

Imperial College London
Department of Computing

Calibration and Estimation for Aerial Robots with Application to Additive Manufacturing

Christopher L. Choi

21st December 2024

Supervised by Dr Stefan Leutenegger
Second Supervisor: Professor Andrew J. Davison

Submitted in part fulfilment of the requirements for the degree of PhD in Computing and the Diploma of Imperial College London. This thesis is entirely my own work, and, except where otherwise indicated, describes my own research.

Copyright Declaration

The copyright of this thesis rests with the author. Unless otherwise indicated, its contents are licensed under a Creative Commons Attribution-Non Commercial 4.0 International Licence (CC BY-NC).

Under this licence, you may copy and redistribute the material in any medium or format. You may also create and distribute modified versions of the work. This is on the condition that: you credit the author and do not use it, or any derivative works, for a commercial purpose.

When reusing or sharing this work, ensure you make the licence terms clear to others by naming the licence and linking to the licence text. Where a work has been adapted, you should indicate that the work has been changed and describe those changes.

Please seek permission from the copyright holder for uses of this work that are not included in this licence or permitted under UK Copyright Law.

Abstract

Additive Manufacturing (AM) has recently entered the construction industry, enabling structures to be built layer by layer. However, existing frameworks often rely on bulky static gantry systems that hinder worker safety and operational flexibility. Ground mobile robots provide an alternative but are constrained by height and maneuverability. This thesis introduces an untethered Aerial Additive Manufacturing (Aerial-AM) framework using autonomous aerial robots to overcome these challenges. A proof-of-concept cylinder structure was successfully printed, demonstrating the feasibility of large-scale construction with multiple autonomous aerial robots.

Autonomous aerial navigation depends on accurate state estimation and Simultaneous Localization and Mapping (SLAM), with Visual-Inertial (VI) sensors being a preferred choice for their affordability and compactness. However, accurate VI sensor calibration is crucial and often challenging. Traditional calibration tools focus on algorithmic refinement without addressing the need for specific sensor measurements. This thesis proposes an active sensing approach using Next-Best-View (NBV) for camera calibration and Next-Best-Trajectory (NBT) for camera-IMU calibration. Experiments show this method provides faster, more accurate, and consistent calibration, enhancing Visual-Inertial Odometry (VIO) and SLAM performance.

Additionally, active sensing addresses limitations in passive perception for state estimation and SLAM. Rigidly mounted cameras on aerial robots often experience motion blur during sudden maneuvers or fail in featureless environments, causing tracking errors. To mitigate these issues, a dual VI-sensor rig was developed, with one sensor rigidly mounted and the other stabilized by a 3-axis gimbal. The stabilized sensor actively reduces motion blur, providing smoother and more reliable measurements. Indoor experiments confirmed that gimbal-stabilized VIO achieves lower trajectory errors compared to body-fixed setups. These advancements in calibration and sensing techniques highlight the potential of active sensing in aerial robotics and additive manufacturing.

Acknowledgements

I would like to express my sincere gratitude to those who have supported me through the course of my PhD.

I would like to first begin by thanking Dr Stefan Leutenegger for giving me the opportunity to pursue a PhD at Imperial College London. His guidance throughout my PhD has been instrumental in shaping my academic path. I deeply appreciate the dedication and support he has shown in fostering a cohesive and supportive atmosphere within the Smart Robotics Lab, from our annual lab trips to Innsbruck and Garmisch-Partenkirchen, to our morning coffee calls during the challenging times of COVID lockdowns. This sense of community has been invaluable to me throughout my doctoral studies, making the journey more enriching and manageable.

As a member of the Smart Robotics Lab it was an honour to work with all the past and present members of the group. Thank you Sotiris for always being there and lending an ear, be it my C++ woes to letting me make fun of your eccentric tastes in computing. Dimos for your inspirational drone control work, and our field trial moments in Snowdonia. Binbin for your support and help in publishing my calibration paper and our discussions on SLAM. Furthermore, I would like to thank the entire Smart Robotics Lab. Thank you Nils, Masha, Felix, Anna for all the great moments we spent together in the lab.

Finally, I owe my deepest gratitude to my mother, who has made many sacrifices for me to pursue my dreams.

