_k^x, \sigma_k^x) + h(\epsilon_k^y, \sigma_k^y) + h(\epsilon_k^z, \sigma_k^z) \quad h(\epsilon, \sigma) = \begin{cases} \frac{1}{2}\epsilon^2(\sigma^2)^{-1} & \text{if } |\epsilon| < \delta, \\ \delta|\epsilon| - \frac{1}{2}\delta^2 & \text{otherwise} \end{cases} \quad (7)$$

We treat each axis separately because, as shown in later sections, state estimation errors along the z -axis are often quite different from the errors on the xy -plane. Our loss function resulted in 12% less error computing displacements in test experiments compared to [5].

5 State Estimation

This network outputs a relative position measurement and uncertainty which is suitable for incorporation into both filtering and smoothing frameworks, as described in the following sections.

5.1 Filtering Estimator

We incorporate the learned displacement measurement into Pronto, an open-source modular legged robot estimation framework [8]. As Pronto maintains a history of states, measurements and covariances, it can easily incorporate relative position measurements. The full system architecture is shown in Fig. 2: the IMU measurements from time t_i to t_j are accumulated and rotated to be gravity aligned at time t_i using the orientation $\mathbf{R}_i$ from the filter’s internal state. They are then passed to the network to estimate displacement and uncertainty. These are fused, using the covariance in Eq. 4, into the filter as a relative position measurement:

$$\mathbf{z}_j = \mathbf{p}_i + \hat{\mathbf{d}}_i \quad (8)$$

where position $\mathbf{p}_i$ at time i is taken from the filter’s history, while displacement is from the network.

We call our proposed filtering method EKF-IKD (EKF with inertial, kinematic, and learned displacement). For all training and experiments, we used a buffer size of $N = 400$ (1 s of 400 Hz IMU data). Displacement measurements were estimated at 20 Hz, which are overlapping. We recognize that this overlap compromises independence of measurements but have not included analysis of this effect due to space limits.

5.2 Factor Graph Estimator

In this section, we describe a novel factor graph architecture that incorporates the learned inertial factor. In contrast to filters, factor graph methods re-optimize the full or partial history of states, given all the measurements connected to them, allowing for more accurate and smoother estimates.

Fig. 3 shows the proposed factor graph architecture. When a new camera keyframe is available, a new node (white circle) is created. Similarly to [17], we connect consecutive nodes $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ with preintegrated IMU and leg odometry factors (blue and orange factors), while visual features are modeled as individual landmarks (gray circles) and factors (yellow factors). In visually degraded environments, the visual landmarks might not be observable for a long period of time (e.g., from node $\mathbf{x}_3$ in the example), so it is important to have robust proprioceptive odometry to fall back on. To this end, we exploit the novel learned displacement factor to further constrain the robot motion

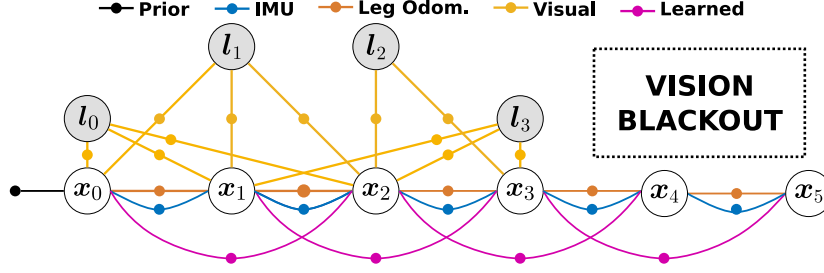


Figure 3: Proposed factor graph framework with IMU factor from [18], kinematics factor from [17] and our learned factor which specially helps in visual blackouts. The learned factor connections are dictated by the rates of the other sensors, as explained in Sec. 5.2.

(magenta line of Fig. 3). We call the factor graph implementation of our method FG-VIKD (factor graph with vision, IMU, kinematics and learned displacement).

The learned displacement measurement $\hat{\mathbf{d}}_i$ estimates the position of the robot between the start and end times of the buffer. The learned factor residual $\mathbf{r}_d$ can then be computed using $\hat{\Sigma}_i^d$ from Eq. 4:

$$\mathbf{r}_d = \left\| \mathbf{R}_i^T (\mathbf{p}_j - \mathbf{p}_i) - \hat{\mathbf{d}}_i \right\|_{\hat{\Sigma}_i^d}^2, \quad (9)$$

Fixed-lag optimization is done over the last 5 s of states each time a node is inserted to the graph. Since the IMU buffer is longer than the period between two keyframes, the nodes connected by the learned factor are not consecutive and instead skip several nodes. In the typical setup the camera creates nodes at 20 Hz and the IMU buffer length is 1 s meaning the learned factor would connect x_0 to x_{20} , x_1 to x_{21} etc. (for simplicity, every two nodes are connected in Fig. 3). We also integrate IMU measurements between optimizations to provide a 400 Hz signal state estimate.

6 Lab Experiments

We performed several experiments with our ANYbotics ANYmal C100 robot as shown in Fig. 4. The robot was teleoperated over different terrains in the lab: flat ground, soft ground, a slippery surface and an obstacle. Ground truth poses were measured using a Vicon motion capture system.

Flat Ground First, we show performance on rigid, flat terrain. In these conditions, kinematic state estimate is very accurate, however over long trajectories there is upward drift (the Z dimension).

Soft Ground Deformable terrain poses a major challenge to legged robot state estimation as the specific moment of contact is ill-defined — the foot continues to move and compress the terrain after a contact is detected. This additional downward motion results in a hallucinated upward Z drift. Some realistic examples of deformable terrain include sand, mud or tall grass. To reproduce this effect we placed several layers of foam padding on the floor.

Slipping When a robot slips, such as on wet or loose terrain, certain kinematic assumptions are violated as the foot is in contact with the terrain but does not have zero velocity. This results in large spikes in odometry error. To create slip events in the lab, we placed a common office white board on the floor and covered it with lubricant (hand sanitizer).

Obstacle Terrain Blind climbing over an uneven obstacle course with ramps. This experiment was designed to show that our method can generalize to motions on non-planar surfaces.

Figure 4: Lab experiments.

Left: Robot foot compressing into soft terrain. **Middle:** Slip event while walking on slippery terrain. **Right:** Robot climbing over an obstacle.



Table 1: Experimental Results for Filter-based methods.

	Absolute Pose Error (APE) [m]				5 m Relative Pose Error (RPE) [m]			
Data	TLIO	TSIF	EKF-IK	EKF-IKD	TLIO	TSIF	EKF-IK	EKF-IKD
FLAT	1.88	0.14	0.42	0.20	0.58	0.09	0.19	0.16
SLIP.	0.28	0.21	0.38	0.20	0.35	0.13	0.20	0.12
SOFT	0.31	0.35	0.53	0.24	0.31	0.15	0.24	0.16
OBST.	0.37	0.19	0.39	0.27	0.44	0.12	0.22	0.22

6.1 Training

For the lab experiments, we collected ten walking sequences for each terrain totaling 2 h16 min in 40 sequences. Four sequences were selected as test experiments and the remaining were split 75:25 into training and validation datasets. We trained a single model from all the data and used this same model in all experiments. We used the Adam optimizer with an initial learning rate of 10^{-4} and a Huber delta of 1.35. Training lasted 200 epochs with the model minimizing validation error used in all lab experiments. Training took 6 hours on a laptop with an Nvidia Quadro P2000 GPU.

We used the Vicon system to collect training data. First, we remove the biases from the raw IMU signal and inject back random bias values. In order to estimate the true IMU biases, we create a factor graph where each node has a prior pose factor from a Vicon measurement. IMU factors connect the nodes allowing the biases to be estimated in batch optimization for all Vicon measurements.

6.2 Filtering Results

We present results comparing **filtering-based** proprioceptive estimator Pronto (which we hereafter refer to as EKF-IK) which is open-source and uses IMU and kinematics and our proposed method which incorporates IMU, kinematics and the learned displacement measurement EKF-IKD. We also show the trajectory provided by the default state estimator for the ANYmal robot which we call TSIF and uses IMU and kinematics as well as the learning-based estimator TLIO.

We used two metrics: absolute pose error (APE) and relative pose error (RPE) – as defined in [19]. APE measures the amount of drift over an entire experiment. RPE measures the drift every X m traveled – we used 5 m for lab experiments and 10 m for the mine experiment. In all experiments, adding the learned measurement improved the state estimation significantly. As shown in Table 1, in comparison with EKF-IK, APE was reduced by 46 % on average (51 % for most challenging terrains: slipping and soft) and RPE was reduced by 22 % on average (37 % on challenging terrains).

Our proposed method out performs TLIO for every experiment which is expected as we use kinematics. The TSIF estimator outperforms our method in situations where the terrain is rigid and stable but is on par when slipping or on soft terrain. We attribute this to the years of fine-tuning by the company on this estimator for this particular robot. As such, we make no claims of beating the state-of-the-art, however we point to the significant improvement gains to EKF-IK from the addition of the learned measurement.

Flat Ground The results for the filtering methods on flat ground are shown on the left of Fig. 5. There is a clear upward Z drift for TSIF and EKF-IK which is significantly reduced using the learned measurement in EKF-IKD.

Soft Ground The results for the filtering methods on soft ground are shown on the right of Fig. 5. The periods where the robot walked on the soft terrain are highlighted in gray. The upward Z drift

Table 2: Experimental Results for Factor-Graph methods.

	Absolute Pose Error (APE) [m]		5 m Relative Pose Error (RPE) [m]	
Data	FG-IK	FG-IKD	FG-IK	FG-IKD
FLAT	0.36	0.22	0.25	0.24
SLIP.	0.23	0.20	0.18	0.17
SOFT	0.44	0.39	0.28	0.23
OBST.	0.23	0.21	0.18	0.18

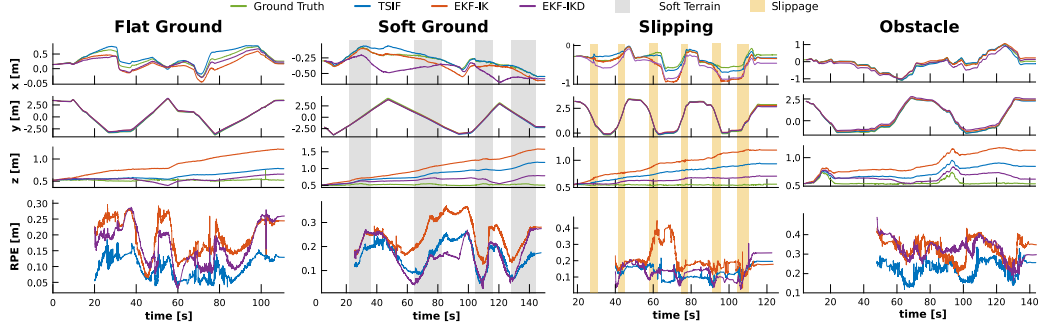


Figure 5: **Top:** Position in W estimate comparison for flat, soft, slippery, and obstacle grounds. Note the upward z drift for TSIF and EKF-IK on the soft (gray area) and slippery (yellow area) terrains. **Bottom:** 5 m Relative Pose Error (no data for first 5 m travelled).

for TSIF and EKF-IK is more pronounced but this is reduced in EKF-IKD. Along the x axis the performance is slightly worse indicating scale error introduced by the learned measurement but overall error is greatly reduced as shown in the RPE plot.

Slipping As shown in Fig. 5, periods when the robot is slipping have high error. The addition of the learned measurement in EKF-IKD significantly reduces spikes in error, particularly at 60 s. Our method does slightly better than TSIF in this experiment.

Obstacle Terrain This experiment demonstrates that our method can generalize to non-planar terrain. The upper right plot of Fig. 5 shows that EKF-IKD reduces Z drift while preserving the true changes in elevation as the robot climbs over the terrain. Initially, we found that if the training dataset did not include examples over elevated terrain, the network assumed constant planar elevation. The resulting state estimator then ignored elevation gain and loss from climbing on terrain.

6.3 Factor Graph Results

We present results comparing two **factor graph-based** proprioceptive estimators: FG-IKD which is our proposed factor graph implementation without vision data (simulating camera failure) and FG-IK which is the same factor graph without vision or the learned displacement. Since there is no vision in these cases, nodes are created at a 40 Hz fixed frequency in sync with the IMU.

We demonstrate that by adding the learned factor to this factor graph without any vision, APE was reduced by 18 % and RPE by 7 %, as shown in Table 2. The error in these factor graph methods is higher than the filtering methods because the robot state is intentionally estimated at a lower rate to allow visual data to be incorporated. Additionally, without landmarks, FG-IK cannot take advantage of the graph structure. Nevertheless, FG-IKD shows an improvement in proprioceptive estimation implying that if a robot’s vision system were to fail, our method would reduce drift.

7 Field Experiment: Underground Mine

While the learned displacement measurement can improve proprioceptive state estimation, most deployments of robots use vision systems. However, performance of these systems will degrade in challenging environments such as underground mines. In these cases, the learned displacement measurement can support the vision-based state estimation system. To demonstrate this, we performed a field experiment in which we teleoperated our ANYmal robot through a decommissioned limestone

Table 3: Experimental results for the Mine Experiment.

Data	Absolute Pose Error (APE) [m]				10 m Relative Pose Error (RPE)[m]			
	TLIO	EKF-IKD	FG-VIK	FG-VIKD	TLIO	EKF-IKD	FG-VIK	FG-VIKD
MINE	3.37	5.43	4.63	4.09	0.66	0.49	0.45	0.35

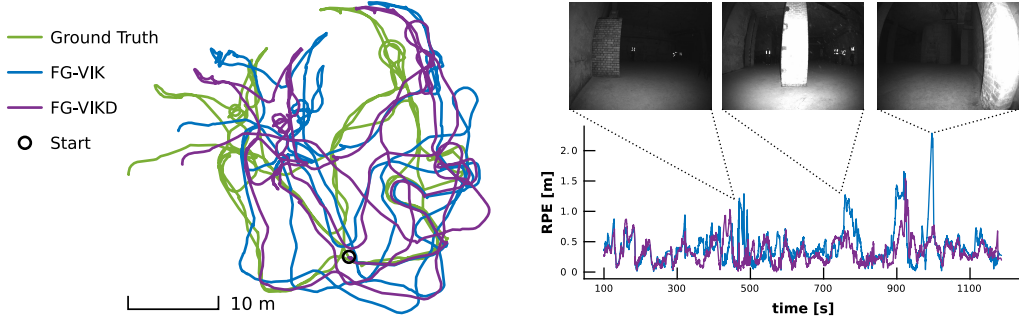


Figure 6: Results from the mine experiment. **Left:** Top-down perspective of FG-VIK vision-based factor graph and our method FG-VIKD in W. **Right:** 10 m RPE for both trajectories. We show images from three times where the error spikes in the baseline FG-VIK due to poor visibility.

mine with large rooms and narrow passageways as shown in Fig. 1. The floor is mostly hard with flat and sloped sections, but thick layers of dust and loose debris create slippery areas.

In the mine the controller from [20] was used instead of the default controller. Thus a new model was trained from lab data as described in Section 6.1, totaling 1 h23 min. Training data for the mine was collected entirely in the lab and the resulting model was successfully applied to the mine.

7.1 Mine Experiment Results

We compared a classical factor graph-based state estimator VILENS [10]; which uses vision, IMU and kinematics (and hereafter we refer to as FG-VIK); to our method FG-VIKD which uses the same sensors as well as the learned displacement factor. In the mine visual sensing is severely affected, forcing FG-VIK to rely on its proprioceptive sensing and, in these situations, the learned factor can help keep error low. Our experiment lasted 20 minutes and over 472 m of walking.

Ground truth was estimated using the robot’s lidar to localize in a prior, high quality, point cloud map. Table 3 shows that RPE was reduced by 22 % by adding the learned factor. We also show TLIO and EKF-IKD as baselines although they do not use cameras. The trajectory and error over time are shown in Fig. 6 along with images from the front facing camera at times where the drift rate spiked. These spikes coincide with periods of poor illumination and visual tracking failure in the dark environment. The robot lights reflected off nearby surfaces triggering the camera to reduce its exposure making the image darker. The performance improvement would likely be reduced in radically different terrain such as sand or snow; however, this experiment demonstrates generalization from the artificial lab terrains to real-world environments.

8 Conclusion

In this paper, we introduced a learned displacement measurement derived from inertial sensing which can greatly reduce state estimation drift for legged robots in challenging situations. We showed how this measurement can be used in both filtering and factor graph frameworks, and that the greatest benefits are when the robot slips or the terrain is deformable. Extensive experiments with the ANYmal quadruped robot showed a reduction of 37 % in relative pose error for filter-based state estimation. We have also shown how the learned displaced measurement can reduce error in a vision driven factor graph framework by 22 % in a underground mine. A video demonstrating these experiments is provided as supplementary material.

For future work, inclusion of kinematic data into the network will be explored. Directly using the joint states as input to the network would likely require a much more complex architecture that might not generalize to different robot models. To this end, we seek to incorporate derivative inputs from the robot model, such as ground reaction forces or foot velocities.

Acknowledgments

This research has been conducted as part of the ANYbotics research community. It was part funded by the EU H2020 Project THING (Grant ID 780883) and a Royal Society University Research Fellowship (Fallon).

References

- [1] 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*, 37(8):1314–1327, 2020. doi:<https://doi.org/10.1002/rob.21964>.
- [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*, 3(1):573–580, 2018. doi:[10.1109/LRA.2017.2776340](https://doi.org/10.1109/LRA.2017.2776340).
- [3] 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*, 2(2):1023–1030, 2017. doi:[10.1109/LRA.2017.2652491](https://doi.org/10.1109/LRA.2017.2652491).
- [4] R. Hartley, J. Mangelson, L. Gan, M. G. Jadidi, J. M. Walls, R. M. Eustice, and J. W. Grizzle. Legged robot state-estimation through combined forward kinematic and preintegrated contact factors. *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4422–4429, 2018. doi:[10.1109/ICRA.2018.8460748](https://doi.org/10.1109/ICRA.2018.8460748).
- [5] 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*, 5(4):5653–5660, 2020. doi:[10.1109/LRA.2020.3007421](https://doi.org/10.1109/LRA.2020.3007421).
- [6] C. Chen, X. Lu, A. Markham, and N. Trigoni. IoNet: Learning to cure the curse of drift in inertial odometry. *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pages 6468–6476, 2018.
- [7] P. Lin, H. Komsuo, and D. E. Koditschek. Sensor Data Fusion for Body State Estimation in a Hexapod Robot With Dynamical Gaits. *IEEE Transactions on Robotics*, 22(5):932–943, 2006.
- [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*, 7, 2020. doi:[10.3389/frobt.2020.00068](https://doi.org/10.3389/frobt.2020.00068).
- [9] 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 Using Factor Graphs. *IEEE International Conference on Intelligent Robots and Systems*, pages 3783–3790, 2018. doi:[10.1109/IROS.2018.8593801](https://doi.org/10.1109/IROS.2018.8593801).
- [10] D. Wisth, M. Camurri, and M. Fallon. Robust Legged Robot State Estimation Using Factor Graph Optimization. *IEEE Robotics and Automation Letters*, 4(4):4507–4514, 2019. doi:[10.1109/LRA.2019.2933768](https://doi.org/10.1109/LRA.2019.2933768).
- [11] 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*, 6(2):1004–1011, Apr 2021. doi:[10.1109/LRA.2021.3056380](https://doi.org/10.1109/LRA.2021.3056380).
- [12] S. Herath, H. Yan, and Y. Furukawa. Ronin: Robust neural inertial navigation in the wild: Benchmark, evaluations, new methods. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3146–3152, 2020. doi:[10.1109/ICRA40945.2020.9196860](https://doi.org/10.1109/ICRA40945.2020.9196860).
- [13] M. Brossard, A. Barrau, and S. Bonnabel. AI-IMU dead-reckoning. *IEEE Transactions on Intelligent Vehicles*, 5(4):585–595, 2020. doi:[10.1109/TIV.2020.2980758](https://doi.org/10.1109/TIV.2020.2980758).
- [14] M. Zhang, M. Zhang, Y. Chen, and M. Li. IMU data processing for inertial aided navigation: A recurrent neural network based approach. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

- [15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi:[10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [16] R. L. Russell and C. Reale. Multivariate uncertainty in deep learning. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
- [17] 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)*, pages 392–398, 2020. doi:[10.1109/ICRA40945.2020.9197214](https://doi.org/10.1109/ICRA40945.2020.9197214).
- [18] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza. IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. *Robotics: Science and Systems*, 11, 2015. doi:[10.15607/RSS.2015.XI.006](https://doi.org/10.15607/RSS.2015.XI.006).
- [19] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 573–580, 2012. doi:[10.1109/IROS.2012.6385773](https://doi.org/10.1109/IROS.2012.6385773).
- [20] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47), 2020. doi:[10.1126/scirobotics.abc5986](https://doi.org/10.1126/scirobotics.abc5986).

PMLR link as required by copyright:

<https://proceedings.mlr.press/v164/buchanan22a.html>

Paper is licensed under Creative Commons 4.0 International (CC BY 4.0)

<https://creativecommons.org/licenses/by/4.0/>

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	Learning Inertial Odometry for Dynamic Legged Robot State Estimation
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Russell Buchanan, Marco Camurri, Frank Dellaert, and Maurice Fallon, "Learning inertial odometry for dynamic legged robot state estimation," in <i>Proceedings of the 5th Conference on Robot Learning (CoRL)</i> , November 2021.

Student Confirmation

Student Name:	Russell Buchanan		
Contribution to the Paper	• Developed the original idea • Wrote all the code adapting open sourced prior work • Collected all the data and trained all models • Analyzed the data • Wrote a large portion of the paper		
Signature		Date	February 24, 2023

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 contributions listed above are an accurate representation of the work Russell contributed			
Signature		Date	February 24, 2023

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

6.2 Markerless Handheld Ultrasound 3D Reconstruction of Bone using Visual-Inertial and Deep Inertial Odometry

The following article is a draft paper intended to be submitted to IEEE Transactions on Biomedical Engineering (TBME).

Markerless Handheld Ultrasound 3D Reconstruction of Bone using Visual-Inertial and Deep Inertial Odometry

Russell Buchanan, Shih-fan Jack Tu, Marco Camurri, Stephen J. Mellon, Maurice Fallon

Abstract—The main limitation to widespread use of 3D bone reconstruction from handheld ultrasound scanning is the need for expensive, external infrastructure, such as motion tracking cameras, to estimate the trajectory of the scanner. In this paper, we assess external infrastructure-free methods for estimation of a handheld ultrasound scanner. We compare Visual-Inertial Odometry (VIO) and deep learning-based inertial-only odometry to estimate the pose of the scanner. These two markerless methods are compared against a motion-capture baseline and the resulting 3D reconstructions are presented. We show how using either VIO or deep inertial odometry we can reconstruct the femur and visualize the location of the patella from handheld ultrasound scans. The visual-inertial method in particular, performs as well as the motion capture method with average reconstruction errors of 1.25 mm and 1.21 mm respectively.

Index Terms—Ultrasound, bone, reconstruction, visual odometry, inertial estimation

I. INTRODUCTION

Ultrasound imaging has enormous potential for bone visualization because it is low cost and radiation free compared to MRI and X-ray imaging [1]. This has resulted in significant interest in using ultrasound for 3D reconstruction of bone [2]–[5].

To perform 3D reconstruction from a stream of ultrasound images, the position and orientation of the device during scanning must be estimated. This problem is known as “state estimation” and is a well established research area in the field of robotics [6]. One major drawback of existing approaches is the need for external infrastructure to facilitate state estimation, e.g. a motion capture system [2], [4], electromagnetic marker tracking [5] or using a robotic arm to perform the scanning [3]. All of these methods have significant

infrastructure requirements which increase cost and limit the usefulness of the tool.

In the field of robotics, there are well established techniques for state estimation without external infrastructure. One major research area is Visual Odometry (VO); whereby visual features in the environment are tracked by a camera to estimate motion [7]. This is often combined with Inertial Measurement Units (IMUs) which are small, low-cost sensors capable of measuring acceleration and rotation rates. Combining IMUs with vision is known as Visual-Inertial Odometry (VIO) and has superior robustness to visual challenges such as darkness or motion blur [8], [9].

Typically, IMU-only estimation has been considered impractical for more than a few seconds due to rapid accumulation of integration error [10]. However, there has recently been study of deep learning methods to enable low-drift IMU-only odometry [11]–[13]. These methods train deep neural networks to learn a motion model of a walking subject e.g. pedestrians or walking robots. The network can then infer displacement measurements from IMU data directly. These learned measurements can be combined with classical IMU integration to create a deep inertial IMU-only estimation method with low drift compared to only using integration. This technique has been successfully used to track the motion of pedestrians and robots [12], [13].

In this work, we propose to apply these recent robotic state estimation techniques to track the motion of a handheld ultrasound scanner. We present a low-cost system for 3D reconstruction from a handheld ultrasound scanner — the estimation sensors cost under \$500 USD. First, we mount a camera on an ultrasound scanner and use a VIO method based on [14] to estimate the pose trajectory. This was similarly done in [15]; however, in their method they use a robot arm to perform scanning and place fiducial markers around the room. In our method, all scanning is done by a human and no external infrastructure is required.

Secondly, we evaluate a deep learning IMU-only estimation method based on our previous work in [13]. Deep inertial odometry depends on periodic, repetitive motion that a neural network can learn. Repeated back and forth scanning motions are well suited for this. In this work, we show how a neural network can be trained to learn the motion of scanning bones from IMU data and then use this model to infer motion in new scanning sessions. We evaluate the accuracy of this IMU-only method and it compare to VIO and a baseline method using

Manuscript received XXX; revised XXX; accepted XXX. Date of publication XXX; date of current version XXX. This work was supported in part by the EPSRC ORCA Robotics Hub under Grant EP/R026173/1, in part by EU H2020 Project THING under Grant 780883, and in part by Royal Society University Research Fellowship (Fallon). For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this submission. (Corresponding author: Russell Buchanan.)

Russell Buchanan, Marco Camurri, and Maurice Fallon are with the Oxford Robotics Institute, University of Oxford, Oxford, UK (e-mail: russell@robots.ox.ac.uk).

