Vision-based Legged Robot Navigation: Localisation, Local Planning, Learning

Matías Mattamala Aravena

Hertford College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

2023

Abstract

The recent advances in legged locomotion control have made legged robots walk up staircases, go deep into underground caves, and walk in the forest. Nevertheless, autonomously achieving this task is still a challenge. Navigating and acomplishing missions in the wild relies not only on robust low-level controllers but also higher-level representations and perceptual systems that are aware of the robot's capabilities.

This thesis addresses the navigation problem for legged robots. The contributions are four systems designed to exploit unique characteristics of these platforms, from the sensing setup to their advanced mobility skills over different terrain. The systems address localisation, scene understanding, and local planning, and advance the capabilities of legged robots in challenging environments.

The first contribution tackles localisation with multi-camera setups available on legged platforms. It proposes a strategy to actively switch between the cameras and stay localised while operating in a visual teach and repeat context—in spite of transient changes in the environment. The second contribution focuses on local planning, effectively adding a safety layer for robot navigation. The approach uses a local map built on-the-fly to generate efficient vector field representations that enable fast and reactive navigation. The third contribution demonstrates how to improve local planning in natural environments by learning robot-specific traversability from demonstrations. The approach leverages classical and learning-based methods to enable online, onboard traversability learning. These systems are demonstrated via different robot deployments on industrial facilities, underground mines, and parklands.

The thesis concludes by presenting a real-world application: an autonomous forest inventory system with legged robots. This last contribution presents a mission planning system for autonomous surveying as well as a data analysis pipeline to extract forestry attributes. The approach was experimentally validated in a field campaign in Finland, evidencing the potential that legged platforms offer for future applications in the wild.

Vision-based Legged Robot Navigation: Localisation, Local Planning, Learning

Matías Mattamala Aravena

Hertford College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

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.

Matías Mattamala Aravena, Hertford College

Acknowledgements

My favourite part of theses are the acknowledgments sections. Yes, we spent at least 4 years working on research, published our papers, and attended conferences that are worth reporting on their own. But that is not everything that happened in those years. Acknowledgment sections give you a chance to learn that.

I first thank my supervisor Maurice Fallon for replying to my emails while I was a master's student in Chile, and for giving me the chance to join the Dynamic Robot Systems (DRS) group. I thank the support and advice these 4 years, as well as showing me the importance of working with real robots.

I am grateful to my examiners, Stefan Leutenegger and Ioannis Havoutis, for taking the time to read my work, as well as the interesting discussions and feedback they gave me during the viva. Stefan's expertise in state estimation as well as Ioannis' in legged robots made them a good match to assess the work presented in this thesis. I also thank Jonathan Gammel and Ingmar Posner for assessing my work in the Transfer and Confirmation of Status, and giving me valuable comments to move forward with my research.

I would like to acknowledge all the friends and colleagues in DRS I met over the years. Milad Ramezani for encouraging me to solve the problems now and not later (and helping me to move home!). Marco Camurri for his technical expertise, but also chats on food and *nerdy* things (like The Cave). Russell Buchanan for sharing his experiences and knowledge that helped me so many times. David Rytz for the many technical (and not-so) conversations over a coffee, as well as painful debugging sessions overnight. David Wisth for developing VILENS and all I learned by just checking his code. Georgi Tinchev for our brief chats, and being the only person of DRS I met in person before I applied. Yiduo Wang for the research collaborations but also for sharing our passion for LEGOs. Lintong Zhang for the hard work, the fun, and Korean fried chicken. Ethan Tao for his ML expertise, book recommendations, and long conversations after work. Nived Chebrolu for his support, the laughs, and the pragmatic advice always needed. Tejaswi Digumarti for the wisdom, and nice recommendations when I visited Zurich. Frank Fu for the chance to discuss crazy ideas at a whiteboard, adjoints, and *pollito frito*. Christina Kassab for the collaborations, the socials, and sharing

weird ideas that were worth exploring. Joseph Rowell for the lunches and jokes, but also valuable feedback. Jianeng Wang for the collaborations and example of hard-work. Jonas Beuchart for his advice and feedback on my thesis, but also for sharing the amazing whereabouts of his turtles. Rowan Border for the technical discussions and advice in the last bits of my PhD.

When I started DRS was a huge group, but that changed over the years. Still, there are many people I am thankful to have shared time with. Mathieu Geisert gave me the chance to join my first field trial, but also we shared many chats in pubs after work with Siddhant Gangapurwala, Romeo Orsolino, David Surovik, and Luigi Campanaro, from which I learned many things about robot control, research, and life decisions. Wolfgang Merkt's technical expertise was always valuable when embarking on new software developments. I enjoyed working with Oliwier Melon and Alexander Mitchell (as well as David Rytz) to develop a new CDT challenge in 2021, from scratch, during COVID times, which we successfully executed (I hope).

While I started the PhD in DRS with Russell and David, we also started in the Oxford Robotics Institute (ORI) with more people I am really grateful to met. I thank Mohamed Baioumy for always being keen to chat, discuss new ideas, and give me a chance to contribute since the beginning, despite the broken English I spoke when I started. I appreciated sharing an office and lunches with David Williams, who always had interesting questions to discuss and showed me how to make a *proper* cup of English tea. Paarth Shah and I had so many chats about research and optimisation, as well as shared so many pictures of cats. I enjoyed the brief chats with Oliver Shorthose and his supportive spirit. And of course thanks to Wenye Ouyang, for being my flatmate for almost 2 years and sharing the pain of the lockdown but the joy of the *Eat Out to Help Out*.

The ORI engineering team contributed enormously to my experience and work these years and I cannot thank them enough. I thank Benoit Casseau for all the work and help in different projects, so many field trials, and all the times he allowed me to bother him with a new question. Michal Staniaszek always helped me to solve technical issues and I am happy to see he is a PhD student now. Chris Prahacs always provided wisdom and the finest support, especially during the hardest times of the pandemic. Tom Dobra always solved *all* my infrastructure questions. I thank Matt Towlson and Wayne Tubby for always supporting any new hardware endeavour, and for being happy to help me with any new enquiry I brought despite how annoying I was. Jon Ody helped me fix ANYmal B during my first hardware experiments, which was a crucial step I needed to finish my first paper in 2020 in spite of all adversity. I lastly thank Tobit Flatscher not only for the C++ and Docker expertise but also for his remarkable willingness to share his knowledge (and hand-made pizzas!).

The ORI admin team made my life a lot easier these years. Rosemary Cameron introduced me to life here and solved all my initial questions. Laura O'Mahony provided all the support (under the hood) to make sure the processes worked. Oliver Barlett helped organise many of the first field trials I did in the ORI. Marysa Chapman helped me with the registration for the first conference I attended. Acacia Nockolds provided crucial support for my long stay in Zurich. Amber Allen helped me organise important field trials I needed for the RSS paper. Elsa Lam always helped me with *any* admin questions I had, from trips to getting new hardware. Timea Thorpe and Daniel Marques made sure that everything worked flawlessly at any event, field trial, or public demo I participated in.

I thank all my other ORI colleagues, for the spontaneous chats in the kitchen, for free food, or for sharing the struggle of working late for a deadline.

In 2022 I had the unique chance to visit the Robotic Systems Lab at ETH Zurich. I recognise it as one of the most important labs in robotics research nowadays and it was an honour to be there. I thank Marco Hutter for kindly allowing me to join them for 6 months and have a chance to contribute to his lab. To Maria Trodella for making this happen, helping me with all the paperwork, and solving all my questions. And to Cesar Cadena for the support and sincere advice, which I particularly appreciate from a fellow South American.

I met many researchers at RSL I learned a lot from. But first of all, I would like to thank Jonas Frey, for being the best research collaborator I could ever have. I really enjoyed working together, and learned so many things just by jointly developing the Wild Visual Navigation project—improving my Python skills being one of them. I wish we can keep collaborating in the future. I also thank the other members of Room H317, the *Navigation office*: Turcan Tuna, Fan Yang, Gabriel Waibel, Julian Nubert, and Viktor Klemm. I appreciate our chats, coffees, postcards, and also the weekend meetups at the Limmat. I also thank other people I had a chance to share and collaborate with: Simone Arreghini, Takahiro Miki, Joonho Lee, Yuntao Ma, Mayank Mittal, Philip Arm, Ruyi Zhou, Alessandro Fulciniti, Konrad Meyer, Markus Montenegro, Lorenz Wellhausen, Timon Homberger, and Marco Tranzatto.

I also must thank the robots at the ORI and RSL: Oxford's ANYmal B "Boxy" (RIP), ANYmal C "Coyote", and RSL's "Camel" and "Cerberus". While you made me suffer, seeing you run the algorithms I implemented and navigate autonomously is one of my biggest satisfactions. This thesis would not be possible without you.

These four years were not only about robots. I also met amazing non-robotics people. I thank my friends at Hertford College, Amelia Lee, Arjaan Buijs, and William Ivison for sharing our flat the first year before COVID hit, as well as

Guopeng Chen, Thibault Jouen, Haylee He, Bruce Li, and Vít Růžička for the welfare walks, board game nights and Oxmas dinners. I also thank Amelia for being my personal tour guide in Seoul when I attended RSS.

The Chilean community in Oxford is also larger than I expected, and I thank all the people I met over these years. Particularly, I thank Arantxa Gutiérrez, Thomas Püschel, and Gonzalo Mena for all the gatherings, laughs, advice, and support. I also appreciate the support of Fernanda Duarte and Raúl Santos, who were my neighbors in my first year and funnily enough we did not notice until weeks after I arrived.

I also thank my other Chilean friends in Europe and around the globe. I did not have the chance to visit all of them, but having the opportunity to share a few words or asking how we were doing, was highly valuable to me, particularly during COVID. I specially thank Daniela Barrientos, Patricio Correa, and Matias Suazo for the weekly meetings, online movie and game nights during the lockdown to stay sane, as well as the nice holidays we spent together once things got better.

I thank my friends in Chile for supporting me in spite of the distance, and for making the time to meet with me when I (finally) could travel there to visit in 2021.

Gracias a mi familia por el apoyo que me han dado siempre, aún sabiendo que no era fácil para ustedes ni para mi decidir estudiar en el extranjero por tanto tiempo. Gracias a mi mamá Mabel, mi papá Joel, mi hermano Tomás y mi abuelita Oli por el apoyo y cariño estos años, por las llamadas cada semana, y por siempre preguntarme cómo estoy. Espero el día en que puedan viajar para mi graduación, y mostrarles los lugares y experiencias que viví estos 4 años. Los amo.

Finally, I thank to my partner Pía Cortés for sharing this adventure from the very beginning. From our preparations for the English tests, searching for PhD programs, going through the processes, rejections, or no responses, to the time it got real and we had to move ahead. While we did not manage to make things happen as we wanted and ended up in completely different places, we made it work. Despite not having the chance to see us often, COVID, and other constraints, we made it work. And despite all the highs and lows of our own PhDs, we made it work. Let us keep making it work.

This PhD was funded by the Chilean National Agency for Research and Development (ANID) / DOCTORADO BECAS CHILE/2019 - 72200291. The research visit to ETH Zurich was partially funded by NCCR Robotics. Other trips were partially funded by Hertford College and the Department of Engineering Science.

Abstract

The recent advances in legged locomotion control have made legged robots walk up staircases, go deep into underground caves, and walk in the forest. Nevertheless, autonomously achieving this task is still a challenge. Navigating and acomplishing missions in the wild relies not only on robust low-level controllers but also higher-level representations and perceptual systems that are aware of the robot's capabilities.

This thesis addresses the navigation problem for legged robots. The contributions are four systems designed to exploit unique characteristics of these platforms, from the sensing setup to their advanced mobility skills over different terrain. The systems address localisation, scene understanding, and local planning, and advance the capabilities of legged robots in challenging environments.

The first contribution tackles localisation with multi-camera setups available on legged platforms. It proposes a strategy to actively switch between the cameras and stay localised while operating in a visual teach and repeat context—in spite of transient changes in the environment. The second contribution focuses on local planning, effectively adding a safety layer for robot navigation. The approach uses a local map built on-the-fly to generate efficient vector field representations that enable fast and reactive navigation. The third contribution demonstrates how to improve local planning in natural environments by learning robot-specific traversability from demonstrations. The approach leverages classical and learning-based methods to enable online, onboard traversability learning. These systems are demonstrated via different robot deployments on industrial facilities, underground mines, and parklands.

The thesis concludes by presenting a real-world application: an autonomous forest inventory system with legged robots. This last contribution presents a mission planning system for autonomous surveying as well as a data analysis pipeline to extract forestry attributes. The approach was experimentally validated in a field campaign in Finland, evidencing the potential that legged platforms offer for future applications in the wild.

Contents

List of Abbreviations

APF	Artificial Potential Field
CPM	Camera Performance Model
CVaR	Conditional Value-at-Risk
DARPA	Defense Advanced Research Projects Agency
DBH	Diameter at Breast Height
DoF	Degrees of Freedom
DSS	Decision Support System
FMM	Fast Marching Method
FoV	Field-of-View
GD	Gradient Descent
GDF	Geodesic Distance Field
GN	Gauss-Newton
GPU	Graphics Processing Unit
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
iSAM	Incremental Smoothing and Mapping
ISP	Image Signal Processor
LAGR	Learning Applied to Ground Vehicles
LiDAR	Light Detection and Ranging
LS	Least Squares
MAP	Maximum A Posteriori
MPC	Model Predictive Control
NLS	Non-linear Least Squares
PnP	Perspective-n-Points
RMP	Riemannian Motion Policy
ROS	Robot Operating System
SDF	Signed Distance Field
SfM	Structure-from-Motion
SGD	Stochastic Gradient Descent
SLAM	Simultaneous Localisation And Mapping
TLS	Terrestrial Laser Scanning

TO	Trajectory Optimisation
ToF	Time-of-Flight
UGV	Unmanned Ground Vehicle
URDF	Unified Robot Description Format
ViT	Visual Transformer
VT&R	Visual Teach and Repeat

1 Introduction

1.1 Motivation

Legged robots have the potential to perform tasks that are extremely challenging or risky for humans, such as exploration of unknown locations or rescue operations. The recent Defense Advanced Research Projects Agency (DARPA) SubT challenge (2018-2021) was a clear demonstration of the advantages of such platforms for exploration in urban and underground environments [3]. When the challenge started in 2018, 3 out of 11 teams were using legged robots for the competition; 3 years later almost every team relied on them as their core platforms [58]. Their versatility when negotiating different kinds of terrain compared to wheeled robots, as well as their increased payload capabilities compared to aerial ones offer a good balance for their deployment in the field.

A key aspect of this success has been the impressive mobility skills achieved by the locomotion controllers developed by the research community over the last few years [102, 197, 160], combined with the consolidation of commercial platforms such as the Boston Dynamics' Spot [34] and ANYbotics' ANYmal [10] quadrupeds, and humanoids such as the Agility Robotics' Digit [5]. This offers an unprecedented scenario for their mass adoption in applications such as last-mile delivery, industrial inspection, monitoring, or rescue.

The new locomotion controllers have unlocked new environments where legged robots can operate: not only warehouses with flat terrain but also industrial environments with complex staircases [34], urban and disaster scenarios [164], underground environments [197], as well as natural scenes such as forests and grasslands [160]. However, this comes with new challenges: while legged robots *can now walk* on challenging terrain, this does not necessarily guarantee that they can *execute autonomous missions* in such environments. For example, a legged robot might have the capacity to walk up a staircase given by its hardware design and locomotion control, but if a motion planner does not consider staircases as valid planning areas such advanced capabilities are futile.

As a consequence, moving from low-level locomotion problems to higher-level tasks such as navigation and mission planning is one of the main research challenges for legged robotics nowadays [284].

1.2 Objective

The main objective of this thesis is to develop new navigation systems for legged robots. We define navigation as the problem of safely moving between two locations A and B while avoiding obstacles and other risky areas [63, 180, 82]. While the definition is concise, developing such capabilities for robotic systems is a complex problem that requires a strong coordination of perception, planning, and control methods.

The second objective is then to develop the specific systems and methods that, when seamlessly integrated, enable legged robot navigation.

Last, we aim to achieve navigation in challenging environments: spaces that can be dynamic, difficult to access by humans, or simply risky. Specific examples include industrial facilities, mines, tunnels, or open natural scenes.

Figure 1.1: Autonomous deployments of legged robots using the contributions presented in this thesis. They include industrial environments, underground mines, public parks, and forests.

1.3 Contributions

The main contribution of this thesis is the investigation, implementation, and field deployment of new navigation systems to expand the use of legged robots in challenging environments, as shown in Fig. 1.1. The systems were designed to be simple but principled, exploiting fundamental ideas of optimisation methods as well as the constraints and advantages that legged robots present.

The specific contributions of this work include:

- An entropy-based approach for active camera selection that robustifies multi-camera localisation systems under transient scene changes.
- A local planning system that relies on local scene representations to achieve safe and reactive navigation.

- A framework for self-supervised online traversability learning that enables fast deployment in unknown environments.
- A mission planning system for autonomous forest inventory with legged robots.
- Hardware integration and deployment of these systems on legged platforms in industrial, underground, and natural environments.

1.4 Publications

The previous contributions have been materialised through three first-author peer-reviewed publications developed during this project. Additional contributions include six non-first-author peer-reviewed articles. Other open-source packages and tutorials are listed in Appendix A.

1.4.1 First-author Publications

1. **Mattamala, M.**, Ramezani, Camurri, M., Fallon, M. (2021). Learning Camera Performance Models for Active Multi-Camera Visual Teach and Repeat. *IEEE International Conference on Robotics and Automation (ICRA)*. [190]

2. **Mattamala, M.**, Chebrolu, N., Fallon, M. (2022). An Efficient Locally Reactive Controller for Safe Navigation in Visual Teach and Repeat Missions. *IEEE Robotics and Automation Letters (RA-L)*. [189]

3. Frey, J.*, **Mattamala, M.***, Chebrolu, N., Cadena, C., Fallon, M., Hutter, M. (2023). Fast Traversability Estimation for Wild Visual Navigation. *Robotics: Science and Systems*. [94] (*equal contribution.)

1.4.2 Co-authored Publications

1. Ramezani, M., Wang, Y., Camurri, M., Wisth, D., **Mattamala, M.**, Fallon, M. (2020). The Newer College Dataset: Handheld LiDAR, Inertial and Vision with Ground Truth. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. [228]

2. Wang, Y., Ramezani, M., **Mattamala, M.**, Fallon, M. (2021). Scalable and Elastic LiDAR Reconstruction in Complex Environments Through Spatial Analysis. *European Conference on Mobile Robots (ECMR)*. [283]

3. Ramezani, M., **Mattamala, M.**, Fallon, M. (2022). AEROS: Adaptive RObust least-Squares for Graph-Based SLAM. *Frontiers in Robotics and AI*. [226]

4. Wang, Y., Ramezani, M., **Mattamala, M.**, Digumarti, T., Fallon, M. (2022). Strategies for Large Scale Elastic and Semantic LiDAR Reconstruction. *Robotics and Autonomous Systems*. [282]

5. Tranzatto, M., Dharmadhikari, M., Bernreiter, *et al.*, including **Mattamala, M.** (2023). Team CERBERUS Wins the DARPA Subterranean Challenge: Technical Overview and Lessons Learned. *Field Robotics*. [274]

6. Erni, G., Frey, J., Miki, T., **Mattamala, M.**, Hutter, M (2023). MEM: Multi-Modal Elevation Mapping for Robotics and Learning. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

1.5 Thesis Outline

This thesis is presented in the *integrated format* of the University of Oxford. Chapters 4-6 consist of peer-reviewed publications, each one accompanied by additional discussion as well as a Statement of Authorship declaring the author's contribution to each work. Chapter 7 is an original chapter describing a real-world application.

The remainder of the thesis is structured as follows:

- **Chapter 2 – Background:** Presentation of the main definitions, theory, and methods used in the thesis.

- **Chapter 3 - Related Work:** Review of the relevant literature on robot navigation, with a particular focus on legged platforms.

- **Chapter 4 – Multi-camera Visual Localisation:** An approach to reliably localising robots by actively choosing the *most informative* camera during deployment.
- **Chapter 5 – Reactive Local Planning:** How to navigate in challenging, narrow environments using a reactive local planning approach.
- **Chapter 6 – Online Traversability Learning:** Overcoming the challenges of data collection and curation in traversability estimation via online self-supervised learning.
- **Chapter 7 – Autonomous Forest Inventory:** Conceptualisation and field deployment of autonomous legged robots for forestry applications.
- **Chapter 8 – Conclusion:** Discussion on the achievements and limitations of this work, lessons learned, and future directions for the field.

2
Background

This chapter introduces the main concepts and methods used in the remainder of this thesis. I begin by presenting basic notation and mathematical definitions in Sec. 2.1. Then I present the core ideas of least squares optimisation in Sec. 2.2, as it is the core machinery used for the systems developed in this work. Sec. 2.3 introduces the main sensors used in this work, providing further discussion on camera modelling. I close this chapter with Sec. 2.4 introducing algorithms for robot navigation, which will be further discussed in the Literature Review (Chapter 3).

2.1 Preliminaries

2.1.1 Notation

To refer to different mathematical objects, such as vectors, matrices or reference frames, we will use different typography styles. The conventions adopted in this thesis are summarised in Tab. 2.1.

2.1.2 Rotations and Poses

Rotation and rigid-body matrices are particularly important in robotics, as they are widely used to describe how objects are placed and moved in space. In this thesis we adopt the formalisation of *Lie groups* to represent them.

Table 2.1: Notation used for mathematical objects.

Quantity	Description	Example
Scalars	Upper/Lowercase italic	The traversability score s
Vectors	Lowercase bold	The 3D point $\mathbf{p}$
Matrices	Uppercase bold	The rotation matrix $\mathbf{R}$
Sets/Manifolds	Uppercase calligraphic	The map points $\mathcal{P}$
Reference Frames	Uppercase typewriter	The *map* frame $\texttt{M}$
	(legacy expression)	The *world* frame $\underrightarrow{\mathcal{F}}_W$

In brief, Lie groups are defined as both a *group* and a *differentiable manifold*. By being a group: (1) they define a binary operation, (2) the operation is associative, (3) there is an identity element, and (4) each element has an inverse. For example, for rotation and rigid-body matrices (1) is defined by matrix multiplication, which also satifies (2). For (3) they have the identity matrix $\mathbf{I}$, and (4) is the matrix inverse.

Their differentiable manifold side reflects the smooth geometric constraints that define a valid member of the group. For example, a 3D orientation has 3 Degrees of Freedom (DoF) and it can be represented by a 3D vector using Euler angles or an axis-angle representation [22]. However, not all vectors in $\mathbb{R}^3$ are valid orientations, and the constraints that define a valid Euler angle are not encoded in the representation. In contrast, a rotation matrix is an over-parametrised representation (9 entries) but with orthonormality constraints; this allows them to effectively represent the intrinsic 3 DoF of a valid orientation in a smooth manner—which defines the manifold.

This section provides basic definitions used across the manuscript; additional machinery is further explained in the specific chapters when required. For an in-depth, technical presentation refer to Solà, Deray, and Atchuthan [260] or Calinon [43]; for a practical overview, see my tutorial [188].

Rotation Matrices

Rotation matrices are formally part of the *Special Orthogonal Group* or $\mathrm{SO}(n)$. It is the set of all matrices $\mathbf{R}$ that satisfy orthonormality $\mathbf{R}^\top\mathbf{R} = \mathbf{I}$, as well as $\det \mathbf{R} = 1$.

In this thesis we are interested in the space of 2D rotations on the plane $\mathrm{SO}(2)$, as well as the space of 3D rotations $\mathrm{SO}(3)$.

Rigid-body Matrices

The group of rigid-body matrices is known as *Special Euclidean Group* or SE(n). They represent a relative 6 DoF transformation or *pose*, i.e. traslation and rotation. Their matrix structure is given by:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0_{1\times 3} & 1 \end{bmatrix} \tag{2.1}$$

In this thesis we are concerned specifically about the group of planar transformations SE(2), where $\mathbf{R} \in \mathrm{SO}(2)$ and $\mathbf{t} \in \mathbb{R}^2$, as well as the group of 3D transformations SE(3), where $\mathbf{R} \in \mathrm{SO}(3)$ and $\mathbf{t} \in \mathbb{R}^3$.

2.1.3 Frames

The mathematical quantities we will use are not abstract concepts but they are associated to the physical world. Concretely, they are represented with respect to *reference frames* that define a *point-of-view* used to describe them. This is particularly important not only for modelling [96] but also the definitions used in software tools widely used in robotics such as the Robot Operating System (ROS) [191]. The main reference frames we use in this work are shown in Tab. 2.2.

Table 2.2: Main reference frames used in this work.

Frame	Description
`B`	*base* or *body* frame specified at the center of the robot's body
`I`	*odometry* or *inertial* fixed frame used by odometry estimators
`M`	*map* fixed frame to specify the origin of the robot's map
`S`	*sensor* frame usually defined with respect to the base frame

We can then endow the aforementioned mathematical notation with special subindices to specify their relationship with the relevant frames. This is graphically illustrated in Fig. 2.1.

2.2 Optimisation

Optimisation is at the core of many robotics problems. In particular, squared cost minimisation problems, known as Least Squares (LS), are one of the most commonly

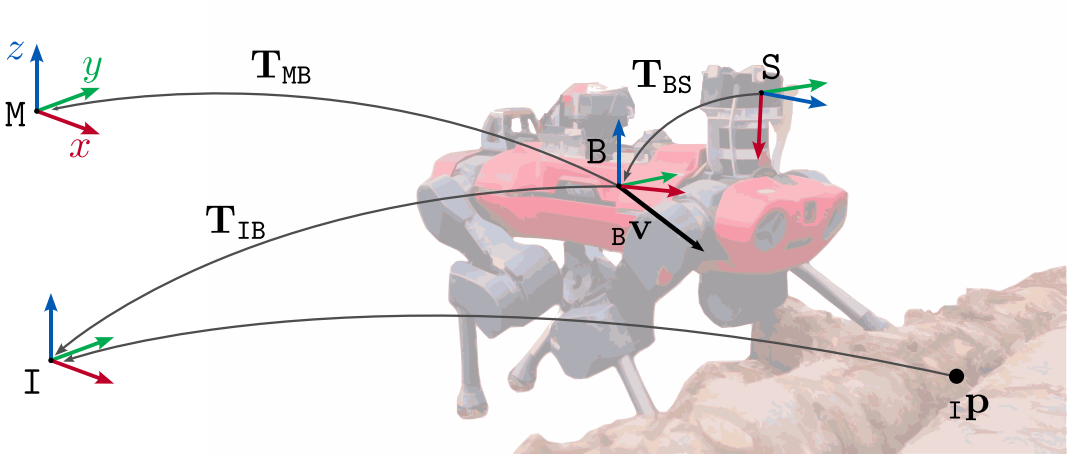

Figure 2.1: Main reference frames and frame-endowed notation used in this thesis. We adopt the axes colour convention with the x-axis in **red**, y-axis in **green**, and z-axis in **blue**. The inertial frame $\mathtt{I}$ and the map frame $\mathtt{M}$ are fixed, the base frame $\mathtt{B}$ moves with the robot, while the sensor frame $\mathtt{S}$ is fixed relative to $\mathtt{B}$ through the rigid-body transformation $\mathbf{T}_{\mathtt{MB}}$. The linear velocity of the base is given by the vector ${}_{\mathtt{B}}\mathbf{v}$. The fixed 3D point in the frame $\mathtt{I}$ is denoted by ${}_{\mathtt{I}}\mathbf{p}$.

used formulations to solve state estimation, motion planning, and learning [71, 28]. We review the basic concepts and methods that will be used in the following chapters of the thesis.

2.2.1 Linear Least Squares

We begin with the simplest LS formulation. Let us consider a linear model:

$$f(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b} \tag{2.2}$$

with $\mathbf{x} \in \mathbb{R}^n$ an unknown variable, and $\mathbf{b} \in \mathbb{R}^m$, and $\mathbf{A} \in \mathbb{R}^{m \times n}$ hyper-parameters of the model.

We would like to estimate the value of the variable $\mathbf{x}$, given empirical samples of the linear model $\mathbf{y}_i \in \mathbb{R}^m$, with $i = 1, \ldots, N$. For some value of $\mathbf{x}$, we can characterise how well the linear model approximates each sample $\mathbf{y}_i$ by means of the *residual function*:

$$\mathbf{r}_i(\mathbf{x}) = \mathbf{y}_i - (\mathbf{A}_i\mathbf{x} + \mathbf{b}_i) \tag{2.3}$$

Then, the LS estimate of $\mathbf{x}$, denoted $\mathbf{x}^\star$, is the estimate that minimises the *cost* or *loss* function $\mathcal{L}(\mathbf{x})$ defined as the sum of the squared residuals for all

the data samples:

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \mathcal{L}(\mathbf{x}) \tag{2.4}$$

$$= \arg\min_{\mathbf{x}} \sum_i^N \| \mathbf{r}_i(\mathbf{x}) \|^2 \tag{2.5}$$

$$= \arg\min_{\mathbf{x}} \sum_i^N (\mathbf{y}_i - (\mathbf{A}_i\mathbf{x} + \mathbf{b}_i))^\top (\mathbf{y}_i - (\mathbf{A}_i\mathbf{x} + \mathbf{b}_i)) \tag{2.6}$$

Sometimes not all the data samples have the same importance, and we would like to scale the contribution of the set of samples or specific dimensions. We introduce an additional matrix $\mathbf{\Sigma}_i$, usually diagonal, as a hyper-parameter to control this:

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \sum_i^N \| \mathbf{r}_i(\mathbf{x}) \|^2_{\mathbf{\Sigma}_i} \tag{2.7}$$

$$= \arg\min_{\mathbf{x}} \sum_i^N \mathbf{r}_i(\mathbf{x})^\top \mathbf{\Sigma}_i^{-1} \mathbf{r}_i(\mathbf{x}) \tag{2.8}$$

$$= \arg\min_{\mathbf{x}} \sum_i^N (\mathbf{y}_i - (\mathbf{A}_i\mathbf{x} + \mathbf{b}_i))^\top \mathbf{\Sigma}_i^{-1} (\mathbf{y}_i - (\mathbf{A}_i\mathbf{x} + \mathbf{b}_i)) \tag{2.9}$$

which corresponds to the most general version of linear LS we will consider in the rest of the manuscript.

Solving the Problem

To solve Eq. (2.7), it is necessary to linearise the quadratic cost. As the problem is convex, the linearisation produces a set of linear equations for each residual known as *normal equations*:

$$\mathbf{A}_1^\top \mathbf{\Sigma}_1^{-1} \mathbf{A}_1 \mathbf{x}^\star = \mathbf{A}_1^\top \mathbf{\Sigma}_1^{-1} \mathbf{b}_1 \tag{2.10}$$

$$\cdots \tag{2.11}$$

$$\mathbf{A}_N^\top \mathbf{\Sigma}_N^{-1} \mathbf{A}_N \mathbf{x}^\star = \mathbf{A}_N^\top \mathbf{\Sigma}_N^{-1} \mathbf{b}_N \tag{2.12}$$

which can be rearranged and stacked together as a single linear system:

$$\left(\mathbf{A}^\top \mathbf{\Sigma}^{-1} \mathbf{A}\right) \mathbf{x}^\star = \mathbf{A}^\top \mathbf{\Sigma}^{-1} \mathbf{b} \tag{2.13}$$

The matrix on the left-hand side is known as the *Hessian*:

$$\mathbf{A}^\top \mathbf{\Sigma}^{-1} \mathbf{A} \tag{2.14}$$

By inverting the Hessian, we can recover the estimate of $\mathbf{x}^\star$ in closed-form:

$$\mathbf{x}^\star = \left(\mathbf{A}^\top \boldsymbol{\Sigma}^{-1} \mathbf{A}\right)^{-1} \mathbf{A}^\top \boldsymbol{\Sigma}^{-1} \mathbf{b} \tag{2.15}$$

While this process is straightforward, it can be expensive to compute when the dimensionality of $\mathbf{x}$ is large. Standard procedures to solve LS include factorising the Hessian in the normal equations (Eq. (2.13)) using Cholesky or QR factorisarion [106].

Exploiting the structure of the Hessian matrix with problem-specific knowledge has also been heavily exploited in robotics to achieve real-time performance. Some examples are Square Root SAM [72] and Incremental Smoothing and Mapping (iSAM) [137] for incremental state estimation, or KOMO for receding-horizon motion planning [272].

Probabilistic Interpretation

The formulation of LS used in Eq. (2.7) has a probabilistic interpretation: It is the Maximum A Posteriori (MAP) formulation of an estimation problem with random variables that follow a Gaussian distribution. In particular, if we consider the residual to follow a zero-mean Gaussian distribution with covariance $\boldsymbol{\Sigma}_i$:

$$\mathbf{r}_i(\mathbf{x}) \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\Sigma}_i\right) \tag{2.16}$$

Then the MAP is given by:

$$\mathbf{x}^\star = \arg\max_{\mathbf{x}} \prod_i^N \frac{1}{\sqrt{(2\pi)^m \det \boldsymbol{\Sigma}_i}} \exp\left(-\frac{1}{2} \mathbf{r}_i(\mathbf{x})^\top \boldsymbol{\Sigma}_i^{-1} \mathbf{r}_i(\mathbf{x})\right) \tag{2.17}$$

To show that it is equivalent to LS, we simply compute the negative logarithm of Eq. (2.17):

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \sum_i^N \frac{1}{2} \mathbf{r}_i(\mathbf{x})^\top \boldsymbol{\Sigma}_i^{-1} \mathbf{r}_i(\mathbf{x}) + \ln\left(\sqrt{(2\pi)^m \det \boldsymbol{\Sigma}_i}\right) \tag{2.18}$$

As we are only interested in the terms that depend on $\mathbf{x}$, we can disregard the term on the right, as well as the scaling $\frac{1}{2}$. This optimisation is then equivalent to Eq. (2.7).

Additionally, we can rewrite Eq. (2.17) as:

$$\mathbf{x}^\star = \arg\max_{\mathbf{x}} \prod_i^N \phi(\mathbf{x}) \tag{2.19}$$

Which expresses the problem as a product of factors. This has an equivalent representation in the graphical models literature, known as a *factor graph* [71]. Hence, for the particular case of Gaussian factor graphs, where factors are defined by Eq. (2.16), solving an inference problem on the factor graph is equivalent to LS.

Interpretations of the Hessian

The Hessian (Eq. (2.14)) has important interpretations. Geometrically, the Hessian is the second derivative of the LS cost function. Hence, it represents the curvature around the minimum. If we consider the cost function as a manifold, then the Hessian also defines the *Riemannian metric* of the manifold [272]—this idea will be important for the local planner presented in Chapter 5. Probabilistically, it is the *Fisher Information Matrix* $\mathbf{I}_{\text{Fisher}}$. Its inverse corresponds to the covariance matrix of the MAP estimate $\mathbf{I}_{\text{Fisher}}^{-1} = \boldsymbol{\Sigma}^\star$. When the posterior of the MAP is fitted with $\mathcal{N}(\mathbf{x}^\star, \boldsymbol{\Sigma}^\star)$, the approximation is known as *Laplace approximation* [28].

The probabilistic interpretation allows us to study the LS solution further. For example, under the Gaussian approximation we can determine the *entropy* H of the estimate, which represents its quality. It is defined as:

$$H(\mathbf{x}^\star) = \frac{1}{2} n \left(1 + \ln(2\pi)\right) + \frac{1}{2} \ln\left(\det \boldsymbol{\Sigma}^\star\right) \tag{2.20}$$

We can use the entropy to compare the quality different LS estimates. However, we observe that (1) it only depends on $\boldsymbol{\Sigma}^\star$, so we can ignore the constant terms, (2) we require to invert $\mathbf{I}_{\text{Fisher}}$ to obtain $\boldsymbol{\Sigma}^\star$. Hence, we can use the *negative entropy* as proposed by Kuo et al. [161]:

$$E(\mathbf{x}^\star) = \ln\left(\det \mathbf{I}_{\text{Fisher}}\right) = \ln\left(\det\left(\mathbf{A}^\top \boldsymbol{\Sigma}^{-1} \mathbf{A}\right)\right) \tag{2.21}$$

This is a positive scalar value that will be used in Chapter 4 to compare different estimates obtained from LS.

2.2.2 Non-linear Least Squares

The previous formulation of LS is useful to understand the basic principles but it is not the one usually found in real robotics problems. A more general approach is minimise a cost $\mathcal{L}(\mathbf{x})$ with squared non-linear residuals, known as Non-linear Least Squares (NLS):

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \sum_i^N \| \mathbf{r}_i(\mathbf{x}) \|^2_{\mathbf{\Sigma}_i} \tag{2.22}$$

$$= \arg\min_{\mathbf{x}} \sum_i^N \| f(\mathbf{x}, \mathbf{y}_i) \|^2_{\mathbf{\Sigma}_i} \tag{2.23}$$

with f a non-linear function of the variables $\mathbf{x}$ and the data $\mathbf{y}_i$.

Solving the Problem

To solve NLS, the most standard procedure is linearising the residuals in Eq. (2.22) with respect to an operation point $\bar{\mathbf{x}}$:

$$\mathbf{r}_i(\mathbf{x}) \approx \mathbf{r}_i(\bar{\mathbf{x}} + \delta\mathbf{x}) \approx \mathbf{r}_i(\bar{\mathbf{x}}) + \mathbf{J}_{\mathbf{r}_i}(\bar{\mathbf{x}})\, \delta\mathbf{x} \tag{2.24}$$

where $\mathbf{J}_{\mathbf{r}_i}$ is the *Jacobian* of the residual function with respect to $\mathbf{x}$. Then, Eq. (2.22) becomes a LS problem on the increment $\delta\mathbf{x}$ that we can solve in closed-form:

$$\delta\mathbf{x}^\star = \arg\min_{\delta\mathbf{x}} \sum_i^N \| \mathbf{r}_i(\bar{\mathbf{x}}) + \mathbf{J}_{\mathbf{r}_i}(\bar{\mathbf{x}})\, \delta\mathbf{x} \|^2_{\mathbf{\Sigma}_i} \tag{2.25}$$

$$= \left(\mathbf{J}^\top \mathbf{\Sigma}^{-1} \mathbf{J}\right)^{-1} \mathbf{J}^\top \mathbf{\Sigma}^{-1} \bar{\mathbf{x}} \tag{2.26}$$

with $\mathbf{J}$ a matrix built by stacking the Jacobians of all the residuals $\mathbf{J}_{\mathbf{r}_i}$. The increment $\delta\mathbf{x}^\star$ is used to update the current estimate of $\mathbf{x}$ by using the rule:

$$\mathbf{x} \leftarrow \bar{\mathbf{x}} + \delta\mathbf{x}^\star \tag{2.27}$$

This is the basis of the Gauss-Newton (GN) algorithm. It solves the NLS problem by starting from an initial estimate, linearising the cost around it (Eq. (2.24)), solving the normal equations (Eq. (2.25)), and iteratively updating the estimate (Eq. (2.27)) until convergence.

Alternative Optimisation Approaches

Although GN provides an straightforward framework to solve NLS, it has some drawbacks, as computing and inverting the Hessian can be expensive. This is particularly the case when the nonlinear function f is parametrised by a neural network. An alternative approach is defining a simpler rule based on the gradient of the cost function $\nabla\mathcal{L}(\mathbf{x})$, known as *gradient descent*:

$$\mathbf{x} \leftarrow \bar{\mathbf{x}} - \gamma\nabla\mathcal{L}(\bar{\mathbf{x}}) \tag{2.28}$$

Where γ is known as *step size* or *learning rate*. In practice, due to the non-convexity of neural networks, Gradient Descent (GD) will not lead to the steepest descent direction. Instead, we can sample random data batches of size B to evaluate the gradient for each sample i, $\nabla\mathcal{L}_i(\mathbf{x})$. Then, we can use the average gradient for the update rule, which is known as Stochastic Gradient Descent (SGD):

$$\mathbf{x} \leftarrow \bar{\mathbf{x}} - \frac{\gamma}{B}\sum_{i}^{B}\nabla\mathcal{L}_i(\bar{\mathbf{x}}) \tag{2.29}$$

Nowadays more sophisticated optimisation methods exists, such as Adam [149], which are based on SGD. Adam is later used to solve an online learning problem in Chapter 6.

On the Interpretations of NLS

As a last remark, the same geometric and probabilistic interpretations of LS hold for NLS but only for the current linearisation point. Further, as the optimisation is non-convex, the estimates of the mean and covariance from the Laplace approximation do not hold globally, and they can under or over estimate the covariance of the estimate [22, p. 109].

2.3 Platforms and Sensors

In this section we review the main sensors and platforms used in this work. We first discuss *exteroceptive* sensors, which measure the environment external to the robot. We focus on visual sensors, i.e. cameras, as they are the main sensor used

Figure 2.2: Main camera units used in this work. **Left:** Intel *Realsense D435i* stereo and depth visual-inertial unit. Source: Intel. **Right:** Sevensense *Core Research* multi-camera visual-inertial sensor (formerly *Alphasense Core*). Source: Sevensense.

in this thesis. Light Detection and Ranging (LiDAR) sensing is also discussed as it is used in later chapters. *Proprioceptive* sensing, which provides information about the robot's internal state, is just briefly presented.

We then provide a general overview about legged robots, their distinctive features, and how they differ from other platforms such as wheeled and aerial. This will prepare us for the last section of this chapter, that discusses robot navigation systems.

2.3.1 Cameras

Cameras are ubiquitous sensors: they are low-cost, provide rich appearance information about the external world, and operate at reasonable frequency (10–30 Hz) given the typical speed of the majority of robotics platforms. When combined with other cameras (stereo systems) or active lighting (e.g. structured light), they can also measure 3D information. The main cameras used in this thesis are the Intel Realsense camera, and the Sevensense Core Research multi-camera unit, shown in Fig. 2.2.

Their basic operating principle is based on a regular array of light-sensitive *pixels*, known as *digital image sensor*. Each pixel outputs a signal that is proportional to the number of photons received in a fixed *exposure time*, which also determines the maximum operating frequency (10–30 Hz). Colour cameras additionally have colour filters for each pixel (red, green, blue); their arrangement is known as *Bayer pattern*.

Producing a visually appealing image from the pixels' signals and the Bayer pattern requires low-level processing provided by a Image Signal Processor (ISP). The ISP is usually implemented on dedicated hardware, and it performs different operations that include demosaicing, gamma correction, white balance, and tone

mapping [69]. While this is usually provided by manufacturers for commercial cameras, this is not necessarily the case for experimental kits such as the Alphasense.

As part this thesis, a ROS-compatible ISP for the Alphasense was developed, which performs debayering, white balancing, and lens undistortion, among other steps. The package is open-source and available as `raw_image_pipeline`[1].

Once monochromatic (greyscale) or colour (RGB) images are produced, we can perform further processing [63]. As we are interested in obtaining 3D information from the world, we need to model the geometry of the image projection process. This is explained in the following sections.

Camera Projection Geometry

We begin by modelling the geometry of the projection process, as it has a primary role in Chapter 4 and Chapter 6.

We consider that the camera has a pose $\mathbf{T}_{\mathtt{MC}} \in \mathrm{SE}(3)$ in a fixed frame $\mathtt{M}$. In computer vision textbooks [266, 116, 184], this matrix is known as the *camera extrinsics*. In robotics we usually estimate the pose of the robot body's instead of the sensor's, so we define the extrinsics as the product $\mathbf{T}_{\mathtt{MC}} = \mathbf{T}_{\mathtt{MB}}\mathbf{T}_{\mathtt{BC}}$, where $\mathbf{T}_{\mathtt{MB}}$ is the *robot's body pose*, and $\mathbf{T}_{\mathtt{BC}}$ is the *extrinsic calibration matrix of the camera with respect to the robot's body*, which is obtained from prior calibration.

Then, a 3D point in homogeneous coordinates ${}_{\mathtt{M}}\mathbf{p} = [X, Y, Z, 1] \in \mathbb{R}^4$ in the map frame $\mathtt{M}$ is related to the point ${}_{\mathtt{S}}\mathbf{p}$ in sensor frame $\mathtt{S}$ by the projection model $\pi\left({}_{\mathtt{S}}\mathbf{p}, {}_{\mathtt{M}}\mathbf{p}\right)$, which is illustrated in Fig. 2.3. The specific steps to achieve this are explained as follows.

Step 1: Represent the Point in the Camera Frame We first change the reference frame of ${}_{\mathtt{M}}\mathbf{p}$ from $\mathtt{M}$ to the *camera* frame $\mathtt{C}$ (also called *optical frame*):

$$ {}_{\mathtt{C}}\mathbf{p} = \mathbf{T}_{\mathtt{MC}}^{-1}\, {}_{\mathtt{M}}\mathbf{p} \tag{2.30} $$

The resulting point ${}_{\mathtt{C}}\mathbf{p}$ is also in homogeneous coordinates but now with respect to the frame $\mathtt{C}$.

[1] https://github.com/leggedrobotics/raw_image_pipeline

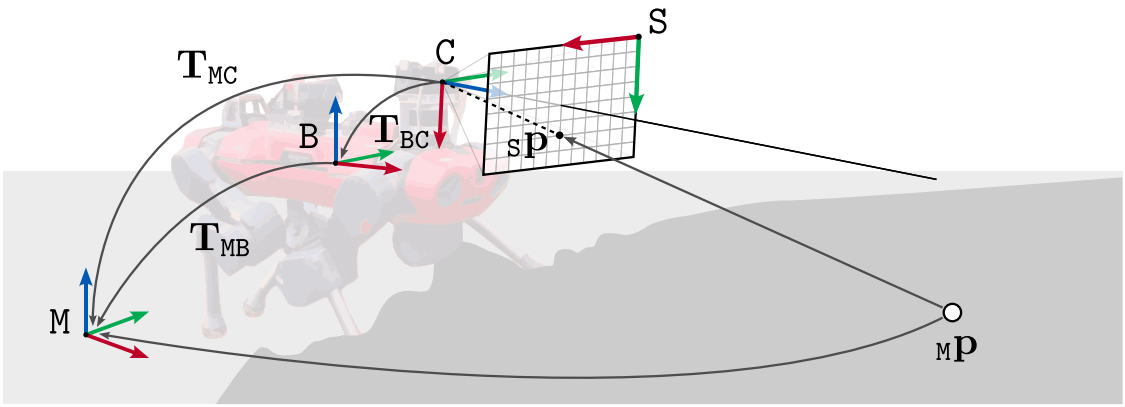

Figure 2.3: The pinhole camera model $\pi\left({}_S\mathbf{p}, {}_M\mathbf{p}\right)$ projects the 3D point ${}_M\mathbf{p}$ in the map frame, to the image cordinates ${}_S\mathbf{p}$, given the extrinsic matrix $\mathbf{T}_{MC}$.

Step 2: Pinhole Projection We then apply the *pinhole projection model*:

$$ {}_S\mathbf{p}' = \mathbf{K}_{SC}\, {}_C\mathbf{p} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} f_x X + c_x Z \\ f_y Y + c_y Z \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \tag{2.31} $$

Where $\mathbf{K}_{SC} \in \mathbb{R}^{4\times4}$ is the *intrinsic calibration* matrix. f_x and f_y are the camera focal lengths, while c_x and c_y are the coordinates of the center of the image. They are obtained used standard camera calibration packages, such as Kalibr [98, 232].

Step 3: Normalise the Coordinates The point on the image plane I, denoted ${}_S\mathbf{p}$, is obtained by dividing ${}_S\mathbf{p}'$ by the third component. This produces the normalised image coordinates (Szeliski [266]):

$$ {}_S\mathbf{p} = \frac{{}_S\mathbf{p}'}{{}_S\mathbf{p}'_{[3]}} = \frac{1}{z}\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x/z \\ y/z \\ 1 \\ 1/z \end{bmatrix} = \begin{bmatrix} u \\ v \\ 1 \\ d \end{bmatrix} \tag{2.32} $$

Here (u, v) denote the pixel coordinates of the projected point on the image plane, and d its *disparity* (inverse depth).

Moving Back from Images to 3D

An advantage of this formulation, is that in order to *back-project* an image point back into the 3D world, we can easily invert the operations:

Step 1: De-normalisation Using the disparity d:

$$ {}_{\mathtt{S}}\mathbf{p}' = \frac{{}_{\mathtt{S}}\mathbf{p}}{{}_{\mathtt{S}}\mathbf{p}_{[4]}} = \frac{1}{d}\begin{bmatrix} u \\ v \\ 1 \\ d \end{bmatrix} = \begin{bmatrix} u/d \\ v/d \\ 1/d \\ 1 \end{bmatrix} \tag{2.33} $$

Step 2: Back-project Using the inverse of the camera instrinsics matrix:

$$ {}_{\mathtt{C}}\mathbf{p} = \mathbf{K}_{\mathtt{SC}}^{-1}\,{}_{\mathtt{I}}\mathbf{p}' = \begin{bmatrix} 1/f_x & 0 & -c_x/f_x & 0 \\ 0 & 1/f_y & -c_y/f_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} u/d \\ v/d \\ 1/d \\ 1 \end{bmatrix} = \begin{bmatrix} f_x/d\,(u - c_x) \\ f_y/d\,(v - c_y) \\ 1/d \\ 1 \end{bmatrix} \tag{2.34} $$

Please note that this is not possible for all kinds of cameras. Monocular cameras only provide the image coordinates (u, v) and the disparity d is not observable. In this case, the resulting ${}_{\mathtt{C}}\mathbf{p}$ is valid *up to a scale factor* $1/d$. This is also interpreted as monocular cameras being *bearing sensors*: they can only measure a ray (direction) to the true 3D point.

For stereo cameras, we can estimate the disparity by using the *baseline* b of the camera, which is typically provided by the manufacturer or custom calibration. Given the measurements (u_l, v_r) and (u_r, v_r) from the left and right cameras of a stereo pair, the disparity is determined by:

$$ d = \frac{u_l - u_r}{b} \tag{2.35} $$

For depth cameras, the distance to the 3D point, denoted z, is measured directly via structured light or Time-of-Flight (ToF) imaging:

$$ d = \frac{1}{z} \tag{2.36} $$

Step 3: Back to World Frame Finally we obtain the 3D point in the world frame $\mathtt{M}$ using the extrinsics matrix:

$$ {}_{\mathtt{M}}\mathbf{p} = \mathbf{T}_{\mathtt{MC}}\,{}_{\mathtt{C}}\mathbf{p} \tag{2.37} $$

The aforementioned steps are usually summarised by the *projection function* $\pi(\cdot)$ and *back-projection function* $\pi^{-1}(\cdot)$.

Applications

The pinhole model expresses the fundamental principles that relate images with the 3D structure of the environment. This enables different applications for robotics that are used in the subsequent chapters of this thesis, which are briefly discussed here.

Visual Localisation In Chapter 4 we will apply the pinhole model for visual localisation using the Perspective-n-Points (PnP) [89] algorithm and factor graph-based pose estimation. These methods minimise the *reprojection error* between a set of image measurements ${}_{\mathtt{S}}\mathbf{z}_i$ and the 2D projection of 3D map points ${}_{\mathtt{M}}\mathbf{p}_i$, as illustrated in Fig. 2.4 and further discussed in Sec. 2.4.1.

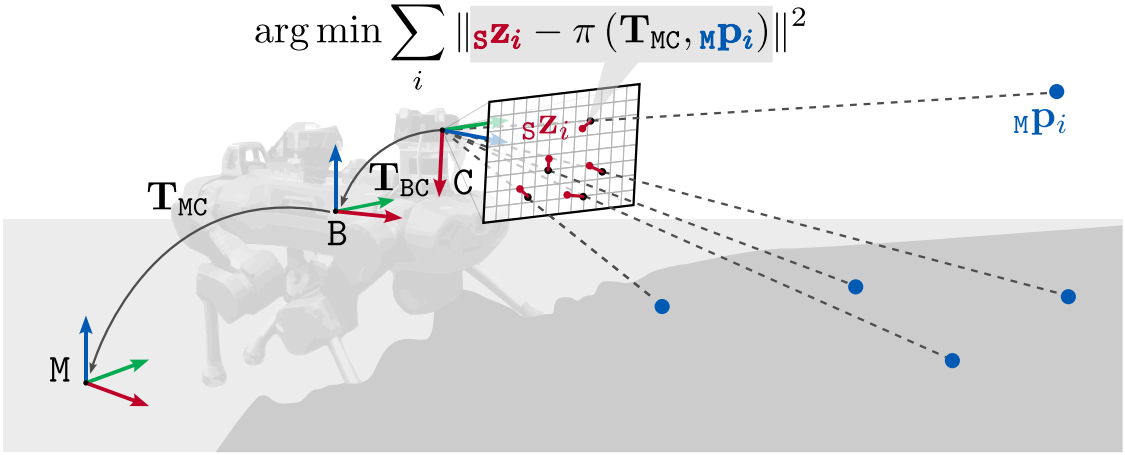

Figure 2.4: *Visual localisation* methods minimise the reprojection error the idealised projection of 3D points ${}_{\mathtt{M}}\mathbf{p}_i$ and image measurements ${}_{\mathtt{S}}\mathbf{z}_i$. We use these principles in Chapter 4.

Elevation Mapping The principles of camera geometry can be applied to produce dense point clouds using stereo cameras or ToF imaging, and then generate a local 2.5D elevation map, as shown in Fig. 2.5. This local representation is used for efficient local planning in Chapter 5.

Self-supervision In Chapter 6, we will use the projection model to project the path traversed by the robot by integrating the footprint over time (see Fig. 2.6). We will use this information as a supervision signal to learn a model to predict which areas are accessible by the robot.

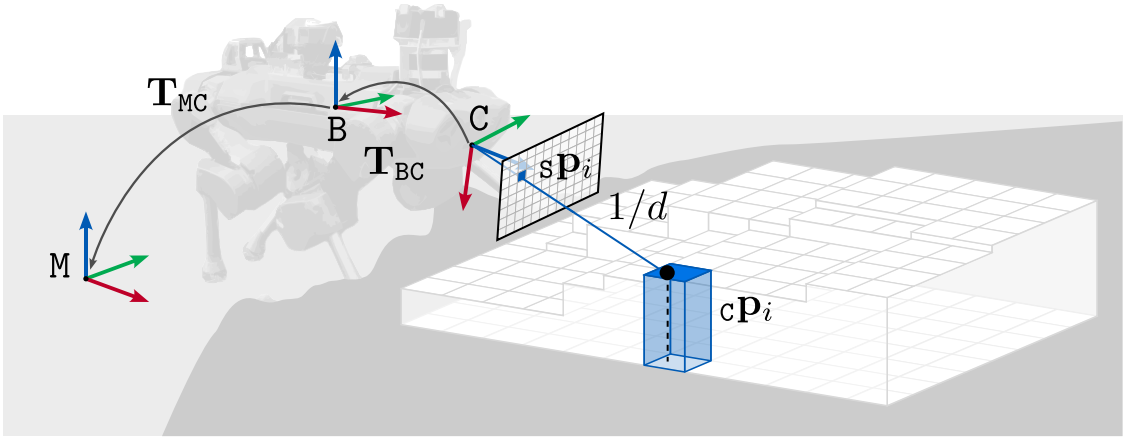

Figure 2.5: *Elevation mapping* relies on the back-projected image points using stereo or depth information to ray-cast them onto a planar grid that encodes the point's height. This grid representation is used for local planning in Chapter 5.

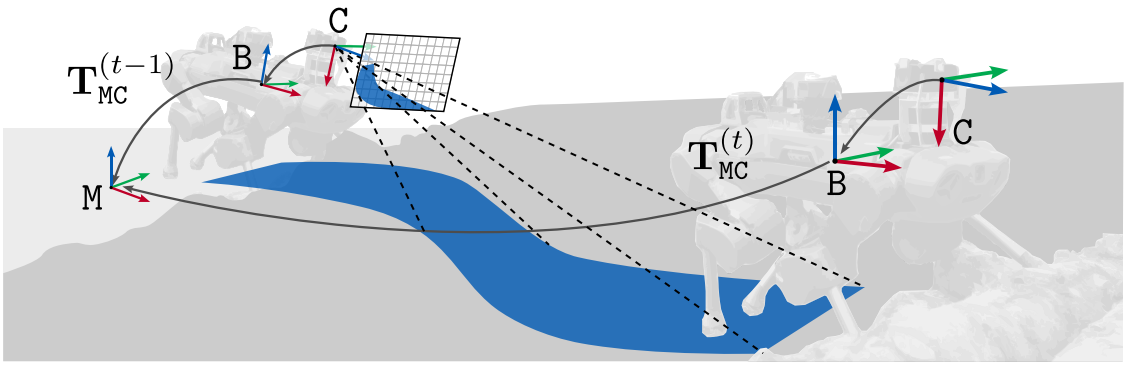

Figure 2.6: By integrating the robot's traversed path over time and projecting it onto past images, we can generate a *supervision signal* used to train models of terrain traversability, as will be shown in Chapter 6

.

Lens Distortion

In the previous applications we did not consider the non-linear effects produced by the camera lenses, which distort how the points are projected on the image plane. Lens distortion models compensate for these effects, so the projections and other operations stay invariant.

For cameras such as the Realsense (Fig. 2.2, left), these effects can be compensated using a *radial-tangential* model [39]. Wide-angle cameras such as the Alphasense unit (Fig. 2.2, right) are modelled by *equidistant* distortion [300] instead.

Figure 2.7: Fisheye lens distortion in the Sevensense Core Research sensor. **Left:** Distorted output before fisheye correction, henceforth displaying curved trees. **Right:** Undistorted image, correctly displaying the trees standing straight. The empty pixels (black) represent areas that are not observed in the original input; these are introduced to match the specified intrinsics after undistortion.

Both models are available in camera calibration packages such as Kalibr [98, 232], which enable us to use the same projection principles regardless of the specific camera being used.

2.3.2 LiDAR

LiDAR is another common exteroceptive sensor in robotics, which provides 3D information about the environment. In contrast to (monocular) cameras, they can directly measure distance, achieving centimetre-accurate range measurement up to 100 m.

Distance measurements are obtained by projecting a laser beam into the environment and measuring the time it takes to return. As this process occurs along a single ray, LiDARs use fixed laser arrays and rotating mirrors to generate 2D or 3D scans. As a result, LiDARs operate at slightly lower frequencies than cameras, up to 20 Hz.

In the remainder of the thesis, LiDARs are used only as a complementary sensor for evaluation purposes (Chapter 4), or as another complementary source of 3D information (Chapter 5 and Chapter 6). In Chapter 7 they will acquire a more important role as they provide the main data input for autonomous forest inventory. Some of the devices discussed in this thesis are shown in Fig. 2.8.

Figure 2.8: LiDARs relevant to this thesis. **Left:** Velodyne *VLP-16* LiDAR. **Center:** RoboSense *BPearl*. **Right:** Leica *RTC360* Terrestrial Laser Scanning (TLS).

2.3.3 Proprioceptive Sensing

Proprioceptive sensors provide information about the robot's internal state. The main examples we consider are Inertial Measurement Units (IMUs) and joint sensors, as they give important context for the navigation algorithms that will be discussed in Sec. 2.4, although their are not the main focus of this thesis.

IMUs provide high-frequency (>100 Hz) angular velocity and linear acceleration measurements. This enables robots to have access to their attitude with respect to gravity, as well as an estimate of the relative translation by time-integration of the measurements [297]. However, they can suffer from thermo-mechanical effects that degrade this estimate, which depends on the IMU grade, and additional actions need to be taken to compensate for this effects, such as sensor fusion [64].

Joint sensors provide the position, velocity, or torque of a specific joint. As they are usually part of the actuators' hardware for joint-level control, they also operate at higher frequencies (>1 kHz). In combination with the robot description (e.g. Unified Robot Description Format (URDF)), they can provide real-time information about the kinematic configuration of the robot or interacting forces.

2.3.4 Legged Robots

Legged robots are a type of mobile robot characterised by the use of limbs to locomote. Among legged platforms, the main distinctive feature is the number of legs used for locomotion. Engineers have developed single-leg hoppers (1 leg), bipeds and humanoids (2 legs), quadrupeds (4 legs), and hexapods (6 legs) [139].

In this thesis we focus on quadrupedal platforms. Quadrupeds offer a good balance between robustness, cost, and control complexity compared to other legged robots, as they have the minimal number of legs to statically locomote [124].

The main advantage legged robots offer over other ground platforms, such as wheeled robots, is their advanced mobility skills to traverse different types of terrain, including human-made and natural ones. Compared to aerial platforms such as small-size drones, they can carry heavier payloads in longer missions.

In Tab. 2.3 we compare the main legged platforms commercially available (as of the writing of this manuscript), to other popular platforms such as wheeled robots and drones. This will be relevant in Chapter 7 to contextualise the use of legged robot in forestry. It also summarises the general characteristics of the ANYbotics ANYmal [125, 10] robot, which is the main platform used for the development of this thesis.

2.4 Algorithms for Robot Navigation

In this last section we discuss algorithms for robot navigation. We consider the robot navigation pipeline shown in Fig. 2.9. It involves the interaction of these modules that address capabilities and representations a robot needs to move in the environment. We will discuss them in the following sections.

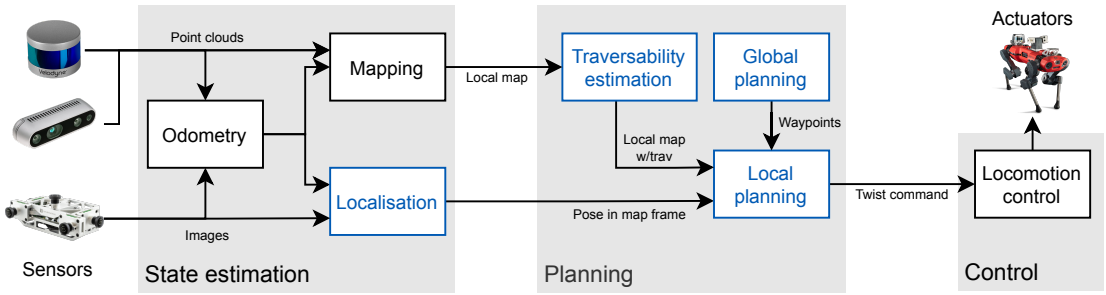

Figure 2.9: Modular robot navigation pipeline. We have highlighted in blue the main modules we are concerned with in this thesis. Please refer to Sec. 2.4 for more details.

Table 2.3: Specifications of different commercial robotic platforms. ANYmal C is the main robot used in this thesis.

Name	Robot	Type	Size (LxWxH)	Weight	Payload	Max Speed	Run time	IP rat.	Web
ANYbotics ANYmal C		Legged	1054 × 520 × 830 mm*	50 kg	10 kg	1.0 m/s	1.5 h	IP67	–
ANYbotics ANYmal D		Legged	930 × 530 × 890 mm*	50 kg	10 kg	1.3 m/s	1.5 h	IP67	link
BD Spot		Legged	1100 × 500 × 610 mm*	32.7 kg	14 kg	1.6 m/s	1.5 h	IP54	link
Unitree B1		Legged	1126 × 467 × 636 mm*	50 kg	20 kg	1.2 m/s	2 h	IP68	link
Clearpath Husky		Wheeled	990 × 760 × 390 mm	50 kg	20 kg	1.0 m/s	3 h	IP44	link
Clearpath Warthog		Wheeled	1520 × 1380 × 830 mm	590 kg	272 kg	5.0 m/s	2.5 h	IP65	link
Leica BLK2Fly		Aerial	530 × 600 × 190 mm	2.6 kg	–	3.0 m/s†	13 min	IP54	link
DJI Matrice 600		Aerial	1668 × 1518 × 759 mm	9.1 kg	5 kg	18 m/s‡	35 min	–	link
Avartek Boxer		Aerial	990 × 745 × 570 mm	16.5 kg	5 kg	–	1.5 h	–	link

* Height when standing in nominal configuration.
‡ Maximum horizontal speed in free space with no wind.