Christopher L. Choi

Contents

1	Introduction	9
1.1	Motivation	9
1.2	Thesis Structure and Contributions	12
1.3	Publications	13
2	Preliminaries	15
2.1	Notation	17
2.2	Camera and Distortion Models	20
2.3	Probabilistic Modelling in Robotics	25
2.4	Information Theory	34
2.5	Brief Literature Review of Camera and Camera-IMU Calibration	40
3	Aerial Additive Manufacturing	47
3.1	Introduction	49
3.2	Related Work	49
3.3	Contributions	52
3.4	The Aerial Additive Manufacturing Framework	53
3.5	Experiment	65
3.6	Conclusion	69
3.7	Future Work	70

4	Accurate And Interactive Visual-Inertial Calibration with Next-Best-View and Next-Best-Trajectory	77
4.1	Introduction	79
4.2	Related Work	81
4.3	Background	82
4.4	System Overview	84
4.5	Camera Intrinsics and Extrinsics Calibration	85
4.6	Camera-IMU Extrinsics Calibration	88
4.7	Experiments	95
4.8	Conclusions	101
5	Towards More Robust Visual-Inertial Odometry with a Gimbal	103
5.1	Introduction	105
5.2	Related Work	108
5.3	System Overview	110
5.4	Gimbal Calibration	114
5.5	Experiments	117
5.6	Conclusions	124
6	Conclusions	125
6.1	Summary of Results	125
6.2	Limitations and Future Work	126
	Bibliography	129

List of Figures

2.1	Types of radial distortions.	22
2.2	Types of tangential distortions.	23
2.3	Pinhole radial-tangential camera model.	24
2.4	Relationship between joint, marginal and conditional entropies with mutual information	37
2.5	Tsai's camera calibration object	41
2.6	Zhang's camera calibration method using planar targets	42
2.7	Pendulum used to calibrate the IMU sensor	44
2.8	Camera-IMU placed on a turntable in different poses for translation calibration	44
2.9	Continuous-time camera-IMU calibration	45
3.1	Comparison of different additive manufacturing robot platforms	50
3.2	Drones used in Aerial Additive Manufacturing	53
3.3	Aerial Additive Manufacturing framework system overview	54
3.4	Aerial Additive Manufacturing system diagram	55
3.5	Height profile analysis	59
3.6	MOCAP marker to camera frame diagram	60
3.7	Mesh reconstructions from the ScanDrone	66
3.8	Printed cylinder structure alongside the BuilDrone and ScanDrone.	67
3.9	Printed cylinder structure.	68

3.10	Print Layer Height Compensation	71
3.11	Print Layer X-Y Plane Compensation	71
3.12	Improving the ScanDrone trajectory to capture more details on the side of the printed structure.	72
3.13	RTK GPS Outdoor Experiment	73
3.14	Static Test - GPS Mode	74
3.15	Static Test - RTK-GPS Mode	74
4.1	Interactive Next-Best-View and Next-Best-Trajectory for camera-IMU calibration	80
4.2	Overview of the VI calibration pipeline	84
4.3	Camera calibration pipeline	87
4.4	NBV candidate poses	87
4.5	Camera-IMU calibration pipeline	88
4.6	Next-Best-Trajectory (NBT) candidates in-front of a calibration target	94
4.7	Comparison of ORBSLAM3 using calibrations from Kalibr and ours in Stereo- VO mode on EuRoC Dataset	97
4.8	Comparison of ORBSLAM3 using calibrations from Kalibr and ours in Stereo- VIO mode on EuRoC Dataset	97
4.9	Comparing calibrations obtained by graduate students across 10 different evaluation VICON room sequences by running ORBSLAM3 in odometry mode	99
4.10	Median total calibration time per person in seconds	99
4.11	Shannon Entropy of camera and camera-IMU calibration across all calibrations	99
5.1	Sensor rig system overview	110
5.2	Sensor rig	111
5.3	Sensor synchronization	112
5.4	Sensor rig frame diagram	112
5.5	Gimbal kinematic chain from gimbal motor 1 $\vec{F}_{M_1}$ to camera 3 $\vec{F}_{C_3}$	113
5.6	Example gimbal kinematics calibration image	119
5.7	Set: 1, Sequence: Circle 0	120