Shih-fan Jack Tu and Stephen J. Mellon are with the Oxford Orthopaedic Engineering Centre, Nuffield Department of Orthopaedics, Rheumatology and Musculoskeletal Sciences, University of Oxford, Oxford, UK

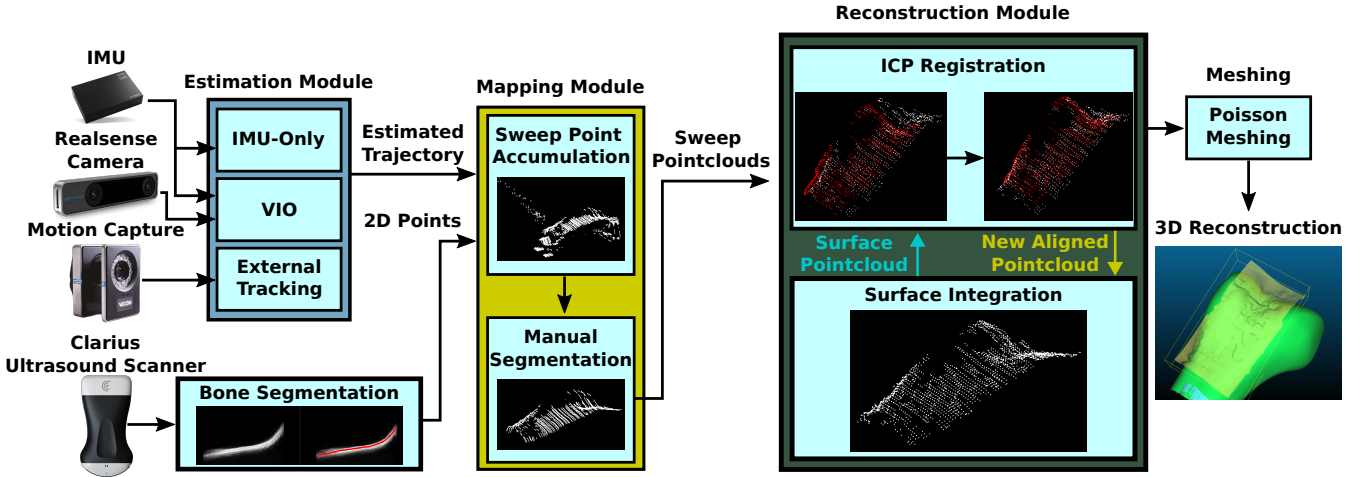


Fig. 1. Flow diagram of complete reconstruction pipeline. The estimation module computes the scanning trajectory using either IMU-only, VIO or motion capture external tracking. Separately, 2D ultrasound points are segmented for the presence of bone. These points are projected into 3D using the estimated trajectory and accumulated into separate point clouds for each sweep over the bone. After manually segmenting out the surface, the point clouds are aligned using ICP and integrated into the surface estimate. Finally, once all the sweeps have been integrated, the final surface is sampled and converted to a mesh using the Poisson method.

external motion tracking.

Finally, in this paper we present a full pipeline for 3D reconstruction of bone. We accumulate scan points and align several sweeps together using Iterative Closest Point (ICP) method [16]. The aligned point clouds are then fused to estimate the bone surface and a final mesh is generated. Unlike in [5], we do not use a statistical shape model (SSM) of human bone to interpolate the reconstruction which could bias the final reconstruction. A diagram of the full system pipeline is provided in Fig. 1.

The proposed methods enable markerless reconstruction of bone from ultrasound images acquired by a human. This promises to make ultrasound reconstruction a more flexible and useful tool in medical settings.

II. ESTIMATION METHODS

We compare three different methods of tracking the ultrasound scanner: motion capture, visual-inertial odometry, and deep learning IMU-only odometry. For motion capture, we place markers on the ultrasound scanner which are then tracked by an Optitrack system (see Fig 3). For VIO, we fuse information from a stereo camera and an IMU mounted on the scanner using the method from [14], which is based on factor graph optimization. For the IMU-only method, we train a neural network to output estimated position increments (i.e., displacements) from raw IMU data. The displacements are fused with classical IMU integration, similar to our previous work [13].

In this section, we will briefly summarize the factor graph-based estimation method from [14], which forms the foundation of our VIO and IMU-only tracking methods. We then briefly explain the training process of the neural network for the learned IMU displacement measurements.

A. Maximum-A-Posteriori (MAP) Estimation

We aim to estimate, in batch, the Maximum-A-Posteriori (MAP) state trajectory of a handheld ultrasound scanner. We present two different sensing modalities: VIO and IMU-only.

1) *State Definition*: The state of the scanner at time t_j is defined as:

$$\mathbf{x}_j \triangleq [\mathbf{R}_j, \mathbf{p}_j, \mathbf{v}_j, \mathbf{b}_j^g, \mathbf{b}_j^a] \in \text{SO}(3) \times \mathbb{R}^{12} \quad (1)$$

where $\mathbf{R}_j = \mathbf{R}_{\text{WB}}(t_j)$ is the orientation of the scanner base frame B with respect to the world frame W (frames are shown in Fig. 3). The estimated position $\mathbf{p}_j = {}_{\text{W}}\mathbf{p}_{\text{WB}}(t_j)$, is the position of the sensor in world frame, $\mathbf{v}_j = {}_{\text{B}}\mathbf{v}_{\text{WB}}(t_j)$ is the linear velocity in base frame, and $\mathbf{b}_j^g = {}_{\text{I}}\mathbf{b}_j^g(t_j)$, $\mathbf{b}_j^a = {}_{\text{I}}\mathbf{b}_j^a(t_j)$ are the slowly changing biases in the gyroscope and accelerometer of the IMU; they are expressed in the IMU frame I which we assume is coincident with B.

From the camera images, K keyframes are selected with time indices j . From these images, L visual landmarks $\mathbf{m}_\ell$ (parameterized as 3D points in Euclidean space) are detected. We can describe the set of all states and visual landmarks as $\mathcal{X} = \{\mathbf{x}_j, \mathbf{m}_\ell\}_{j \in K, \ell \in L}$ or, in the IMU-only case, only the states $\mathcal{X} = \{\mathbf{x}_j\}_{j \in K}$.

2) *Measurement Definitions*: All IMU measurements are accumulated into buffers. Buffers between consecutive keyframes are denoted as $\mathcal{I}_{ij}$, where $i = j - 1$ and $i \in K$. These measurements are used for classical integration between the camera keyframes.

Another set of IMU buffers is collected between time indices k and j . These are used for the inertial displacement learning and consist of a fixed number of N measurements.

We can describe the set of all measurements for VIO as $\mathcal{Z} = \{\mathcal{I}_{ij}, \mathcal{C}_j\}_{i,j \in K}$ where $\mathcal{C}_j$ are the keyframe images. In the IMU-only case, there are no keyframes and instead states are estimated at a specified frequency. Therefore, with slight abuse of notation, we also use K in this case to represent the number

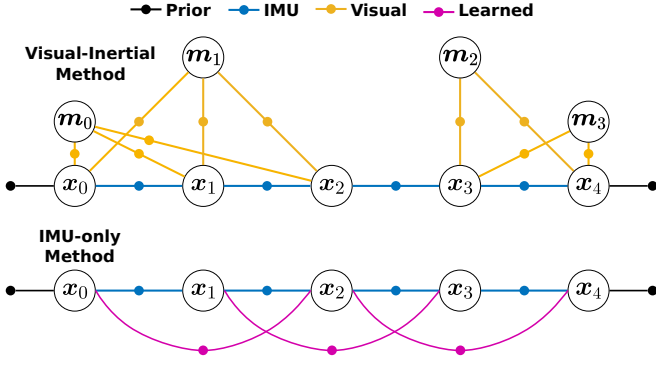


Fig. 2. Two example factor graph frameworks for five states $j = 0, 1, 2, 3, 4$. The four types of factors are shown: prior factors constraining the start and end pose to be equivalent, IMU integration factor from [17], visual landmark factors from [14] and our learned displacement factor from [13]. On top is the VIO method which uses prior, visual and IMU factors. On bottom is the learned inertial odometry method which only uses IMU and learned factors.

of key times the state is estimated. Therefore, in the IMU-only case we use $\mathcal{Z} = \{\mathcal{I}_{ij}, \mathcal{I}_{kj}\}_{k,j \in K}$

3) *Least Squares Minimization*: For each estimation method, we maximize the likelihood of the measurements $\mathcal{Z}$, given the history of states $\mathcal{X}$:

$$\mathcal{X}^* = \arg \max_{\mathcal{X}} p(\mathcal{X}|\mathcal{Z}) \propto p(\mathbf{x}_0)p(\mathcal{Z}|\mathcal{X}) \quad (2)$$

where $\mathcal{X}^*$ is our MAP estimate of the trajectory.

We assume the measurements are conditionally independent and corrupted by zero-mean Gaussian noise. Therefore, (2) can be expressed as the following least squares minimizations, first for VIO:

$$\mathcal{X}^* = \arg \min_{\mathcal{X}} \|\mathbf{r}_{\text{prior}}\|_{\Sigma_{\text{prior}}}^2 + \sum_{j \in K} \left(\|\mathbf{r}_{\mathcal{I}_{ij}}\|_{\Sigma_{\mathcal{I}_{ij}}}^2 + \sum_{\ell \in M} \|\mathbf{r}_{\mathbf{x}_j, \mathbf{m}_\ell}\|_{\Sigma_{\mathbf{x}_j, \mathbf{m}_\ell}}^2 \right) \quad (3)$$

or in the IMU-only case:

$$\mathcal{X}^* = \arg \min_{\mathcal{X}} \|\mathbf{r}_{\text{prior}}\|_{\Sigma_{\text{prior}}}^2 + \sum_{j \in K} \left(\|\mathbf{r}_{\mathcal{I}_{ij}}\|_{\Sigma_{\mathcal{I}_{ij}}}^2 + \|\mathbf{r}_{\mathbf{d}_j}\|_{\Sigma_{\mathbf{d}_j}}^2 \right) \quad (4)$$

where each term is the residual $\mathbf{r}$ associated to a measurement type, weighted by the inverse of its covariance matrix. The residual $\mathbf{r}_{\text{prior}}$ represents prior information on the state, $\mathbf{r}_{\mathcal{I}_{ij}}$ for IMU integration between consecutive states, $\mathbf{r}_{\mathbf{x}_j, \mathbf{m}_\ell}$ for visual landmarks and $\mathbf{r}_{\mathbf{d}_j}$ for learned displacements. Details of each residual are provided in the following sections.

A factor graph can be used to graphically represent (3) and (4) as shown in Fig. 2.

B. Prior Factors

In this work, we assume each trajectory has a “loop closure,” meaning the scanner trajectory ends with the same position and orientation as at the start. We also assume zero velocity at start and end. Therefore, we include this constraint as prior factors in the graph using the following definition:

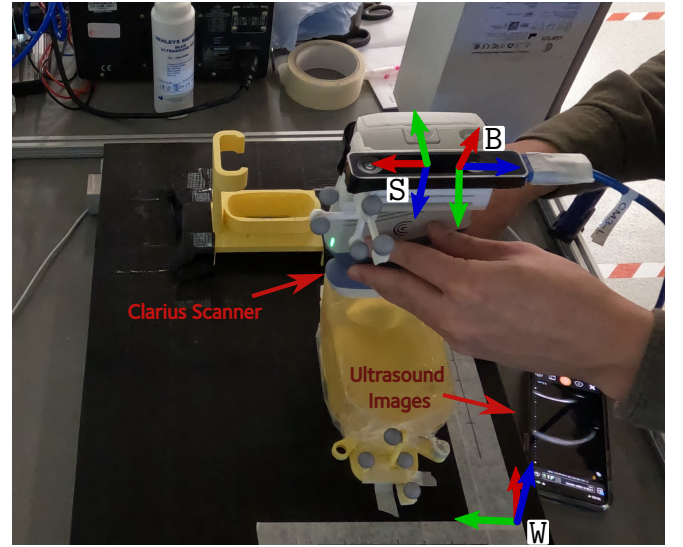
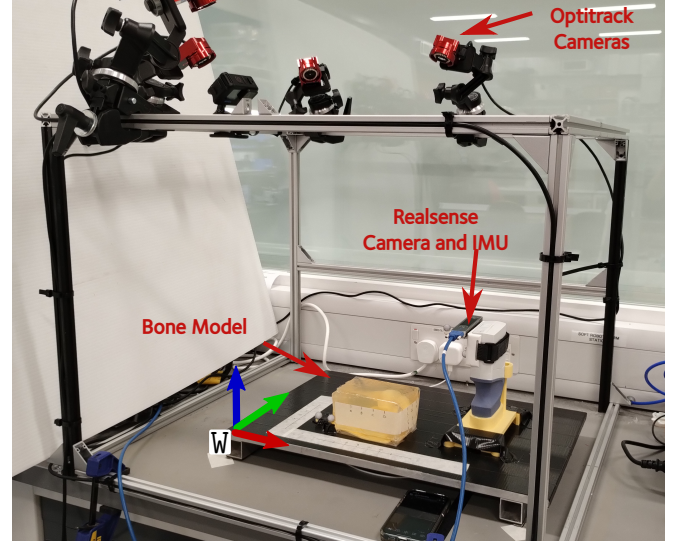


Fig. 3. Experimental setup for collecting ultrasound dataset. A Realsense stereo camera is mounted on the Clarius ultrasound scanner and a mount holds the scanner in place. The human operator picks up the scanner and passes it over the bone model several times before returning it to the mount.

$$\mathbf{r}_{\text{prior}} = [\mathbf{r}_{\mathbf{p}}^T, \mathbf{r}_{\mathbf{v}}^T, \mathbf{r}_{\mathbf{R}}^T]^T \quad (5)$$

where

$$\begin{aligned} \mathbf{r}_{\mathbf{p}} &= \mathbf{p}_j - \mathbf{p}_0 \\ \mathbf{r}_{\mathbf{v}} &= \mathbf{v}_j - \mathbf{0}_{3 \times 1} \\ \mathbf{r}_{\mathbf{R}} &= \mathbf{R}_j^T \mathbf{R}_0. \end{aligned} \quad (6)$$

Two prior factors are used, one for the first pose $j = 0$ and one for the last pose $j = K$ as shown in Fig 2.

C. Preintegrated IMU Factors

We use standard IMU preintegration theory to connect consecutive nodes in the graph, as in [17]. This relates the pose, velocity, and biases between two consecutive states. The

residual has the following form:

$$\mathbf{r}_{\mathcal{I}_{i,j}} = [\mathbf{r}_{\Delta\mathbf{R}_{i,j}}^T, \mathbf{r}_{\Delta\mathbf{v}_{i,j}}^T, \mathbf{r}_{\Delta\mathbf{p}_{i,j}}^T, \mathbf{r}_{\mathbf{b}_{i,j}^a}^T, \mathbf{r}_{\mathbf{b}_{i,j}^g}^T]^T \quad (7)$$

For a detailed definition of the above residuals, see [17].

D. Stereo Landmark Factors

A visual landmark in Euclidean space $\mathbf{m}_\ell \in \mathbb{R}^3$ is projected onto the keyframe image plane by the function $\pi : \text{SE}(3) \times \mathbb{R}^3 \mapsto \mathbb{R}^2$, given the platform pose $\mathbf{T}_j = \{\mathbf{p}_j, \mathbf{R}_j\} \in \text{SE}(3)$. Given a landmark $\mathbf{m}_\ell$ and its coordinates $(u_\ell, v_\ell) \in \mathbb{R}^2$ on the image plane, the residual at state j is computed as:

$$\mathbf{r}_{\mathbf{x}_j, \mathbf{m}_\ell} = \begin{pmatrix} \pi_u^L(\mathbf{T}_j, \mathbf{m}_\ell) - u_{j,\ell}^L \\ \pi_u^R(\mathbf{T}_j, \mathbf{m}_\ell) - u_{j,\ell}^R \\ \pi_v(\mathbf{T}_j, \mathbf{m}_\ell) - v_{j,\ell} \end{pmatrix} \quad (8)$$

where (u^L, v) , (u^R, v) are the pixel locations of the landmark.

E. Learned Displacement Factors

The buffers $\mathcal{I}_{kj}$, of N consecutive IMU measurements, are collected from time t_k to t_j and aligned to the gravity vector using $\mathbf{R}_k$. From $\mathcal{I}_{kj}$, we want to predict the scanner's position increment $\hat{\mathbf{d}}_j \in \mathbb{R}^3$ between t_k and t_j . To this end, we define a nonlinear function $\mathcal{F}_\theta$ that will be approximated by a neural network:

$$\mathcal{F}_\theta : (\mathcal{I}_{kj}) \mapsto (\hat{\mathbf{d}}_j, \hat{\mathbf{u}}_j) \quad (9)$$

where $\hat{\mathbf{u}}_j = (\hat{u}_j^x, \hat{u}_j^y, \hat{u}_j^z)$ contains the logarithms of standard deviations from which the covariance of the displacement $\hat{\Sigma}_j^d \in \mathbb{R}^{3 \times 3}$ can be computed:

$$\hat{\Sigma}_j^d = \text{diag}(\exp(2\hat{u}_j^x), \exp(2\hat{u}_j^y), \exp(2\hat{u}_j^z)) \quad (10)$$

We use the same neural network and training loss function as in our previous work [13].

The learned displacement measurement $\hat{\mathbf{d}}_j$ and covariance $\hat{\Sigma}_j^d$ can now be incorporated into the factor graph with the following residual:

$$\mathbf{r}_{\mathbf{d}_j} = \|\mathbf{R}_i^T(\mathbf{p}_j - \mathbf{p}_i) - \hat{\mathbf{d}}_j\|_{\hat{\Sigma}_j^d}^2 \quad (11)$$

F. Implementation

The state estimator was implemented using the factor-graph library GTSAM [18] and built on the work from [14] while the neural network was the same as in [13] and developed using the PyTorch library.

The camera was sampled at 30 Hz with one in three frames selected as a keyframe. Therefore, in the VIO method, nodes are added to the graph at 10 Hz, with IMU integration factors linking each consecutive pair.

IMU data was recorded at 200 Hz. For the neural network, the data samples were buffered in windows of 1 s ($N = 200$). Neural network inference was done at a rate of 30 Hz, which we empirically found to give the best performance. Therefore, for each learned displacement factor, $k = j - 30$, meaning each factor linked two states 1 s apart. For simplicity, Fig 2 shows the learned displacement factor connecting every second node. In practice, because inference is done at 30 Hz and because

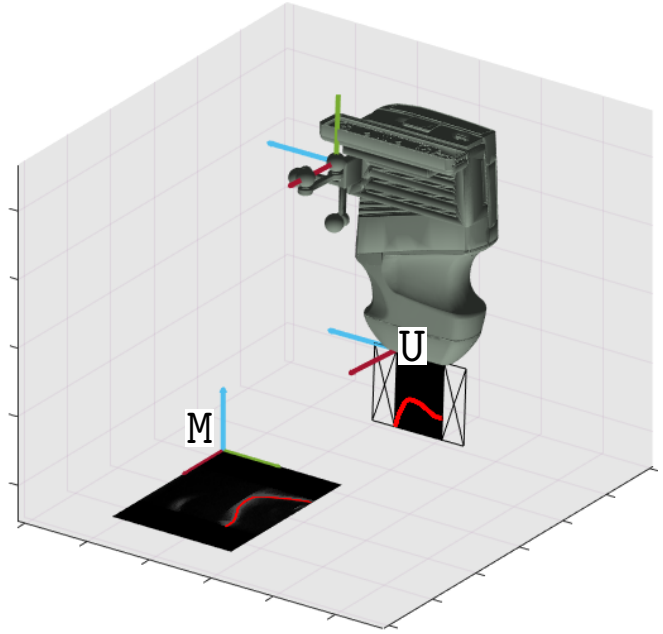


Fig. 4. Ultrasound images collected through the Clarius API are in a size of 640×480 . The pixel (1,1) of the image was set as the origin of the ultrasound image frame $\mathbf{M}$. When projecting the segmented points into 3D coordinates, a transformation matrix was applied to all bone surface points to transform them from the $\mathbf{M}$ into the ultrasound scanner's frame $\mathbf{U}$. Finally, the segmented points in the scanner's local coordinate system were transformed to the scanner positions in the global coordinate system according to the odometry source from the tracking modules.

$k = j - 30$, the factors connect every 30th node, i.e., node 0 to 30, 1 to 31 etc. Therefore, the first 1 s of IMU data is needed before the neural network can start producing outputs.

Motion estimation using external cameras, such as Optitrack, is expected to have error under 1 mm when properly calibrated. However, it is still possible for occlusions to cause “jumps” in the Optitrack estimation, which can introduce estimation errors. Therefore, we apply a simple median filter (window size 9) to the Optitrack data which was recorded at 200 Hz. This filtered trajectory was then used as “ground truth” for comparison of other estimation methods.

III. RECONSTRUCTION METHODS

The following section describes the reconstruction pipeline. These methods take inspiration from previous work in [4] but use weighted projective fusion for point cloud integration and surface reconstruction. The final output is a mesh which can be directly compared with ground truth meshes of bone phantom models. The full pipeline is shown in Fig. 1.

A. Bone Segmentation and Mapping Module

The first step in reconstruction is segmentation of bone from ultrasound images. We do this using the method from [19] which applies dynamic programming to solve for a best fit bone curve and detect which pixels correspond to the bone as shown in Fig 4.

The segmented pixels are then projected into 3D as points. With the odometry source from the tracking module, the

projected points are accumulated into point clouds according to the “sweep” number over the bone. This is done manually according to timestamps in the recorded data.

Finally, the reconstructed point clouds are reviewed and manually cleaned by removing mislabeled points that were caused by the misclassification of the other solid objects, such as the table and mounts, and reflection artifacts. Once removed, several point clouds of the bone surface, each corresponding to a separate sweep, were carried on for further evaluation.

B. Alignment of Point Cloud Sweeps

For VIO and IMU-only methods there is incremental drift while estimating the scanner’s pose. Knowing that we are scanning the same bone over several sweeps, we can correct for this drift and align the sweep points using ICP [16]. However, the sweeps are treated as rigid point clouds and therefore any error caused by odometry drift while moving across the bone cannot be corrected. We assume that the drift incurred over this small distance is not enough to severely degrade the reconstruction.

While the Optitrack-estimated sensor pose is not subject to drive, we find that aligning point clouds also slightly improves reconstruction for this configuration.

C. Surface Reconstruction and Meshing

Aligning point clouds using ICP will result in a reconstruction of the bone structure but, due to noise within each sweep, the reconstructed surface will be rough and contain artifacts. Fusing the aligned point clouds will result in a smoother reconstructed surface.

We use a Truncated Signed Distance Fields (TSDF) representation and integration algorithm from [20]. TSDFs are a volumetric representation where voxel values correspond to the signed distance from the surface boundary, being positive outside the surface and negative when penetrating inside the surface. The voxels are stored up to a truncated distance from the surface $\pm\mu$. The TSDF $\mathbf{S}(\mathbf{p})$ at point $\mathbf{p} \in \mathbb{R}^3$ can be expressed as:

$$\mathbf{S}(\mathbf{p}) \rightarrow [F(\mathbf{p}), W(\mathbf{p})] \quad (12)$$

where $F(\mathbf{p})$ represents the signed distance and $W(\mathbf{p})$ is a corresponding weight.

In order to integrate the point clouds into a TSDF they must first be converted to a depth image $R(u, v)$ where (u, v) are pixel coordinates in the image plane. As the sweep point clouds are formed by multiple ultrasound viewpoints, we assume there exists an ideal depth camera capable of viewing the same point cloud from a viewing pose $\mathbf{T}_{ws}(t_j)$ where S is defined on the end of the scanner, opposite from the ultrasound image plane (see Fig. 3).

We record the position of the scanner ${}_{w}\mathbf{P}_{ws}(t_j)$, for each sweep, at the beginning of the sweep. We then compute rotation $\mathbf{R}_{ws}(t_j)$ which aligns the z axis of S (see Fig. 3) with a vector pointing in the direction of the bone model. This is the viewing angle of the ideal depth camera and

together with position gives the viewing pose $\mathbf{T}_{ws}(t_j) = \{{}_{w}\mathbf{P}_{ws}(t_j), \mathbf{R}_{ws}(t_j)\}$.

With this pose, we use an ideal camera projection matrix to project the 3D points into a 2D depth image $R(u, v)$. Now, using the method from [20], we fuse the depth images using a weighted average of measurements:

$$F_k(\mathbf{p}) = \frac{W_{j-1}(\mathbf{p})F_{j-1}(\mathbf{p}) + W_{R_j}(\mathbf{p})F_{R_j}(\mathbf{p})}{W_{j-1}(\mathbf{p}) + W_{R_j}(\mathbf{p})} \quad (13)$$

$$W_k(\mathbf{p}) = W_{j-1}(\mathbf{p}) + W_{R_j}(\mathbf{p})$$

where $F_{R_j}(\mathbf{p})$ is computed from raycasting the depth measurement to the surface and $W_{R_j}(\mathbf{p})$ is nominally set to 1 with $W_j(\mathbf{p})$ truncated to some value. More details are provided in [20].

Alignment and surface reconstruction are done in a loop whereby point clouds are first aligned, then integrated into the surface. A point cloud sampling of the TSDF surface (voxels where $F_k(\mathbf{p}) = 0$) is then extracted and used to align the next point cloud. Once all the sweeps have been aligned and integrated the final surface, then point cloud is converted to a mesh using the Poisson method [21].

D. Implementation

We use the open source library libpointmatcher [22] to perform ICP. We use 1 mm voxel filtering and Point-to-Plane matching to improve robustness to outlier points. We also constrain ICP to position and yaw (rotation about the gravity axis) only. Due to its accelerometer, the pitch and roll estimate from the IMU will be very accurate over the short time period of a single scan.

For TSDF integration, we use the open source library Open3D [23]. We compute the TSDF with 2 mm resolution and 1 cm truncation.

IV. EXPERIMENTS

In this section we explain the experimental setup which was used to create a dataset of ultrasound scans for training the neural network.

A. Experimental Setup

The experimental setup used is shown in Fig 3. The ultrasound scanner used was the handheld Clarius L7 HD3¹. We mounted an Intel Realsense T265 stereo camera which includes a Bosch BMI085 IMU. The Clarius scanner also contains an IMU however due to difficulty accessing the raw data in real time through the API we opted to use the Realsense’s onboard IMU for these experiments.