2.4.1 State Estimation

State estimation creates representations from sensor data, which are used by the downstream tasks. They include internal state representations (odometry, localisation) as well as scene representations (global and local maps). This is illustrated in Fig. 2.10.

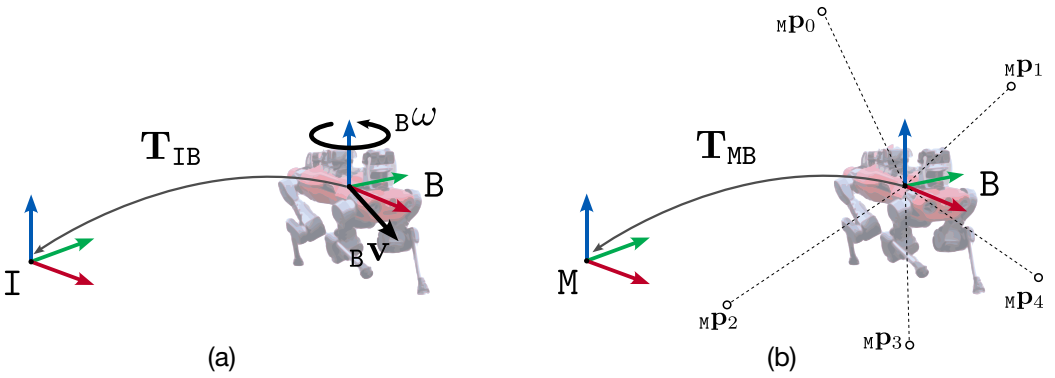

Figure 2.10: Odometry vs localisation systems as explained in Sec. 2.4.1. **(a) Odometry:** It determines the robot's state, usually its pose in the inertial frame `I` and the body velocity. **(b) Localisation:** It is concerned with estimating a consistent pose of the robot with respect to a map in the frame `M`.

Odometry

Odometry systems mainly provide a high-frequency (>100 Hz) estimate of the robot's pose and velocity. The pose estimate is smooth but only locally accurate as it suffers from incremental drift. For this reason, it is represented in a fixed, inertial frame `I`, known as *odom* frame. This frame will change every time the robot is deployed, as it is usually set at the starting location.

For legged robots, the odometry is estimated via multi-sensor fusion, in which high-frequency proprioceptive sensing and legged kinematics are combined with low-rate exteroceptive sensors using filtering algorithms [30, 29] or factor graph-based smoothing [117, 295]. In this thesis, we assume that the robot odometry is given, hence we always have access to the robot's pose in the `I` frame as well as the base velocity.

Localisation

On the other hand, *localisation* systems provide a pose estimate that is consistent with the environment and repeatable with every deployment. Hence they rely on a representation of the environment—a map—which defines its own reference frame `M`. Localisation systems use exteroceptive sensing to find the robot's location within the map, which is usually implemented in two steps, first finding initial

candidates of its approximate location (*place recognition*), and then a metric pose verification from the place candidates (*registration*).

Compared to odometry systems, localisation is less robot-specific, as it mainly relies on exteroceptive sensing. Still, the sensor setup of each robot is important to define the localisation strategy. In Chapter 4 we exploit the different cameras available on a legged robot to achieve robust visual localisation.

Global and Local Mapping

Global mapping systems build large-scale maps, which are used for localisation and global planning. Global maps do not necessarily need to be globally accurate but they must connect all the overlapping areas. *Local mapping* systems instead only represent the local neighborhood around the robot, being particularly useful for local planning. For this reason, they usually keep a rolling estimate of the map around the robot, which updates as the robot moves.

Mapping systems can produce different kinds of maps, which are tailored to different applications. We describe some possible distinctions as follows, following the presentation of Burgard, Hebert, and Bennewitz [41].

Metric vs Topological: *Metric maps* represent the world with precise coordinates, while *topological maps* represent places and relationships between them. *Topo-metric maps* combine elements of both approaches by using a topological representation for large-scale that defines local reference frames for each place [122].

2D vs 2.5D vs 3D: *2D maps* are better suited for flat areas, single-floor indoor environments, such as occupancy-based floor maps. *2.5D* maps also include elevation or height information, which is useful for field applications but cannot characterise overhanging obstacles. *3D maps*, such as voxel or octree maps, fully describe the space and are the most general though more computationally expensive.

Sparse vs Dense: *Sparse maps* only store landmarks, i.e. features that are mainly useful for localisation (e.g. ORB-SLAM's map [204]). *Dense maps* characterise surfaces and volumes, which can be easier for humans to understand, as well as being more suitable for collision avoidance and navigation (e.g. OctoMaps [121]).

Discrete vs Continuous: Most of the maps we are familiar with are *discrete*, as they explicitly represent the space as nodes, points, 2D cells, or voxels, among others. Maps can also parametrise the space using continuous functions such as a mixture of Gaussian distributions, making them *continuous maps* (e.g. Hilbert maps [229]).

Geometric vs Semantic: *Geometric maps* only represent geometric features such as volume, normals, or occupancy. *Semantic maps* store human-understandable features, such as classes, object labels, or other task-dependent features, such as energy consumption or *traversability*.

2.4.2 Local Planning

Local planning systems aim to safely drive the robot through the environment to reach a goal. To achieve this they typically use the local map, and execute path planning or reactive control algorithms to output a *twist* command (linear and angular velocity) to be executed by the locomotion controller. Fig. 2.11 illustrates this process.

Traversability

In order to determine what effort is required to reach the goal and which hazards exist along the route, local planners must know which areas are accessible by the robot, and what is the effort and hazards associated to move there as well as user preferences—this is known as *traversability* [33]. The concept of traversability is abstract, and for it to be useful we need to represent it numerically. In the literature, it is more common to use *occupancy*, which corresponds to a binary metric of traversability. More broadly, the concepts of *cost*, *risk*, *reward*, or simply

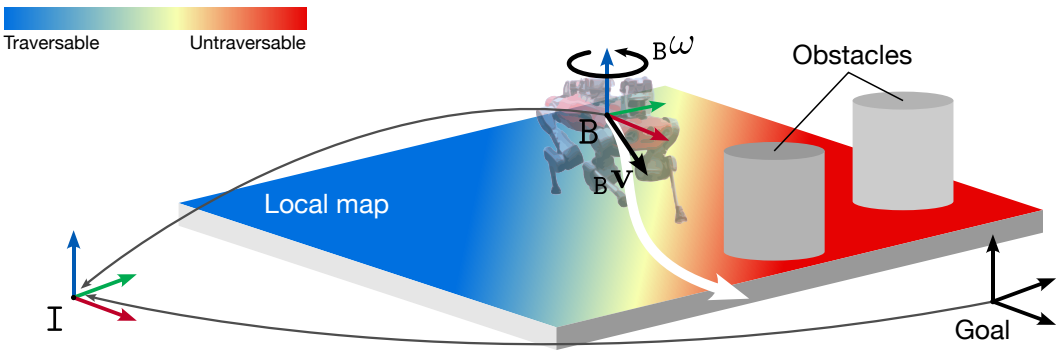

Figure 2.11: Local planning systems aim to reach a goal while avoiding obstacles. To achieve this, they use the traversability information encoded in the local map to find a safe path (white) used to determine the twist command which would allow the locomotion controller to follow it.

traversability score, are continuous metrics of traversability used to distinguish between different levels of navigation preference.

Fig. 2.12 presents a taxonomy for traversability estimation approaches we use in Sec. 3.3 to review the state of the art. This will be useful to contextualise the contributions of Chapter 6, where we present an online learning approach to determine traversability in legged platforms.

Motion Generation Strategies

The traversability map is used to find a safe path or control command to steer the robot towards the goal. There are different strategies to generate the desired motion:

Grid-based Local Planning: It uses a grid representation of the environment that encodes the traversability. By considering the grid as a lattice graph, we plan using algorithms such as Dijkstra [76], A* [115], or Fast Marching Method (FMM) [253]. The path is then tracked by a dedicated module that produces twists commands.

Sampling-based Local Planning: This approach finds explicit paths by sampling the space and checking the validity of the paths by evaluating its traversability and other costs. If the sampling is guided to find a path towards the goal, it is known as a *single-query planner*; RRT [163] is a classical example. If the search

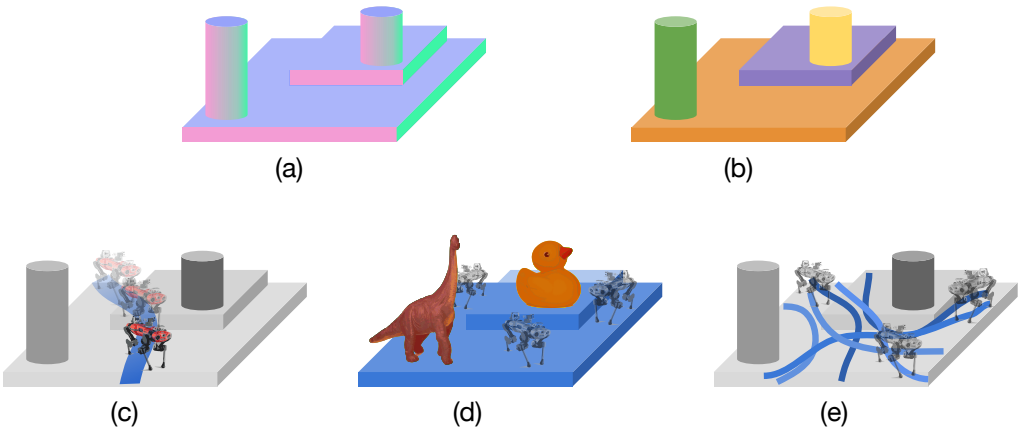

Figure 2.12: Different approaches to determine the traversability score in a local map. **(a) From geometry:** It is obtained from geometric features of the local map, such as surface normals. **(b) From semantics:** By using semantic segmentation approaches, it is possible to determine what is traversable by assigning costs to different semantic classes. **(c) From self-supervision:** It uses information that the robot generates itself from past trajectories or future predictions. **(d) From anomalies:** It considers the visited places as traversable, and any out-of-distribution sample as untraversable. **(e) From demonstrations:** It obtains a cost map from demonstrations in an inverse reinforcement learning fashion.

covers the full local map and the path network can be reused if the goal changes, it is a *multi-query planner*—such as PRMs [141]. These methods also need a dedicated module to track the found path.

Optimisation-based Local Planning: Instead parametrises the path with a continuous function and formulates an optimisation that enforces a minimum cost, defined by a combination of the traversability along the path, plus other criteria like path smoothness. Some examples are CHOMP [314] and GPMP2 [203].

Motion Primitives: Also known as *lattice planners*, these methods directly define candidate paths by kinematic models (e.g. Dynamic Window Approach [92]), sampling (e.g. Model Predictive Path Integral (MPPI) control [293]), or pre-computed trajectories (e.g. FALCO Zhang et al. [305]).

Reactive Approaches: These approaches do not find an explicit path but determine the output twist based on the instantaneous local map information or

raw sensing data. This is achieved by generating potential functions or manifolds for navigation, such as Artificial Potential Fields (APFs) [144] and Riemannian Motion Policies (RMPs) [231], or via end-to-end neural networks [138].

Deciding which local planning approach to use ultimately depends on the robot platform, its sensing capabilities, computation budget, and locomotion strategy. Grid-based, sampling, and optimisation approaches are usually preferred when the robot has slow dynamics and the planning horizon is larger, so finding an optimal path that leverages all the costs is more important that the time it takes to determine it. Motion primitives and reactive approaches are instead preferred when the platforms are highly dynamic, and staying safe is a stronger requirement than the optimality of the path. In Chapter 5 we present a local planner that enables a legged robot to navigate in challenging narrow spaces leveraging reactive and grid-based ideas.

2.4.3 Locomotion Control

The twist command produced by the local planner is executed by a *low-level locomotion controller*, which translates the reference into position and torque commands for the actuators using robot-specific models. For legged robots, these are *whole-body locomotion* controllers that fully specify the body configuration to generate the walking motion. If the controllers integrate exteroceptive sensing or local maps they are named *perceptive*; otherwise, they are called *blind* controllers.

Model-based Controllers: They implement the walking behaviour as an optimisation problem [291, 88, 187]. Given a desired gait pattern and reference base pose, they solve a Trajectory Optimisation (TO) or Model Predictive Control (MPC) problem to determine the footstep sequence and base configuration. This allows for precise control of the gait and feet positions but they are less robust to unmodelled perturbations or perception noise.

Reinforcement Learning-based Controllers: They frame legged locomotion as learning a *policy* that maximises a reward function that motivates walking behaviour [127, 257]. They are usually trained in simulation through millions of trials, where the robot specifications, perception input, and environment conditions are heavily randomised to handle unexpected situations during real deployment. Although they have demonstrated impressive performance in real-world applications, they still lack the precise locomotion control from model-based approaches.

In this thesis we do not approach the legged locomotion problem and we assume that controllers are available from the manufacturer or research collaborators [164, 197]. However, the solutions we present are designed to comply with their specifications and build on top of their capabilities to achieve autonomous navigation.

2.4.4 Mission Planning

For the purposes of this thesis, we will consider as a *mission* a list of potential goals the robot must navigate to. Then, *mission planners* are the interface to design and execute missions: on the one hand, they enable human operators to specify the places that must be visited; on the other, they schedule the goals and interact with the local planner to execute the navigation behaviour.

In Chapter 4 and Chapter 5, the mission planner will be specified by a Visual Teach and Repeat (VT&R) system, in which a human operator defines a path in advance, and then the robot has to track it for inspection and monitoring applications. Later, in Chapter 7 we will present a mission planner to autonomously survey forests.

3
Literature Review

This chapter provides an overview on the state of the art of the different navigation components we introduced in Sec. 2.4. Our discussion focuses on localisation methods, traversability estimation, and local planning, as they are the main topics covered in the following chapters. The goal is to provide a general overview as well as complementing the literature review included in each specific chapter with up-to-date references.

This chapter finally provides a bigger picture of real-world deployments of legged systems in the context of competitions, industrial applications, and deployments in natural environments.

3.1 Localisation and Mapping

We first review localisation methods, as they are revelant for Chapter 4. Our taxonomy is based on the underlying representation used for localisation—specifically metric, topological, or topo-metric.

3.1.1 Metric

Metric localisation is the most common approach to address localisation, as it involves estimating the robot's pose in a metric map [201]. Vision-based localisation

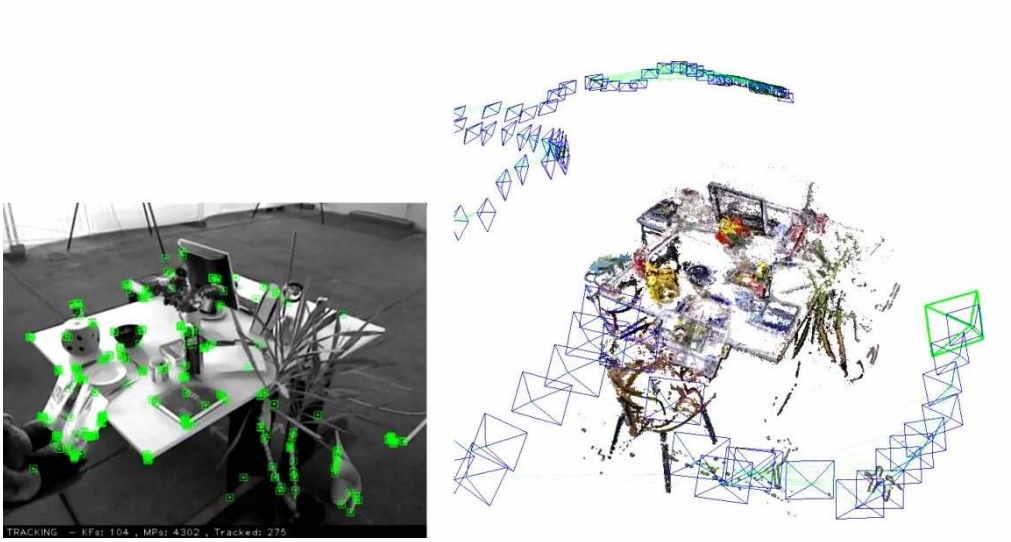

Figure 3.1: Metric localisation methods such as ORB-SLAM [204, 205, 206, 46] build a 3D map that is globally accurate. Source: University of Zaragoza.

(also known as *relocalisation*) requires a prior, globally-accurate map built from vision sensors. Such maps can be built offline with Structure-from-Motion (SfM) systems, or in an online fashion with visual Simultaneous Localisation And Mapping (SLAM) frameworks, as shown in Fig. 3.1.

While the core ideas of SfM go back to the work of Ullman [277], nowadays different software packages exist, such as COLMAP [246] or Meshroom [109]. On the visual SLAM community, different open-source alternatives have been developed, including systems based on sparse maps like ORB-SLAM [204, 205, 46], OKVIS [167] and Maplab [245], or dense reconstructions like RTAB-MAP [162] and ElasticFusion [290].

Regardless of the method, a visual map will be generally defined by a set of *keyframes*, i.e. poses and images that were used during mapping, and a dense or sparse 3D map. Then, localising in this map requires one to perform (1) *place recognition* to find the most similar keyframes to where the image was taken from, and (2) *registration* to determine the exact metric pose using the candidate keyframes.

Place Recognition

Place recognition [178], also named *image retrieval* in this context, performs a coarse search of candidate keyframes that could correspond to the location where the image

was taken. Methods to achieve this tipically extract a *global descriptor* from each image, and frame the problem as image retrieval on a database of keyframes. Some traditional approaches include DBoW2 [100] and FAB-MAP [65]. Modern methods use neural networks to determine a global descriptor, such as NetVLAD [12]. We will discuss them in more depth in the context of topological localisation (Sec. 3.1.2).

Registration

Once a set of candidates has been found, the registration step attempts to find an SE(3) estimate of the camera pose. This can be achieved via *feature matching* or *photometric alignment*.

Feature-based methods extract *local features* (distinctive points + descriptor) from the input image, which are matched against features stored in the map. This requires that the map must be built using the *same* features. Then a geometric 2D-3D pose estimation method is used to estimate the pose of the camera, such as PnP [89].

For the local features, classical methods such as SIFT [177], SURF [24], ORB [236], BRISK [166] and AKAZE [6] have been widely used for different applications, including visual SLAM systems such as ORB-SLAM or OKVIS. However, they cannot deal with drastic perspective and appearance changes. Learning-based features have overcome some of these challenges, enabling visual localisation with wide baselines or with changes such as day-night. Some examples include SuperPoint [73], R2D2 [233], DISK [276]. An extensive evaluation of local features in the context of wide-baseline pose estimation was recently done by Mishkin [200].

Photometric alignment does not require explicit matching and pose estimation but instead aims to align the current image to a projection of the visual map. The pose is obtained as a by-product of the alignment process, which is implemented as a minimisation of a *photometric residual* that quantifies the pixel intensity difference [83, 212]. Photometric alignment is more challenging, as it requires a good initialisation and cannot deal with large perspective changes. However, learning-based methods have enabled this approach by doing *dense feature map*

alignment. Here the concept of a feature is not the sparse, locally-generated feature as in feature matching methods, but a dense pixel-wise feature map obtained from a neural network architecture. Such methods are then called *feature-metric*, with GN-Net [263] and Pixloc [242, 171] being some examples.

Alternative Methods

Lastly, with the rise of deep learning, it has been also proposed to use neural models that directly regress the pose from images, without intermediate steps. PoseNet [143] is one of the earliest examples. However, they have been shown to poorly generalise to new environments, and hybrid approaches that combine learned features with traditional geometric verification methods are preferred; HLoc is an example of this approach [241].

Other localisation strategies use additional sensors, such as IMUs, as they can provide attitude and pose priors for the place recognition step. Visual-Inertial ORB-SLAM [206] or Lynen et al. [182] are some examples.

3.1.2 Topological

Topological localisation represents the space as a topological map of places [159]. Then, the localisation system must determine which node the robot is in, regardless of the metric pose.

Visual topological localisation (or *appearance-based localisation*) implement this as place recognition on a graph of keyframes. Each keyframe stores raw images or some global (*holistic*) descriptors that are used for retrieval. The descriptors are used to generate the *similarity matrix*, which encodes the descriptor distance to all the nodes in the graph and it is used to determine the robot's location. This is shown in Fig. 3.2.

One of the earliest appearance-based systems is by Ulrich and Nourbakhsh [278], using colour histograms as descriptors. FAB-MAP [65] and DBoW [100] brought the ideas to a wider audience, by computing global descriptors from local features such as SIFT or ORB using *bag-of-words* methods. Later approaches such

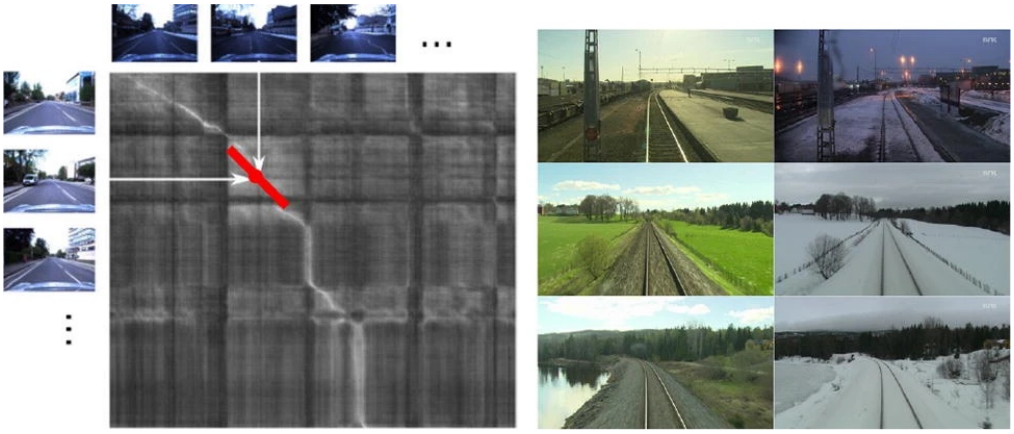

Figure 3.2: Topological methods, e.g. SeqSLAM [199], only rely on sequences of images without metric information. **Left:** The similarity matrix is obtained by computing the global descriptor distance between all the images in a sequence. **Right**: Examples of same-place localisation matches across different seasons that can be obtained by methods such as SeqSLAM. Source: Neubert, Schubert, and Protzel [208].

as SeqSLAM [199], Lynen et al. [181], and Vysotska and Stachniss [281] exploited image sequences in the similarity matrix to determine the current location.

Most of these methods are currently used as place recognition systems to reduce the search space during metric localisation, or as *loop candidate detectors* for loop closure optimisation. Schubert et al. [247] provides a comprehensive tutorial on place recognition systems and their applications.

The simplicity of topological localisation methods makes them easier to couple with navigation tasks, which might not require an explicit localisation process. These methods are usually based on determining *heading corrections* from a reference image in a visual-servoing fashion. Some of these approaches include Booij et al. [32] and Krajnik et al. [155] using local features, and Dame and Marchand [67] and Dall'Osto, Fischer, and Milford [66] based on photometric information.

3.1.3 Topo-metric

Topo-metric systems are a hybrid of the previous methods, see Fig. 3.3. Brooks [38] and Chatila and Laumond [51] argued that navigation could be achieved by simply having maps that are *relational*, describing local frames instead of a single global one. Globally they are represented by a topological map, while locally they

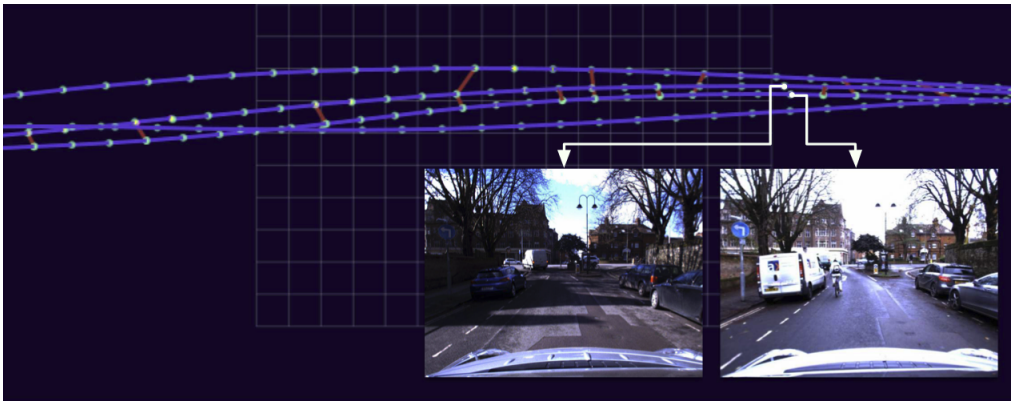

Figure 3.3: Dub4 [174, 174] is an example of topo-metric localisation as it builds a topological map at large-scale that stores metric information locally. Source: Linegar [172].

store metrically accurate information. This has the advantage of disregarding some of the hard constraints of purely metric approaches, such as global accuracy, while also enabling local metric estimation for navigation tasks.

Thrun et al. [268] and Simhon and Dudek [258] showed one of the earliest implementations of this idea. Howard, Sukhatme, and Mataric [122] later formalised it in the context of multi-robots SLAM as *manifold representations*.

Furgale and Barfoot [97] introduced a visual navigation pipeline for VT&R navigation, which made explicit that in order to re-traverse a previously visited path the robot only needed the topo-metric representation. This enabled kilometre-long deployment in spite of visual-odometry drift

Churchill and Newman [60] pushed the topo-metric approach further to aggregate multiple runs at different times of the day, named *experiences*, into joint topo-metric representations. Experiences are built in a similar manner to metric maps but (1) they do not necessarily enforce global metric accuracy by performing pose graph optimisation or bundle adjustment, and (2) they explicitly define local coordinate frames to represent the current pose. Localisation is performed similarly to metric approaches, via place recognition to identify the reference node and coordinate frame, and the metrically against that frame.

Experience-based localisation enabled visual localisation through day, night, and even seasonal changes. However, this introduced new challenges related to

experience-retrieval, which come as a previous step to the place recognition and registration procedures. Follow-up work by Churchill and Newman [59], Linegar, Churchill, and Newman [174, 173], Paton et al. [213], MacTavish, Paton, and Barfoot [186] addressed the multi-experience localisation problem, explicitly designing online strategies to query the most relevant experience for localisation.

Recently, motivated by the progress in learned network models, some authors have proposed to disregard the explicit experiences and rely on the expressiveness of the local features instead. For example, Sun et al. [264] trained a dedicated model to address day-night shifts; Gridseth and Barfoot [108] exploited classically-built multi-experience frameworks to obtain season-invariant features. These methods have been shown to achieve multi-experience localisation from a single experience.

3.2 Local Planning

Here we discuss local planning approaches, centering the review on legged platforms. Similarly to the previous section, we also define a taxonomy to organise the presentation of these different methods, which we split into *classical*, *hybrid*, and *end-to-end learning* methods.

3.2.1 Classical

We consider as classical methods those that implement local planning using analytical models, generally using geometric definitions of obstacles and traversable space. Some examples are shown in Fig. 3.4.

Chestnutt and Kuffner [54] demonstrated one of the earliest formulations for navigation with legged platforms. They used A* to generate feasible trajectories to command a humanoid robot in different maze-like environments. Chilian and Hirschmüller [55] presented the first method using a local elevation map, and relied on a grid-based D* algorithm for planning with an hexapod robot.

Wooden et al. [296] showed first outdoor demonstrations on the Boston Dynamics BigDog robot. Their approach was based on segmenting geometric obstacles out of a point cloud, projecting them onto a local cost map, and computing an A*

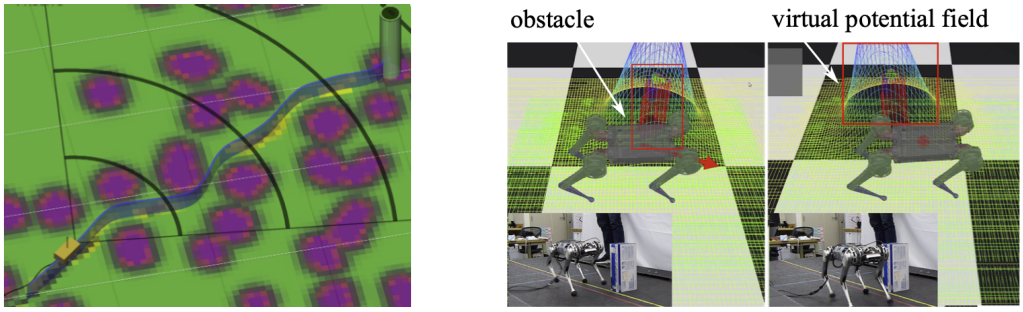

Figure 3.4: Classical local planning uses traditional planning techniques to direct the robot towards a goal defined by a human or mission planning system. **Left:** A*-based planning on the BigDog quadruped. Source: Wooden et al. [296]. **Right:** Potential field-based local planner for the MiniCheetah robot. Source: Kim et al. [146].

path using the cost map at 1 Hz. Johnson et al. [135] used a reactive planner to achieve autonomous hill climbing with small-scale legged platform, using local geometry information provided by a LiDAR.

Arain et al. [11] evaluated different grid-based planning algorithms in the context of quadruped navigation on rough terrain, including Dijkstra, A*, and other variants. The evaluations were carried out on indoor testing terrains. Wermelinger et al. [289] presented a pipeline for the StarlETH quadruped, based on a local elevation map [87] that was analysed geometrically to determine traversability, combined with RRT* to generate navigation plans. For navigation in confined spaces, [40] presented a whole-body motion planner, using an elevation map for representing the ground and ceiling. Planning was offline, based on CHOMP, and the execution was carried out open-loop.

Recent approaches have focused on real-time planning. Kim et al. [146] presented an approach for local planning with small quadrupeds based on potential fields. For 3D planning, Dudzik et al. [78] proposed an A* formulation, while Norby and Johnson [210] used RRT-Connect for faster planning enforcing dynamic constraints.

In the context of the DARPA SubT Challenge, the teams developed efficient local planning solutions to deal with rough terrain in urban and underground environments. Fan et al. [86] used a TO planner that minimised the Conditional Value-at-Risk (CVaR) to determine the safest path to the goal, which was then tracked with a kinematic MPC. Wellhausen presented ArtPlanner, a legged robot-specific planning solution based on LazyPRM*, which continuously updated a dense

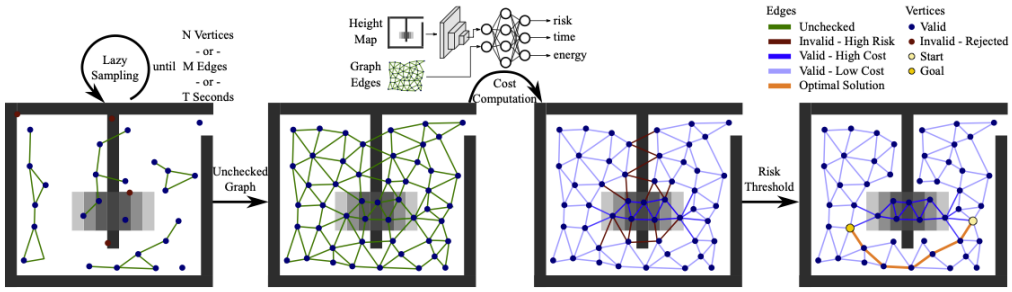

Figure 3.5: ArtPlanner is a hybrid local planning method that uses learned edge costs within a classical LazyPRM* planning scheme. Source: Wellhausen and Hutter [286].

local navigation graph for multi-query planning [287, 286].

In the context of classical planners, dealing with moving obstacles has not been widely explored. Gaertner et al. [99] presented an approach for collision avoidance directly into locomotion MPC. While the environment was represented with an occupancy grid, moving obstacles were treated separately and represented as cylinders.

3.2.2 Hybrid

By hybrid methods we meant those who fundamentally treat the local planning problem in a classical way but introduce learning-based components for improved performance (Fig. 3.5). Note that for this review we do not consider methods that rely on learned traversability, as they will be reviewed in Sec. 3.3.

Guzzi et al. [110] presented an approach that learned local motion estimations, used as edge costs in a sampling-based planner. These costs were parametrised as a neural network, and they were trained in simulation. Yang et al. [301] extended this idea to legged robot navigation, augmenting the predictions to risk, energy and traversal time. This approach was later used by the CERBERUS team in the DARPA SubT for local planning on their legged platforms [286].

Learning has also been used to integrate proprioceptive information into local planners. Fu et al. [95] trained a model to predict collisions from proprioception, which was used to spawn new obstacles in a cost map used by a grid-based local planner. This was deployed on a small quadruped, and it enabled navigation in environments with glass walls that were not detected by depth sensing.

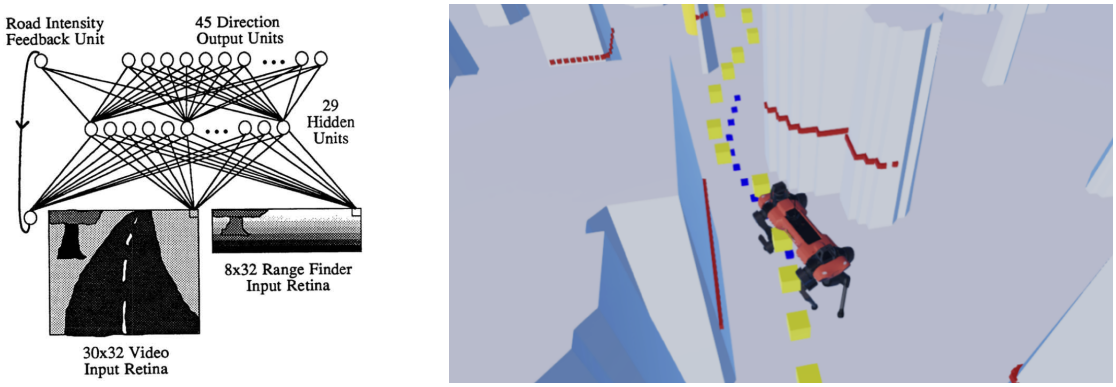

Figure 3.6: End-to-end methods use neural models from raw-sensing to control commands. **Left:** ALVINN is one of the earliest applications of this approach for autonomous driving. Source: Pomerleau [218]. **Right:** An application of end-to-end methods for navigation using sparse LiDAR data. Source: Kim, Kim, and Hwangbo [148].

Another strategy has been using neural models to provide priors for classical planners. Richter and Roy [234] trained an autoencoder model to predict collision probability in a Partially Observable Markov Decision Process (POMDP) formulation. The model was also used to identify unknown scenarios and make the planner more conservative. Yang et al. [302] trained a neural model that predicts a set of waypoints used to initialise a TO planner from raw depth sensing, which was deployed on an ANYmal quadruped.

3.2.3 End-to-end

Lastly, we review systems that disregard any planning procedure and achieve local planning and navigation by means of end-to-end neural models, see Fig. 3.6. These methods are an active area of research nowadays, enabled by the progress in deep learning and computing hardware.

The earliest use of end-to-end learning for control are with the ALVINN system by Pomerleau [218], which introduced the core ideas of end-to-end methods, by processing raw input (images) and producing a control command (Fig. 3.6, left). ALVINN was developed for autonomous driving applications, and demonstrated impressive results on hardware for the time [61].

Michels, Saxena, and Ng [195] used reinforcement learning to learn control policies for outdoor navigation, which is also the approach used more recently in BADGR [138]. Giusti et al. [105] instead formulated the problem as supervised learning, using data collected by humans in forest trails, and learning a control policy to correct a drone's heading. Pfeiffer et al. [215] used a similar approach but relying on a global planner in simulation as a supervision signal.

In the context of legged robots, Lobos-Tsunekawa, Leiva, and Ruiz-del-Solar [176] used reinforcement learning to learn control policies from colour-segmented images for navigation in a robot football field. Hoeller et al. [119] used depth cameras to learn a local planning policy that enabled a quadruped robot to navigate in narrow spaces with moving obstacles. Kim, Kim, and Hwangbo [148] also trained an end-to-end local planning framework for navigation with sparse LiDAR sensing.

The main advantage of end-to-end approaches is that they enable a tighter interaction between perception, local planning and locomotion. This enables more complex behavior that are difficult to design with a classical pipeline otherwise. Rudin et al. [237] recently presented an approach for highly dynamic motion skills, that enabled a quadruped robot to jump over large gaps and climbing high obstacles. The key of the approach was on formulating the problem as *reaching a goal given a time budget*, instead of *velocity tracking* as it has been used in previous works. This enabled the locomotion policy to learned a diverse set of locomotion behaviours that were not constrained to trotting gaits.

On the other side, the entanglement of perception and action that end-to-end systems propose also makes it more difficult to improve the systems without retraining. It also presents challenges when RGB vision or semantics need to be included, as they are more difficult to simulate. In the next section we discuss traversability estimation methods, which can easily integrate these modalities by exploiting the modularity that classical and hybrid methods rely on.

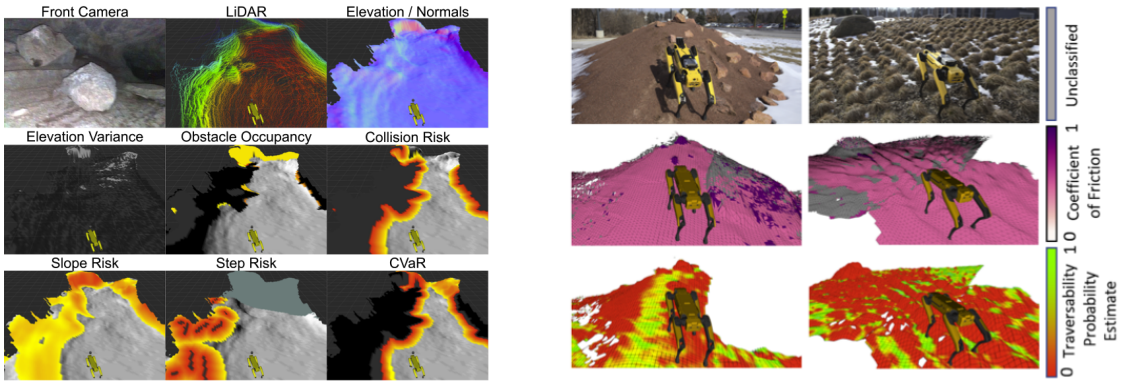

Figure 3.7: Geometric and semantic approaches to traversability estimation. **Left:** STEP [86] uses different geometric features to get a risk metric (CVaR). **Right:** SELMap [85] predicts terrain parameters using a semantic segmentation network.

3.3 Traversability Estimation

Traversability estimation describes which areas can be traversed by a specific robotic platform [33]. Here we provide a deeper review of the approaches we summarised in Fig. 2.12.

3.3.1 Geometry

The most basic definition of traversability is based on the concept of occupancy [201]. A similar idea was used for the autonomous deployment of BigDog by geometrically characterising obstacles [296]. More sophisticated geometric analysis was later used to deploy the robot day and night, including analysis of vegetation and negative obstacles [21].

Wermelinger et al. [289] performed a unified geometric traversability analysis on a multi-layer elevation map, which enabled the seamless integration of different costs as part of the same representation. A similar idea was later used by Fan et al. [86] to define the CVaR from different geometric terrain analyses (Fig. 3.7, left).

Learning-based models have been also used to define traversability from geometry. For legged platforms, they have mostly relied on the local elevation map used for navigation as the main input for neural network models trained from simulation data [52, 301]. Wellhausen and Hutter [287] also used the local map and a few

hand-labelled samples to train a network that scored the areas where a legged robot could step on.

Frey et al. [93] extended the approach by Yang et al. [301] to 3D representations, such as voxel maps. They learned a sparse 3D convolutional model to regress legged robot traversability from simulated data equivalent to 57 years of real-world locomotion experience. Ruetz et al. [238] also used volumetric representations to determine traversability for a wheeled platform in forests.

The previous methods assume that terrain traversability can be directly obtained from geometric information only, although natural scenes challenge this assumption. Homberger et al. [120] proposed to tackle this problem by estimating the underlying ground surface hidden by vegetation by using a 2D Gaussian Process fitting the robot's footsteps. An alternative approach has been estimating traversability directly from the terrain semantics, which is covered in the following section.

3.3.2 Semantics

For legged robots, geometry does not necessarily encode all the information that is needed for navigation. Semantic information provides additional input that can guide the navigation process in more challenging environments, such as natural, off-road scenes.

Bradley et al. [36] presented an approach for scene understanding from volumetric representations for the LS3 quadruped. It was based on a random forest model to determine semantic labels trained with a large dataset of outdoor environments collected over 28 months. A similar idea was explored by Belter, Wietrzykowski, and Skrzypczynski [25] to develop a multi-legged whole-body planning system able to negotiate different terrain properties. An alternative semantic approach was presented by Vasilopoulos et al. [280], who developed a reactive navigation approach that relied on a 2D representation encoding semantic information given by object detections in indoor environments.

More recently, it has been proposed to use semantic systems that directly encode locomotion-specific layers. For example, Ewen et al. [85] presented a *semantic*

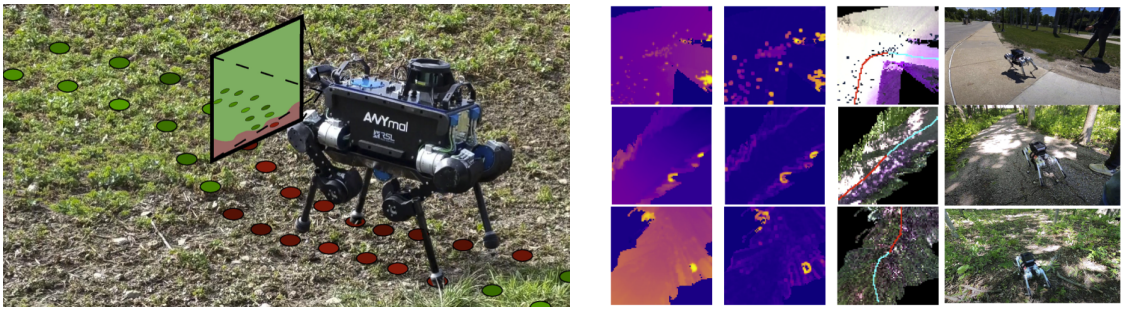

Figure 3.8: Traversability from self-supervision and demonstrations. **Left:** Self-supervision based on the reprojection of footholds into images, presented by Wellhausen et al. [285]. **Right:** Explicit demonstrations can be leveraged in an inverse reinforcement learning framework, as carried out by Gan et al. [101].

elevation map that encodes friction and contact model parameters (Fig. 3.7, right). Yang et al. [303] learned a model that determines the gait parameters and speed that the robot should use for different terrains.

Beyond legged robots, the recent DARPA RACER challenge is pushing new developments for semantic off-road understanding of wheeled platforms [254].

3.3.3 Self-supervised

A different approach to traversability estimation that does not require to define geometric features or semantic classes, aims to exploit the robot's experiences during the data collection process. As the robot has navigated certain places in the past, there is an implicit prior on those places being traversable, which is knowledge that can be used for self-supervision.

This idea was heavily explored as part of the Learning Applied to Ground Vehicles (LAGR) program [130], a DARPA project to foster learning-based solutions for off-road wheeled navigation. Some of the approaches developed include Sofman et al. [259], Kim et al. [147] and Bajracharya et al. [20], which presented self-supervised, online approaches to traversability estimation. Hadsell et al. [112] also showed first applications of neural networks in this context as image feature extractors.

Wellhausen et al. [285] presented one of the earliest self-supervised approaches for legged robots (Fig. 3.8, left). It reprojected the robot's foothold positions into recent camera images to create a supervision signal to learn a neural model to

predict the ground friction and other terrain properties. Other works for wheeled robots include WayFast [103], which used a similar approach for labeling but with the predicted path from an MPC. Sathyamoorthy et al. [243] used IMU signals and odometry error to predict cost maps from images. Castro et al. [50] used IMU data as a proprioceptive signal as well as the robot velocity to produce velocity-conditioned traversability maps.

Zürn, Burgard, and Valada [315] also learned to predict different terrain properties from images. However, they used the sound generated by the robot when traversing different terrain as a complementary signal to guide a weak semantic segmentation process instead of an explicit label. Similar self-supervised learning schemes have been explored recently by Seo, Sim, and Shim [252].

3.3.4 Anomaly Detection

Many self-supervised methods suffer from data imbalance from the self-supervision process: most of the labels are from traversable areas, as it is hard to obtain negative labels without hitting obstacles. *Anomaly or novelty detection* methods use this as a proxy for traversability: by only modelling the distribution of places that have been visited (positive samples), any out-of-distribution sample that is not modelled by the distribution (outliers) will be considered as untraversable.

Pomerleau [219] presented one of the earliest applications of anomaly detection in robotics. As part of the ALVINN neural driving system, they introduced a reconstruction loss to quantify the reliability of the output, and consequently change between neural models trained for different situations. Many years later, Richter and Roy [234] used a similar scheme to determine when the robot was navigating an unknown environment, making the behaviour more conservative. Recently Ji et al. [134] used a similar approach to anticipate failures when navigating in crop fields.

Wellhausen, Ranftl, and Hutter [288] presented an explicit application of anomaly detection for traversability by learning a distribution of known terrain from vision and then classifying out-of-distribution samples as untraversable. The prediction was projected on a local map, and used for classical planning with a legged plaform.

Recently, systems such as ScaTe [251] have presented further extensions based on 3D LiDAR data that leverage anomaly detection and traversability estimation, in a similar way to the system we will present in Chapter 6.

3.3.5 Learning from Demonstrations

An important line of research takes a different approach to traversability estimation motivated by Markov Decision Processes (MDPs). The idea is that, given expert navigation demonstrations, it is possible to determine a reward function that mimics the expert's behaviour. As the problem deals with infering the reward from roll-outs of a policy, which is the opposite of what reinforcement learning algorithms do, it is known as *inverse reinforcement learning* [209, 1].

Ratliff, Bagnell, and Zinkevich [230] presented an approach to learn reward cost-maps from aerial imaginery. Given some map features, such as elevation and map colours, it aimed to find an optimal reward map expressed as a linear combination of the features. For this it used the maximum margin principle, which enforced that the given demonstration was better represented in the output reward map compared to other possible behaviours. Ziebart et al. [313] observed that this approach did not necessarily lead to the optimal behaviour for all the demonstrations, as it aimed for a unique reward map. Instead, they proposed to use the maximum entropy principle (MaxEnt) to find an optimal reward distribution over all the demonstrations. Wulfmeier et al. [298] extended the MaxEnt approach to learn non-linear reward maps by parametrising it with a deep neural network, also enabling them to learn features directly from raw data.

The MaxEnt framework has become the standard approach for different field applications. Zhang et al. [309] extended it to introduce kinematic data that conditioned the reward maps to the vehicle velocity; Gan et al. [101] followed-up this approach to use IMU sensing instead of the kinematic models and applied it to legged robot navigation (Fig. 3.8, right).

3.4 Real-world Deployment of Legged Robots

In this final section we review different applications and deployments of legged robots in the real world. The aim is providing a general overview of the achievements to date as well as contextualising the autonomous forest inventory system that is introduced in Chapter 7.

3.4.1 Competitions

Competitions are an intermediate step between lab research and real applications. They provide a test bench for research and a motivation for new questions. One of the earliest examples is the RoboCup (1997 to date) [150]. With the purpose of developing a *soccer* (football) team to play against the world champions in 2050, it has been a driver for the development of quadruped and humanoid robotics. It has involved not only the development of locomotion and navigation algorithms but also cooperative behaviour [15].

Figure 3.9: Worldwide competitions have fostered the development of legged robots over the years. **Left:** RoboCup Standard Platform League (SPL). Source: ubahnverleih, Wikimedia Commons. **Center:** DARPA Robotics Challenge. Source: MIT DRC team. **Right:** DARPA SubT Challenge. Source: NASA/JPL-Caltech.

The DARPA Robotics Challenge (DRC), held from 2012 to 2015, was an effort to develop humanoid robotics for disaster response scenarios [156]. It involved highly complex challenges from locomotion over uneven terrain and door opening, to using manual tools and driving a vehicle. Atkeson et al. [17] summarises the main learnings from the teams to deploy humanoid robots in the real world.

More recently, the DARPA SubT challenge (2018-2021) was a general challenge on robot exploration and search in underground environments Chung, Orekhov, and Maio [58]. Nevertheless, it ended up being an ideal use-case to highlight the

advantages that legged robots offer in unstructured environments, reflected by the growing adoption of legged platforms across the competition. In the finals, team CERBERUS [274] achieved up 700 m of autonomous operation with ANYmal platforms, while CSIRO Data61 [154] and CoSTAR [4] covered similar distances with Spot robots.

Similarly, the ESA-ESRIC Space Resource Challenge (2021-2022) was another competition that motivated the potential use of legged platforms in space applications, particularly moon prospection [13, 14].

3.4.2 Industrial Deployments

One of the main applications of legged platforms has been inspection and monitoring of industrial facilities. Boston Dynamics' Spot has been widely adopted as a platform for inspection of manufacturing, construction, and mining sites [34]. One of the main features it offers is the AutoWalk system [80], which enables to deploy the robot autonomously in a VT&R fashion. ANYbotics–and their quadruped ANYmal [10]–has focused on oil and gas companies where the robot needs to be certified against water and dust (IP67), and even explosive atmospheres (ANYmal X). They offer similar autonomous inspection solutions that enable periodic monitoring in industrial facilities.

Other real-world deployments include sewer inspection. Kolvenbach et al. [153] showed a research-driven application of legged robots for concrete inspection using haptic sensing. The mission was executed using a shared autonomy approach.

Last mile delivery and warehousing are other use-cases currently explored. For example, Agility Robotics are promoting the use of their Digit biped robot [5] for such applications.

3.4.3 Natural Deployments

Legged robots have been envisioned as an ideal platform for operation in natural environments. One of the earliest cases is the autonomous deployment of the Boston Dynamics BigDog robot in forests [296, 21]. The Boston Dynamics LS3 was also used

Figure 3.10: Industrial deployments of legged platforms. **Left:** Spot used for radiation mapping. Source: Boston Dynamics. **Center:** ANYmal X is a platform certified for explosive atmospheres. Source: ANYbotics. **Right:** Digit robots executing a warehousing demo at ProMat 2023. Source: Agility Robotics.

for a 2-years long data collection in forests, rainforests, and medows, among other locations [36]; autonomy was only reported for *follow-the-leader* behaviour [68].

The RHex quadruped from the University of Pennsylvania [240] has been widely deployed in different environments for environmental studies. Qian et al. [221] deployed the robot in the White Sands National Monument in New Mexico to autonomously collect data from aeolian proccesses, i.e. erosion, transportation and deposition of desiment by the wind. In follow-up experiments in the same environments it was deployed for soil prospection [223].

Ilhan, Johnson, and Koditschek [129] executed a proof-of-concept kilometre-long autonomous hill climbing in a forest with RHex. Wilson et al. [294] later used it to study the soil strenght distribution at the Wissahickon Valley Park, Philadelphia in a teleoperated mission.

A robotic crocodile, K-Rock [193, 192], was deployed in Murchinson Falls, Uganda, for two weeks to film crocodiles and lizards in the wild. Though the goal was on entertainment (filming the *Spy in the Wild* BBC documentary), it illustrates the versatility that legged platforms offer.

In Chapter 7 we present a system for autonomoust forest surveying with legged robots. Our approach addresses the full navigation stack as well as a domain-specific pipeline to analyse trees and extract relevant forestry attributes. Through extensive field deployments, this work demonstrates the potential of legged platforms to achieve important tasks in uncertain, natural environments.

Figure 3.11: Natural deployments of legged platforms. **Left:** BigDog robot autonomously navigating in a forest. Source: BostonDynamics [296]. **Center:** RHex robot collecting soil data in White Sands National Monument. Source: University of Pennsylvania. **Right:** K-Rock in crocodile mode in Uganda. Source: EPFL.

3.5 Conclusion

In this chapter we reviewed systems and methods for robot navigation, focusing on solutions developed for legged platforms. As developing navigation systems requires the interaction of different sub-systems from state estimation to planning, our literature review presented the state of the art of the main modules involved as well as fully integrated systems in context.

In the next four chapters, we will present specific contributions to topo-metric localisation, reactive local planning, learned traversability estimation, and real-world deployment of legged robots. Each chapter is self-contained, and will also provide a project-specific literature review of the state of the art by the time the work was developed. After each chapter, we will discuss the main contributions, its relationship with other newer approaches we have reviewed in this chapter, and present open challenges that will be further discussed in Chapter 8.

4
Multi-camera Visual Localisation

We first address the localisation problem. Specifically, we consider the case of industrial inspection using VT&R approaches, where robots must execute a *mission* which as been defined by a human operator (*teach* step) and then perform reliable point-to-point navigation through the previously traversed environment (*repeat* step). The main sensors used are stereo cameras placed on the legged robot's body.

As the robot is expected to operate in industrial facilities, the main challenge is given by transient changes around the robot that can affect the visual input from the different cameras. A critical issue is that passers-by could partially or fully block the Field-of-View (FoV) of the onboard cameras, deteriorating the current localisation estimate.

This chapter presents a new localisation approach that uses multiple cameras carried by a legged robot. In contrast to previous methods that addressed this in a single estimation procedure using all the available hardware-synchronised cameras, we investigated an active strategy that *prioritises* a single view from a set of non-synchronised cameras. This is achieved by measuring the contribution of each camera during the *teach* step, given by the negative entropy scores computed from each camera when localising along the path. We used these scores to learn Camera Performance Models (CPMs), which are later used during the *repeat* traversals to select the single most suitable camera.

4.1 Learning Camera Performance Models for Active Multi-Camera Visual Teach and Repeat

The article in this section was presented at the *IEEE International Conference on Robotics and Automation (ICRA)* 2021 [190]. An accompanying video is available online at: https://youtu.be/iAY0lyjAnqY.

Remark The following paper uses $\underrightarrow{\mathcal{F}}_C$ to denote the camera frame `C`.

Errata Equation (1) in the paper is missing the covariance of the reprojection term. Instead, it should correctly state:

$$\arg\min \sum_{\mathbf{z}\in\mathcal{Z},\mathbf{p}\in\mathcal{P}} \| \mathbf{r}_{\text{reproj}}(\mathbf{z},\mathbf{p}) \|^2_{\mathbf{\Sigma}_{\text{reproj}}} + \| \mathbf{r}_{\text{prior}}(\mathbf{T}_{\text{prior}}) \|^2_{\mathbf{\Sigma}_{\text{prior}}}$$

Learning Camera Performance Models for Active Multi-Camera Visual Teach and Repeat

Matias Mattamala, Milad Ramezani, Marco Camurri, and Maurice Fallon

***Abstract*— In dynamic and cramped industrial environments, achieving reliable Visual Teach and Repeat (VT&R) with a single-camera is challenging. In this work, we develop a robust method for non-synchronized multi-camera VT&R. Our contribution are expected Camera Performance Models (CPM) which evaluate the camera streams from the teach step to determine the most informative one for localization during the repeat step. By actively selecting the most suitable camera for localization, we are able to successfully complete missions when one of the cameras is occluded, faces into feature poor locations or if the environment has changed. Furthermore, we explore the specific challenges of achieving VT&R on a dynamic quadruped robot, ANYmal. The camera does not follow a linear path (due to the walking gait and holonomicity) such that precise path-following cannot be achieved. Our experiments feature forward and backward facing stereo cameras showing VT&R performance in cluttered indoor and outdoor scenarios. We compared the trajectories the robot executed during the repeat steps demonstrating typical tracking precision of less than 10 cm on average. With a view towards omni-directional localization, we show how the approach generalizes to four cameras in simulation.**

I. Introduction

Following previously traversed paths is a useful capability for mobile robots. This is essential for missions, such as autonomous inspection and monitoring, where the same path is repeatedly traversed. This has motivated research into mapping and localization systems [1]. In particular, vision-based navigation systems such as Visual Teach and Repeat (VT&R) [2] have enabled different robots to repeat known routes without requiring metrically accurate maps. Visual sensors are inexpensive, lightweight, and provide both appearance and geometric information about the robot's surroundings.

We are interested in legged robots, which are promising for inspection tasks due to their versatile mobility on challenging terrains. However, quadrupeds such as ANYmal [3] are holonomic and move with dynamic gaits, such as trotting and climbing stairs. These motions cause rapid feature change, blur and tracking failure, making it difficult to achieve VT&R with a single camera. Since cameras have a limited Field-of-View (FoV), redundancy in visual sensing, i.e using multiple cameras, allows such platforms to increase their vision capabilities, but at an increased computational cost and integration complexity (synchronization and calibration).

In this work, we present a visual navigation system for mobile robots based on the VT&R paradigm that takes advantage of multiple cameras to stay localized in presence of clutter in narrow spaces (Fig. 1). In contrast to previous approaches, which typically process hardware synchronized cameras simultaneously [4], [5], [6], we instead select the camera providing the best performance for each segment of the path. This approach allows us to achieve accurate path tracking while also being robust to dynamic changes in the environment. Since cameras hardware synchronization is not required, our approach is more flexible, scalable and easier to deploy than traditional methods. The contributions of our work are summarized as follows:

Fig. 1: Visual Teach and Repeat (VT&R) allowed us to quickly deploy the ANYmal robot for industrial routine inspections. In this work, we extend previous approaches by augmenting the topo-metric map with a *Camera Performance Model* (CPM). This allows us to dynamically choose the most reliable camera during a repeat, such as the front (blue) and rear (orange) cameras illustrated.

- A VT&R system that uses multiple cameras during the *teach* step and learns performance models for each stream. These models are used in the *repeat* step to actively select the most informative camera.
- Deployment of our VT&R system on a quadruped robot, which enables it to autonomously follow a path it has traveled before using only vision, in spite of occlusions and dynamic locomotion gaits.
- Evaluation of the system in simulated and real scenarios with an ANYmal quadruped where peculiarities of legged systems are discussed. To the best of our knowledge, this is the first academic demonstration of VT&R on a legged platform.

The remainder of this paper is structured as follows. Sec. II discusses the related work. Sec. III describes our Active Multi-Camera VT&R system. Experimental results are presented in Sec. IV and conclusions are drawn in Sec. V.

The authors are with the Oxford Robotics Institute at the University of Oxford, UK. {matias, milad, mcamurri, mfallon}@robots.ox.ac.uk

II. Related Work

This section discusses previous VT&R approaches as well as methods that exploit mapping or teach steps to improve the performance in subsequent traversals.

A. Visual Teach and Repeat

A variety of VT&R systems have been developed for wheeled robots [7], [8] and drones [9], [10], [11]. The main idea behind VT&R is that a topo-metric feature map is collected during a teach run. The map can then be used to guide the robot along the path learned during the teach run. Only local consistency between the path and the map is required to achieve path following.

In the past 10 years, most research on VT&R has been focused on improving the robustness against long-term environmental changes, which can compromise visual navigation. The problem has been commonly approached by creating and analyzing a varied set of traversals of the same route (also called *experiences*). This approach is known as *Experience-Based Navigation (EBN)* [12] or *Multi-Experience Localization (MEL)* [13]. Both systems have been deployed in autonomous cars, ground, and aerial platforms, with emphasis on seasonal and day-night reliability [14], [15], [10].

In the past, using multiple cameras in the context of VT&R has only been applied to deal with appearance changes. Paton *et al.* [16] used front-view and rear-view synchronized cameras on a Husky robot to make their VT&R system more robust to lighting conditions. However, this process was passive, as both cameras were processed together, and no prior information about the path was utilized.

In this work, we are less concerned about long term changes, such as day-night shifts or seasonal changes. Instead, we focus on abrupt changes, such as motion blur due to aggressive motions, occlusions caused by people or vehicles, camera exposure changes, and rapid scene change in cluttered locations. To this end, we also explore the use of multiple cameras, but we actively select the most informative camera during the repeat step. This is done by comparing online the current and the expected performance inferred from previous traversals.

B. Leveraging Past Experiences

Because VT&R systems have an explicit teach step, the knowledge collected in this step can be used to optimize the performance during the repeat execution. As with other methods, the main assumption is that the route taken does not change drastically, so the collected information (features and models) can be leveraged in future operation.

The work of Churchill *et al.* [17] collected localization statistics from several teach passes to train a Gaussian Process (GP) model of the *localization envelope* of a given path. This embedded the localization performance of the system, and it was used to predict potential failures.

Ondruska *et al.* [18] showed how to reduce the energy requirements of a planetary rover by scheduling when to use its cameras. They showed that significant energy savings could be achieved while still reliably localizing in open, park-like environments. Warren *et al.* [19] exploited previous VT&R experiences on a drone to actively control a gimbal system so as to reduce the orientation error between the camera and the viewpoint used while recording experiences.

Recently, Zhang *et al.* presented a perception-aware navigation approach, named *Fisher Information Fields* [20], [21]. This map representation allocates the expected localization performance (given by the Fisher Information matrix) in a discrete grid, which is used to compute the expected information for arbitrary poses. While we base our localization performance metrics on similar principles, in this work we focus on topo-metric representations instead.

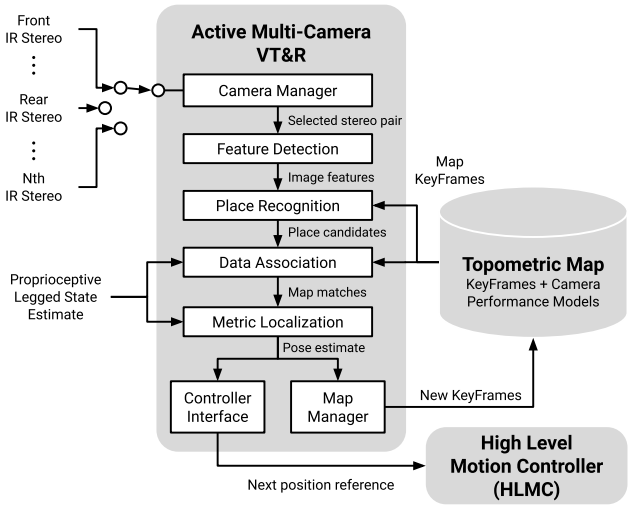

Fig. 2: Block diagram of our active multi-camera VT&R system. We augmented the topo-metric map with a Camera Performance Model (CPM) for each camera. The Camera Manager queries the models online to select the active camera.

III. Method

Our goal is to develop a multi-camera Visual Teach and Repeat system for robots with multiple cameras, with a focus on quadruped robots.

A. System Overview

The main modules of our system are shown in Fig. 2. The structure follows the approach taken by Furgale and Barfoot [2], with three main differences:

- We use a slowly drifting proprioceptive state estimate (for a legged robot, provided by a system such as TSIF [22]) instead of visual odometry to simplify the mapping process and reduce the computational burden in the teach step (Sec. III-B).
- The topo-metric map is augmented with a *Camera Performance Model* (CPM) for each camera, which is learned using the teach trajectory (Sec. III-C).
- A new *Camera Manager* module determines which camera should be processed at the current instance. During the teach, the manager processes every camera in turn. During repeat, it exploits the CPM to select the most suitable camera for a specific part of the path (Sec. III-D).