5.8	Set: 1, Sequence: Circle 1	121
5.9	Set: 1, Sequence: Figure8 - 0	121
5.10	Set: 1, Sequence: Figure8 - 1	121
5.11	Set: 1, Sequence: Random - 0	122
5.12	Set: 1, Sequence: Random - 1	122
5.13	Set: 2, Sequence: Circle - 0	122
5.14	Set: 2, Sequence: Circle - 1	123
5.15	Set: 2, Sequence: Figure 8 - 0	123
5.16	Set: 2, Sequence: Figure 8 - 1	123
5.17	Set: 2, Sequence: Random - 0	124
5.18	Set: 2, Sequence: Random - 1	124

List of Tables

4.1	Comparing Camera Parameters Between Kalibr and Our Method	100
4.2	Comparing Extrinsic Parameters Between Kalibr and Our Method	101
5.1	Root-Mean-Squared Absolute Trajectory Error between body-fixed and gimbal-stabilised VI-sensor	120

Introduction

Contents of Chapter

1.1	Motivation	9
1.2	Thesis Structure and Contributions	12
1.3	Publications	13

1.1 Motivation

In recent years, additive manufacturing has found its way into construction, where structures are built layer by layer, offering unprecedented design freedom and customization. There is a growing interest in automating additive manufacturing using robots to improve the productivity and safety in construction activities [Delgado et al., 2019, Tay et al., 2017], such as assembly of building elements [Lindsey et al., 2011, Reinhardt et al., 2016, Stuart-Smith, 2016, Furrer et al., 2017, Kayser et al., 2018] and free-form continuous additive manufacturing [Lim et al., 2012, Gosselin et al., 2016, Keating et al., 2017, Buswell et al., 2018]. The advantage free-form continuous additive includes added flexibility in building geometrically variable designs, and the opportunity to reduce material waste and cost. Current state of the art additive manufacturing techniques involve using static [Lim et al., 2012, Gosselin et al., 2016, Keating et al., 2017, Buswell et al., 2018] and mobile robots for

on-site construction [Sustarevas et al., 2018, Zhang et al., 2018]. The main limitations of static systems is that the robot hardware often needs to scale up to a larger dimension than the desired manufacturing volume, making it difficult and dangerous for the system to run in parallel with workers inside the building site, in turn making inspection maintenance and repair difficult during operation. Mobile robots on-site construction systems, on the other hand, is predominantly focused on utilising mobile ground robots, where there is an inherent operational height limitation. Additionally, the building geometry needs to be feasible for the ground robot to achieve. We argue that a collective construction that leverages a network of untethered aerial mobile robots addresses the limitations posed by large static systems and mobile ground robots.

A vital component for mobile construction robots to navigate and interact with the world is robot state-estimation. Research in state-estimation such as Simultaneous Localization and Mapping (SLAM) and Visual-Inertial-Odometry (VIO) techniques is well established, thanks to the popularity of Visual-Inertial (VI) sensors due to the relative low cost, small size, and weight. However, in order to use VI-sensors for state-estimation tasks the calibration parameters must first be accurately obtained.

While state of the art offline VI calibration tools such as Kalibr [Furgale et al., 2013] can estimate the calibration parameters accurately, a common challenge in robotics is the capture of informative visual-inertial sensor measurements required to yield the accurate calibration parameters is not trivial for a non-expert. State-estimation algorithms such as OKVIS [Leutenegger et al., 2014], VINS-MONO [Qin et al., 2018], and OpenVINS [Geneva et al., 2020] can in theory optimize the calibration parameters online, however, this assumes the calibration parameters are sufficiently accurate to begin with. Further, online calibration assumes subsequent visual-inertial measurements contain sufficient visual features and motion excitation to yield the calibration parameters observable.

To tackle this problem, we present an interactive VI-sensor calibration pipeline that incorporates active sensing to guide non-experts in collecting *informative* calibration data. By providing Next-Best-View (NBV) and Next-Best-Trajectory (NBT) suggestions, our

approach ensures the efficient acquisition of high-quality calibration data. Active sensing techniques help identify the most informative views and trajectories, significantly improving the accuracy of the calibration process. Through this method, users can achieve sound calibrations more efficiently, enhancing the performance of state-estimation algorithms in mobile construction robots.

Aerial mobile robots, such as quadcopters, often have rigidly mounted cameras for state estimation. The challenge with rigidly mounted cameras on quadcopters, is that they frequently observe motion blur caused by sudden attitude changes due to wind or aggressive controls. This results in sub-optimal visual-inertial measurements for state estimation algorithms, often causing the algorithms to lose tracking.