A 3D printed mount held the scanner and camera together. As a scanning target, we used two different phantom models of the same distal femur with the patella in two different positions. For model 1, the patella is in the position of 40° knee flexion. For model 2, the patella is in the position of full

¹Clarius Mobile Health Corporation, Burnaby, BC, Canada

TABLE I
POSITION ESTIMATION ERROR [MM]

Method	APE						RPE (20 mm)					
Bone	Bone 1		Bone 2		Bone 3		Bone 1		Bone 2		Bone 3	
Scan Type	A	B	A	B	A	B	A	B	A	B	A	B
VIO	8.7	3.8	3.5	4.2	12.8	5.0	2.3	1.9	1.6	1.2	2.7	2.1
VIO - Loop	8.3	3.8	4.9	4.2	12.8	5.0	2.3	1.9	1.6	1.2	2.8	2.1
IMU-Only	53.2	124.4	112.2	18.9	70.5	101.8	10	14.7	14.6	9.7	11.4	13.7
IMU-Only - Loop	49.0	41.9	132.7	34.7	115.5	47.0	9.9	11.8	15.3	9.7	14.4	10.7

knee extension. We intend to reconstruct these two models to visualize the different positions of the patella.

The bone models were bolted to a metal plate so that their position did not change during scanning. We encased the models in a gelatin medium to enable ultrasound scanning. Infrared markers were placed on the bone and on the scanner to enable tracking with an Optitrack system as a baseline.

A single cable was needed to connect the Realsense camera to the recording PC over USB. The same PC was used to record all sensor data including Optitrack and ultrasound images so that timestamps could be recorded from a common clock. However, the Clarius scanner only allows connection over Wi-Fi which introduces an unknown, non-fixed, time delay between ultrasound image acquisition and recording on the PC. This is a limitation of our setup, however, we found the time delay was not so severe as to prevent reconstruction.

B. Data Set Collection and Neural Network Training

A total of 150 scans were recorded by a human user. Each scan consisted of the user picking up the scanner and placing it at a different position on the target. In 130 of the scans, the user slowly panned across the target and, once at the end, would pan back to the initial position. This scanning motion was labeled a ‘2 sweeps’ motion. This was repeated for different starting positions and scanning directions with both 2 and 6 sweeps. The remaining 20 scans were similar except that rather than panning over the same section of bone, these scans moved laterally while panning so that the scanner covered the whole target over the course of the scan.

These scans were split 80:20 into training and validation data sets. We trained a single model from all the data and used this same model in all experiments for IMU-only estimation. We used the Adam optimizer with a fixed learning rate of 10^{-6} over 2000 epochs with the model minimizing validation error used. Training took approximately 6 hours on an NVIDIA GeForce RTX 3080 Ti GPU.

A second set of 6 scans were recorded for testing reconstruction. In this case the user also moved the scanner laterally while panning, attempted to reconstruct the whole bone and patella surface. Two scans were recorded for Bone 1, one for the x direction which we label as Scan Type A and one for the y direction which we label as Scan Type B. This was repeated for the Bone 2 model. Bone 3 scans are of the same model as Bone 1 but placed in a different position and orientation.

V. RESULTS AND DISCUSSION

In this section we present the experimental results and discuss the usability of this tool. We then provide a discussion of potential future works.

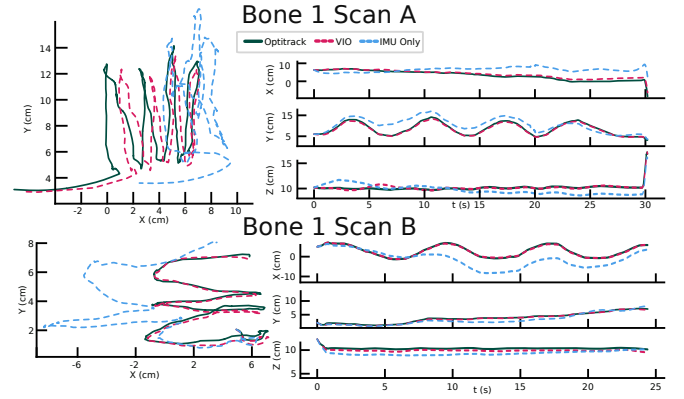


Fig. 5. Trajectories for the different algorithms while scanning Bone 1. Visual-inertial and learned inertial methods both use the loop closure constraint. In these plots, the trajectory is cropped to start when the scanner is in contact with the bone and cropped to end when scanning is finished. The first pose is aligned with ground truth in position and yaw, not roll or pitch.

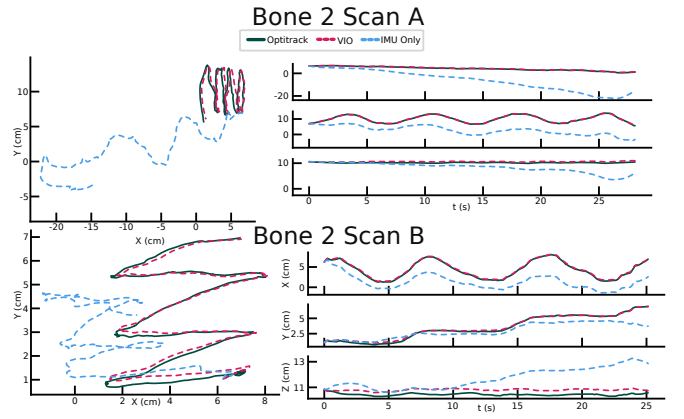


Fig. 6. Trajectories for the different algorithms while scanning Bone 2. Visual-inertial and learned inertial methods both use the loop closure constraint. Alignment is the same as for Fig. 5.

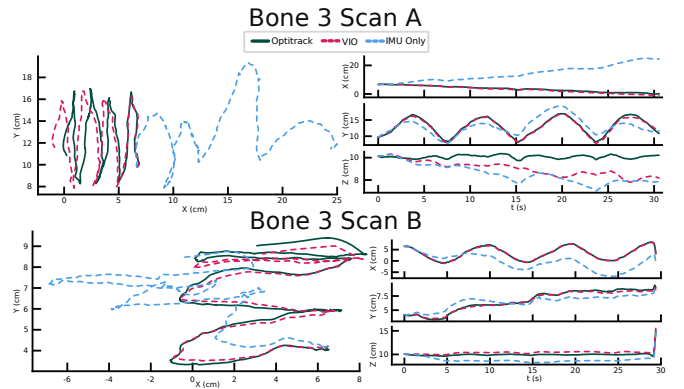


Fig. 7. Trajectories for the different algorithms while scanning Bone 3. Visual-inertial and learned inertial methods both use the loop closure constraint. Alignment is the same as for Fig. 5.

A. Tracking Results

In Table I we show the estimation error for each tracking method compared to the Optitrack trajectory — which we will treat as ground truth. We present the values for visual-inertial and deep learning inertial only with and without the

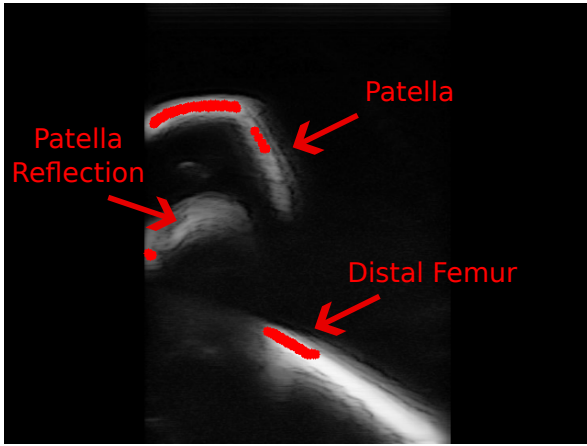


Fig. 8. Example segmented image for Bone 2. Bone segmentation struggles when multiple layers of bone exist.

loop closure constraint from the start and end prior factors.

We computed the translation components of both Absolute Pose Error (APE) and Relative Pose Error (RPE). APE represents the mean error of each trajectory compared to the ground truth trajectory at any point in time. RPE is the rate of drift over a given distance, for example in Table I we have computed RPE over 20 mm segments which means the value represents the average drift for every 20 mm traveled. More explanation of these error metrics is available in [24].

From inspecting Table I we can see that the addition of the loop closure generally reduces APE by a small amount, except Bone 2 Scan A, and has no effect on RPE. This is because the VIO method is already very accurate — drifting on average 1.8 mm for every 20 mm.

The IMU-only methods have much larger error than VIO. Adding the loop closure does reduce both RPE and APE for most trajectories; however, it remains higher than VIO. We also plot each of the trajectories for all six scans. Fig. 5 shows the Bone 1 scans while Fig. 6 plots the trajectories for Bone 2 and Fig. 7 shows the scans for Bone 3. At these scales there is a clear difference in magnitude of error between VIO and IMU. The VIO methods appear nearly perfectly aligned with Optitrack. The IMU-Only method has larger error. However, for 5 of the 6 trajectories the method largely followed the Optitrack trajectory with an offset. It is clear the network has learned the scanning motion and is providing useful information to the estimator. In the reconstruction step we attempt to correct for these errors using ICP.

B. Reconstruction Results

In Table II we present the mean and worse distance errors for each method when compared to the ground truth mesh model of the phantom bone. The average errors of the motion capture method and the visual-inertial method are 1.21 mm and 1.25 mm respectively (3% difference). The IMU-only method had an average error of 1.85 mm which was 35% higher compared to the motion capture method. This is due to the higher drift rates as shown in Table I.

For all methods, error is higher for Bone 2 which we believe is due to the position of the patella, directly above the

TABLE II
MEAN/WORST RECONSTRUCTION ERROR [MM]

Method	Bone 1		Bone 2		Bone 3	
Sweep Type	A	B	A	B	A	B
Optitrack	0.99 / 7.71	1.00 / 7.50	1.53 / 9.95	1.76 / 5.11	0.91 / 8.35	1.05 / 6.5
VIO	0.85 / 5.2	1.10 / 3.93	1.70 / 6.43	1.80 / 6.32	1.19 / 8.47	0.87 / 5.97
IMU-Only	1.8 / 7.4	1.51 / 7.67	2.56 / 10.03	1.98 / 7.93	1.70 / 10.3	1.55 / 6.52

distal femur, in the section of the bone which was scanned. Our ultrasound segmentation method is not currently capable of segmenting two layers of bone — one above the other. Therefore when scanning the Bone 2 patella, only partial segmentations are possible as shown in Fig. 8. Incomplete segmentations result in sparser point cloud sweeps which are more difficult to align using ICP.

In Figures 9 and 10, we show the resulting reconstructions for Bone 1 and Bone 2 for each scanning type. For each scan we show the resulting mesh colorized by distance error from the ground truth model. We have overlaid the surface reconstruction on top of the ground truth model of each bone. Another source of error present in the mesh reconstructions is a “flaring out” of the mesh at the edges. This is particularly noticeable for the Bone 1 Scan A with Optitrack in Fig. 9.

While the performance demonstrated here has slightly higher error than other published work; we reiterate that the proposed method does not require expensive external infrastructure. Mahfouz et al. [5] used EM trackers placed on the skin to track the scanner. They reported an average of 1.05 mm error from scanning the distal femur and proximal tibia of 15 cadaveric specimens. Kerr et al [3] used a robotic arm to perform the scanning. They reported an average error of 0.85 mm (using their best results) across two distal femurs (one human model and one bovine) and an average worst error of 9.06 mm.

Additionally, these other reconstruction methods can achieve wider coverage of the bone; however, they introduce other issues. Using an SSM as in [5] can introduce bias to the reconstruction. Additionally, submerging the sensor in water (such that there is no change in the propagation medium), as in [3], is impractical. In contrast, we use a general purpose scanner, held by a human, and no SSM. This makes our method more general and practical.

C. Patella Position Visualization

One application of this work is to develop a markerless handheld tool to visualize the patella. In Fig. 11 we take the best reconstruction for each estimation method and scanning type. We show the two reconstructions of the femur and patella side-by-side and make the true model STL transparent. Here we can see how the patella is reconstructed for each method.

The VIO method provides as good coverage as Optitrack with nearly the same error. Is it possible to identify the shape and position of the patella making this a viable tool for markerless handheld reconstruction.

The IMU-only method has less coverage and higher error; however it is also possible to identify the position of the patella. This shows promise of the IMU-only method, but is

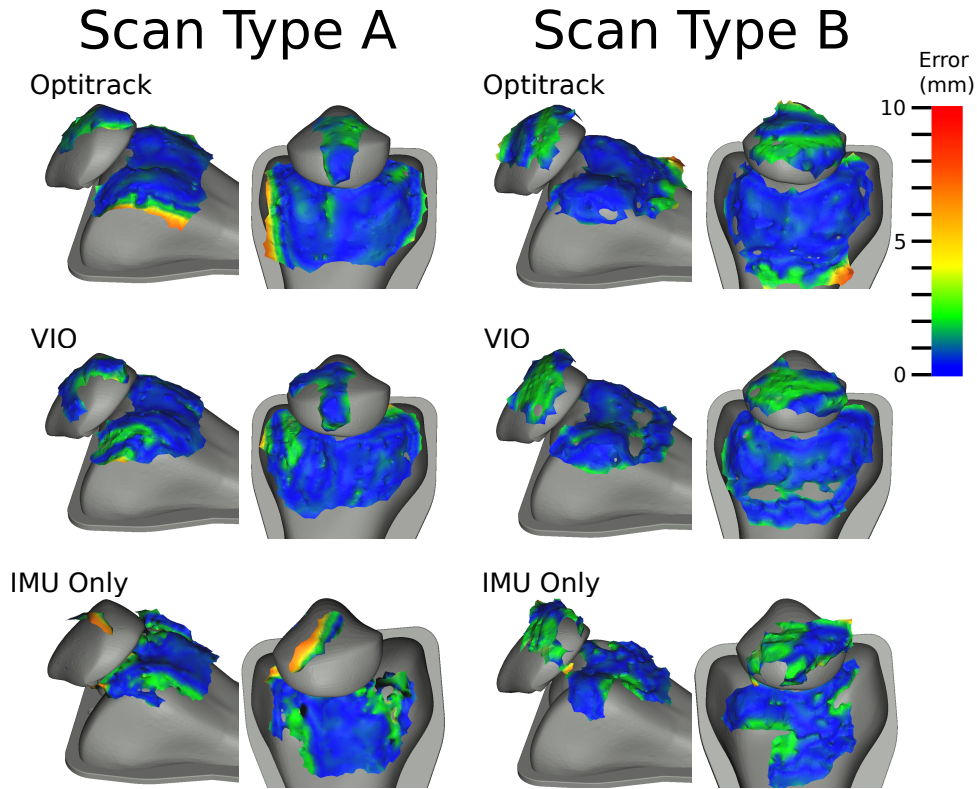


Fig. 9. Reconstructions of Bone 1 which is in 40° flexion. We show results for each estimation methods and for both scanning types A and B.

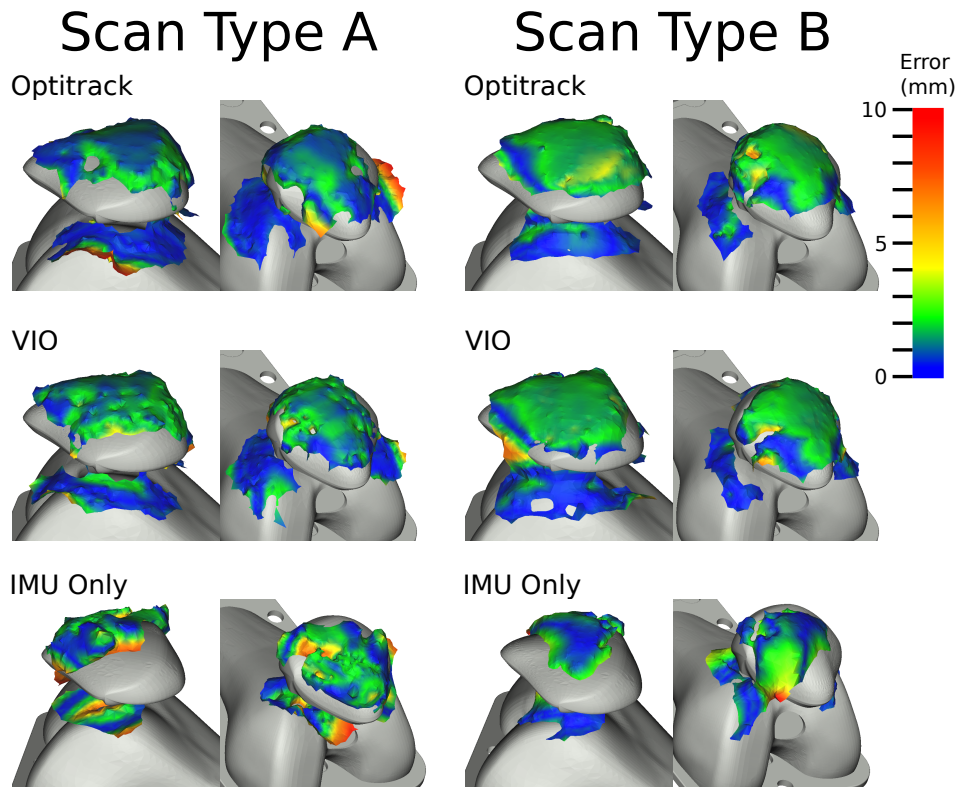


Fig. 10. Reconstructions of Bone 2 which is in full extension. The patella is now directly over the distal femur. We show results for each estimation methods and for both scanning types A and B.

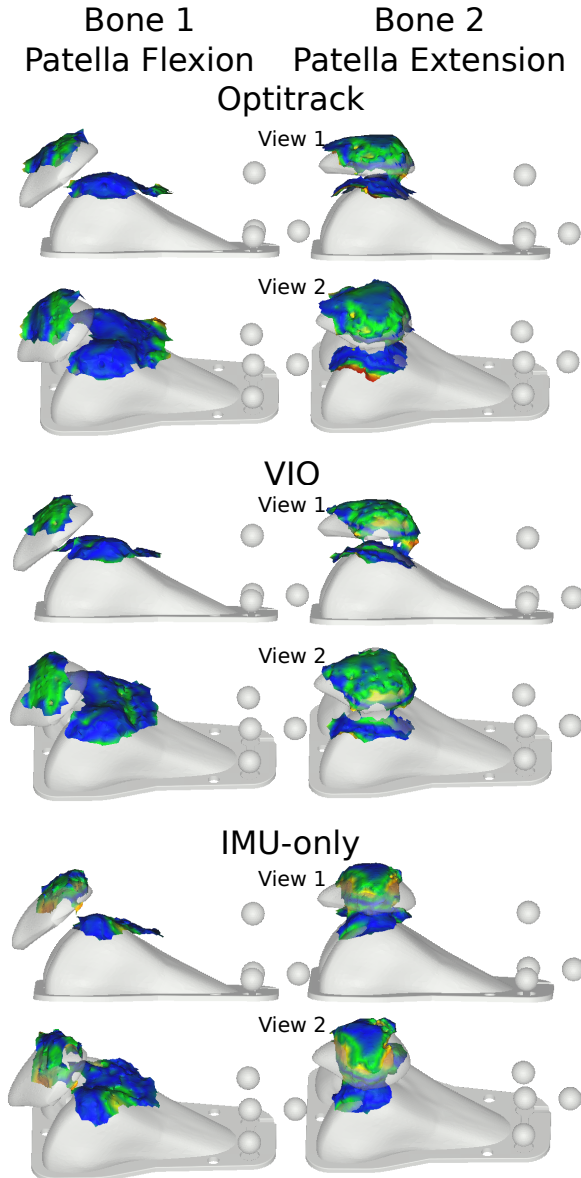


Fig. 11. Reconstructions of the patella in different positions.

likely not good enough on its own for markerless handheld reconstruction.

D. Future Work

In this work we have demonstrated that tracking the pose of a ultrasound probe with outward facing visual-inertial odometry is sufficiently accurate such that, when paired with ICP-based reconstruction, it can be used for markerless, handheld ultrasound reconstruction. Future work on developing this method should focus on the hardware, possibly integrating the camera with the ultrasound scanner.

The learning-based IMU tracking method shows promise, but likely has too high a drift rate for the method to be useful at present. It may, however, be possible to use more prior information to make the IMU-only method practical. Since the drift rate is larger, it may be better to accumulate point

clouds not per sweep, but instead over shorter distances. This would, however, make ICP-based matching more difficult, but this could be mitigated by using an SSM. While incorporating an SSM into the pipeline could introduce a bias into the reconstruction, if the goal is to estimate the position of the patella, assuming no abnormalities in the shape of the bone, this may be attractive.

We additionally investigated if the amount of available data significantly affected the results. We tested models trained with half as much data and found that RPE only increased by 2.3%. Using a quarter data increases error by 15%. Therefore, it is unlikely that more training data is needed to improve our IMU-only method. Instead, possibly new neural network architectures should be explored.

VI. CONCLUSION

In conclusion, this paper presents a novel approach for markerless 3D reconstruction of bones using a handheld ultrasound scanner. Our method combines recent robotic estimation techniques, including visual-inertial odometry, based on [14], and learning inertial motion tracking, based on [13], to estimate the pose of the ultrasound scanner. We have shown that when using VIO, it is possible to reconstruct the bone with error comparable to methods using external tracking infrastructure. This tool can then be used to visualize the position of the patella with different degrees of flexion. We also investigated the usability of a learning-based inertial tracking approach based on deep learning. We have shown the method has promise and that a neural network is able to learn the scanning motion; however, drift is likely too high to be immediately useful. Instead, we believe with the addition on an SSM, an IMU-only method for 3D reconstruction could be possible.

REFERENCES

- [1] L. N. Nazarian, "The top 10 reasons musculoskeletal sonography is an important complementary or alternative technique to mri," *American Journal of Roentgenology*, vol. 190, no. 6, pp. 1621–1626, 2008.
- [2] E. Świątek-Najwer, K. Otto, P. Krowicki, K. Krysztoforski, P. Keppler, and J. Kozak, "3d bone shape modelling basing on dataset recorded by ultrasound free-hand navigated probe," in *Information Technologies in Biomedicine, Volume 4*, E. Piętko, J. Kawa, and W. Wicławek, Eds. Cham: Springer International Publishing, 2014, pp. 45–56.
- [3] W. Kerr, P. Rowe, and S. G. Pierce, "Accurate 3d reconstruction of bony surfaces using ultrasonic synthetic aperture techniques for robotic knee arthroplasty," *Computerized Medical Imaging and Graphics*, vol. 58, pp. 23–32, 2017.
- [4] R. Jia, P. Monk, D. Murray, J. A. Noble, and S. Mellon, "CAT & MAUS: A novel system for true dynamic motion measurement of underlying bony structures with compensation for soft tissue movement," *J Biomech*, vol. 62, pp. 156–164, 2017.
- [5] M. R. Mahfouz, E. E. Abdel Fatah, J. M. Johnson, and R. D. Komistek, "A novel approach to 3d bone creation in minutes," *The Bone Joint Journal*, vol. 103-B, no. 6 Supple A, pp. 81–86, 2021.
- [6] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2017.
- [7] 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.
- [8] 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.

- [9] C. Campos, R. Elvira, J. J. Gómez, J. M. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM," *IEEE Transactions on Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.
- [10] J. Farrell, *Aided Navigation: GPS with High Rate Sensors*, 1st ed. USA: McGraw-Hill, Inc., 2008.
- [11] C. Chen, X. Lu, A. Markham, and N. Trigoni, "IONet: Learning to Cure the Curse of Drift in Inertial Odometry," *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*, pp. 6468–6476, 2018.
- [12] W. Liu, D. Caruso, E. Ilg, J. Dong, A. I. Mourikis, K. Daniilidis, *et al.*, "TLIO: Tight learned inertial odometry," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5653–5660, 2020.
- [13] R. Buchanan, M. Camurri, F. Dellaert, and M. Fallon, "Learning inertial odometry for dynamic legged robot state estimation," in *Proceedings of the 5th Conference on Robot Learning*, vol. 164, 2021, pp. 1575–1584.
- [14] D. Wisth, M. Camurri, and M. Fallon, "Vilens: Visual, inertial, lidar, and leg odometry for all-terrain legged robots," *IEEE Transactions on Robotics*, vol. 39, no. 1, pp. 309–326, 2022.
- [15] B. Busam, P. Ruhkamp, S. Virga, B. Lentes, J. Rackerseder, N. Navab, *et al.*, "Markerless inside-out tracking for interventional applications," 2018. [Online]. Available: <https://arxiv.org/abs/1804.01708>
- [16] F. Pomerleau, F. Colas, and R. Siegwart, "A review of point cloud registration algorithms for mobile robotics," *Foundations and Trends in Robotics*, vol. 4, pp. 1–104, 05 2015.
- [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] F. Dellaert, "Factor Graphs and GTSAM: A Hands-on Introduction," Georgia Institute of Technology, Tech. Rep. September, 2012. [Online]. Available: <https://smartech.gatech.edu/handle/1853/45226>
- [19] R. Jia, S. J. Mellon, S. Hansjee, A. P. Monk, D. W. Murray, and J. A. Noble, "Automatic bone segmentation in ultrasound images using local phase features and dynamic programming," in *IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, 2016, pp. 1005–1008.
- [20] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, *et al.*, "KinectFusion: Real-time dense surface mapping and tracking," in *10th IEEE International Symposium on Mixed and Augmented Reality*, 2011, pp. 127–136.
- [21] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson Surface Reconstruction," in *Symposium on Geometry Processing*, A. Sheffer and K. Polthier, Eds. The Eurographics Association, 2006.
- [22] 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.
- [23] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3d: A modern library for 3d data processing," 2018. [Online]. Available: <https://arxiv.org/abs/1801.09847>
- [24] R. Kümmerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, C. Stachniss, *et al.*, "On measuring the accuracy of slam algorithms," *Autonomous Robots*, vol. 27, no. 4, pp. 387–407, 2009.

In accordance with URKI policy on Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript version arising from this draft.