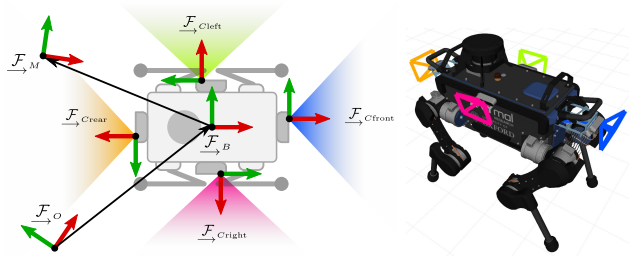

Fig. 3: Top-view diagram of the frames and color convention to identify the cameras throughout this paper.

We consider that up to 4 cameras could be attached to the main body (front, rear, left and right). Fig. 3 illustrates the coordinate frames and the position of the cameras.

The proprioceptive state estimate is defined in the fixed odometry frame $\underrightarrow{\mathcal{F}}_O$, while the VT&R localization is defined in the fixed map frame $\underrightarrow{\mathcal{F}}_M$ corresponding to the teach path. The moving frame $\underrightarrow{\mathcal{F}}_B$ is rigidly attached to the robot's chassis, as well as the four camera frames $\underrightarrow{\mathcal{F}}_C$.

B. Topo-metric Mapping with Multiple Cameras

In our system, the teach step builds a topo-metric map of the path over which the robot is teleoperated. The map is represented by a collection of *keyframes* $\mathcal{K}$ connected by relative transformations (as in [23]) which are expressed in the map frame $\underrightarrow{\mathcal{F}}_M$ relative to the body frame $\underrightarrow{\mathcal{F}}_B$. Each keyframe $k \in \mathcal{K}$ stores:

- A stereo image pair, tagged with the source camera.
- Triangulated AKAZE [24] features $\mathcal{P}$ for metric pose estimation relative to the keyframe.
- A Bag-of-Visual-Words vector, based on DBoW2 [25].
- The body pose of the keyframe in the map frame $\mathbf{T}_{MB} \in \mathrm{SE}(3)$.
- The intrinsic calibration between the camera and the body pose $\mathbf{T}_{BC} \in \mathrm{SE}(3)$.
- A CPM of each available camera in the current setup.

The teach step performs the mapping process using a single camera at a time. For each stereo image frame, the *Feature Detector* module extracts features $\mathcal{Z}$, which are then matched against the current map by the *Data Association* module. We use the quadruped's proprioceptive state estimate as a motion prior which helps to guide feature matching.

The *Metric Localization* module uses the matches to perform a registration against the map points $\mathcal{P}$ in the last created keyframe: we first use Perspective-n-Points (PnP) to obtain an initial estimate, which is later refined via pose-only optimization using the reprojection residual $\mathbf{r}_{\text{reproj}}(\mathbf{z}, \mathbf{p})$ with covariance $\mathbf{\Sigma}_{\text{reproj}}$, and a pose prior residual $\mathbf{r}_{\text{prior}}$ given by the previous estimate $\mathbf{T}_{\text{prior}}$ with covariance $\mathbf{\Sigma}_{\text{prior}}$:

$$\arg\min \sum_{\mathbf{z}\in\mathcal{Z},\mathbf{p}\in\mathcal{P}} \|\mathbf{r}_{\text{reproj}}(\mathbf{z},\mathbf{p})\|^2_{+} \|\mathbf{r}_{\text{prior}}(\mathbf{T}_{\text{prior}})\|^2_{\mathbf{\Sigma}_{\text{prior}}} \quad (1)$$

From this optimization we recover an estimate of the covariance $\mathbf{\Sigma}_{\text{visual}}$ of the optimization solution $\mathbf{T}_{\text{visual}}$, given by the Fisher information matrix [26]. Further, the covariance is used to compute the negative entropy E:

$$E = -\log(|\mathbf{\Sigma}_{\text{visual}}|) \quad (2)$$

E is a scalar that characterizes the performance of the localization at a given pose: a larger negative entropy implies a better localization estimate, and vice-versa. An advantage of this method over other criteria, such as the number of tracked features, is that it characterizes the whole localization process. For instance, tracking a small number of nearby features or a large number of distant features are treated similarly, because both situations lead to poor localization estimates. E is of particular importance for our system since:

- It is used by the *Map Manager* module as a criterion when creating new keyframes, using the running average filter strategy proposed by Kuo *et al.* [26].
- The poses and negative entropies of frames that are not used to create new keyframes are stored in the neighbor keyframes as *performance samples* $\mathcal{S}$ of the actual camera in a specific part of the path.

The process is executed for each camera individually, so as to sample their performance assuming no other cameras are available. This could generate inconsistent trajectories for each camera, so we enforce smoothness along the path by prioritizing the use of the proprioceptive state estimate for the teach step. The output is shown in Fig. 4 (a).

C. Learning Performance Models for Each Camera

As previously described, the teach step generates not only a topo-metric map, but also a set of performance samples $\mathcal{S}$ from the whole path. A performance sample s is defined as a tuple $(\mathbf{T}_s, E_s, c_s)$, where $\mathbf{T}_s$ is the pose of the sample expressed in the map frame, E_s is the negative entropy computed for that specific pose, and c_s is a tag to identify the particular camera that generated that sample.

We use the samples to learn a CPM, which is a model that embeds the performance of a camera for a given teach path. The CPM for a single camera c is expressed by a collection of Gaussian distributions defined for every keyframe in the path. Their parameters (μ_c, σ_c) are the result of the learning process. We define CPMs for all the cameras available in the robot, and we are able to query them at each keyframe of the path to determine which camera is likely to provide the best localization estimate.

The learning algorithm, defined in Alg. 1, performs a spatially weighted averaging of samples around each keyframe. The averaging weights are computed using a *radial basis function* kernel [27] denoted by $\kappa(\mathbf{T}_1, \mathbf{T}_2)$ and defined as:

$$\kappa(\mathbf{T}_1, \mathbf{T}_2) = \exp\left(-\frac{d(\mathbf{T}_1, \mathbf{T}_2)^2}{2l}\right) \quad (3)$$

where, $d(\cdot,\cdot)$ is a function that computes the distance between the two poses $\mathbf{T}_1$ and $\mathbf{T}_2$, ignoring the rotational component, while l is a hyperparameter (scale length) that controls smoothness. Fig. 4 (b) shows the output of the learning process for all the keyframes along a given path. This is similar to Gaussian Process regression, but is defined on the discrete space of keyframes, independently from spatial coordinates.

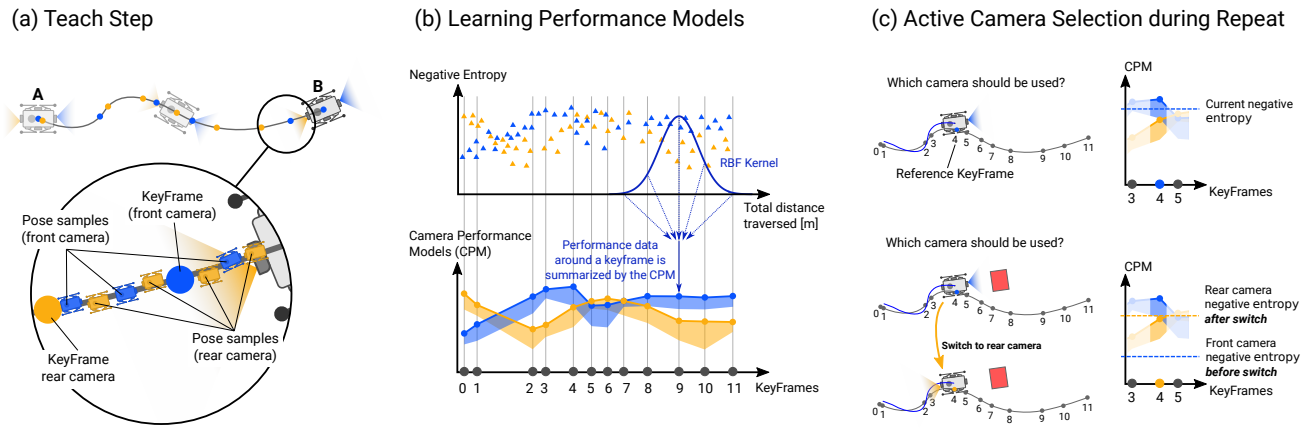

Fig. 4: Main steps of our VT&R system. **(a) Teach step:** The robot is first teleoperated from A to B to build a topo-metric map of the environment. Keyframes are created for each camera, denoted by the different colors. Along with the map, sampled poses and negative entropies are also computed and stored. **(b) Learning Performance Models:** For each keyframe, the closest negative entropy samples within a radius $d_{\max}$ are grouped and are averaged using an RBF kernel to learn a model of performance for each camera (Alg. 1). (c) **Repeat step with active camera selection:** The learned CPMs are used to select the camera with highest predicted performance at each segment of the map, or to change the selected camera if the predicted performance is not as expected.

Algorithm 1: CPM learning using teach path

Input: Keyframes $\mathcal{K}$, performance samples $\mathcal{S}$, maximum distance for sample search $d_{\max}$, kernel weighting function $\kappa(\cdot,\cdot)$
Output: CPM for each camera
foreach *keyframe* k *in* $\mathcal{K}$ **do**
  $\mathbf{T}_k \leftarrow$ GetKeyFramePose(k)
  $\mathcal{S}* \leftarrow$ SearchSamplesWithinRadius(k, $\mathcal{S}$, $d_{\max}$)
  foreach *camera* c *in* $\mathcal{S}*$ **do**
    $$w_c \leftarrow \sum_{s \in \mathcal{S}*} \kappa(\mathbf{T}_s, \mathbf{T}_k)$$
    $$\mu_c \leftarrow \frac{1}{w_c} \sum_{s \in \mathcal{S}*} E_s \; \kappa(\mathbf{T}_s, \mathbf{T}_k)$$
    $$\sigma_c \leftarrow \sqrt{\frac{1}{w_c} \sum_{s \in \mathcal{S}*} (E_s - \mu_c)^2 \; \kappa(\mathbf{T}_s, \mathbf{T}_k)}$$
    UpdateCPM(k, c, μ_c, σ_c)

D. Repeat Step with Active Camera Selection

The repeat step involves different procedures depending on the status of the system:

a) Global Relocalization: If the system is initializing, the status is *lost*, and an arbitrary camera will be chosen to attempt relocalization. First, the Place Recognition module searches candidate keyframes using Bag-of-Words. Then, the Data Association module performs a standard descriptor matching. The matches are later verified by the Metric Localization module by attempting a PnP registration and then using optimization refinement to discard outliers. The keyframe with the most matches is selected as the reference keyframe and the system status is set to *localized*. Since the previous procedure is agnostic to which camera generated the keyframe, we also compute the reference keyframes for the other available cameras. This is done by checking all the neighbor keyframes and associating to each camera the closest keyframe with the most similar orientation.

b) Path Traversal with Active Camera Selection: Once the system has computed the reference keyframes and an initial pose has been estimated, the repeat step is ready for execution. Details regarding the integration with the quadruped's controller are described later in Sec. III-E.

While the robot traverses the path and the Feature Detection, Data Association, and Metric Localization are being executed for the teach step, the Camera Manager module analyzes the different image streams and *actively* changes the current camera if: 1) there is another camera that can provide better performance at that specific part of the path, or 2) the current performance is not as the model describes, typically due to a change in the environment.

The first case only requires to query each μ_c in the CPM to find the best camera for the current reference keyframe. The second case requires comparison between the current negative entropy E_t and a lower bound $E_t < \mu_c - k\sigma_c$ computed from the CPM parameters (μ_c, σ_c) associated to the reference keyframe. The bound defines a *margin* so as to not select a new camera unless performance has decreased significantly, which can be tuned with the hyperparameter k.

In general, with accurate path tracking and with similar visual conditions in the teach and repeat steps, the previous inequality will never be satisfied, and the negative entropy will stay within the limits. However, if the environment changes, the feature extraction will be affected, potentially degrading the visual pose estimate and decreasing the negative entropy below the lower bound. When this occurs, the camera is flagged and cannot be used for a fixed time. The remaining CPMs are queried to find the next best camera for the given path section. An example of this procedure is illustrated in Fig. 4 (c). If all the cameras are flagged and the system loses visual tracking, it reports *Tracking Lost*.

c) Tracking Lost: When the system loses visual tracking, we use the last successful localization estimate and predict the current pose by using relative motion estimates from the legged proprioceptive state estimator. Meanwhile, the system will attempt a re-localization by matching against the neighbor keyframes within a certain radius, a procedure we call *Local Relocalization*. If the system cannot succeed after 10 seconds, it declares itself *lost* and will stop navigating until it is reset by the user.

E. Closed-loop Integration in a Legged Robot

The motion of a wheeled robot or a car is smooth and without any sharp jerks. A state-of-the-art VT&R system can track precisely enough to keep a UGV within the tram-lines of previous runs (as in [2]). This degree of smoothness has a stabilizing effect on VT&R localization. In contrast, the same degree of smoothness in camera motion is not possible on legged robots. The robot's gait induces a sharp, jerky motion, so that exact teach trajectory tracking is impossible.

The quadruped's Whole Body Controller (WBC) controls its 12 joints to achieve goals such as a desired base velocity. We interface with it through a High Level Motion Controller (HLMC), which computes a base velocity reference given a desired base pose. The VT&R system generates a sequence of waypoints from the teach path expressed in the map frame. Given the current localization estimate, the VT&R system selects the closest waypoint to the robot and sends it as the next desired base pose reference to the HLMC. In this way, we circumvent the need to precisely replicate the base motions of original teach trajectory, but we keep the robot close to it at all times.

Finally, in contrast to wheeled platforms, legged robots are holonomic and can strafe or turn in place to execute inspection tasks; we illustrate how the VT&R handles such situations in our attached video.

IV. Experimental Results

Experiments were performed with an ANYmal B300 robot equipped with two unsynchronized Intel RealSense D435i stereo cameras angled down by 12°; we used the IR stereo pairs as the visual input for our system. The robot also carries a Velodyne VLP-16 LiDAR, which we used in post-processing to obtain 10 Hz ground truth by registering scans within a prior map using Iterative Closest Point (ICP) [28]. Our system ran onboard on a single Intel i5 CPU shared along with other required modules and drivers.

For evaluation, we compared repeat trajectories to the initial teach run. We determined the instantaneous tracking error as the perpendicular distance between each pose during the repeat run and a line fit to the nearest neighbor points of the teach step path. The mean *Path Tracking Error* (PTE) is a measure of tracking performance for a full run.

Our VT&R system was tested in two experimental scenarios: an indoor workshop (E1) and a larger industrial environment outdoor (E2). Tab. I summarizes the path tracking performance for all the runs using the VT&R pose estimates. Finally, we ran our system in simulation with 4 cameras to demonstrate our approach with more complex camera setups.

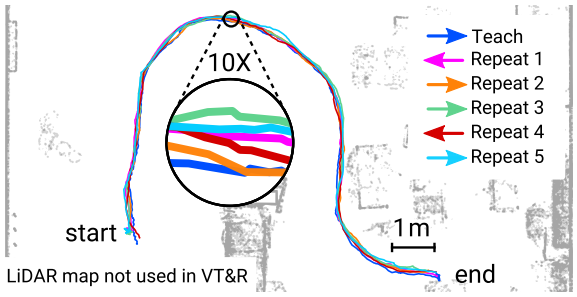

Fig. 5: Experiment 1 (Indoor): Ground truth trajectories of teach (blue) and repeat runs. Direction of motion is indicated by the legend. Overall, the robot never exceeded 20 cm of path tracking error, with an average of 7 cm.

TABLE I: Quantitative results for the indoor (E1, 5 repeats) and outdoor experiments (E2, 4 repeats).

	Path Tracking Error (PTE) $\mu \pm \sigma$ [m]				
	R1	**R2**	**R3**	**R4**	**R5**
E1	0.05 ± 0.03	0.06 ± 0.03	0.09 ± 0.06	0.06 ± 0.03	0.09 ± 0.04
E2	0.09 ± 0.04	0.13 ± 0.05	0.07 ± 0.05	0.14 ± 0.07	-

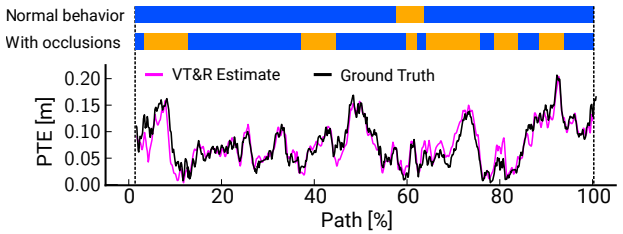

Fig. 6: Indoor experiment (E1): Estimated (magenta) and ground truth (black) PTE between teach and repeat R5. Color bars show the switch between the front camera (blue) and the rear camera (orange) in normal operation (top) and when occluded (bottom).

A. Experiment 1: Indoor

We first performed a series of experiments in a cluttered lab environment. The robot was teleoperated to walk between furniture, machines and other equipment, covering a distance of 15 m. It then autonomously returned to the initial position (backwards), and repeated the path back and forth 5 times.

The robot demonstrated stable navigation in all these runs, and it was able to stay within 20 cm of the teach path at all times, regardless of the walking direction (Fig. 5). Numerical comparisons in Tab. I, demonstrate the low tracking error obtained for each run.

In Fig. 6, we evaluated the tracking performance (as estimated by VT&R system online) by comparing it to the true tracking error (computed using the LiDAR ground truth) for the fifth repeat run of these experiments. The high degrees of correlation between the two estimates demonstrates that the VT&R system can accurately localize the robot against the teach path even if one of the cameras is occluded. Deviations are due to the shape of the teach path and the responsiveness of the tracking controller.

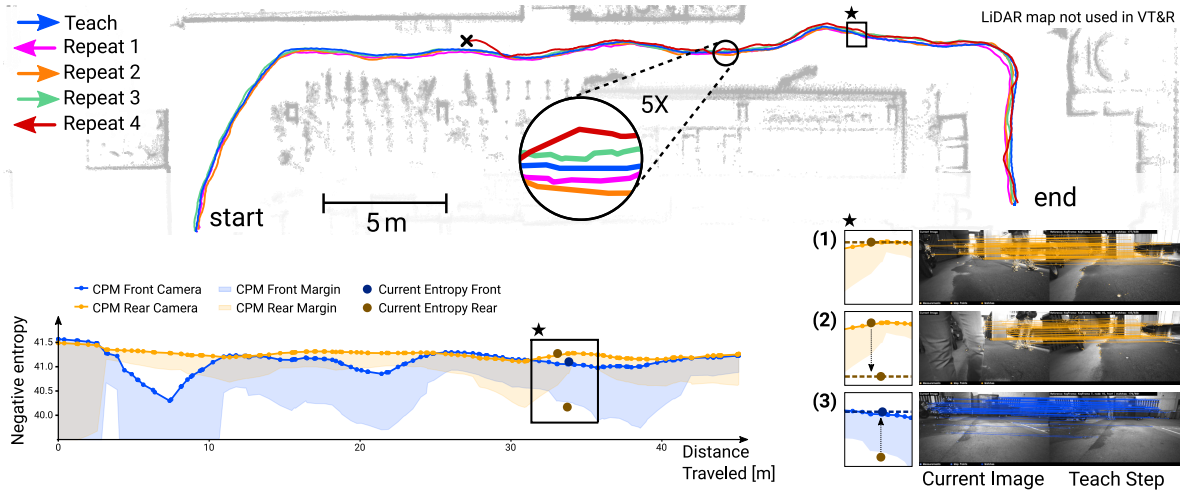

Fig. 7: Outdoor experiment (E2): *Top:* Ground truth trajectories of the teach (blue) and repeat runs. The robot never exceeded 20 cm of tracking error, with an average of 11 cm. *Bottom:* Left plot illustrates the CPM computed for the whole path, right images are examples of matches between the live stream and the teach path. A segment of the trajectory during Repeat 2 in which the camera is occluded is marked with a ★ symbol: (1) Rear camera performance (in orange) is within the CPM limits. (2) The camera is occluded, leading to a drop in the negative entropy, triggering a camera switch. (3) After the system switched to the front camera, its current negative entropy (blue) is closer to the CPM prediction.

B. Experiment 2: Outdoor

For our second experiment, we tested our system in an outdoor environment with adverse lighting and repetitive, industrial structure. We teleoperated the robot to walk a 45 meter long path, which was successfully traversed 3 times in repeat mode (Fig. 7). On a fourth repeat run the system was interrupted after the localization module diverged due to poor visual feature tracking. It was caused by changing lighting conditions, which is subject to future work.

During the second repeat run we occluded the cameras by having a person walking in front of the robot. Our VT&R successfully changed the active camera and completed the mission regardless. If the robot had used a single camera (i.e, no active selection available), such situations could have severely affected the visual tracking, degrading the performance or even failing to finish the mission in case of prolonged occlusions.

C. Experiment 3: Qualitative Experiments with 4 Cameras

Lastly, we performed experiments in simulation by equipping the ANYmal with 4 cameras. The goal was to demonstrate that our approach naturally generalizes to other camera configurations and is applicable to the latest version of ANYmal, the C-series, which has a similar 4 camera configuration. For the teach step, we made the robot walk through the environment with motion in all directions (forward, sideways and turning). Fig. 8 shows an example of the trajectories traversed in the simulation. Further experiments with the real robot will be a focus of future work.

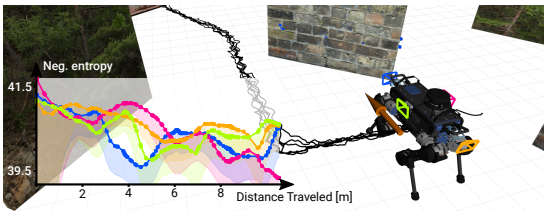

Fig. 8: Experiment 3 (Simulation): Tracking performance and CPM with 4 cameras. Black lines denote the ground truth paths for 6 consecutive repeats. The orange arrow is the next waypoint.

V. Conclusions

We presented a novel VT&R system that utilizes multiple non-synchronized cameras to perform autonomous point-to-point navigation. By exploiting information collected during a teach step, we learned a performance model for each camera that preserved the topo-metric structure. We demonstrated how the system utilized the learned models online to actively select the most informative camera and be resilient to sudden changes in the environment due to occlusions.

In a series of real and simulated navigation scenarios on a quadruped robot, our system successfully followed a previously taught route in spite of the complexities of jerky motion and people (intentionally) occluding the cameras.

In future, we plan to extend our VT&R system with other visual cues to improve its performance in more complex locomotion regimes such as stair climbing and obstacle traversals which cause the visual scene to change more dramatically.

Acknowledgement

This research is supported by the ESPRC/UKRI ORCA Robotics Hub (EP/R026173/1), a Royal Society University Research Fellowship (Fallon) and the National Agency for Research and Development of Chile (ANID) / Scholarship Program / DOCTORADO BECAS CHILE / 2019-72200291 (Mattamala).

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[2] 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.

[3] M. Hutter, C. Gehring, A. Lauber, F. Gunther, C. D. Bellicoso, V. Tsounis, P. Fankhauser, R. Diethelm, S. Bachmann, M. Bloesch, H. Kolvenbach, M. Bjelonic, L. Isler, and K. Meyer, "ANYmal - toward legged robots for harsh environments," *Advanced Robotics*, vol. 31, no. 17, pp. 918–931, 2017.

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

[5] P. Liu, M. Geppert, L. Heng, T. Sattler, A. Geiger, and M. Pollefeys, "Towards Robust Visual Odometry with a Multi-Camera System," *IEEE International Conference on Intelligent Robots and Systems*, pp. 1154–1161, 2018.

[6] C. Won, H. Seok, Z. Cui, M. Pollefeys, and J. Lim, "OmniSLAM: Omnidirectional Localization and Dense Mapping for Wide-baseline Multi-camera Systems," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[7] P. Furgale and T. Barfoot, "Stereo mapping and localization for long-range path following on rough terrain," in *Proceedings - IEEE International Conference on Robotics and Automation*, 2010, pp. 4410–4416.

[8] T. Krajnik, F. Majer, L. Halodova, and T. Vintr, "Navigation without localisation: Reliable teach and repeat based on the convergence theorem," in *IEEE International Conference on Intelligent Robots and Systems*, 2018, pp. 1657–1664.

[9] F. Gao, L. Wang, B. Zhou, X. Zhou, J. Pan, and S. Shen, "Teach-Repeat-Replan: A Complete and Robust System for Aggressive Flight in Complex Environments," *IEEE Transactions on Robotics*, pp. 1–20, 2020.

[10] M. Warren, M. Greeff, B. Patel, J. Collier, A. P. Schoellig, and T. D. Barfoot, "There's no place like home: Visual teach and repeat for emergency return of multirotor UAVs during GPS failure," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 161–168, 2019.

[11] M. Nitsche, F. Pessacg, and J. Civera, "Visual-inertial teach and repeat," *Robotics and Autonomous Systems*, vol. 131, p. 103577, 2020.

[12] W. Churchill and P. Newman, "Practice makes perfect? Managing and leveraging visual experiences for lifelong navigation," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 4525–4532, 2012.

[13] M. Paton, K. Mactavish, M. Warren, and T. D. Barfoot, "Bridging the appearance gap: Multi-experience localization for long-term visual teach and repeat," *IEEE International Conference on Intelligent Robots and Systems*, vol. 2016-Novem, pp. 1918–1925, 2016.

[14] C. Linegar, W. Churchill, and P. Newman, "Made to measure: Bespoke landmarks for 24-hour, all-weather localisation with a camera," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2016-June, pp. 787–794, 2016.

[15] M. Paton, F. Pomerleau, K. MacTavish, C. J. Ostafew, and T. D. Barfoot, "Expanding the Limits of Vision-based Localization for Long-term Route-following Autonomy," *Journal of Field Robotics*, vol. 34, no. 1, pp. 98–122, 2017.

[16] M. Paton, F. Pomerleau, and T. D. Barfoot, "Eyes in the Back of Your Head: Robust Visual Teach & Repeat Using Multiple Stereo Cameras," *Proceedings -2015 12th Conference on Computer and Robot Vision, CRV 2015*, no. June, pp. 46–53, 2015.

[17] W. Churchill, C. H. Tong, C. Gurău, I. Posner, and P. Newman, "Know your limits: Embedding localiser performance models in teach and repeat maps," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, pp. 4238–4244, 2015.

[18] P. Ondrúška, C. Gurău, L. Marchegiani, C. H. Tong, and I. Posner, "Scheduled perception for energy-efficient path following," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2015-June, no. June, pp. 4799–4806, 2015.

[19] M. Warren, A. P. Schoellig, and T. D. Barfoot, "Level-Headed: Evaluating Gimbal-Stabilised Visual Teach and Repeat for Improved Localisation Performance," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 7239–7246, 2018.

[20] Z. Zhang and D. Scaramuzza, "Perception-aware receding horizon navigation for MAVs," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2534–2541, 2018.

[21] Z. Zhang and D. Scaramuzza, "Beyond point clouds: Fisher information field for active visual localization," in *IEEE International Conference on Robotics and Automation*, 2019, pp. 5986–5992.

[22] 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.

[23] W. Churchill and P. Newman, "Experience Based Navigation: Theory, Practice and Implementation," Ph.D. dissertation, University of Oxford, 2012. [Online]. Available: http://www.robots.ox.ac.uk/~mobile/Theses/WinstonThesis.pdf

[24] P. F. Alcantarilla, J. Nuevo, and A. Bartoli, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," *BMVC 2013 - Electronic Proceedings of the British Machine Vision Conference 2013*, 2013.

[25] D. Gálvez-López and J. D. Tardós, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, oct 2012.

[26] J. Kuo, M. Muglikar, Z. Zhang, and D. Scaramuzza, "Redesigning SLAM for Arbitrary Multi-Camera Systems," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[27] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer Science+Business Media Inc., 2006.

[28] P. Besl and N. D. McKay, "A method for registration of 3-D shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 239–256, feb 1992.

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 Camera Performance Models for Active Multi-Camera Visual Teach and Repeat
Publication Status	x Published □ Accepted for Publication □ Submitted for Publication □ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Matias Mattamala, Milad Ramezani, Marco Camurri, Maurice Fallon, "Learning Camera Performance Models for Active Multi-Camera Visual Teach and Repeat," *International Conference on Robotics and Automation*, 2021.

Student Confirmation

Student Name:	Matias Mattamala		
Contribution to the Paper	• Contributed to development of the idea • Wrote the complete visual teach and repeat pipeline • Executed closed-loop field experiments • Performed data analysis and processing • Wrote large portions of the paper		
Signature 		Date	September 8, 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 Matias was the lead investigator, implementer and researcher on this project and wrote the bulk of the paper		
Signature 	Date	September 8, 2023

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

4.2 Discussion

This chapter presented an active strategy to exploit different cameras available on a legged robot, which does not require them to be hardware-synchronised. Our system relied on a topo-metric representation of the desired path, which we observed it was sufficient to re-traverse the path autonomously in a VT&R fashion.

For localisation, we adopted an optimisation framework that relied on the basic principles of LS optimisation that we reviewed in Chapter 2. While the implementation was inspired by filtering methods by fusing exteroceptive observations with a state prior, in practice we did not use the uncertainty estimate from the previous state and instead set a fixed covariance for that term instead. This was required to avoid a monotonically decreasing negative entropy of the pose estimate during the teach phase, which would have led to undesired behaviour in the CPMs formulation. While we demonstrated that the approach produced sensible results in practice, the estimator is inherently inconsistent, which is a potential avenue for future research in this space.

Regarding the decision-making aspects of the system, it is worth mentioning that the proposed approach aligns with ideas of the *active perception* literature [19]. Such methods aim to improve *perception by action*—localisation in this case. The most relevant recent example is the approach by Zhang and Scaramuzza [310], *Fisher information fields*, who performed localisation-aware path planning by building a representation that encoded the localisation quality. Our work was inspired by their basic principles but applying it to a topological map instead, which simplified the decision-making process to choosing a camera conditioned on the current node. Potential extensions of this work could relax the constraints on the robot orientation—which was asummed to be aligned with the reference path—, and extend the CPMs to characterise the localisation performance at different orientations or even spatial neighborhoods, effectively expressing a *localisation performance basin* around the reference node.

A recent paper by Hausler et al. [118] presented a similar system for camera selection but in the context of autonomous driving. As the paths they dealt with

are kilometre-long, they split them into segments of ∼40 m long to discretise the decision space. Each camera's performance is also measured by metric localisation errors (translation error in their case) and used to train kernel density estimators of the performance similar to our CPMs. While their evaluation was focused on localisation performance, their conclusions are aligned with our own findings in the context of VT&R navigation: switching cameras enables robust localisation when the camera processing budget is limited.

Lastly, this work did not exploit the full capabilities and characteristics of legged platforms apart from their sensor setup and omni-directional motion capabilities. However, it allowed us to develop an initial navigation pipeline to study legged-specific challenges, which I further developed in the following chapters.

5
Reactive Local Planning

The localisation strategy presented in Chapter 4 addressed important challenges to enable visual navigation in the environment in spite of transient changes in the scene. However, if the prioritised camera was occluded by a passer-by, or if the robot approached a wall while passing through a narrow space, the system would only try to stay localised but it would not actively avoid a collision.

This chapter presents an approach to address this problem by introducing local awareness. Instead of designing a platform-specific perceptive locomotion controller or a sampling-based local planning approach, we relied on the topo-metric representation for coarse guidance. We exploited a local metric representation built from depth cameras to generate vector field representations that encoded planning information—Geodesic Distance Field (GDF)—and obstacle information—Signed Distance Field (SDF)—used to generate collision-free motion.

The vector fields were combined in a principled manner using the RMPs framework, exploiting the geometric interpretation of LS optimisation. This reactive formulation allowed the local planner to handle noise and partial measurements in the local representation while also being applicable to locomotion controllers with different dynamics. When deployed, it enabled safe navigation not only when the reference path was partly occluded but also in challenging underground environments with narrow doorways and tight corridors.

5.1 An Efficient Locally Reactive Controller for Safe Navigation in Visual Teach and Repeat Missions

The following article was published in the *IEEE Robotics and Automation Letters (RA-L)* and it was also presented at the *IEEE International Conference on Robotics and Automation (ICRA)* 2022 [189]. An accompanying video is available online at: https://youtu.be/G_AwNec5AwU.

An Efficient Locally Reactive Controller for Safe Navigation in Visual Teach and Repeat Missions

Matias Mattamala[1], Nived Chebrolu[1], and Maurice Fallon[1]

***Abstract*—To achieve successful field autonomy, mobile robots need to freely adapt to changes in their environment. Visual navigation systems such as Visual Teach and Repeat (VT&R) often assume the space around the reference trajectory is free, but if the environment is obstructed path tracking can fail or the robot could collide with a previously unseen obstacle. In this work, we present a locally reactive controller for a VT&R system that allows a robot to navigate safely despite physical changes to the environment. Our controller uses a local elevation map to compute vector representations and outputs twist commands for navigation at 10 Hz. They are combined in a Riemannian Motion Policies (RMP) controller that requires < 2 ms to run on a CPU. We integrated our controller with a VT&R system onboard an ANYmal C robot and tested it in indoor cluttered spaces and a large-scale underground mine. We demonstrate that our locally reactive controller keeps the robot safe when physical occlusions or loss of visual tracking occur such as when walking close to walls, crossing doorways, or traversing narrow corridors.**

***Index Terms*—Vision-Based Navigation; Legged Robots; Sensor-based Control**

I. Introduction

SAFE navigation is a fundamental capability required to successfully deploy robots in natural and human-made environments. Forests, mines, and industrial facilities are challenging places due to clutter, narrow passages, or moving objects that effect the configuration space in which a robot operates. Dealing with such difficulties requires a sense of local awareness, in order to effectively adapt the robot's behavior to the task and state of the environment.

Visual Teach and Repeat (VT&R) systems are a practical approach to enable inspections or patrols of known places, or to navigate from point-to-point [1], [2]. Instead of building a precise metric map, VT&R relies on a *topo-metric* representation built during a *teach* phase, which is then used during the *repeat* phase to execute point-to-point missions using only camera input to track the reference path. However, in practice, the visual appearance of real environments can change (due to the lighting conditions or because of onboard illumination [3]) as can the physical layout (such as when a pathway is blocked), risking the success of the mission.

Manuscript received: September, 9, 2021; Revised November, 25, 2021; Accepted December, 29, 2021.

This paper was recommended for publication by Editor Eric Marchand upon evaluation of the Associate Editor and Reviewers' comments. This work is supported by the ESPRC/UKRI ORCA Robotics Hub (EP/R026173/1), a Royal Society University Research Fellowship (Fallon), and the ANID / Scholarship Program / DOCTORADO BECAS CHILE / 2019-72200291 (Mattamala). This research has been conducted as part of the ANYmal research community.

[1]The authors are with the Oxford Robotics Institute at the University of Oxford, UK. {matias, nived, mfallon}@robots.ox.ac.uk.

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

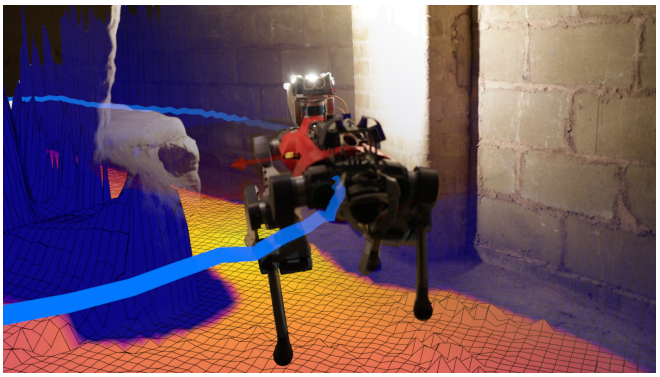

Fig. 1: Our locally reactive controller allows an ANYmal robot to safely execute Visual Teach and Repeat navigation through narrow corridors in a decommissioned underground mine. We use the local map built on-the-fly to compute efficient vector field representations, such as the *geodesic distance field* pictured, and combine them in a Riemannian Motion Policies controller that runs at 10 Hz.
Video: `https://youtu.be/G_AwNec5AwU`

In this paper, we present a locally reactive controller that works as a safety layer to help a VT&R system achieve robust operation after significant changes to the environment. It allows a robot to be aware of its surroundings using a local map built online. We argue that for VT&R missions, we do not need to rely on a global and a local planner to navigate these environments. Instead, we exploit the fact that the taught path effectively represents a *global plan in free space*, only requiring a fast, reactive controller to track the path safely, which is the main focus of this work.

Our controller uses the local map to generate efficient vector fields that provide the cues necessary for navigation, such as avoiding obstacles or reaching (local) goals around corners. The fields are combined in a principled manner using a Riemannian Motion Policy (RMP) formulation [4], which allows us to dynamically enable specific behaviors depending on the robot's state. We demonstrate its closed-loop integration with a VT&R system, showing safe and fast navigation in cluttered indoor spaces and challenging large scale environments.

The main contributions of our work are the following:

- A reactive RMP-based twist controller that uses a local elevation map built online as the main perceptual input.
- Online computation of vector fields representations computed from the local map at 10 Hz on a CPU.
- Closed-loop integration with our VT&R system [5] on the ANYmal platform.
- Real-world validation using the ANYmal C quadruped in realistic environments including scenarios with substantial

change to the scene and narrow pathways.
- First demonstration of a RMP reactive controller with a full-sized dynamic robot in large-scale environments.

II. Related Work

A. Collision-free Navigation

The navigation problem aims to determine an action (or set of actions) that allows a robot to reach a specific goal without colliding with obstacles. This is typically achieved through *reactive controllers* or *local planners* [6]. Reactive controllers leverage the information available at a specific time and determine a single action to be directly executed by the robot. In contrast, local planners typically generate a collision-free trajectory by considering a short look-ahead and solving an optimization problem over that region.

For legged robots, determining the optimal action or trajectory can involve the computation of footsteps, which is known as *whole-body control/planning*. Buchanan *et al.* [7] combine the whole-body planning problem with obstacle information in the form of a 3D signed distance field that is used for collision checking. This allows the robot to navigate through narrow spaces while planning individual footsteps through traversable regions. Similarly, Gaertner *et al.* [8] demonstrate collision-free locomotion by means of a Model Predictive Controller (MPC). These approaches couple the collision-free navigation task with footstep computation, thereby making them harder to generalize to different platforms.

In this work, we use a twist interface to control the legged robot's body velocity. It has been successfully demonstrated on several platforms such as ANYmal C [9], Mini Cheetah [10], and Vision60 [11]. Our controller is based on the Riemannian Motion Policies (RMP) framework [4], a recent approach to robot motion generation. It allows us to generate reactive behaviors by combining different *tasks* –such as obstacle avoidance or heading corrections– depending on the robot's state. By formulating the problem as reactive twist-based control, we can easily make this local safety layer robot and gait independent.

B. Representations for Local Navigation

Classical navigation approaches typically consider only the geometric information of the environment, which can be either represented by occupancy maps [12], voxels [13], point clouds [14], meshes [15] or height maps [16]. Typically, a traversibility map is computed from these representations giving a measure of how safe a region is for robot navigation.

Semantics can often help to expand the notion of traversable space. Bradley *et al.* [13] learn four types of terrain used for navigation in a forest environment with the LS3 quadruped. Nardi and Stachniss [17] as well as Wellhausen *et al.* [18] learn terrain properties in a self-supervised fashion which are then used as costs for a path planning algorithm that enables them to navigate safely.

Representations such as Signed Distance Fields (SDFs) [19] are particularly suitable for navigation, since they encode the distance to the obstacle surfaces in the environment. The SDFs are also differentiable and lend themselves to computation of a vector field, which can be seamlessly integrated in optimization-based planners or vector-based controllers. Another useful representation are the Geodesic Distance Fields (GDF) which capture the shortest path to goal from different positions in the map [20]. These fields can be computed efficiently using the Fast Marching Method [21] or the Heat Method [22] and have been used in the past for navigation [23], [24]. In this work, we also take advantage of these representations and their vector fields to generate specific behaviors. However, we only compute them in the local map, which suffices for reactive control, efficiently exploiting the 2.5D geometry for online operation.

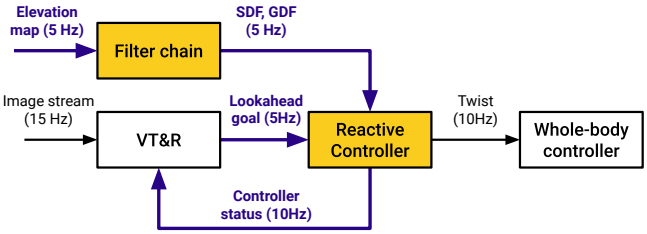

Fig. 2: Main pipeline of our safe VT&R system integrated with a quadruped robot. The *filter chain* module computes vector field representations from the elevation map (Sec. III-B). They are used by the *reactive controller* module (Sec. III-C) to generate the command sent to the whole-body controller.

C. Safe Navigation for VT&R

In the context of VT&R systems, it is usually assumed that the path is collision-free and only a few works have addressed safe navigation explicitly. Ostafew *et al.* [25] formulate the collision-free navigation problem as a non-linear MPC controller with constraint satisfaction. The constraints are given by potential obstacles or localization limits, i.e, forcing the robot to stay close to the reference path to avoid localization failures. Berczi and Barfoot [26] learn models of traversability over multiple runs, which are used to detect unexpected obstacles when compared to the reference experience.

Recently, Bista *et al.* [12] presented a topological navigation system for indoor environments which heuristically combined the output of a visual servoing system with the solution of a local plan computed in a 2D grid with A*. While we share some general ideas, the fundamental differences are that our VT&R system explicitly computes a 6 DOF pose –because legged robots effectively move in 3D space–, while it also presents a principled way to combine different desired behaviors within the same control framework.

III. Method

Our locally reactive controller and its integration with the VT&R system is summarized in Fig. 2. Our approach is a safety mechanism operating on the robot's local elevation map based on Riemannian Motion Policies (Sec. III-A). First, the *Filter chain* module (Sec. III-B) determines the traversability of the local map and computes efficient vector field representations, such as SDFs and GDFs. These fields are fed as input to

the *Reactive Controller* module, which implements the RMP controller (Sec. III-C). Finally, we describe the integration of the reactive controller with our VT&R system in Sec. III-D.

A. Preliminaries: Riemannian Motion Policies

We base our controller on the Riemannian Motion Policies framework [4], [27], which we briefly introduce. We consider a set of different desired behaviors or *tasks* specified by tuples $\{\mathbf{f}, \mathbf{M}\}$. Here $\mathbf{f} = \mathbf{f}(\mathbf{x}, \dot{\mathbf{x}})$ is an acceleration field or *motion policy* that describes the behavior, namely goal reaching, obstacle avoidance, or damping. $\mathbf{M} = \mathbf{M}(\mathbf{x}, \dot{\mathbf{x}})$, is a matrix known as *Riemannian metric*, which smoothly varies with the robot's state $\mathbf{x}$ and determines the importance of the field $\mathbf{f}$. The tuple is hence known as a *Riemannian Motion Policy* (RMP).

This framework allows us to combine RMPs $\{\mathbf{f}, \mathbf{M}\}$ from different sources to determine the optimal control $\ddot{\mathbf{x}}$ by minimizing:

$$\arg\min_{\ddot{\mathbf{x}}} \sum_{\{\mathbf{f}, \mathbf{M}\}} \|\mathbf{f} - \ddot{\mathbf{x}}\|^2_{\mathbf{M}}. \tag{1}$$

Since the optimization is a linear least-squares problem, it can be solved in closed form, and the desired acceleration is the weighted average of the RMPs. The optimal $\ddot{\mathbf{x}}$ is forward integrated to obtain the twist command.

While the framework is general and some of its theoretical advantages have been shown [27], designing the different RMPs depends on the robot's sensors and the environment. In this work we are interested in exploiting 2.5D elevation maps commonly used by legged and other ground robots to represent the local environment, and in generating efficient vector field representations from them to design our RMPs.

B. Efficient Vector Field Representations

Elevation maps are a popular representation of the local environment as they (1) preserve 2.5D information about the terrain, (2) can be easily computed from depth cameras or LiDARs via ray-casting [28], (3) they are now usually available in legged robots since they are used for locomotion tasks, and (4) they can be manipulated as images, making them suitable to efficient computer vision operations.

Thus, we assume that a local elevation map reconstructing a limited space around the robot is available. We base our implementation on the universal grid map library [16], which represents the local elevation map in a multi-layered fixed-sized grid. We use a 8 m $\times$ 8 m map with a resolution of 4 cm, giving a grid of 200×200 cells. To obtain the vector field representations, we follow the steps explained here:

1) Filling unknown areas: We first run an in-painting filter on the elevation map. Image-based in-painting methods fill the missing data using nearby pixels. We use the inpainting approach [29] implemented in OpenCV with a radius of 10 cm. Please refer to Sec. IV-B for a further discussion.

2) Traversability computation: We compute a continuous measure for terrain traversability using the slope and roughness of the local map [30]. We additionally take into account the height of terrain with respect to the body pose. This means that if the robot is inclined, e.g. on a hill, the height is measured normal to the surface and the sloped surface is treated as traversable. An example of the continuous traversability map is shown in Fig. 3 left. The continuous value is thresholded to produce a binary classification $t \in [0, 1]$, where 1 means fully traversable and 0 is not traversable.

3) Signed Distance Field (SDF): The SDF layer (Fig. 3 center) is computed using the 2D Euclidean distance transform [31]. Given the binarized traversability layer, we apply the distance transform to obtain the field f_{SDF}. To compute the normalized gradient ∇f_{SDF}, we use the Sobel edge-detector, which is further smoothed using image filters (Gaussian, median, etc). By relying on 2D image operations, we are able to compute the SDF online at 10 Hz, which is crucial to allow the controller to be responsive to new obstacles.

4) Geodesic Distance Field (GDF): Geodesic distance fields are potential functions with an unique global minimum. This is used to reach a goal from any point in a local map. Similar to the SDF, we use the Fast Marching Method (FMM) [21] on the binarized traversability layer to build the GDF layer. The FMM solves the Eikonal equation [32] from a seed point, from which a wavefront is propagated to fill the entire space. In our implementation, the seed is given by the goal position in the local map or, if the goal falls outside the map, the closest point in the map is chosen. The resulting field is shown in Fig. 3 right. After f_{GDF} is computed, we obtain normalized gradients using the Sobel operator, which define the *geodesic flow* ∇f_{GDF} to reach the goal.

C. Locally Reactive Controller

The reactive controller module in Fig. 2 implements the RMP controller presented in Eq. (1), using the vector fields obtained from the elevation map to create goal reaching and obstacle avoidance RMPs. Tab. I summarizes all the RMPs considered along the expressions for the fields and metrics. However, there are aspects to consider for their implementation and integration with the navigation pipeline:

1) State representation and reference frames: Our formulation considers the frame conventions illustrated in Fig. 4. The state $\mathbf{x}$ of the robot is given by the pose of the robot's body $\mathbf{T}_{\mathtt{IB}} \in \mathrm{SE}(2)$ expressed in the fixed, inertial frame `I`. This pose is given by our VT&R system or by other autonomy system. The robot's velocity $\dot{\mathbf{x}}$ and acceleration $\ddot{\mathbf{x}}$ are also expressed in the body frame `B` and will be denoted onward by ${}_{\mathtt{B}}\mathbf{v} = ({}_{\mathtt{B}}v_x, {}_{\mathtt{B}}v_y, {}_{\mathtt{B}}v_\theta) \in \mathbb{R}^3$ and ${}_{\mathtt{B}}\mathbf{a} = ({}_{\mathtt{B}}a_x, {}_{\mathtt{B}}a_y, {}_{\mathtt{B}}a_\theta) \in \mathbb{R}^3$.

The goal pose is expressed in the inertial frame $\mathbf{T}_{\mathtt{IG}} \in \mathrm{SE}(2)$. The local map is computed in the map frame `M` which has a translation offset of ${}_{\mathtt{B}}\mathbf{t}_{\mathtt{MB}} \in \mathbb{R}^2$ to coincide with the center of the robot body.

Note that the reference frame used by the motion policies are described in body frame `B`, whereas ∇f_{SDF} and ∇f_{GDF} are defined in the local map frame `M`. These gradients need to be properly transformed to `B` before being integrated in the optimization problem in Eq. (1). Please refer to the Appendix for technical details.

2) Collision spheres: To represent the robot's volume, we follow the same approach as previous works by using a set of

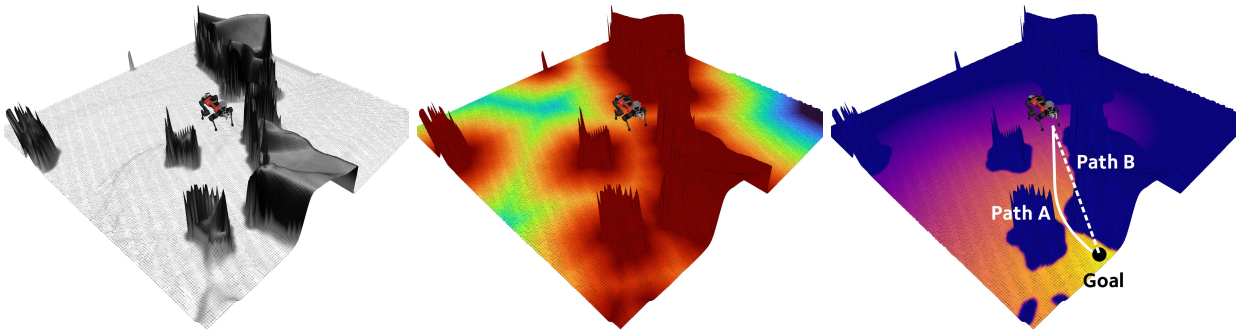

Fig. 3: Representations computed from the elevation map, as described in Sec. III-B. **Left:** Traversability layer computed from geometric analysis; light gray means traversable, whereas black is not traversable. **Center:** Signed Distance Field (SDF); color indicates the distance to obstacles from red (closer) to blue (further). **Right:** Geodesic Distance Field (GDF); the goal is shown in yellow, and further areas in dark blue. Using the GDF, path A has a smaller cost compared to B, which goes over the obstacles.

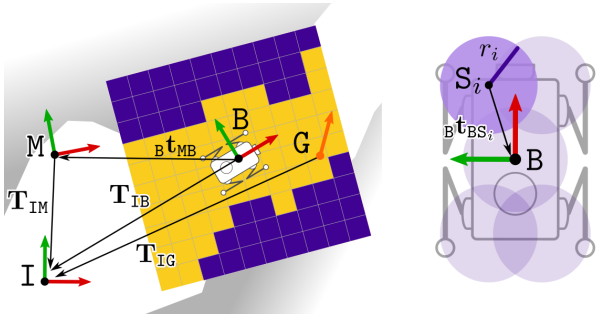

Fig. 4: Reference frames used in this work. **Left:** The pose of the robot's body $\mathtt{B}$ in the inertial frame $\mathtt{I}$ is given by $\mathbf{T}_{\mathtt{IB}} \in \mathrm{SE}(2)$. The local map, expressed in frame $\mathtt{B}$, represents a local representation of the environment. **Right:** The collision spheres $\mathcal{S}$ define the frame $\mathtt{S}_i$ and are used to represent the robot's volume, their positions are expressed in the body frame by the vector ${}_{\mathtt{B}}\mathbf{t}_{\mathtt{BS}_i}$.

collision spheres $\mathcal{S}$ [8], as shown in Fig. 4, right. Each sphere defines a reference frame $\mathtt{S}_i$, and it is parametrized by a fixed translation ${}_{\mathtt{B}}\mathbf{t}_{\mathtt{BS}_i}$ and a radius r_i.

Our final formulation computes the optimal acceleration of the body ${}_{B}\mathbf{a}$ given the RMPs summarized in Tab. I as the following optimization:

$$\arg\min_{{}_{\mathtt{B}}\mathbf{a}} \underbrace{\sum_{\{{}_{\mathtt{B}}\mathbf{f},\,\mathbf{M}\}} \|{}_{\mathtt{B}}\mathbf{f} - {}_{\mathtt{B}}\mathbf{a}\|^2_{\mathbf{M}}}_{\text{RMPs applied to body}} + \underbrace{\sum_{\mathcal{S}} \sum_{\{{}_{\mathtt{S}}\mathbf{f},\,\mathbf{M}\}} \|{}_{\mathtt{S}}\mathbf{f} - \mathbf{J}_{\mathtt{SB}}\,{}_{\mathtt{B}}\mathbf{a}\|^2_{\mathbf{M}}}_{\text{RMPs applied to collision spheres}}. \tag{2}$$

In Eq. (2), we expand the original formulation in Eq. (1) to explicitly state where each motion policy is being applied. This can be either the body frame $\mathtt{B}$ or the collision spheres $\mathcal{S}$. When applying RMPs to collision spheres, they only affect the center point, and therefore the accelerations only have translational components. Hence, the Jacobian –or *pullback*– $\mathbf{J}_{\mathtt{SB}}$ is required to map the effect of the desired acceleration of the body ${}_{\mathtt{B}}\mathbf{a}$ to the corresponding collision sphere.

The resulting acceleration ${}_{\mathtt{B}}\mathbf{a}$ is time-integrated at the controller rate (10 Hz) to obtain a body twist ${}_{\mathtt{B}}\mathbf{v}$ which is then passed to the whole-body controller (WBC). For real experiments we use a WBC, developed by collaborators and based on Lee *et al.* [33], that generates a valid walking motion given a desired velocity using a learned gait model, which is robust over uneven terrains. Our simulation experiments use the default model-based WBC from the manufacturer.

D. Integration with the VT&R System

For closed-loop integration with the VT&R system, we use the *carrot-on-a-stick* strategy, which sends a look-ahead goal at a distance d_{carrot} which ensures that the robot moves continuously along reference trajectory. Additionally, we implemented status feedback from the controller to the VT&R system, which not only reports when a goal is reached, but also if a goal is unreachable, e.g, when the path is blocked. As reported in previous work [5], our original VT&R system achieves less than 20 cm error, and 10 cm on average.

IV. Results

We performed experiments both in simulation and on the ANYmal C100 robot with an Intel i7 computer onboard. All the processing was CPU-based. The robot has been modified from the stock version by replacing the front and rear cameras with two RoboSense BPearl LiDARs. The data from the LiDARs was used to build local elevation maps using the approach of Fankhauser *et al.* [28]. Additionally, a front wide-angle stereo camera developed by Sevensense Robotics was installed on top and was used to feed the VT&R system.

The parameters in Tab. I were tuned using line search, and validated online in simulation and the real robot. We tested our system in a cluttered indoor space, as well as a large-scale underground mine. We evaluated the performance of our system both quantitatively and qualitatively in the two scenarios, as well as the computation time required. Simulation experiments were used to provide a qualitative study of the contribution of individual motion policies and compare against other baselines.

A. Study 1: Influence of each RMP

In this experiment, we provide a qualitative analysis performed in a simulated environment to understand the influence of the different RMPs considered. In Fig. 5, we show three examples which use different combinations of the RMPs. The robot was given a goal on the other side of the wall

TABLE I: Main Riemannian Motion Policies (RMP) used for the reactive controller formulation.

RMP	GDF-based Goal Reaching	Obstacle-free Goal Reaching	Obstacle Avoidance	Heading Correction	Damping	Regularization
Description	Uses the GDF to reach the goal	Minimizes the position and orientation error assuming free space	Uses the SDF to push the robot away from obstacles	Aligns the robot with the actual translation-only velocity vector	Damps the output acceleration depending on the current velocity	Smooths the output acceleration using the last acceleration command
Applied to	Body $\mathtt{B}$	Body $\mathtt{B}$	Collision sphere $\mathtt{S}_i$	Body $\mathtt{B}$	Body $\mathtt{B}$	Body $\mathtt{B}$
Acceleration field $\mathbf{f}$	$-k\,\nabla f_{\text{GDF}}({}_{\mathtt{M}}\mathbf{t}_{\mathtt{MB}})$	$k\,\text{Log}\left(\mathbf{T}_{\mathtt{IB}}^{-1}\mathbf{T}_{\mathtt{IG}}\right)^{\star}$	$-k\frac{\nabla f_{\text{SDF}}({}_{\mathtt{M}}\mathbf{t}_{\mathtt{MB}})}{(f_{\text{SDF}}({}_{\mathtt{M}}\mathbf{t}_{\mathtt{MB}})-r_i)}$	$k\,\text{atan2}({}_{\mathtt{B}}v_y, {}_{\mathtt{B}}v_x)$	$-k\,{}_{\mathtt{B}}\mathbf{v}$	${}_{\mathtt{B}}\mathbf{a}_{t-1}{}^{\dagger}$
Components affected	Only translation, ${}_{\mathtt{B}}f_\theta = 0$	Translation and orientation	Only translation, ${}_{\mathtt{S}}f_\theta = 0$	Only orientation, ${}_{\mathtt{B}}f_x = {}_{\mathtt{B}}f_y = 0$	Translation and orientation	Translation and orientation
Metric $\mathbf{M}$	$\sigma(d, d_c)\,\mathbf{I}_{3\times3}$ ‡	$\sigma(d, d_c)\,\mathbf{I}_{3\times3}$	$\sigma(d, d_c)\,\mathbf{I}_{2\times2}$	$\sigma(d, d_c)\,\mathbf{I}_{3\times3}$	$\sigma(d, d_c)\,\mathbf{I}_{3\times3}$	$s\,\mathbf{I}_{3\times3}$
When Enabled?	Distance to goal d is **further** than d_c	Distance to goal d is **closer** than d_c	Distance to obstacle d is **closer** than d_c	Distance to goal d is **further** than d_c	Always	Always

* $\text{Log}\,(\cdot)$ is the logarithm map of $\text{SE}(2)$ [34].
† ${}_{\mathtt{B}}\mathbf{a}_{t-1}$ corresponds to the acceleration computed in the previous iteration.
‡ The function $\sigma(d)$ is implemented as a logistic/inverse logistic function that becomes 1 when d is closer/farther than d_c depending on the case.

with the heading in the same direction as the starting pose. Fig. 5 (a) shows that by using only *Obstacle-free Goal Reaching* + *Obstacle Avoidance* RMPs, the solution falls into a local minimum as the goal was at the other side of the wall, though the robot kept a safe distance from the wall.

By using the *GDF-based Goal Reaching* and *Obstacle Avoidance* RMPs, the robot is able to go around the wall. However, the final orientation of the robot does not match the goal, having ≈90° error as seen in Fig. 5 (b), as the GDF-based RMP does not have an orientation component.

In Fig. 5 (c), the weighted combination of RMPs via distance-enabled metrics allows the robot to go around the obstacle by relying on the GDF when the goal is farther than a distance d_c, whereas it tries to minimize orientation error using the *Obstacle-free Goal Reaching* RMP within a distance d_c of the goal, where d_c = 1 m in our experiments.

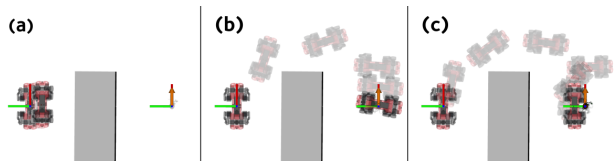

Fig. 5: Study 1: Influence of RMPs. **(a)** *Obstacle-free Goal Reaching + Obstacle Avoidance RMPs*. **(b)** *GDF-based Goal Reaching + Obstacle Avoidance RMPs*. **(c)** All the RMPs in Tab. I.

B. Study 2: Computation time

We measured the computation time of the filter chain and the controller. We ran our controller on the real robot to collect data for about 5 min during a typical VT&R mission. Fig. 6 shows the computation time across the whole experiment, obtaining an upper bound of 80 ms (≈12 Hz) on average. Most of the computation is spent in the in-painting operation, followed by the GDF computation which requires about 13 ms on average. Using coarser grid resolutions, smaller local maps, or simpler in-painting strategies could help to increase the operation frequency if required.

By comparison, the controller has minimum computational cost, requiring <2 ms on average. Setting up the RMPs and solving the optimization problem also has a negligible cost due to the simplicity of the approach. This could allow us to operate at higher frequency if we reuse the last elevation map processed by the filter chain.

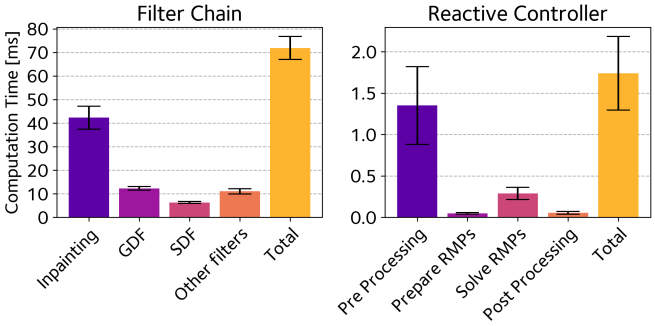

Fig. 6: Study 2: Computation time of the filter chain and reactive controller.

C. Experiment 1: Cluttered Indoor Space

This experiment was designed to demonstrate and evaluate the performance of our VT&R system integrated with the reactive controller in a cluttered indoor environment. We performed a teach step along a short trajectory (18 m) in our lab environment. We placed objects along the route to create an obstacle course to demonstrate (1) the performance of our reactive controller in terms of avoiding obstacles, and (2) the robustness of the overall system to deal with the degraded appearance of the environment introduced due to obstacles. We ran the system in repeat mode four times (twice walking forward and twice backward), moving the obstacles between each run. Fig. 7 illustrates some of the behaviors emerging after introducing obstacles to the reference path.

We computed the *path tracking error* (PTE) to evaluate the tracking performance of VT&R systems [5] shown in Fig. 8. Far from obstacles, the system performed similar to our previous approach [5], but the tracking performance decreased near them (indicated with the spikes in the errors, numbered ② and ③), which is consistent as the controller forces the robot to

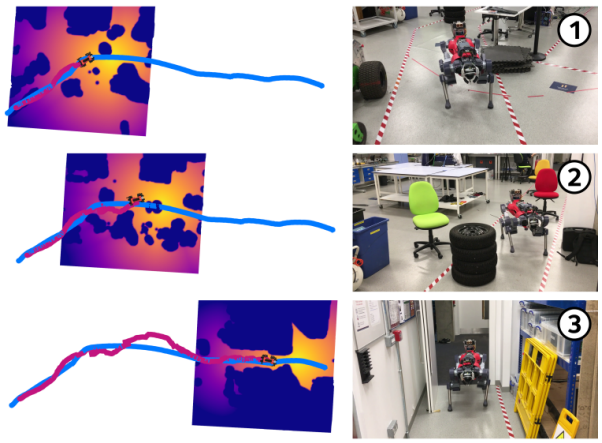

Fig. 7: Experiment 1: Indoor Space. Example of a repeat traversal. Blue ■ is the original teach path; magenta ■ the repeat traversals after introducing obstacles. **Left:** traversed path as well as the GDF computed online. **Right:** Footage from the actual traversal.

deviate from the reference path using the cues given by the GDF as well as the obstacle avoidance policies. In the first run (*Forward1*), we used a carrot distance $d_{\text{carrot}} = 0.7\,\text{m}$ resulting in an average speed of 0.16 m/s. In the other traversals, we used $d_{\text{carrot}} = 1.5\,\text{m}$ which allowed the robot to run faster at 0.31 m/s on average.

D. Experiment 2: Large Underground Mine

Our second experiment was executed in an unlit decommissioned underground mine in Wiltshire, UK. We teleoperated the robot along a 60 m long teach path which passed through narrow passages close to walls and large areas that challenged the reliability of stereo vision. The robot carried onboard lights and we enhanced the images using CLAHE equalization [35] to deal with the poor lighting conditions, as shown in Fig. 9 and the attached video.