Recently, advances in deep learning has introduced various methods to make state estimation more robust. These methods, however, introduce several disadvantages that make deployment on a compute resource constrained mobile robots difficult. Namely, deep learning models often require significant resources during inference in terms of processing power and memory. In addition, training these models require large corpus of high quality data that maybe specific to the sensor configuration. With these limitations in mind it is often not practical for in-field robot deployment. An argument can be made that degraded sensor measurements represent a fundamental loss of information, a gap that no amount of algorithmic manipulation can recover from.

To mitigate these issues, we advocate for a hardware approach where we stabilize a VI-sensor using a 3-axis gimbal for VIO. Additionally, we incorporate active sensing to enhance the performance of the stabilized sensor. By actively adjusting the sensor's orientation through the gimbal, we can mitigate motion blur and ensure that the sensor captures the ideal visual-inertial measurements. This proactive approach not only stabilizes the sensor but also improves data quality for state estimation algorithms. Through this method, we demonstrate that active sensing, combined with hardware stabilization, significantly improves the robustness and accuracy of visual-inertial odometry in aerial mobile robots.

In summary, this thesis introduces a novel aerial additive manufacturing framework with multiple untethered aerial drones to improve the productivity and safety of construction activities. It promotes an active sensing approach to tackle the challenges associated with calibrating VI-sensors, utilising Next-Best-View and Next-Best-Trajectory strategies. Additionally, it emphasizes the importance of achieving more reliable visual-inertial sensing for state estimation, facilitated by the integration of a 3-axis gimbal.

1.2 Thesis Structure and Contributions

The remainder of the thesis is structured as follows:

Chapter 2 introduces basic notations used throughout this thesis, common camera models used in calibration and state estimation, how probability theory is used in robotics to estimate the robot's state, to using information theory to quantify the amount of information a random variable has, and finally a brief literature review on sensor calibration, specifically camera and camera-IMU calibration methods.

Chapter 3 presents an Aerial Additive Manufacturing (Aerial-AM) framework that couples the merits of natural precedences with engineering principals, and demonstrate an aerial, untethered, unbounded three-dimensional printing system. We will mainly focus on the contributions towards the trajectory planning for the construction of a cylindrical geometry. The scanning pipeline used to perform map reconstruction of the printed structure for inspection and validation. A motion-capture (MOCAP) marker to camera calibration that resolves the relative transformation between MOCAP marker and camera sensor used on the ScanDrone. Estimating the map accuracy using first order error propagation. To evaluate the Aerial-AM, a foam cylinder structure at 2.05m tall with a 0.3m diameter was constructed.

Chapter 4 describes a novel accurate and interactive visual-inertial calibration framework. We present an information-theoretic procedure to identify the most informative Next-Best-View and Next-Best-Trajectory among a pre-defined set of view points and

trajectory primitives, along with an interactive graphical user interface for guiding the user through the calibration data collection process. Through experiments we show that our proposed method is faster, more accurate and more reliable compared to state of the art traditional non-guided calibration methods, such as Kalibr [Furgale et al., 2013], even when used by *novices* in our user study ($n = 16$).

Chapter 5 presents a unique dual VI-sensor rig, where the first VI-sensor is rigidly mounted onto the body of the sensor rig, and the second is stabilised with a 3-axis gimbal. The aim of this chapter is to demonstrate the benefits of a VI-sensor for Visual-Inertial Odometry (VIO) or state-estimation tasks, when compared to the convention of fixing a VI-sensor to the body of a sensor rig. The hardware approach of providing better visual inertial sensor measurements for a VIO system is advocated, as supposed to using deep learning approaches to perform state estimation using sub-optimal measurement data.

Chapter 6 concludes this thesis with a summary of the results presented and suggestions for future work.

1.3 Publications

The work described in this thesis resulted in the following publications:

- Ketao Zhang, Pisak Chermprayong, Feng Xiao, Dimos Tzoumanikas, Barrie Dams, Sebastian Kay, Basaran Bahadir Kocer, Alec Burns, Lachlan Orr, Talib Alhinai, Christopher Choi, Durgesh Dattatray Darekar, Wenbin Li, Steven Hirschmann, Valentina Soana, Shamsiah Awang Ngah, Clément Grillo, Sina Sareh, Ashutosh Choubey, Laura Margheri, Vijay M. Pawar, Richard J. Ball, Chris Williams, Paul Shepherd, Stefan Leutenegger, Robert Stuart-Smith and Mirko Kovac (2022). **Aerial additive manufacturing with multiple autonomous robots**. *Nature*. [Zhang et al., 2022].