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	Markerless Handheld Ultrasound 3D Reconstruction of Bone using Visual-Inertial and Deep Inertial Odometry
Publication Status	☐ Published ☐ Accepted for Publication ☐ Submitted for Publication ☒ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Russell Buchanan, Shifan Jack Tu, Marco Camurri, Stephen J. Mellon, Maurice Fallon "Markerless Ultrasound Tracking for 3D Reconstruction of Bones using Deep Inertial Odometry" draft paper

Student Confirmation

Student Name:	Russell Buchanan		
Contribution to the Paper	- Adapted existing code for neural network and state estimator for new application of ultrasound reconstruction. - Wrote all code for mapping and reconstruction of bone surface. 50% contribution to data collection and experiments. - Wrote majority of draft paper.		
Signature		Date	February 24, 2023

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 contributions listed above are an accurate representation of the work Russell contributed			
Signature		Date	February 24, 2023

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

6.3 Discussion

In this chapter I presented work which applied inertial learning techniques to both legged robots and tracking an ultrasound scanner. I showed how this can reduce drift in proprioceptive and exteroceptive odometry systems for legged robots and how this can be used for IMU-only estimation to enable infrastructure-free 3D reconstruction from ultrasound.

In Section 6.1, the reduction in legged robot odometry drift was observed when the learned displacement measurements were added to the baseline EKF estimator Pronto [21]. However, the improved EKF performance was still worse in experiments on rigid terrain compared to the default proprioceptive estimator on the ANYmal robot as shown in Table 1 of the paper. We attribute this to the fine-tuning by the company for their specific robot, whereas we used a general purpose, open source estimator and added learned displacements.

Additionally, all methods experienced increased drift when deployed in field experiments. This likely due to the increased challenges in the mine which included more slippery terrain and degraded vision. It is difficult to compare results between the lab and field experiments, even for the proprioceptive methods, because of the different methods for generating ground truth. In the lab experiments Vicon motion tracking was used; however, in the mine, a lidar on the robot was used to localise in a high resolution map of the mine.

In Section 6.2, inertial learning enabled IMU-only estimation resulting in 3D reconstruction of a bone phantom. However, we found that the method was not accurate enough to have the desired less than 1 mm error. However, VIO was found to be sufficiently accurate. The VIO method involved a camera mounted on the ultrasound scanner facing upwards and tracking features on the ceiling using FAST corner detection [120] with KLT feature tracking [138].

The same visual feature tracking pipeline is also used in legged robot experiments in Section 6.1, but in the ultrasound experiments we found that the tracking was much more accurate, enabling <1 mm reconstruction error. I believe the reasons for this are largely that the ultrasound mounted camera is virtually always looking at

the same region of the ceiling. This means it can track the same features throughout the entire scanning processes. On the other hand, the legged robot moved through different rooms and would have to continuously detect new features and forget old features. Additionally, for ultrasound scanning, the room was well lit, the camera experienced minimal motion blur and the start and end pose of the scanner were known *a priori*, all of which contributed to much better odometry performance.

In this chapter I apply inertial learning to very different domains: legged robots and handheld scanners. But there are many other ways in which deep learning can be used for state estimation, beyond learning displacement measurements. In Section 8.1.2 I cover several ideas and challenges to future work in this area.

7

Learning the Bias Evolution of Inertial Sensors

The previous chapters mainly focus on proprioceptive state estimation, either using contacts for localisation or learning inertial motion models. However, visual-inertial odometry remains a more accurate and more widely used method.

In VIO systems, when visual tracking fails, the system must rely purely on IMU integration which will rapidly accumulate error. In Section 6.1, this was mitigated by fusing learned displacement measurements from a neural network, however this method is restricted to motions present in the training dataset.

The main reason IMU-only estimation using mechanisation requires additional, corrective sensing (either visual or learned motions) is because of the accumulation of noise as explained in Section 2.3. One significant source of noise are the IMU biases which are constantly changing. Using additional sensing allows the biases to become observable. If there was a way to observe the biases directly it would significantly improve any state estimation method which uses IMUs.

In this chapter I propose a new method which uses deep learning to estimate IMU biases directly from IMU measurements and past biases. In the attached paper, it is shown that a neural network can learn the evolution process of the biases of a specific IMU rather than assuming random walk. As a result, this method

is *device specific, not locomotion specific*, unlike inertial motion model learning approaches, and does not require periodic motion.

7.1 Deep IMU Bias Inference for Robust Visual-Inertial Odometry with Factor Graphs

The following article was published in IEEE Robotics and Automation Letters [13].

Deep IMU Bias Inference for Robust Visual-Inertial Odometry with Factor Graphs

Russell Buchanan¹, Varun Agrawal², Marco Camurri¹, Frank Dellaert², Maurice Fallon¹

Abstract—Visual Inertial Odometry (VIO) is one of the most established state estimation methods for mobile platforms. However, when visual tracking fails, VIO algorithms quickly diverge due to rapid error accumulation during inertial data integration. This error is typically modeled as a combination of additive Gaussian noise and a slowly changing bias which evolves as a random walk. In this work, we propose to train a neural network to learn the true bias evolution. We implement and compare two common sequential deep learning architectures: LSTMs and Transformers. Our approach follows from recent learning-based inertial estimators, but, instead of learning a motion model, we target IMU bias explicitly, which allows us to generalize to locomotion patterns unseen in training. We show that our proposed method improves state estimation in visually challenging situations across a wide range of motions by quadrupedal robots, walking humans, and drones. Our experiments show an average 15% reduction in drift rate, with much larger reductions when there is total vision failure. Importantly, we also demonstrate that models trained with one locomotion pattern (human walking) can be applied to another (quadruped robot trotting) without retraining.

Index Terms—Visual-Inertial SLAM; Sensor Fusion; Deep Learning Methods

I. INTRODUCTION

State estimation for lightweight, mobile systems is a fundamental problem in robotics. Visual Inertial Odometry (VIO) is a common solution due to the small size and low cost of cameras and IMUs. The main weak-point of VIO is that when visual feature tracking fails, only the Inertial Measurement Unit (IMU) can be used.

Inexpensive and miniaturized micro-electromechanical systems (MEMS) IMUs have become ubiquitous in robotics and pedestrian tracking [1]–[4], however IMU-only state estimation is severely affected by drift. This drift is the result of the accumulation of various errors in IMU integration collectively modeled as a combination of additive zero-mean Gaussian noise and a slowly changing bias. As a result, estimation which relies



Fig. 1. Two of the platforms used in this work. **Left:** handheld device including a SevenSense AlphaSense visual-inertial sensor. **Right:** the same device mounted on a Boston Dynamics Spot quadruped. An IMU bias model from a neural network trained on handheld data was used to infer the biases on the Spot dataset without any retraining — despite the significantly different motion pattern. The lidar was not used in these experiments.

purely on IMU measurement integration is not feasible for more than a few seconds due to explosive accumulation of drift.

VIO is effective so long as visual features are available because they constrain the drift of IMU integration [5] and estimate the biases. When visual tracking fails completely, the system relies purely on IMU integration. When there is error in visual tracking, the estimator may update the bias estimates to make sense of the data. In other words, the estimated bias is not necessarily related to the underlying physical process, but a quantity that minimizes the residuals. If there were another way to infer the IMU biases, these problems could be mitigated.

Most VIO works have focused on improving visual tracking, however in this work we take a novel approach of *improving IMU bias modeling*. We propose a new method which uses deep learning to estimate IMU biases directly from IMU measurements and past biases. We train a neural network to learn the evolution process of the biases of a specific IMU rather than assuming Brownian Motion. As a result, our method is *device specific, not locomotion specific*, unlike similar IMU learning approaches, and does not require periodic motion. Our contributions can be listed as follows:

- A neural network capable of estimating IMU biases from a history of measurements and biases. To the best of the authors' knowledge, this is the first method capable of explicitly inferring the bias evolution of an IMU.
- A performance comparison of two different network implementations (LSTM and Transformer) and their integration as unary factors into a state estimator based on factor graphs, for improved estimation in visually challenging scenarios.
- Real-world experiments on three different platforms with different types of motions: pedestrian handheld,

Manuscript received: July, 13, 2022; Revised October, 7, 2022; Accepted November, 3, 2022. This paper was recommended for publication by Editor Sven Behnke upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the EPSRC ORCA Robotics Hub (EP/R026173/1), the EU H2020 Project THING (Grant ID 780883) and a Royal Society University Research Fellowship (Fallon). For the purpose of Open Access, the author has applied a CC BY public copyright licence to any Author Accepted Manuscript (AAM) version arising from this submission.

¹ Oxford Robotics Institute, University of Oxford, UK {russell, mcamurri, mfallon}@robots.ox.ac.uk

² Institute for Robotics and Intelligent Machines, College of Computing, Georgia Institute of Technology {varunagrwal, dellaert}@cc.gatech.edu

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

quadrupedal robot and drones. To the best of the authors' knowledge, this is the first work that demonstrates an IMU learned model trained on one locomotion modality and tested on another (handheld to quadrupedal).

An additional minor contribution is the development of a ROS compatible, open-source tool for calibrating an IMU using the Allan Variance method¹.

II. RELATED WORK

In this section, we summarize the growing field of inertial learning from which our work follows. In Section II-A, we discuss methods which learn motion models from inertial data, while Section II-B covers methods which learn IMU noise models primarily for drone state estimation.

A. Learning Inertial Motion Models

Recent works have trained neural networks with IMU data to learn motion models (typically of pedestrians) and to output estimates of velocity directly from IMU measurements.

The first data driven Inertial Navigation System (INS) was IoNet by Chen et al. [6] which inferred 2D displacement and orientation change from buffered IMU data. Later, Herath et al. [7] proposed RoNIN, a similar network which inferred 2D velocity and orientation from raw IMU data. Liu et al. [8] proposed TLIO, a method to infer 3D motion and estimate high-fidelity trajectories in a filtering framework. They used an Extended Kalman Filter (EKF) to estimate the full 6 DoF state. Process updates were performed by traditional IMU mechanization while measurement updates from the network were used as relative position measurements. This allowed the filter to implicitly estimate IMU biases.

In our prior work [9], we adapted the approach of [8] to incorporate additional sensors such as cameras, lidar or legged robot kinematics in a factor graph. However, this method, like the ones discussed above, was based on learning a motion model and was therefore susceptible to failure if applied outside of the training domain. For example, in Fig. 2 we show results of TLIO which was trained with handheld walking data on flat ground. As shown in Fig. 2a), the velocity estimates are reasonable for IMU-only odometry. In Fig. 2b), we applied the same network to stair climbing, which was not present in the training set and as a result, the z velocity estimation fails completely. In Fig. 2c) we tested the network with a quadruped and the model fails completely. This motivates the need for methods which are more IMU specific rather than locomotion specific.

B. Learning IMU Noise Models

A different approach is to learn the IMU noise model. In this case, the IMU measurements are passed through a network trained to “denoise” them and output an estimate of the perfect IMU measurements. These denoised signals can then be used directly as input to a VIO pipeline.

Brossard et al. [10] used a Convolutional Neural Network (CNN) to denoise a gyroscope using the output to correct for

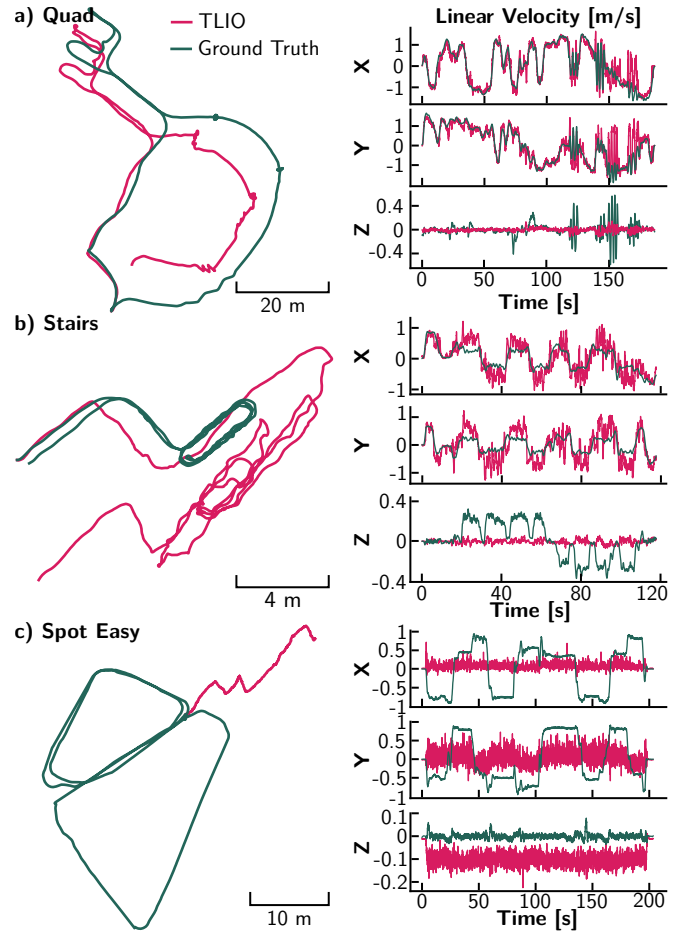


Fig. 2. **Top:** TLIO [8] trained on handheld data on flat ground, tested on flat ground. **Middle:** same network tested on a sequence in which the person walks up and down stairs. Because stair climbing was not present in the training set, inference fails significantly (Note z velocity estimates). **Bottom:** when applied to a quadrupedal robot, because of the completely different locomotion modality, TLIO is unable to estimate velocity.

the true angular velocity. The CNN used dilated convolutions to increase the temporal coverage over longer sequences. Zhang et al. [11] trained a Recurrent Neural Network (RNN) to denoise both gyroscope and accelerometer measurements. Their network used IMU measurements as input to estimate a corrected measurement which, when integrated, reduced pose error. This eliminated the need for the model to learn the underlying mechanization equations. Similarly, Steinbrener et al. [12] performed denoising on IMU measurements, comparing LSTM and Transformer architectures, finding the LSTM to be more effective.

The main drawback of these methods is that they do not distinguish between different noise sources and, as a result, it is not clear what the network has learned to remove from the IMU measurements. For example, most results were primarily demonstrated on drones which, when flying, introduce high frequency vibrations affecting the IMU measurements. It is unclear if the noise characteristics being estimated were caused by the vibrations, the internal state of the IMU (i.e. the true bias), the motion of the drone, or a combination of the above. By explicitly modeling the IMU bias, in this paper we seek a

¹https://github.com/ori-drs/allan_variance_ros

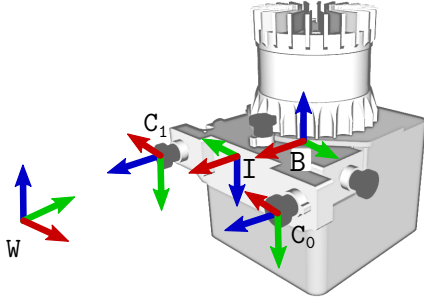


Fig. 3. Reference frames convention for the handheld device. The world frame W is a fixed frame, while the base B , camera optical C and IMU I frames are attached to the moving device.

motion independent method with more explainable outputs.

III. PROBLEM STATEMENT

We aim to estimate the trajectory of a mobile platform equipped with an IMU and a stereo camera using sliding window based smoothing. The relevant reference frames of the platform are shown in Fig. 3 and includes an Earth-fixed world frame W , the platform-fixed base frame B , left and right camera frames C_0, C_1 , and IMU sensor frame I .

A. Definition of State Vector and Measurements

The state of the platform at time t_i is defined as:

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

where: $\mathbf{R}_i = \mathbf{R}_{WB}(t_i)$ is the orientation of B with respect to W , $\mathbf{p}_i = {}_W\mathbf{p}_{WB}(t_i)$ is the position, $\mathbf{v}_i = {}_B\mathbf{v}_{WB}(t_i)$ is the linear velocity, and $\mathbf{b}_i^g = {}_I\mathbf{b}^g(t_i)$, $\mathbf{b}_i^a = {}_I\mathbf{b}^a(t_i)$ are the IMU gyroscope and accelerometer biases, respectively.

We indicate the set of all states in the window as $\mathcal{X}_k = \{\mathbf{x}_i\}_{i \in K_k}$ where K_k are all the keyframe indices up to t_k . Similarly, the measurements within the window are $\mathcal{Z}_k = \{\mathcal{I}_{ij}, C_i\}_{i,j \in K_k}$, where $\mathcal{I}_{ij}$ are the IMU measurements between two camera keyframes (with $i = j - 1$), while C_i include the stereo image pairs at time t_i .

B. Maximum-A-Posteriori (MAP) Estimation

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

$$\mathcal{X}_k^* = \arg \max_{\mathcal{X}_k} p(\mathcal{X}_k | \mathcal{Z}_k) \propto p(\mathcal{X}_0) p(\mathcal{Z}_k | \mathcal{X}_k) \quad (2)$$

The measurements are formulated as conditionally independent and corrupted by white Gaussian noise. Therefore, (2) can be expressed as the following least squares minimization:

$$\mathcal{X}_k^* = \arg \min_{\mathcal{X}_k} \|\mathbf{r}_0\|_{\Sigma_0}^2 + \sum_{i \in K_k} \left(\|\mathbf{r}_{\mathcal{I}_{ij}}\|_{\Sigma_{\mathcal{I}_{ij}}}^2 + \sum_{\ell \in M_i} \|\mathbf{r}_{\mathbf{x}_i, \mathbf{m}_\ell}\|_{\Sigma_{\mathbf{x}_i, \mathbf{m}_\ell}}^2 + \|\mathbf{r}_{\mathbf{b}_j^g}\|_{\Sigma_{\mathbf{b}^g}}^2 + \|\mathbf{r}_{\mathbf{b}_j^a}\|_{\Sigma_{\mathbf{b}^a}}^2 \right) \quad (3)$$

where each term is the residual associated to a measurement type, weighted by the inverse of its covariance matrix, and will be detailed in Section IV.

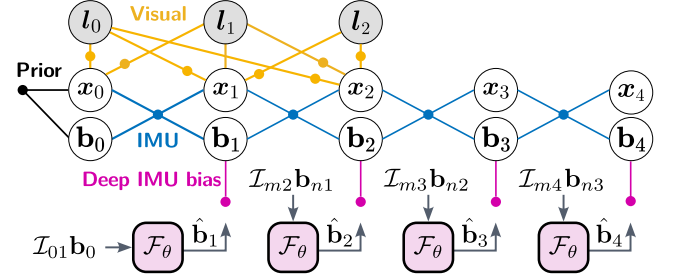


Fig. 4. Proposed factor graph framework with learned IMU estimates. A neural network $\mathcal{F}_\theta$ trained on ground truth IMU biases takes a window of past IMU measurements $\mathcal{I}_{m,j}$ and previous bias estimates $\mathbf{b}_{ni}$ from the optimizer and outputs the current bias estimate $\hat{\mathbf{b}}_j$. This is integrated into the factor graph as unary factor (magenta).

IV. FACTOR GRAPH FORMULATION

A factor graph can be used to graphically represent (3) as in Fig. 4. With slight abuse of notation, the IMU biases are shown separately from the state nodes $\mathbf{x}_i$ to highlight our contribution. In addition to the prior factors (black), the graph includes: IMU preintegration (blue), visual tracking (yellow) and our novel deep IMU bias (magenta) factors. For convenience, the first two are briefly reported in this section, while the last one is detailed in Section V.

A. Preintegrated IMU Factors

We follow the standard manner of IMU measurement integration from [5] to constrain the pose, velocity, and biases between two consecutive nodes of the graph. The residual has the following form:

$$\mathbf{r}_{\mathcal{I}_{ij}} = [\mathbf{r}_{\Delta\mathbf{R}_{ij}}^T, \mathbf{r}_{\Delta\mathbf{v}_{ij}}^T, \mathbf{r}_{\Delta\mathbf{p}_{ij}}^T, \mathbf{r}_{\mathbf{b}_i^g}^T, \mathbf{r}_{\mathbf{b}_j^g}^T]^T \quad (4)$$

For a detailed definition of the above residuals, see [5].

B. Stereo Landmark Factors

A visual landmark in Euclidean space $\mathbf{m}_\ell \in \mathbb{R}^3$ is projected onto the image plane by the function $\pi : \text{SE}(3) \times \mathbb{R}^3 \mapsto \mathbb{R}^2$, given the platform pose $\mathbf{T}_i = \{\mathbf{p}_i, \mathbf{R}_i\} \in \text{SE}(3)$. Given a landmark $\mathbf{m}_\ell$ and its coordinates $(u_\ell, v_\ell) \in \mathbb{R}^2$ on the image plane, the residual at state is computed as:

$$\mathbf{r}_{\mathbf{x}_i, \mathbf{m}_\ell} = \begin{pmatrix} \pi_u^L(\mathbf{T}_i, \mathbf{m}_\ell) - u_{i,\ell}^L \\ \pi_u^R(\mathbf{T}_i, \mathbf{m}_\ell) - u_{i,\ell}^R \\ \pi_v(\mathbf{T}_i, \mathbf{m}_\ell) - v_{i,\ell} \end{pmatrix} \quad (5)$$

where (u^L, v) , (u^R, v) are the pixel locations of the landmark. Σ_m is computed using an uncertainty of 0.25 px with the Dynamic Covariance Scaling robust loss function as in [13].

V. DEEP IMU BIAS FACTORS

A MEMS IMU measures its proper acceleration (e.g., equal to Earth's gravity at rest) $\hat{\mathbf{a}} \in \mathbb{R}^3$ and rotational rate $\hat{\boldsymbol{\omega}} \in \mathbb{R}^3$. The absolute linear acceleration $\mathbf{a}$ (e.g., null at rest) and the rotation rate $\boldsymbol{\omega}$ of a body expressed in an Earth-fixed coordinate frame W can be recovered as follows:

$$\begin{aligned} \hat{\mathbf{a}}(t) &= \mathbf{R}_{WI}^T(t)(\mathbf{a}(t) - {}_W\mathbf{g}) + \mathbf{b}^a(t) + \boldsymbol{\eta}^a(t) \\ \hat{\boldsymbol{\omega}}(t) &= \boldsymbol{\omega}(t) + \mathbf{b}^g(t) + \boldsymbol{\eta}^g(t) \end{aligned} \quad (6)$$

Where $\mathbf{R}_{\mathbf{W}} \in \text{SO}(3)$ is the absolute orientation of the IMU and ${}_{\mathbf{W}}\mathbf{g}$ is the acceleration due to gravity expressed in $\mathbf{W}$. The quantities $\boldsymbol{\eta}^a$ and $\boldsymbol{\eta}^g$ represent additive noise present in all IMUs and is modeled as zero-mean Gaussian distributions.

A. Bias Noise Model

The IMU biases $\mathbf{b}^a$ and $\mathbf{b}^g$ are due to physical properties of the IMU. They change with each power cycle and continue to change slowly during operation [14]. Typically, their evolution is modeled as a Brownian noise process, whose derivatives are sampled from zero-mean Gaussian distributions:

$$\dot{\mathbf{b}}^a(t) = \boldsymbol{\eta}^{ba} \quad \dot{\mathbf{b}}^g(t) = \boldsymbol{\eta}^{bg} \quad (7)$$

This can be re-written in discrete time by integrating between two time steps $[t_i, t_j]$:

$$\mathbf{b}_j^a = \mathbf{b}_i^a + \boldsymbol{\eta}^{bad} \quad \mathbf{b}_j^g = \mathbf{b}_i^g + \boldsymbol{\eta}^{bgd} \quad (8)$$

Where $\boldsymbol{\eta}^{bad}$ and $\boldsymbol{\eta}^{bgd}$ are discrete zero mean Gaussian distributions with covariance $\boldsymbol{\Sigma}^{bad}$ and $\boldsymbol{\Sigma}^{bgd}$ respectively [5].

B. Learning the Bias Process Model

The Brownian noise model for IMU biases in (8) is an approximation which does not hold for long periods of time. In practice, the true underlying dynamics are a highly nonlinear function depending on vibrations, impacts, and physical properties of the device [14]. Thus, modeling the IMU dynamics is a problem that lends itself to deep learning as it allows us to approximate highly nonlinear functions. To this end, instead of (8), we propose a deep neural network $\mathcal{F}$ with parameters θ defined as follows:

$$\mathcal{F}_\theta : (\mathcal{I}_{mj}, \mathbf{b}_{ni}^a, \mathbf{b}_{ni}^g) \mapsto (\hat{\mathbf{b}}_j^a, \hat{\mathbf{b}}_j^g) \quad (9)$$

where the inputs are a buffer of IMU measurements $\mathcal{I}_{mj}$ between times t_m and t_j , and the previous bias values $\mathbf{b}_{ni}^a, \mathbf{b}_{ni}^g$ between t_n and t_i . The network outputs $(\hat{\mathbf{b}}_j^a, \hat{\mathbf{b}}_j^g)$ are the estimates of the IMU bias value at time t_j . Without loss of generality, in this section we assume the output of the network is generated at the camera keyframe rate.

C. Deep IMU Bias Factors

We incorporate the bias estimates from the network into a factor graph-based state estimator. The estimates are modeled as unary factors on the bias state, as shown in Fig. 4. The last two residuals of (3), $\mathbf{r}_{\mathbf{b}_j^a}$ and $\mathbf{r}_{\mathbf{b}_j^g}$ correspond to:

$$\mathbf{r}_{\mathbf{b}_j^a} = \mathbf{b}_j^a - \hat{\mathbf{b}}_j^a \quad \mathbf{r}_{\mathbf{b}_j^g} = \mathbf{b}_j^g - \hat{\mathbf{b}}_j^g \quad (10)$$

The covariances $\hat{\boldsymbol{\Sigma}}_{\mathbf{b}^a}$ and $\hat{\boldsymbol{\Sigma}}_{\mathbf{b}^g}$ were tuned to fixed values for our experiments, with $\hat{\boldsymbol{\Sigma}}_{\mathbf{b}^a} = \mathbf{I}_3 \cdot 2.5e^{-3}$ and $\hat{\boldsymbol{\Sigma}}_{\mathbf{b}^g} = \mathbf{I}_3 \cdot 2.5e^{-5}$. In future work, we intend to train the networks to provide measurement uncertainty estimates as in [8], [9].

VI. NETWORK ARCHITECTURE

Since our learning task involves sequential data, two common network architectures are well suited when implementing (9). The first one is based on Long Short-Term Memory (LSTM) [15] which is lightweight (see Fig. 5) and the second is based on Transformers [16] which have been shown to have better performance for longer-term sequences (see Fig. 6).