In repeat mode, we had the robot traverse the path twice – forward as well as backward. Fig. 9 shows the resulting trajectory traversed by the robot (center), as well as some examples of visual tracking from the VT&R system (a-d on the top). We also illustrate some of the challenging areas that the robot traversed during the repeat phase, and highlight these areas along the trajectory (Fig. 9, ①–⑧ on the sides). The robot also safely navigated narrow corridors with a width of 60 cm with the opening being only slightly larger (< 30 cm) than the robot body's width, as shown in ⑥.

As for the previous experiment, we also show the PTE in Fig. 10. The robot achieved better tracking when traversing in the same direction (orange ■) than the reverse traversal (purple ■); this is consistent with previous studies of the optimal placement of cameras for visual odometry [36]. The most significant deviations with respect to the reference path were when rounding corners when walking backward, see locations b and d in Fig. 9 and Fig. 10. Here, visual tracking failed causing the system to fall on the propagated estimate using the robot's odometry until recovery. Despite the loss of visual tracking, the robot traversed in both runs without collision even in narrow corridors at 0.3 m/s on average.

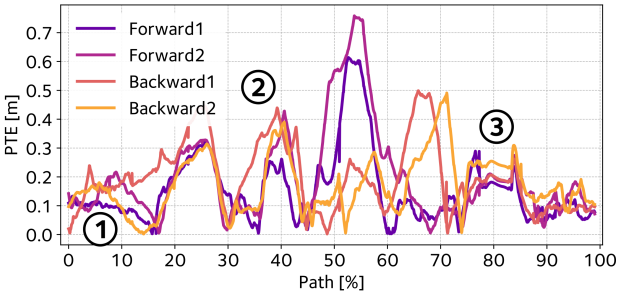

Fig. 8: Experiment 1: Path tracking error for the four different traversals in the indoor environment.

E. Experiment 3: Simulation Baselines

As a final experiment, we showed the advantages of our approach compared to other local planners/controllers in a simulated environment. We built the setup shown in Fig. 11, recorded a collision-free reference path in free space, and then we introduced artificial obstacles to force the robot to deviate from that path, similarly to Experiment 1. To avoid the overhead of running our full VT&R system, we only implemented the carrot-on-a-stick approach using the reference path to send look-ahead goals to the controllers. The baseline methods used were:

- *FALCO-based local planner*: We adapted the planner released by CMU Exploration[1], based on collision-checking and scoring of precomputed trajectories [37], and tracking the best-scored path with a P controller.
- *Potential Field (PF)*: We made our RMP controller behave like a PF by using *only* the *Obstacle-free Goal Reaching* + *Obstacle Avoidance* RMPs, with fixed metrics.
- *GDF-only*: Similarly, we only kept enabled the *GDF-based* and *Obstacle-free Goal Reaching* RMPs.

The controllers were tuned using line search, and executed 10 times each; we recorded the trajectories and number of collisions. Fig. 11 shows representative trajectories obtained with each method, as well as failure cases.

Our results show that, while most of the methods are able to reach the final goal, all the controllers except our proposed method hit the walls more than 10 times. The GDF controller was able to reach the final position but repeatedly bumped into the walls due to the lack of local awareness ①. PF managed to traverse the first third of the trajectory without collisions, but in 5 out of 10 trials it got stuck in a corner ② due to local minima. The precomputed trajectories of FALCO (yellow ■) allowed it to avoid most of the obstacles smoothly, but the robot was unable to pass through sharp curves such as ③, since the planner was unable to find free paths, or the P controller for path tracking was not able to react appropriately.

In contrast, our proposed RMP controller showed local awareness as PFs do, but the metrics (red circles ■ in Fig. 11) allowed it to leverage the fields dynamically, avoiding most of the obstacles. We only observed 3 minor bumps near the goal position ④, which are shown in the attached video.

[1]https://www.cmu-exploration.com/tare-planner

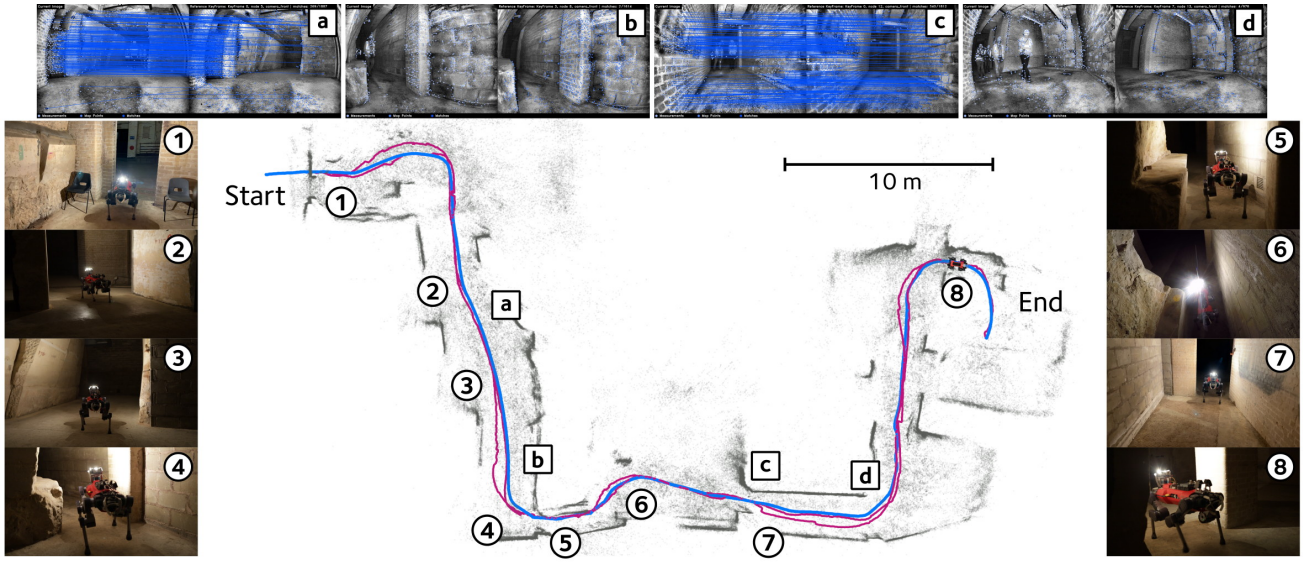

Fig. 9: Experiment 2: Underground Mine. Blue ■ 60 m teach trajectory; magenta ■ two VT&R-estimated repeat trajectories (forward and backward). The map in gray ■ was not used by the VT&R system and is only shown to illustrate the test area. The pictures on the sides depict the robot executing the mission, while examples of visual tracking are shown on top.

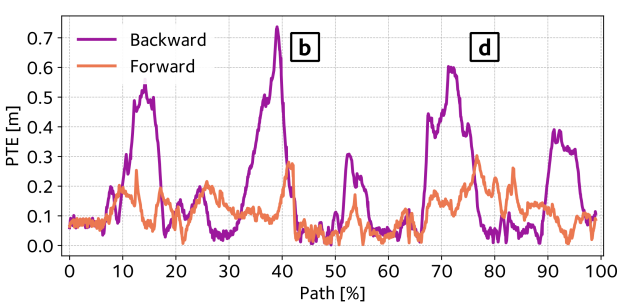

Fig. 10: Experiment 2: Path tracking error obtained in the underground mine for forward and backward traversals.

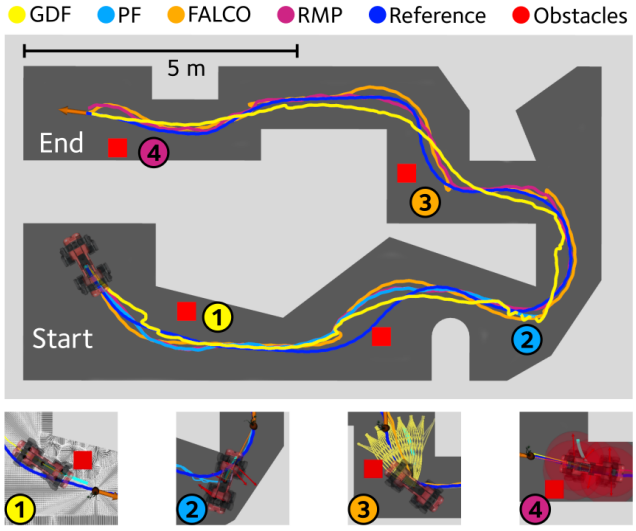

Fig. 11: Experiment 3: Representative trajectories tested in simulation for different controllers. The numbered captions illustrate some failure cases observed with each method. Please refer to the text and complementary video for more details.

V. Conclusions

In this work we presented a locally reactive controller coupled with our VT&R navigation system that allowed safe navigation in challenging environments. Our approach is based on computing efficient vector field representations from a local map, which are used to compute acceleration fields. We combined them using a Riemannian Motion Policies controller to generate a twist command at 10 Hz. We deployed our system on an ANYmal robot and showed that our locally reactive controller allowed the robot to traverse cluttered paths taught without obstacles. We also deployed the system in an underground mine, demonstrating safe navigation through challenging rooms and corridors, walking at 0.30 m/s on average.

Appendix
Frame transformations of acceleration fields

The *GDF-based Goal Reaching* and *Obstacle Avoidance RMPs* require to evaluate the gradients of SDF and GDF to generate acceleration fields in the body frame B. We determine the pose of the body in the map frame M as (see Fig. 4):

$$\mathbf{T}_{\mathtt{MB}} = \mathbf{T}_{\mathtt{IM}}^{-1}\mathbf{T}_{\mathtt{IB}} \tag{3}$$

This is used to evaluate the ∇f_{SDF} and ∇f_{GDF} at the desired location ${}_{\mathtt{M}}\mathbf{t}_{\mathtt{MB}}$ (translation components of $\mathbf{T}_{\mathtt{MB}}$). However, as the gradients are in M, we need to transform them to frame B.

The SE(2) adjoint of $\mathbf{T}_{\mathtt{MB}}$, denoted by $\mathrm{Ad}\left(\mathbf{T}_{\mathtt{MB}}\right)$, maps vectors defined in B to the frame M [34]. Its inverse does the opposite, allowing us to transform vectors defined in M to B:

$$\mathrm{Ad}\left(\mathbf{T}_{\mathtt{MB}}\right)^{-1} = \mathrm{Ad}\left(\mathbf{T}_{\mathtt{MB}}^{-1}\right) = \mathrm{Ad}\left(\mathbf{T}_{\mathtt{BM}}\right). \tag{4}$$

The adjoint is given by:

$$\mathrm{Ad}\left(\mathbf{T}_{\mathtt{BM}}\right) = \begin{bmatrix} \mathbf{R}_{\mathtt{BM}} & \mathbf{1}^{\wedge}{}_{\mathtt{M}}\mathbf{t}_{\mathtt{BM}} \\ 0_{1\times 2} & 1 \end{bmatrix} \tag{5}$$

where the hat operator $(\cdot)^{\wedge}$ builds a skew-symmetric matrix from the vector, and $\mathbf{R}_{\mathtt{BM}}$ and ${}_{\mathtt{M}}\mathbf{t}_{\mathtt{BM}}$ are the rotation and translation components of $\mathbf{T}_{\mathtt{BM}}$, respectively.

The gradient ∇f_{GDF} is a vector in $\mathbb{R}^2$ so we can add an extra zero for the angular component to define $\nabla f'_{\mathrm{SDF}} \in \mathbb{R}^3$. We then use the adjoint to transform the gradients to $\mathtt{B}$:

$${}_{\mathtt{B}}\nabla f'_{\mathrm{GDF}} = \mathrm{Ad}\left(\mathbf{T}_{\mathtt{BM}}\right) \nabla f'_{\mathrm{GDF}}. \tag{6}$$

We expand Eq. (6) using Eq. (5) to obtain:

$${}_{\mathtt{B}}\nabla f_{\mathrm{GDF}} = \mathbf{R}_{\mathtt{BM}} \nabla f_{\mathrm{GDF}} \tag{7}$$

which basically rotates the gradients to match the frame $\mathtt{B}$.

Evaluating the SDF is similar but we add a transformation defined at the center of the collision sphere at $\mathtt{S}_i$:

$$\mathbf{T}_{\mathtt{MS}_i} = \mathbf{T}_{\mathtt{IM}}^{-1}\, \mathbf{T}_{\mathtt{IB}}\, \mathbf{T}({}_{\mathtt{B}}\mathbf{t}_{\mathtt{BS}_i}) \tag{8}$$

with $\mathbf{T}({}_{\mathtt{B}}\mathbf{t}_{\mathtt{BS}_i})$ a transformation matrix built from the position of the sphere in the body frame.

REFERENCES

[1] 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.

[2] M. Fehr, T. Schneider, and R. Siegwart, "Visual-Inertial Teach and Repeat Powered by Google Tango," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018.

[3] W. Churchill and P. Newman, "Practice makes perfect? Managing and Leveraging Visual Experiences for Lifelong Navigation," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2012.

[4] N. D. Ratliff, J. Issac, D. Kappler, S. Birchfield, and D. Fox, "Riemannian Motion Policies," *CoRR*, 2018.

[5] M. Mattamala, M. Ramezani, M. Camurri, and M. Fallon, "Learning Camera Performance Models for Active Multi-Camera Visual Teach and Repeat," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2021.

[6] D. Kappler *et al.*, "Real-time Perception meets Reactive Motion Generation," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1864–1871, 2018.

[7] R. Buchanan, L. Wellhausen, M. Bjelonic, T. Bandyopadhyay, N. Kottege, and M. Hutter, "Perceptive Whole-body Planning for Multilegged Robots in Confined Spaces," *Journal of Field Robotics*, vol. 38, no. 1, pp. 68–84, 2021.

[8] M. Gaertner, M. Bjelonic, F. Farshidian, and M. Hutter, "Collision-Free MPC for Legged Robots in Static and Dynamic Scenes," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2021.

[9] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter, "Learning a State Representation and Navigation in Cluttered and Dynamic Environments," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 5081–5088, 2021.

[10] D. Kim *et al.*, "Vision Aided Dynamic Exploration of Unstructured Terrain with a Small-Scale Quadruped Robot," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.

[11] V. Vasilopoulos *et al.*, "Reactive Semantic Planning in Unexplored Semantic Environments Using Deep Perceptual Feedback," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4455–4462, 2020.

[12] S. R. Bista, B. Ward, and P. Corke, "Image-Based Indoor Topological Navigation with Collision Avoidance for Resource-Constrained Mobile Robots," *Journal of Intelligent and Robotic Systems*, vol. 102, no. 3, pp. 1–24, 2021.

[13] D. M. Bradley *et al.*, "Scene understanding for a high-mobility walking robot," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[14] P. Krüsi, P. Furgale, M. Bosse, and R. Siegwart, "Driving on Point Clouds: Motion Planning, Trajectory Optimization, and Terrain Assessment in Generic Nonplanar Environments," *Journal of Field Robotics*, vol. 34, no. 5, pp. 940–984, 2017.

[15] M. Brandão, O. B. Aladag, and I. Havoutis, "GaitMesh: Controller-Aware Navigation Meshes for Long-Range Legged Locomotion Planning in Multi-Layered Environments," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3596–3603, 2020.

[16] 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*, A. Koubaa, Ed. Springer International Publishing, 2016, pp. 99–120.

[17] L. Nardi and C. Stachniss, "Actively Improving Robot Navigation on Different Terrains Using Gaussian Process Mixture Models," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2019.

[18] 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.

[19] H. Oleynikova, A. Millane, Z. Taylor, E. Galceran, J. Nieto, and R. Siegwart, "Signed Distance Fields: A Natural Representation for Both Mapping and Planning," in *Robotics: Science and Systems*, 2016.

[20] J. Mainprice, N. Ratliff, M. Toussaint, and S. Schaal, "An Interior Point Method Solving Motion Planning Problems with Narrow Passages," in *IEEE Intl. Conf. on Robot and Human Interactive Communication (RO-MAN)*, 2020.

[21] J. A. Sethian, "A Fast Marching Level Set Method for Monotonically Advancing Fronts," *Proceedings of the National Academy of Sciences*, vol. 93, no. 4, pp. 1591–1595, 1996.

[22] K. Crane, C. Weischedel, and M. Wardetzky, "The Heat Method for Distance Computation," *Communications of the ACM*, vol. 60, no. 11, p. 90–99, 2017.

[23] A. Valero-Gomez, J. V. Gomez, S. Garrido, and L. Moreno, "The Path to Efficiency: Fast Marching Method for Safer, more Efficient Mobile Robot Trajectories," *IEEE Robotics and Automation Magazine*, vol. 20, no. 4, pp. 111–120, 2013.

[24] S. Pütz, T. Wiemann, M. Kleine Piening, and J. Hertzberg, "Continuous shortest paths vector field navigation on 3D triangular meshes for mobile robots," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2021.

[25] C. J. Ostafew, A. P. Schoellig, and T. D. Barfoot, "Robust Constrained Learning-based NMPC enabling reliable mobile robot path tracking," *International Journal of Robotics Research*, vol. 33, no. 1, pp. 133–152, 2016.

[26] L. P. Berczi and T. D. Barfoot, "Looking High and Low: Learning Place-dependent Gaussian Mixture Height Models for Terrain Assessment," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2017.

[27] C.-A. Cheng *et al.*, "RMPflow: A geometric framework for generation of multitask motion policies," *IEEE Transactions on Automation Science and Engineering*, vol. 18, no. 3, pp. 968–987, 2021.

[28] P. Fankhauser, M. Bloesch, and M. Hutter, "Probabilistic Terrain Mapping for Mobile Robots With Uncertain Localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3019–3026, 2018.

[29] A. Telea, "An Image Inpainting Technique Based on the Fast Marching Method," *Journal of Graphics Tools*, vol. 9, no. 1, pp. 23–34, 2004.

[30] M. Wermelinger, P. Fankhauser, R. Diethelm, P. A. Krüsi, R. Siegwart, and M. Hutter, "Navigation Planning for Legged Robots in Challenging Terrain," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016.

[31] P. F. Felzenszwalb and D. P. Huttenlocher, "Distance Transforms of Sampled Functions," *Theory of Computing*, vol. 8, no. 19, pp. 415–428, 2012.

[32] G. Peyré, M. Péchaud, R. Keriven, and L. D. Cohen, "Geodesic Methods in Computer Vision and Graphics," *Foundations and Trends in Computer Graphics and Vision*, 2010.

[33] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter, "Learning quadrupedal locomotion over challenging terrain," *Science Robotics*, vol. 5, no. 47, 2020.

[34] J. Solà, J. Deray, and D. Atchuthan, "A micro Lie theory for state estimation in robotics," *arXiv*, 2018.

[35] K. Zuiderveld, "Contrast Limited Adaptive Histograph Equalization." in *Graphic Gems IV*. Academic Press Professional, 1994.

[36] V. Peretroukhin, J. Kelly, and T. D. Barfoot, "Optimizing Camera Perspective for Stereo Visual Odometry," in *Canadian Conference on Computer and Robot Vision*, 2014.

[37] J. Zhang, C. Hu, R. G. Chadha, and S. Singh, "FALCO: Fast likelihood-based collision avoidance with extension to human-guided navigation," *Journal of Field Robotics*, vol. 37, no. 8, pp. 1300–1313, 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	An Efficient Locally Reactive Controller for Safe Navigation in Visual Teach and Repeat Missions
Publication Status	x Published □ Accepted for Publication □ Submitted for Publication □ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Matias Mattamala, Nived Chebrolu, Maurice Fallon, "An Efficient Locally Reactive Controller for Safe Navigation in Visual Teach and Repeat Missions," *IEEE Robotics and Automation Letters*, 2022.

Student Confirmation

Student Name:	Matias Mattamala		
Contribution to the Paper	• Developed the main idea • Wrote the local planner and baselines • Executed closed-loop field experiments • Performed data analysis and processing • Wrote large portions of the paper		
Signature 		Date	September 8, 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 Matias was the lead investigator, implementer and researcher on this project and wrote the bulk of the paper		
Signature 	Date	September 8, 2023

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

5.2 Additional Remarks

5.2.1 Computation of Pullbacks

In Eq. (2), we presented the RMP formulation of the local planning problem as:

$$\arg\min_{\mathtt{B}\mathbf{a}} \sum_{\{\mathtt{B}\mathbf{f},\mathtt{M}\}} \|\mathtt{B}\mathbf{f} - \mathtt{B}\mathbf{a}\|^2_{\mathtt{M}} + \sum_{\mathcal{S}} \sum_{\{\mathtt{S}\mathbf{f},\mathbf{M}\}} \|\mathtt{S}\mathbf{f} - \mathbf{J}_{\mathtt{SB}}\,\mathtt{B}\mathbf{a}\|^2_{\mathbf{M}}.$$

However, due to the page limit constraints of the original publication, not enough details were given about the computation of the pullback $\mathbf{J}_{\mathtt{SB}}$, which maps accelerations defined in the body frame B to the collision sphere's $\mathtt{S}$.

Following Fig. 4 on the article, the relative (planar) rigid body transformation between the robot's body and the collision sphere $\mathtt{S}_i$ is given by:

$$\mathbf{T}_{\mathtt{S}_i\mathtt{B}} = \begin{bmatrix} \mathbf{I}_{2\times 2} & {}_{\mathtt{S}}\mathbf{t}_{\mathtt{S}_i\mathtt{B}} \\ \mathbf{0} & 1 \end{bmatrix} \in \mathrm{SE}(2),$$

where we consider the relative orientation between both frames to be zero. Then, following Dellaert [70], the Jacobian of $\mathbf{T}_{\mathtt{S}_i\mathtt{B}}$ with respect to ${}_{\mathtt{S}}\mathbf{t}_{\mathtt{S}_i\mathtt{B}}$ is:

$$\mathbf{J}_{\mathtt{SB}} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

5.3 Discussion

In this chapter we presented a solution to endow the VT&R system from Chapter 4 with local awareness. While we called it a *locally reactive controller* in the article to be in line with the research that motivated this work, such as Kappler et al. [140] and Meng et al. [194], a more *correct* name would be *reactive local planner* to sit within other works we discussed in Sec. 2.4.

In the context of VT&R navigation, the approach by Krüsi et al. [158], which we did not include in our original publication, is one of the earliest works in this space. They presented a LiDAR Teach & Repeat system that used a sampling-based local planner to generate candidate paths that deviated from the original one to stay safe. A recent work by Sehn, Wu, and Barfoot [250] also addressed safe VT&R

navigation with a sampling based local planner, reducing the sampling space to a *safe corridor* around the reference path.

The motivation to formulate a local planning approach based on RMPs, was that it relies on the same LS principles we discussed in Chapter 2, from the geometric point-of-view. It does not require sampling states or optimising a trajectory, and it directly provides the twist command required by the legged locomotion controller.

A particular advantage of this approach, is that it relied on *different* representations of the local map. Representations such as SDFs are commonly used along TO methods to obtain gradients that guide the optimisation [203]. On the other hand, methods such as FMM are alternatively used for local planning by their own [279]. By using the RMPs framework we could combine them in a seamless way, and control their contribution via the Riemannian metrics. This allowed us to effectively combine representations that perform explicit planning, such as the GDF (via FMM), with other analytically-specified behaviours.

This is also one of the main limitations nonetheless, as the combined policy loses some of the guarantees that the source ones offer. For example, the GDF is guaranteed to find a field that guides the robot to the goal, but combining it with a SDF to provide additional close-obstacle awareness can still lead the robot to local minima. Parameter tuning becomes fundamental to balance the contribution of each motion policy and their respective Riemannian metrics, which can be improved by data-driven methods [202] or more advanced theoretical frameworks [42, 299].

This work motivated new questions to enable navigation of legged robots in more challenging environments. Simple local planners could generate safe motion without the computational overheads of other methods as long as the traversability representation was consistent with the robot capabilities. This is the idea I explore in the next chapter.

6
Online Traversability Learning

The navigation systems presented up to this chapter were designed to operate in structured (e.g. laboratories, industrial facilities) and semi-structured (e.g. mines) environments, where traversable space can be easily defined using geometry. Natural scenes such as parklands and forests challenge these definitions of traversability, as geometry alone cannot be used to determine if elements such as grass or bushes are traversable by a given platform. As it was discussed in Chapter 1, legged platforms have shown impressive mobility skills over different terrain but they can be limited by mission planners if traversability is not well defined.

This chapter presents a system to rapidly estimate robot-specific traversability via online learning. It exploits pre-trained Visual Transformer (ViT) models combined with an online approach for self-labeling, allowing the robot to learn the traversable areas from visual data, after a few minutes of teleoperation in the field.

We integrated the learned traversability measure with the reactive local planner from Chapter 5 to achieve autonomous navigation. The approach allowed us to quickly deploy a legged robot for navigation tasks in new environments such as forests and grasslands without prior knowledge.

6.1 Fast Traversability Estimation for Wild Visual Navigation

The following project was started during a research stay at the Robotic Systems Lab (RSL) at ETH Zurich, which I visited from April to September 2022. The work was jointly developed with Jonas Frey, doctoral student at the RSL. After I returned to Oxford in October, we continued working on the project until January, when it was submitted. As the project was jointly developed, Jonas and I share the first-authorship. The statement of authorship provides further details on my specific contributions to this work.

The article summarising this project was presented in *Robotics: Science and Systems (RSS)* 2023 [94]. Accompanying videos are available online at: https://sites.google.com/leggedrobotics.com/wild-visual-navigation.

Fast Traversability Estimation for Wild Visual Navigation

Jonas Frey*[,1,3] Matias Mattamala*[,2] Nived Chebrolu[2] Cesar Cadena[1] Maurice Fallon[2] Marco Hutter[1]
[1] ETH Zurich [2] University of Oxford [3] Max Planck Institute for Intelligent Systems
* Equal contribution. jonfrey@ethz.ch, matias@robots.ox.ac.uk

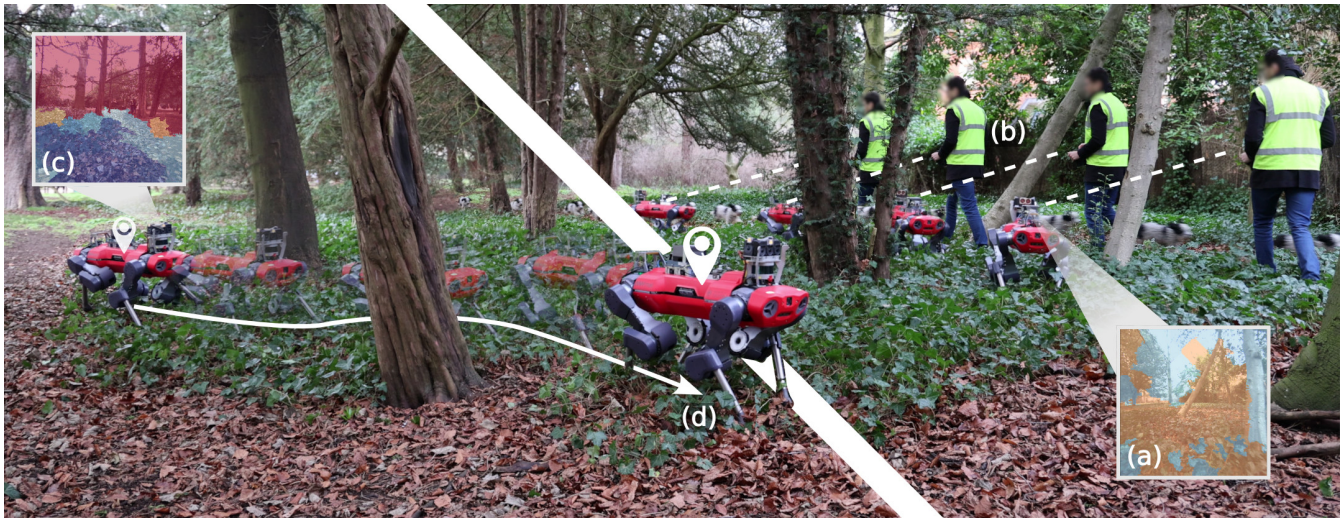

Fig. 1: Wild Visual Navigation (WVN) learns to predict traversability from images via online self-supervised learning. Starting from a randomly initialized traversability estimation network without prior assumptions about the environment (a), a human operator drives the robot around areas that are traversable for the given platform (b). After a few minutes of operation, WVN learns to distinguish between traversable and untraversable areas (c), enabling the robot to navigate autonomously and safely within the environment (d).

***Abstract*—Natural environments such as forests and grasslands are challenging for robotic navigation because of the false perception of rigid obstacles from high grass, twigs, or bushes. In this work, we propose Wild Visual Navigation (WVN), an online self-supervised learning system for traversability estimation which uses only vision. The system is able to continuously adapt from a short human demonstration in the field. It leverages high-dimensional features from self-supervised visual transformer models, with an online scheme for supervision generation that runs in real-time on the robot. We demonstrate the advantages of our approach with experiments and ablation studies in challenging environments in forests, parks, and grasslands. Our system is able to bootstrap the traversable terrain segmentation in less than 5 min of in-field training time, enabling the robot to navigate in complex outdoor terrains — negotiating obstacles in high grass as well as a 1.4 km footpath following. While our experiments were executed with a quadruped robot, ANYmal, the approach presented can generalize to any ground robot. Project page: bit.ly/3M6nMHH**

I. Introduction

Traversability estimation is a core capability needed to allow robots to autonomous navigate in field environments. It is understood as the *affordance* [14] necessary for a robot to navigate within its environment, i.e to understand which areas can be accessed and navigated through and at what cost. While the topic has been widely studied for wheeled or flying robots supported by 3D sensors using the traditional approach of occupancy mapping [27], the development of new platforms with advanced mobility skills, such as legged robots, prompts a reconsideration of current definitions of traversability, as new and more complex types of natural terrain can be traversed [25].

It is difficult to infer traversability within natural terrains, such as high grass or forest undergrowth. Occupancy-based navigation systems based on 3D sensing are often confused by high grass and incorrectly classify such terrain as an obstacle which is untraversable — even if the platform is actually able to pass through. Semantic understanding is important in such environments to determine which terrain is actually passable for a particular robot.

Existing approaches, which build upon deep neural models for semantic segmentation [24] or anomaly detection [37], have demonstrated navigation in off-road environments; however there are recurring problems with the collection and labelling of large amounts of relevant training data. In addition to the effort required to curate these datasets, specific class labels (tree, branch, bush, grass) do not directly correspond to the capabilities of the robotic platform.

Self-supervised systems have addressed this challenge by generating labeled datasets from past robot deployments, using classification carried out in hindsight [36] or using predictions of the robot motion [13]. Nevertheless, these previous methods are still trained on robot-specific datasets and subsequently deployed without further adaptation. If they were to be tested in a new environment or on a new robotic platform, new data would need to be collected and the models would need to be retrained, limiting applicability.

Achieving online self-supervision and learning are key to overcoming these aforementioned limitations, as traversability could be more easily learned in the field. The Learning Applied to Ground Vehicles (LAGR) program [19, 15] pioneered this direction, generating supervision during the mission and training machine learning models for traversability estimation on the fly. They showed first demonstrations of autonomous robots navigating off-road environments under this approach.

In this work, we build upon such advances and present a system for vision-based traversability estimation that achieves online, self-supervised adaptation, by improving on various components of previous works. We name the approach Wild Visual Navigation (WVN), as it is capable of learning which terrain is traversable by a robot after a few minutes of manual demonstrations *in the wild*. The system builds upon four core ideas that we consider the key contributions of our work:

- **A self-supervision system** designed for real-time operation, which concurrently generates supervision signals from vision and traversability measurements from proprioception and control performance.
- **A learning approach** that leverages high-dimensional, self-supervised visual features extracted using pre-trained vision transformer models, which are fed into a small neural network and efficiently trained online.
- **A new formulation for traversability estimation** that combines supervised learning with anomaly detection, accounting for uncertainties due to the sparse supervision.
- **Closed-loop integration** with local mapping and planning methods which demonstrate that these traversability estimates are suitable for autonomous navigation tasks.

We have extensively validated our approach with ablation studies which compare against similar approaches that are trained in an offline fashion. Further, we deployed our system on real hardware, the ANYbotics ANYmal C robot, showing that it can be easily trained for navigation tasks in environments where it would not be possible to traverse using geometric mapping alone. We demonstrate fast adaptation within minutes to determine the traversable areas in a natural environment, achieving closed-loop operation in a forest with different understory foliage and terrain, and a kilometer-scale path-following task in a park where the semantic class of the path was not explicitly labeled. While our experimental results have been demonstrated on a legged robot, the principles we present are general and applicable to other ground platforms.

II. Related Work

A. Traversability from geometry

Classical approaches for traversability estimation analyze the geometry of the environment using 3D sensing [27]. Recent examples from the DARPA SubT Challenge [7], used different representations such as point clouds [4] and meshes [16] to evaluate navigational metrics such as risk or stepping difficulty [9].

However, a purely geometric analysis is not sufficient to completely capture traversability for a given platform. Data-driven methods can bridge this gap by learning platform-specific traversability from data or simulations. For example, Chavez-Garcia et al. [6] learned traversability from simulations of a ground robot moving on an elevation map. Yang et al. [40] extended this approach for legged platforms, capturing the risk of failure, energy cost and time required for navigation. Recently, Frey et al. [11] expanded this approach to volumetric data and massive parallelization in data collection from simulation. While we recognize the contribution those previous works make in using robot-specific geometric data to determine traversability, using geometry-only does not succeed in natural environments containing natural growth such as high grass, branches or bushes. We instead focus on vision-based methods herein, as they describe semantic features that are challenging to capture otherwise.

B. Traversability from semantics

Semantic segmentation methods have been proposed to address the aforementioned challenges by assigning navigation costs to the different semantic classes. Bradley et al. [3] presented a scene understanding system trained and evaluated using geographically diverse data. Maturana et al. [24] demonstrated autonomous off-road navigation using semantics projected onto 3D map around a wheeled platform. Schilling et al. [33] used semantically segmented features that were classified into fixed classes using a random forest. Recently Shaban et al. [35] presented an approach for off-road navigation that learns a dense traversability map from sparse point-clouds.

We note that these methods typically rely on pre-trained or fine-tuned semantic segmentation models. This requires specific class labels to be defined; and these semantic classifiers can be difficult to reuse in different environments. Nevertheless, new advances in self-supervised models, such as DINO-ViT [5], are able to segment semantically meaningful classes without manual supervision. We exploit these promising tools in this work.

C. Traversability from self-supervision

Purely semantic methods are challenging to use *in the wild* because (1) it is difficult to represent the relevant classes of these operating environments without labeled data, and (2) assigning a traversability cost to each class, which can also be arbitrary. Kim et al. [19] exploited information about the areas visited by a wheeled robot to determine positive traversable samples, while a bumper was used for negative labeling.

Bajracharya et al. [2] used a similar approach combined with a height heuristic to determine long-range negative samples.

Modern methods rely on deep-learned models trained from weakly supervised data, and the supervision strongly depends on the hardware platform. Wellhausen et al. [36] used the reprojected footholds from a legged robot to provide supervision; Zürn et al. [41] exploited sounds produced by the platform moving on different terrain as a proxy for supervision; Gasparino et al. [13] instead used the receding-horizon trajectory of a Model Predictive Controller (MPC). Recently, TerraPN [32] used odometry and IMU signals as supervision and could learn a traversability model in 25 min – including data collection and learning. Our work is inspired by these approaches and follows similar self-supervision strategies but we aim for concurrent supervision signal generation and learning achieving orders of magnitude faster adaptation.

D. Traversability from anomalies

Anomaly detection methods are motivated by one of the key challenges of using self-supervised approaches, namely an imbalance in the number of positive and negative samples. Instead of training a discriminative model of traversability, they focus on learning generative models of the traversed terrain. This distribution can then used as a proxy to set out-of-distribution (OOD) inputs as being untraversable. Richter and Roy [31] developed a visual navigation system that relied on an autoencoder to predict OOD scenes from images, switching to safer navigation behaviors when traversing novel environments. Wellhausen et al. [37] used multi-modal sensing from haptics, vision and depth to classify as anomalies visual elements such as flames and water reflections in navigation tasks. Ji et al. [17] formulated a proactive anomaly detection approach that evaluated candidate trajectories for local planning depending on their probability of failure. While we do not explicitly use anomalies to determine traversability, we do use it as a confidence metric to leverage the sparse supervision signals. Similar ideas have been recently presented by Seo et al. [34] to determine traversability with point cloud data.

E. Traversability from demonstrations

Inverse Reinforcement Learning (IRL) is a framework which can learn a reward function from demonstrations. In our context this can be interpreted as a traversability that encodes the preference to navigate certain areas. Ratliff et al. [30] showed how IRL methods could be used to generate large-scale mission plans from aerial images. Later, Wulfmeier et al. [39] learned cost maps to encode driving preferences using deep neural networks, which was extended by Gan et al. [12] to guide the navigation of a legged robot in natural environments using a local reward map. While our approach uses the examples from a human operator in our self-supervised framework instead of IRL; we show that it can also be used to encode preferences depending on the demonstration and representation of the terrain.

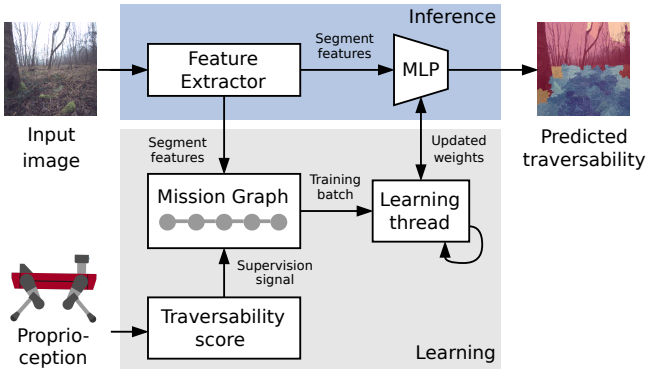

Fig. 2: System overview: WVN only requires monocular RGB images and proprioceptive data as input, which are processed to extract features and supervision signals used for online learning and inference of traversability (see Sec. III).

Symbol	Definition
$\mathbf{I}$	RGB image with height H and width W
$\mathbf{F}$	Feature map with dimensions $E \times H \times W$, $E = 90$
$\mathbf{M}$	Weak segmentation mask with height H and width W
$\mathbf{S}$	Reprojected supervision with dimensions $H \times W \in [0, 1]$
τ	Traversability score $\in [0, 1]$
$\mathbf{f}_n$	Per-segment embedding of dimension $E = 90$
τ_n	Per-segment traversability score

TABLE I: Main definitions used in this work

F. Adaptive traversability estimation

A few works have demonstrated adaptive traversability estimation and navigation behavior. As part of LAGR, Kim et al. [19] ran a clustering algorithm during deployment, and assigned positive and negative labels to the clusters through interactions to determine the traversable areas. Hadsell et al. [15] followed a similar approach to learn different binary classifiers during exploration, using features learned from a Convolutional Neural Network (CNN), making it one of the first works to use neural models for this purpose. Later Lee et al. [21] used Bayesian clustering on SLIC superpixels using color and texture features supervised by motion priors; this approach was able to propagate the traversability labels to similar segments during operation. More recently, BADGR [18] presented an end-to-end policy designed to adapt from experiences, though it was not framed for online learning.

Our approach builds upon the LAGR ideas for self-supervision and online learning but we extend them with more powerful semantic features from a pre-trained visual transformer that allowed us to deploy our system in a variety of natural environments.

III. Method

A. System Overview

The objective of this work is to design a navigation system that estimates dense traversability from RGB images using a neural network model learned online, in a self-supervised manner, using labels generated by a robot interacting with its

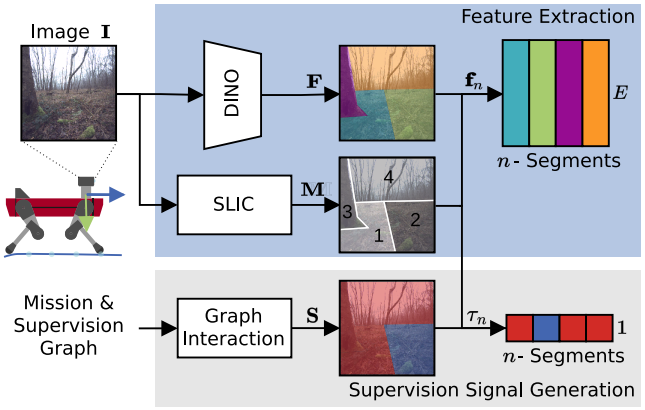

Fig. 3: Feature Extraction: Our approach extracts dense DINO-ViT features $\mathbf{F}$ from an RGB image $\mathbf{I}$, and $n = 100$ segments $\mathbf{M}$ using SLIC. For each of the n segments we average the corresponding features to obtain per-segment embeddings $\mathbf{f}_n$. A traversability score τ_n is computed for each segment based on the score resulting from the graph interaction process (for more detail refer to Sec. III-D).

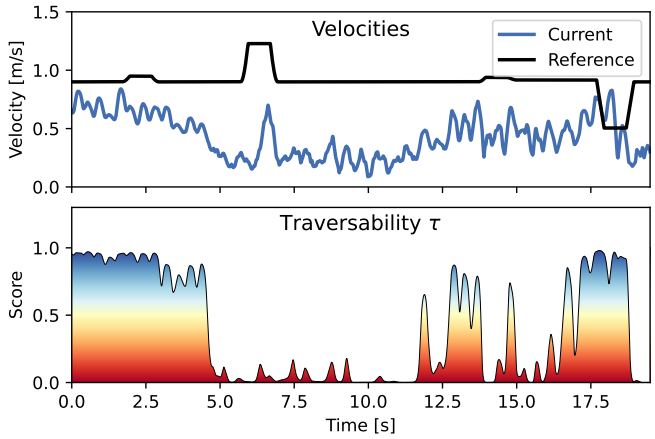

Fig. 4: Traversability score generation: We compute a score based on the difference of an (external) reference velocity command and the current velocity estimated by the robot. Closely tracking the reference is interpreted as moving over traversable terrain (in blue ■), high discrepancies indicate otherwise (red ■).

environment. We target a system which requires only a brief demonstration from a human operator for data collection and learning.

Our proposed system WVN, builds upon different modules shown in Fig. 2. *Feature extraction* (Sec. III-B) and *traversability score generation* (Sec. III-C) process the incoming sensor data to extract features and traversability scores. The *mission and supervision graphs* (Sec. III-D) manage the data and generate the self-supervision signals, while the *learning thread* (Sec. III-E) performs online traversability and anomaly learning. The interaction between the different modules is illustrated in Fig. 2, and their implementation is covered in the following sections. The main definitions used in the rest of the paper are summarized in Tab. I.

B. Feature Extraction

Given an RGB image $\mathbf{I}$, we first extract dense, pixel-wise visual feature maps (*embeddings*) $\mathbf{F}$. In contrast to previous works based on fine-tuned CNNs, we rely on recent self-supervised network architectures to leverage a Vision Transformer (ViT) trained using the DINO method [5] – DINO-ViT. These learned representations have been demonstrated to encode meaningful semantic and instance information without requiring any labels.

Since we aim for real-time operation, the processing and storage of the full dense features on a mobile robot is prohibited by the limited compute and storage availability. Instead, we follow previous works [21] and compute a weak segmentation mask $\mathbf{M}$ of the input image $\mathbf{I}$ using superpixels. We use SLIC [1] to extract 100 segments per image. We then average the feature maps segment-wise resulting in a single embedding $\mathbf{f}_n$ per segment. Fig. 3 illustrates the complete feature extraction process.

C. Traversability Score Generation

Defining which terrain is traversable or not depends on the capabilities of the specific platform. We define a continuous *traversability score* $\tau \in [0, 1]$, where 0 is untraversable and 1 fully traversable. Previous works have used ground reaction forces, audio supervision, or predictions from a MPC as a proxy for this score. Instead, we adopt a simpler approach that uses the discrepancy between the robot's current linear (x, y) velocity as estimated by the robot $\mathbf{v}$, and the reference velocity command $\bar{\mathbf{v}}$ given by an external human operator or planning systems. The intuition is that when the robot moves on terrain that is easily traversable it should closely track the reference command. In contrast, if the robot struggles to track the reference, the discrepancy will grow, and we can interpret it as a less traversable terrain. We define the mean squared velocity error as:

$$v_{\text{error}} = \frac{1}{2}\left((\bar{v}_x - v_x)^2 + (\bar{v}_y - v_y)^2\right) \in \mathbb{R} \tag{1}$$

As v_{error} will be a noisy scalar signal, we smooth it with a 1-D Kalman Filter before passing it through a sigmoid function to obtain a valid traversability score:

$$\tau = \text{sigmoid}\left(-k\left(v_{\text{error}} - v_{\text{thr}}\right)\right) \tag{2}$$

with k the steepness of the sigmoid, and v_{thr} the midpoint of the sigmoid that assigns a traversability score of 0.5. These values can be calibrated depending on the motion specifications of each platform and determine how the velocity error is stretched to the $[0, 1]$ interval.

D. Supervision and Mission Graphs

In contrast to other methods that generate the supervision signal in post-processing [36, 13, 32], we execute this process online by accumulating information about the recent history of operation. Our approach is inspired by graphical SLAM

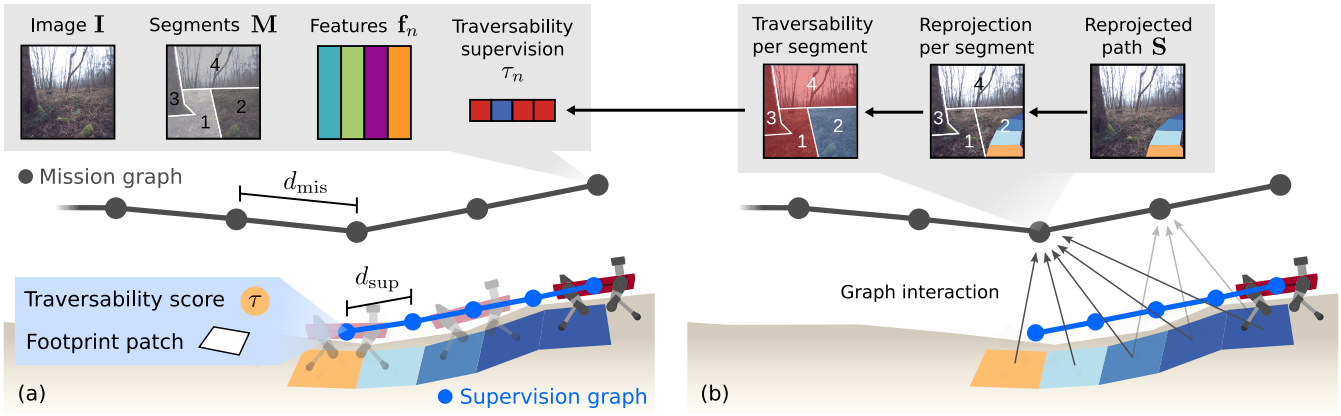

Fig. 5: Supervision and mission graphs: (a) Information stored in each graph over the mission. While the Supervision Graph only stores temporary information about the robot's footprint in a sliding window, the Mission Graph saves the data required for online learning over the full mission. The color of the footprint patches indicates the generated traversability score. (b) The interaction between graphs updates the traversability in the mission nodes by reprojecting the robot's footprint and traversability scores.

pipelines that often leverage both local and global graphs to integrate measurements: we maintain a *Supervision Graph* to store short-horizon traversability data, and a global *Mission Graph* which stores the generated training data during a mission, shown in Fig. 5.

1) Supervision Graph: The supervision graph stores within its nodes information about the current time, robot pose, and estimated traversability score (Sec. III-C). This graph is implemented as a ring buffer, which only keeps a fixed number of nodes N_{sup}, separated from each other by a distance distance d_{sup}. The product $N_{\text{sup}}\, d_{\text{sup}}$ determines the maximum length (in physical distance) of the supervision graph.

The stored information is a footprint track with traversability scores τ, used to generate supervision signals in hindsight that are reprojected onto previous camera viewpoints as it is explained in the following sections.

2) Mission Graph: On the other hand, the mission graph stores all the information required for learning over the mission. The mission nodes are added to the graph after feature extraction if the distance w.r.t the last added node is larger than d_{mis}. Each mission node contains the RGB image $\mathbf{I}$, the weak segmentation mask $\mathbf{M}$ and per-segment features $\mathbf{f}_n$ with their corresponding traversability supervision τ_n.

3) Graph interaction for supervision generation: When a new mission node is added, it triggers an interaction between the graphs to update the supervision labels τ_n stored in each mission node (Fig. 5b). We reproject the footprint track and corresponding traversability scores τ onto all the images of the mission nodes that are within the range of the supervision graph. This ensures the information we reproject stays locally consistent in spite of potential state estimator drift.

Each mission node then has an auxiliary image with the reprojected path, $\mathbf{S}$. We use the weak segmentation mask $\mathbf{M}$ to assign per-segment traversability supervision values τ_n by averaging the score over each segment. Segments that do not overlap with the reprojected footprint track are set to zero (i.e untraversable). Then we obtain pairs of per-segment features $\mathbf{f}_n$ and traversability score τ_n for each mission node ready for training.

Fig. 5 illustrates the supervision and mission graphs, and their interaction to generate supervision signals online.

E. Traversability and Anomaly Learning

We aim to train a small neural network in an online fashion that determines the feature traversability score τ_n from a given segment feature $\mathbf{f}_n$. This allows us to predict the traversability of the scene in front of the robot from a full image by inferring only about those segments. Additionally, we explicitly model the uncertainty about the unvisited (and hence, unlabeled) areas by using anomaly detection techniques to bootstrap a confidence estimate. Our formulation also deals with non-stationary data distributions induced by continuously updating the training data and model weights.

First, we will elaborate on how a confidence score for a segment is obtained; and then we will describe the traversability estimation task which takes as input the confidence and is jointly trained.

1) Confidence Estimation: To obtain a segment-wise confidence estimate we aim to learn the distribution over all traversed segment features $\mathbf{f}_n$. A encoder-decoder network $f^{\theta_r}_{\text{reco}}$ is trained to compress the segment feature $\mathbf{f}_n$ into a low dimensional latent space and consecutively reconstruct the original input features $\mathbf{f}_n$. The reconstruction loss is given by the Mean Squared Error (MSE) between the predicted features and the original feature compute over all channels E:

$$\mathcal{L}_{\text{reco}}(\mathbf{f}_n) = \delta_{\tau_n \neq 0} \frac{1}{E} \sum_{e} \| f^{\theta_r}_{\text{reco}}(\mathbf{f}_{n,e}) - \mathbf{f}_{n,e} \|^2, \qquad (3)$$

where $\delta_{\tau \neq 0}$ is 1 if the segments feature traversability score τ_n is not zero, and 0 otherwise; this ensures that the network only learns to reconstruct the embeddings that are labeled in an anomaly detection fashion. As a consequence, the trained network reconstructs known (*positive*) feature embeddings, i.e. similar to the traversable segments, with small reconstruction

loss; feature embeddings of unknown (*anomalous*) segments the network was never tasked to reconstruct, such as trees or sky, induce a high reconstruction loss. Since the network is trained online, this results in a multi-modal reconstruction loss distribution that evolves over time, as shown in Fig. 6.

The unbounded reconstruction loss $\mathcal{L}_{\text{reco}}$ for a segment is mapped to a confidence measure $c(\mathcal{L}_{\text{reco}}) \in [0, 1]$ by first identifying the mode of the traversed segment losses. For this we fit a Gaussian distribution $\mathcal{N}(\mu_{\text{pos}}, \sigma_{\text{pos}})$ over the reconstruction losses per batch of the traversed segments (i.e, positive samples):

$$n_{\text{trav}} = \sum_{\mathbf{f}} \delta_{\tau_n \neq 0}, \tag{4}$$

$$\mu_{\text{pos}} = \frac{1}{n_{\text{trav}}} \sum_{\mathbf{f}\,:\,\tau_n \neq 0} \mathcal{L}_{\text{reco}}(\mathbf{f}_n), \tag{5}$$

$$\sigma_{\text{pos}} = \sqrt{\frac{1}{n_{\text{trav}}} \sum_{\mathbf{f}\,:\,\tau_n \neq 0} \left(\mathcal{L}_{\text{reco}}(\mathbf{f}_n) - \mu_{\text{pos}}\right)^2} \tag{6}$$

We set the segment confidence to 1 if the loss of the segment is smaller than μ_{pos} and otherwise to the unnormalized Gaussian likelihood:

$$c(\mathcal{L}_{\text{reco}}(\mathbf{f}_n)) = \exp\left(\frac{(\mathcal{L}_{\text{reco}}(\mathbf{f}_n) - \mu_{\text{pos}})^2}{2(\sigma_{\text{pos}}\, k_\sigma)^2}\right), \tag{7}$$

where we introduce the tuning parameter k_σ, which allows to scale the confidence.

2) Traversability Estimation: A small network $f^{\theta_t}_{\text{trav}}$ with a single channel output is trained to regress on the provided segment traversability score τ. For the untraversed segments with unknown traversability score we follow a conservative approach and we assume it to be zero $\tau = 0$, though we use the confidence score to scale their overall contribution. The loss for traversability estimation is computed using the confidence-weighted MSE:

$$\mathcal{L}_{\text{trav}}(\mathbf{f}) = \delta_{\tau_n = 0} \sum_n (1 - c(\mathbf{f}_n)) \left\| f^{\theta_t}_{\text{trav}}(\mathbf{f}_n) - 0 \right\|^2 + \delta_{\tau_n \neq 0} \sum_n \left\| f^{\theta_t}_{\text{trav}}(\mathbf{f}_n) - \tau_n \right\|^2.$$

Effectively, for segments where a traversability score is available by interaction the MSE is computed. For unlabeled segments, the traversability is assumed to be zero but weighted based on the confidence score. Areas similar to the one traversed should be assigned a $c(\mathbf{f})$ close to 1, therefore contributing insignificantly to the total loss. On the other hand anomaly areas (never traversed before, low $c(\mathbf{f})$ score) induce a high loss if predicted with a high traversability score by f_{trav}.

As we aim to provide the estimated traversability as input for a local planning system, it is desired to automatically define a threshold to determine the traversable and untraversable areas. We propose a strategy to select the traversability threshold τ_{thr} by measuring the current performance of the system in a self-supervised manner. We compute the Receiver Operating Characteristic (ROC) throughout training by classifying all segments with confidence under 0.5 as negative and traversed

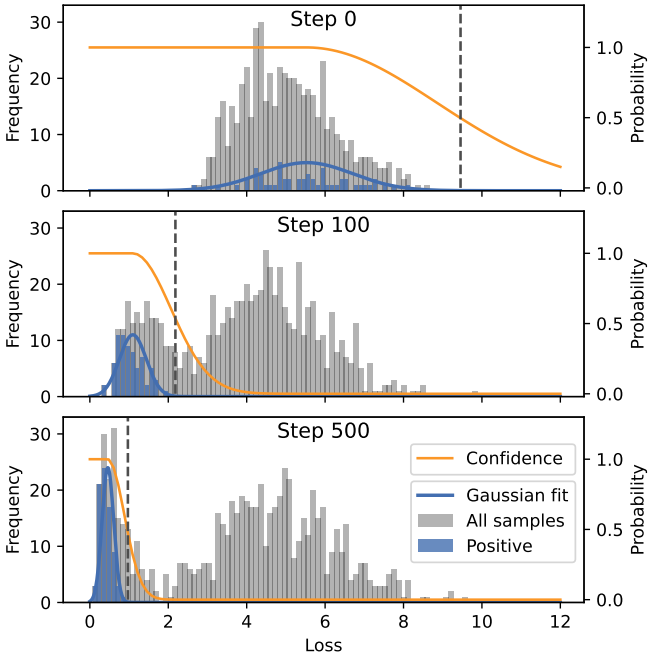

Fig. 6: Histogram of the reconstruction loss $\mathcal{L}_{\text{reco}}$ distribution for all segments within a batch at optimization step 0, 100, and 500. Traversed segments (positive samples) are shown in blue and non-traversed segments in grey. We observed that the total loss distribution becomes bi-modal as the training develops, which we use to determine a threshold to scale the confidence estimate (see Sec. III-E1). At 500 steps all segments with a reconstruction loss $\mathcal{L}_{\text{reco}}$ over 2 are identified as fully anomalous (unconfident) and therefore induce a high loss in the traversability score when identified as traversable. The vertical grey dashed line indicates the decision boundary if a segment is detected as traversable or untraversable by the anomaly detection.

segments as positive labels. Then, we decide on the traversability threshold only by setting the desired False Positive Ratio (FPR), though other metrics such as the Youden's index can also be used.

3) Implementation details: In our implementation, $f^{\theta_r}_{\text{reco}}$ and $f^{\theta_t}_{\text{trav}}$ are implemented by a two-layer Multi-Layer Perceptron (MLP) with [256, 32] unit dense layers and ReLU non-linear activation functions. Both networks share the weights of the hidden layers. $f^{\theta_r}_{\text{reco}}$ has a reconstruction head with E output neurons and $f^{\theta_t}_{\text{trav}}$ a single channel traversability head followed by a sigmoid activation. The 32-channel hidden layer functions as the bottleneck of the encoder-decoder structure. The total loss per segment during training is given by:

$$\mathcal{L}_{\text{total}}(\mathbf{f}) = w_{\text{trav}} \mathcal{L}_{\text{trav}}(\mathbf{f}) + w_{\text{reco}}\, \mathcal{L}_{\text{reco}}(\mathbf{f}). \tag{8}$$

with w_{trav} and w_{reco} allowing to weigh the traversability and reconstruction loss respectively. We used Adam [20] to jointly train the networks with a fixed constant learning rate of 0.001. For a single update step, 8 valid mission nodes are randomly chosen to form a data batch; we defined a node as valid if at least a single segment of the node has non-zero traversability score. For all our experiments we set $k_\sigma = 2$, $w_{\text{trav}} = 0.03$,

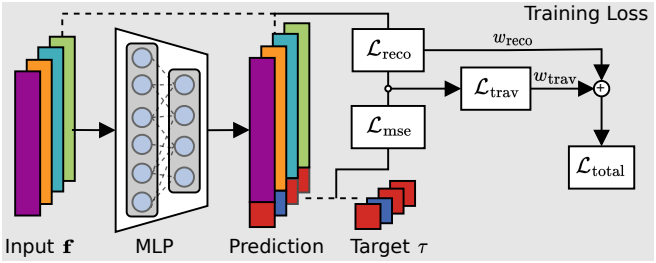

Fig. 7: Network architecture and per-segment losses used for online training.

$w_{\text{reco}} = 0.5$ and use a maximum FPR of 0.15 to determine the traversability threshold.

In Sec. V-C, we present and evaluate different design choices for learning methods, features and architectures. We also evaluate the advantages of our loss formulation compared to other common approaches for traversability, such as anomaly detection (learning a distribution over positive samples) and regression on the traversability score without modeling the uncertainty.

IV. Closed-Loop Integration

We integrated the learned traversability into a standard navigation pipeline to achieve autonomous navigation with a quadrupedal platform. The details of each module of the system are explained in the following sections.

A. Local terrain mapping

To map the environment surrounding the robot we used an open-source terrain mapping framework [26] to efficiently obtain a robot-centric 2.5D elevation map from the onboard depth cameras and LiDAR sensing. We extended this framework in Erni et al. [8] to fuse our predicted traversability into the local map representation. As our predicted traversability is an image, we used raycasting to take into account the occlusions with the terrain, establishing correspondences between pixel-wise traversability values in the image plane and the local map's grid cells. This procedure allows for temporal fusion of the traversability information in the map while preserving the values of previously observed areas via exponential averaging. Since this indirectly uses geometry, the projection method is susceptible to artifacts due to spikes in the elevation map.

B. Local planning

We used the projected visual traversability from the local map as a costmap for local planning. A median filter removed undesired noise and artifacts before the distance transform method [10] was used to obtain a Signed Distance Field (SDF), which represents the distance to the closest untraversable object. We implemented a local planner method based on Mattamala et al. [23], which exploits the SDF and the local goal to generate a $\mathrm{SE}(2)$ twist command which drives the robot towards the goal while avoiding untraversable terrain. Finally, the twist command becomes the input to a robust learning-based locomotion controller based on the work by Miki et al. [25], which is able to traverse rough terrain typically inaccessible to wheeled robots.

C. Smart carrot for autonomous exploration

Lastly, to generate an autonomous navigation behavior we implemented a simple exploration strategy by analyzing the robot-centric SDF created by the local planner, by choosing a *moving carrot* that drives the robot forward.

The goal pose is given by analyzing a section of the SDF in front of the robot and selecting the position with the largest distance from all obstacles, ensuring the robot stays at the center of the traversable space. The goal is continuously updated when the SDF is recomputed with new traversability information. While this strategy was simple it could safely guide the robot to follow a footpath autonomously using the predicted traversability without requiring a global plan or a large-scale representation of the environment. Nevertheless, further improvement could be achieved by using a sophisticated exploration planning system.

V. Experiments

A. Platform Description

For our experiments we used an ANYbotics ANYmal C legged robot. The robot is equipped with an additional NVidia Jetson Orin AGX to run WVN onboard, which was implemented in pure Python code using PyTorch [28] and ROS 1 [29].

The main sensing input for our system are monocular, wide Field of View (FoV) color images from a single global shutter Sevensense Alphasense Core camera. Additionally, the default state estimator provides $\mathrm{SE}(3)$ pose and body velocity measurements. The LiDAR and depth cameras available on the robot were only used for the local terrain mapping module as described in Sec. IV-A.

B. Real-world deployments

We executed different deployments to validate WVN in different environments. The first experiments highlight adaptation to new environments and the advantages of the visual traversability estimation for local planning tasks. The last two experiments demonstrate autonomous navigation among obstacles and fully autonomous large-scale path following.

1) Fast adaptation on hardware: Our first experiment involved teleoperating the robot around 3 loops of a park environment walking on grass and dirt, on open areas and around trees. The goal was to evaluate the fast adaptation capabilities of WVN while running on the robot.