- Christopher Choi, Binbin Xu, Stefan Leutenegger (2023). **Accurate and Interactive Visual-Inertial Sensor Calibration with Next-Best-View and Next-Best-Trajectory Suggestion**. *In the Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. [[Choi et al., 2023](#)].

The following publications, though not explicitly mentioned, were conducted in conjunction with this thesis:

- Yifei Ren, Binbin Xu, Christopher Choi, and Stefan Leutenegger (2022). **Visual-Inertial Multi-Instance Dynamic SLAM with Object-level Relocalisation**. *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. [[Ren et al., 2022](#)].
- Sotiris Papatheodorou, Nils Funk, Dimos Tzoumanikas, Christopher Choi, Binbin Xu, and Stefan Leutenegger (2023). **Finding Things in the Unknown: Semantic Object-Centric Exploration with an MAV**. *In the Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*. [[Papatheodorou et al., 2023](#)].
- Dorian F. Henning, Christopher Choi, Simon Schaefer, and Stefan Leutenegger (2023). **BodySLAM++ Fast and Tightly-Coupled Visual-Inertial Camera and Human Motion Tracking**. *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. [[Henning et al., 2023](#)]

My contributions to [[Ren et al., 2022](#)] were primarily editorial. I helped shape the narrative of the paper, refined the explanation of the method, and summarized the results to build a compelling case that the method is both sound and effective. For [[Papatheodorou et al., 2023](#)] and [[Henning et al., 2023](#)], both publications relied on the calibration software I developed to resolve the extrinsic parameters between the motion capture markers and the camera frame, which was crucial for their evaluations.

Preliminaries

In this chapter, we will present the fundamental concepts used in this thesis. The chapter will begin with the mathematical notation used throughout this thesis and continue with fundamental computer vision concepts covering the camera and distortion model, then continue with introducing probabilistic modeling in robotics, and information theory before concluding the chapter with a brief literature overview of sensor calibration.

Contents of Chapter

2.1	Notation	17
2.1.1	General notation	17
2.1.2	Spaces and manifolds	18
2.1.3	Frames and transformations	19
2.2	Camera and Distortion Models	20
2.2.1	Pinhole Camera Model	20
2.2.2	Radial Distortion Model	21
2.2.3	Tangential Distortion Model	22
2.2.4	Pinhole Radial-Tangential Model	23
2.3	Probabilistic Modelling in Robotics	25
2.3.1	Probability Theory	25
2.3.2	Bayes Rule	27
2.3.3	Maximum a Posteriori Inference	28
2.3.4	Nonlinear Least Squares	29
2.4	Information Theory	34
2.4.1	Information Content	35
2.4.2	Entropy	35
2.4.3	Joint Entropy	35
2.4.4	Conditional Entropy	36
2.4.5	Mutual Information	37
2.4.6	Continuous Variables	38
2.4.7	Mutual Information in Robotics	38
2.5	Brief Literature Review of Camera and Camera-IMU Calibration . .	40
2.5.1	Camera Calibration	40
2.5.2	Camera-IMU Calibration	43

2.1 Notation

This section introduces the notation used throughout this thesis.

2.1.1 General notation

a A lower-case symbol denotes a scalar.

$\mathbf{a}$ A bold lower-case symbol denotes an m -dimensional column vector. With a_i the i^{th} element as:

$$\mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{bmatrix}, \mathbf{a}^\top = [a_1, a_2, \dots, a_m]. \quad (2.1)$$

We use $\mathbf{a}_{i:j}$ to denote the vector consisting of the elements of $\mathbf{a}$ with indices in the $[i, j]$ range.

$\mathbf{a}$ A bold italic lower-case symbol represents an homogeneous vector defined as: $\mathbf{a} = [\mathbf{a}, 1]^\top$.

$\mathbf{A}$ A bold capital symbol denotes an $m \times n$ matrix. With $a_{i,j}$ the element at the i^{th} row and j^{th} column:

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \quad (2.2)$$

$\mathbf{I}$ The identity matrix, optionally with dimensions as subscript.