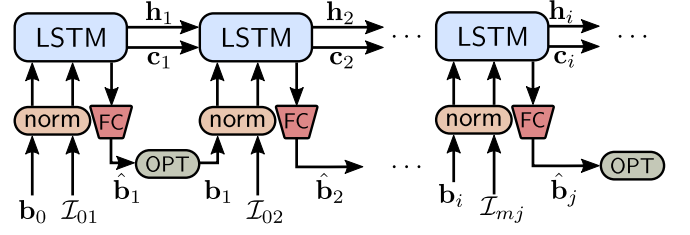


Fig. 5. LSTM architecture. A IMU data window $\mathcal{I}_{mj}$ of size w (with $m = j - w$) and the previous bias $\mathbf{b}_i$ are first normalized then passed to the LSTM. The hidden state is preserved for the next inference step and the output is passed through a fully connected layer to predict a bias.

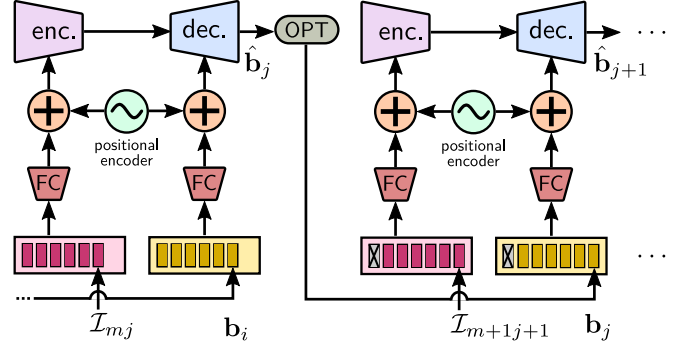


Fig. 6. Transformer architecture. A history of l IMU windows $\mathcal{I}_{mj}$ and biases $\mathbf{b}_i$ are concatenated and summed with a positional encoding before being passed to the Transformer.

A. LSTM

The input to the LSTM is a single window of size w of IMU measurements $\mathcal{I}_{mj}$ (with $m = j - w$) and the previous bias estimate $\mathbf{b}_i$ (i.e., $t_n = t_i$), which comes from the factor graph optimizer. This exposes the network to bias estimates resulting from the fusion of additional sensors. These are normalized then passed to the LSTM with states $\mathbf{h}_i$ and $\mathbf{c}_i$ which are preserved for the next bias estimate. In this way, the LSTM receives batches of w IMU measurements while the memory can observe the bias evolution over time. One limitation to this method is that, over a long trajectory, the LSTM will eventually forget information from old inputs.

B. Transformer

The Transformer input is a history of l windows of IMU measurements and biases (i.e., $m = j - l \cdot w, n = l$). Similar to the LSTM, biases added to the history come from the estimator's optimizer. A history of information allows the Transformer attention mechanism to recall older information.

C. Loss Function

We use the Mean Square Error (MSE) as a loss function:

$$\mathcal{L}(\mathbf{b}, \hat{\mathbf{b}}) = \frac{1}{n} \sum_{k=1}^n \|\mathbf{b}_k - \hat{\mathbf{b}}_k\|^2 \quad (11)$$

where two separate instances of the network are trained with (11) for accelerometer and gyroscope biases, respectively.

TABLE I
DATASETS USED FOR EXPERIMENTS.

Dataset	Duration [s]	Length [m]	IMU	Locom.
NCD-Multi	3163	4512	BMI085*	Handheld
Spot	515	399	BMI085*	Quadruped
EuRoC	1349	893	ADIS16448	Drone

*Same IMU device.

VII. NETWORK IMPLEMENTATION

A. Datasets

The datasets used for training and experiments are listed in Table I. The first is the Newer College Multi-Camera Dataset (NCD-Multi) [17] which uses an AlphaSense inertial and multi-camera sensor (see Fig. 1), although we only use the front facing stereo cameras. NCD-Multi was collected by a human operator walking while carrying the device inside New College, Oxford. An additional non-public sequence was collected using the same device in a limestone mine [18].

The second dataset, Spot, was recorded to demonstrate generalization to motions patterns unseen in training. We placed the same sensor payload used in NCD-Multi on a quadruped robot (see Fig. 1, right) and recorded two sequences only for testing.

Finally, we present results with the public EuRoC dataset [19]. We included this dataset to show results alongside similar methods and to demonstrate application to a very different platform: a drone.

B. Training

We trained one LSTM and one Transformer each for the public datasets (NCD-Multi and EuRoC), for a total of four models. For each dataset, we selected hard sequences for testing (see VIII-A) and the remaining were split 75:25 for training and validation. The Spot dataset was only used for testing because, as detailed in Section VIII, we demonstrate generalization to different locomotion modalities by training a model on NCD-Multi and testing it on the Spot dataset.

For training, we used teacher forcing [20], providing the network with ground truth biases. The ground truth biases for NCD-Multi were estimated using the lidar and a high resolution 3D model of the environment while the EuRoC dataset provides IMU bias estimates. We added noise to the ground truth biases which was sampled from a zero-mean Gaussian distribution with covariances Σ^{bad} and Σ^{bgd} estimated from Allan Variance analysis.

As in [8], we rotate the IMU measurement windows $\mathcal{I}_{m,j}$ into a gravity aligned frame. This prevents the effect of gravity from significantly changing the apparent accelerometer biases.

We used the Adam optimizer with a learning rate of 10^{-5} and batch size of 32. Training lasted 200 epochs for both LSTM and Transformer with the model minimizing validation error used. Both networks take ~ 2 h to train on a desktop computer with one Nvidia Titan X with 12 GB of memory.

C. Networks Models

1) *LSTM*: We used a 2-layer single direction LSTM with hidden state size of 256. We found that a larger network



Fig. 7. Visual challenges in the NCD-Multi Dataset. **Top Row**: Images from NCD-Multi Stairs sequence with door opening (obscuring the cameras) shown on the left. On the right is a tight space about 1.5 m wide and a mirror. **Bottom Row**: Images from NCD-Multi Mine sequence with darkness and the challenge of camera being held up close to a wall.

provided no improvement in performance while a lighter-weight LSTM is capable of running online with several inferences per second on a standard laptop with low-grade GPU. After testing different options, we found that 1 s of IMU data (i.e., $w = 10$ for 100 Hz IMU) and an inference rate of 2 Hz (50% data overlap between consecutive inferences) to give the best performance and used these settings for all experiments.

2) *Transformer*: We used an 8 headed Transformer with 2 encoder and decoder layers and an embedding size of 512. Similar to LSTM, we used 1 s IMU windows with a maximum history size of $l = 100$. This was based on analysis of the Allan Variance plots which found the IMU bias instability to dominate noise generally around 100 s sampling times. Inference was kept at 1 Hz since using a history of IMU windows makes overlapping of input data unnecessary.

VIII. EXPERIMENTAL RESULTS

In this section, we describe the experiments and results of the proposed state estimator. We report relative position and orientation errors as described in [21]. We compare our method which uses bias predictions as in Fig. 4 to our baseline method, which consists of the same factor graph but without the unary factor described in Section IV.

A. Handheld Experiments

There are many common situations or environments in which visual odometry systems degrade or even fully fail such as door-opening, narrow spaces or fast rotations. We selected several sequences from NCD-Multi which exhibit these challenges to test our method. These sequences were only used for testing and contained situations unseen in training.

1) *NCD-Multi Stairs*: This sequence included narrow spaces, mirrors, door opening and large rotations as shown in Fig. 7. As a result, the baseline stereo visual odometry struggled and lost track of all features several times.

TABLE II
10m RELATIVE TRANSLATION/ROTATION ERROR [M]/[°]

Experiment	VIO baseline	LSTM	Transformer
NCD-Multi Stairs*	0.23 / 3.22	0.19 / 2.88	0.17 / 2.70
NCD-Multi Mine	0.46 / 2.59	0.33 / 1.71	0.42 / 2.26
NCD-Multi Quad	0.35 / 1.38	0.28 / 1.36	0.25 / 1.29
Spot Easy	0.31 / 1.21	0.27 / 1.02	0.31 / 0.88
Spot Hard	0.38 / 1.57	0.34 / 1.42	0.30 / 1.42

*5m RPE due to shorter total distance traveled.

We show in Fig. 8 that the learned bias predictions reduced relative pose errors. This is most clearly seen in the magnified area where the door opening caused a significant drop in tracked visual features. Our proposed approach improves upon pure IMU-only mechanization in these situations where visual feature tracking fails.

2) *NCD-Multi Mine*: The Mine sequence was recorded in an abandoned limestone mine in Corsham, UK. It was dark and dusty and the handheld platform was intentionally held up very close to walls as shown in Fig. 7 on several occasions. This was also the longest trajectory.

The results can be seen in Fig. 9, where we highlight two particular sections in which the baseline VIO became unstable. In one section the camera faced a wall for several seconds and another involved large rotations in the dark. We see that by adding the bias prediction, trajectories are smoother and have lower error overall.

3) *NCD-Multi Quad*: This sequence is long and in a large-scale open space and includes several dynamic motions when the handheld platform was shaken. Our method reduces error compared to the baseline. A summary of these results and for the other NCD-Multi sequences is provided in Table II.

B. Quadruped Robot Experiments

For the Spot dataset, the same handheld sensor used in the NCD-Multi dataset was mounted on a Boston Dynamics Spot (See Fig. 1) which was then teleoperated around a courtyard in two separate trajectories. The Easy trajectory is entirely on flat ground in good lighting conditions while the Hard trajectory includes transitions from well-lit to shady areas and walking on small ramps.

For these experiments, both the LSTM and Transformer were trained *using only handheld data*. There was no fine-tuning for the quadrupedal robot. The information used for training and testing was therefore exactly the same as in Fig. 2. Numerical results are provided in Table II. This demonstrates that our method is independent from the locomotion modality and that the model can generalize across different modalities.

C. Drone Experiments

Two similar works to ours are Zhang et al. [11] and Brosard et al [10]. Zhang et al. used an LSTM to denoise accelerometer and gyroscope measurements which are then passed to their visual-inertial pipeline based on [22]. Brosard et al. used a CNN to denoise gyroscope measurements only which are then passed to Open-VINS [3]. Since both works presented their results using the EuRoC dataset [19], we also use this dataset to

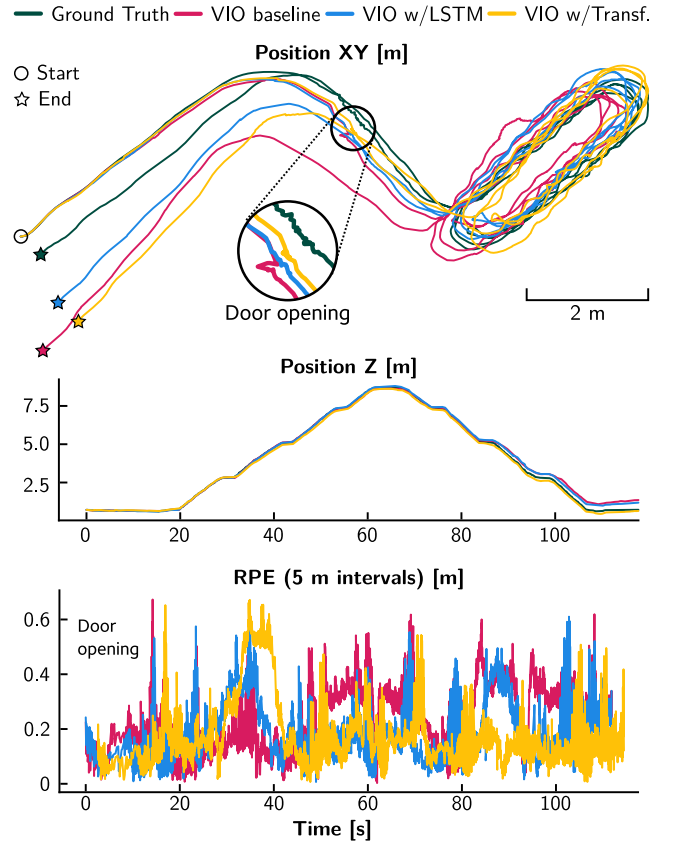


Fig. 8. NCD-Multi Stairs results. **Top**: Position of the baseline VIO, LSTM and Transformer methods on xy -plane. We magnify the door opening section of the trajectory where the baseline clearly diverges due to loss of visual feature tracking and poor bias estimation. **Middle**: Position of the trajectory for the z -axis. **Bottom**: 5m RPE for each algorithm.

demonstrate that our method can be applied to drones. Because the baselines were evaluated differently, drawing conclusions from direct comparisons is difficult. Therefore, we show relative improvement for each method.

Tables III and IV summarize these results in terms of absolute error. Note that [11] reported positional RMSE while [10] reported orientation RMSE and therefore we report both. For the V101 sequence, we use the updated ground truth from [3] as did Brosard et al. [10].

Brossard et al. additionally reported relative orientation errors by averaging 7 m, 21 m and 32 m relative orientation error for all trajectories. Averaged over the three distances they report 1.59° while we achieve 1.41° and 1.36° for LSTM and Transformer respectively.

D. Illustrative Experiments

We present two additional experiments to further analyze our system's performance. In both cases we used the Spot Easy trajectory and models trained with only handheld data.

1) *Blackout*: We “black out” the images for 5 s, setting all pixels to 0. As shown in Fig. 10 there is an error spike during the blackout where the estimator must rely on IMU data only. We also experiment with “locking” the IMU biases during the blackout rather than allowing the random walk.

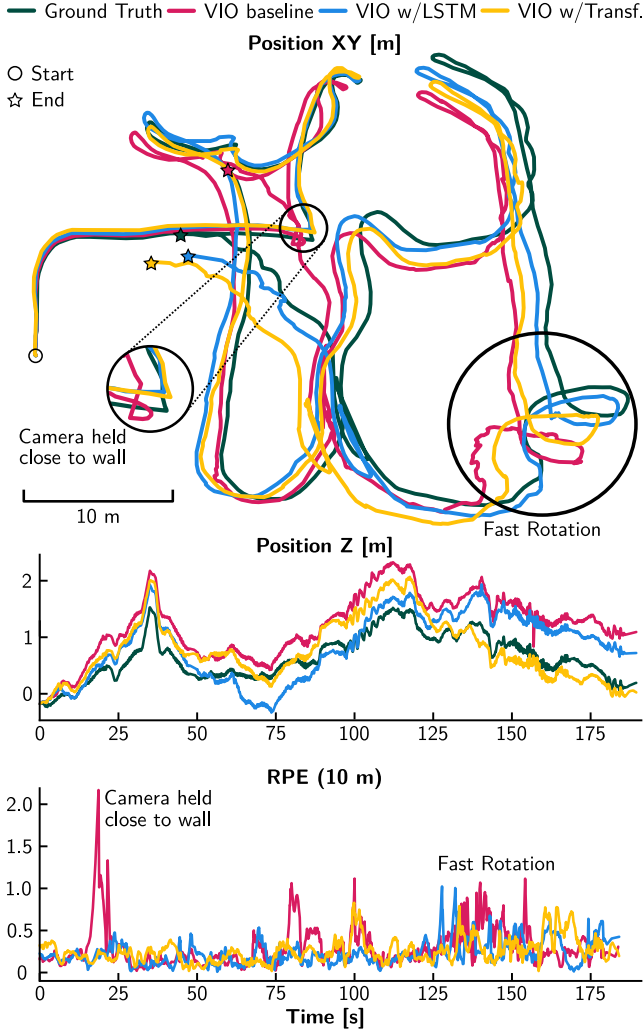


Fig. 9. Results from the Mine sequence. **Top:** Position on xy -plane of the baseline VIO, LSTM and Transformer methods. We magnify a part of the sequence where the camera was held facing and very close to a wall. The baseline diverges due to loss of visual feature tracking and poor bias estimation. We also highlight a section with large rotations where the baseline has multiple divergences. **Middle:** Position on z -axis. **Bottom:** 10m RPE for each algorithm.

2) *Image Distortion:* When there are errors in visual tracking, such as poor camera calibration or wrong correspondences, the associated error can be erroneously attributed to the IMU bias. Learned bias estimates can mitigate this by providing estimates of the correct bias. To demonstrate this, we intentionally inject visual tracking error by left shifting both stereo images by 10 pixels for 10 s. This is interpreted as an apparent rotation on the z -axis, attributed by the system to a change in yaw gyro bias. In Fig. 11 we plot the bias estimates during this time. Even though we initialize all methods with the ground truth bias and use a robust loss function, each method's estimate of the gyroscope z bias incorrectly drifts during the period of image distortion.

IX. DISCUSSION

In Section VIII-A we show our methods achieving reduced error compared to the baseline in several visually challenging

TABLE III
EuRoC: ABSOLUTE POSITION ERROR & IMPROVEMENT [m (%)]

Seq.	Zhang et al. [11]		Ours		
	Baseline	Method	Baseline	LSTM	Transformer
MH02	0.19	0.15 (21%)	0.37	0.23 (38%)	0.13 (65%)
MH04	0.15	0.14 (7%)	0.33	0.25 (24%)	0.25 (24%)
V101	0.08	0.07 (13%)	0.17	0.08 (53%)	0.08 (53%)
V103	0.27	0.15 (44%)	0.27	0.27 (0%)	0.17 (37%)
V202	0.11	0.10 (9%)	0.29	0.23 (21%)	0.10 (66%)

TABLE IV
EuRoC: ABSOLUTE ROTATION ERROR & IMPROVEMENT [deg (%)]

Seq.	Brossard et al. [10]		Ours		
	Baseline	Method	Baseline	LSTM	Transformer
MH02	1.11	1.21 (-9%)	3.21	3.10 (3%)	2.86 (11%)
MH04	1.60	1.40 (13%)	0.89	1.00 (-12%)	0.76 (15%)
V101	0.80	0.80 (0%)	2.56	1.17 (54%)	1.66 (35%)
V103	2.32	2.25 (3%)	4.78	3.54 (26%)	1.87 (61%)
V202	1.85	1.81 (2%)	3.78	2.53 (33%)	1.31 (65%)

situations. On average, the LSTM and Transformer have reduced error in Table II by 15%, but in certain situations, for example the camera held close to the wall in the Mine trajectory, the improvement is over 300%. This is both because the learned estimates improve bias estimation and because they stabilize the optimizer in situations where error is incorrectly attributed to the IMU bias.

In Section VIII-B we take a model trained with handheld data and apply it to quadrupedal data to demonstrate our method is locomotion agnostic. Additionally, it should be noted the Stairs trajectory is the only handheld trajectory with stair climbing and therefore also demonstrates locomotion agnosticism. The reduced error of our method compared to the baseline shows our method is sensor specific and not locomotion specific.

In the Section VIII-C drone experiments, our method has similar accuracy as other methods. However, each method uses a different VIO baseline. Our method improves upon our baseline VIO more so than other methods improve on their baselines. Also, as the EuRoC dataset provides ground truth bias estimates, we examine the accelerometer and gyroscope bias ($\hat{\mathbf{b}}^a, \hat{\mathbf{b}}^g$) estimation errors and find that, on average, error is reduced by (35%, 23%) for LSTM and (32%, 12%) for Transformer.

In the illustrative experiments in Section VIII-D we first present the Blackout experiment where our proposed methods reduced error during IMU-only estimation when no visual features could be tracked. Locking the bias did not improve on the baseline. This is because the bias random walk is already sufficiently constrained due to calibration. There is a second spike in error for all methods after vision is restored. This is because the pose estimate is suddenly corrected by a large amount, which locally appears as a relative error spike.

In the Image Distortion experiment, a bias shift affects all methods but the learned methods experience less error over the 10 s and a faster recovery period. During distortion, the error of LSTM and Transformer methods are 41% and 12% lower (RMSE) than the baseline.

LSTM and Transformer provide similar improvements to positional error with Transformer slightly better at rotational error. This makes sense because the gyroscope biases change

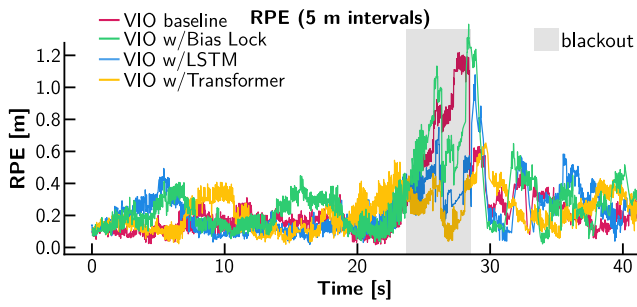


Fig. 10. **Blackout Experiment:** 5 m RPE Performance where the cameras were disabled for 5 s (gray area) during the Spot Easy trajectory. Mean 5 m RPE was 0.26 m, 0.28 m, 0.22 m and 0.20 m for the baseline, bias-lock, LSTM and Transformer methods respectively.

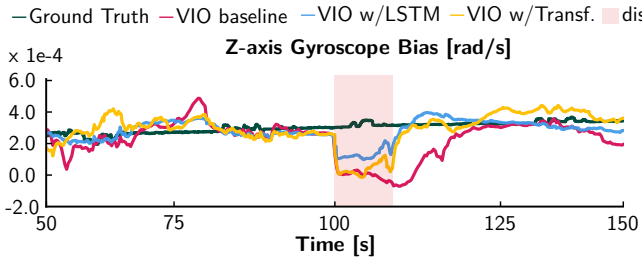


Fig. 11. **Image Distortion Experiment:** At 100 s of the Spot Easy trajectory we apply a small distortion to both stereo images (pink area). The baseline VIO develops a significant offset in z bias as the error is attributed to the gyroscope. LSTM and Transformer methods have reduced magnitude of error and recovery time.

more slowly and we expect the Transformer to be more accurate with longer-term dependencies. In the distortion experiment the Transformer instead performs more poorly than the LSTM, which we believe is due to the LSTM operating at 2 Hz allowing it to react to disturbances more quickly. In terms of model size, the LSTM is superior with learned models as small as 30 MB compared to 200 MB for the transformer. Additionally, the LSTM forward pass takes less than 6 ms as compared to 20 ms for the Transformer.

X. CONCLUSION

We present a novel application of machine learning to inertial navigation which is more interpretable than similar methods. By learning bias predictions the proposed method is more generally applicable because it is not robot specific and does not require periodic locomotion which we demonstrate with a wide variety of experiments on quadrupeds, handheld sensors and drones. We show that our method reduces 5 m RPE of our baseline VIO system by 15% on average in handheld and quadrupedal experiments with RPE being reduced by as much as 300% in certain situations where vision fails. In the future, we will build on this work by integrating uncertainty estimation into the model and experimenting with non-patterned locomotion such as wheeled robots.

REFERENCES

[1] 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, p. 1004–1011, 2021.

[2] S. Zhao, H. Zhang, P. Wang, L. Nogueira, and S. Scherer, “Super Odometry: IMU-centric LiDAR-visual-inertial estimator for challenging environments,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021, pp. 8729–8736.

[3] P. Geneva, K. Eickenhoff, W. Lee, Y. Yang, and G. Huang, “OpenVINS: A research platform for visual-inertial estimation,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 4666–4672.

[4] E. Foxlin, “Pedestrian tracking with shoe-mounted inertial sensors,” *IEEE Computer Graphics and Applications*, vol. 25, no. 6, pp. 38–46, 2005.

[5] 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.

[6] C. Chen, X. Lu, A. Markham, and N. Trigoni, “IoNet: Learning to cure the curse of drift in inertial odometry,” in *AAAI Conference on Artificial Intelligence*, 2018, pp. 6468–6476.

[7] S. Herath, H. Yan, and Y. Furukawa, “RoNIN: Robust neural inertial navigation in the wild: Benchmark, evaluations, & new methods,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 3146–3152.

[8] W. Liu, D. Caruso, E. Ilg, J. Dong, A. I. Mourikis, K. Daniilidis, et al., “TLIO: Tight learned inertial odometry,” *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5653–5660, 2020.

[9] R. Buchanan, M. Camurri, F. Dellaert, and M. Fallon, “Learning inertial odometry for dynamic legged robot state estimation,” in *Conference on Robot Learning*, vol. 164, 2022, pp. 1575–1584.

[10] M. Brossard, S. Bonnabel, and A. Barrau, “Denoising IMU gyroscopes with deep learning for open-loop attitude estimation,” *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4796–4803, 2020.

[11] M. Zhang, M. Zhang, Y. Chen, and M. Li, “IMU data processing for inertial aided navigation: A recurrent neural network based approach,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 3992–3998.

[12] J. Steinbrener, C. Brommer, T. Jantos, A. Fornasier, and S. Weiss, “Improved state propagation through AI-based pre-processing and down-sampling of high-speed inertial data,” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2022, pp. 6084–6090.

[13] D. Wisth, M. Camurri, and M. Fallon, “Vilens: Visual, inertial, lidar, and leg odometry for all-terrain legged robots,” *IEEE Transactions on Robotics*, pp. 1–18, 2022, doi: 10.1109/TRO.2022.3193788.

[14] D. Titterton and J. Weston, *Strapdown Inertial Navigation Technology*. Institution of Engineering and Technology, 2004.

[15] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, et al., “Attention is all you need,” in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[17] L. Zhang, M. Camurri, D. Wisth, and M. Fallon, “Multi-camera lidar inertial extension to the Newer College Dataset,” Tech. Rep., 2021. [Online]. Available: <https://arxiv.org/abs/2112.08854>

[18] 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.

[19] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, et al., “The EuRoC micro aerial vehicle datasets,” *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[20] R. J. Williams and D. Zipser, “A learning algorithm for continually running fully recurrent neural networks,” *Neural Computation*, vol. 1, no. 2, pp. 270–280, 1989.

[21] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of RGB-D SLAM systems,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012, pp. 573–580.

[22] M. Zhang, X. Zuo, Y. Chen, Y. Liu, and M. Li, “Pose estimation for ground robots: On manifold representation, integration, reparameterization, and optimization,” *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1081–1099, 2021.

©2022 IEEE. Reprinted, with permission, from Russell Buchanan, Varun Agrawal, Marco Camurri, Frank Dellaert and Maurice Fallon, “Deep IMU Bias Inference for Robust Visual-Inertial Odometry with Factor Graphs,” in *Robotics and Automation Letters*, Accepted November 2022.