Fig. 8 illustrates the main outcomes of the experiment, showing that the system learned to predict robot-specific traversability over the 3 loops. In particular, section **(a)** shows how the robot starts with a very poor segmentation after 9 steps of training (21 s), this greatly improves after 800 steps (2 min) where it can correctly segment the dirt as traversable terrain while keeping the tree untraversable. Similar behavior occurs in section **(b)** in which the segmentation is conservative at the beginning but it extends across the other grass patches in

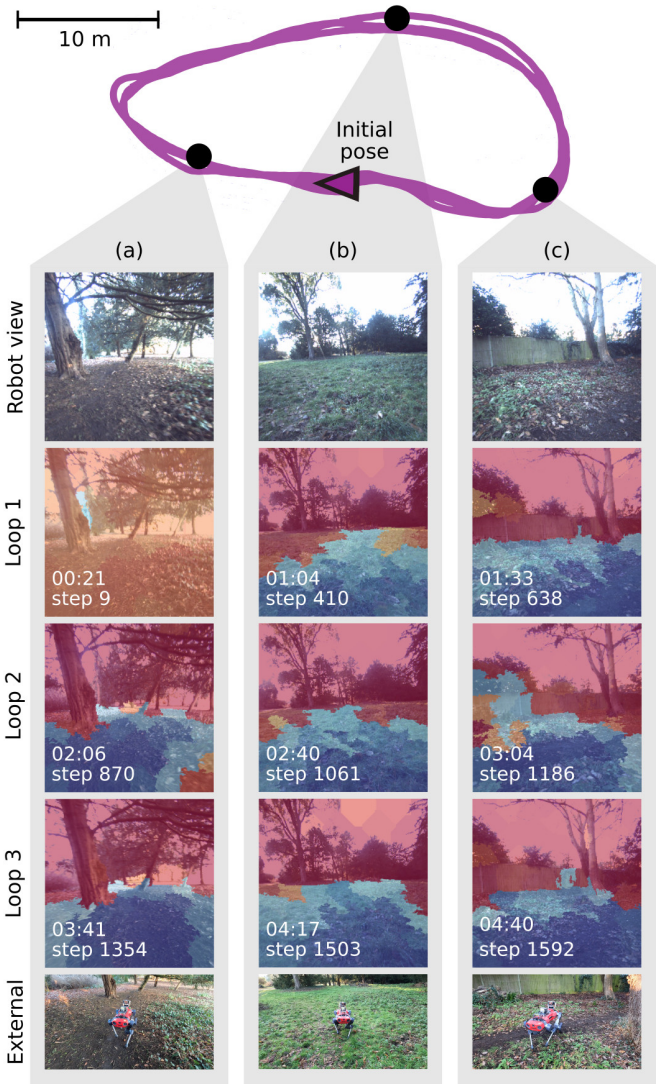

Fig. 8: Adaptation on real hardware: We tested the online adaptation capabilities of our system by teleoperating the robot to complete 3 loops in a park (top, route shown in ■). The columns show different parts of the loop (a,b,c); each row displays the improvement of the traversability estimate over time and training steps.

later iterations. Section **(c)** also illustrates some issues related to the SLIC segmentation, as some segments of the wooden wall (step 1186) are incorrectly clustered with patches of the grass, which is not observed in the other captures.

2) Benefits of visual traversability vs geometric methods: Our second experiment aimed to illustrate the advantages of visual traversability estimation in challenging natural environments. Similarly to the previous experiment, we teleoperated the robot — but in a forest with high grass, loose branches, and bushes. Fig. 9, bottom right, shows a representative shot of the experiment, a robot's view input image and WVN's prediction, which illustrates the challenges of the environment for both vision and geometry based approaches.

To compare the different traversability methods, we used the terrain mapping module described in Sec. IV-A, as it allowed us to compare geometry-only and visual traversability. In particular, we compared against two geometric methods that are real-time capable and have been used in previous literature:

- Geometric method based on heuristics such as height and slope of the terrain [38].
- Geometric method based on a learned model of traversability, which is part of the terrain mapping system [26].
- Visual traversability provided by WVN, raycasted onto the terrain map.

The geometric methods only require an elevation representation of the surface, and can directly determine traversability from the 2.5D geometry. For WVN we executed a training procedure driving the robot around the environment for a few minutes first.

Fig. 9 illustrates the output *traversability map* obtained by all the methods (bottom), as well as the corresponding SDFs generated from them. (top) The geometric methods correctly determine the trees as untraversable areas, as they are based on the 2.5D representation. Our system is also able to successfully discriminate the trees, confirming the findings observed in Sec. V-B1. However, the important advantages of our method are observed in high-grass areas, which are represented as elevation spikes in the map that are classified as untraversable by the geometric approaches. WVN correctly characterized the capabilities of the robot to successfully traverse the terrain, as demonstrated by the human operator.

When comparing the SDFs such differences become more evident, as all the areas with low traversability scores become obstacles. Our system correctly determines that all the grass patches are traversable, hence the SDFs displays the right classification and disregards the geometry. The only limitation of our current method is that we use just a single camera and can only predict traversability for the areas in sight. To ensure safety, unknown areas are assumed to be untraversable but future work will take advance of the robot's other cameras.

3) Point-to-point autonomous navigation between trees: After validating our approach in teleoperated settings, we executed closed-loop navigation tasks to demonstrate WVN can easily adapt to a new environment, and the learned traversability estimate can be used to deploy the robot autonomously.

We taught the robot to navigate in a woodland area containing dirt, high grass, and trees. A human operator drove the robot for 2 min through loose dirt and grass — an area that can be easily traversed by the legged platform. Then we commanded the local planner to execute autonomous point-to-point navigation avoiding obstacles, only using the visual traversability for closed-loop planning Sec. IV.

Fig. 10 illustrates the scene used for the experiment and the trajectories used for training and testing autonomous navigation. The robot successfully managed to reach 8 out of 8 goals, where the human operator deliberately chose targets behind trees to challenge the system. This was achieved even

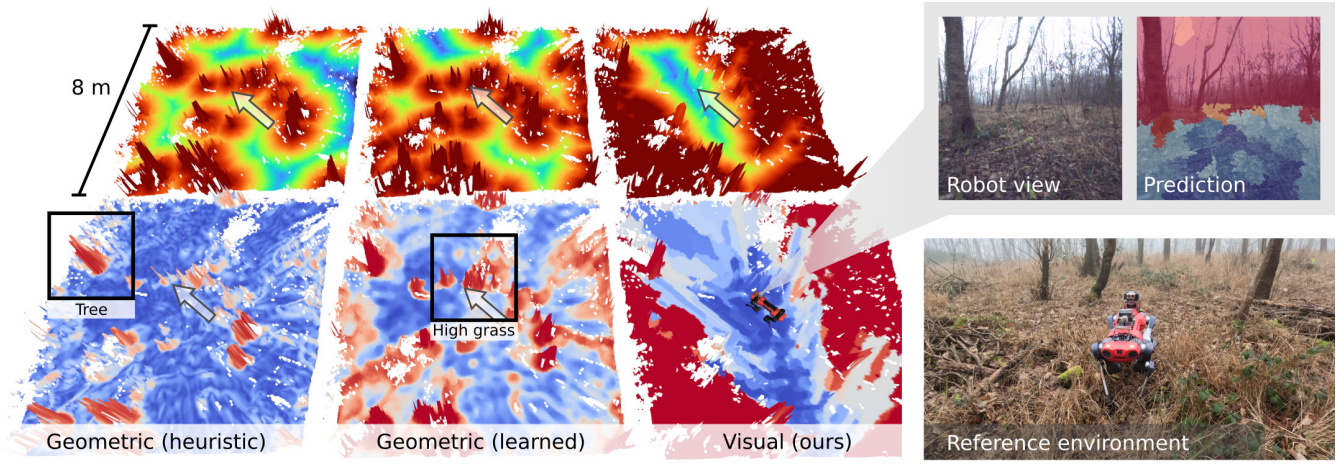

Fig. 9: Visual vs geometric traversability: Illustration of traversability map (bottom row) and corresponding SDF (top row) for three different traversability estimation methods applied to the same terrain patch. Our visual traversability estimate provides clear advantages for local planning compared to geometric methods, where the latter get heavily affected by traversable high grass or branches (bottom row). This is evident when comparing the SDF's, where geometry-based methods are more sensitive to the spikes produced by high grass areas (top row).

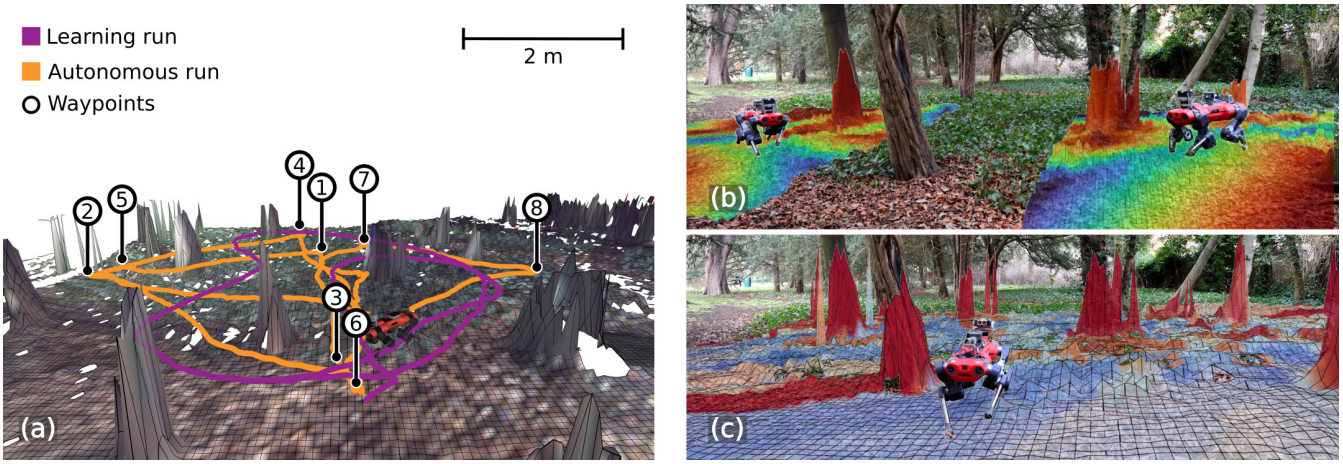

Fig. 10: Point-to-point autonomous navigation: (a) After teleoperating the robot for 2 min (path shown in ■), we successfully achieved autonomous navigation in a woodland environment (path shown in ■). (b) Some of the SDFs generated from the predicted traversability during autonomous operation. (c) Global 2.5D reconstruction of the testing area and predicted traversability, generated in post-processing to illustrate the capabilities of our approach.

though neither geometry nor any additional assumptions about the environment were used during training.

We also show some examples of the SDFs generated during operation used by the local planner in subfigure (b), which indicate the trees as obstacles. Lastly, in processing we fused the predicted traversability measures into a complete map in subfigure (c), which correctly aligned with the trees. However, we did observe some obstacle artifacts due to limitations of the approach, namely the use of a single camera for the predictions, the coarse segmentation from SLIC, and the raycasting process, which we further discuss in Sec. V-D.

4) Kilometer-scale autonomous navigation in the park: As a last experiment, we demonstrated that WVN can also be used to achieve preference-aware path-following behavior as a result of the human demonstrations and the online learning capabilities of the system.

We executed 3 experiments to demonstrate this in a park. Similarly to our previous experiments, we trained the system for less than 2 min along the footpath. However, we then disabled the learning thread to ensure that the predicted traversability strictly mimics the human preference during the demonstration run. The goal for the robot to follow is given by the *smart carrot* module described in Sec. IV-C, which autonomously guided the robot forward along the path.

In the 3 runs the robot was able to follow the path for hundreds of meters — mostly staying in the center of the path, avoiding grass, bushes, benches, and pedestrians. Fig. 11 shows the trajectories followed in each run, starting from different points in the footpath. For runs 1 and 3 we used the same parameters, $k_\sigma = 2$ and FPR$= 0.15$. In run 2 we

Fig. 11: Kilometer-scale navigation: We deployed our system to learn to segment the footpath of a park after training for a few steps. We executed 3 runs starting from different points in the park: ■ *run 1* (0.55 km), ■ *run 2* (0.5 km), and ■ *run 3* (1.4 km). Minor interventions were applied to guide the robot in intersections; major interventions (⋆) were required for some areas when the robot miss-classified muddy patches for the path.

relaxed the parameters to $k_\sigma = 3$ and FPR$= 0.3$, producing a less conservative behavior that drove the robot to other visually similar areas in the park (very muddy grass) requiring manual intervention to correct the heading. When the robot approached an intersection we adjusted, if necessary, the heading to follow the desired footpath.

Overall, we achieved autonomous behavior that would have been difficult to achieve using only geometry, as the path boundaries were often geometrically not distinguishable. On the other hand, instead of training and using a semantic segmentation system to learn *all* the possible traversable classes in the park (pavement, gravel path, roadway or grass), we showed that this short teleoperated demonstration of the gravel footpath was enough for WVN to generate semantic cues to achieve the desired path following behavior.

C. Offline validation via ablation studies

To complement the results of our hardware experiments and validate our design decisions, we performed different ablation studies of the individual components of WVN. All the experiments were executed offline on an Nvidia RTX3080 Laptop GPU with Intel i7-11800H CPU.

1) Dataset overview: For offline analysis we used 3 large-scale datasets of new areas not used for the real experiments:

- *Hilly:* a hillside with dense vegetation and fruit trees.
- *Forest:* a fir forest with hiking paths.
- *Grass:* a grassland area with moderate inclines and varying vegetation surrounding a small lake.

The datasets were also recorded with a teleoperated ANYmal C platform and similar sensing setup to the previous experiments. Fig. 12 shows aerial views of the paths that were used for data collection, as well as some samples of the specific areas that were traversed during this operation.

We organized the collected data into training, validation, and testing data, which is summarized in Tab. II. The longest sequence recorded in each site (Fig. 12, shown in purple ■) is used for training and validation purposes. The first 80% of the sequence are used to generate training data, with the remaining 20% kept for validation. The remaining sequences of each scene are subsampled and exclusively used for testing.

Regarding the labels, we manually segmented images from the test split into traversable (1) and untraversable (0) classes, which we named *ground truth labels* (*GT*). These binary labels reflect the intuition of an expert robot operator on which places are safely accessible for the robot, and were used for quantitative assessment of the design decisions. While simulation-based evaluations could provide a precise traversability score, we disregarded it because of the limitation of sim-to-real transfer of vision-based methods. Human supervision was straightforward and could incorporate more subtle risks and preferences.

On the other hand, we also used the labels generated by executing WVN over the sequences using the self-supervision approach, which we name *SELF*. Since the *GT* labels were binary due to intrinsic challenges of producing ground truth continuous traversability signals, we also binarized the *SELF* labels for a fair comparison.

Lastly, even though we only had access to binary ground truth labels we did not change the regression formulation presented in Sec. III-E, as we could also obtain a binary output by thresholding the continuous signal proposed by our system. This resulted in a classification task that could be evaluated using metrics such as Accuracy (Acc).

2) Evaluation method: For all of our experiments we trained and tested in the same environment using the splits previously presented, unless stated otherwise. Regarding the

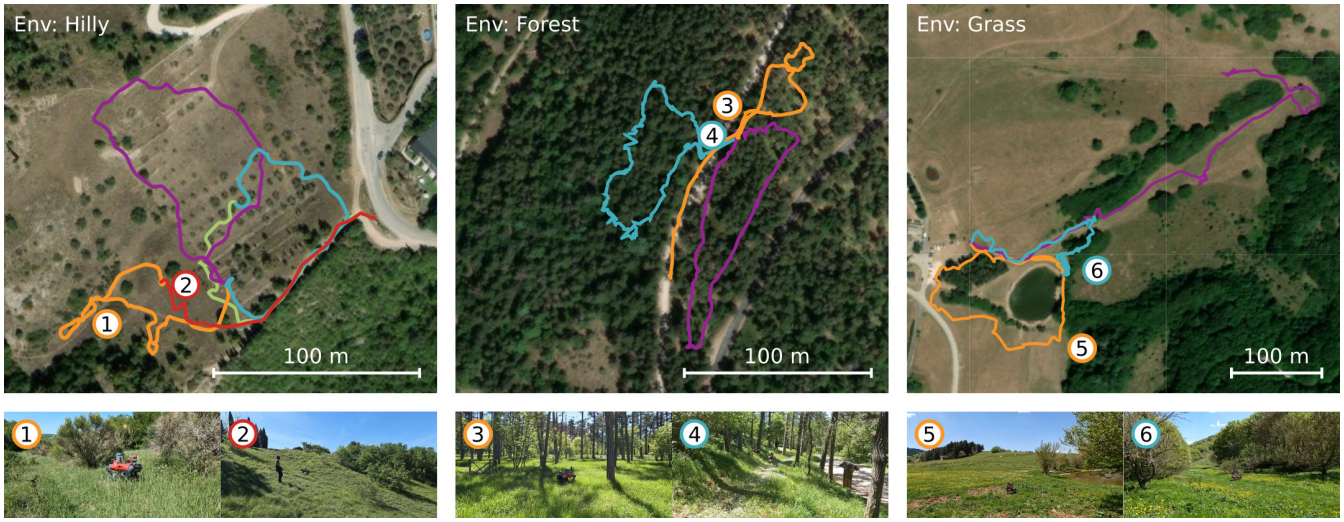

Fig. 12: Aerial views of the 3 environments used for offline testing of our system, illustrating the paths used for data collection and scene examples. The purple ■ trajectories are used for training and the remaining for validation.

TABLE II: Dataset Overview

Env	Split	Duration	Distance	# Traj	# Image	Label
Hilly	Train	512.4 s	262 m	1	920	*SELF*
	Val	121.1 s	66 m	1	230	*SELF*
	Test	1202.2 s	840 m	4	55	*GT*
Forest⋆	Train	402.1 s	606 m	1	991	*SELF*
	Val	134.0 s	151 m	1	247	*SELF*
	Test	970.5 s	896 m	2	41	*GT*
Grass	Train	860.3 s	857 m	1	2050	*SELF*
	Val	242.5 s	214 m	1	512	*SELF*
	Test	2196.2 s	1224 m	3	113	*GT*

⋆ Length measured using RTK-GPS and may not reflect the real-distance traversed within the forest.

Env	Metric	RF	SVC-RBF	SVC-Poly	WVN
Hilly	*GT*	71.36 ± 0.0	65.46 ± 0.53	73.96 ± 1.61	**81.05** ± 1.11
	SELF	88.32 ± 0.0	87.51 ± 0.25	84.81 ± 0.31	78.58 ± 0.82
Forest	*GT*	**83.07** ± 0.44	74.66 ± 0.55	76.28 ± 1.34	**82.45** ±1.10
	SELF	81.95 ± 0.44	90.19 ± 0.20	88.09 ± 0.53	85.26 ± 1.24
Grass	*GT*	59.16 ± 0.0	63.64 ± 0.33	69.02 ± 2.00	**78.21** ± 2.39
	SELF	88.14 ± 0.0	87.79 ± 0.14	86.74 ± 0.39	82.60 ± 0.29

TABLE III: Learning Method: Traversability Accuracy of Random Forest (RF), Support Vector Classifier (SVC), and WVN with respect to the binary ground truth labels *GT* and self-supervised labels *SELF* on the ablation environments.

metrics, we evaluated our system in a binary classification setting due to the limitations of the *GT* labels previously discussed. Hence, we reported the Acc for all of our experiments. The accuracy is computed in image space (i.e. pixel-wise) as opposed to segment-wise, which accounts for misclassifications induced by image segments containing both traversable and untraversable terrain.

Lastly, all compared neural network models are trained for 1000 steps with 5 different random seeds, and we report confidence intervals for all the metrics.

3) Study 1: Learning Methods: Our first study compared our approach to *classical* machine learning methods previously used for traversability estimation, namely Random Forest (RF) [33] and Support Vector Classifier (SVC) [2]. For SVC we followed their work and use a polynomial kernel of degree 2 and Radial Basis Function (RBF) kernel. All methods are trained using the DINO-ViT features.

Tab. III summarizes the main results of this study. We observed that *our* method based on an MLP performs consistently good across all three environments with respect to the *GT* labels (see Tab. III). Interestingly, the RF is competitive and still performs strongly across scenes, specially well in the *Forest* scene. The results confirm that our chosen model overall outperforms machine learning methods previously used in the literature, while allowing us to continuously adapt the model using gradient descent during the mission.

4) Study 2: Training Objective: We then studied the impact of various training objectives on the model performance. In particular, we compared four approaches:

- *Trav*: We trained our network to directly regress on the traversability by assuming all untraversed segments are untraversable: no reconstruction loss $w_{\text{reco}} = 0$, with full confidence $c(\mathbf{f}) = 1$, and without self-supervised τ_{thr} thresholding.
- *Fixed threshold*: We fixed the traversability threshold τ_{thr} to a value of 0.5.
- *Anomaly detection*: We used the confidence score to classify features with a confidence over 0.5 as traversable.
- WVN: Our full method.

Our proposed method consistently outperforms the other settings (Tab. IV). A significant performance increase (+5.17%) was achieved over the *Trav* simplest traversability setting by adding the confidence-weighted traversability loss formulation

Env	Metric	Trav	Fixed-Threshold	Anom	WVN
Hilly	*GT*	65.46 ± 0.53	73.96 ± 1.61	72.92 ± 0.82	**81.05** ± 1.11
	SELF	87.51 ± 0.25	84.81 ± 0.31	69.21 ± 0.85	78.58 ± 0.82
Forest	*GT*	74.66 ± 0.55	76.28 ± 1.34	70.31 ± 1.12	**82.45** ±1.10
	SELF	90.19 ± 0.20	88.09 ± 0.53	68.37 ± 1.44	85.26 ± 1.24
Grass	*GT*	63.64 ± 0.33	69.02 ± 2.00	77.64 ± 0.44	**78.21** ± 2.39
	SELF	87.79 ± 0.14	86.74 ± 0.39	75.29 ± 0.21	82.60 ± 0.29

TABLE IV: Training Objective: Traversability Accuracy for different learning objectives with respect to the binary ground truth labels *GT* and self-supervised labels *SELF* on the ablation environments. Refer to *Study 2: Training Objective* for further details.

(*Fixed-threshold*). Generally, in the *Trav*-setting more regions are classified as untraversable leading to an over-conservative traversability estimation, which performed well on the *SELF*-labels but does not reflect the *GT* traversability. Further improvement of 7.33% can be achieved by adding the online traversability threshold scaling. Here it is important to mention that a fixed threshold is insufficient during deployment given the online adaptation of the network. Lastly, the experiment using only *Anom* detection performs reasonably well, suggesting that a meaningful confidence score was learned across various environments. This indicates that the learned anomaly detection is useful for guiding the traversability learning of our method.

5) Study 3: Features: The quality of extracted segment features can have a significant impact on the performance of traversability prediction, given that we only consider a fixed feature extraction backbone. Hence, we evaluated the performance of the selected self-supervised pre-trained DINO-ViT backbone against popular residual network architectures pre-trained on ImageNet (ResNet-50, EfficientNet-B4), and also trained using self-supervised learning, such as ResNet-50 trained with DINO (DINO-ReN). We also compared against other classical features, such as dense SIFT [22] features over RGB channels. We fed the features to the *trav*-setting model introduced in the previous study, to isolate the impact of the feature extractor on the traversability estimation performance from other procedures. We present the model parameter count for each feature extractor and inference time of all network architectures on a Jetson Orin in Tab. V. The features generated by methods using self-supervised pre-training clearly outperformed pre-trained models on ImageNet, even when using the same architecture, aligning with the findings of Caron et al. [5]. The short inference time measured on the Orin board also validates the choice of DINO-ViT as the backbone suitable for WVN.

6) Study 4: Scene Adaptation: We evaluated the performance of WVN when trained on one environment and tested on all the others, to test the necessity for online adaptation. Tab. VI shows the resulting accuracy for each scene combination. We observed that in general the best performance is achieved when testing on the same training environment as expected, dropping otherwise. The model trained on *Hilly* performs overall the best and showed specially good performance on *Grass*, which we suspect is due to the visual similarity of the scenes (see Fig. 12).

	Architecture	Param	Time	*GT*	*SELF*
Hilly	DINO-ViT	21M	17.0 ms	**65.46** ± 0.53	87.51 ± 0.25
	DINO-ReN	23.5M	15.9 ms	64.05 ± 0.08	86.71 ± 0.03
	EffNet-B4	17.5M	26.1 ms	62.55 ± 0.07	86.09 ± 0.02
	ResNet-50	23.5M	15.9 ms	61.87 ± 0.08	85.37 ± 0.02
	SIFT	0	-	58.78 ± 0.73	82.80 ± 0.03
Forest	DINO-ViT	21M	17.0 ms	**74.66** ± 0.55	90.19 ± 0.20
	DINO-ReN	23.5M	15.9 ms	74.14 ± 0.24	89.12 ± 0.21
	EffNet-B4	17.5M	26.1 ms	73.00 ± 0.76	89.63 ± 0.29
	ResNet-50	23.5M	15.9 ms	72.71 ± 0.22	88.49 ± 0.10
	SIFT	0	-	60.94 ± 0.00	83.27 ± 0.00
Grass	DINO-ViT	21M	17.0 ms	63.64 ± 0.33	87.79 ± 0.14
	DINO-ReN	23.5M	15.9 ms	**67.08** ± 0.36	88.15 ± 0.13
	EffNet-B4	17.5M	26.1 ms	61.01 ± 0.78	85.91 ± 0.00
	ResNet-50	23.5M	15.9 ms	62.99 ± 0.03	85.23 ± 0.12
	SIFT	0	-	57.61 ± 0.91	83.79 ± 0.02

TABLE V: Comparison of feature extraction backbones.

Training	**Hilly**	**Forest**	**Grass**
Hilly	**81.05** ±1.11	82.14 ± 1.78	**82.14** ± 0.63
Forest	75.86 ± 2.18	**82.45** ±1.10	75.80 ± 2.80
Grass	77.49 ± 4.36	73.22 ± 6.38	78.21 ±2.39

TABLE VI: Scene Adaptation: Traversability Accuracy with respect to the *GT* labels. Each row corresponds to a training run on the specific environment and testing on all environments.

In general, we remark that even though the robot was deployed within scenes featuring similar semantic classes (e.g. trees or high grass), on the same day and within a few kilometers radius, the performance still degraded. This suggests even worse performance drops for changing seasons or urban to natural environment scene changes. We argue that even though this can be hypothetically mitigated by increasing the amount of training data, this is costly, and online adaptation provides a practical solution to enable the deployment of robots in new or changing environments.

7) Study 5: Adaptation Speed & Dataset Size: For our final study we investigated how fast can WVN adapt to the new environments and how many data samples are needed. To examine this we designed an experiment in which we trained the network for 1000 steps for different training dataset sizes, ranging from 10% to 100% of the original size. We measured the accuracy each 2nd step and every 10% increment of the dataset size.

As a result, we obtained heatmaps displaying the performance evolution across these 2 variables, shown in Fig. 13. For all environments starting from a randomly initialized network, we observed that good performance can be achieved within 200 steps. We argue that this is due to use of segments: adding a single image provides 100 new training samples for the network. During continuous training, we also observed some slight fluctuation with respect to the test accuracy. Fig. 14

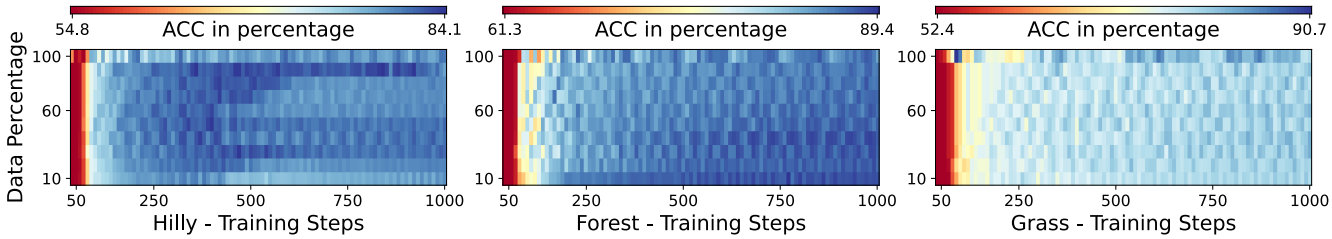

Fig. 13: Adaptation Speed vs Dataset Size: The performance measured by the accuracy increases over training as expected. In the limit case of training for a few steps (< 50), the performance is equally degraded — independent of the dataset size. A small dataset size is sufficient for good performance. Please observe the different color scales for each environment. In the *Grass* environment the color scale is distorted by an outlier when using 100% of the data after 70 steps.

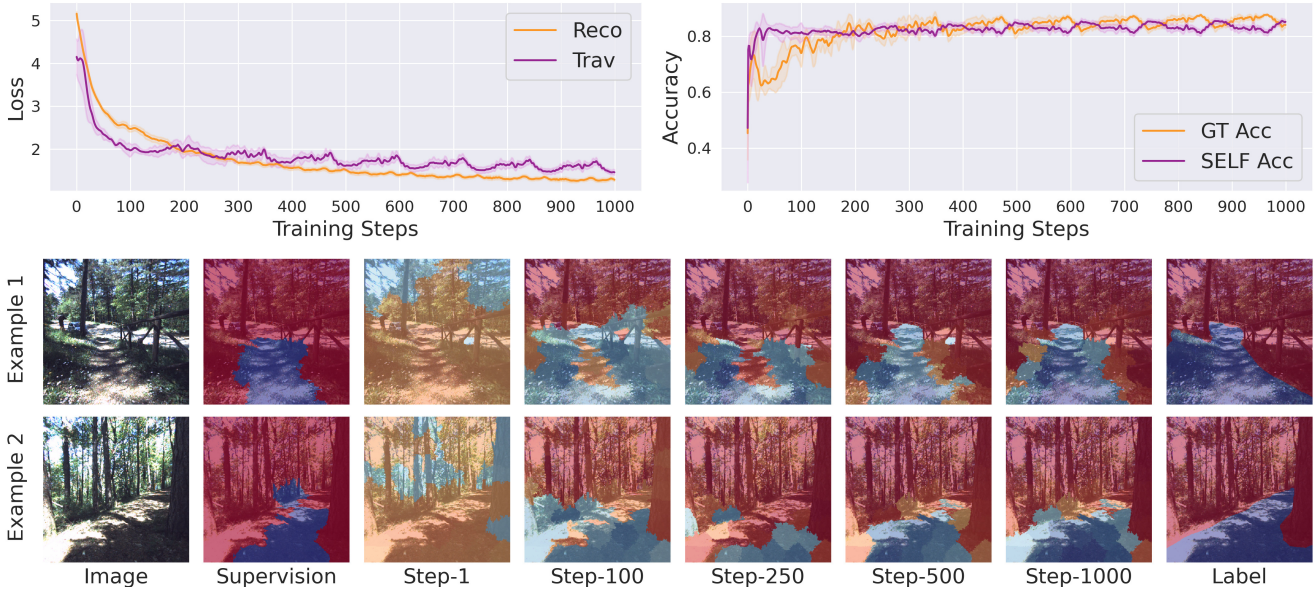

Fig. 14: Training Process: We detail the incremental training process executed by WVN in terms of the loss (top left), accuracy (top right), and visual examples (bottom).

shows the training loss accuracy over time, illustrating some of the fluctuating behavior, as well as example images of the output segmentation over training.

D. Limitations

We have demonstrated that WVN can learned a traversability estimate online, allowing immediate deployment of robots in new environments. However, we observed some limitations during our ablation studies and field deployment.

- WVN runs at 2.5 Hz on the NVidia Orin board with unoptimized code. Improving the efficiency of the pipeline would allow for faster training and better autonomous navigation performance.
- The use of superpixels to reduce the computational complexity limits the accuracy of the output segmentation, as the SLIC segments are only similar from a color-space sense, consequently affecting the computation of features per segment by averaging semantically different features.
- While the use of velocity tracking error was a simple proxy for the traversability score, it does not fully characterize the different interactions the robot can have with the environment. Further investigation is required to determine alternative metrics that can be more suitable for specific platforms.
- For closed-loop integration we projected the traversability prediction onto a local terrain map, which allowed for a straightforward integration with the local planner. However, this presented important drawbacks due to perspective projection, limited FoV due to single camera usage, and raycasting on an inaccurate 2.5D map that required additional filtering stages.
- Lastly, our system could be also framed within the continual learning paradigm. While we do not address it explicitly, we believe that WVN could greatly benefit from the advances of continual learning — not only to adapt within a single mission — but between different test environments.

VI. Conclusion

We presented Wild Visual Navigation (WVN), a system that leverages the latest advances in pre-trained self-supervised networks with a scheme to generate supervision signals while a robot operates, to achieve online, onboard visual traversability estimation. The fast adaptation capabilities of our system allowed us to easily deploy robots for navigation tasks in new environments after just a few minutes of learning from human demonstrations. We validated WVN through different ablation studies and real-world experiments, illustrating its fast adaptation capabilities, the consistency of its traversability prediction for local planning, and 1.4 km closed-loop navigation experiments in natural scenes. Our experiments show that WVN can enable autonomous robot navigation by learning from small data *in the wild*. We aim to tackle the current limitations of our system by exploring data-driven self-supervised methods for segment extraction, possibly mitigating artifacts induced by segments containing traversable and untraversable terrain. We also plan to extend the current implementation to multiple cameras — allowing the system to learn from different inputs and estimate traversability in different directions for more complex local planning. For future work specific to legged systems capable to negotiate challenging terrain, we aim to further close the loop between WVN's traversability prediction and feedback provided directly by the locomotion policy about the traversability of the terrain.

Acknowledgments

This work was supported by the Swiss National Science Foundation (SNSF) through project 188596, the National Centre of Competence in Research Robotics (NCCR Robotics), the European Union's Horizon 2020 research and innovation program under grant agreement No 101016970, No 101070405, and No 852044, and an ETH Zurich Research Grant. Jonas Frey is supported by the Max Planck ETH Center for Learning Systems. Matias Mattamala is supported by the National Agency for Research and Development (ANID) / DOCTORADO BECAS CHILE/2019 - 72200291 and NCCR Robotics. Maurice Fallon is supported by a Royal Society University Research Fellowship.

References

[1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, 2012. III-B

[2] Max Bajracharya, Andrew Howard, Larry H. Matthies, Benyang Tang, and Michael Turmon. Autonomous off-road navigation with end-to-end learning for the lagr program. *Journal of Field Robotics*, 26(1):3–25, 2009. II-C, V-C3

[3] David M. Bradley, Jonathan K. Chang, David Silver, Matthew Powers, Herman Herman, Peter Rander, and Anthony Stentz. Scene understanding for a high-mobility walking robot. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 1144–1151, 2015. II-B

[4] Chao Cao, Hongbiao Zhu, Fan Yang, Yukun Xia, Howie Choset, Jean Oh, and Ji Zhang. Autonomous Exploration Development Environment and the Planning Algorithms. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, page 8921–8928, 2022. II-A

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging Properties in Self-Supervised Vision Transformers. In *International Conference on Computer Vision (ICCV)*, 2021. II-B, III-B, V-C5

[6] R. Omar Chavez-Garcia, Jérôme Guzzi, Luca M. Gambardella, and Alessandro Giusti. Learning Ground Traversability From Simulations. *IEEE Robotics and Automation Letters*, 3(3):1695–1702, 2018. II-A

[7] Timothy H. Chung, Viktor Orekhov, and Angela Maio. Into the Robotic Depths: Analysis and Insights from the DARPA Subterranean Challenge. *Annual Review of Control, Robotics, and Autonomous Systems*, 6(1), 2023. II-A

[8] Gian Erni, Frey Jonas, Miki Takahiro, Matias Mattamala, and Hutter Marco. MEM: Multi-Modal Elevation Mapping for Robotics and Learning. 2023. IV-A

[9] David D. Fan, Kyohei Otsu, Yuki Kubo, Anushri Dixit, Joel Burdick, and Ali-Akbar Agha-Mohammadi. STEP: Stochastic Traversability Evaluation and Planning for Safe Off-road Navigation. In *Robotics: Science and Systems*, 2021. II-A

[10] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Distance Transforms of Sampled Functions. *Theory of Computing*, 8(19):415–428, 2012. IV-B

[11] Jonas Frey, David Hoeller, Shehryar Khattak, and Marco Hutter. Locomotion Policy Guided Traversability Learning using Volumetric Representations of Complex Environments. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022. II-A

[12] Lu Gan, Jessy W. Grizzle, Ryan M. Eustice, and Maani Ghaffari. Energy-Based Legged Robots Terrain Traversability Modeling via Deep Inverse Reinforcement Learning. *IEEE Robotics and Automation Letters*, 7(4): 8807–8814, 2022. II-E

[13] Mateus V. Gasparino, Arun N. Sivakumar, Yixiao Liu, Andres E. B. Velasquez, Vitor A. H. Higuti, John Rogers, Huy Tran, and Girish Chowdhary. WayFAST: Navigation With Predictive Traversability in the Field. *IEEE Robotics and Automation Letters*, 7(4):10651–10658, 2022. I, II-C, III-D

[14] James J Gibson. *The Ecological Approach to Visual Perception*. Boston: Houghton Mifflin, 1979. I

[15] Raia Hadsell, Pierre Sermanet, Jan Ben, Ayse Erkan, Marco Scoffier, Koray Kavukcuoglu, Urs Muller, and Yann LeCun. Learning Long-range Vision for Autonomous Off-road Driving. *Journal of Field Robotics*, 26(2):120–144, 2009. I, II-F

[16] Nicolas Hudson, Fletcher Talbot, Mark Cox, Jason Williams, Thomas Hines, Alex Pitt, Brett Wood, Dennis Frousheger, Katrina Lo Surdo, Thomas Molnar, Ryan Steindl, Matt Wildie, Inkyu Sa, Navinda Kottege, Kazys Stepanas, Emili Hernandez, Gavin Catt, William Docherty, Brendan Tidd, Benjamin Tam, Simon Murrell, Mitchell Bessell, Lauren Hanson, Lachlan Tychsen-Smith, Hajime Suzuki, Leslie Overs, Farid Kendoul, Glenn Wagner, Duncan Palmer, Peter Milani, Matthew O'Brien, Shu Jiang, Shengkang Chen, and Ronald Arkin. Heterogeneous Ground and Air Platforms, Homogeneous Sensing: Team CSIRO Data61's Approach to the DARPA Subterranean Challenge. *Field Robotics*, 2(1):595–636, 2022. II-A

[17] Tianchen Ji, Arun Narenthiran Sivakumar, Girish Chowdhary, and Katherine Driggs-Campbell. Proactive Anomaly Detection for Robot Navigation With Multi-Sensor Fusion. *IEEE Robotics and Automation Letters*, 7(2):4975–4982, 2022. II-D

[18] Gregory Kahn, Pieter Abbeel, and Sergey Levine. BADGR: An Autonomous Self-Supervised Learning-Based Navigation System. *IEEE Robotics and Automation Letters*, 6(2):1312–1319, 2021. II-F

[19] Dongshin Kim, Jie Sun, Sang Min Oh, J.M. Rehg, and A.F. Bobick. Traversability Classification using Unsupervised On-line Visual Learning for Outdoor Robot Navigation. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 518–525, 2006. I, II-C, II-F

[20] Diederick P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*, 2015. III-E3

[21] Honggu Lee, Kiho Kwak, and Sungho Jo. An Incremental Nonparametric Bayesian Clustering-based Traversable Region Detection Method. *Autonomous Robots*, 41(4): 795–810, 2017. II-F, III-B

[22] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. V-C5

[23] Matias Mattamala, Nived Chebrolu, and Maurice Fallon. An Efficient Locally Reactive Controller for Safe Navigation in Visual Teach and Repeat Missions. *IEEE Robotics and Automation Letters*, 7(2):2353–2360, 2022. IV-B

[24] Daniel Maturana, Po-Wei Chou, Masashi Uenoyama, and Sebastian Scherer. Real-time Semantic Mapping for Autonomous Off-Road Navigation. In *International Conference on Field and Service Robotics (FSR)*, pages 335 – 350, 2017. I, II-B

[25] Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning Robust Perceptive Locomotion for Quadrupedal Robots in the Wild. *Science Robotics*, 7(62), 2022. I, IV-B

[26] Takahiro Miki, Lorenz Wellhausen, Ruben Grandia, Fabian Jenelten, Timon Homberger, and Marco Hutter. Elevation Mapping for Locomotion and Navigation using GPU. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 2273–2280, 2022. IV-A, V-B2

[27] Hans Moravec and Alberto Elfes. High Resolution Maps from Wide Angle Sonar. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, volume 2, pages 116–121, 1985. I, II-A

[28] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *International Conference on Neural Information Processing Systems (NeurIPS)*, 2019. V-A

[29] Morgan Quigley, Brian Gerkey, Ken Conley, Josh Faust, Tully Foote, Jeremy Leibs, Eric Berger, Rob Wheeler, and Andrew Ng. ROS: an open-source Robot Operating System. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2009. V-A

[30] Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. Maximum margin planning. In *International Conference on Machine Learning (ICML)*, ICML '06, page 729–736, 2006. II-E

[31] Charles Richter and Nicholas Roy. Safe visual navigation via deep learning and novelty detection. In *Robotics: Science and Systems*, Cambridge, Massachusetts, July 2017. doi: 10.15607/RSS.2017.XIII.064. II-D

[32] Adarsh Jagan Sathyamoorthy, Kasun Weerakoon, Tianrui Guan, Jing Liang, and Dinesh Manocha. Terrapn: Unstructured terrain navigation using online self-supervised learning. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 7197–7204, 2022. II-C, III-D

[33] Fabian Schilling, Xi Chen, John Folkesson, and Patric Jensfelt. Geometric and Visual Terrain Classification for Autonomous Mobile Navigation. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 2678–2684, 2017. II-B, V-C3

[34] Junwon Seo, Taekyung Kim, Kiho Kwak, Jihong Min, and Inwook Shim. Scate: A scalable framework for self- supervised traversability estimation in unstructured environments. *IEEE Robotics and Automation Letters*, 8 (2):888–895, 2023. II-D

[35] Amirreza Shaban, Xiangyun Meng, JoonHo Lee, Byron Boots, and Dieter Fox. Semantic Terrain Classification for Off-Road Autonomous Driving. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 619–629, 2022. II-B

[36] 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. *IEEE Robotics and Automation Letters*, 4(2):1509 – 1516, 2019-04. I, II-C, III-D

[37] Lorenz Wellhausen, René Ranftl, and Marco Hutter. Safe

Robot Navigation Via Multi-Modal Anomaly Detection. *IEEE Robotics and Automation Letters*, 2020. I, II-D

[38] Martin Wermelinger, Péter Fankhauser, Remo Diethelm, Philipp Andreas Krüsi, Roland Siegwart, and Marco Hutter. Navigation Planning for Legged Robots in Challenging Terrain. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016. V-B2

[39] Markus Wulfmeier, Dushyant Rao, Dominic Zeng Wang, Peter Ondruska, and Ingmar Posner. Large-scale cost function learning for path planning using deep inverse reinforcement learning. *The International Journal of Robotics Research*, 36(10):1073–1087, 2017. II-E

[40] Bowen Yang, Lorenz Wellhausen, Takahiro Miki, Ming Liu, and Marco Hutter. Real-time Optimal Navigation Planning Using Learned Motion Costs. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 9283 – 9289, 2021. II-A

[41] Jannik Zürn, Wolfram Burgard, and Abhinav Valada. Self-supervised visual terrain classification from unsupervised acoustic feature learning. *IEEE Transactions on Robotics*, 37(2):466–481, 2021. II-C

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	Fast Traversability Estimation for Wild Visual Navigation
Publication Status	x Published □ Accepted for Publication □ Submitted for Publication □ Unpublished and unsubmitted work written in a manuscript style
Publication Details	Jonas Frey*, Matias Mattamala*, Nived Chebrolu, Cesar Cadena, Maurice Fallon, Marco Hutter, "Fast Traversability Estimation for Wild Visual Navigation," *Robotics: Science and Systems*, 2023. (*equal contribution)

Student Confirmation

Student Name:	Matias Mattamala		
Contribution to the Paper	• Co-developed the main idea • Co-implemented the method, mainly the self-supervision strategy and onboard system integration • Executed closed-loop field experiments • Performed data analysis of field experiments • Discussed and analyzed offline dataset experiments • Wrote half of the sections of the paper		
Signature 		Date	September 8, 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 Matias was the co-lead investigator, implementer and researcher on this project and wrote about half of the paper. This paper was a typical jointly lead collaboration with an equal balance of input from Jonas and Matias. The work was particularly extensive (more than twice Matias' other papers).		
Signature 	Date	September 8, 2023

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

6.2 Additional Remarks

In this section we provide additional notes and clarifications of the previous article in response to the main questions raised by other researchers after publication.

6.2.1 On Positive and Negative Samples

A recurrent comment was that our system was only trained with positive samples collected during the human demonstrations. This raised two important questions about (1) how we obtained negative samples, and (2) the validity of our assumptions and approach to deal with this issue.

Regarding the first point, it is important to recall that while our aim was to enable the robot to traverse areas that are challenging to characterise from geometric information only, we also wanted to avoid collecting data in regions that can be dangerous for the robot, such as cliffs or pits. To tackle this problem, we assumed a *conservative* (or *pessimistic*) strategy that only assigned valid traversability scores to the areas that were traversed in the past, while everything else was assumed untraversable (zero traversability score). This enabled our system to learn traversability via the traversability loss term $\mathcal{L}_{trav}$ in Eq. (8) of the article.

However, this brought up concerns about the validity of this assumption: Is it sensible to assume the terrain to be untraversable if we have no information about it? What happens if we traverse one of those areas in the future? How do we revert this *labelling* when acquiring new positive samples? This is where the reconstruction loss term, $\mathcal{L}_{reco}$ comes into play.

The $\mathcal{L}_{reco}$ (Eq. (3)) was computed using the traversed segments only, effectively motivating learning the distribution of positive samples, which is shown in Fig. 6. The reconstruction loss allowed us to bootstrap a value that expressed the confidence of the traversability score that we assigned to each segment. We used this score within the traversability loss $\mathcal{L}_{trav}$ defined on the paper:

$$\mathcal{L}_{\text{trav}}(\mathbf{f}) = \delta_{\tau_n=0} \sum_n (1 - c(\mathbf{f}_n)) \, \| f^{\theta_t}_{\text{trav}}(\mathbf{f}_n) - 0 \|^2 + \delta_{\tau_n \neq 0} \sum_n \| f^{\theta_t}_{\text{trav}}(\mathbf{f}_n) - \tau_n \|^2,$$

with $\delta_{\tau_n=0}$ indicating the segments that did not have a traversability score, and $\delta_{\tau_n\neq 0}$ indicating the ones that had a traversability score assigned. However, this notation was slightly confusing, as it mixed conceptual and implementation details of this loss.

A more precise definition of the traversability loss $\mathcal{L}_{trav}$ is:

$$\mathcal{L}_{trav}(\mathbf{f}) = \underbrace{\sum_{n\in\mathcal{T}} \|f^{\theta_t}_{\text{trav}}(\mathbf{f}_n) - \tau_n\|^2}_{\substack{\text{Contribution of traversed} \\ (\textit{labelled}) \text{ segments}}} + \underbrace{\sum_{n\in\mathcal{T}^C} (1 - c(\mathbf{f}_n)) \, \|f^{\theta_t}_{\text{trav}}(\mathbf{f}_n) - 0\|^2}_{\text{Contribution of untraversed segments}},$$

with $\mathcal{T}$ the set of segments that were traversed, i.e. have a traversability score τ_n computed from robot sensing data. From this definition it becomes more explicit how each segment contributes to the final loss:

- If the segment n was traversed: it will contribute to the loss using the assigned traversability score: $\mathcal{L}_{trav}(\mathbf{f}_n) = \|f^{\theta_t}_{\text{trav}}(\mathbf{f}_n) - \tau_n\|^2$
- If the segment n was untraversed and it does not resemble a positive sample: its confidence will be low $c(\mathbf{f}_n) \to 0$ and $\mathcal{L}_{trav}(\mathbf{f}_n) \to \|f^{\theta_t}_{\text{trav}}(\mathbf{f}_n) - 0\|^2$
- If the segment n was untraversed but it does like like a positive sample: its confidence $c(\mathbf{f}_n) \to 1$ and $\mathcal{L}_{trav}(\mathbf{f}_n) \to 0$, effectively not contributing to the loss anymore, which enables the learned model to predict traversability scores for those segments closer to the training samples.

6.2.2 Adaptation Speed vs Dataset Size

Study 5 on the paper discussed how fast WVN could learn the traversability of the new environment and the amount of data required for it. This analysis was offline, using the dataset described in Sec. V.

In Table II we presented the total number of training images per dataset: *hilly* consisted of 920 images, *forest* had 991, and *grass* 2050; the 10 % of the datasets was then between 90 and 200 images. In Fig. 13 we showed that beyond 200 steps of training the performance was close to the optimal achievable per dataset regardless of the percentage of data. This raised a question regarding

the behavior at the *data limit* of the system: If the steps are the crucial factor, how much data do we actually need?

Before moving ahead with this hypothesis, it is important to recall that this offline study was executed in an idealistic regime, where we had access to data samples from the full sequence—the dataset was randomised to determine the splits. Hence, it reflects an upper bound on expected performance and adaptation speed, since it is implicitly relying on a more diverse dataset that the one we could expect in an incremental, online learning task. Further, the dataset used for evaluation was human-labelled, which itself can also differ from the performance expected on real platforms.

As a conclusion, further studies would be required to effectively assess the data limit under which the approach still works. This would require to either study them in a more constrained setting with real data (e.g. data amount vs finite number of different terrains) or via large-scale simulation (when it becomes possible to simulate the large diversity of environments and physical factors that affect off-road navigation).

6.2.3 Onboard System Performance

Lastly, we provide additional insights into the implementation, onboard integration, and runtime execution of our system on the Jetson Orin board.

- The system was implemented in pure Python code, using ROS as the communication middleware to handle all the sensor signals.
- WVN executed 2 main threads: a *main* thread implementing callback functions to handle image and proprioceptive signals, as well as a *learning* thread running the training process.
- The *image callback* produced the main runtime overhead, as it was responsible for processing new incoming images, extracting SLIC [2] and DINO [49] features, adding nodes in the mission graph and generating the supervision signals by reprojection, as well as running the forward pass of the most updated

traversability network $f_{\text{trav}}^{\theta_t}$. The predicted traversability was published at 2.5 Hz (400 ms), where most of the time was spent on data conversions between ROS and Pytorch and generating the supervision signals.

- The *learning* thread ran asynchronously, sampling random batches from the most updated mission graph. In practice, the thread ran close to 6 iter/s across the different experiments.

6.3 Discussion

This chapter presented a system to learn robot-specific traversability from human demonstrations. Our approach combined a supervision signal obtained by classical geometric computer vision methods, with the latest advances in self-supervised models. Together, they allowed us to develop a system able to learn online, onboard, and deploy a legged robot in the wild after a few minutes of demonstration.

Our work was not the first to achieve this: vision-based traversability, online learning, and self-supervision were ideas already explored by the teams in the LAGR program from 2004 to 2008. However, as we discussed in Sec. 3.3, these ideas were not further explored after LAGR finished, and the interests of the community shifted to learning-based methods trained with offline data instead.

On the other hand, this chapter did not discuss other alternative approaches to deal with vegetation, namely Homberger et al. [120] and more recently Li et al. [168]. These works aimed to estimate the underlying ground surface in spite of the terrain map artifacts introduced by high grass or foliage. The estimated terrain can be used to determine the traversability of the terrain by geometric methods, as we reviewed in Sec. 3.3. However, such approach would fail to characterise the difficulties that the robot could face due to the vegetation itself, which we aimed to capture by the traversability score. Nevertheless, further investigations are required to study more expressive scores that take into account the full robot state, as well as multi-modal information to leverage past robot experiences via demonstrations, deployments, but also simulations.

From a general perspective, an important challenge in machine learning for robotics is about priors [265]. Priors can reduce the amount of data we need as well as what we need to learn from it. An important contribution of this work was relying on *general* priors provided by large models trained with massive datasets, such as DINO [49]. The priors were encoded as high-dimensional, pixel-wise features that greatly simplified the learning task, also enabling us to solve it onboard.

The outcomes of this approach also pose new questions. Some of them involve accumulating knowledge and learning over time, which is known as *continual learning* [165, 111]. Others relate to the benefits that other large models, aka *foundation models* [31], could offer to robotics. We continue this discussion in Chapter 8.

Lastly, with this work we demonstrated autonomous operation of legged robots in natural environments, operating on terrains that only the largest Unmanned Ground Vehicles (UGVs) could traverse. This opens new opportunities for their application in natural monitoring and forestry applications. In the next chapter I present a full system for this purpose.

7

Autonomous Forest Inventory with Legged Robots

Over the last 3 years, legged robots have become commonly used for inspection and monitoring operations in industrial facilities. Their rapid evolution from experimental platforms accessible products has shifted the research interests of the community, as well as motivating this particular project. The work presented in Chapter 6 motivated us to test legged platforms in challenging natural environments,.

In this last chapter, I present a system for autonomous forest inventory with legged robots. Compared to other approaches such as Terrestrial Laser Scanning (TLS) or drone mapping, legged robots are able to carry heavy measuring payloads such as LiDAR for long under-canopy surveying runs, with minimal soil impact.

I present the idea of an *autonomous legged forester robot*, designing and implementing a complete autonomy system that involves the interaction of state estimation, dense mapping, planning, and locomotion. We experimentally deployed and demonstrated the approach for surveying missions in Finland in the context of the Horizon Europe *DigiForest* project [75], demonstrating the benefits that legged platforms offer for future applications in the wild.

7.1 Contributions

The main contributions of this original chapter include:

- The development of a complete autonomy stack for forest surveying operations.
- A data analysis package that determines forestry-relevant attributes from point cloud data.
- Hardware validation via autonomous deployments with the ANYmal robot in field campaigns in Finland.
- Analysis of the performance and potential of the proposed system for forestry applications.

7.2 Related Work

The development of a legged forester robot combines ideas related to forestry and tree mensuration, as well as real-world deployment of robotic platforms. While we already reviewed some applications of legged platforms in Sec. 3.4 in the context of scientific data acquitision, in this section we will specifically review forestry inventory applications. We will start by presenting some general techniques and tools used in forestry, and then we will present state-of-the-art robotics systems for the same purpose.

7.2.1 Forest Inventory Methods

Forestry is concerned with the preservation and responsible harvesting of forests and woodlands. To achieve this, it involves the mensuration, data collection, monitoring, and management of trees and biodiversity. Modern forestry follows sustainable principles, where environmental, societal, and economic indicators are integrated in an iterative process across the whole forestry operations [248].

A fundamental part of the operations are the Decision Support Systems (DSSs), which enable foresters to make informed decisions to manage the forest by leveraging data and models [249]. The data is fed into the DSS in the form of *forest inventories*,

which summarise the main attributes of the individuals, such as the tree height, species, Diameter at Breast Height (DBH), and canopy volume [35]. These also enable the determination of holistic traits such as the tree density and social structure [7]. By having access to these attributes, the DSS can enable foresters to prevent detrimental environmental effects such as soil compaction and desertification [179].

Henceforth, an key aspect of forestry concerns the programatic measuring and monitoring of forest attributes. Traditionally, tree mensuration has carried out manually, where attributes such as the DBH are measured tree by tree, by using diameter tapes and specialised calipers [114, 35], see Fig. 7.1 (a). Nowadays, forestry instead relies on remote sensing techniques, such as LiDAR [77] and aerial photographs [113], Fig. 7.1 (b). These techniques can be more accurate than the attributes that were formerly acquired by hand, but also they have enabled the measurement of other relevant features such as stem volume and biomass components [169].

Figure 7.1: (a) Traditional forestry techniques rely on manual measurements using diameter tapes to determine attributes such as the DBH. (b) Nowadays, remote sensing techniques such as TLS provide more accurate measurements in a systematic way. Source: DigiForest EU project, Stein am Rhein Field Trial.

The combination of remote sensing tools with unpiloted platforms such as drones and satellites has enabled large-scale forest inventories [183]. Lightweight aerial platforms have been recognised as a breakthrough technology to support spatial ecology [9]. This has been demonstrated by several applications combining aircrafts and LiDAR sensing, enabling the accurate measuring and modeling of forest

traits [16] and canopy attributes [207]. However, these methods are incapable to provide closed-range, under-canopy information to the DSSs, limiting the level of decision making.

Ground-level measurements are obtained manually, or by using TLS systems [169]. While they ensure high-precision measurement (up to millimetre scale), they still require specialised teams to operate and move the tripod devices in the forest. Robotics technologies offer promising, and potentially cost effective, solutions to this problem, which we review in the following section.

7.2.2 Robotics-enabled Forestry

Robotics has the potential to scale up forest mensuration, by offering technologies and platforms to automate this task. Technologies such as SLAM [261] can accelerate ground-based forest mensuration by integrating LiDAR scans from a sensing payload, carried around by a moving human surveyor or mobile robot. The produced maps can be post-processed to extract the desired attributes, such as the tree positions and DBH, and to produce forest inventories [196, 267, 275].

The incremental nature of SLAM and the data acquisition process also suggests that inventories can be constructed along with the map. This idea has been recently explored in the works of Chen et al. [53] and Proudman et al. [220], demonstrating real-time tree detection and DBH estimation using small-scale quadcopters and handheld devices, respectively.

On the other hand, robotic platforms can support surveying tasks by freeing humans from collecting the data themselves, which in turn can focus on supervising the autonomous survey missions [9]. This has been demonstrated in the context of aerial surveying, where autonomous fixed-wing aircrafts [152], small-scale helicopters [142] and more recently hexacopter platforms [8] have been used for over-canopy data collection. As these platforms fly over the trees, most of the robotic challenges concern achieving maximum area coverage given energy [255] and time constraints [244].

Under-canopy data collection has shown more challenging, as it requires robots to operate in cluttered, unstructures spaces. For aerial platforms, this requires the use less performant drones compared to the ones that can fly over canopy—with smaller payloads and battery size [311]. Teleoperated small-scale drones have been used for more than ten years for surveying [170, 128] though autonomous operation has been only demonstrated in recent years for forest inventories [53, 175] and DNA sampling [18].

Ground platforms are able to provide under-canopy measurements while carrying heavier sensing payloads, enabling longer surveys. Wheeled and tracked robot platforms have previously been used for forest data collection [196, 216, 271] but an important application has been on the harvesting process [235, 132, 133]. However, these harvesting machines have larger footprint and mass, consequently contributing to soil compaction damage due to trampling [91, 44], which has motivated policies to favour lighter equipment and continuous cover forestry [62].

The recent advances in hardware and control of legged robots makes them an compelling alternative for ground-level surveying. As we have discussed earlier in this thesis, over the last five years there has been a widespread development of legged hardware in both academia and industry. In parallel, the robustness and versatility demonstrated by learning-based legged locomotion controllers has enabled the deployment of these platforms on harsh, unstructured terrain [197]. In Sec. 3.4 we discussed applications in which legged robots were used for teleoperated soil sampling [294] as well as wildlife recording [193, 192]. Autonomous operation in natural environments has also been demonstrated for rescue operations [274], load carrying tasks [296], measurement of aeolian processes [221, 223], and planetary analog exploration [14]. Forestry applications have remain unexplored, though their potential advantages have been discussed by some authors in the past [269], such as carrying payloads for a longer period, or lower soil impact compared to other ground-based platforms.

In the following sections we present an *autonomous legged forester robot*—a legged robotic solution that autonomously collects forestry data to feed the DSS.

We present the system design, implementation, and early results for autonomous deployments executed in Finnish forests.

7.3 Autonomous Legged Forester Robot

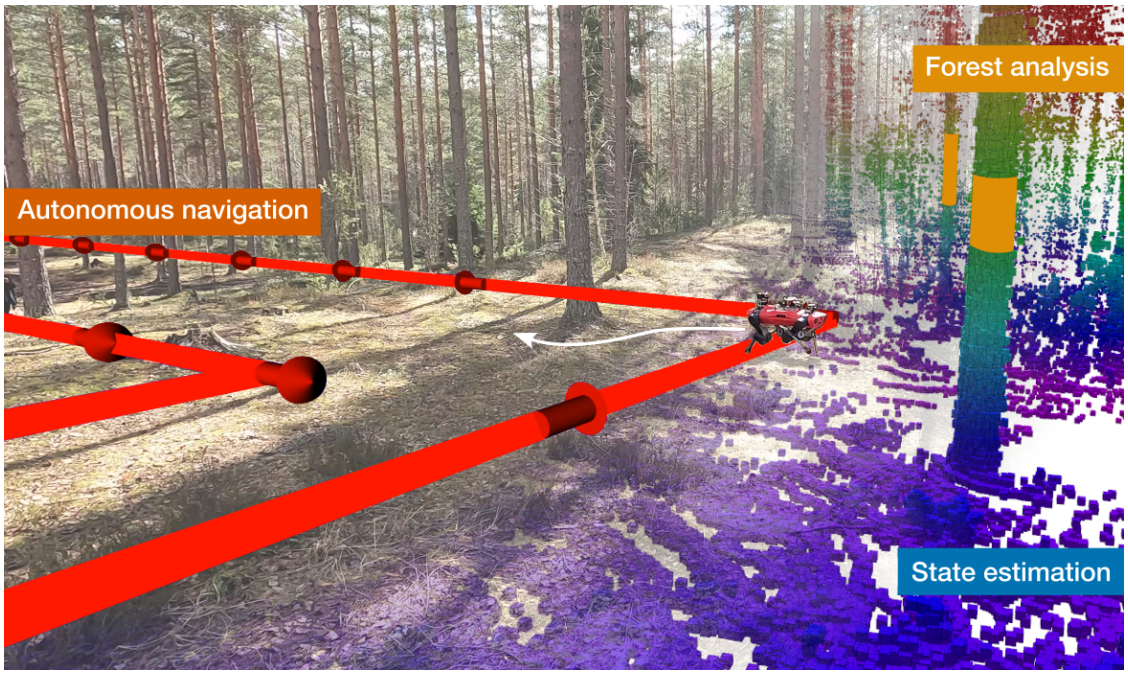

Figure 7.2: The autonomous legged forester robot navigates the forest autonomously, following the proposed plan indicated by the red lines and arrows. This enables it to collect 3D data, which is analysed to detect trees and to determine other forest attributes to build an inventory. The inventory feeds the DSS used in future forestry operations.

Our vision for the autonomous legged forester robot is presented in Fig. 7.2. A human operator defines an unknown patch of the forest they are interested in surveying, which is the input for the mission planning system. The mission planner proposes a rough plan for the robot to navigate the environment, given some predefined constraints, e.g. maximising mapping density and consistency (via loop closures).

The plan is closely executed by the robot by relying on a reactive local planning approach to negotiate obstacles and other unknowns of the environment. As the local planner enables safe navigation, it reports to the mission planner any discrepancies between the proposed plan and the actual environment, enabling

dynamic global replanning of the survey mission. The main data collected by the legged forester are 3D scans, which are consistently integrated by the SLAM system running onboard. They are used for online forest analysis to determine attributes such as the DBH of the surrounding forests, and reported in real-time to the operator, providing live feedback on the mission.

The mission is completed with the robot returning to the starting position. The full dataset can then be recovered, analysed, and refined in post-processing to generate the forest inventory for the DSS.

7.3.1 System overview

The main modules and signals of our system are presented in Fig. 7.3, which are explained in the following sections.

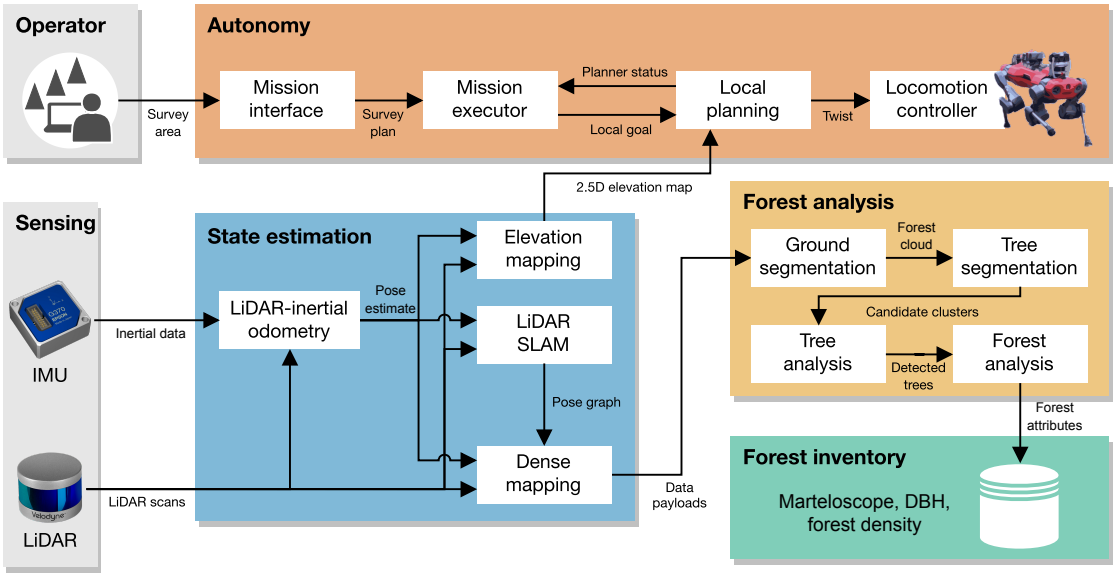

Figure 7.3: System overview of the autonomous legged forester robot. The **autonomy system** executes the mission from a survey reference given by a human operator. **State estimation** provides a consistent trajectory and representation, as well as dense clouds which are used for further analysis. **Forest analysis** implements a pipeline for tree segmentation and characterisation from point clouds. The main output of the system is a forest inventory with the main attributes fof the surveyed forest.

7.3.2 State Estimation

Our state estimation solution involves four modules: odometry estimation, trajectory estimation, large-scale mapping and local terrain mapping, which provide representations for forest analysis as well as navigation.

Odometry For odometry we use a LiDAR-inertial factor graphs-based odometry solution, VILENS [295], which provides high-frequency pose and velocity estimates (∼200 Hz) in a fixed inertial frame. The system performs sensor fusion of relative pose displacements from LiDAR using the Iterative Closest Point (ICP) algorithm [26] with preintegrated IMU measurements [90].

Trajectory and Map Estimation The high-frequency pose and the incoming LiDAR clouds feed a pose-graph SLAM system, VILENS-SLAM [227], which incrementally builds a sparse pose graph as a lightweight but consistent representation of the environment. Each graph node stores a LiDAR scan, and the edge factors correspond to incremental odometry estimates from VILENS.

Additionally, we detect loop closures from geometric candidates obtained within 10–15 m from the robot. The candidates are geometrically verified using ICP, and the passing candidates are added as additional factors in the pose graph. The full graph is optimised incremantally using the iSAM2 algorithm [136].

Dense Mapping As the SLAM system only stores sparse LiDAR scans within the pose graph, this map is not dense enough for further forestry analysis. We implemented a local sub-mapping module to accumulate all the incoming LiDAR scans and generate dense scans in sensor frame, which we denote *data payloads*.

The payloads are published at a slower rate (∼0.1 Hz) for online analysis, and they are also stored to disk for offline post-processing. Further, as they are also attached to the pose graph, they are accordingly moved along upon loop closure corrections, ensuring a consistent dense reconstruction.

Local Terrain Mapping In addition to the large-scale mapping modules, we also rely on a local terrain mapping module for local planning. We used a 2.5D grid-based representation provided by a terrain mapping pipeline [198], which was executed on a Graphics Processing Unit (GPU) (Jetson Xavier) available on the robot.

7.3.3 Autonomy System

Our autonomy system is multi-leveled, spanning from the human operator interface (Level 3) down to the low-level locomotion control (Level 0).

Mission Planning (Level 3) The user interface represents the higher level of planning of our system, as it directly interacts with the human operator. We implemented it as a graphical user interface (GUI) on RViz using interactive markers [107]. Given the initial pose of the robot, the GUI allows the operator to define a survey area and initialise a survey plan, given by a boustrophedon decomposition, also known as lawn mower pattern [57].

The survey plan is concretely defined as a sequence of equally-spaced 6 DoF goals, which enforce some minimum constraints to ensure that the SLAM system is able to find loop closures during operation and keep the trajectory and the map consistent.

The mission planner also provides an interface for the operator to interrupt the mission at any time, which triggers a safe stopping behaviour through the lower levels. It also enables the operator to modify and resume the plan from any specified goal.

Mission Executor (Level 2) The mission executor processes the survey plan and executes it by interacting with the local planner. In particular, it schedules the goals sent to the local planner, and it also processes any feedback obtained by it. For example, if the local planner reports a goal as unreachable, the mission executor chooses a new goal based on the survey plan state.

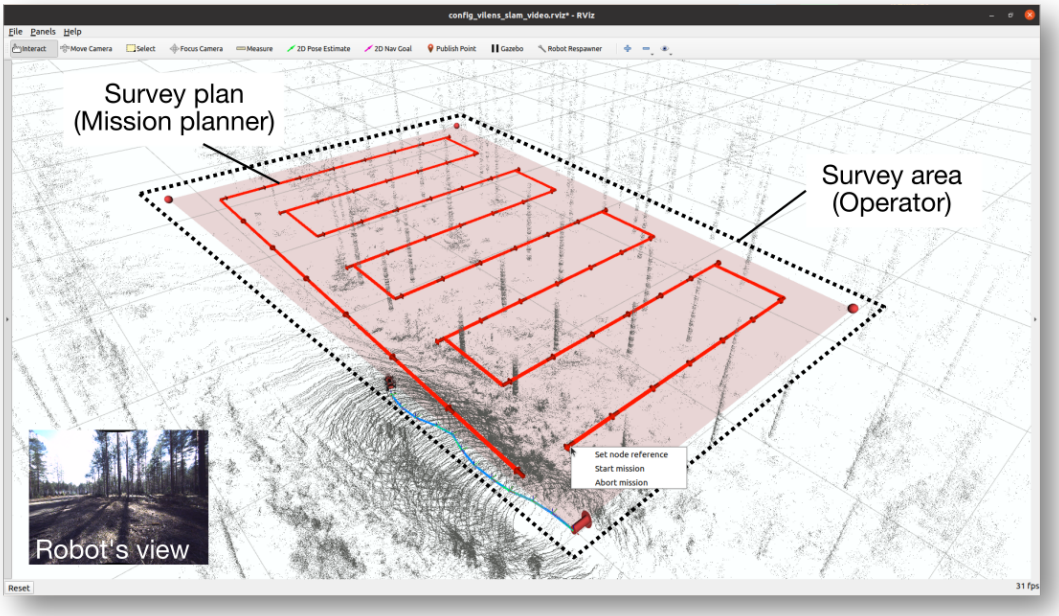

Figure 7.4: Mission Planning GUI for the operator, implemented on RViz using interactive markers. The survey area is specified by the operator using the GUI, while the survey plan is automatically proposed by the mission planner.

Local Planning (Level 1) The local planning aims to reach the proposed goals of the plan and check its feasibility given the state of the local environment encoded in the terrain map. For this, a traversability estimation system processes the local terrain map and extracts geometric features to score traversable terrain. The traversability score s_{trav} is specifically designed to enable navigation in the forest in spite of branches and twigs.

The traversability representation is processed by the reactive local planner presented in Chapter 5 [189] that combines different vector fields for navigation. We extended the approach to compute a GDF on a continuous cost-to-go $c_{\text{to-go}}$ obtained by the heuristic:

$$c_{\text{to-go}} = w_{\text{trav}} \ (1.0 - s_{\text{trav}}) + w_{\text{unkn}} \ s_{\text{unkn}} \tag{7.1}$$

where s_{unkn} is a fixed score assigned to empty cells in the terrain map, and w_{trav}, w_{unkn} are manually defined weights to leverage the contribution of the traversability and unknown space scores.

Similarly to the GDF presented in Chapter 5, this continuous version also has a single minimum at the local goal. However, the continuous cost is able to characterise the difficulty of traversing the terrain at a finer level.

The local planner is able to detect when the robot was stuck by measuring the expected progress to reach the goal. If it determined that the robot is unable to approach it, for example because of dense foliage, the local planner reports this situation up one level up to the mission executor.

Locomotion Controller (Level 0) The output of the local planner is a 3 DoF (planar translation and rotation) velocity command, which is executed by a low-level reinforcement learning-based locomotion controller [197]. We used a pre-trained policy and we did not implement any specific modifications to the architecture or training environment to operate in the forest.

7.3.4 Forest Analysis

To analyse the point clouds and extract forest attributes, we implemented a pipeline that enables the modular processing of gravity-aligned 3D point clouds with normals. The pipeline can be integrated as an online module that processes the data payloads generated by the dense mapping system, or as an offline tool to process larger point clouds collected by our legged forester.

Preprocessing

Before performing any operations on the cloud, we apply preprocessing steps:

- We crop the cloud to a fixed area of $80 \times 80\,\mathrm{m}$, as we observed that beyond that range measurements are very sparse.
- We eliminate spurious points by applying a radius-based outlier filter.
- We apply voxel filtering to have consistent point density in the following stages.
- If the cloud has any intensity information, we also remove points with intensity lower than a fixed threshold, as they often correspond to foliage.

The output of the preprocessing module is a cropped cloud. Fig. 7.5 illustrates the original payload cloud in blue ■, while the cropped version is displayed in red ■.

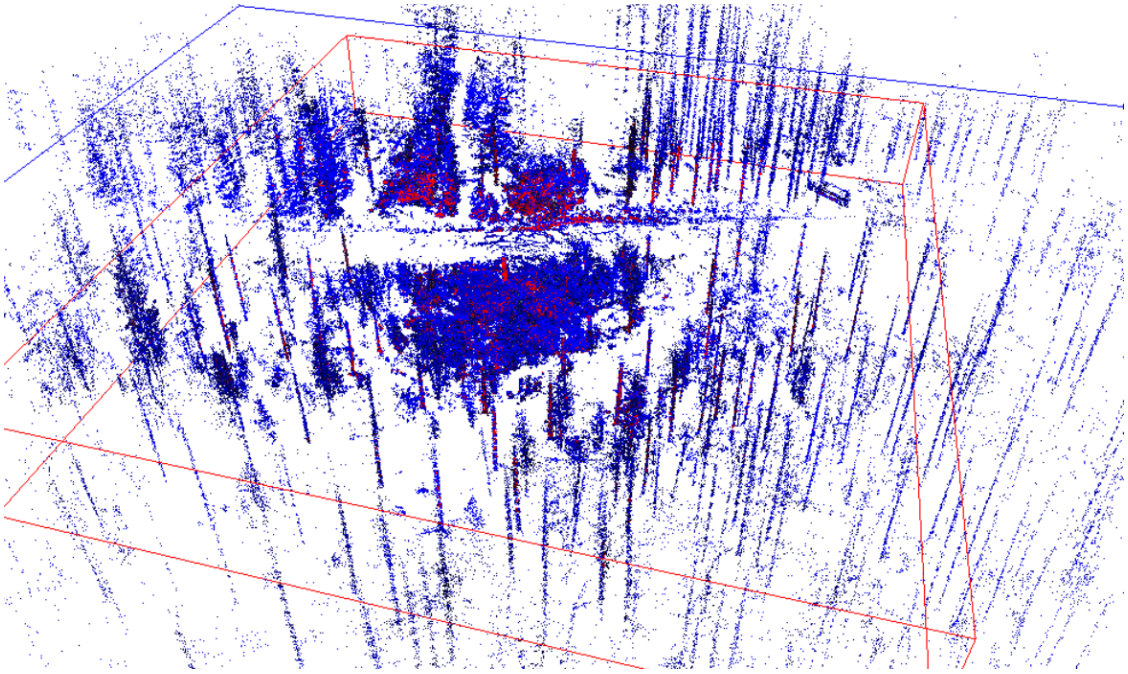

Figure 7.5: The *preprocessing* step crops the input cloud (blue ■) and it applies outlier filtering to obtained a filtered cloud for the next stages (red ■)

Ground Segmentation

The next step of the forest analysis pipeline aims to identify the points that belong to the ground surface, segmenting them from the points that should correspond to potential trees and other objects in the ground canopy.

We first classify the input point cloud using the normal information: we consider the points as part of the ground if $n_z > \tau_{\text{ground}}$, where n_z is the z-axis component of the normals of the cloud point $\mathbf{p}$ and τ_{ground} is a threshold, we used 0.9 in our experiments. Then, we arrange the cloud in a fixed-size 2D grid aligned with the cloud's center, and fit a local plane for each $4 \times 4\,\text{m}$ grid cell. The points satisfying the local plane model are accepted as part of the ground and the rest are discarded.

Fig. 7.6 shows an example of the output of this module for a cropped data payload using the proposed algorithm, though more sophisticated techniques such as cloth simulation-based filter could also be used [308]. The segmented ground is

shown in blue ■, while points that did not pass the normals check to determine the ground cloud are considered part of the potential *forest* cloud, shown in red ■.

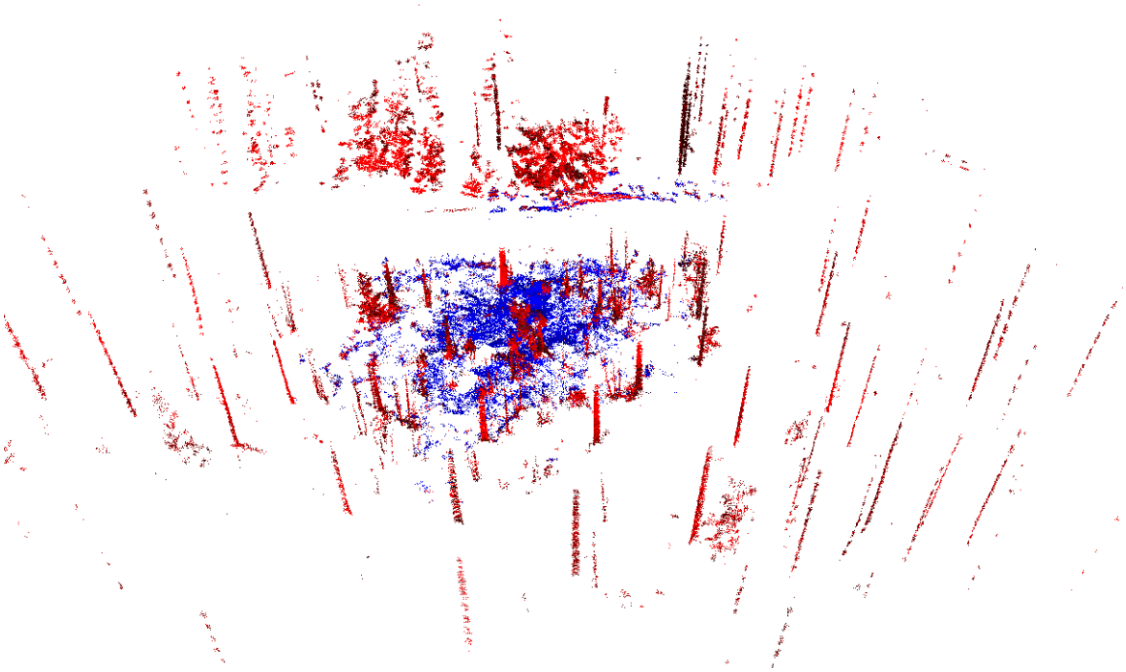

Figure 7.6: *Ground segmentation* splits the cloud into a ground (blue ■) and forest (red ■) clouds. The forest clouds are used in the next stages of the pipeline. Please note the sparsity of the ground points due to the limited vertical FoV of the LiDAR, as well as the sensor's location on the legged robot, around 60 cm from the ground.

Tree Segmentation

The forest cloud is segmented to find tree candidates. We use a geometric segmentation approach based on 3D clustering, specifically the HDBSCAN [45] algorithm. Compared to other approaches such as K-means [185], Euclidean clustering [239] and DBSCAN [84] that require to set a fixed distance threshold to aggregate the clusters, HDBSCAN produced consistent results regardless of the sparsity of the input point cloud as it automatically performs the clustering at different thresholds.

Once the initial clustering is done, small candidate clusters are discarded, as they usually correspond to either noise or small isolated bushes in the forest. Large clusters were usually associated to coarse segmentation patches, where two or

more trees were asigned to the same cluster. In those cases, we re-cluster them to partition them into individual tree candidates.

The output of the segmentation process is illustrated in Fig. 7.7 (c), where the candidate clusters are indicated by bounding boxes. The solid coloured bases denote the bounding boxes' size, which is used to discard some candidates (shown in black).

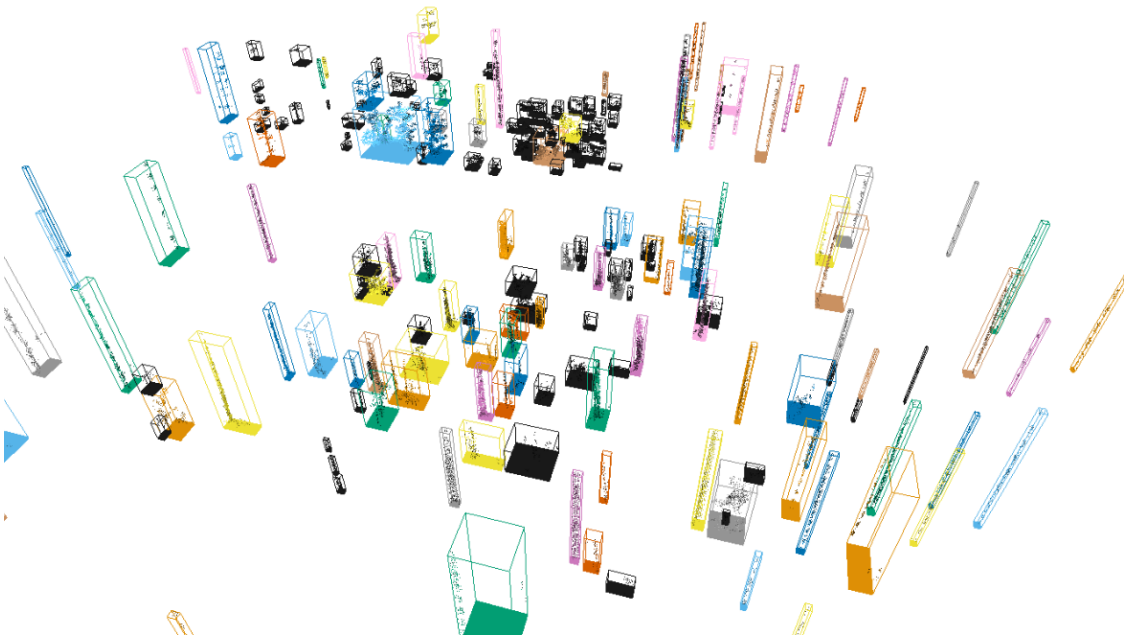

Figure 7.7: *Tree segmentation* applies 3D clustering algorithms to obtain sub clouds corresponding to tree candidates, shown by the coloured bounding boxes.

Individual Tree Analysis

Each candidate cluster is analyzed independently to extract tree attributes, specifically the DBH. For this, we fit a cylinder model [81] to estimate the DBH of each stem, as it has been done in previous work [275, 53]. The cylinder model f is parametrised by its center $\mathbf{c}$, its principal axis given by the unit vector $\mathbf{w}$, and radius r:

$$f(\mathbf{x}) = (\mathbf{x} - \mathbf{c})^\top \left(\mathbf{I} - \mathbf{w}^\top \mathbf{w}\right) (\mathbf{x} - \mathbf{c}) - r^2 \tag{7.2}$$

The model is used to define a residual $\mathbf{r}(\mathbf{x}, \mathbf{c}, \mathbf{w}, r)$ in a NLS formulation. Given the cluster cloud points $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, with N the size of the cloud, we can

estimate the cylinder parameters by solving Eq. (7.3).

$$\arg\min_{\mathbf{c},\mathbf{w},r} \sum_{i}^{N} \rho\left(\mathbf{r}_i\left(\mathbf{x},\mathbf{c},\mathbf{w},r\right)\right) \tag{7.3}$$

where ρ is a *robust loss* or *kernel* function to downweigh the effect of outliers. In our experiments we used the Geman-McClure loss [104], though other options could be considered. See Barron [23] for more discussion about robust loss functions.

We optimised the problem using the L-BFGS-B algorithm [312], which enabled us to enforce bounds on the parameters and guide the optimisation to valid solutions. In particular, we bounded the direction of $\mathbf{w}$ to be close to the gravity direction. The tree radii were also constrained in the optimisation to be within sensible ranges for the trees (0.1–0.5 m). The results of the fitted cylinders are shown in Fig. 7.8.

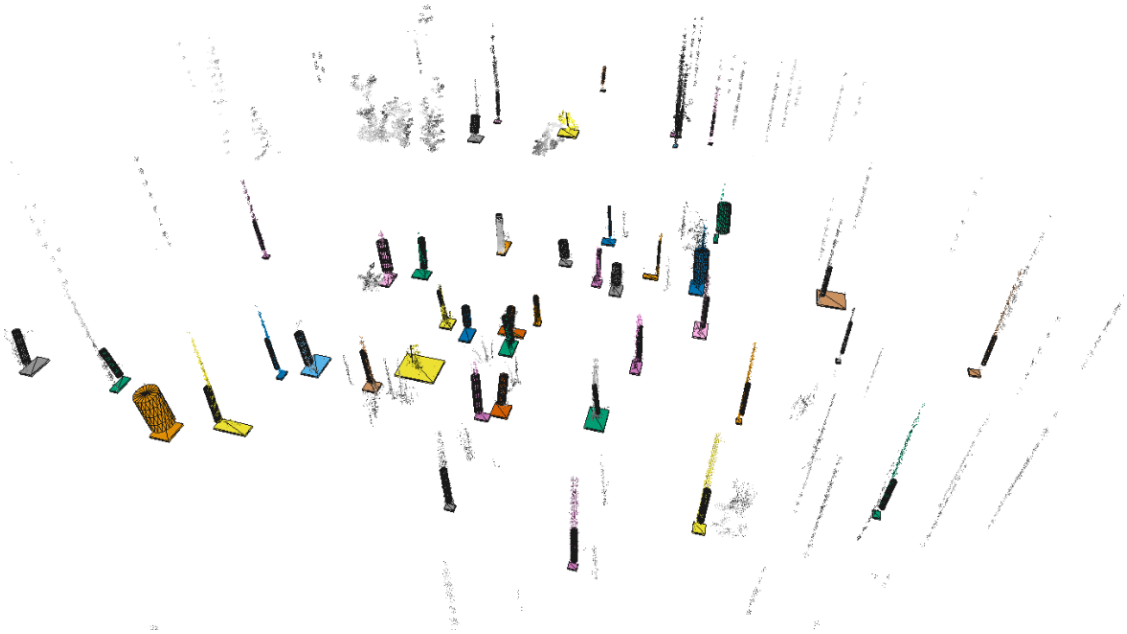

Figure 7.8: *Tree analysis* tries to fit a cylinder model to each tree candidate cloud to estimate the DBH.

Forest Analysis

Finally, we use the estimated positions in the local frame as well as the DBH to generate a *marteloscope* [157], which is a standard tool in forestry to represent trees within patches of 0.4–1.0 ha. An example of this output is shown in Fig. 7.9.

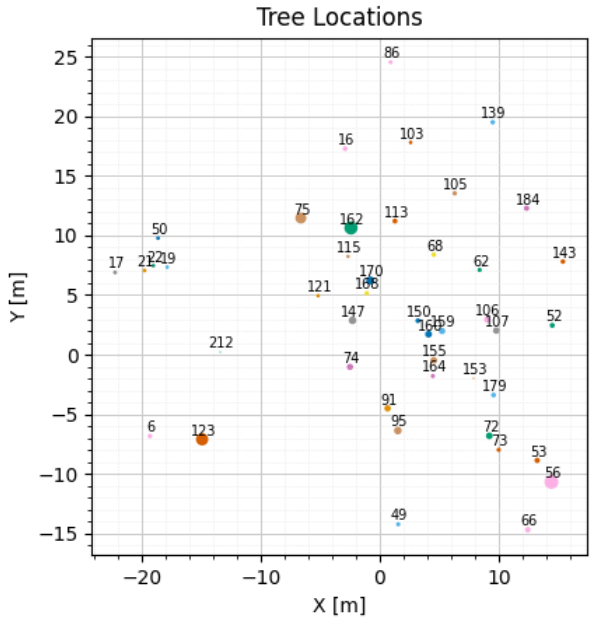

Figure 7.9: A preliminary example of the marteloscope produced by the forest analysis step. This output is directly generated from the valid trees detected by the tree analysis module, where the diameter of the circles is proportional to the DBH estimated from the cylinder model.

Additionally, we also extract other attributes such as the area covered and tree density, which are also included as part of the forest inventory.

7.4 Results

7.4.1 Campaign Overview

In May 2023, a field campaign was carried out for the DigiForest project in Evo, Finland. A Nordic coniferous (evergreen) forest, part of the Häme University of Applied Sciences (HAMK) Forestry school, was used for the experiments. We tested the described system during 3 days of autonomous experiments.

Fig. 7.10 shows an aerial view of the testing areas for the different deployments, while Tab. 7.1 summarises the missions, reporting the location, time, and area covered during each deployment.

The experiments were executed with an ANYbotics' ANYmal C platform, equipped with a Velodyne VLP-16 as the main sensor for forest mapping. Online

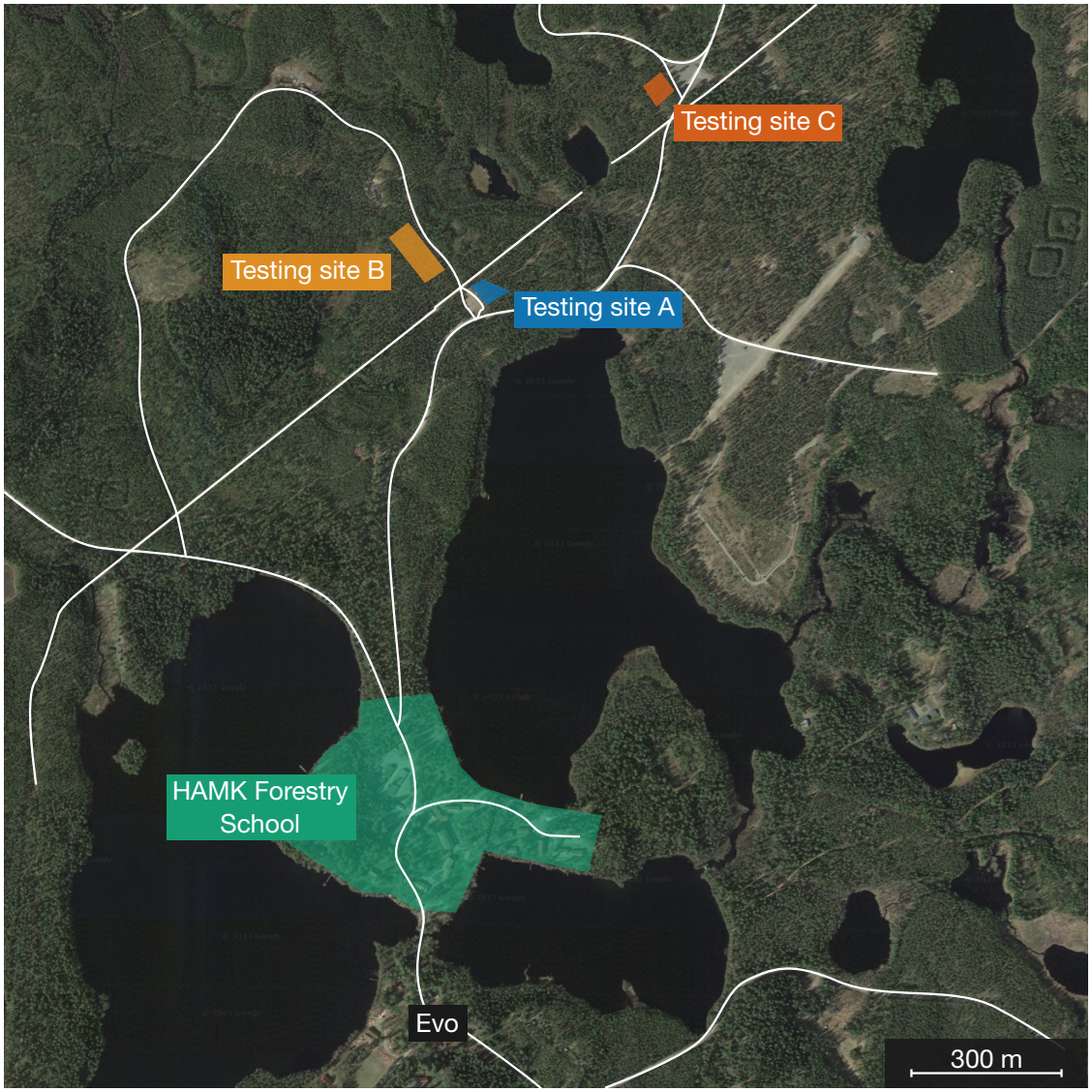

Figure 7.10: Main sites of the Evo campaign. The HAMK Forestry school is located at 61.196, 25.107 (latitude-longitude, in decimal degrees). The autonomous deployments were executed in the testing sites A, B, and C. Aerial map obtained from Google Earth.

during the robot deployments we did not use the odometry system previously described, but another implementation based on CompSLAM [145], a state estimation system relying on complementary sources such as leg [29] and LiDAR [307] odometry. CompSLAM was not properly tuned for the environments we tested on, consequently affecting the closed-loop performance of the system. However, all the mapping results we report in this section are post-processed using the raw sensor logs (rosbags) and the state estimation systems described in Sec. 7.3.2.

Mission	Date	Site	Path length	Surveyed area	Duration
M1	2023-05-03	Site B	263.7 m	3279.4 m^2 / 0.31 ha	563.2 s
M2	2023-05-04	Site B	239.0 m	3125.7 m^2 / 0.31 ha	414.4 s
M3	2023-05-04	Site B	260.5 m	3551.4 m^2 / 0.35 ha	720.0 s
M4	2023-05-04	Site C	300.4 m	3589.4 m^2 / 0.35 ha	978.3 s
M5	2023-05-05	Site A	600.0 m	5743.6 m^2 / 0.57 ha	1250.6 s

Table 7.1: Summary of the missions executed in the campaign.

For each mission we started the robot from a fixed position, where the *robot operator* was located. As the deployments involved autonomous exploration of unseen forest patches, a *safety operator* walked along with the robot at a safe distance, carrying the robot's remote controller. The safety operator intervened in situations that could not be handled by the autonomy system or remotely by the robot operator. We catalogued all the interventions to determine the duration when the robot was not operating autonomously.

7.4.2 Autonomous Deployments

In this section we discuss the five autonomous deployments, focusing on the performance of the autonomy system obtained across the experiments. We report illustrations of the maps generated by our state estimation system, as well as quantitative analysis of the missions in terms of coverage, speed, autonomy and potential soil impact. Fig. 7.11 illustrates some examples of the robot autonomously navigating on each mission.

Missions Overview

We first qualitatively describe the different missions, situations faced on each deployment, and the output of the mapping process.

Mission M1 The first mission was executed on site B, with a duration of 9 min. The produced map is shown in Fig. 7.12.

Figure 7.11: Illustrative examples of each of the five missions, from M1 to M5. We started in M1 and M2 with a clear forest patch on site B. M3 was executed on the same site but different patch, while experiencing light snow. Unfortunately, in this mission the robot got trapped, as shown on the second row. M4 was executed the same day, after recovering the robot, on site C. M5 was the last and largest mission, executed the last day on site A.

Mission M2 This mission was also executed on the same area within site B, to validate the previous mission. In contrast to M1, it took 7 min and it ran fully autonomously without safety interventions. The output map is shown in Fig. 7.13.

Mission M3 The mission was also executed within site B, but in a different forest patch. In this experiment, the mission was interrupted due to the robot getting trapped in a damp area. After close-up inspection, it was confirmed that the state was unrecoverable: the legs had penetrated the boggy surface down to the knee shanks, about 30 cm (see Fig. 7.11, M3). In Fig. 7.14 we can observe the

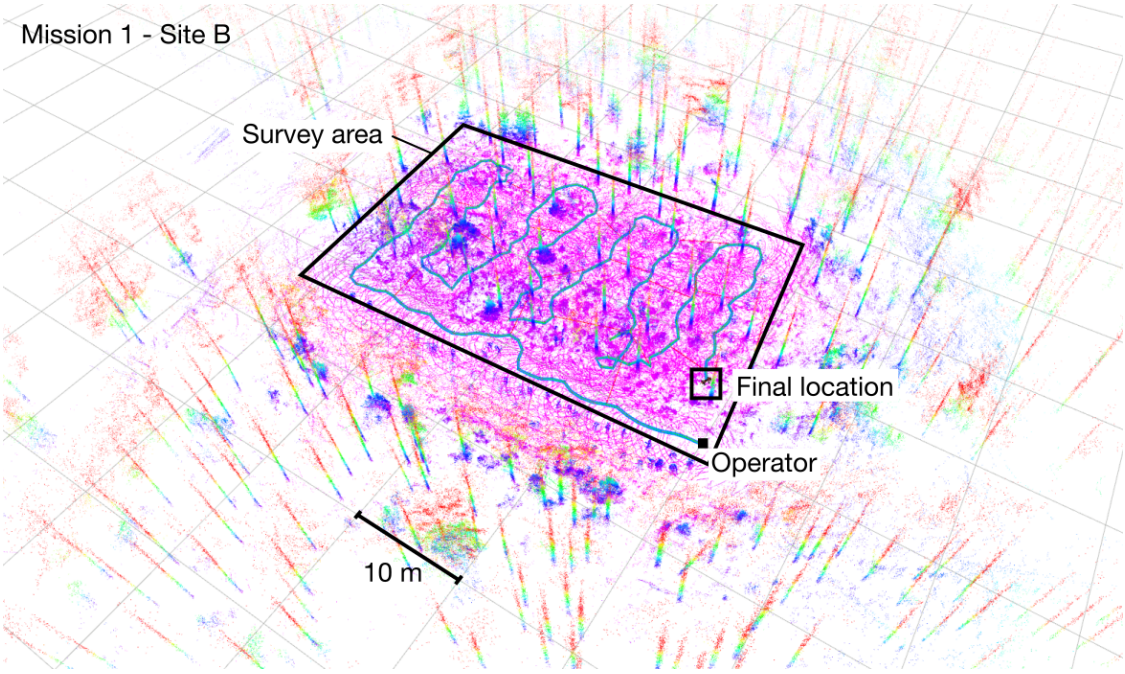

Figure 7.12: Map and trajectory obtained in Mission 1, executed on site B. This was the first large-scale deployment of the campaign.

final location of the robot after interrupting the mission.

In Sec. 7.5 we discuss the limitations of the current autonomy solution to face these situations, and we share some potential solutions to improve the system's reliability in future deployments.

Mission M4 It was executed on site C, as shown in Fig. 7.10, in a forest patch located between two roads. The mission was almost entirely autonomous, requiring some interventions in the denser areas of the forest, which we discuss in Sec. 7.4.2. The trajectory is shown in Fig. 7.15.

Mission M5 The last mission was executed on site A, next to a wide open area used by other equipment during the DigiForest trials. In this experiment we noted severe drift along the z rotation axis (yaw), producing an inconsistent coverage of the intended survey area, as well as problems for the robot to return to the operator's location. This is shown in Fig. 7.16. The orientation drift was caused by incorrect settings in the sensor fusion parameters of CompSLAM, which produced

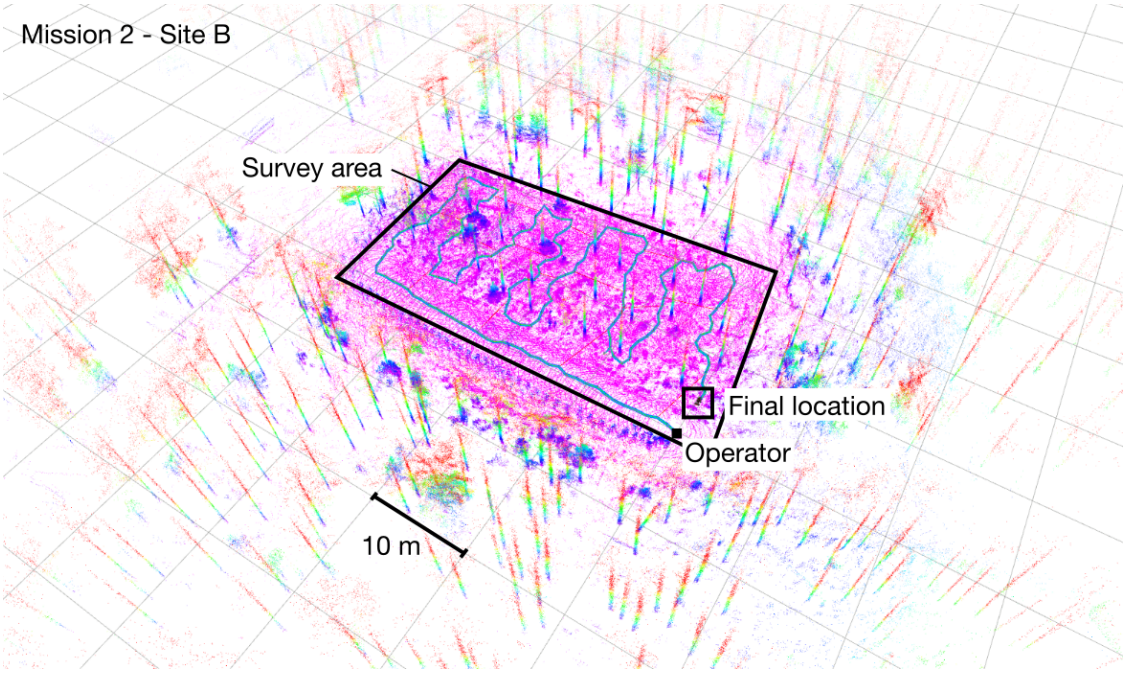

Figure 7.13: Map and trajectory obtained in Mission 2, executed on site B. This mission was fully autonomous.

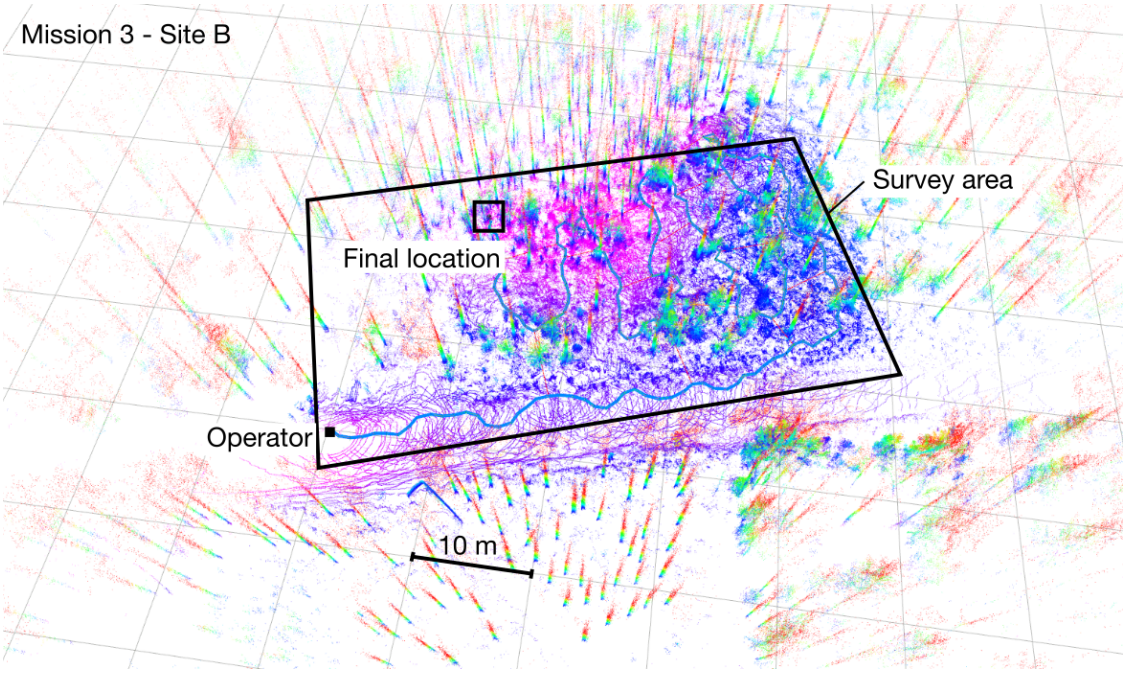

Figure 7.14: Map and trajectory obtained in Mission 3, executed on site B. This mission was interrupted, due to the robot getting stuck on a damp area.

inaccurate fusion of leg odometry with the LiDAR odometry in this environment.

We provide further discussion in Sec. 7.5.

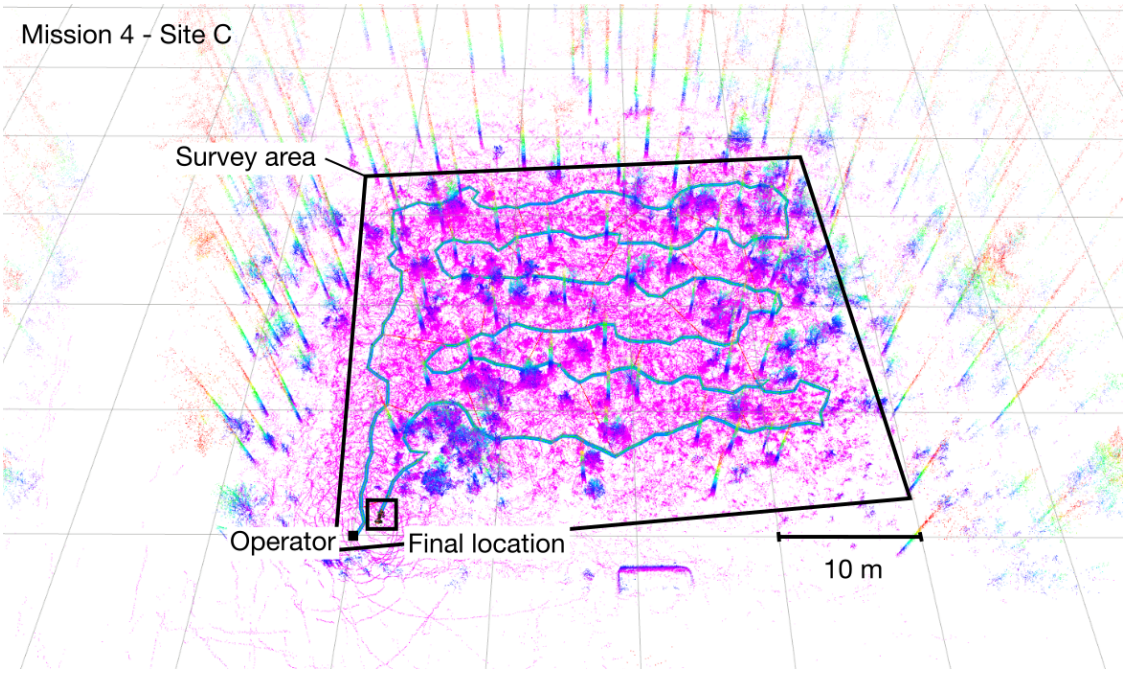

Figure 7.15: Map and trajectory obtained in Mission 4, executed on site C.

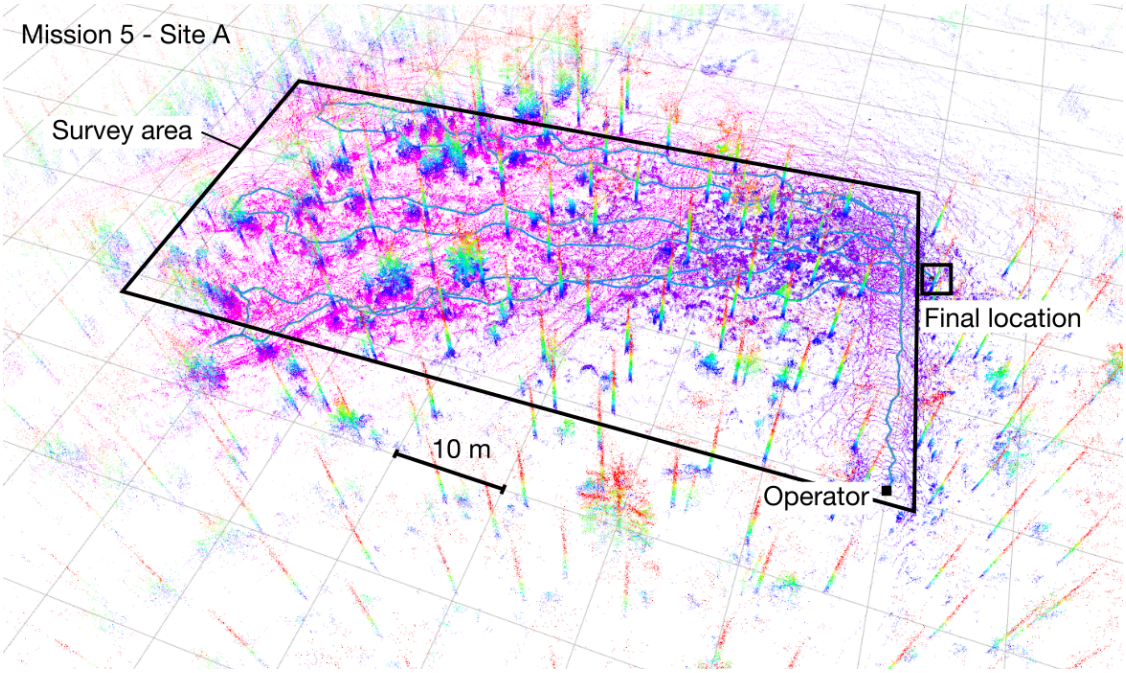

Figure 7.16: Map and trajectory obtained in Mission 5, executed on site A. State estimation errors caused a significant drift that impeded the robot from returning to the operator's location.

Mission Statistics

We report different statistics to provide a better overview of the performance of the system across the missions. In particular, we analyse the speed statistics and

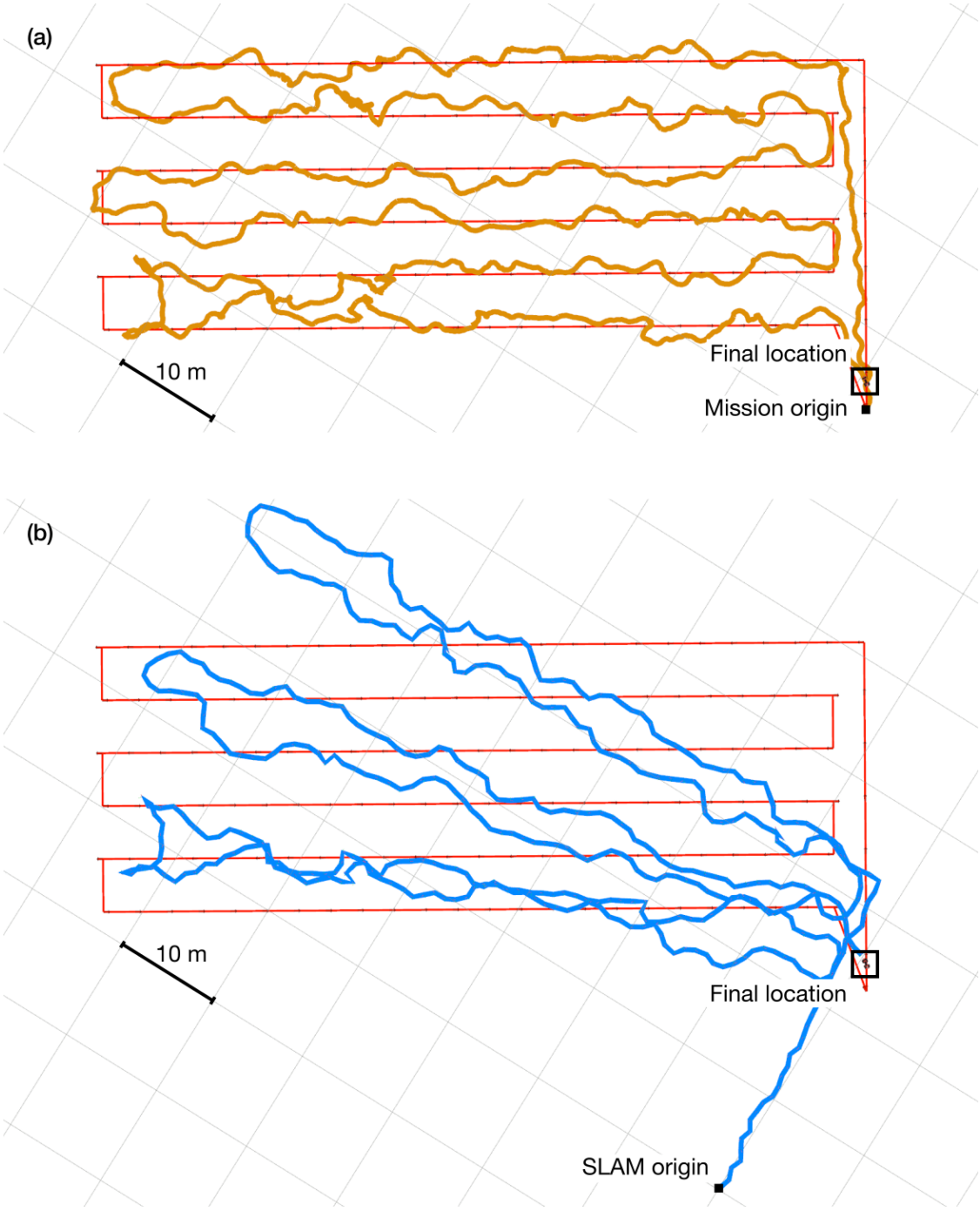

Figure 7.17: Comparison of (a) the internal state estimate (in orange ■), and (b) the post-processed SLAM estimate (in blue ■) for mission M5. Paths are shown for the same final pose and 10 × 10 m, aligned with the SLAM estimate. In this case it becomes evident that the autonomy system performed as intended, closely following the proposed plan (in red ■) and returning to the mission's starting position. However, the huge misalignment with the SLAM estimate evidences the drift that impeded the robot from mapping the intended area completely.

coverage, the autonomy level achieved by the solution, as well as indirect measures of the environmental impact of using legged platforms for these tasks.

Speed and Coverage In Tab. 7.2, we report the average linear and angular speed achieved in all the missions. We provide detailed linear speed profiles comparing the commanded and tracked velocity, in Appendix B.

For most of the experiments we limited the maximum linear speed of the local planner command to 0.8 m/s. From our analyses, we observed that the robot actually walked at 0.45 ± 0.22 m/s on average, which was almost half of the speed limit. The main reason that the robot did not to walk at full speed was due to the local planning strategy, which preferred to avoid bumps and short bushes instead of walking straight.

Combining the average linear speed with the total area covered in the missions we can estimate the mapping rate in ha/h, which we also report in Tab. 7.2. On average, we can conclude that for the current system design and parameters, the legged forester should be able to survey 1.5 ha/h with a single battery charge, given the specifications previously reported in Tab. 2.3.

Mission	Speed (lin)	Speed (ang)	Surveying speed
M1	0.48 ± 0.21 m/s	0.38 ± 0.23 rad/s	2.09 ha/h
M2	0.55 ± 0.24 m/s	0.41 ± 0.23 rad/s	2.77 ha/h
M3	0.40 ± 0.26 m/s	0.33 ± 0.25 rad/s	1.77 ha/h
M4	0.34 ± 0.19 m/s	0.37 ± 0.22 rad/s	1.32 ha/h
M5	0.49 ± 0.21 m/s	0.40 ± 0.24 rad/s	1.65 ha/h

Table 7.2: Robot speed and coverage statistics. On average, the robot walked at 0.45 ± 0.22 m/s in the missions, which suggests that it should be possible to survey 1.5 ha/h.

Autonomy We analysed the mission duration and interactions on each deployment. We defined an *intervention* as every unplanned action executed by the safety operator that overruled the autonomy system (via remote controller) to avoid undesired situations [262]. All the actions within 25 s were considered part of a single intervention. Fig. 7.18 illustrates the results for the five missions.

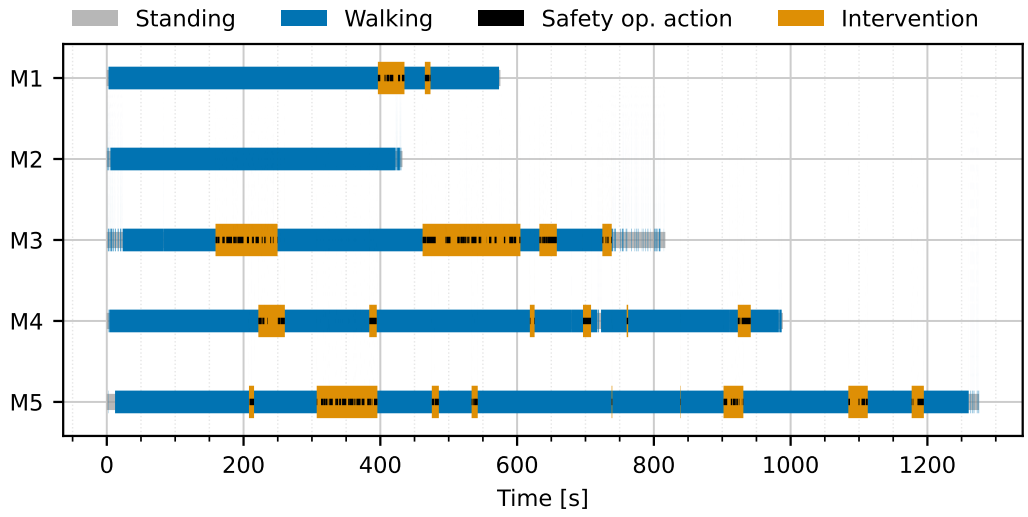

Figure 7.18: Graphical summary of the autonomous performance across all the missions. The safety operator actions (in black ■) were clustered into the same *intervention* (in orange ■) if the time difference between their action was within 25 s.

We also report the main autonomy statistics in Tab. 7.3, including the number of interventions N_{int}, total duration of the interventions t_{int}, as well as the *autonomy percentage* computed over the total mission time t_{int}, as defined by Eq. (7.4):

$$p_{\text{auto}} = 100\,\frac{(t_{tot} - t_{int})}{t_{tot}}\,\% \tag{7.4}$$

In 3 out of 5 experiments we achieved autonomy scores above 90 %, even reaching 100 % in mission M2. In the other two missions we reported worse performance, potentially due to the difficulties of the specific testings sites apart from the specific challenges we observed in missions M3 and M5. Additionally, as a metric of the autonomous performance achieved in the experiments, we also report the distribution of time between interventions in Fig. 7.19. We observe that the distribution mass in centred around 100 s, indicating that the robost operated autonomously for ∼2 min most of the time. This is the main metric we aim to improve in future trials.

As a last comment, it must be noted all the missions were executed with the same person acting as the safety operator. This ensures consistency across the experiments, though the statistics still suffer from an implicit bias from the safety operator's decisions, as they were the only person directly supervising the robot's behaviour during the mission.

Mission	Duration t_{tot}	Interv. N_{int}	Interv. time t_{int}	Autonomy p_{auto}
M1	563.2 s	2	47.14 s	91.81 %
M2	414.4 s	0	0.00 s	100 %
M3*	720.0 s	4	273.28 s	66.54 %
M4	978.3 s	6	89.66 s	90.93 %
M5	1250.6 s	9	192.68 s	84.89 %

Table 7.3: Autonomy statistics. We report as *interventions* every time the safety operator had to take control over the robot to avoid undesired situations (e.g. hitting a tree).

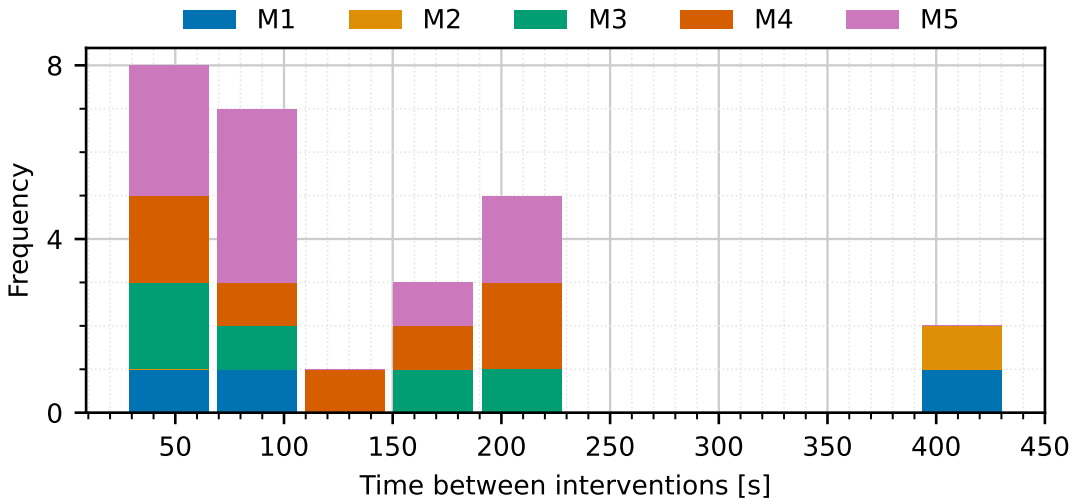

Figure 7.19: Distribution of time between interventions, the statistics for each mission are indicated by different colours.

Soil Impact Lastly, we provide some insights on the potential that a legged robot system has for reducing soil damage. We estimated the number of steps the robot walked on each mission to determine the area A_{steps} that was impacted by the robot during operation, using Eq. (7.5).

$$A_{\text{steps}} = N_{\text{steps}}\, A_{\text{foot}}, = N_{\text{steps}}\, \pi r_{\text{foot}}^2 \tag{7.5}$$

The number of steps N_{steps} was estimated in post-processing by analysing the kinematic gait cycles. We measured height change of the feet positions with respect to the body frame, measured from the robot kinematics. We assumed that the lowest peaks in this curve corresponded to the stance phases of the gait, which counted for all the legs. We assumed a circular foot contact to estimate the

foot area A_{foot}, using the foot radius from the CAD model, $r_{\text{foot}} = 0.03\,\text{m}$. This procedure allowed us to avoid torque-based contact estimation issues due to the lack of contact sensors at the feet [126, 47].

The results for all the missions are reported in Tab. 7.4. We observed that the impacted area was negligible compared to the actual area surveyed, suggesting that the legged robot can be a viable platform for these tasks. However, further investigations are required to determine the effective effects of these operations in the forest, and how legged platforms compare to wheeled platforms or human surveyors.

Mission	Surveyed area	Footsteps	Stepped area	Percentage
M1	$3279.4\,\text{m}^2$	3129	$8.84\,\text{m}^2$	0.0027 %
M2	$3125.7\,\text{m}^2$	2350	$6.64\,\text{m}^2$	0.0021 %
M3	$3551.4\,\text{m}^2$	3679	$10.40\,\text{m}^2$	0.0029 %
M4	$3589.4\,\text{m}^2$	5264	$14.88\,\text{m}^2$	0.0041 %
M5	$5743.6\,\text{m}^2$	6674	$18.87\,\text{m}^2$	0.0033 %

Table 7.4: Soil impact. We provide an estimate for the potential soil impact of the legged robot using the model in Eq. (7.5). In all the missions, the area that was impacted by the feet was negligible compared to the surveyed area.

7.4.3 Forest Analysis

We provide real examples of the output produced by the forest analysis pipeline described in Sec. 7.4.3. The pipeline is still under active development, therefore the results in this section are mostly qualitative, and highlight the next challenges that need to be addressed for its future deployment.

Payload Data

We first show qualitative results when running the forest analysis pipeline on data payloads. These were produced by the dense mapping module by accumulating point clouds from the Velodyne VLP-16 LiDAR. Fig. 7.20 shows some examples of tree detection and cylinder fitting for missions M1 and M5.

The performance of the current pipeline is varied. Some cylinder models closely follow the tree shape (Fig. 7.21 (a)), and even sensible fittings were observed even for cases with partial observations (Fig. 7.21 (b)). When working with payloads,

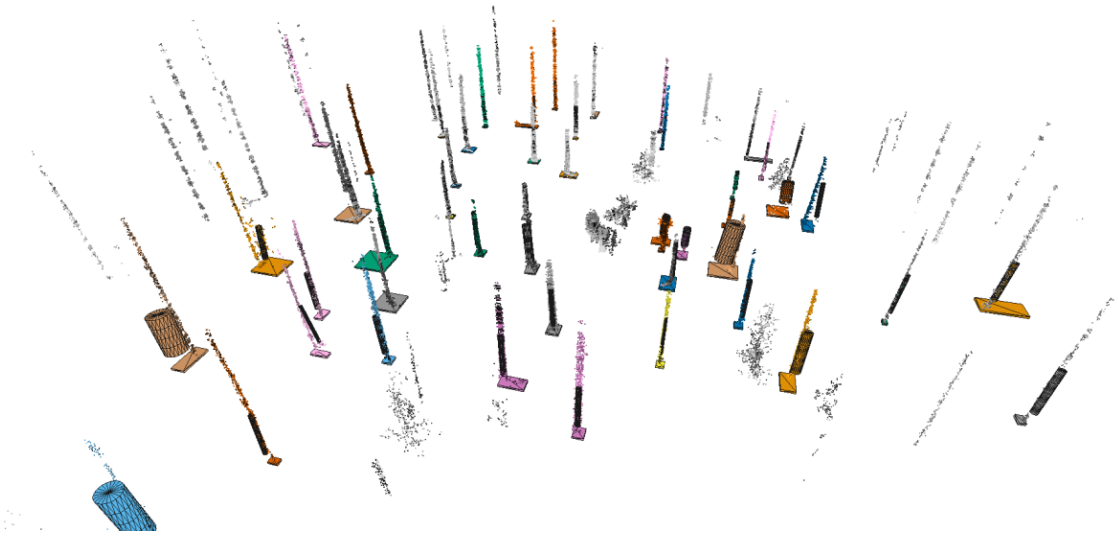

Figure 7.20: Detected trees and cylinder fitting executed on a payload from mission M1.

partial observations are inevitable since the accumulated scans are affected by the robot's direction of motion.

Figure 7.21: Correct cylinder fitting for different payload data. (a) An example of correct detections in one of data payloads. (b) Successful cylinder fitting in spite of the partial observability.

Fig. 7.21 shows cases of incorrect fittings, which are caused by a diversity of factors. Case (a) is produced by candidate clusters with multiple modes that were not distinguishable by DBSCAN in first place, but were small enough to avoid the re-clustering. Case (b) is produced by lack of support points to properly constrain the

cylinder model, usually when many points lie on the same plane and do not capture the curvature. Case (c) shows a bias in the estimated cylinder's position due to undergrowth that does not qualify as an outlier for the Geman-McClure robust loss.

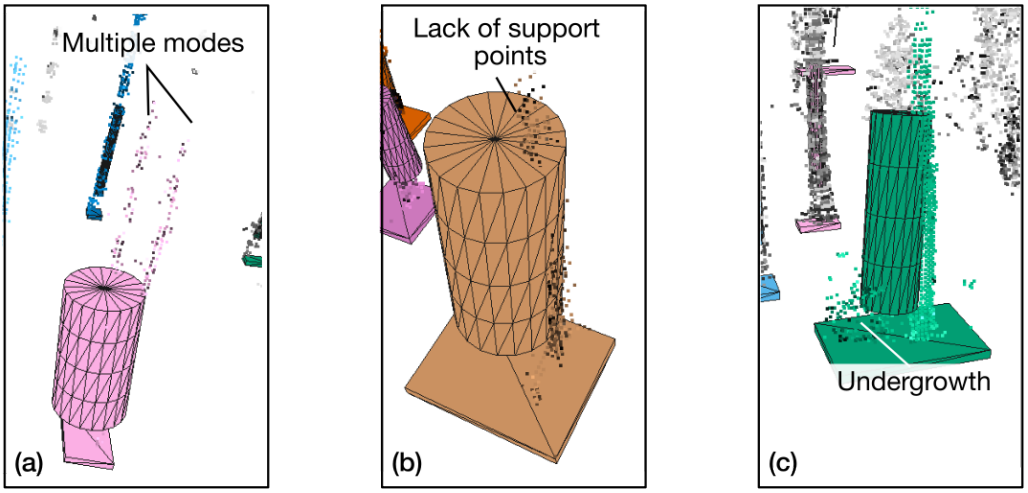

Figure 7.22: Failure cases of incorrect cylinder fitting, due to (a) multiple modes within the candidate cluster, (b) the lack of support points for a cylinder model, or (c) biases due to undergrowth.