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	Deep IMU Bias Inference for Robust Visual-Inertial Odometry with Factor Graphs
Publication Status	☒ Published ☐ Accepted for Publication ☐ Submitted for Publication ☐ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Russell Buchanan, Varun Agrawal, Marco Camurri, Frank Dellaert, and Maurice Fallon "Deep IMU Bias Inference for Robust Visual-Inertial Odometry with Factor Graphs," accepted for publication in IEEE Robotics and Automation Letters. 2022

Student Confirmation

Student Name:	Russell Buchanan		
Contribution to the Paper	• Developed original idea • Wrote majority of the code • Lead data collection • Analyzed all experiment data • Wrote large portions of the paper		
Signature		Date	February 24, 2023

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 contributions listed above are an accurate representation of the work Russell contributed			
Signature		Date	February 24, 2023

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

7.2 Discussion

In this chapter I introduced a new method for applying deep learning to VIO in which a neural network learns a model of the IMU bias process. I showed how this can help reduce drift in very challenging situations such as motion blur or large occlusions. Because the method learns the process of a given IMU, it is device specific, not locomotion specific and I showed how a network trained on human walking data can be used on legged robot data.

For the VIO implementation I use the same visual feature tracking scheme as in Chapter 6 using FAST corner detection and KLT feature tracking. Unlike the ultrasound scanner in Section 6.2, the VIO is used to track a mobile device (carried by a human or robot) which moves through different environments. This means that new features must be detected while old ones are lost and in certain situations, all features may become lost for up to several seconds. This motivates the need for accurate IMU estimation to reduce drift in these situations.

In this chapter I present a novel application of deep learning to IMUs. To the best of my knowledge, this was the first method to train a network to explicitly learn an IMU bias evolution model. This opens several possibilities for future work in the area of learning IMU noise models and in Section 8.1.3 I cover several of these ideas.

Part IV

Conclusion

8

Future Work and Conclusion

8.1 Future Work

In this thesis I set out several new methods to improve legged robot state estimation in visually challenging environments. The first two of these methods are entirely proprioceptive and enable localisation using haptic information. Then, I successfully adapted deep learning inertial odometry from the pedestrian tracking research community and applied it to legged robots. This can be fused with other proprioceptive sensors, such as legged robot joint sensors, or with exteroceptive sensors, such as cameras. I also later adapted this method to enable 3D reconstruction of human bones from entirely IMU-only odometry. Finally, I introduced a method for learning of IMU bias models which can help improve robustness of visual-inertial odometry. While exploring these various avenues of research, I have identified several promising directions for future work to improve legged robot state estimation in visually challenging environments.

8.1.1 Haptic SLAM

In [14], we showed how a legged robot can localise in a prior map using terrain type semantics and geometry information. This was based on the assumption that a prior map of the environment was available which may not always be the case.

Instead, it may be possible for the robot to learn the map as it traverses the terrain, registering geometric and semantic information in a map. Deep learning could be used to complete terrain maps from the sparse information [132].

If the robot can build its own map while walking, this would be a step towards haptic SLAM. So long as the robot has a low-drift method of odometry and the trajectory contains loops, the robot could start with no prior map and build a haptic map as it explores. This would require the haptic descriptor to be sufficiently unique so as not to introduce erroneous loop closures. Geometric information could also be used in this way: walking up steps or stairs could enable re-localisation.

Deep learning methods for localization might also be an interesting direction for future work. Differential particle filters (DPFs) [77] are a new type of learning-based particle filter. Instead of using a fixed distribution (e.g. Gaussian) to propagate the particles or for the measurement noise, the distributions are learned by a neural network for a given problem. It could be possible for the DPF to jointly learn a motion model from IMU measurements and a measurement model from the terrain map.

In [98] we used learned terrain representations – or descriptors – eliminating the need for explicit terrain type classification. Instead, the robot makes a first pass over the terrain, encodes the descriptors in a map and then on subsequent passes over the terrain uses the map to re-localise. This is similar to the concept of “Visual Teach & Repeat” (VT&R) [50], however in this work we did not perform closed loop navigation. One of the challenges to VT&R is that during the repeat phase, the robot cannot stray too far from the path of the teach phase. For haptic VT&R this may be even more restrictive since the robot must physically touch the same places as in the teach phase.

Extending these ideas to enable entirely haptic based mapping and navigation would enable legged robots to be deployed in challenging real-world scenarios without any prior information. For example, a dark subterranean environment, or a smoke-filled stairwell, both of which are significantly challenging for cameras or lidar. It would be necessary to implement probing behaviours to check for

obstacles or hazards before taking a step (such as in [135]). Vision of some sort, e.g. infrared or thermal, would still likely be needed to some extent to perform useful tasks in these situations.

8.1.2 Learning for Legged Robot State Estimation

Deep learning has shown to be an effective way to approximate complex non-linear functions which do not have an analytical solution; for example, human motion from inertial data [24] or visual tracking from camera images [150]. In [15] I applied inertial learning techniques to legged robots to improve state estimation in challenging situations, where kinematic sensing is less reliable e.g. slipping or deformable terrain. More generally there are several other directions where deep learning could be applied to improve legged robot state estimation.

Contact Detection

Detecting contact states remains a challenge for legged robots which has typically been approached by applying threshold-based detector on the ground reaction forces or velocities [22, 73]. These methods can be quite robust to different terrain types but require tuning for the specific robot and will be less accurate on very soft or slippery terrain. More recently, machine learning has been applied [95, 113] to this problem. Piperakis et al [113] trained a contact detection model in simulation and showed it could generalise to different robots.

To reduce state estimation error from contact misclassification, learning based contacted detectors could be combined with hybrid discrete-continuous smoothers as proposed by [76, 39]. Hybrid smoothers could exploit the contact predictions from the network, as well as the uncertainty values, and include this in the optimisation problem.

Learning from Kinematics

The original works on inertial learning were intended for pedestrians with only IMUs available as sensors [24, 96]. In my work, I applied this method to legged robots and showed how to fuse the learned models with other sensors for robust

estimation [15]. However, there are a variety of additional directions in which the additional sensing on a legged robot could be used to improved motion learning.

One aspect, already explore in the extent of micro aerial vehicles [30] is the use of control input commands as well as IMU signal as network inputs.

In the case of legged robots, joint sensing could also be a useful input to the network. While it may be possible to improve displacement prediction by adding the raw joint position and torque measurements, this would essentially require the network to learn the robot forward kinematics and dynamics. Instead, it would likely be more useful to provide the network with processed measurements such as end-effector positions and velocities. This would likely improve the learned motion model prediction for legged robots as it would provide more platform-specific cues that can be further exploited as features for learning.

Learning with IMU Data

Future work should also explore extending inertial learning. In [15], I used a neural network to predict displacement measurements from inertial data but it did not predict changes in orientation. Roll and pitch can be accurately estimated from a calibrated IMU using integration; however, the yaw rotation is unobservable and in a VIO or IMU-kinematic system will drift. It would be useful to train a network to also predict yaw change measurements from inertial data thereby enabling full 6 DoF estimation from only IMU data. This may be easier for legged robots than for pedestrians due to the very regular walking pattern.

8.1.3 Deep Learning for IMU de-noising

In [13] we presented a novel method for learning how the biases of an IMU evolve over time. This method focused on learning the process model of the IMU bias evolution which was in contrast with existing work on de-noising which instead removed the high-frequency noise sources such as the IMU white noise or the noise arising from locomotion [11, 129]. In principle, there is no reason why these methods cannot be used together, correcting for the high frequency noises and the slowly changing bias.

A different neural network architecture might also improve the performance. It may be possible to express the IMU noise dynamics as a differential equation which could be learned using the latest methods such as Neural Ordinary Differential Equations (NODE) [28]. In our previous work we trained a network $\mathcal{F}_\theta$ to predict the next bias from past IMU and bias information:

$$\mathcal{F}_\theta : (\mathcal{I}_{mj}, \mathbf{b}_{ni}^a, \mathbf{b}_{ni}^g) \mapsto (\hat{\mathbf{b}}_j^a, \hat{\mathbf{b}}_j^g) \quad (8.1)$$

Using an NODE, we could instead explicitly learn the bias transition function

$$\mathcal{F}_\theta(\mathcal{I}_{mj}, \mathbf{b}_{ni}^a) = \frac{\partial \hat{\mathbf{b}}_i^a}{\partial t} \quad (8.2)$$

$$\mathcal{F}_\theta(\mathcal{I}_{mj}, \mathbf{b}_{ni}^g) = \frac{\partial \hat{\mathbf{b}}_i^g}{\partial t} \quad (8.3)$$

$$(8.4)$$

This would mean that the network could predict the derivative of the bias. This could then be added to a factor graph as a factor between two bias states.

Another possible change of architecture could be to explore recent new methods for learning systems with very long-term dependencies. IMU biases change over the course of several minutes but have sampling rates on the order of 100 Hz. Using every single IMU measurement would not be feasible for LSTMs or Transformers. For this reason, in [13] we sub-sampled the IMU to 100 Hz which meant some information was discarded. If we want to use all the data, we need new architectures for long-term dependencies.

Recent work from Gu et al. [57] has shown promise using state space models to massively increase the capability for learning long sequences. They report significantly improved performance on long term sequential datasets (80% average accuracy compared to 60% for several Transformer models). Additionally, their method was the only one capable of learning from sequences of 16k time steps. This network implementation could improve learning of the IMU bias evolution model because the network would be able to use more data from longer sequences.

Finally, in addition to improving the noise model learning, it would also be interesting future work to combine these techniques with existing learning-based

VIO methods. In SelfVIO [2] separate, pre-processing networks are trained each for IMU and visual measurements to extract features which are then passed to an LSTM for pose prediction. Instead, the LSTM could predict pose + current IMU biases. The LSTM bias predictions could then be provided to a module which was trained separately for a given IMU, to predict the next bias values. The next bias predictions can then be passed back to the IMU pre-processing in a closed loop.

8.2 Conclusion

The motivation behind the research described in this thesis lies in the enormous potential of legged robots to benefit society. They have the ability to navigate the world, which is primarily built for walking humans, by traversing through challenging environments such as stairs, snow and mud. This would make legged robots ideal for automating jobs that are too dangerous, dull, or dirty for humans. Currently available quadrupeds can carry payloads of several kilograms and even manipulate objects such as door handles, making them suitable for industrial applications such as inspecting sewers, mapping construction sites and monitoring offshore platforms. Additionally, there is growing interest in the potential of legged robots to assist first responders in emergency situations. Despite the potential for legged robots, there remains a significant gap in the research, preventing the real-world deployment of these robots.

Accurate state estimation is a fundamental requirement for any autonomous system. While current methods using vision and lidar are very accurate, they often fail when deployed in the exact places we wish legged robots to be used. From darkness and fog to airborne particles such as dust, these are common environmental challenges that limit the reliability of vision-based state estimation. This in turn, significantly limits the usefulness of legged robots.

In summary, in this thesis I have addressed some of the challenges that exist by developing new methods of state estimation specifically for vision-denied or visually challenging environments. The methods primarily focus on legged robots and handheld devices, but could be applied to other mobile platforms as well.

First, I exploited haptic information to achieve proprioceptive localisation within a 3D environment. Then I applied inertial learning to legged robots, improving state estimation in visually challenging environments. I further implemented a method for 3D reconstruction of bones from ultrasound using only IMU sensing. Finally, I used deep learning to infer the evolution model of IMU biases for improved state estimation in exteroceptive systems.

Each of these methods have been extensively tested and demonstrated in real experiments on various robots and handheld devices, including for scanning phantom bones. In conclusion, this thesis presents a significant and substantial contribution to legged robot state estimation by filling the gap of methods developed for visually challenging situations.

Appendices



Derivation of Covariance Propagation of Imu Factors in $SO(3)$

In this thesis, the “CombinedImuFactor” of the GTSAM library [36] is used extensively. This is an implementation of the preintegrated IMU factor introduced by Forster et al. [46] which was based on preintegration theory introduced by Lupton et al. [97]. In the CombinedImuFactor, the noise propagation of the biases is handled together with the pose estimate uncertainty. This is in contrast to the “ImuFactor” implementation which handles bias uncertainty and pose uncertainty separately. While both these IMU factor implementations have been used extensively in research [15, 130, 153, 164], there has not been a thorough comparison of the two factors.

In this section, I present derivations of the uncertainty propagation of both the ImuFactor and the CombinedImuFactor. These derivations were first done by Forster et al. [46] Appendix A and by Dellaert and Agrawal [35]. Next, I prove that they are not equivalent, even when the additional noise sources in CombinedImuFactor are set to zero.

A.1 The ImuFactor

Referring to the IMU mechanisation equations of (2.18) and the bias process model of (2.10) we can express the pre-integration equations as follows:

$$\begin{aligned}
\Delta \mathbf{R}_{t+1} &= \Delta \mathbf{R}_t \text{Exp} \left((\tilde{\boldsymbol{\omega}}_t - \mathbf{b}_t^g - \boldsymbol{\eta}_t^{gd}) \Delta t \right) \\
\Delta \mathbf{p}_{t+1} &= \Delta \mathbf{p}_t + \Delta \mathbf{v}_t + \Delta \mathbf{R}_t \left((\tilde{\mathbf{a}}_t - \mathbf{b}_t^a - \boldsymbol{\eta}_t^{ad}) \Delta t \right) + \boldsymbol{\eta}_{int} \\
\Delta \mathbf{v}_{t+1} &= \Delta \mathbf{v}_t + \Delta \mathbf{R}_t \left(\tilde{\mathbf{a}}_t - \mathbf{b}_t^a - \boldsymbol{\eta}_t^{ad} \right) \Delta t \\
\Delta \mathbf{b}_{t+1}^a &= \Delta \mathbf{b}_t^a + \boldsymbol{\eta}_t^{bad} \\
\Delta \mathbf{b}_{t+1}^g &= \Delta \mathbf{b}_t^g + \boldsymbol{\eta}_t^{bgd},
\end{aligned} \tag{A.1}$$

where, according to preintegration theory, $\Delta \mathbf{R}_0 = I_3, \Delta \mathbf{p}_0 = 0_3, \Delta \mathbf{v}_0 = 0_3$ [97]. We are integrating measurements between $[t_1, t_2]$ so that $\Delta \mathbf{v}_t$ is the change in velocity at t with respect to t_1 (where $t_1 < t < t_2$). Similarly $\Delta \mathbf{R}_t$ transforms a vector in the body frame at time t into the body frame at t_1 . And $\Delta \mathbf{p}_t$ is the change in position at t compared to t_1 .

Additionally, we add $\boldsymbol{\eta}_{int}$ to represent the uncertainty in integrating position from velocities [35]. It is assumed that during preintegration the biases are constant (i.e. $\Delta \mathbf{b}_{t+1}^a = \Delta \mathbf{b}_t^a$) however, we keep the random walk terms here to analyse the noise propagation.

Equation A.1 can be simplified into the following general preintegration equations which are a function of the state $\mathbf{x} = \{\Delta \mathbf{R}_t, \Delta \mathbf{p}_t, \Delta \mathbf{v}_t, \Delta \mathbf{b}_t^a, \Delta \mathbf{b}_t^g\}$ and the measurements $\{\tilde{\mathbf{a}}_t, \tilde{\boldsymbol{\omega}}_t\}$:

$$\begin{aligned}
\Delta \mathbf{R}_{t+1} &= f_{\mathbf{R}} (\Delta \mathbf{R}_t, \Delta \mathbf{b}_t^g, \tilde{\boldsymbol{\omega}}_t) \\
\Delta \mathbf{p}_{t+1} &= f_p (\Delta \mathbf{R}_t, \Delta \mathbf{p}_t, \Delta \mathbf{v}_t, \Delta \mathbf{b}_t^a, \tilde{\mathbf{a}}_t) \\
\Delta \mathbf{v}_{t+1} &= f_v (\Delta \mathbf{R}_t, \Delta \mathbf{v}_t, \Delta \mathbf{b}_t^a, \tilde{\mathbf{a}}_t) \\
\Delta \mathbf{b}_{t+1}^a &= f_{\mathbf{b}^a} (\Delta \mathbf{b}_t^a) \\
\Delta \mathbf{b}_{t+1}^g &= f_{\mathbf{b}^g} (\Delta \mathbf{b}_t^g).
\end{aligned} \tag{A.2}$$

From Lupton et al. [97] we can write the covariance propagation $\mathbf{P} \in \mathbb{R}^{15 \times 15}$ during preintegration as:

$$\mathbf{P}_{t+1} = \mathbf{F}_t \mathbf{P}_t \mathbf{F}_t^T + \mathbf{G}_t \mathbf{Q}_t \mathbf{G}_t^T, \quad (\text{A.3})$$

where $\mathbf{P}_0 = \mathbf{0}_{15 \times 15}$. The matrix $\mathbf{F}_t \in \mathbb{R}^{15 \times 15}$ is the Jacobian of (A.2) with respect to the state $\mathbf{x}$. $\mathbf{G}_t \in \mathbb{R}^{15 \times 6}$ is the Jacobian with respect to the measurement uncertainty $\boldsymbol{\eta}^{ad}, \boldsymbol{\eta}^{gd}$ and $\mathbf{Q}_t \in \mathbb{R}^{6 \times 6}$ is a diagonal matrix containing the measurement noise sources $\{\boldsymbol{\eta}^{ad}, \boldsymbol{\eta}^{gd}\}$.

$\mathbf{F}$ and $\mathbf{G}$ can be easily computed from (A.1):

$$\mathbf{F}_t = \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{R}_t} & \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{p}_t} & \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{v}_t} & \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{b}_t^a} & \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{b}_t^g} \\ \frac{\partial f_p}{\partial \Delta \mathbf{R}_t} & \frac{\partial f_p}{\partial \Delta \mathbf{p}_t} & \frac{\partial f_p}{\partial \Delta \mathbf{v}_t} & \frac{\partial f_p}{\partial \Delta \mathbf{b}_t^a} & \frac{\partial f_p}{\partial \Delta \mathbf{b}_t^g} \\ \frac{\partial f_v}{\partial \Delta \mathbf{R}_t} & \frac{\partial f_v}{\partial \Delta \mathbf{p}_t} & \frac{\partial f_v}{\partial \Delta \mathbf{v}_t} & \frac{\partial f_v}{\partial \Delta \mathbf{b}_t^a} & \frac{\partial f_v}{\partial \Delta \mathbf{b}_t^g} \\ \frac{\partial f_{\mathbf{b}}^a}{\partial \Delta \mathbf{R}_t} & \frac{\partial f_{\mathbf{b}}^a}{\partial \Delta \mathbf{p}_t} & \frac{\partial f_{\mathbf{b}}^a}{\partial \Delta \mathbf{v}_t} & \frac{\partial f_{\mathbf{b}}^a}{\partial \Delta \mathbf{b}_t^a} & \frac{\partial f_{\mathbf{b}}^a}{\partial \Delta \mathbf{b}_t^g} \\ \frac{\partial f_{\mathbf{b}}^g}{\partial \Delta \mathbf{R}_t} & \frac{\partial f_{\mathbf{b}}^g}{\partial \Delta \mathbf{p}_t} & \frac{\partial f_{\mathbf{b}}^g}{\partial \Delta \mathbf{v}_t} & \frac{\partial f_{\mathbf{b}}^g}{\partial \Delta \mathbf{b}_t^a} & \frac{\partial f_{\mathbf{b}}^g}{\partial \Delta \mathbf{b}_t^g} \end{bmatrix} = \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{R}_t} & 0 & 0 & 0 & \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{b}_t^g} \\ \frac{\partial f_p}{\partial \Delta \mathbf{R}_t} & \mathbf{I} & \frac{\partial f_p}{\partial \Delta \mathbf{v}_t} & \frac{\partial f_p}{\partial \Delta \mathbf{b}_t^a} & 0 \\ \frac{\partial f_v}{\partial \Delta \mathbf{R}_t} & 0 & \mathbf{I} & \frac{\partial f_v}{\partial \Delta \mathbf{b}_t^a} & 0 \\ 0 & 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I} \end{bmatrix}, \quad (\text{A.4})$$

$$\mathbf{G}_t = \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{ad}} & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{gd}} \\ \frac{\partial f_p}{\partial \boldsymbol{\eta}^{ad}} & \frac{\partial f_p}{\partial \boldsymbol{\eta}^{gd}} \\ \frac{\partial f_v}{\partial \boldsymbol{\eta}^{ad}} & \frac{\partial f_v}{\partial \boldsymbol{\eta}^{gd}} \\ \frac{\partial f_{\mathbf{b}}^a}{\partial \boldsymbol{\eta}^{ad}} & \frac{\partial f_{\mathbf{b}}^a}{\partial \boldsymbol{\eta}^{gd}} \\ \frac{\partial f_{\mathbf{b}}^g}{\partial \boldsymbol{\eta}^{ad}} & \frac{\partial f_{\mathbf{b}}^g}{\partial \boldsymbol{\eta}^{gd}} \end{bmatrix} = \begin{bmatrix} 0 & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{gd}} \\ \frac{\partial f_p}{\partial \boldsymbol{\eta}^{ad}} & 0 \\ \frac{\partial f_v}{\partial \boldsymbol{\eta}^{ad}} & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}. \quad (\text{A.5})$$

Now a few observations can be made. Assuming $\mathbf{P}_0 = \mathbf{0}_{15 \times 15}$ at t_1 , it is clear the bottom rows of $\mathbf{G}$ that the covariance of the bias random walk does not contribute to the covariance of the state and can therefore be ignored. Additionally, if we augment the noise vector to $\tilde{\mathbf{Q}}_t = \text{diag}(\boldsymbol{\eta}_{int}, \boldsymbol{\eta}^{ad}, \boldsymbol{\eta}^{gd})$ to include the integration noise we can augment $\mathbf{G}$ to $\tilde{\mathbf{G}}$ with an extra column, while removing the zero rows. This gives us the augmented matrices of the ImuFactor:

$$\tilde{\mathbf{F}}_t = \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{R}_t} & 0 & 0 \\ \frac{\partial f_p}{\partial \Delta \mathbf{R}_t} & \mathbf{I} & \frac{\partial f_p}{\partial \Delta \mathbf{v}_t} \\ \frac{\partial f_v}{\partial \Delta \mathbf{R}_t} & 0 & \mathbf{I} \end{bmatrix}, \quad (\text{A.6})$$

$$\tilde{\mathbf{G}}_t = \begin{bmatrix} 0 & 0 & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{gd}} \\ \mathbf{I} & \frac{\partial f_p}{\partial \boldsymbol{\eta}^{ad}} & 0 \\ 0 & \frac{\partial f_v}{\partial \boldsymbol{\eta}^{ad}} & 0 \end{bmatrix}, \quad (\text{A.7})$$

$$\tilde{\mathbf{Q}}_t = \begin{bmatrix} \boldsymbol{\eta}_{int} & 0 & 0 \\ 0 & \boldsymbol{\eta}^{ad} & 0 \\ 0 & 0 & \boldsymbol{\eta}^{gd} \end{bmatrix}, \quad (\text{A.8})$$

which together computes the augmented covariance propagation:

$$\tilde{\mathbf{P}}_{t+1} = \tilde{\mathbf{F}}_t \tilde{\mathbf{P}}_t \tilde{\mathbf{F}}_t^T + \tilde{\mathbf{G}}_t \tilde{\mathbf{Q}}_t \tilde{\mathbf{G}}_t^T. \quad (\text{A.9})$$

Solutions for the derivatives in (A.6) and (A.7) are provided in [35].