In general, we observed that most of the problems are caused by either (1) the lack of data due to the sparsity of the LiDAR beams in spite of the accumulation process, or (2) issues due to the candidate clusters proposed by the tree segmentation step. While the latter requires algorithmic improvements or parameter tuning, the former can be improved by using a different sensor payload, which we discuss in the next section.

Dense Offline Data

To gain further insights on the performance of the pipeline with denser LiDAR clouds, we tested the proposed forestry pipeline with data generated with a Hesai PandarXT-32 unit, with 32 beams and larger vertical FoV.

We manually collected data using an integrated handheld sensing unit. We ran the same state estimation pipeline previously presented to produce payload clouds. Additionally, all the payloads were fused in an offline post-processing procedure, in order to generate a dense point cloud of the full mission. The

cloud was then split into a fixed-size grid, where each cell was $20 \times 20\,\text{m}$. We denominate this dense cloud as *tile*.

We executed our forestry pipeline on the tiles to analyze the results obtained with the pipeline with denser data. Fig. 7.23 illustrates the steps of the pipeline for one of the tiles, while Fig. 7.24 provides a close-up of the fitted cylinders.

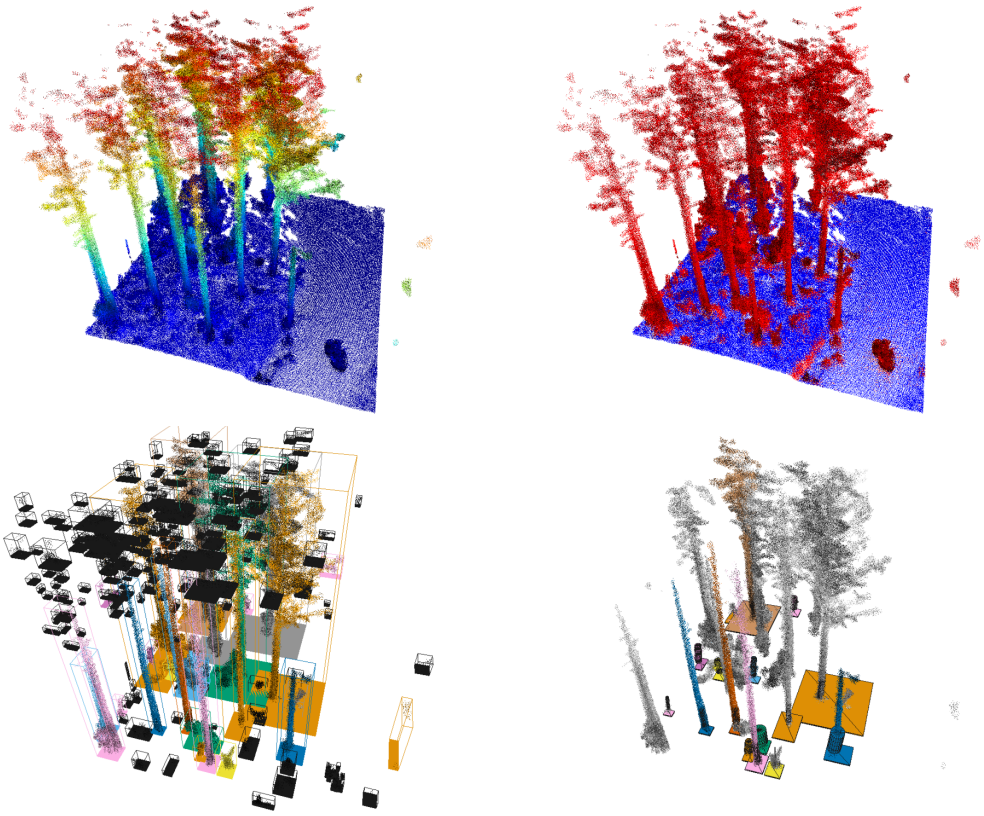

Figure 7.23: Results applying the proposed forestry pipeline to dense *tile* clouds generated with a Hesai LiDAR with larger vertical FoV.

In contrast to the payload data, which suffers from partial observations, the tile data obtained with the denser data was more complete, allowing the ground to be easily distinguished as well as the tree candidate clusters. However, due to the increased cloud density, we observed new challenges in the tree analysis step when fitting the cylinder model to estimate the DBH. The increased density revealed small trees and bushes adjacent to the big stems, which were often cluster together as the same tree candidate in spit of the two-level clustering strategy. As a consequence, the cylinder fitting was biased by these objects (Fig. 7.25 (a)) or it

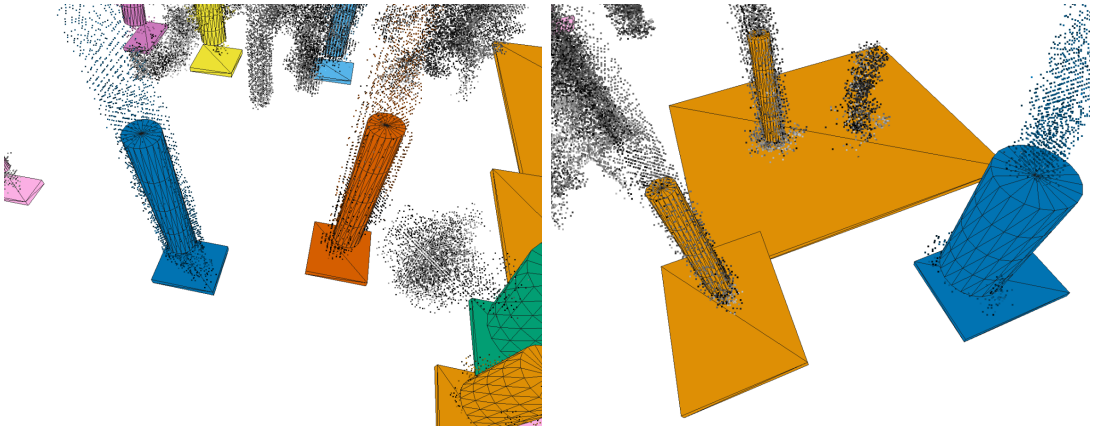

Figure 7.24: Close-up view of the cylinder fitting results obtained for the tile data. While the cylinders are correctly located, the diameter is over or under estimated in some cases. Please refer to the text for further discussion.

failed to detect a tree (Fig. 7.25 (b)), as it was difficult for the NLS optimisation to find the right cylinder when multiple modes were present.

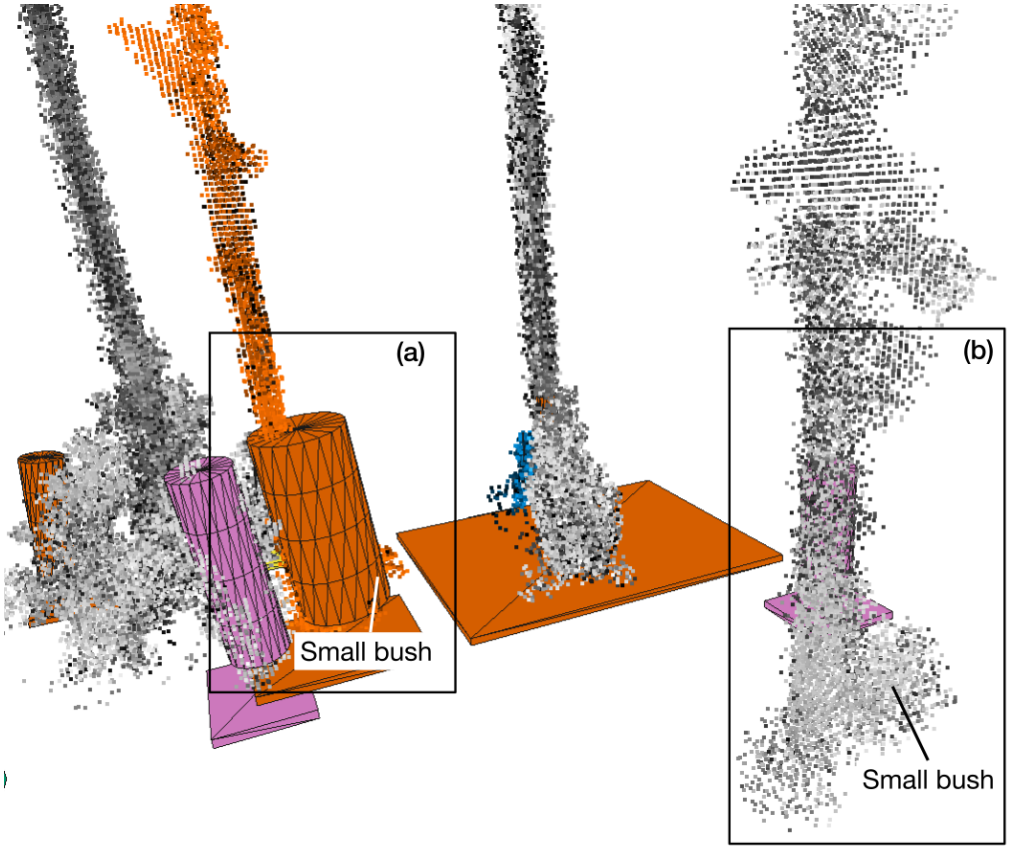

Figure 7.25: Failure cases observed with tile data. New points associated to bushes or small trees affected the cylinder fitting procedure, then producing biased estimates (a) or even failing to detect trees at all (b).

These results suggest that after the LiDAR density is solved by using a different sensor, there still are algorithmic improvements to introduce in the pipeline, particularly in the tree segmentation step to improve the clustering, as well as in the tree analysis step to improve the performance of the optimisation-based cylinder fitting.

7.5 Discussion

In this chapter we presented an autonomous forest inventory solution with legged robots. Our solution relies on the state estimation machinery to provide consistent representations of the environment, an multi-levelled autonomy solution to achieve autonomous navigation, and a capable forest analysis pipeline that produces relevant attributes for the DSS and other forestry tasks.

While the system is still under active development, the autonomous deployments we executed in the Finland campaign provided a meaningful measure of the performance of the system, which will shape the next steps. In particular, during the deployments we identified five main challenges for future campaigns.

State estimation in the wild In our autonomous deployments we relied on a CompSLAM version that was successfully used for exploration in underground environments in the context of the DARPA SubT Challenge [273]. In mines and caves, the geometric constraints obtained from LiDAR sensing are accurate but they suffer from repetitive structural patterns, where optimisation-based formulations suffer from ill-conditioned Hessians along the tunnel direction when performing the GN updates [306].

Methods based on complementary sensor fusion, such as CompSLAM, aim to solve this problem by identifying such failures and relying on alternative modalities, namely leg odometry, to compensate this failure. In structured environments with rigid terrain, such as the SubT Challenge, relying on leg odometry more is a sensible decision that does not transfer to forest environments, where the softness of the

terrain makes leg odometry drift severily. In forests we prefer to trust geometric exteroceptive sensing, i.e LiDAR, instead.

During our post-processing, we used VILENS with LiDAR and inertial sensing, obtaining consistent trajectory estimates that when combined with a SLAM system effectively reflected the path followed by the robot. This remarks the importance of having a SLAM system in the loop to ensure the autonomy system performs as intended.

Failure detection In mission M3, we reported an important failure due to the robot getting trapped in a damp area up to its knees. In the experiment, we could directly assess the situation and confirm that the robot was in an unrecoverable state. However, for a real deployment we require a mechanism to anticipate the failure to prevent it, or at least identify unrecoverable anomalous states and report this information for the operator to intervene.

Failure prediction and detection is a research problem that broadly relates to the anomaly detection methods we discussed in Sec. 3.3. While some recent works have explored this problem in the context of agriculture [134] as well as underground exploration [74], it is still an open question how to define failures in the context of forestry, how to detect them given the sensors available, and how to integrate such information as part of the planning system.

Effective navigation Our experiments relied on the local planner introduced in Chapter 5, and we used a geometry-based traversability estimation approach to simplify the modules being assessed in the campaign. As a consequence, the behavior displayed by the robot was conservative, avoiding any bump or bush that could imply a higher motion cost.

The learned traversability estimation approach presented in Chapter 6 could provide improvements on the locar planner's performance by *smoothing out* the effects and artifacts induced by grass, branches and twigs. However, this needs to be properly assessed before it is deployed in future campaigns, to ensure that the navigation system performs as intended.

In addition, we aim to improve our replanning strategy when goals are unreachable. When the local planner detected unreachable goals, we proceeded with the next goal in the survey plan, which helped the local planner to get out of local minima. However, a better strategy could be replanning to a node that was successfully traversed in the past, located in a parallel track to the original goal. By exploiting this information about the previously visited areas in the forest, we could potentially improve the autonomous performance of the system, which we aim to continue studying along the new developments.

Forest inventory The forest analysis pipeline that was presented is still under development. The initial results we report have allowed us to assess the data collected by the legged forester robot. A first major issue we observed was reduced FoV of the Velodyne VLP-16 LiDAR used in the deployments. This sensor is insufficient to fully capture the trees, completely missing the crown information, which we require for further forestry analysis.

On the other hand, we need to perform a quantitative evaluation of the algorithms and estimated attributes to identify limitations of the current strategy. In particular, the cylinder fitting must be further develop to improve on issues related to spurious points. Additionally, we need to quantitatively study its performance against ground truth data available from our partners within the DigiForest project, in order to assess the quality of the estimated DBH.

Nevertheless, the pipeline represents an initial step towards the development of an integrated autonomous forest surveying system. We plan to continue improving the current pipeline, also benefitting from the research currently developed by our partners that are specifically working on estimating forest attributes.

Real-time integration and online forestry reporting Finally, as mentioned in Sec. 7.4.3, we need to integrate the forest analysis pipeline for online operation, providing live reporting of the forest inventory. This also needs to be assessed by our forester partners, in order to get feedback on the possibilities that online reporting provides for the forestry operations.

7.6 Conclusion

In this chapter I presented a system for forest inventory with legged robots, as well as preliminary results from a series of field experiments. The tight integration of state estimation with capable autonomy systems enabled large-scale autonomous deployment across different locations in Evo, Finland. Preliminary analyses suggest that the current system and robotic platform should enable mapping of 1.5 ha in 1 h of operation, with negligible impact on the soil compared to the surveyed area.

Additional studies must be performed to confirm these claims, including larger and longer deployments in different forests. Factors such as seasons, which affect the ground conditions of the forest (e.g. undergrowth), can affect the statistics we reported here. In future work, we aim to improve the sensing payload and autonomy algorithms, and to use a more advanced legged platforms for futher trials.

8 Conclusion

This thesis has studied the navigation problem for legged robots through four contributions that exploit their unique capabilities. The systems presented faced the problem from different perspectives: sensing modalities, testing environments, and real-world applications. All the systems and algorithms developed in this thesis have been integrated on real robots, and tested in closed-loop autonomy experiments.

In this last chapter, I will present a critical discussion on what was done, what was learned, and how we can improve it in the future.

8.1 Achievements and Limitations

First, I will discuss the contributions presented in this thesis, complementing the ideas shared at the end of each chapter, and providing additional connections between the different systems.

Localisation In Chapter 4, I presented a VT&R system that enabled the practical deployment of a legged robot in industrial facilities. The system effectively addressed the navigation problem by closing the loop between the topo-metric representation and the locomotion controller, which was demonstrated through different real-world experiments. The other important contribution of this work was studying the

uncertainty of vision-based localisation, given by the negative entropy, and using this information for decision making (camera selection) in subsequent deployments.

When we talk about legged localisation, the challenges are not so different from any other robotic platform. I argue that localisation systems *per se* do not require legged information to succeed, and most of the challenges depend on environment and sensing instead. In Chapter 4 and Chapter 5 I used vision, as it was more suitable for path following in narrow environments, though in Chapter 7 I relied on a LiDAR SLAM system instead, as the environment provided enough geometric features to localise reliably; none of them required to introduce any aspects unique to legged platforms as part of the localisation process. In contrast, these aspects become highly relevant in scene representations used for planning through concepts such as traversability, which I discuss below.

Local Planning Chapter 5 introduced a reactive local planning framework to achieve safe navigation, motivated by challenges we observed in previous VT&R deployments. The local planner enabled the VT&R system presented in Chapter 4 to safely navigate in more complex environments, and react to under local changes found along the path. It also demonstrated competitive performance compared to other local planning approaches based on potential fields and motion primitives.

This work explored the idea of combining different scene representations to generate reactive motion, instead of using a single occupancy-based representation. Another important aspect of this contribution is that it became a core component in all the subsequent developments and field deployments showed in Chapter 6 and Chapter 7.

However, there are limitations related to its traditional local planning nature. The approach requires a local map, so its performance is limited by the representation and its capacity to encode the actual robot's capabilities, which I address in Chapter 6. On the other hand, local planners are generally designed to avoid collisions. However, when navigating in natural environments (e.g Chapter 7), collisions might

be inevitable due to sensing and state estimation errors, or limitations of the scene representation (e.g. discretisation). I discuss this further in the future directions.

Traversability Learning Chapter 6 presented an online learning approach to determine which areas could be traversed by a legged robot, without any prior information. This work was motivated by the necessity to encode the robot's capabilities in the local navigation map in a practical way. Previous approaches required extensive expert tuning, massive datasets, or complex labeling solutions. Instead, we proposed an approach that supervised the data and learned a neural model while the robot operated.

A key idea of this work was relying on deep neural models pre-trained in a self-supervised manner (foundation models). This enabled us to release the neural model from learning *everything* from scratch [36, 285, 103], and instead used the pre-trained features to focus on learning the relevant task. I believe that such models have great potential for other robotics applications, so I will give greater discussion of this topic later in this chapter.

The main limitations of this approach concern the design decisions and hardware limitations. The use of superpixels to reduce computational complexity, the coarse footprint-based self-supervision approach, and the local map integration via-raycasting are the main issues, reducing the fidelity of the estimated traversability score.

Forestry Applications Lastly, Chapter 7 introduced a real application of the aforementioned methods for forestry tasks. I presented a solution based on three components, including a consistent state estimation system, a multi-levelled autonomy approach, and a forestry pipeline that operated on point cloud data.

We executed five autonomous missions on different forest areas in Finland. We demonstrated that our solution enabled a legged platform to autonomously navigate within the forest, and collect valuable 3D data for futher processing. These experiments also allowed to contextualise the research developed in the previous chapters. In particular, the local planner from Chapter 5 was a key component in

the autonomy system, while the traversability estimation system from Chapter 6 has the potential to improve performance in future deployments.

The solution is currently under active development, and I plan to further improve the solution using the insights gained in the field deployments reported.

8.2 Lessons Learned

During this research I also learned many general lessons on robotic system design and deployment, as well as the navigation problem. The goal of this section is to provide a broader perspective on important challenges I recognise in the field.

Representations The idea of *representations for robotics* is present across the different chapters of this thesis. *Good* representations can massively simplify the solutions we design, allowing us to focus on the important aspects of the problem.

In this thesis I adopted a topological approach to represent the robot missions and global plans, both in the VT&R system as well as the *survey plan* used for the legged forester (Chapter 7). Metric information was only used for visual localisation against a reference node in a topo-metric map (Chapter 4), or local elevation maps for planning (Chapter 5).

Similarly, the learned pre-trained features obtained from large neural models have been effective for a variety of tasks, such as object detection, place recognition, and panoptic segmentation. In Chapter 6 we used them for traversability learning, releasing us from learning the features relevant for this task by ourselves.

A good representation can encode meaningful information about the task. For example, in Chapter 5, we used a GDF to encode the navigation information needed to reach the goal. This relates to the ideas of *navigation functions* [151], which enable robots to navigate, collision-free, just by implementing a control law that follows the gradient of the navigation function. This supposes that the navigation problem can be solved, in principle, just by having the appropriate representation. More recent works have also explored similar ideas for manipulation [211], navigation [93, 123], or semantic understanding [214, 131].

Hierarchical Architectures When developing robotic solutions, one common desire is designing solutions that solve *all the problems*. For example, local planners *must* guide a robot through any environment without collisions to be successful. Similarly, localisation systems *must* determine the robot's pose at any location regardless of the conditions.

However, when developing complex, integrated systems, it might be preferred to have modules that are *aware of their own limitations*. Similar ideas have been explored in the past, namely as elastic bands (coupling local planning and control) [224], subsumption architectures (layered decision-making modules) [37], as well as real-world deployments in the DARPA challenges [4, 274, 154]. In Chapter 7, instead of designing a perfect local planning solution able to deal with the any kind of environments, we implemented a strategy to identify when it was struggling to move forward. This information was reported to the upper level of the autonomy system, which instead selected a different local goal to continue the mission.

While these hierarchical interactions are common in control and planning, they have been less explored for state estimation solutions. *Active perception* [19] and *next-best-view* [27] approaches are examples that improve perception by decision-making and planning in a proactive fashion. The active camera selection system presented in Chapter 4 also relates to these ideas, as it anticipated perception degradation along the path using the learned CPMs.

Uncertainty estimates produced by odometry and SLAM systems are also a mechanism to report potential issues to other modules in the stack. However, as they are unable to provide guarantees, current *certification* methods aim to address these limitations [48].

Fundamental Ideas The last lesson regards the fundamental underlying ideas of the robotics toolbox. Optimisation and geometry are the backbone of many robotics problems, and identifying the connections between apparently unrelated problems can provide novel solutions and valuable insights.

The duality of the Least Squares (LS) problem in visual localisation and optimisation-based planning, motivated the use of RMPs in Chapter 5. While the localisation problem is originated from a probabilistic perspective [71], RMPs are motivated by purely geometric principles [231]. These dualities have been previously discussed in the past by some authors, for example see Todorov [270] and Toussaint [272].

Similarly, in Chapter 6 our online self-supervision scheme was motivated by the sliding window-based approaches to state estimation [256]. However, we did not formulate the learning problem as a windowed optimisation. Recent works have explored a tighter relationship of state estimation and learning [217], for example using the state estimation problem as self-supervision [304].

8.3 Future Directions

To conclude, I present future directions for legged navigation research, that I hope can motivate future projects in the field.

Pre-trained Models Large pre-trained (foundation) models have revolutionised robotics in the last two years. I expect that this topic will continue trending in the short term, as robotics presents specific challenges that go far beyond the training domain of such models. In Chapter 6, we used the DINO-ViT model [49] to obtain expressive features for traversability learning. While these features were trained in a self-supervised way to learn joint embeddings in spite of image augmentations, they are able to represent high-dimensional semantic features of the images.

My personal view is that the effect of these pre-trained features in the robotics community is like SIFT [177]: While we *might* not need to fully understand the meaning of the SIFT descriptors, we know they can be used to develop SfM solutions [246], visual odometry and SLAM systems [204], as well as place recognition methods based on feature quantisation [65, 100]. Pre-trained models are being used for semantic understanding, visual-language navigation, as well as place recognition, and I foresee more applications to come in the following years.

For navigation, these large pre-trained models can contribute with general priors that require some kind of expressive features. For example, we observed lighting robustness of these features in the experiments in Chapter 6, which can benefit localisation tasks. Similarly, models like CLIP [225] can provide semantic understanding via natural language, which can support better interfaces and representations for navigation [131]. However, it remains to be seen how these principles apply to other sensor modalities, such as LiDAR, as well as the conditions under which the datasets they were trained on break, especially for outdoor applications.

Interaction As previously mentioned, navigation is usually framed in a way that the robot must avoid contact with the environment, because it can either damage itself or the surroundings. For robots to succeed at navigating in the wild, we should free them from this constrain and enable some haptic interaction with the environment [222, 292].

Enabling this on legged robots is a manifold problem: it involves the design of new sensing capabilities (e.g. artificial skin, touch and temperature sensing), new estimation algorithms and controllers (to take into account this input for state estimation and control policies), as well as system architectures (what to do with this new information and what behaviours are available?).

For legged navigation, I would argue that interactions should be handled at the locomotion level, similarly to the locomotion policies developed by Lee et al. [164], Miki et al. [197], Kumar et al. [160], and Choi et al. [56]. However, instead of operating with nominal trotting gaits, they should enable changes in the gait (e.g. walk slowly) by leveraging feet and body contact information, as well as uncertainties about the environment. Ideally, they should also introduce semantic information from vision to improve performance on natural environments. Locomotion policies are either blind [164, 56] or disregard exteroceptive data when unreliable [197]. Semantics can provide additional cues about the terrain, which could improve performance to walk in the forest, similarly to the traversability

learning approach in Chapter 6. However, it is an open question how to train such policies combining simulation and real data.

With the locomotion policies becoming more capable, the higher level modules will get simpler. If the policies are able to deal with the short-range neighborhood (e.g. 1 m radius), they will be solving part of the local navigation, as it has been shown for structured environments [237]. This suggest that instead of having a explicit traversability estimation and local planning modules we could have a heading controller to guide local navigation instead, such as BADGR [138]. Nevertheless, I believe that such policies should not only provide an action to execute, but also feedback on their behaviour to be integrated into larger autonomy stacks.

Operation in Natural Environments As legged robots are becoming more affordable, novel applications will be unlocked in new domains. Legged robots are a versatile platform, that can enable automated and systematic monitoring of natural traits. In Chapter 7 we demonstrated an application for autonomous surveying and forest inventory. However, this use-case considered single, independent exploration missions to collect data from a determined area in the forest.

Long-term autonomous operation with legged robots has been demonstrated for monitoring in offshore and other industrial facilities [34, 10]. As the environments are structured and the operating conditions well-defined, they present the ideal scenario for their extended deployment. However, long-term deployments in natural environments have not been extensively explored.

Wildfire prevention, biodiversity monitoring, or general scientific data acquisition are potential areas where legged robots can contribute [79]. For such tasks, robots might perform periodic data collection missions, spread across several weeks, months, or even years. This imposes several multiple challenges for robotics as a field.

Robot hardware must be robust, efficient and compliant with the environment. On the other hand, state estimation must be reliable under different environmental conditions, lighting, seasons and also failures. However, it might not be necessary for this to be millimiter-accurate. Navigation and planning systems must be primarly

safe, to make sure the robot does not get damaged but it does not harm the environment during operation either.

Lastly, they need to comply with the specifications and requirements of the task. Because in the end is not about the robots, but the opportunities they offer to transform humanity for better.

Appendices

A
Other contributions

In addition to the publications that were presented in Sec. 1.4, additional contributions of this thesis include open-source software and tutorials:

1. `raw_image_pipeline`: An open-source raw image processing pipeline for debayering, white balancing, and undistortion:
   https://github.com/leggedrobotics/raw_image_pipeline

2. `digiforest_analysis` and `digiforest_analysis_ros`: A Python package and ROS wrapper that implement the forest analysis pipeline presented in Chapter 7:
   https://github.com/ori-drs/digiforest_drs

3. **Tutorial on optimisation-based state estimation**: A comprehensive review of optimisation on manifolds and uncertainty propagation on Lie groups.

   - Linear and non-linear state estimation:
     https://gtsam.org/2021/02/23/uncertainties-part1.html
   - Frames frames, manifolds, and Lie groups:
     https://gtsam.org/2021/02/23/uncertainties-part2.html
   - Uncertainty propagation on Lie groups:
     https://gtsam.org/2021/02/23/uncertainties-part3.html

B

Autonomous Legged Forester Robot: Additional Plots

B.1 Speed Profiles

We report detailed speed profiles to provide further details on the performance of the autonomous legged forester robot on the five missions described in Tab. 7.1. The speed profiles show the commanded velocity (from the local planner), the measured velocity (from the odometry system), as well as the safety interventions (done by the safety operator).

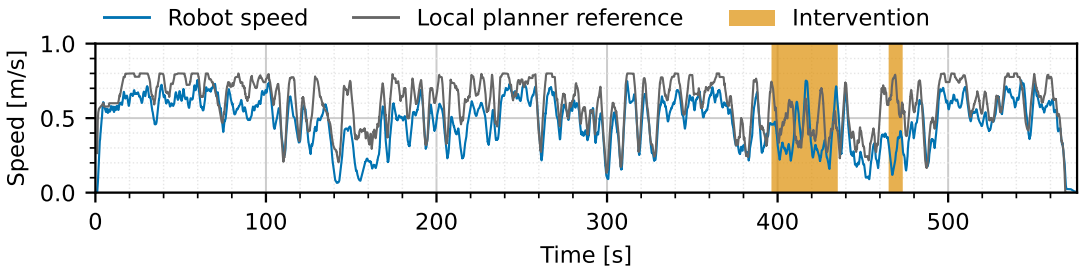

Figure B.1: Speed profile and interventions in mission M1.

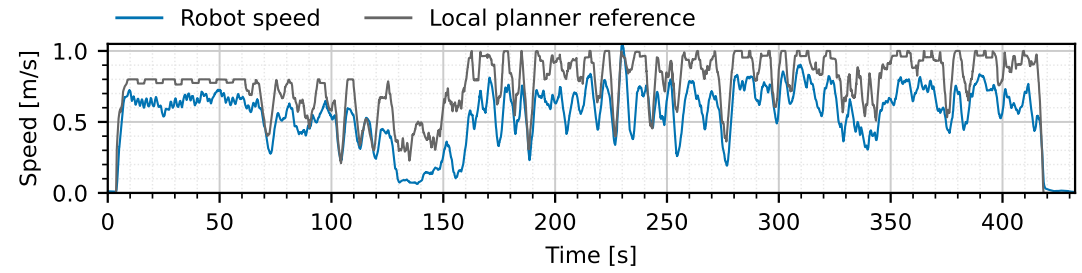

Figure B.2: Speed profile and interventions in mission M2.

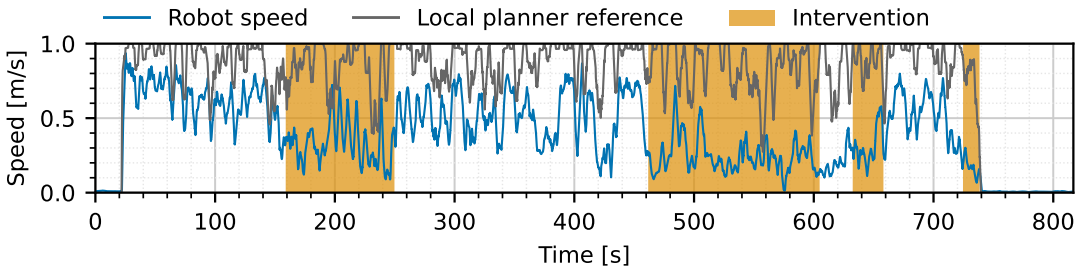

Figure B.3: Speed profile and interventions in mission M3.

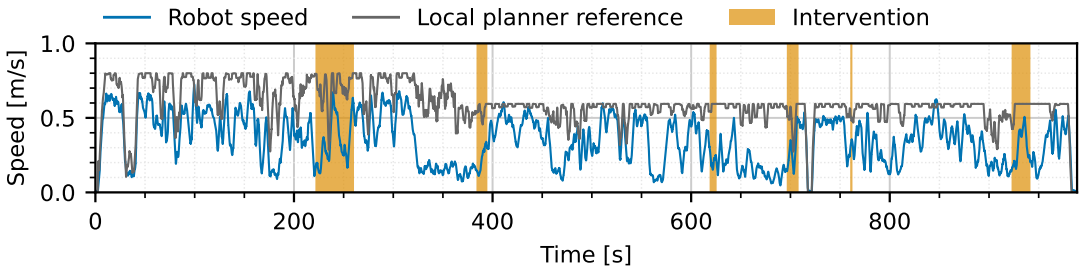

Figure B.4: Speed profile and interventions in mission M4.

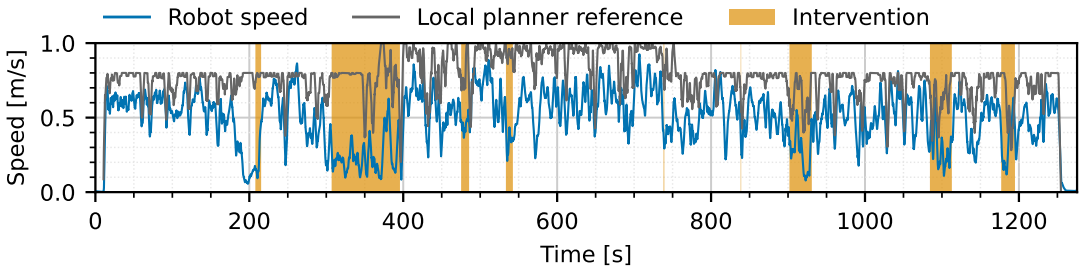

Figure B.5: Speed profile and interventions in mission M5.

References

[1] Pieter Abbeel and Andrew Y. Ng. "Apprenticeship learning via inverse reinforcement learning". In: *Intl. Conf. on Machine Learning (ICML)*. Vol. 69. 2004. DOI: 10.1145/1015330.1015430 (page 48).

[2] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 34.11 (2012), pp. 2274–2282. DOI: 10.1109/TPAMI.2012.120 (page 99).

[3] Evan Ackerman. "Robots Conquer the Underground: What DARPA's Subterranean Challenge Means for the Future of Autonomous Robots". In: *IEEE Spectrum* 59.5 (2022), pp. 30–37 (page 1).

[4] Ali Agha et al. "NeBula: Quest for Robotic Autonomy in Challenging Environments; TEAM CoSTAR at the DARPA Subterranean Challenge". In: *CoRR* abs/2103.11470 (2021). arXiv: 2103.11470 (pages 50, 141).

[5] Agility Robotics. *Robots*. https://agilityrobotics.com/robots. [Online; accessed 10-Feb-2023]. 2023 (pages 1, 50).

[6] Pablo Fernández Alcantarilla, Jesús Nuevo, and Adrien Bartoli. "Fast Explicit Diffusion for Accelerated Features in Nonlinear Scale Spaces". In: *British Machine Vision Conf. (BMVC)*. 2013. DOI: 10.5244/C.27.13 (page 35).

[7] Raquel Alfaro-Sánchez, Elena Valdés-Correcher, Josep Maria Espelta, Arndt Hampe, and Didier Bert. "How do social status and tree architecture influence radial growth, wood density and drought response in spontaneously established oak forests?" In: *Annals of Forest Science* 77.2 (2020), p. 49. DOI: 10.1007/s13595-020-00949-x (page 104).

[8] Danilo Roberti Alves Almeida, Angelica Maria Almeyda Zambrano, Eben North Broadbent, Amanda L. Wendt, Paul Foster, Benjamin E. Wilkinson, Carl Salk, Daniel De Almeida Papa, Scott Christopher Stark, Ruben Valbuena, Eric Bastos Gorgens, Carlos Alberto Silva, Pedro Henrique Santin Brancalion, Matthew Fagan, Paula Meli, and Robin Chazdon. "Detecting successional changes in tropical forest structure using GatorEye drone-borne lidar". en. In: *Biotropica* 52.6 (Nov. 2020), pp. 1155–1167. ISSN: 0006-3606, 1744-7429. DOI: 10.1111/btp.12814 (page 105).

[9] Karen Anderson and Kevin J Gaston. "Lightweight unmanned aerial vehicles will revolutionize spatial ecology". en. In: *Frontiers in Ecology and the Environment* 11.3 (2013), pp. 138–146. ISSN: 1540-9309. DOI: 10.1890/120150 (pages 104, 105).

[10] ANYbotics. *ANYmal - Autonomous Legged Robot*. https://www.anybotics.com/anymal-autonomous-legged-robot/. [Online; accessed 10-Feb-2023]. 2023 (pages 1, 24, 50, 144).

[11] Muhammad Asif Arain, Ioannis Havoutis, Claudio Semini, Jonas Buchli, and Darwin G. Caldwell. "A comparison of search-based planners for a legged robot". In: *Workshop on Robot Motion and Control (RoMoCo)*. 2013, pp. 104–109. DOI: 10.1109/RoMoCo.2013.6614593 (page 40).

[12] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. "NetVLAD: CNN architecture for weakly supervised place recognition". In: *IEEE Int. Conf. Computer Vision and Pattern Recognition*. 2016, pp. 5297–5307. DOI: 10.1109/CVPR.2016.572 (page 35).

[13] Philip Arm, Gabriel Waibel, Gabriela Ligeza, Valentin Bickel, Marco Tranzatto, Samuel Zimmermann, Timon Homberger, Lars Horvath, Hugo Umbers, Florian Kehl, Hendrik Kolvenbach, and Marco Hutter. "Results and Lessons Learned from the First Field Trial of the ESA-ESRIC Space Resources Challenge of Team GLIMPSE". In: *Symposium on Advanced Space Technologies in Robotics and Automation (ASTRA)*. 2022. DOI: 10.3929/ethz-b-000545465 (page 50).

[14] Philip Arm, Gabriel Waibel, Jan Preisig, Turcan Tuna, Ruyi Zhou, Valentin Bickel, Gabriela Ligeza, Takahiro Miki, Florian Kehl, Hendrik Kolvenbach, and Marco Hutter. "Scientific exploration of challenging planetary analog environments with a team of legged robots". In: *Sci. Robot.* 8.80 (2023), eade9548. DOI: 10.1126/scirobotics.ade9548 (pages 50, 106).

[15] Minoru Asada, Peter Stone, Manuela Veloso, Daniel D. Lee, and Daniele Nardi. "RoboCup: A Treasure Trove of Rich Diversity for Research Issues and Interdisciplinary Connections [TC Spotlight]". In: *IEEE Robotics & Automation Magazine* 26.3 (2019), pp. 99–102. DOI: 10.1109/MRA.2019.2928959 (page 49).

[16] G. P. Asner, R. E. Martin, D. E. Knapp, R. Tupayachi, C. B. Anderson, F. Sinca, N. R. Vaughn, and W. Llactayo. "Airborne laser-guided imaging spectroscopy to map forest trait diversity and guide conservation". en. In: *Science* 355.6323 (Jan. 2017), pp. 385–389. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aaj1987 (page 105).

[17] Christopher G. Atkeson et al. "What Happened at the DARPA Robotics Challenge Finals". In: *The DARPA Robotics Challenge Finals: Humanoid Robots To The Rescue*. Ed. by Matthew Spenko, Stephen Buerger, and Karl Iagnemma. Springer International Publishing, 2018, pp. 667–684. ISBN: 978-3-319-74666-1. DOI: 10.1007/978-3-319-74666-1_17 (page 49).

[18] Emanuele Aucone, Steffen Kirchgeorg, Alice Valentini, Loïc Pellissier, Kristy Deiner, and Stefano Mintchev. "Drone-assisted collection of environmental DNA from tree branches for biodiversity monitoring". In: *Sci. Robot.* 8.74 (2023), eadd5762. DOI: 10.1126/scirobotics.add5762 (page 106).

[19] Ruzena Bajcsy, Yiannis Aloimonos, and John K. Tsotsos. "Revisiting active perception". In: *Autonomous Robots* 42.2 (2018), pp. 177–196. DOI: 10.1007/s10514-017-9615-3 (pages 63, 141).

[20] Max Bajracharya, Andrew Howard, Larry H. Matthies, Benyang Tang, and Michael Turmon. "Autonomous off-road navigation with end-to-end learning for the LAGR program". In: *J. Field Robot.* 26.1 (2009), pp. 3–25. DOI: 10.1002/rob.20269 (page 46).

[21] Max Bajracharya, Jeremy Ma, Matthew Malchano, Alex Perkins, Alfred A. Rizzi, and Larry H. Matthies. "High fidelity day/night stereo mapping with vegetation and negative obstacle detection for vision-in-the-loop walking". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2013, pp. 3663–3670. DOI: 10.1109/IROS.2013.6696879 (pages 44, 50).

[22] T.D. Barfoot. *State Estimation for Robotics*. Cambridge University Press, 2017. ISBN: 9781108509718 (pages 8, 15).

[23] Jonathan T. Barron. "A General and Adaptive Robust Loss Function". In: *IEEE Int. Conf. Computer Vision and Pattern Recognition*. 2019, pp. 4331–4339 (page 116).

[24] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. "SURF: Speeded Up Robust Features". In: *Eur. Conf. on Computer Vision (ECCV)*. Vol. 3951. 2006, pp. 404–417. DOI: 10.1007/11744023_32 (page 35).

[25] Dominik Belter, Jan Wietrzykowski, and Piotr Skrzypczynski. "Employing Natural Terrain Semantics in Motion Planning for a Multi-Legged Robot". In: *J. Intell. Robotic Syst.* 93.3-4 (2019), pp. 723–743. DOI: 10.1007/S10846-018-0865-X (page 45).

[26] Paul J. Besl and Neil D. McKay. "A Method for Registration of 3-D Shapes". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 14.2 (1992), pp. 239–256. DOI: 10.1109/34.121791 (page 109).

[27] Andreas Bircher, Mina Kamel, Kostas Alexis, Helen Oleynikova, and Roland Siegwart. "Receding Horizon "Next-Best-View" Planner for 3D Exploration". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2016, pp. 1462–1468. DOI: 10.1109/ICRA.2016.7487281 (page 141).

[28] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN: 0387310738 (pages 10, 13).

[29] Michael Bloesch, Michael Burri, Hannes Sommer, Roland Siegwart, and Marco Hutter. "The Two-State Implicit Filter: Recursive Estimation for Mobile Robots". In: *IEEE Robot. Autom. Lett. (RA-L)* 3.1 (2018), pp. 573–580. DOI: 10.1109/LRA.2017.2776340 (pages 26, 118).

[30] Michael Bloesch, Marco Hutter, Mark A Hoepflinger, Stefan Leutenegger, Christian Gehring, C David Remy, and Roland Siegwart. "State Estimation for Legged Robots - Consistent Fusion of Leg Kinematics and IMU". In: *Robotics: Science and Systems (RSS)*. 2012. DOI: 10.15607/RSS.2012.VIII.003 (page 26).

[31] Rishi Bommasani et al. "On the Opportunities and Risks of Foundation Models". In: *CoRR* abs/2108.07258 (2021). arXiv: 2108.07258 (page 101).

[32] Olaf Booij, Bas Terwijn, Zoran Zivkovic, and Ben J. A. Kröse. "Navigation using an appearance based topological map". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2007, pp. 3927–3932. DOI: 10.1109/ROBOT.2007.364081 (page 37).

[33] Paulo Borges, Thierry Peynot, Sisi Liang, Bilal Arain, Matthew Wildie, Melih Minareci, Serge Lichman, Garima Samvedi, Inkyu Sa, Nicolas Hudson, et al. "A Survey on Terrain Traversability Analysis for Autonomous Ground Vehicles: Methods, Sensors, and Challenges". In: *Field Robotics* 2.1 (2022), pp. 1567–1627 (pages 28, 44).

[34] Boston Dynamics. *Spot ®- The Agile Mobile Robot*. https://www.bostondynamics.com/products/spot. [Online; accessed 10-Feb-2023]. 2023 (pages 1, 2, 50, 144).

[35] Christopher L. Brack. *Forest Measurement and Modeling - Measuring trees, stands and forests for effective forest management*. Tech. rep. Australian National University, 2001. URL: http://fennerschool-associated.anu.edu.au/mensuration/index.htm (page 104).

[36] David M. Bradley, Jonathan K. Chang, David Silver, Matthew Powers, Herman Herman, Peter Rander, and Anthony Stentz. "Scene understanding for a high-mobility walking robot". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2015, pp. 1144–1151. DOI: 10.1109/IROS.2015.7353514 (pages 45, 51, 139).

[37] Rodney A. Brooks. "A robust layered control system for a mobile robot". In: *IEEE J. Robotics Autom.* 2.1 (1986), pp. 14–23. DOI: 10.1109/JRA.1986.1087032 (page 141).

[38] Rodney A. Brooks. "Visual map making for a mobile robot". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 1985, pp. 824–829. DOI: 10.1109/ROBOT.1985.1087348 (page 37).

[39] Duane C Brown. "Decentering distortion of lenses". In: *Photogrammetric Engineering* 32.3 (1966), pp. 444–462 (page 21).

[40] Russell Buchanan, Tirthankar Bandyopadhyay, Marko Bjelonic, Lorenz Wellhausen, Marco Hutter, and Navinda Kottege. "Walking Posture Adaptation for Legged Robot Navigation in Confined Spaces". In: *IEEE Robot. Autom. Lett. (RA-L)* 4.2 (2019), pp. 2148–2155. DOI: 10.1109/LRA.2019.2899664 (page 40).

[41] Wolfram Burgard, Martial Hebert, and Maren Bennewitz. "World Modeling". In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Springer Handbooks. Springer, 2016, pp. 1135–1152. DOI: 10.1007/978-3-319-32552-1_45 (page 27).

[42] Andrew Bylard, Riccardo Bonalli, and Marco Pavone. "Composable Geometric Motion Policies using Multi-Task Pullback Bundle Dynamical Systems". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2021, pp. 7464–7470. DOI: 10.1109/ICRA48506.2021.9561320 (page 77).

[43] Sylvain Calinon. "Gaussians on Riemannian Manifolds: Applications for Robot Learning and Adaptive Control". In: *IEEE Robotics & Automation Magazine* 27.2 (2020), pp. 33–45. DOI: 10.1109/MRA.2020.2980548 (page 8).

[44] A. Calleja-Huerta, M. Lamandé, O. Green, and L.J. Munkholm. "Vertical and horizontal stresses from a lightweight autonomous field robot during repeated wheeling". en. In: *Soil and Tillage Research* 233 (Sept. 2023), p. 105790. ISSN: 01671987. DOI: 10.1016/j.still.2023.105790 (page 106).

[45] Ricardo J. G. B. Campello, Davoud Moulavi, and Joerg Sander. "Density-Based Clustering Based on Hierarchical Density Estimates". en. In: *Advances in Knowledge Discovery and Data Mining*. Ed. by Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2013, pp. 160–172. ISBN: 978-3-642-37456-2. DOI: 10.1007/978-3-642-37456-2_14 (page 114).

[46] Carlos Campos, Richard Elvira, Juan J. Gómez Rodríguez, José M. M. Montiel, and Juan D. Tardós. "ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM". In: *IEEE Trans. Robot.* 37.6 (2021), pp. 1874–1890. DOI: 10.1109/TRO.2021.3075644 (page 34).

[47] Marco Camurri, Maurice F. Fallon, Stéphane Bazeille, Andreea Radulescu, Victor Barasuol, Darwin G. Caldwell, and Claudio Semini. "Probabilistic Contact Estimation and Impact Detection for State Estimation of Quadruped Robots". In: *IEEE Robot. Autom. Lett. (RA-L)* 2.2 (2017), pp. 1023–1030. DOI: 10.1109/LRA.2017.2652491 (page 128).

[48] Luca Carlone. "Estimation Contracts for Outlier-Robust Geometric Perception". In: *Found. Trends Robotics* 11.2-3 (2023), pp. 90–224. DOI: 10.1561/2300000077 (page 141).

[49] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. "Emerging Properties in Self-Supervised Vision Transformers". In: *Intl. Conf. on Computer Vision (ICCV)*. 2021, pp. 9630–9640. DOI: 10.1109/ICCV48922.2021.00951 (pages 99, 101, 142).

[50] Mateo Guaman Castro, Samuel Triest, Wenshan Wang, Jason M. Gregory, Felix Sanchez, John G. Rogers III, and Sebastian Scherer. "How Does It Feel? Self-Supervised Costmap Learning for Off-Road Vehicle Traversability". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2023 (page 47).

[51] Raja Chatila and Jean-Paul Laumond. "Position referencing and consistent world modeling for mobile robots". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 1985, pp. 138–145. DOI: 10.1109/ROBOT.1985.1087373 (page 37).

[52] R. Omar Chavez-Garcia, Jérôme Guzzi, Luca M. Gambardella, and Alessandro Giusti. "Learning Ground Traversability From Simulations". In: *IEEE Robot. Autom. Lett. (RA-L)* 3.3 (2018), pp. 1695–1702. DOI: 10.1109/LRA.2018.2801794 (page 44).

[53] Steven W. Chen, Guilherme V. Nardari, Elijah S. Lee, Chao Qu, Xu Liu, Roseli Ap. Francelin Romero, and Vijay Kumar. "SLOAM: Semantic Lidar Odometry and Mapping for Forest Inventory". In: *IEEE Robot. Autom. Lett. (RA-L)* 5.2 (2020), pp. 612–619. DOI: 10.1109/LRA.2019.2963823 (pages 105, 106, 115).

[54] Joel E. Chestnutt and James J. Kuffner. "A tiered planning strategy for biped navigation". In: *IEEE/RSJ Int. Conf. on Humanoid Robots*. 2004, pp. 422–436. DOI: 10.1109/ICHR.2004.1442135 (page 39).

[55] Annett Chilian and Heiko Hirschmüller. "Stereo camera based navigation of mobile robots on rough terrain". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2009, pp. 4571–4576. DOI: 10.1109/IROS.2009.5354535 (page 39).

[56] Suyoung Choi, Gwanghyeon Ji, Jeongsoo Park, Hyeongjun Kim, Juhyeok Mun, Jeonghyun Lee, and Jemin Hwangbo. "Learning quadrupedal locomotion on deformable terrain". In: *Sci. Robot.* 8.74 (2023). DOI: 10.1126/scirobotics.ade2256 (page 143).

[57] Howie Choset. "Coverage of Known Spaces: The Boustrophedon Cellular Decomposition". In: *Autonomous Robots* 9.3 (2000), pp. 247–253. DOI: 10.1023/A:1008958800904 (page 110).

[58] Timothy H. Chung, Viktor Orekhov, and Angela Maio. "Into the Robotic Depths: Analysis and Insights from the DARPA Subterranean Challenge". In: *Annual Review of Control, Robotics, and Autonomous Systems* 6.1 (2023) (pages 1, 49).

[59] Winston Churchill and Paul Newman. "Experience-based navigation for long-term localisation". In: *Intl. J. of Robot. Res.* 32.14 (2013), pp. 1645–1661. DOI: 10.1177/0278364913499193 (page 39).

[60] Winston Churchill and Paul Newman. "Practice makes perfect? Managing and Leveraging Visual Experiences for Lifelong Navigation". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2012. DOI: 10.1109/ICRA.2012.6224596 (page 38).

[61] CMU Robotics Institute. *History Channel 1998 : Driverless Car Technology Overview at Carnegie Mellon University*. https://www.youtube.com/watch?v=2KMAAmkz9go. [Online; accessed 16-Jun-2023]. 2017 (page 42).

[62] European Commission. *Commission communication: New EU forest strategy for 2030*. 2021. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52021DC0572 (page 106).

[63] Peter Corke. *Robotics, Vision and Control: Fundamental Algorithms in MATLAB*. Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg, 2011. ISBN: 9783642201431 (pages 2, 17).

[64] Peter Corke, Jorge Lobo, and Jorge Dias. "An Introduction to Inertial and Visual Sensing". In: *Intl. J. of Robot. Res.* 26.6 (2007), pp. 519–535. DOI: 10.1177/0278364907079279 (page 23).

[65] Mark Joseph Cummins and Paul M. Newman. "FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance". In: *Intl. J. of Robot. Res.* 27.6 (2008), pp. 647–665. DOI: 10.1177/0278364908090961 (pages 35, 36, 142).

[66] Dominic Dall'Osto, Tobias Fischer, and Michael Milford. "Fast and Robust Bio-inspired Teach and Repeat Navigation". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2021, pp. 500–507. DOI: 10.1109/IROS51168.2021.9636334 (page 37).

[67] Amaury Dame and Eric Marchand. "Using mutual information for appearance-based visual path following". In: *Robot. Auton. Syst.* 61.3 (2013), pp. 259–270. ISSN: 09218890. DOI: 10.1016/j.robot.2012.11.004 (page 37).

[68] DARPA. *LS3 Follow Tight*. https://www.youtube.com/watch?v=hNUeSUXOc-w. [Online; accessed 4-Jul-2023]. 2012 (page 51).

[69] Mauricio Delbracio, Damien Kelly, Michael S. Brown, and Peyman Milanfar. "Mobile Computational Photography: A Tour". In: *Annual Review of Vision Science* 7.1 (2021). PMID: 34524880, pp. 571–604. DOI: 10.1146/annurev-vision-093019-115521 (page 17).

[70] Frank Dellaert. *Derivatives and Differentials*. Tech. rep. Georgia Institute of Technology, 2022. URL: https://github.com/borglab/gtsam/blob/develop/doc/math.pdf (page 76).

[71] Frank Dellaert and Michael Kaess. *Factor Graphs for Robot Perception*. Hanover, MA, USA: Now Publishers Inc., 2017. ISBN: 168083326X (pages 10, 13, 142).

[72] Frank Dellaert and Michael Kaess. "Square Root SAM: Simultaneous Localization and Mapping via Square Root Information Smoothing". In: *Intl. J. of Robot. Res.* 25.12 (2006), pp. 1181–1203. DOI: 10.1177/0278364906072768 (page 12).

[73] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. "Superpoint: Self-supervised interest point detection and description". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 224–236 (page 35).

[74] Sharmita Dey, David D. Fan, Robin Schmid, Anushri Dixit, Kyohei Otsu, Thomas Touma, Arndt F. Schilling, and Ali-Akbar Agha-Mohammadi. "PrePARE: Predictive Proprioception for Agile Failure Event Detection in Robotic Exploration of Extreme Terrains". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2022, pp. 4338–4343. DOI: 10.1109/IROS47612.2022.9981660 (page 134).

[75] DigiForest Consortium. *Digiforest | Digital Analytics and Robotics for Sustainable Forestry*. https://digiforest.eu. [Online; accessed 01-Sep-2023]. 2023 (page 102).

[76] Edsger W. Dijkstra. "A note on two problems in connexion with graphs". In: *Numerische Mathematik* 1 (1959), pp. 269–271. DOI: 10.1007/BF01386390 (page 29).

[77] Ralph O. Dubayah and Jason B. Drake. "Lidar Remote Sensing for Forestry". In: *Journal of Forestry* 98.6 (June 2000), pp. 44–46. ISSN: 0022-1201. DOI: 10.1093/jof/98.6.44 (page 104).

[78] Thomas Dudzik, Matthew Chignoli, Gerardo Bledt, Bryan Lim, Adam Miller, Donghyun Kim, and Sangbae Kim. "Robust Autonomous Navigation of a Small-Scale Quadruped Robot in Real-World Environments". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2020, pp. 3664–3671. DOI: 10.1109/IROS45743.2020.9340701 (page 40).

[79] Matthew Dunbabin and Lino Marques. "Robots for Environmental Monitoring: Significant Advancements and Applications". In: *IEEE Robotics & Automation Magazine* 19.1 (2012), pp. 24–39. DOI: 10.1109/MRA.2011.2181683 (page 144).

[80] Boston Dynamics. *Getting Started with Autowalk*. https://support.bostondynamics.com/s/article/Getting-Started-with-Autowalk. [Online; accessed 15-Jun-2023] (page 50).

[81] David Eberly. *Least Squares Fitting of Data by Linear or Quadratic Structures*. en. Tech. rep. Geometric Tools, 1999. URL: https://www.geometrictools.com/Documentation/LeastSquaresFitting.pdf (page 115).

[82] Arne D. Ekstrom, Hugo J. Spiers, Véronique D. Bohbot, and R. Shayna Rosenbaum. *Human Spatial Navigation*. Princeton University Press, 2018. ISBN: 9780691171746 (page 2).

[83] Jakob Engel, Thomas Schöps, and Daniel Cremers. "LSD-SLAM: Large-Scale Direct Monocular SLAM". In: *Eur. Conf. on Computer Vision (ECCV)*. Vol. 8690. Lecture Notes in Computer Science. Springer, 2014, pp. 834–849. DOI: 10.1007/978-3-319-10605-2_54 (page 35).

[84] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. "A density-based algorithm for discovering clusters in large spatial databases with noise". In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD'96. Portland, Oregon: AAAI Press, Aug. 1996, pp. 226–231 (page 114).

[85] Parker Ewen, Adam Li, Yuxin Chen, Steven Hong, and Ram Vasudevan. "These Maps are Made for Walking: Real-Time Terrain Property Estimation for Mobile Robots". In: *IEEE Robot. Autom. Lett. (RA-L)* 7.3 (2022), pp. 7083–7090. DOI: 10.1109/LRA.2022.3180439 (pages 44, 45).

[86] David D. Fan, Kyohei Otsu, Yuki Kubo, Anushri Dixit, Joel Burdick, and Ali-Akbar Agha-Mohammadi. "STEP: Stochastic Traversability Evaluation and Planning for Safe Off-road Navigation". In: *Robotics: Science and Systems (RSS)*. 2021 (pages 40, 44).

[87] Péter Fankhauser, Michael Bloesch, and Marco Hutter. "Probabilistic Terrain Mapping for Mobile Robots With Uncertain Localization". In: *IEEE Robot. Autom. Lett. (RA-L)* 3.4 (2018), pp. 3019–3026. DOI: 10.1109/LRA.2018.2849506 (page 40).

[88] Farbod Farshidian, Michael Neunert, Alexander W. Winkler, Gonzalo Rey, and Jonas Buchli. "An efficient optimal planning and control framework for quadrupedal locomotion". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2017, pp. 93–100. DOI: 10.1109/ICRA.2017.7989016 (page 31).

[89] Martin A. Fischler and Robert C. Bolles. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography". In: *Commun. ACM* 24.6 (1981), 381–395. ISSN: 0001-0782. DOI: 10.1145/358669.358692 (pages 20, 35).

[90] Christian Forster, Luca Carlone, Frank Dellaert, and Davide Scaramuzza. "On-Manifold Preintegration for Real-Time Visual–Inertial Odometry". In: *IEEE Trans. Robot.* 33.1 (2017), pp. 1–21. DOI: 10.1109/TRO.2016.2597321 (page 109).

[91] S. Fountas, T. A. Gemtos, and S. Blackmore. "Robotics and Sustainability in Soil Engineering". In: *Soil Engineering*. Ed. by Athanasios P. Dedousis and Thomas Bartzanas. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 69–80. ISBN: 978-3-642-03681-1. DOI: 10.1007/978-3-642-03681-1_5 (page 106).

[92] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. "The dynamic window approach to collision avoidance". In: *IEEE Robotics & Automation Magazine* 4.1 (1997), pp. 23–33. DOI: 10.1109/100.580977 (page 30).

[93] Jonas Frey, David Hoeller, Shehryar Khattak, and Marco Hutter. "Locomotion Policy Guided Traversability Learning using Volumetric Representations of Complex Environments". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2022, pp. 5722–5729. DOI: 10.1109/IROS47612.2022.9982190 (pages 45, 140).

[94] Jonas Frey, Matias Mattamala, Nived Chebrolu, Cesar Cadena, Maurice Fallon, and Marco Hutter. "Fast Traversability Estimation for Wild Visual Navigation". In: *Robotics: Science and Systems (RSS)*. 2023 (pages 4, 79).

[95] Zipeng Fu, Ashish Kumar, Ananye Agarwal, Haozhi Qi, Jitendra Malik, and Deepak Pathak. "Coupling Vision and Proprioception for Navigation of Legged Robots". In: *IEEE Int. Conf. Computer Vision and Pattern Recognition*. 2022, pp. 17252–17262. DOI: 10.1109/CVPR52688.2022.01676 (page 41).

[96] Paul Furgale. "Representing robot pose: The good, the bad, and the ugly". In: *What Sucks in Robotics and How to Fix It: Lessons Learned from Building Complex Systems (ICRA Workshop)*. 2014. URL: http://paulfurgale.info/news/2014/6/9/representing-robot-pose-the-good-the-bad-and-the-ugly (page 9).

[97] Paul Furgale and Timothy D. Barfoot. "Visual teach and repeat for long-range rover autonomy". In: *J. Field Robot.* 27.5 (2010), pp. 534–560. DOI: 10.1002/rob.20342 (page 38).

[98] Paul Furgale, Joern Rehder, and Roland Siegwart. "Unified temporal and spatial calibration for multi-sensor systems". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2013, pp. 1280–1286. DOI: 10.1109/IROS.2013.6696514 (pages 18, 22).

[99] Magnus Gaertner, Marko Bjelonic, Farbod Farshidian, and Marco Hutter. "Collision-Free MPC for Legged Robots in Static and Dynamic Scenes". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2021, pp. 8266–8272. DOI: 10.1109/ICRA48506.2021.9561326 (page 41).

[100] Dorian Gálvez-López and Juan D. Tardós. "Bags of binary words for fast place recognition in image sequences". In: *IEEE Trans. Robot.* 28.5 (2012), pp. 1188–1197. ISSN: 15523098. DOI: 10.1109/TRO.2012.2197158 (pages 35, 36, 142).

[101] Lu Gan, Jessy W. Grizzle, Ryan M. Eustice, and Maani Ghaffari. "Energy-Based Legged Robots Terrain Traversability Modeling via Deep Inverse Reinforcement Learning". In: *IEEE Robot. Autom. Lett. (RA-L)* 7.4 (2022), pp. 8807–8814. DOI: 10.1109/LRA.2022.3188100 (pages 46, 48).

[102] Siddhant Gangapurwala, Mathieu Geisert, Romeo Orsolino, Maurice F. Fallon, and Ioannis Havoutis. "RLOC: Terrain-Aware Legged Locomotion Using Reinforcement Learning and Optimal Control". In: *IEEE Trans. Robot.* 38.5 (2022), pp. 2908–2927. DOI: 10.1109/TRO.2022.3172469 (page 1).

[103] Mateus Valverde Gasparino, Arun Narenthiran Sivakumar, Yixiao Liu, Andres E. Baquero Velasquez, Vitor A. H. Higuti, John Rogers, Huy T. Tran, and Girish Chowdhary. "WayFAST: Navigation With Predictive Traversability in the Field". In: *IEEE Robot. Autom. Lett. (RA-L)* 7.4 (2022), pp. 10651–10658. DOI: 10.1109/LRA.2022.3193464 (pages 47, 139).

[104] Donald Geman and Stuart Geman. "Bayesian image analysis". In: *Disordered systems and biological organization*. Ed. by E. Bienenstock, F. Fogelman Soulié, and G. Weisbuch. Berlin, Heidelberg: Springer Berlin Heidelberg, 1986, pp. 301–319. ISBN: 978-3-642-82657-3 (page 116).

[105] Alessandro Giusti, Jerome Guzzi, Dan C. Ciresan, Fang-Lin He, Juan P. Rodriguez, Flavio Fontana, Matthias Faessler, Christian Forster, Jürgen Schmidhuber, Gianni Di Caro, Davide Scaramuzza, and Luca Maria Gambardella. "A Machine Learning Approach to Visual Perception of Forest Trails for Mobile Robots". In: *IEEE Robot. Autom. Lett. (RA-L)* 1.2 (2016), pp. 661–667. DOI: 10.1109/LRA.2015.2509024 (page 43).

[106] Gene. H. Golub and Charles .F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 2013. ISBN: 9781421408590 (page 12).