It will be useful later to solve for $\tilde{\mathbf{G}}_t \tilde{\mathbf{Q}}_t \tilde{\mathbf{G}}_t^T$ in terms of the symbols in (A.7) and (A.8). It is possible to solve the matrix multiplication as:

$$\tilde{\mathbf{G}}_t \tilde{\mathbf{Q}}_t \tilde{\mathbf{G}}_t^T = \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{gd}} \boldsymbol{\eta}^{gd} \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{gd}}^T & 0 & 0 \\ 0 & \frac{\partial f_p}{\partial \boldsymbol{\eta}^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_p}{\partial \boldsymbol{\eta}^{ad}}^T + \boldsymbol{\eta}_{int} & \frac{\partial f_p}{\partial \boldsymbol{\eta}^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_v}{\partial \boldsymbol{\eta}^{ad}}^T \\ 0 & \frac{\partial f_v}{\partial \boldsymbol{\eta}^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_p}{\partial \boldsymbol{\eta}^{ad}}^T & \frac{\partial f_v}{\partial \boldsymbol{\eta}^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_v}{\partial \boldsymbol{\eta}^{ad}}^T \end{bmatrix} \quad (\text{A.10})$$

A.2 The CombinedImuFactor

The primary insight behind the CombinedImuFactor is that the biases at t_1 are not known and are instead estimated with some uncertainty. The goal of the CombinedImuFactor is to propagate this uncertainty in the state estimation. First, we re-write the preintegration equations adding noise to represent uncertainty in bias estimation for accelerometer and gyroscope biases respectively $\{\boldsymbol{\eta}_{init}^a, \boldsymbol{\eta}_{init}^g\}$:

$$\begin{aligned} \Delta \mathbf{R}_{t+1} &= \Delta \mathbf{R}_t \text{Exp} \left((\tilde{\boldsymbol{\omega}}_t - \mathbf{b}_{t1}^g - \Delta \mathbf{b}_t^g - \boldsymbol{\eta}_t^{gd} - \boldsymbol{\eta}_{init}^g) \Delta t \right) \\ \Delta \mathbf{p}_{t+1} &= \Delta \mathbf{p}_t + \Delta \mathbf{v}_t + \Delta \mathbf{R}_t \left((\tilde{\mathbf{a}}_t - \mathbf{b}_{t1}^a - \Delta \mathbf{b}_t^a - \boldsymbol{\eta}_t^{gd} - \boldsymbol{\eta}_{init}^a) \Delta t \right) + \boldsymbol{\eta}_{int} \\ \Delta \mathbf{v}_{t+1} &= \Delta \mathbf{v}_t + \Delta \mathbf{R}_t \left(\tilde{\mathbf{a}}_t - \mathbf{b}_{t1}^a - \Delta \mathbf{b}_t^a - \boldsymbol{\eta}_t^{ad} - \boldsymbol{\eta}_{init}^a \right) \Delta t \\ \Delta \mathbf{b}_{t+1}^a &= \Delta \mathbf{b}_t^a + \boldsymbol{\eta}_t^{bad} \\ \Delta \mathbf{b}_{t+1}^g &= \Delta \mathbf{b}_t^g + \boldsymbol{\eta}_t^{bgd} \end{aligned} \quad (\text{A.11})$$

which then allows us to write the generalised preintegration functions as functions of the noise terms as well:

$$\begin{aligned}
\Delta \mathbf{R}_{t+1} &= f_{\mathbf{R}} \left(\Delta \mathbf{R}_t, \Delta \mathbf{b}_t^g, \mathbf{b}_{t1}^g, \tilde{\omega}_t, \boldsymbol{\eta}^{gd}, \boldsymbol{\eta}_{init}^g \right) \\
\Delta \mathbf{p}_{t+1} &= f_p \left(\Delta \mathbf{R}_t, \Delta \mathbf{p}_t, \Delta \mathbf{v}_t, \Delta \mathbf{b}_t^a, \mathbf{b}_{t1}^a, \tilde{\mathbf{a}}_t, \boldsymbol{\eta}_{int}, \boldsymbol{\eta}^{ad}, \boldsymbol{\eta}_{init}^a \right) \\
\Delta \mathbf{v}_{t+1} &= f_v \left(\Delta \mathbf{R}_t, \Delta \mathbf{v}_t, \Delta \mathbf{b}_t^a, \mathbf{b}_{t1}^a, \tilde{\mathbf{a}}_t, \boldsymbol{\eta}^{ad}, \boldsymbol{\eta}_{init}^a \right) \\
\Delta \mathbf{b}_{t+1}^a &= f_{\Delta \mathbf{b}^a} \left(\Delta \mathbf{b}_t^a, \boldsymbol{\eta}_t^{bad} \right) \\
\Delta \mathbf{b}_{t+1}^g &= f_{\Delta \mathbf{b}^g} \left(\Delta \mathbf{b}_t^g, \boldsymbol{\eta}_t^{bgd} \right).
\end{aligned} \tag{A.12}$$

Now we compute the covariance propagation matrices as in Section A.1:

$$\mathbf{P}_{t+1} = \mathbf{F}_t \mathbf{P}_t \mathbf{F}_t^T + \mathbf{G}_t \mathbf{Q}_t \mathbf{G}_t^T \tag{A.13}$$

Using the state $\mathbf{x} = \{\Delta \mathbf{R}, \Delta \mathbf{p}, \Delta \mathbf{v}, \Delta \mathbf{b}^a, \Delta \mathbf{b}^g\}$ we can compute the matrix $\mathbf{F}_t$ again:

$$\mathbf{F}_t = \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{R}} & 0 & 0 & 0 & \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{b}_t^g} \\ \frac{\partial f_p}{\partial \Delta \mathbf{R}} & \mathbf{I} & \frac{\partial f_p}{\partial \Delta \mathbf{v}_t} & \frac{\partial f_p}{\partial \Delta \mathbf{b}_t^a} & 0 \\ \frac{\partial f_v}{\partial \Delta \mathbf{R}} & 0 & \mathbf{I} & \frac{\partial f_v}{\partial \Delta \mathbf{b}_t^a} & 0 \\ 0 & 0 & 0 & \mathbf{I} & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I} \end{bmatrix} \tag{A.14}$$

which is equivalent to the ImuFactor $\mathbf{F}_t$ matrix in (A.4). Now we use the augmented noise matrix $\hat{\mathbf{Q}}_t = \text{diag}(\boldsymbol{\eta}_{int}, \boldsymbol{\eta}^{ad}, \boldsymbol{\eta}^{gd}, \boldsymbol{\eta}^{bad}, \boldsymbol{\eta}^{bgd}, \boldsymbol{\eta}_{init}^a, \boldsymbol{\eta}_{init}^g) \in \mathbb{R}^{21 \times 21}$. The terms $\boldsymbol{\eta}_{init}^a, \boldsymbol{\eta}_{init}^g$ can either be treated separately as diagonal terms in $\hat{\mathbf{Q}}_t$, assuming no correlation, or they can be merged into $\boldsymbol{\eta}_{init} \in \mathbb{R}^{6 \times 6}$. This gives the combined noise matrix:

$$\hat{\mathbf{Q}}_t = \begin{bmatrix} \boldsymbol{\eta}_{int} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \boldsymbol{\eta}^{ad} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \boldsymbol{\eta}^{gd} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \boldsymbol{\eta}^{bad} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{\eta}^{bgd} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \boldsymbol{\eta}_{init11} & \boldsymbol{\eta}_{init12} \\ 0 & 0 & 0 & 0 & 0 & \boldsymbol{\eta}_{init21} & \boldsymbol{\eta}_{init22} \end{bmatrix}. \tag{A.15}$$

Now we solve for $\hat{\mathbf{G}}$ which is the Jacobian of A.13 with respect to the noise terms:

$$\begin{aligned}
\hat{\mathbf{G}}_t &= \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}_{int}} & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{ad}} & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{gd}} & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{bad}} & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{bgd}} & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}_{init}^a} & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}_{init}^g} \\ \frac{\partial f_p}{\partial \boldsymbol{\eta}_{int}} & \frac{\partial f_p}{\partial \boldsymbol{\eta}^{ad}} & \frac{\partial f_p}{\partial \boldsymbol{\eta}^{gd}} & \frac{\partial f_p}{\partial \boldsymbol{\eta}^{bad}} & \frac{\partial f_p}{\partial \boldsymbol{\eta}^{bgd}} & \frac{\partial f_p}{\partial \boldsymbol{\eta}_{init}^a} & \frac{\partial f_p}{\partial \boldsymbol{\eta}_{init}^g} \\ \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_v} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_v} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_v} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_v} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_v} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_v} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_v} \\ \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^a}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^a}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^a}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^a}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^a}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^a}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^a}} \\ \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^g}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^g}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^g}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^g}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^g}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^g}} & \frac{\partial \boldsymbol{\eta}_{int}}{\partial f_{\mathbf{b}^g}} \end{bmatrix} \\
&= \begin{bmatrix} 0 & 0 & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}^{gd}} & 0 & 0 & 0 & \frac{\partial f_{\mathbf{R}}}{\partial \boldsymbol{\eta}_{init}^g} \\ \frac{\partial f_p}{\partial \boldsymbol{\eta}_{int}} & \frac{\partial f_p}{\partial \boldsymbol{\eta}^{ad}} & 0 & 0 & 0 & \frac{\partial f_p}{\partial \boldsymbol{\eta}_{init}^a} & 0 \\ 0 & \frac{\partial f_v}{\partial \boldsymbol{\eta}^{ad}} & 0 & 0 & 0 & \frac{\partial f_v}{\partial \boldsymbol{\eta}_{init}^a} & 0 \\ 0 & 0 & 0 & \mathbf{I} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{I} & 0 & 0 \end{bmatrix}.
\end{aligned} \tag{A.16}$$

So the covariance propagation for the CombinedImuFactor is given as:

$$\hat{\mathbf{P}}_{t+1} = \mathbf{F}_t \hat{\mathbf{P}}_t \mathbf{F}_t^T + \hat{\mathbf{G}}_t \hat{\mathbf{Q}}_t \hat{\mathbf{G}}_t^T. \tag{A.17}$$

.

Now we compute the second term in A.17 $\hat{\mathbf{G}}_t \hat{\mathbf{Q}}_t \hat{\mathbf{G}}_t^T$. The steps of matrix

multiplication are shown in [35], we show only the solution here for simplicity. The

final matrix is broken into a sum of matrices of the diagonal and off-diagonal terms:

$$\begin{aligned}
\hat{\mathbf{G}}_t \hat{\mathbf{Q}}_t \hat{\mathbf{G}}_t^T = & \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \eta^{gd}} \boldsymbol{\eta}^{gd} \frac{\partial f_{\mathbf{R}}}{\partial \eta^{gd}}^T & 0 & 0 & 0 & 0 \\ 0 & \frac{\partial f_p}{\partial \eta^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_p}{\partial \eta^{ad}}^T & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial f_v}{\partial \eta^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_v}{\partial \eta^{ad}}^T & 0 & 0 \\ 0 & 0 & 0 & \boldsymbol{\eta}^{bad} & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{\eta}^{bgd} \end{bmatrix} + \\
& \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \eta_{init}^g} \boldsymbol{\eta}_{init22} \frac{\partial f_{\mathbf{R}}}{\partial \eta_{init}^g}^T & 0 & 0 & 0 & 0 \\ 0 & \frac{\partial f_p}{\partial \eta_{int}} \boldsymbol{\eta}_{int} \frac{\partial f_p}{\partial \eta_{int}}^T + \frac{\partial f_{\mathbf{b}^a}}{\partial \eta_{init}^a} \boldsymbol{\eta}_{init11} \frac{\partial f_{\mathbf{b}^a}}{\partial \eta_{init}^a}^T & 0 & 0 & 0 \\ 0 & 0 & \frac{\partial f_v}{\partial \eta_{init}^a} \boldsymbol{\eta}_{init11} \frac{\partial f_v}{\partial \eta_{init}^a}^T & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \\
& \begin{bmatrix} 0 & \frac{\partial f_{\mathbf{R}}}{\partial \eta_{init}^g} \boldsymbol{\eta}_{init21} \frac{\partial f_p}{\partial \eta_{init}^g}^T & \frac{\partial f_{\mathbf{R}}}{\partial \eta_{init}^g} \boldsymbol{\eta}_{init21} \frac{\partial f_v}{\partial \eta_{init}^g}^T & 0 & 0 \\ \frac{\partial f_p}{\partial \eta_{init}^a} \boldsymbol{\eta}_{init12} \frac{\partial f_{\mathbf{R}}}{\partial \eta_{init}^g}^T & 0 & \frac{\partial f_p}{\partial \eta_{init}^a} \boldsymbol{\eta}_{init12} \frac{\partial f_v}{\partial \eta_{init}^g}^T + \frac{\partial f_p}{\partial \eta_{init}^a} \boldsymbol{\eta}_{init11} \frac{\partial f_p}{\partial \eta_{init}^a}^T & 0 & 0 \\ \frac{\partial f_v}{\partial \eta_{init}^a} \boldsymbol{\eta}_{init12} \frac{\partial f_{\mathbf{R}}}{\partial \eta_{init}^g}^T & \frac{\partial f_v}{\partial \eta_{init}^a} \boldsymbol{\eta}_{init12} \frac{\partial f_p}{\partial \eta_{init}^g}^T + \frac{\partial f_v}{\partial \eta_{init}^a} \boldsymbol{\eta}_{init11} \frac{\partial f_p}{\partial \eta_{init}^g}^T & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{A.18}
\end{aligned}$$

Now we wish to understand the difference between the ImuFactor and CombinedImuFactor. If we set $\boldsymbol{\eta}_{init} = \mathbf{0}_{6 \times 6}$ and note that $\frac{\partial f_p}{\partial \eta_{int}} = \mathbf{I}$, the previous matrix reduces to:

$$\begin{aligned}
\hat{\mathbf{G}}_t \hat{\mathbf{Q}}_t \hat{\mathbf{G}}_t^T = & \begin{bmatrix} \frac{\partial f_{\mathbf{R}}}{\partial \eta^{gd}} \boldsymbol{\eta}^{gd} \frac{\partial f_{\mathbf{R}}}{\partial \eta^{gd}}^T & 0 & 0 & 0 & 0 \\ 0 & \frac{\partial f_p}{\partial \eta^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_p}{\partial \eta^{ad}}^T + \boldsymbol{\eta}_{int} & \frac{\partial f_p}{\partial \eta^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_v}{\partial \eta^{ad}}^T & 0 & 0 \\ 0 & \frac{\partial f_v}{\partial \eta^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_p}{\partial \eta^{ad}}^T & \frac{\partial f_v}{\partial \eta^{ad}} \boldsymbol{\eta}^{ad} \frac{\partial f_v}{\partial \eta^{ad}}^T & 0 & 0 \\ 0 & 0 & 0 & \boldsymbol{\eta}^{bad} & 0 \\ 0 & 0 & 0 & 0 & \boldsymbol{\eta}^{bgd} \end{bmatrix}, \tag{A.19}
\end{aligned}$$

which, in the top left 9x9 block, is equivalent to $\tilde{\mathbf{G}}_t \tilde{\mathbf{Q}}_t \tilde{\mathbf{G}}_t^T$ of the ImuFactor.

Since we have already shown that $\mathbf{F}$ is equivalent for both factors we can now say that the top left 9x9 block of $\hat{\mathbf{P}}$ is equivalent to $\tilde{\mathbf{P}}$ if $\boldsymbol{\eta}_{init} = \mathbf{0}_{6 \times 6}$.

A.3 Comparison of ImuFactor and CombinedImuFactor

The ImuFactor contains a residual for orientation $r_{\Delta\mathbf{R}}$, position and velocity $r_{\Delta pv}$ which are conditioned on the covariance $\tilde{\mathbf{P}}$. In the final cost, a residual for the bias $r_{\Delta b}$ is added separately and conditioned on the bias random walk noise $\text{diag}(\boldsymbol{\eta}^{bad}, \boldsymbol{\eta}^{bgd})$. Put together, the cost for the ImuFactor C_{imu} is:

$$C_{imu} = \begin{bmatrix} r_{\Delta\mathbf{R}} \\ r_{\Delta pv} \end{bmatrix}^T \begin{bmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{bmatrix} \begin{bmatrix} r_{\Delta\mathbf{R}} \\ r_{\Delta pv} \end{bmatrix} + r_{\Delta b}^T S_b r_{\Delta b} \quad (\text{A.20})$$

where we use the symbols S_{XX} to indicate matrix blocks of $\tilde{\mathbf{P}}$. If the matrices are multiplied through the cost becomes:

$$C_{imu} = r_{\Delta\mathbf{R}}^T S_{11} r_{\Delta\mathbf{R}} + r_{\Delta pv}^T S_{21} r_{\Delta\mathbf{R}} + r_{\Delta\mathbf{R}}^T S_{12} r_{\Delta pv} + r_{\Delta pv}^T S_{22} r_{\Delta pv} + r_{\Delta b}^T S_b r_{\Delta b} \quad (\text{A.21})$$

The CombinedImuFactor uses the same residuals but they are jointly conditioned on the covariance $\hat{\mathbf{P}}$. This can be written out as:

$$\begin{aligned} C_{combined} &= \begin{bmatrix} r_{\Delta\mathbf{R}} \\ r_{\Delta pv} \\ r_{\Delta b} \end{bmatrix}^T \begin{bmatrix} S_{11} & S_{12} & S_{13} \\ S_{21} & S_{22} & S_{23} \\ S_{31} & S_{32} & S_{33} \end{bmatrix} \begin{bmatrix} r_{\Delta\mathbf{R}} \\ r_{\Delta pv} \\ r_{\Delta b} \end{bmatrix} \\ &= r_{\Delta\mathbf{R}}^T S_{11} r_{\Delta\mathbf{R}} + r_{\Delta pv}^T S_{21} r_{\Delta\mathbf{R}} + r_{\Delta b}^T S_{13} r_{\Delta\mathbf{R}} + \\ &\quad r_{\Delta\mathbf{R}}^T S_{12} r_{\Delta pv} + r_{\Delta pv}^T S_{22} r_{\Delta pv} + r_{\Delta b}^T S_{23} r_{\Delta pv} + \\ &\quad r_{\Delta\mathbf{R}}^T S_{31} r_{\Delta b} + r_{\Delta pv}^T S_{32} r_{\Delta b} + r_{\Delta b}^T S_{33} r_{\Delta b} \end{aligned} \quad (\text{A.22})$$

We have proven in the previous section that the top left 9x9 block of $\hat{\mathbf{P}}$ is equivalent to $\tilde{\mathbf{P}}$ if $\boldsymbol{\eta}_{init} = 0_{6 \times 6}$. Therefore we know S_{11} , S_{12} , S_{21} and S_{22} are the same in (A.21) and (A.22). Also, from inspection of $\hat{\mathbf{G}}_t \hat{\mathbf{Q}}_t \hat{\mathbf{G}}_t^T$ it is clear that $S_b = S_{33}$.

Now we will prove the costs are not equal. We set the costs to equal and cancel out and equivalent terms:

$$\begin{aligned} C_{imu} &= C_{combined} \\ 0 &= r_{\Delta b}^T S_{13} r_{\Delta\mathbf{R}} + r_{\Delta b}^T S_{23} r_{\Delta pv} + \\ &\quad r_{\Delta\mathbf{R}}^T S_{31} r_{\Delta b} + r_{\Delta pv}^T S_{32} r_{\Delta b} \end{aligned} \quad (\text{A.23})$$

From inspecting $\mathbf{F}$ and $\hat{\mathbf{G}}_t \hat{\mathbf{Q}}_t \hat{\mathbf{G}}_t^T$ we can see that S_{31} and S_{32} will always be zero. This leaves S_{13} and S_{23} which relate to the terms in $\mathbf{F}$: $\frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{b}_t^g}$, $\frac{\partial f_p}{\partial \Delta \mathbf{b}_t^a}$, $\frac{\partial f_v}{\partial \Delta \mathbf{b}_t^a}$.

Therefore the two factors are not entirely equivalent due to the extra cost on the relation between IMU biases and the state.

$$0 \neq r_{\Delta b}^T S_{13} r_{\Delta \mathbf{R}} + r_{\Delta pv}^T S_{23} r_{\Delta b}$$

$$\therefore \tag{A.24}$$

$$C_{combined} = C_{imu} + r_{\Delta b}^T S_{13} r_{\Delta \mathbf{R}} + r_{\Delta pv}^T S_{23} r_{\Delta b}$$

where

$$S_{13} = N \frac{\partial f_{\mathbf{R}}}{\partial \Delta \mathbf{b}_t^g} \boldsymbol{\eta}^{bgd}$$

$$S_{23} = N \begin{bmatrix} \frac{\partial f_p}{\partial \Delta \mathbf{b}_t^a} \\ \frac{\partial f_v}{\partial \Delta \mathbf{b}_t^a} \end{bmatrix} \boldsymbol{\eta}^{bad}, \tag{A.25}$$

and N is the number of IMU measurements integrated.

References

- [1] Sameer Agarwal and Keir Mierle. *Ceres Solver*. Accessed 15 March, 2023. URL: <http://ceres-solver.org>.
- [2] Yasin Almalioglu, Mehmet Turan, Muhamad Risqi U. Saputra, Pedro P.B. de Gusmão, Andrew Markham, and Niki Trigoni. “SelfVIO: Self-supervised deep monocular Visual-Inertial Odometry and depth estimation”. In: *Neural Networks* 150 (2022), pp. 119–136.
- [3] Michael Angermann and Patrick Robertson. “FootSLAM: Pedestrian Simultaneous Localization and Mapping Without Exteroceptive Sensors Hitchhiking on Human Perception and Cognition”. In: *Proceedings of the IEEE* vol 100. no. Special Centennial Issue (2012), pp. 1840–1848.
- [4] Timothy D. Barfoot. *State Estimation for Robotics*. Cambridge University Press, 2017.
- [5] Jakub Bednarek, Michal Bednarek, Lorenz Wellhausen, Marco Hutter, and Krzysztof Walas. “What am I touching? Learning to classify terrain via haptic sensing”. In: *International Conference on Robotics and Automation (ICRA)*. 2019, pp. 7187–7193.
- [6] Michael Bloesch, Michael Burri, Sammy Omari, Marco Hutter, and Roland Siegwart. “Iterated extended Kalman filter based visual-inertial odometry using direct photometric feedback”. In: *The International Journal of Robotics Research* 36.10 (2017), pp. 1053–1072.
- [7] Michael Bloesch, Michael Burri, Hannes Sommer, Roland Siegwart, and Marco Hutter. “The Two-State Implicit Filter Recursive Estimation for Mobile Robots”. In: *IEEE Robotics and Automation Letters* 3.1 (2018), pp. 573–580.
- [8] Michael Bloesch et al. “State estimation for legged robots: Consistent fusion of leg kinematics and IMU”. In: *Robotics: Science and Systems* 8 (2013), pp. 17–24.
- [9] Fan Bo, Jia Li, and Weibing Wang. “Mode-Independent Stride Length Estimation With IMUs in Smartphones”. In: *IEEE Sensors Journal* 22.6 (2022), pp. 5824–5833.
- [10] Martin Brossard, Axel Barrau, and Silvére Bonnabel. “AI-IMU Dead-Reckoning”. In: *IEEE Transactions on Intelligent Vehicles* 5.4 (2020), pp. 585–595.
- [11] Martin Brossard, Silvére Bonnabel, and Axel Barrau. “Denoising IMU Gyroscopes With Deep Learning for Open-Loop Attitude Estimation”. In: *IEEE Robotics and Automation Letters* 5.3 (2020), pp. 4796–4803.
- [12] Russell Buchanan. *Allan Variance ROS*. Accessed 15 March, 2023. URL: https://github.com/ori-drs/allan_variance_ros.

- [13] Russell Buchanan, Varun Agrawal, Marco Camurri, Frank Dellaert, and Maurice Fallon. “Deep IMU Bias Inference for Robust Visual-Inertial Odometry With Factor Graphs”. In: *IEEE Robotics and Automation Letters* 8.1 (2023), pp. 41–48.
- [14] Russell Buchanan, Jakub Bednarek, Marco Camurri, Michał R. Nowicki, Krzysztof Walas, and Maurice Fallon. “Navigating by touch: haptic Monte Carlo localization via geometric sensing and terrain classification”. In: *Autonomous Robots* 45.6 (2021), pp. 843–857.
- [15] Russell Buchanan, Marco Camurri, Frank Dellaert, and Maurice Fallon. “Learning Inertial Odometry for Dynamic Legged Robot State Estimation”. In: *Proceedings of the 5th Conference on Robot Learning*. Vol. 164. 2021, pp. 1575–1584.
- [16] Russell Buchanan, Marco Camurri, and Maurice Fallon. “Haptic Sequential Monte Carlo Localization for Quadrupedal Locomotion in Vision-Denied Scenarios”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 3657–3663.
- [17] Russell Buchanan, Shih-fan Jack Tu, Marco Camurri, Stephen J. Mellon, and Maurice Fallon. *Markerless Handheld Ultrasound 3D Reconstruction of Bone using Visual-Inertial and Deep Inertial Odometry*. Draft Paper. 2023.
- [18] Russell Buchanan, Lorenz Wellhausen, Marko Bjelonic, Tirthankar Bandyopadhyay, Navinda Kottege, and Marco Hutter. “Perceptive whole-body planning for multilegged robots in confined spaces”. In: *Journal of Field Robotics* 38.1 (2021), pp. 68–84.
- [19] Cesar Cadena et al. “Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age”. In: *IEEE Transactions on Robotics* 32.6 (2016), pp. 1309–1332.
- [20] Carlos Campos, Richard Elvira, Juan J. Gómez, José M. M. Montiel, and Juan D. Tardós. “ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM”. In: *IEEE Transactions on Robotics* 37.6 (2021), pp. 1874–1890.
- [21] Marco Camurri, Milad Ramezani, Simona Nobili, and Maurice Fallon. “Pronto: A Multi-Sensor State Estimator for Legged Robots in Real-World Scenarios”. In: *Frontiers in Robotics and AI* 7 (2020).
- [22] Marco Camurri et al. “Probabilistic Contact Estimation and Impact Detection for State Estimation of Quadruped Robots”. In: *IEEE Robotics and Automation Letters* 2.2 (2017), pp. 1023–1030.
- [23] Maxime Chalon, Jens Reinecke, and Martin Pfanne. “Online in-hand object localization”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2013, pp. 2977–2984.
- [24] Changhao Chen, Xiaoxuan Lu, Andrew Markham, and Niki Trigoni. “IONet: Learning to Cure the Curse of Drift in Inertial Odometry”. In: *32nd AAAI Conference on Artificial Intelligence, AAAI 2018* (2018), pp. 6468–6476.
- [25] Changhao Chen, Stefano Rosa, Chris Xiaoxuan Lu, Bing Wang, Niki Trigoni, and Andrew Markham. “Learning Selective Sensor Fusion for State Estimation”. In: *IEEE Transactions on Neural Networks and Learning Systems* (2022), pp. 1–15.

- [26] Changhao Chen et al. “MotionTransformer: Transferring Neural Inertial Tracking between Domains”. In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01 (2019), pp. 8009–8016.
- [27] Changhao Chen et al. “Selective Sensor Fusion for Neural Visual-Inertial Odometry”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 10534–10543.
- [28] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. “Neural Ordinary Differential Equations”. In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Vol. 31. Curran Associates, Inc., 2018.
- [29] Sachin Chitta et al. “Proprioceptive localization for a quadrupedal robot on known terrain”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2007, pp. 4582–4587.
- [30] Giovanni Cioffi, Leonard Bauersfeld, Elia Kaufmann, and Davide Scaramuzza. “Learned Inertial Odometry for Autonomous Drone Racing”. In: (2022). URL: <https://arxiv.org/abs/2210.15287>.
- [31] Ronald Clark, Sen Wang, Hongkai Wen, Andrew Markham, and Niki Trigoni. “VINet: Visual-Inertial Odometry as a Sequence-to-Sequence Learning Problem”. In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. 2017, pp. 3995–4001.
- [32] Troy P. Cordie et al. “Modular field robot deployment for inspection of dilapidated buildings”. In: *Journal of Field Robotics* 36.4 (2019), pp. 641–655.
- [33] Andrea De Maio and Simon Lacroix. “Deep Bayesian ICP Covariance Estimation”. In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022, pp. 6519–6525.
- [34] Frank Dellaert. *Factor Graphs and GTSAM: A Hands-on Introduction*. Tech. rep. September. Georgia Institute of Technology, 2012. URL: <https://smartech.gatech.edu/handle/1853/45226>.
- [35] Frank Dellaert and Varun Agrawal. *The New IMU Factor*. Tech. rep. Georgia Institute of Technology, 2022. URL: <https://github.com/borglab/gtsam/blob/develop/doc/ImuFactor.pdf>.
- [36] Frank Dellaert and GTSAM Contributors. *borglab/gtsam*. Version 4.2. May 2022. URL: <https://github.com/borglab/gtsam>.
- [37] Frank Dellaert and Michael Kaess. “Factor Graphs for Robot Perception”. In: *Foundations and Trends in Robotics* 6.1-2 (2017).
- [38] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. “SuperPoint: Self-Supervised Interest Point Detection and Description”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (2018), pp. 337–33712.
- [39] Kevin J. Doherty, Ziqi Lu, Kurran Singh, and John J. Leonard. “Discrete-Continuous Smoothing and Mapping”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 12395–12402.