[107] David Gossow, Adam Leeper, Dave Hershberger, and Matei Ciocarlie. "Interactive Markers: 3-D User Interfaces for ROS Applications [ROS Topics]". In: *IEEE Robotics & Automation Magazine* 18.4 (Dec. 2011), pp. 14–15. ISSN: 1558-223X. DOI: 10.1109/MRA.2011.943230 (page 110).

[108] Mona Gridseth and Timothy D. Barfoot. "Keeping an Eye on Things: Deep Learned Features for Long-Term Visual Localization". In: *IEEE Robot. Autom. Lett. (RA-L)* 7.2 (2022), pp. 1016–1023. DOI: 10.1109/LRA.2021.3136867 (page 39).

[109] Carsten Griwodz, Simone Gasparini, Lilian Calvet, Pierre Gurdjos, Fabien Castan, Benoit Maujean, Gregoire De Lillo, and Yann Lanthony. "AliceVision Meshroom: An open-source 3D reconstruction pipeline". In: *ACM Multimedia Systems Conference (MMSys)*. ACM Press, 2021. DOI: 10.1145/3458305.3478443 (page 34).

[110] Jérôme Guzzi, R. Omar Chavez-Garcia, Mirko Nava, Luca Maria Gambardella, and Alessandro Giusti. "Path Planning With Local Motion Estimations". In: *IEEE Robot. Autom. Lett. (RA-L)* 5.2 (2020), pp. 2586–2593. DOI: 10.1109/LRA.2020.2972849 (page 41).

[111] Raia Hadsell, Dushyant Rao, Andrei A. Rusu, and Razvan Pascanu. "Embracing Change: Continual Learning in Deep Neural Networks". In: *Trends in Cognitive Sciences* 24.12 (2020), pp. 1028–1040. DOI: 10.1016/j.tics.2020.09.004 (page 101).

[112] Raia Hadsell, Pierre Sermanet, Jan Ben, Ayse Erkan, Marco Scoffier, Koray Kavukcuoglu, Urs Muller, and Yann LeCun. "Learning Long-range Vision for Autonomous Off-road Driving". In: *J. Field Robot.* 26.2 (2009), pp. 120–144. DOI: 10.1002/rob.20276 (page 46).

[113] Ronald J. Hall. "The Roles of Aerial Photographs in Forestry Remote Sensing Image Analysis". en. In: *Remote Sensing of Forest Environments: Concepts and Case Studies*. Ed. by Michael A. Wulder and Steven E. Franklin. Boston, MA: Springer US, 2003, pp. 47–75. ISBN: 978-1-4615-0306-4. DOI: 10.1007/978-1-4615-0306-4_3 (page 104).

[114] Graham John Hamilton. *Forest mensuration handbook*. Forestry Commission, 1988 (page 104).

[115] Peter E. Hart, Nils J. Nilsson, and Bertram Raphael. "A Formal Basis for the Heuristic Determination of Minimum Cost Paths". In: *IEEE Trans. Syst. Sci. Cybern.* 4.2 (1968), pp. 100–107. DOI: 10.1109/TSSC.1968.300136 (page 29).

[116] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004. ISBN: 9780511811685. DOI: 10.1017/cbo9780511811685 (page 17).

[117] Ross Hartley, Josh Mangelson, Lu Gan, Maani Ghaffari Jadidi, Jeffrey M. Walls, Ryan M. Eustice, and Jessy W. Grizzle. "Legged Robot State-Estimation Through Combined Forward Kinematic and Preintegrated Contact Factors". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2018, pp. 1–8. DOI: 10.1109/ICRA.2018.8460748 (page 26).

[118] Stephen Hausler, Ming Xu, Sourav Garg, Punarjay Chakravarty, Shubham Shrivastava, Ankit Vora, and Michael Milford. "Improving Worst Case Visual Localization Coverage via Place-Specific Sub-Selection in Multi-Camera Systems". In: *IEEE Robot. Autom. Lett. (RA-L)* 7.4 (2022), pp. 10112–10119. DOI: 10.1109/LRA.2022.3191174 (page 63).

[119] David Hoeller, Lorenz Wellhausen, Farbod Farshidian, and Marco Hutter. "Learning a State Representation and Navigation in Cluttered and Dynamic Environments". In: *IEEE Robot. Autom. Lett. (RA-L)* 6.3 (2021), pp. 5081–5088. DOI: 10.1109/LRA.2021.3068639 (page 43).

[120] Timon Homberger, Lorenz Wellhausen, Peter Fankhauser, and Marco Hutter. "Support Surface Estimation for Legged Robots". In: *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*. 2019, pp. 8470–8476. DOI: 10.1109/ICRA.2019.8793646 (pages 45, 100).

[121] Armin Hornung, Kai M. Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. "OctoMap: an efficient probabilistic 3D mapping framework based on octrees". In: *Autonomous Robots* 34.3 (2013), pp. 189–206. DOI: 10.1007/s10514-012-9321-0 (page 28).

[122] Andrew Howard, Gaurav S. Sukhatme, and Maja J. Mataric. "Multirobot Simultaneous Localization and Mapping Using Manifold Representations". In: *Proc. of the IEEE* 94.7 (2006), pp. 1360–1369. DOI: 10.1109/JPROC.2006.876922 (pages 27, 38).

[123] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. "Visual Language Maps for Robot Navigation". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2023 (page 140).

[124] Marco Hutter. "StarlETH & Co.. Design and control of legged robots with compliant actuation". en. Doctoral Thesis. Zürich: ETH Zurich, 2013. DOI: 10.3929/ethz-a-009915229 (page 24).

[125] Marco Hutter, Christian Gehring, Andreas Lauber, Fabian Günther, Carmine Dario Bellicoso, Vassilios Tsounis, Peter Fankhauser, Remo Diethelm, Samuel Bachmann, Michael Blösch, Hendrik Kolvenbach, Marko Bjelonic, Linus Isler, and Konrad Meyer. "ANYmal - toward legged robots for harsh environments". In: *Adv. Robotics* 31.17 (2017), pp. 918–931. DOI: 10.1080/01691864.2017.1378591 (page 24).

[126] Jemin Hwangbo, Carmine Dario Bellicoso, Peter Fankhauser, and Marco Hutter. "Probabilistic foot contact estimation by fusing information from dynamics and differential/forward kinematics". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2016, pp. 3872–3878. DOI: 10.1109/IROS.2016.7759570 (page 128).

[127] Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. "Learning agile and dynamic motor skills for legged robots". In: *Sci. Robot.* 4.26 (2019), eaau5872. DOI: 10.1126/scirobotics.aau5872 (page 32).

[128] Eric Hyyppä, Juha Hyyppä, Teemu Hakala, Antero Kukko, Michael A. Wulder, Joanne C. White, Jiri Pyörälä, Xiaowei Yu, Yunsheng Wang, Juho-Pekka Virtanen, Onni Pohjavirta, Xinlian Liang, Markus Holopainen, and Harri Kaartinen. "Under-canopy UAV laser scanning for accurate forest field measurements". en. In: *ISPRS Journal of Photogrammetry and Remote Sensing* 164 (June 2020), pp. 41–60. ISSN: 09242716. DOI: 10.1016/j.isprsjprs.2020.03.021 (page 106).

[129] B. Deniz Ilhan, Aaron M. Johnson, and Daniel E. Koditschek. "Autonomous legged hill ascent". In: *J. Field Robot.* 35.5 (2018), pp. 802–832. DOI: doi.org/10.1002/rob.21779 (page 51).

[130] Larry D. Jackel, Eric Krotkov, Michael Perschbacher, James Pippine, and Chad Sullivan. "The DARPA LAGR program: Goals, Challenges, Methodology, and Phase I Results". In: *J. Field Robot.* 23.11-12 (2006), pp. 945–973. DOI: 10.1002/rob.20161 (page 46).

[131] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, Ayush Tewari, Joshua B. Tenenbaum, Celso Miguel de Melo, Madhava Krishna, Liam Paull, Florian Shkurti, and Antonio Torralba. "ConceptFusion: Open-set Multimodal 3D Mapping". In: *Robotics: Science and Systems (RSS)*. 2023 (pages 140, 143).

[132] Edo Jelavic, Dominic Jud, Pascal Egli, and Marco Hutter. "Robotic Precision Harvesting: Mapping, Localization, Planning and Control for a Legged Tree Harvester". In: *Field Robotics* (2021) (page 106).

[133] Edo Jelavic, Tun Kapgen, Simon Kerscher, Dominic Jud, and Marco Hutter. "Harveri : A Small (Semi-)Autonomous Precision Tree Harvester". en. In: *Innovation in Forestry Robotics: Research and Industry Adoption (ICRA 2022)*. May 2022. DOI: 10.3929/ETHZ-B-000549974 (page 106).

[134] Tianchen Ji, Arun Narenthiran Sivakumar, Girish Chowdhary, and Katherine Driggs-Campbell. "Proactive Anomaly Detection for Robot Navigation With Multi-Sensor Fusion". In: *IEEE Robot. Autom. Lett. (RA-L)* 7.2 (2022), pp. 4975–4982. DOI: 10.1109/LRA.2022.3153989 (pages 47, 134).

[135] Aaron M. Johnson, Matthew T. Hale, G. C. Haynes, and Daniel E. Koditschek. "Autonomous legged hill and stairwell ascent". In: *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. 2011, pp. 134–142. DOI: 10.1109/SSRR.2011.6106785 (page 40).

[136] Michael Kaess, Hordur Johannsson, Richard Roberts, Viorela Ila, John J. Leonard, and Frank Dellaert. "iSAM2: Incremental smoothing and mapping using the Bayes tree". In: *Intl. J. of Robot. Res.* 31.2 (2012), pp. 216–235. DOI: 10.1177/0278364911430419 (page 109).

[137] Michael Kaess, Ananth Ranganathan, and Frank Dellaert. "iSAM: Incremental Smoothing and Mapping". In: *IEEE Trans. Robot.* 24.6 (2008), pp. 1365–1378. DOI: 10.1109/TRO.2008.2006706 (page 12).

[138] Gregory Kahn, Pieter Abbeel, and Sergey Levine. "BADGR: An Autonomous Self-Supervised Learning-Based Navigation System". In: *IEEE Robot. Autom. Lett. (RA-L)* 6.2 (2021), pp. 1312–1319. DOI: 10.1109/LRA.2021.3057023 (pages 31, 43, 144).

[139] Shuuji Kajita and Christian Ott. "Limbed Systems". In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Springer Handbooks. Springer, 2016, pp. 419–442. DOI: 10.1007/978-3-319-32552-1_17 (page 23).

[140] Daniel Kappler, Franziska Meier, Jan Issac, Jim Mainprice, Cristina Garcia Cifuentes, Manuel Wuthrich, Vincent Berenz, Stefan Schaal, Nathan D. Ratliff, and Jeannette Bohg. "Real-Time Perception Meets Reactive Motion Generation". In: *IEEE Robot. Autom. Lett. (RA-L)* 3.3 (2018), pp. 1864–1871. DOI: 10.1109/LRA.2018.2795645 (page 76).

[141] Lydia E. Kavraki, Petr Svestka, Jean-Claude Latombe, and Mark H. Overmars. "Probabilistic roadmaps for path planning in high-dimensional configuration spaces". In: *IEEE Trans. Robotics Autom.* 12.4 (1996), pp. 566–580. DOI: 10.1109/70.508439 (page 30).

[142] James R. Kellner, John Armston, Markus Birrer, K. C. Cushman, Laura Duncanson, Christoph Eck, Christoph Falleger, Benedikt Imbach, Kamil Král, Martin Krůček, Jan Trochta, Tomáš Vrška, and Carlo Zgraggen. "New Opportunities for Forest Remote Sensing Through Ultra-High-Density Drone Lidar". en. In: *Surveys in Geophysics* 40.4 (July 2019), pp. 959–977. ISSN: 0169-3298, 1573-0956. DOI: 10.1007/s10712-019-09529-9 (page 105).

[143] Alex Kendall, Matthew Grimes, and Roberto Cipolla. "PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization". In: *Intl. Conf. on Computer Vision (ICCV)*. 2015, pp. 2938–2946. DOI: 10.1109/ICCV.2015.336 (page 36).

[144] Oussama Khatib. "Real-time obstacle avoidance for manipulators and mobile robots". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 1985, pp. 500–505. DOI: 10.1109/ROBOT.1985.1087247 (page 31).

[145] Shehryar Khattak, Huan Nguyen, Frank Mascarich, Tung Dang, and Kostas Alexis. "Complementary Multi–Modal Sensor Fusion for Resilient Robot Pose Estimation in Subterranean Environments". In: *Int. Conf. on Unmanned Aircraft Systems (ICUAS)*. 2020, pp. 1024–1029. DOI: 10.1109/ICUAS48674.2020.9213865 (page 118).

[146] Donghyun Kim, D. Carballo, Jared Di Carlo, Benjamin Katz, Gerardo Bledt, Bryan Lim, and Sangbae Kim. "Vision Aided Dynamic Exploration of Unstructured Terrain with a Small-Scale Quadruped Robot". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2020, pp. 2464–2470. DOI: 10.1109/ICRA40945.2020.9196777 (page 40).

[147] Dongshin Kim, Jie Sun, Sang Min Oh, J.M. Rehg, and A.F. Bobick. "Traversability Classification using Unsupervised On-line Visual Learning for Outdoor Robot Navigation". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2006, pp. 518–525. DOI: 10.1109/ROBOT.2006.1641763 (page 46).

[148] Yunho Kim, Chanyoung Kim, and Jemin Hwangbo. "Learning Forward Dynamics Model and Informed Trajectory Sampler for Safe Quadruped Navigation". In: *Robotics: Science and Systems (RSS)*. 2022. DOI: 10.48550/arXiv.2204.08647 (pages 42, 43).

[149] Diederick P Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *Intl. Conf. on Learning Representations (ICLR)*. 2015 (page 15).

[150] Hiroaki Kitano, Minoru Asada, Yasuo Kuniyoshi, Itsuki Noda, Eiichi Osawa, and Hitoshi Matsubara. "RoboCup: A Challenge Problem for AI". In: *AI Mag.* 18.1 (1997), pp. 73–85. DOI: 10.1609/aimag.v18i1.1276 (page 49).

[151] Daniel E. Koditschek. "Exact robot navigation by means of potential functions: Some topological considerations". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 1987, pp. 1–6. DOI: 10.1109/ROBOT.1987.1088038 (page 140).

[152] Lian Pin Koh and Serge A. Wich. "Dawn of Drone Ecology: Low-Cost Autonomous Aerial Vehicles for Conservation". In: *Tropical Conservation Science* 5.2 (June 2012), pp. 121–132. ISSN: 1940-0829, 1940-0829. DOI: 10.1177/194008291200500202 (page 105).

[153] Hendrik Kolvenbach, David Wisth, Russell Buchanan, Giorgio Valsecchi, Ruben Grandia, Maurice F. Fallon, and Marco Hutter. "Towards autonomous inspection of concrete deterioration in sewers with legged robots". In: *J. Field Robot.* 37.8 (2020), pp. 1314–1327. DOI: 10.1002/rob.21964 (page 50).

[154] Navinda Kottege et al. *Heterogeneous robot teams with unified perception and autonomy: How Team CSIRO Data61 tied for the top score at the DARPA Subterranean Challenge*. 2023. arXiv: 2302.13230 [cs.RO] (pages 50, 141).

[155] Tomas Krajnik, Filip Majer, Lucie Halodova, and Tomas Vintr. "Navigation without localisation: Reliable teach and repeat based on the convergence theorem". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2018, pp. 1657–1664. ISBN: 9781538680940. DOI: 10.1109/IROS.2018.8593803. eprint: 1711.05348 (page 37).

[156] Eric Krotkov, Douglas Hackett, Lawrence D. Jackel, Michael Perschbacher, James Pippine, Jesse Strauss, Gill A. Pratt, and Christopher Orlowski. "The DARPA Robotics Challenge Finals: Results and Perspectives". In: *J. Field Robot.* 34.2 (2017), pp. 229–240. DOI: 10.1002/rob.21683 (page 49).

[157] "Marteloscopes as training tools for the retention and conservation of habitat trees in forests". In: *Schweizerische Zeitschrift für Forstwesen* (2019). Ed. by Frank Krumm, Thibault Lachat, Andreas Schuck, Rita Bütler Sauvain, and Daniel Kraus. DOI: 10.3188/szf.2019.0086 (page 116).

[158] Philipp Krüsi, Bastian Bücheler, François Pomerleau, Ulrich Schwesinger, Roland Siegwart, and Paul Timothy Furgale. "Lighting-invariant Adaptive Route Following Using Iterative Closest Point Matching". In: *J. Field Robot.* 32.4 (2015), pp. 534–564. DOI: 10.1002/rob.21524 (page 76).

[159] Benjamin Kuipers. "Modeling Spatial Knowledge". In: *Intl. Joint Conf. on Artificial Intelligence*. Ed. by Raj Reddy. William Kaufmann, 1977, pp. 292–298 (page 36).

[160] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. "RMA: Rapid Motor Adaptation for Legged Robots". In: *Robotics: Science and Systems (RSS)*. Ed. by Dylan A. Shell, Marc Toussaint, and M. Ani Hsieh. 2021. DOI: 10.15607/RSS.2021.XVII.011 (pages 1, 2, 143).

[161] Juichung Kuo, Manasi Muglikar, Zichao Zhang, and Davide Scaramuzza. "Redesigning SLAM for Arbitrary Multi-Camera Systems". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2020, pp. 2116–2122. DOI: 10.1109/ICRA40945.2020.9197553 (page 13).

[162] Mathieu Labbé and François Michaud. "RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation". In: *J. Field Robot.* 36.2 (2019), pp. 416–446. DOI: 10.1002/rob.21831 (page 34).

[163] Steven M. LaValle. *Rapidly-exploring random trees : a new tool for path planning*. Tech. rep. TR 98-11. Computer Science Dept., Iowa State University, 1998 (page 29).

[164] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. "Learning quadrupedal locomotion over challenging terrain". In: *Sci. Robot.* 5.47 (2020), p. 5986. DOI: 10.1126/scirobotics.abc5986 (pages 2, 32, 143).

[165] Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz Rodríguez. "Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges". In: *Inf. Fusion* 58 (2020), pp. 52–68. DOI: 10.1016/j.inffus.2019.12.004 (page 101).

[166] Stefan Leutenegger, Margarita Chli, and Roland Siegwart. "BRISK: Binary Robust invariant scalable keypoints". In: *Intl. Conf. on Computer Vision (ICCV)*. Ed. by Dimitris N. Metaxas, Long Quan, Alberto Sanfeliu, and Luc Van Gool. 2011, pp. 2548–2555. DOI: 10.1109/ICCV.2011.6126542 (page 35).

[167] Stefan Leutenegger, Paul Timothy Furgale, Vincent Rabaud, Margarita Chli, Kurt Konolige, and Roland Siegwart. "Keyframe-Based Visual-Inertial SLAM using Nonlinear Optimization". In: *Robotics: Science and Systems (RSS)*. Ed. by Paul Newman, Dieter Fox, and David Hsu. 2013. DOI: 10.15607/RSS.2013.IX.037 (page 34).

[168] Anqiao Li, Chenyu Yang, Jonas Frey, Joonho Lee, Cesar Cadena, and Marco Hutter. "Seeing Through the Grass: Semantic Pointcloud Filter for Support Surface Learning". In: *IEEE Robot. Autom. Lett. (RA-L)* 8.11 (2023), pp. 7687–7694. DOI: 10.1109/LRA.2023.3320016 (page 100).

[169] Xinlian Liang, Ville Kankare, Juha Hyyppä, Yunsheng Wang, Antero Kukko, Henrik Haggrén, Xiaowei Yu, Harri Kaartinen, Anttoni Jaakkola, Fengying Guan, Markus Holopainen, and Mikko Vastaranta. "Terrestrial laser scanning in forest inventories". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 115 (2016), pp. 63–77. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2016.01.006 (pages 104, 105).

[170] Yi Lin, Juha Hyyppa, and Anttoni Jaakkola. "Mini-UAV-Borne LIDAR for Fine-Scale Mapping". en. In: *IEEE Geoscience and Remote Sensing Letters* 8.3 (May 2011), pp. 426–430. ISSN: 1545-598X, 1558-0571. DOI: 10.1109/LGRS.2010.2079913 (page 106).

[171] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. "Pixel-Perfect Structure-from-Motion with Featuremetric Refinement". In: *Intl. Conf. on Computer Vision (ICCV)*. 2021, pp. 5967–5977. DOI: 10.1109/ICCV48922.2021.00593 (page 36).

[172] Chris Linegar. "Vision-only localisation under extreme appearance change". en. Doctoral Thesis. United Kingdom: University of Oxford, 2016 (page 38).

[173] Chris Linegar, Winston Churchill, and Paul Newman. "Made to measure: Bespoke landmarks for 24-hour, all-weather localisation with a camera". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2016, pp. 787–794. DOI: 10.1109/ICRA.2016.7487208 (page 39).

[174] Chris Linegar, Winston Churchill, and Paul Newman. "Work smart, not hard: Recalling relevant experiences for vast-scale but time-constrained localisation". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. June. IEEE, 2015, pp. 90–97. ISBN: 9781479969234. DOI: 10.1109/ICRA.2015.7138985 (pages 38, 39).

[175] Xu Liu, Guilherme V. Nardari, Fernando Cladera Ojeda, Yuezhan Tao, Alex Zhou, Thomas Donnelly, Chao Qu, Steven W. Chen, Roseli A. F. Romero, Camillo J. Taylor, and Vijay Kumar. "Large-Scale Autonomous Flight With Real-Time Semantic SLAM Under Dense Forest Canopy". In: *IEEE Robot. Autom. Lett. (RA-L)* 7.2 (2022), pp. 5512–5519. DOI: 10.1109/LRA.2022.3154047 (page 106).

[176] Kenzo Lobos-Tsunekawa, Francisco Leiva, and Javier Ruiz-del-Solar. "Visual Navigation for Biped Humanoid Robots Using Deep Reinforcement Learning". In: *IEEE Robot. Autom. Lett. (RA-L)* 3.4 (2018), pp. 3247–3254. DOI: 10.1109/LRA.2018.2851148 (page 43).

[177] David G. Lowe. "Distinctive image features from scale-invariant keypoints". In: *Intl. J. of Computer Vision* 60.2 (2004), pp. 91–110. DOI: 10.1023/B:VISI.0000029664.99615.94 (pages 35, 142).

[178] Stephanie M. Lowry, Niko Sünderhauf, Paul Newman, John J. Leonard, David D. Cox, Peter I. Corke, and Michael J. Milford. "Visual Place Recognition: A Survey". In: *IEEE Trans. Robot.* 32.1 (2016), pp. 1–19. DOI: 10.1109/TRO.2015.2496823 (page 34).

[179] Howard William Lull. *Soil Compaction on Forest and Range Lands*. en. Forest Service, U.S. Department of Agriculture, 1959 (page 104).

[180] Kevin M. Lynch and Frank C. Park. *Modern Robotics*. Cambridge University Press, 2017. ISBN: 9781107156302 (page 2).

[181] Simon Lynen, Michael Bosse, Paul Timothy Furgale, and Roland Siegwart. "Placeless Place-Recognition". In: *IEEE Intl. Conf. on 3D Vision*. 2014, pp. 303–310. DOI: 10.1109/3DV.2014.36 (page 37).

[182] Simon Lynen, Torsten Sattler, Michael Bosse, Joel A. Hesch, Marc Pollefeys, and Roland Siegwart. "Get Out of My Lab: Large-scale, Real-Time Visual-Inertial Localization". In: *Robotics: Science and Systems (RSS)*. Ed. by Lydia E. Kavraki, David Hsu, and Jonas Buchli. 2015. DOI: 10.15607/RSS.2015.XI.037 (page 36).

[183] Qin Ma, Yanjun Su, and Qinghua Guo. "Comparison of canopy cover estimations from airborne LiDAR, aerial imagery, and satellite imagery". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 10.9 (2017), pp. 4225–4236 (page 104).

[184] Yi Ma, Stefano Soatto, Jana Košecká, and S. Shankar Sastry. "Introduction". In: *An Invitation to 3-D Vision: From Images to Geometric Models*. New York, NY: Springer New York, 2004. ISBN: 978-0-387-21779-6. DOI: 10.1007/978-0-387-21779-6_1 (page 17).

[185] J. MacQueen. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. Vol. 5.1. University of California Press, Jan. 1967, pp. 281–298 (page 114).

[186] Kirk MacTavish, Michael Paton, and Timothy D. Barfoot. "Selective memory: Recalling relevant experience for long-term visual localization". In: *J. Field Robot.* 35.8 (2018), pp. 1265–1292. ISSN: 15564967. DOI: 10.1002/rob.21838 (page 39).

[187] Carlos Mastalli, Rohan Budhiraja, Wolfgang Merkt, Guilhem Saurel, Bilal Hammoud, Maximilien Naveau, Justin Carpentier, Ludovic Righetti, Sethu Vijayakumar, and Nicolas Mansard. "Crocoddyl: An Efficient and Versatile Framework for Multi-Contact Optimal Control". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2020, pp. 2536–2542. DOI: 10.1109/ICRA40945.2020.9196673 (page 31).

[188] Matias Mattamala. *Reducing the uncertainty about the uncertainties, part 2: Frames and manifolds*. https://gtsam.org/2021/02/23/uncertainties-part2.html. [Online; accessed 10-Feb-2023]. 2021 (page 8).

[189] Matías Mattamala, Nived Chebrolu, and Maurice F. Fallon. "An Efficient Locally Reactive Controller for Safe Navigation in Visual Teach and Repeat Missions". In: *IEEE Robot. Autom. Lett. (RA-L)* 7.2 (2022), pp. 2353–2360. DOI: 10.1109/LRA.2022.3143196 (pages 4, 66, 111).

[190] Matías Mattamala, Milad Ramezani, Marco Camurri, and Maurice F. Fallon. "Learning Camera Performance Models for Active Multi-Camera Visual Teach and Repeat". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2021, pp. 14346–14352. DOI: 10.1109/ICRA48506.2021.9561650 (pages 4, 54).

[191] Wim Meeussen. *Rep 105 — Coordinate Frames for Mobile Platforms*. https://www.ros.org/reps/rep-0105.html. [Online; accessed 17-Aug-2023]. 2010 (page 9).

[192] Kamilo Melo, Tomislav Horvat, and Auke J. Ijspeert. "Animal robots in the African wilderness: Lessons learned and outlook for field robotics". In: *Sci. Robot.* 8.85 (2023), eadd8662. DOI: 10.1126/scirobotics.add8662 (pages 51, 106).

[193] Kamilo Melo, Tomislav Horvat, and Auke J Ijspeert. "K-rock, a bio-robot outside the lab, back in nature". In: *International Symposium on Adaptive Motion of Animals and Machines (AMAM)*. 2017 (pages 51, 106).

[194] Xiangyun Meng, Nathan D. Ratliff, Yu Xiang, and Dieter Fox. "Neural Autonomous Navigation with Riemannian Motion Policy". In: (2019), pp. 8860–8866. DOI: 10.1109/ICRA.2019.8794223 (page 76).

[195] Jeff Michels, Ashutosh Saxena, and Andrew Y. Ng. "High speed obstacle avoidance using monocular vision and reinforcement learning". In: *Intl. Conf. on Machine Learning (ICML)*. Ed. by Luc De Raedt and Stefan Wrobel. Vol. 119. ACM International Conference Proceeding Series. 2005, pp. 593–600. DOI: 10.1145/1102351.1102426 (page 43).

[196] Mikko Miettinen, Matti Ohman, Arto Visala, and Pekka Forsman. "Simultaneous Localization and Mapping for Forest Harvesters". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2007, pp. 517–522. DOI: 10.1109/ROBOT.2007.363838 (pages 105, 106).

[197] Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. "Learning Robust Perceptive Locomotion for Quadrupedal Robots in the Wild". In: *Sci. Robot.* 7.62 (2022). DOI: 10.1126/scirobotics.abk2822 (pages 1, 2, 32, 106, 112, 143).

[198] Takahiro Miki, Lorenz Wellhausen, Ruben Grandia, Fabian Jenelten, Timon Homberger, and Marco Hutter. "Elevation Mapping for Locomotion and Navigation using GPU". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2022, pp. 2273–2280. DOI: 10.1109/IROS47612.2022.9981507 (page 110).

[199] Michael Milford and Gordon F. Wyeth. "SeqSLAM: Visual route-based navigation for sunny summer days and stormy winter nights". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2012, pp. 1643–1649. DOI: 10.1109/ICRA.2012.6224623 (page 37).

[200] Dmytro Mishkin. "Learning and Crafting for the Wide Multiple Baseline Stereo". en. Doctoral Thesis. Czech Republic: Czech Technical University, 2021 (page 35).

[201] Hans P. Moravec. "The Stanford Cart and the CMU Rover". In: *Proc. of the IEEE* 71 (1983), pp. 872–884 (pages 33, 44).

[202] Mustafa Mukadam, Ching-An Cheng, Dieter Fox, Byron Boots, and Nathan D. Ratliff. "Riemannian Motion Policy Fusion through Learnable Lyapunov Function Reshaping". In: *Conf. on Robot Learning (CoRL)*. Vol. 100. Proceedings of Machine Learning Research. 2019, pp. 204–219 (page 77).

[203] Mustafa Mukadam, Jing Dong, Xinyan Yan, Frank Dellaert, and Byron Boots. "Continuous-time Gaussian process motion planning via probabilistic inference". In: *Intl. J. of Robot. Res.* 37.11 (2018). DOI: 10.1177/0278364918790369 (pages 30, 77).

[204] Raul Mur-Artal, J. M. M. Montiel, and Juan D. Tardós. "ORB-SLAM: A Versatile and Accurate Monocular SLAM System". In: *IEEE Trans. Robot.* 31.5 (2015), pp. 1147–1163. DOI: 10.1109/TRO.2015.2463671 (pages 28, 34, 142).

[205] Raul Mur-Artal and Juan D. Tardós. "ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras". In: *IEEE Trans. Robot.* 33.5 (2017), pp. 1255–1262. DOI: 10.1109/TRO.2017.2705103 (page 34).

[206] Raul Mur-Artal and Juan D. Tardós. "Visual-Inertial Monocular SLAM With Map Reuse". In: *IEEE Robot. Autom. Lett. (RA-L)* 2.2 (2017), pp. 796–803. DOI: 10.1109/LRA.2017.2653359 (pages 34, 36).

[207] Akihiro Nakamura, Roger L. Kitching, Min Cao, Thomas J. Creedy, Tom M. Fayle, Martin Freiberg, C.N. Hewitt, Takao Itioka, Lian Pin Koh, Keping Ma, Yadvinder Malhi, Andrew Mitchell, Vojtech Novotny, Claire M.P. Ozanne, Liang Song, Han Wang, and Louise A. Ashton. "Forests and Their Canopies: Achievements and Horizons in Canopy Science". In: *Trends in Ecology & Evolution* 32.6 (2017), pp. 438–451. ISSN: 0169-5347. DOI: https://doi.org/10.1016/j.tree.2017.02.020 (page 105).

[208] Peer Neubert, Stefan Schubert, and Peter Protzel. "An Introduction to Hyperdimensional Computing for Robotics". In: *Künstliche Intell.* 33.4 (2019), pp. 319–330. DOI: 10.1007/s13218-019-00623-z (page 37).

[209] Andrew Y. Ng and Stuart Russell. "Algorithms for Inverse Reinforcement Learning". In: *Intl. Conf. on Machine Learning (ICML)*. 2000, pp. 663–670 (page 48).

[210] Joseph Norby and Aaron M. Johnson. "Fast Global Motion Planning for Dynamic Legged Robots". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2020, pp. 3829–3836. DOI: 10.1109/IROS45743.2020.9341438 (page 40).

[211] Jyothish Pari, Nur Muhammad (Mahi) Shafiullah, Sridhar Pandian Arunachalam, and Lerrel Pinto. "The Surprising Effectiveness of Representation Learning for Visual Imitation". In: *Robotics: Science and Systems (RSS)*. Ed. by Kris Hauser, Dylan A. Shell, and Shoudong Huang. 2022. DOI: 10.15607/RSS.2022.XVIII.010 (page 140).

[212] Geoffrey Pascoe, William P. Maddern, Alexander D. Stewart, and Paul Newman. "FARLAP: Fast robust localisation using appearance priors". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2015, pp. 6366–6373. DOI: 10.1109/ICRA.2015.7140093 (page 35).

[213] Michael Paton, Kirk MacTavish, Michael Warren, and Timothy D. Barfoot. "Bridging the appearance gap: Multi-experience localization for long-term visual teach and repeat". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2016, pp. 1918–1925. DOI: 10.1109/IROS.2016.7759303 (page 39).

[214] Songyou Peng, Kyle Genova, Chiyu "Max" Jiang, Andrea Tagliasacchi, Marc Pollefeys, and Thomas Funkhouser. "OpenScene: 3D Scene Understanding with Open Vocabularies". In: *IEEE Int. Conf. Computer Vision and Pattern Recognition*. 2023 (page 140).

[215] Mark Pfeiffer, Michael Schaeuble, Juan I. Nieto, Roland Siegwart, and Cesar Cadena. "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2017, pp. 1527–1533. DOI: 10.1109/ICRA.2017.7989182 (page 43).

[216] Marek Pierzchała, Philippe Giguère, and Rasmus Astrup. "Mapping forests using an unmanned ground vehicle with 3D LiDAR and graph-SLAM". en. In: *Computers and Electronics in Agriculture* 145 (Feb. 2018), pp. 217–225. ISSN: 01681699. DOI: 10.1016/j.compag.2017.12.034 (page 106).

[217] Luis Pineda, Taosha Fan, Maurizio Monge, Shobha Venkataraman, Paloma Sodhi, Ricky T. Q. Chen, Joseph Ortiz, Daniel DeTone, Austin S. Wang, Stuart Anderson, Jing Dong, Brandon Amos, and Mustafa Mukadam. "Theseus: A Library for Differentiable Nonlinear Optimization". In: *Intl. Conf. on Neural Information Processing Systems (NeurIPS)*. 2022 (page 142).

[218] Dean Pomerleau. "ALVINN: An Autonomous Land Vehicle in a Neural Network". In: *Intl. Conf. on Neural Information Processing Systems (NeurIPS)*. 1988, pp. 305–313 (page 42).

[219] Dean Pomerleau. "Input Reconstruction Reliability Estimation". In: *Intl. Conf. on Neural Information Processing Systems (NeurIPS)*. 1992, pp. 279–286 (page 47).

[220] Alexander Proudman, Milad Ramezani, Sundara Tejaswi Digumarti, Nived Chebrolu, and Maurice Fallon. "Towards real-time forest inventory using handheld LiDAR". In: *Robotics and Autonomous Systems* 157 (2022), p. 104240. ISSN: 0921-8890. DOI: https://doi.org/10.1016/j.robot.2022.104240 (page 105).

[221] Feifei Qian, Douglas Jerolmack, Nicholas Lancaster, George Nikolich, Paul Reverdy, Sonia Roberts, Thomas Shipley, R. Scott Van Pelt, Ted M. Zobeck, and Daniel E. Koditschek. "Ground robotic measurement of aeolian processes". In: *Aeolian Research* 27 (2017), pp. 1–11. ISSN: 1875-9637. DOI: 10.1016/j.aeolia.2017.04.004 (pages 51, 106).

[222] Feifei Qian and Daniel E. Koditschek. "An obstacle disturbance selection framework: emergent robot steady states under repeated collisions". In: *Intl. J. of Robot. Res.* 39.13 (2020). DOI: 10.1177/0278364920935514 (page 143).

[223] Feifei Qian, Dylan Lee, George Nikolich, Daniel Koditschek, and Douglas Jerolmack. "Rapid In Situ Characterization of Soil Erodibility With a Field Deployable Robot". In: *Journal of Geophysical Research: Earth Surface* 124.5 (2019), pp. 1261–1280. DOI: 10.1029/2018JF004887 (pages 51, 106).

[224] Sean Quinlan and Oussama Khatib. "Elastic Bands: Connecting Path Planning and Control". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 1993, pp. 802–807. DOI: 10.1109/ROBOT.1993.291936 (page 141).

[225] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. "Learning Transferable Visual Models From Natural Language Supervision". In: *Intl. Conf. on Machine Learning (ICML)*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 8748–8763 (page 143).

[226] Milad Ramezani, Matias Mattamala, and Maurice Fallon. "AEROS: AdaptivE RObust Least-Squares for Graph-Based SLAM". In: *Frontiers in Robotics and AI* 9 (2022). ISSN: 2296-9144. DOI: 10.3389/frobt.2022.789444 (page 5).

[227] Milad Ramezani, Georgi Tinchev, Egor Iuganov, and Maurice Fallon. "Online LiDAR-SLAM for Legged Robots with Robust Registration and Deep-Learned Loop Closure". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2020, pp. 4158–4164. DOI: 10.1109/ICRA40945.2020.9196769 (page 109).

[228] 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 Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2020, pp. 4353–4360. DOI: 10.1109/IROS45743.2020.9340849 (page 4).

[229] Fabio Tozeto Ramos and Lionel Ott. "Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent". In: *Robotics: Science and Systems (RSS)*. 2015. DOI: 10.15607/RSS.2015.XI.002 (page 28).

[230] Nathan D. Ratliff, J. Andrew Bagnell, and Martin A. Zinkevich. "Maximum Margin Planning". In: *Intl. Conf. on Machine Learning (ICML)*. Vol. 148. ACM International Conference Proceeding Series. 2006, 729–736. DOI: 10.1145/1143844.1143936 (page 48).

[231] Nathan D. Ratliff, Jan Issac, and Daniel Kappler. "Riemannian Motion Policies". In: *CoRR* abs/1801.02854 (2018). arXiv: 1801.02854 (pages 31, 142).

[232] 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 Int. Conf. Robot. Autom. (ICRA)*. 2016, pp. 4304–4311. DOI: 10.1109/ICRA.2016.7487628 (pages 18, 22).

[233] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. "R2D2: Reliable and Repeatable Detector and Descriptor". In: *Intl. Conf. on Neural Information Processing Systems (NeurIPS)*. Vol. 32. 2019, pp. 12405–12415 (page 35).

[234] Charles Richter and Nicholas Roy. "Safe Visual Navigation via Deep Learning and Novelty Detection". In: *Robotics: Science and Systems (RSS)*. 2017. DOI: 10.15607/RSS.2017.XIII.064 (pages 42, 47).

[235] Juergen Rossmann, Michael Schluse, Christian Schlette, Arno Buecken, Petra Krahwinkler, and Markus Emde. "Realization of a highly accurate mobile robot system for multi purpose precision forestry applications". In: *2009 International Conference on Advanced Robotics*. June 2009, pp. 1–6 (page 106).

[236] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R. Bradski. "ORB: An efficient alternative to SIFT or SURF". In: *Intl. Conf. on Computer Vision (ICCV)*. 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544 (page 35).

[237] Nikita Rudin, David Hoeller, Marko Bjelonic, and Marco Hutter. "Advanced Skills by Learning Locomotion and Local Navigation End-to-End". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2022, pp. 2497–2503. DOI: 10.1109/IROS47612.2022.9981198 (pages 43, 144).

[238] Fabio Ruetz, Nicholas Lawrance, Emili Hernández, Paulo Borges, and Thierry Peynot. *ForestTrav: Accurate, Efficient and Deployable Forest Traversability Estimation for Autonomous Ground Vehicles*. 2023. DOI: 10.48550/arXiv.2305.12705. arXiv: 2305.12705 (page 45).

[239] Radu Bogdan Rusu. "Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments". en. In: *KI - Künstliche Intelligenz* 24.4 (Nov. 2010), pp. 345–348. ISSN: 0933-1875, 1610-1987. DOI: 10.1007/s13218-010-0059-6 (page 114).

[240] Uluc Saranli, Martin Buehler, and Daniel E. Koditschek. "RHex: A Simple and Highly Mobile Hexapod Robot". In: *Intl. J. of Robot. Res.* 20.7 (2001), pp. 616–631. DOI: 10.1177/02783640122067570 (page 51).

[241] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. "From Coarse to Fine: Robust Hierarchical Localization at Large Scale". In: *IEEE Int. Conf. Computer Vision and Pattern Recognition*. 2019, pp. 12716–12725. DOI: 10.1109/CVPR.2019.01300 (page 36).

[242] Paul-Edouard Sarlin, Ajaykumar Unagar, Måns Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, and Torsten Sattler. "Back to the Feature: Learning Robust Camera Localization From Pixels To Pose". In: *IEEE Int. Conf. Computer Vision and Pattern Recognition*. 2021, pp. 3247–3257. DOI: 10.1109/CVPR46437.2021.00326 (page 36).

[243] Adarsh Jagan Sathyamoorthy, Kasun Weerakoon, Tianrui Guan, Jing Liang, and Dinesh Manocha. "TerraPN: Unstructured Terrain Navigation using Online Self-Supervised Learning". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2022, pp. 7197–7204. DOI: 10.1109/IROS47612.2022.9981942 (page 47).

[244] D. C. Schedl, I. Kurmi, and O. Bimber. "An autonomous drone for search and rescue in forests using airborne optical sectioning". In: *Sci. Robot.* 6.55 (2021), eabg1188. DOI: 10.1126/scirobotics.abg1188 (page 105).

[245] Thomas Schneider, Marcin Dymczyk, Marius Fehr, Kevin Egger, Simon Lynen, Igor Gilitschenski, and Roland Siegwart. "Maplab: An Open Framework for Research in Visual-Inertial Mapping and Localization". In: *IEEE Robot. Autom. Lett. (RA-L)* 3.2 (2018), pp. 1418–1425. DOI: 10.1109/LRA.2018.2800113 (page 34).

[246] Johannes L Schonberger and Jan-Michael Frahm. "Structure-from-motion revisited". In: *IEEE Int. Conf. Computer Vision and Pattern Recognition*. 2016, pp. 4104–4113. DOI: 10.1109/CVPR.2016.445 (pages 34, 142).

[247] Stefan Schubert, Peer Neubert, Sourav Garg, Michael Milford, and Tobias Fischer. "Visual Place Recognition: A Tutorial". In: *arXiv preprint arXiv:2303.03281* (2023) (page 37).

[248] Janine Schweier, Natascia Magagnotti, Eric R. Labelle, and Dimitris Athanassiadis. "Sustainability Impact Assessment of Forest Operations: a Review". In: *Current Forestry Reports* 5.3 (2019), pp. 101–113. DOI: 10.1007/s40725-019-00091-6 (page 103).

[249] Marina Segura, Duncan Ray, and Concepción Maroto. "Decision support systems for forest management: A comparative analysis and assessment". en. In: *Computers and Electronics in Agriculture* 101 (Feb. 2014), pp. 55–67. ISSN: 01681699. DOI: 10.1016/j.compag.2013.12.005 (page 103).

[250] Jordy Sehn, Yuchen Wu, and Timothy D. Barfoot. "Along Similar Lines: Local Obstacle Avoidance for Long-term Autonomous Path Following". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2023. DOI: 10.48550/arXiv.2211.02047 (page 76).

[251] Junwon Seo, Taekyung Kim, Kiho Kwak, Jihong Min, and Inwook Shim. "ScaTE: A Scalable Framework for Self- Supervised Traversability Estimation in Unstructured Environments". In: *IEEE Robot. Autom. Lett. (RA-L)* 8.2 (2023), pp. 888–895. DOI: 10.1109/LRA.2023.3234768 (page 48).

[252] Junwon Seo, Sungdae Sim, and Inwook Shim. "Learning Off-Road Terrain Traversability with Self-Supervisions Only". In: *IEEE Robot. Autom. Lett. (RA-L)* (2023), pp. 1–8. DOI: 10.1109/LRA.2023.3284356 (page 47).

[253] J A Sethian. "A Fast Marching Level Set Method for Monotonically Advancing Fronts". In: *Proc. of the National Academy of Sciences (PNAS)* 93.4 (1996), pp. 1591–1595. DOI: 10.1073/pnas.93.4.1591 (page 29).

[254] Amirreza Shaban, Xiangyun Meng, JoonHo Lee, Byron Boots, and Dieter Fox. "Semantic Terrain Classification for Off-Road Autonomous Driving". In: *Conf. on Robot Learning (CoRL)*. Vol. 164. Proceedings of Machine Learning Research. 2022, pp. 619–629 (page 46).

[255] Kunal Shah, Grant Ballard, Annie Schmidt, and Mac Schwager. "Multidrone aerial surveys of penguin colonies in Antarctica". In: *Sci. Robot.* 5.47 (2020), eabc3000. DOI: 10.1126/scirobotics.abc3000 (page 105).

[256] Gabe Sibley, Larry H. Matthies, and Gaurav S. Sukhatme. "Sliding window filter with application to planetary landing". In: *J. Field Robot.* 27.5 (2010), pp. 587–608. DOI: 10.1002/rob.20360 (page 142).

[257] Jonah Siekmann, Kevin R. Green, John Warila, Alan Fern, and Jonathan W. Hurst. "Blind Bipedal Stair Traversal via Sim-to-Real Reinforcement Learning". In: *Robotics: Science and Systems (RSS)*. 2021. DOI: 10.15607/RSS.2021.XVII.061 (page 32).

[258] Saul Simhon and Gregory Dudek. "A global topological map formed by local metric maps". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. Vol. 3. 1998, 1708–1714 vol.3. DOI: 10.1109/IROS.1998.724844 (page 38).

[259] Boris Sofman, Ellie Lin, J. Andrew Bagnell, Nicolas Vandapel, and Anthony Stentz. "Improving Robot Navigation Through Self-Supervised Online Learning". In: *Robotics: Science and Systems (RSS)*. 2006. DOI: 10.15607/RSS.2006.II.004 (page 46).

[260] Joan Solà, Jeremie Deray, and Dinesh Atchuthan. "A Micro Lie Theory for State Estimation in Robotics". In: *arXiv* (2018) (page 8).

[261] Cyrill Stachniss, John J. Leonard, and Sebastian Thrun. "Simultaneous Localization and Mapping". In: *Springer Handbook of Robotics*. Cham: Springer International Publishing, 2016, pp. 1153–1176. ISBN: 978-3-319-32552-1. DOI: 10.1007/978-3-319-32552-1_46 (page 105).

[262] Aaron Steinfeld, Terrence Fong, David B. Kaber, Michael Lewis, Jean Scholtz, Alan C. Schultz, and Michael A. Goodrich. "Common metrics for human-robot interaction". In: *ACM/IEEE Int. Conf. Hum.-Robot Interact.* Ed. by Michael A. Goodrich, Alan C. Schultz, and David J. Bruemmer. 2006, pp. 33–40. DOI: 10.1145/1121241.1121249 (page 125).

[263] Lukas von Stumberg, Patrick Wenzel, Qadeer Khan, and Daniel Cremers. "GN-Net: The Gauss-Newton Loss for Multi-Weather Relocalization". In: *IEEE Robot. Autom. Lett. (RA-L)* 5.2 (2020), pp. 890–897. DOI: 10.1109/LRA.2020.2965031 (page 36).

[264] Li Sun, Marwan Taher, Christopher Wild, Cheng Zhao, Filip Majer, Yan Zhi, Krajnk Tomas, Tony Prescott, and Tom Duckett. "Robust and long-term monocular teach-and-repeat navigation using a single-experience map". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2021, pp. 2635–2642. DOI: 10.1109/IROS51168.2021.9635886 (page 39).

[265] Niko Sünderhauf, Oliver Brock, Walter J. Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, and Peter Corke. "The limits and potentials of deep learning for robotics". In: *Intl. J. of Robot. Res.* 37.4-5 (2018), pp. 405–420. DOI: 10.1177/0278364918770733 (page 101).

[266] Richard Szeliski. *Computer Vision - Algorithms and Applications, Second Edition*. Texts in Computer Science. Springer International Publishing, 2022. ISBN: 978-3-030-34371-2. DOI: 10.1007/978-3-030-34372-9 (pages 17, 18).

[267] Jian Tang, Yuwei Chen, Antero Kukko, Harri Kaartinen, Anttoni Jaakkola, Ehsan Khoramshahi, Teemu Hakala, Juha Hyyppä, Markus Holopainen, and Hannu Hyyppä. "SLAM-Aided Stem Mapping for Forest Inventory with Small-Footprint Mobile LiDAR". en. In: *Forests* 6.12 (Dec. 2015), pp. 4588–4606. ISSN: 1999-4907. DOI: 10.3390/f6124390 (page 105).

[268] Sebastian Thrun, Jens-Steffen Gutmann, Dieter Fox, Wolfram Burgard, and Benjamin Kuipers. "Integrating Topological and Metric Maps for Mobile Robot Navigation: A Statistical Approach". In: *AAAI Conf. on Artificial Intelligence*. AAAI Press / The MIT Press, 1998, pp. 989–995 (page 38).

[269] D. J. Todd. "Applications of walking machines". In: *Walking Machines: An Introduction to Legged Robots*. Boston, MA: Springer US, 1985, pp. 169–177. ISBN: 978-1-4684-6858-8. DOI: 10.1007/978-1-4684-6858-8_8 (page 106).

[270] Emanuel Todorov. "General duality between optimal control and estimation". In: *IEEE Conf. on Decision and Control (CDC)*. 2008, pp. 4286–4292 (page 142).

[271] Ayumu Tominaga, Hayashi Eiji, and Abbe Mowshowitz. "Development of Navigation System in Field Robot for Forest Management". en. In: *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*. Toyama, Japan, Dec. 2018, pp. 1142–1147. ISBN: 978-1-5386-2633-7. DOI: 10.1109/SCIS-ISIS.2018.00180 (page 106).

[272] Marc Toussaint. "A Tutorial on Newton Methods for Constrained Trajectory Optimization and Relations to SLAM, Gaussian Process Smoothing, Optimal Control, and Probabilistic Inference". In: *Geometric and Numerical Foundations of Movements*. Ed. by Jean-Paul Laumond, Nicolas Mansard, and Jean-Bernard Lasserre. Springer Tracts in Advanced Robotics. Springer International Publishing, 2017, pp. 361–392. DOI: 10.1007/978-3-319-51547-2_15 (pages 12, 13, 142).

[273] Marco Tranzatto, Takahiro Miki, Mihir Dharmadhikari, Lukas Bernreiter, Mihir Kulkarni, Frank Mascarich, Olov Andersson, Shehryar Khattak, Marco Hutter, Roland Siegwart, and Kostas Alexis. "CERBERUS in the DARPA Subterranean Challenge". In: *Sci. Robot.* 7.66 (2022). DOI: 10.1126/scirobotics.abp9742 (page 133).

[274] Marco Tranzatto et al. "Team CERBERUS Wins the DARPA Subterranean Challenge: Technical Overview and Lessons Learned". In: *Field Robotics* abs/2201.07067 (2023) (pages 5, 50, 106, 141).

[275] Jean-François Tremblay, Martin Béland, Richard Gagnon, François Pomerleau, and Philippe Giguère. "Automatic three-dimensional mapping for tree diameter measurements in inventory operations". In: *J. Field Robot.* 37.8 (2020), pp. 1328–1346. DOI: https://doi.org/10.1002/rob.21980 (pages 105, 115).

[276] Michał Tyszkiewicz, Pascal Fua, and Eduard Trulls. "DISK: Learning local features with policy gradient". In: *Intl. Conf. on Neural Information Processing Systems (NeurIPS)* (2020) (page 35).

[277] Shimon Ullman. "The interpretation of structure from motion". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 203.1153 (1979), pp. 405–426. DOI: 10.1098/rspb.1979.0006 (page 34).

[278] I. Ulrich and I. Nourbakhsh. "Appearance-based place recognition for topological localization". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. Vol. 2. 2000, 1023–1029 vol.2. DOI: 10.1109/ROBOT.2000.844734 (page 36).

[279] Alberto Valero-Gomez, Javier V. Gómez, Santiago Garrido, and Luis Moreno. "The Path to Efficiency: Fast Marching Method for Safer, More Efficient Mobile Robot Trajectories". In: *IEEE Robotics & Automation Magazine* 20.4 (2013), pp. 111–120. DOI: 10.1109/MRA.2013.2248309 (page 77).

[280] Vasileios Vasilopoulos, Georgios Pavlakos, Sean L. Bowman, J. Diego Caporale, Kostas Daniilidis, George J. Pappas, and Daniel E. Koditschek. "Reactive Semantic Planning in Unexplored Semantic Environments Using Deep Perceptual Feedback". In: *IEEE Robot. Autom. Lett. (RA-L)* 5.3 (2020), pp. 4455–4462. DOI: 10.1109/LRA.2020.3001496 (page 45).

[281] Olga Vysotska and Cyrill Stachniss. "Effective visual place recognition using multi-sequence maps". In: *IEEE Robot. Autom. Lett. (RA-L)* 4.2 (2019), pp. 1730–1736. DOI: 10.1109/LRA.2019.2897160 (page 37).

[282] Yiduo Wang, Milad Ramezani, Matias Mattamala, Sundara Tejaswi Digumarti, and Maurice Fallon. "Strategies for large scale elastic and semantic LiDAR reconstruction". In: *Robot. Auton. Syst.* 155 (2022), p. 104185. ISSN: 0921-8890. DOI: doi.org/10.1016/j.robot.2022.104185 (page 5).

[283] Yiduo Wang, Milad Ramezani, Matias Mattamala, and Maurice Fallon. "Scalable and Elastic LiDAR Reconstruction in Complex Environments Through Spatial Analysis". In: *European Conference on Mobile Robotics (ECMR)*. 2021, pp. 1–8. DOI: 10.1109/ECMR50962.2021.9568844 (page 5).

[284] Lorenz Wellhausen. "Towards Autonomous Legged Robot Navigation in Natural, Unstructured Environments". en. Doctoral Thesis. Zurich: ETH Zurich, 2022. DOI: 10.3929/ethz-b-000584934 (page 2).

[285] 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 Robot. Autom. Lett. (RA-L)* 4.2 (2019), pp. 1509–1516. DOI: 10.1109/LRA.2019.2895390 (pages 46, 139).

[286] Lorenz Wellhausen and Marco Hutter. "ArtPlanner: Robust Legged Robot Navigation in the Field". In: *Field Robotics* (2023) (page 41).

[287] Lorenz Wellhausen and Marco Hutter. "Rough Terrain Navigation for Legged Robots using Reachability Planning and Template Learning". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2021. DOI: 10.1109/IROS51168.2021.9636358 (pages 41, 44).

[288] Lorenz Wellhausen, René Ranftl, and Marco Hutter. "Safe Robot Navigation Via Multi-Modal Anomaly Detection". In: *IEEE Robot. Autom. Lett. (RA-L)* 5.2 (2020), pp. 1326–1333. DOI: 10.1109/LRA.2020.2967706 (page 47).

[289] Martin Wermelinger, Péter Fankhauser, Remo Diethelm, Philipp Andreas Krüsi, Roland Siegwart, and Marco Hutter. "Navigation Planning for Legged Robots in Challenging Terrain". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. 2016, pp. 1184–1189. DOI: 10.1109/IROS.2016.7759199 (pages 40, 44).

[290] Thomas Whelan, Stefan Leutenegger, Renato F. Salas-Moreno, Ben Glocker, and Andrew J. Davison. "ElasticFusion: Dense SLAM Without A Pose Graph". In: *Robotics: Science and Systems (RSS)*. 2015. DOI: 10.15607/RSS.2015.XI.001 (page 34).

[291] Pierre-Brice Wieber, Russ Tedrake, and Scott Kuindersma. "Modeling and Control of Legged Robots". In: *Springer Handbook of Robotics*. Ed. by Bruno Siciliano and Oussama Khatib. Springer Handbooks. Springer, 2016, pp. 1203–1234. DOI: 10.1007/978-3-319-32552-1_48 (page 31).

[292] Erik Wijmans, Manolis Savva, Irfan Essa, Stefan Lee, Ari S. Morcos, and Dhruv Batra. "Emergence of Maps in the Memories of Blind Navigation Agents". In: *Intl. Conf. on Learning Representations (ICLR)*. 2023. DOI: 10.48550/arXiv.2301.13261 (page 143).

[293] Grady Williams, Paul Drews, Brian Goldfain, James M. Rehg, and Evangelos A. Theodorou. "Aggressive driving with model predictive path integral control". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2016, pp. 1433–1440. DOI: 10.1109/ICRA.2016.7487277 (page 30).

[294] Cristina G. Wilson, Feifei Qian, Douglas J. Jerolmack, Sonia Roberts, Jonathan Ham, Daniel Koditschek, and Thomas F. Shipley. "Spatially and temporally distributed data foraging decisions in disciplinary field science". In: *Cognitive Research: Principles and Implications* 6.1 (2021), p. 29. DOI: 10.1186/s41235-021-00296-z (pages 51, 106).

[295] David Wisth, Marco Camurri, and Maurice Fallon. "VILENS: Visual, Inertial, Lidar, and Leg Odometry for All-Terrain Legged Robots". In: *IEEE Trans. Robot.* 39.1 (2023), pp. 309–326. DOI: 10.1109/TRO.2022.3193788 (pages 26, 109).

[296] David Wooden, Matthew Malchano, Kevin Blankespoor, Andrew Howard, Alfred A. Rizzi, and Marc H. Raibert. "Autonomous navigation for BigDog". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2010, pp. 4736–4741. DOI: 10.1109/ROBOT.2010.5509226 (pages 39, 40, 44, 50, 52, 106).

[297] Oliver J Woodman. *An introduction to inertial navigation*. Tech. rep. Computer Laboratory, University of Cambridge, 2007 (page 23).

[298] Markus Wulfmeier, Dushyant Rao, Dominic Zeng Wang, Peter Ondruska, and Ingmar Posner. "Large-scale cost function learning for path planning using deep inverse reinforcement learning". In: *Intl. J. of Robot. Res.* 36.10 (2017), pp. 1073–1087. DOI: 10.1177/0278364917722396 (page 48).

[299] Karl Van Wyk, Mandy Xie, Anqi Li, Muhammad Asif Rana, Buck Babich, Bryan Peele, Qian Wan, Iretiayo Akinola, Balakumar Sundaralingam, Dieter Fox, Byron Boots, and Nathan D. Ratliff. "Geometric Fabrics: Generalizing Classical Mechanics to Capture the Physics of Behavior". In: *IEEE Robot. Autom. Lett. (RA-L)* 7.2 (2022), pp. 3202–3209. DOI: 10.1109/LRA.2022.3143311 (page 77).

[300] Yalin Xiong and Kenneth Turkowski. "Creating image-based VR using a self-calibrating fisheye lens". In: *IEEE Int. Conf. Computer Vision and Pattern Recognition*. 1997, pp. 237–243. DOI: 10.1109/CVPR.1997.609326 (page 21).

[301] Bowen Yang, Lorenz Wellhausen, Takahiro Miki, Ming Liu, and Marco Hutter. "Real-time Optimal Navigation Planning Using Learned Motion Costs". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2021, pp. 9283 –9289. DOI: 10.1109/ICRA48506.2021.9561861 (pages 41, 44, 45).

[302] Fan Yang, Chen Wang, César Cadena, and Marco Hutter. "iPlanner: Imperative Path Planning". In: *Robotics: Science and Systems (RSS)*. 2023 (page 42).

[303] Yuxiang Yang, Xiangyun Meng, Wenhao Yu, Tingnan Zhang, Jie Tan, and Byron Boots. "Learning Semantics-Aware Locomotion Skills from Human Demonstration". In: *Conf. on Robot Learning (CoRL)*. Ed. by Karen Liu, Dana Kulic, and Jeffrey Ichnowski. Vol. 205. Proceedings of Machine Learning Research. PMLR, 2022, pp. 2205–2214 (page 46).

[304] David J. Yoon, Haowei Zhang, Mona Gridseth, Hugues Thomas, and Timothy D. Barfoot. "Unsupervised Learning of Lidar Features for Use ina Probabilistic Trajectory Estimator". In: *IEEE Robot. Autom. Lett. (RA-L)* 6.2 (2021), pp. 2130–2138. DOI: 10.1109/LRA.2021.3060407 (page 142).

[305] Ji Zhang, Chen Hu, Rushat Gupta Chadha, and Sanjiv Singh. "Falco: Fast likelihood-based collision avoidance with extension to human-guided navigation". In: *J. Field Robot.* 37.8 (2020), pp. 1300–1313. DOI: doi.org/10.1002/rob.21952 (page 30).

[306] Ji Zhang, Michael Kaess, and Sanjiv Singh. "On degeneracy of optimization-based state estimation problems". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. Ed. by Danica Kragic, Antonio Bicchi, and Alessandro De Luca. 2016, pp. 809–816. DOI: 10.1109/ICRA.2016.7487211 (page 133).

[307] Ji Zhang and Sanjiv Singh. "LOAM: Lidar Odometry and Mapping in Real-time". In: *Robotics: Science and Systems (RSS)*. Ed. by Dieter Fox, Lydia E. Kavraki, and Hanna Kurniawati. 2014. DOI: 10.15607/RSS.2014.X.007 (page 118).

[308] Wuming Zhang, Jianbo Qi, Peng Wan, Hongtao Wang, Donghui Xie, Xiaoyan Wang, and Guangjian Yan. "An Easy-to-Use Airborne LiDAR Data Filtering Method Based on Cloth Simulation". In: *Remote. Sens.* 8.6 (2016), p. 501. DOI: 10.3390/rs8060501 (page 113).

[309] Yanfu Zhang, Wenshan Wang, Rogerio Bonatti, Daniel Maturana, and Sebastian A. Scherer. "Integrating kinematics and environment context into deep inverse reinforcement learning for predicting off-road vehicle trajectories". In: *Conf. on Robot Learning (CoRL)*. Vol. 87. Proceedings of Machine Learning Research. PMLR, 2018, pp. 894–905 (page 48).

[310] Zichao Zhang and Davide Scaramuzza. "Beyond Point Clouds: Fisher Information Field for Active Visual Localization". In: *IEEE Int. Conf. Robot. Autom. (ICRA)*. 2019, pp. 5986–5992. DOI: 10.1109/ICRA.2019.8793680 (page 63).

[311] Xin Zhou, Xiangyong Wen, Zhepei Wang, Yuman Gao, Haojia Li, Qianhao Wang, Tiankai Yang, Haojian Lu, Yanjun Cao, Chao Xu, and Fei Gao. "Swarm of micro flying robots in the wild". In: *Sci. Robot.* 7.66 (2022), eabm5954. DOI: 10.1126/scirobotics.abm5954 (page 106).

[312] Ciyou Zhu, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal. "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization". en. In: *ACM Transactions on Mathematical Software* 23.4 (Dec. 1997), pp. 550–560. ISSN: 0098-3500, 1557-7295. DOI: 10.1145/279232.279236 (page 116).

[313] Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. "Maximum Entropy Inverse Reinforcement Learning". In: *AAAI Conf. on Artificial Intelligence*. 2008, pp. 1433–1438 (page 48).

[314] Matthew Zucker, Nathan D. Ratliff, Anca D. Dragan, Mihail Pivtoraiko, Matthew Klingensmith, Christopher M. Dellin, J. Andrew Bagnell, and Siddhartha S. Srinivasa. "CHOMP: Covariant Hamiltonian optimization for motion planning". In: *Intl. J. of Robot. Res.* 32.9-10 (2013), pp. 1164–1193. DOI: 10.1177/0278364913488805 (page 30).

[315] Jannik Zürn, Wolfram Burgard, and Abhinav Valada. "Self-Supervised Visual Terrain Classification From Unsupervised Acoustic Feature Learning". In: *IEEE Trans. Robot.* 37.2 (2021), pp. 466–481. DOI: 10.1109/TRO.2020.3031214 (page 47).