- [40] Kevin Eickenhoff, Patrick Geneva, and Guoquan Huang. “Closed-form preintegration methods for graph-based visual-inertial navigation”. In: *The International Journal of Robotics Research* 38.5 (2019), pp. 563–586.
- [41] Jakob Engel, Vladlen Koltun, and Daniel Cremers. “Direct Sparse Odometry”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.3 (2018), pp. 611–625.
- [42] Jakob Engel, Jörg Stückler, and Daniel Cremers. “Large-scale direct SLAM with stereo cameras”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 1935–1942.
- [43] Maurice F. Fallon, Matthew Antone, Nicholas Roy, and Seth Teller. “Drift-free humanoid state estimation fusing kinematic, inertial and LIDAR sensing”. In: *14th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. 2014, pp. 112–119.
- [44] Jay Farrell. *Aided Navigation: GPS with High Rate Sensors*. 1st ed. USA: McGraw-Hill, Inc., 2008.
- [45] Jay A. Farrell, Felipe O. Silva, Farzana Rahman, and Jan Wendel. “Inertial Measurement Unit Error Modeling Tutorial: Inertial Navigation System State Estimation with Real-Time Sensor Calibration”. In: *IEEE Control Systems Magazine* 42.6 (2022), pp. 40–66.
- [46] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. “On-Manifold Preintegration for Real-Time Visual-Inertial Odometry”. In: *IEEE Transactions on Robotics* 33.1 (2017), pp. 1–21.
- [47] Christian Forster, Zichao Zhang, Michael Gassner, Manuel Werlberger, and Davide Scaramuzza. “SVO: Semidirect Visual Odometry for Monocular and Multicamera Systems”. In: *IEEE Transactions on Robotics* 33.2 (2017), pp. 249–265.
- [48] Dieter Fox, Sebastian Thrun, Wolfram Burgard, and Frank Dellaert. “Particle Filters for Mobile Robot Localization”. In: *Sequential Monte Carlo Methods in Practice*. Ed. by Arnaud Doucet, Nando de Freitas, and Neil Gordon. New York, NY: Springer New York, 2001, pp. 401–428.
- [49] Eric Foxlin. “Pedestrian tracking with shoe-mounted inertial sensors”. In: *IEEE Computer Graphics and Applications* 25.6 (2005), pp. 38–46.
- [50] Paul Furgale and Timothy D. Barfoot. “Visual teach and repeat for long-range rover autonomy”. In: *Journal of Field Robotics* 27.5 (2010), pp. 534–560.
- [51] Paul Furgale, Joern Rehder, and Roland Siegwart. “Unified temporal and spatial calibration for multi-sensor systems”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2013, pp. 1280–1286.
- [52] Christian. Gehring et al. “ANYmal in the Field: Solving Industrial Inspection of an Offshore HVDC Platform with a Quadrupedal Robot”, booktitle="Field and Service Robotics". In: ed. by Genya Ishigami and Kazuya Yoshida. Singapore: Springer, 2021, pp. 247–260.

- [53] Patrick Geneva, Kevin Ekenhoff, Woosik Lee, Yulin Yang, and Guoquan Huang. “OpenVINS: A Research Platform for Visual-Inertial Estimation”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 4666–4672.
- [54] Mariia Gladkova, Rui Wang, Niclas Zeller, and Daniel Cremers. “Tight Integration of Feature-based Relocalization in Monocular Direct Visual Odometry”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 9608–9614.
- [55] Mohinder Grewal and Angus Andrews. “How Good Is Your Gyro [Ask the Experts]”. In: *IEEE Control Systems Magazine* 30.1 (2010), pp. 12–86.
- [56] Michael Grupp. *evo: Python package for the evaluation of odometry and SLAM*. <https://github.com/MichaelGrupp/evo>. 2017.
- [57] Albert Gu, Karan Goel, and Christopher Ré. “Efficiently Modeling Long Sequences with Structured State Spaces”. In: *International Conference on Learning Representations*. 2022. URL: <https://openreview.net/forum?id=uYLFoz1v1AC>.
- [58] Sami Haddadin, Alessandro De Luca, and Alin Albu-Schaffer. “Robot Collisions: A Survey on Detection, Isolation, and Identification”. In: *IEEE Transactions on Robotics* 33.6 (2017), pp. 1292–1312.
- [59] Ross Hartley, Maani Ghaffari, Ryan M Eustice, and Jessy W Grizzle. “Contact-aided invariant extended Kalman filtering for robot state estimation”. In: *The International Journal of Robotics Research* 39.4 (2020), pp. 402–430.
- [60] Ross Hartley, Maani Ghaffari Jadidi, Lu Gan, Jiunn Kai Huang, Jessy W. Grizzle, and Ryan M. Eustice. “Hybrid Contact Preintegration for Visual-Inertial-Contact State Estimation Using Factor Graphs”. In: *IEEE International Conference on Intelligent Robots and Systems (IROS)* (2018), pp. 3783–3790.
- [61] Ross Hartley et al. “Legged Robot State-Estimation Through Combined Forward Kinematic and Preintegrated Contact Factors”. In: *IEEE International Conference on Robotics and Automation (ICRA)* (2018), pp. 4422–4429.
- [62] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.
- [63] Sachini Herath, David Caruso, Chen Liu, Yufan Chen, and Yasutaka Furukawa. “Neural Inertial Localization”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 6604–6613.
- [64] Sachini Herath, Hang Yan, and Yasutaka Furukawa. “RoNIN: Robust Neural Inertial Navigation in the Wild: Benchmark, Evaluations, & New Methods”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 3146–3152.
- [65] Sepp Hochreiter and Jürgen Schmidhuber. “Long short-term memory”. In: *Neural computation* 9.8 (1997), pp. 1735–1780.
- [66] Mark A Hoepflinger, C David Remy, Marco Hutter, and Roland Siegwart. “Haptic Terrain Classification on Natural Terrains for Legged Robots”. In: *International Conference on Climbing and Walking Robots (CLAWAR)*. 2010, pp. 785–792.

- [67] Francois R. Hogan, Jose Ballester, Siyuan Dong, and Alberto Rodriguez. “Tactile Dexterity: Manipulation Primitives with Tactile Feedback”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 8863–8869.
- [68] Yinghao Huang, Manuel Kaufmann, Emre Aksan, Michael J. Black, Otmar Hilliges, and Gerard Pons-Moll. “Deep Inertial Poser: Learning to Reconstruct Human Pose from Sparse Inertial Measurements in Real Time”. In: *ACM Trans. Graph.* 37.6 (2018).
- [69] Marco Hutter et al. “ANYmal - A Highly Mobile and Dynamic Quadrupedal Robot”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 38–44.
- [70] “IEEE Standard Specification Format Guide and Test Procedure for Linear, Single-Axis, Non-Gyroscopic Accelerometers”. In: *IEEE Std 1293-1998* (1999), pp. 1–252. DOI: 10.1109/IEEESTD.1999.89857.
- [71] “IEEE Standard Specification Format Guide and Test Procedure for Linear Single-Axis, Nongyroscopic Accelerometers”. In: *IEEE Std 1293-2018 (Revision of IEEE Std 1293-1998)* (2019), pp. 1–271. DOI: 10.1109/IEEESTD.2019.8653544.
- [72] “IEEE Standard Specification Format Guide and Test Procedure for Single-Axis Interferometric Fiber Optic Gyros”. In: *IEEE Std 952-1997* (1998), pp. 1–84. DOI: 10.1109/IEEESTD.1998.86153.
- [73] Fabian Jenelten, Jemin Hwangbo, Fabian Tresoldi, C. Dario Bellicoso, and Marco Hutter. “Dynamic Locomotion on Slippery Ground”. In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 4170–4176.
- [74] Gwanghyeon Ji, Juhyeok Mun, Hyeongjun Kim, and Jemin Hwangbo. “Concurrent Training of a Control Policy and a State Estimator for Dynamic and Robust Legged Locomotion”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 4630–4637.
- [75] Rui Jia, Paul Monk, David Murray, J Alison Noble, and Stephen Mellon. “CAT & MAUS: A novel system for true dynamic motion measurement of underlying bony structures with compensation for soft tissue movement”. In: *J Biomech* 62 (2017), pp. 156–164.
- [76] Fan Jiang, Varun Agrawal, Russell Buchanan, Maurice Fallon, and Frank Dellaert. *iMHS: An Incremental Multi-Hypothesis Smoother*. 2021. URL: <https://arxiv.org/abs/2103.13178>.
- [77] Rico Jonschkowski, Divyam Rastogi, and Oliver Brock. “Differentiable Particle Filters: End-to-End Learning with Algorithmic Priors”. In: *Robotics: Science and Systems*. 2018.
- [78] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John J. Leonard, and Frank Dellaert. “ISAM2: Incremental smoothing and mapping using the Bayes tree”. In: *International Journal of Robotics Research* 31.2 (2012), pp. 216–235.
- [79] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. “iSAM: Incremental Smoothing and Mapping”. In: *IEEE Transactions on Robotics* 24.6 (2008), pp. 1365–1378.

- [80] Rudolf. E. Kalman. “A New Approach to Linear Filtering and Prediction Problems”. In: *Journal of Basic Engineering* 82.1 (1960), pp. 35–45.
- [81] Joon-Ha Kim et al. “Legged Robot State Estimation With Dynamic Contact Event Information”. In: *IEEE Robotics and Automation Letters* 6.4 (2021), pp. 6733–6740.
- [82] Tobias Klamt et al. “Flexible Disaster Response of Tomorrow: Final Presentation and Evaluation of the CENTAURO System”. In: *IEEE Robotics & Automation Magazine* 26.4 (2019), pp. 59–72.
- [83] Georg Klein and David Murray. “Parallel Tracking and Mapping for Small AR Workspaces”. In: *6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. 2007, pp. 225–234.
- [84] Hendrik Kolvenbach, Christian Bärtschi, Lorenz Wellhausen, Ruben Grandia, and Marco Hutter. “Haptic Inspection of Planetary Soils With Legged Robots”. In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1626–1632.
- [85] Hendrik Kolvenbach et al. “Towards autonomous inspection of concrete deterioration in sewers with legged robots”. In: *Journal of Field Robotics* 37.8 (2020), pp. 1314–1327.
- [86] Navinda Kottege, Luis Sentis, and Dimitrios Kanoulas. “Editorial: Towards Real-World Deployment of Legged Robots”. In: *Frontiers in Robotics and AI* 8 (2022).
- [87] Eric Krotkov. “Active perception for legged locomotion: every step is an experiment”. In: *5th IEEE International Symposium on Intelligent Control*. 1990, 227–232 vol.1.
- [88] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. “G2o: A general framework for graph optimization”. In: *2011 IEEE International Conference on Robotics and Automation*. 2011, pp. 3607–3613.
- [89] Rainer Kümmerle et al. “On measuring the accuracy of SLAM algorithms”. In: *Autonomous Robots* 27.4 (2009), pp. 387–407.
- [90] Cedric Le Gentil and Teresa Vidal-Calleja. “Continuous Integration over $SO(3)$ for IMU Preintegration”. In: *Robotics: Science and Systems*. 2021.
- [91] Cedric Le Gentil, Teresa Vidal-calleja, and Shoudong Huang. “Gaussian Process Preintegration for Inertial-Aided State Estimation”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 2108–2114.
- [92] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. “Learning quadrupedal locomotion over challenging terrain”. In: *Science Robotics* 5.47 (2020).
- [93] Stefan Leutenegger, Simon Lynen, Michael Bosse, Roland Siegwart, and Paul Furgale. “Keyframe-based visual-inertial odometry using nonlinear optimization”. In: *The International Journal of Robotics Research* 34.3 (2015), pp. 314–334.
- [94] Pei-chun Lin, Haldun Komsuo, and Daniel E Koditschek. “Sensor Data Fusion for Body State Estimation in a Hexapod Robot With Dynamical Gaits”. In: *IEEE Transactions on Robotics* 22.5 (2006), pp. 932–943.

- [95] Tzu-Yuan Lin, Ray Zhang, Justin Yu, and Maani Ghaffari. “Legged Robot State Estimation using Invariant Kalman Filtering and Learned Contact Events”. In: *Proceedings of the 5th Conference on Robot Learning*. Vol. 164. 2021, pp. 1057–1066.
- [96] Wenxin Liu et al. “TLIO: Tight Learned Inertial Odometry”. In: *IEEE Robotics and Automation Letters* 5.4 (2020), pp. 5653–5660.
- [97] Todd Lupton and Salah Sukkarieh. “Visual-Inertial-Aided Navigation for High-Dynamic Motion in Built Environments Without Initial Conditions”. In: *IEEE Transactions on Robotics* 28.1 (2012), pp. 61–76.
- [98] Mikołaj Łysakowski, Michał R. Nowicki, Russell Buchanan, Marco Camurri, Maurice Fallon, and Krzysztof Walas. “Unsupervised Learning of Terrain Representations for Haptic Monte Carlo Localization”. In: *International Conference on Robotics and Automation (ICRA)*. 2022, pp. 4642–4648.
- [99] Perla Maiolino, Marco Maggiali, Giorgio Cannata, Giorgio Metta, and Lorenzo Natale. “A flexible and robust large scale capacitive tactile system for robots”. In: *IEEE Sensors Journal* 13.10 (2013), pp. 3910–3917.
- [100] Lucas Manuelli and Russ Tedrake. “Localizing external contact using proprioceptive sensors: The Contact Particle Filter”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 5062–5069.
- [101] Agostino Martinelli. “Closed-Form Solution of Visual-Inertial Structure from Motion”. In: *International Journal of Computer Vision* 106.2 (2014), pp. 138–152.
- [102] Ruben Mascaro, Martin Wermelinger, Marco Hutter, and Margarita Chli. “Towards automating construction tasks: Large-scale object mapping, segmentation, and manipulation”. In: *Journal of Field Robotics* 38.5 (2021), pp. 684–699.
- [103] Leonard A. McGee and Stanley F. Schmidt. *Discovery of the Kalman filter as a practical tool for aerospace and industry*. Tech. rep. NASA-TM-86847, N86-13311. National Aeronautics and Space Administration, 1985.
- [104] D Montgomery. “The Pentagon’s \$82 Million Super Bowl of Robots”. In: *The Washington Post Magazine* (2021). URL: <https://www.washingtonpost.com/magazine/2021/11/10/darpa-robot-competition/>.
- [105] Anastasios I. Mourikis and Stergios I. Roumeliotis. “A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2007, pp. 3565–3572.
- [106] Elias Mueggler, Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. “Continuous-Time Visual-Inertial Odometry for Event Cameras”. In: *IEEE Transactions on Robotics* 34.6 (2018), pp. 1425–1440.
- [107] Raúl Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. “ORB-SLAM: a Versatile and Accurate Monocular SLAM System”. In: *IEEE Transactions on Robotics* 31.5 (2015), pp. 1147–1163.
- [108] Raúl Mur-Artal and Juan D. Tardós. “ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras”. In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1255–1262.

- [109] Michael P. Murphy, Aaron Saunders, Cassie Moreira, Alfred A. Rizzi, and Marc Raibert. “The LittleDog robot”. In: *International Journal of Robotics Research* 30.2 (2011), pp. 145–149.
- [110] Levon N. Nazarian. “The Top 10 Reasons Musculoskeletal Sonography Is an Important Complementary or Alternative Technique to MRI”. In: *American Journal of Roentgenology* 190.6 (2008), pp. 1621–1626.
- [111] Chanoh Park, Peyman Moghadam, Jason L. Williams, Soohwan Kim, Sridha Sridharan, and Clinton Fookes. “Elasticity Meets Continuous-Time: Map-Centric Dense 3D LiDAR SLAM”. In: *IEEE Transactions on Robotics* 38.2 (2022), pp. 978–997.
- [112] Stylianos Piperakis, Dimitrios Kanoulas, Nikos G. Tsagarakis, and Panos Trahanias. “Outlier-Robust State Estimation for Humanoid Robots”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 706–713.
- [113] Stylianos Piperakis, Michael Maravgakis, Dimitrios Kanoulas, and Panos Trahanias. “Robust Contact State Estimation in Humanoid Walking Gaits”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022, pp. 6732–6738.
- [114] François Pomerleau, Francis Colas, and Roland Siegwart. “A Review of Point Cloud Registration Algorithms for Mobile Robotics”. In: *Foundations and Trends in Robotics* 4 (May 2015), pp. 1–104.
- [115] Tong Qin, Peiliang Li, and Shaojie Shen. “VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator”. In: *IEEE Transactions on Robotics* 34.4 (2018), pp. 1004–1020.
- [116] Milad Ramezani, Kouros Khoshelham, and Laurent Kneip. “Omnidirectional visual-inertial odometry using multi-state constraint Kalman filter”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 1317–1323.
- [117] Milad Ramezani, Yiduo Wang, Marco Camurri, David Wisth, Matias Mattamala, and Maurice 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, pp. 4353–4360.
- [118] Joern Rehder, Janosch Nikolic, Thomas Schneider, Timo Hinzmann, and Roland Siegwart. “Extending kalibr: Calibrating the extrinsics of multiple IMUs and of individual axes”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 4304–4311.
- [119] Daniel Roetenberg. “Inertial and Magnetic Sensing of Human Motion”. PhD thesis. University of Twente, 2006.
- [120] Edward Rosten, Reid Porter, and Tom Drummond. “Faster and Better: A Machine Learning Approach to Corner Detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.1 (2010), pp. 105–119.
- [121] G.P. Roston and E.P. Krotkov. “Dead Reckoning Navigation For Walking Robots”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Vol. 1. 1992, pp. 607–612.

- [122] Simo Särkkä. *Bayesian Filtering and Smoothing*. Institute of Mathematical Statistics Textbooks. Cambridge University Press, 2013.
- [123] Martin Schepers, Matteo Giuberti, and Giovanni Bellusci. *Xsens MVN: Consistent Tracking of Human Motion Using Inertial Sensing*. Tech. rep. White Paper. Xsens Technologies B.V. URL: <https://content.xsens.com/mvn-whitepaper> (visited on 10/25/2022).
- [124] Thomas Schneider et al. “Maplab: An Open Framework for Research in Visual-Inertial Mapping and Localization”. In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 1418–1425.
- [125] Jakob Schwendner, Sylvain Joyeux, and Frank Kirchner. “Using Embodied Data for Localization and Mapping”. In: *Journal of Field Robotics* 31.2 (2014), pp. 263–295.
- [126] Roland Siegwart, Illah R. Nourbakhsh, and Davide Scaramuzza. *Introduction to Autonomous Mobile Robots*. 2nd. The MIT Press, 2011.
- [127] Dan Simon. *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. USA: Wiley-Interscience, 2006.
- [128] Isaac Skog, John Olof Nilsson, and Peter Händel. “Evaluation of zero-velocity detectors for foot-mounted inertial navigation systems”. In: *International Conference on Indoor Positioning and Indoor Navigation (IPIN)* (2010), pp. 1–6.
- [129] Jan Steinbrener, Christian Brommer, Thomas Jantos, Alessandro Fornasier, and Stephan Weiss. “Improved State Propagation through AI-based Pre-processing and Down-sampling of High-Speed Inertial Data”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2022.
- [130] Lukas von Stumberg and Daniel Cremers. “DM-VIO: Delayed Marginalization Visual-Inertial Odometry”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 1408–1415.
- [131] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. “A benchmark for the evaluation of RGB-D SLAM systems”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2012, pp. 573–580.
- [132] Maximilian Stölzle, Takahiro Miki, Levin Gerdes, Martin Azkarate, and Marco Hutter. “Reconstructing Occluded Elevation Information in Terrain Maps With Self-Supervised Learning”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 1697–1704.
- [133] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. “iMAP: Implicit Mapping and Positioning in Real-Time”. In: *IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 6209–6218.
- [134] Zachary Teed and Jia Deng. “DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan. 2021. URL: https://openreview.net/forum?id=ZBfUo_dr4H.

- [135] Eranda Tennakoon, Thierry Peynot, Jonathan Roberts, and Navinda Kottege. “Probe-before-step walking strategy for multi-legged robots on terrain with risk of collapse”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 5530–5536.
- [136] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic robotics*. Intelligent robotics and autonomous agents. Cambridge, Mass. ; London: MIT Press, 2005.
- [137] David Titterton and John Weston. *Strapdown Inertial Navigation Technology*. Institution of Engineering and Technology, 2004.
- [138] Carlo Tomasi and Takeo Kanade. *Detection and Tracking of Point Features*. Tech. rep. Technical Report CMU-CS-91-132. Carnegie Mellon University, 1991.
- [139] Marco Tranzatto et al. “CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge”. In: *Field Robotics 2* (2022), pp. 274–324. DOI: 10.55417/fr.2022011.
- [140] Marco Tranzatto et al. *Team CERBERUS Wins the DARPA Subterranean Challenge: Technical Overview and Lessons Learned*. Available Online. 2022. URL: <https://arxiv.org/abs/2207.04914>.
- [141] Giorgio Valsecchi, Ruben Grandia, and Marco Hutter. “Quadrupedal Locomotion on Uneven Terrain with Sensorized Feet”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pp. 1548–1555.
- [142] Rudolph Van Der Merwe, Arnaud Doucet, Nando De Freitas, and Eric Wan. “The Unscented Particle Filter”. In: *Proceedings of the 13th International Conference on Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, 2000, pp. 563–569.
- [143] Ashish Vaswani et al. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017.
- [144] Giulia Vezzani, Ugo Pattacini, Giorgio Battistelli, Luigi Chisci, and Lorenzo Natale. “Memory Unscented Particle Filter for 6-DOF Tactile Localization”. In: *IEEE Transactions on Robotics* 33.5 (2017), pp. 1139–1155.
- [145] Matthieu Vigne, Antonio El Khoury, Florent Di Meglio, and Nicolas Petit. “State Estimation for a Legged Robot with Multiple Flexibilities Using IMUs: A Kinematic Approach”. In: *IEEE Robotics and Automation Letters* 5.1 (2020), pp. 195–202.
- [146] Brandon Wagstaff, Valentin Peretroukhin, and Jonathan Kelly. “Robust Data-Driven Zero-Velocity Detection for Foot-Mounted Inertial Navigation”. In: *IEEE Sensors Journal* 20.2 (2020), pp. 957–967.
- [147] Brandon Wagstaff, Valentin Peretroukhin, and Jonathan Kelly. “Improving foot-mounted inertial navigation through real-time motion classification”. In: *International Conference on Indoor Positioning and Indoor Navigation (IPIN)* (2017), pp. 1–8.
- [148] Johan Wahlstrom, Andrew Markham, and Niki Trigoni. “FootSLAM meets Adaptive Thresholding”. In: *IEEE Sensors Journal* (2020), pp. 1–1.

- [149] Liqiang Wang, Hailiang Tang, Tisheng Zhang, Qijin Chen, Jinwei Shi, and Xiaoji Niu. “Improving the Navigation Performance of the MEMS IMU Array by Precise Calibration”. In: *IEEE Sensors Journal* 21.22 (2021), pp. 26050–26058.
- [150] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. “DeepVO: Towards end-to-end visual odometry with deep Recurrent Convolutional Neural Networks”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 2043–2050.
- [151] Lorenz Wellhausen, Alexey Dosovitskiy, René Ranftl, Krzysztof Walas, Cesar Cadena, and Marco Hutter. “Where Should I Walk? Predicting Terrain Properties From Images Via Self-Supervised Learning”. In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1509–1516.
- [152] David Wisth. “Multisensor fusion for state estimation via factor graphs”. PhD thesis. University of Oxford, 2021.
- [153] David Wisth, Marco Camurri, Sandipan Das, and Maurice Fallon. “Unified Multi-Modal Landmark Tracking for Tightly Coupled Lidar-Visual-Inertial Odometry”. In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 1004–1011.
- [154] David Wisth, Marco Camurri, and Maurice 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.
- [155] David Wisth, Marco Camurri, and Maurice Fallon. “Robust Legged Robot State Estimation Using Factor Graph Optimization”. In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 4507–4514.
- [156] David Wisth, Marco Camurri, and Maurice Fallon. “VILENS: Visual, Inertial, Lidar, and Leg Odometry for All-Terrain Legged Robots”. In: *IEEE Transactions on Robotics* (2022), pp. 1–18.
- [157] Oliver J. Woodman. *An introduction to inertial navigation*. Tech. rep. UCAM-CL-TR-696. University of Cambridge, Computer Laboratory, 2007.
- [158] Yulin Yang, Benzun Pious Wisely Babu, Chuchu Chen, Guoquan Huang, and Liu Ren. “Analytic Combined IMU Integration (ACI2) For Visual Inertial Navigation”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 4680–4686.
- [159] Xinyu Yi et al. “Physical Inertial Poser (PIP): Physics-aware Real-time Human Motion Tracking from Sparse Inertial Sensors”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022.
- [160] Ji Zhang and Sanjiv Singh. “LOAM: Lidar Odometry and Mapping in Real-time”. In: *Proceedings of Robotics: Science and Systems*. Berkeley, USA, 2014.
- [161] Lintong Zhang, Marco Camurri, and Maurice Fallon. *Multi-Camera LiDAR Inertial Extension to the Newer College Dataset*. 2021. arXiv: 2112.08854.
- [162] Lintong Zhang, David Wisth, Marco Camurri, and Maurice Fallon. “Balancing the Budget: Feature Selection and Tracking for Multi-Camera Visual-Inertial Odometry”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 1182–1189.

- [163] Ming Zhang, Mingming Zhang, Yiming Chen, and Mingyang Li. “IMU Data Processing For Inertial Aided Navigation: A Recurrent Neural Network Based Approach”. In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2021.
- [164] Shibo Zhao, Hengrui Zhang, Peng Wang, Lucas Nogueira, and Sebastian Scherer. “Super Odometry: IMU-centric LiDAR-Visual-Inertial Estimator for Challenging Environments”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2021, pp. 8729–8736.
- [165] Jannik Zürn, Wolfram Burgard, and Abhinav Valada. “Self-Supervised Visual Terrain Classification From Unsupervised Acoustic Feature Learning”. In: *IEEE Transactions on Robotics* 37.2 (2021), pp. 466–481.