$\mathbf{0}$ The zero matrix, optionally with dimensions as subscript.

$[\cdot]_{\times}$ The cross-product matrix that produces a skew symmetric matrix from a 3D vector such that $\mathbf{a} \times \mathbf{b} = [\mathbf{a}]_{\times} \mathbf{b}$. Given the vector $\mathbf{a} = [a_x, a_y, a_z]^T$, $[\mathbf{a}]_{\times}$ can be obtained by:

$$[\mathbf{a}]_{\times} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}. \quad (2.3)$$

The properties of the cross product operation can be applied, such as the anti-commutative property: $[\mathbf{a}]_{\times} \mathbf{b} = -[\mathbf{b}]_{\times} \mathbf{a}$, as well as the skew-symmetric property: $([\mathbf{a}]_{\times})^T = -[\mathbf{a}]_{\times}$. The combined properties can lead some useful equation, such as $\mathbf{a}^T [\mathbf{b}]_{\times} = -([\mathbf{b}]_{\times} \mathbf{a})^T$.

$\mathcal{A}$ A calligraphic capital symbol denotes a set.

$\mathcal{N}(\mu, \Sigma)$ The Normal distribution with mean μ and covariance Σ .

2.1.2 Spaces and manifolds

$\mathbb{R}$ The set of real numbers.

$\mathbb{R}^m$ The vector space of real m -dimensional vectors.

$\mathbb{R}^{m \times n}$ The vector space of real $m \times n$ -dimensional matrices.

S^3 The 3-sphere group.

$SO(3)$ Special Orthogonal group: the group of 3D rotations.

$so(3)$ lie algebra associated with $SO(3)$.

$SE(3)$ Special Euclidean group: the group of 3D rigid transformations.

$\boxplus$ The “box-plus” operator that applies a small perturbation expressed in a tangent space to a manifold state, e.g. $SO(3) \times \mathbb{R}^3 \rightarrow SO(3)$.

$\boxminus$ The “box-minus” operator that expresses the difference of two manifold states in the tangent space.

2.1.3 Frames and transformations

$\underline{\mathcal{F}}_A$ A cartesian coordinate frame in $\mathbb{R}^3$.

${}_A\mathbf{p}$ A vector $\mathbf{p}$ expressed in the frame $\underline{\mathcal{F}}_A$, for example a 3D point in the frame $\underline{\mathcal{F}}_A$.

${}_A\mathbf{r}_{AB}$ The translation vector from $\underline{\mathcal{F}}_A$ to $\underline{\mathcal{F}}_B$ expressed in $\underline{\mathcal{F}}_A$.

${}_A\mathbf{v}_{AB}$ The linear velocity of $\underline{\mathcal{F}}_B$ with respect to $\underline{\mathcal{F}}_A$ expressed in $\underline{\mathcal{F}}_A$.

${}_A\mathbf{a}_{AB}$ The linear acceleration of $\underline{\mathcal{F}}_B$ with respect to $\underline{\mathcal{F}}_A$ expressed in $\underline{\mathcal{F}}_A$.

${}_A\boldsymbol{\omega}_{AB}$ The angular velocity of $\underline{\mathcal{F}}_B$ with respect to $\underline{\mathcal{F}}_A$ expressed in $\underline{\mathcal{F}}_A$.

ϕ The roll angle.

θ The pitch angle.

ψ The yaw angle.

$\mathbf{C}_{AB}$ The rotation matrix that transforms a vector $\mathbf{v}_B$ expressed in $\underline{\mathcal{F}}_B$ to one expressed in $\underline{\mathcal{F}}_A$ as: $\mathbf{v}_A = \mathbf{C}_{AB}\mathbf{v}_B$. The inverse rotation $\mathbf{C}_{BA}$ can be computed as: $\mathbf{C}_{BA} = \mathbf{C}_{AB}^{-1} = \mathbf{C}_{AB}^\top$.

$\mathbf{q}$ The quaternion following the Hamiltonian convention with vector part $\mathbf{q}_v$ and real part q_w give by: $\mathbf{q} = [\mathbf{q}_v^\top, q_w]^\top$. When used for representing orientation, the unit length $\mathbf{q}_{AB}$ represents the quaternion equivalent of the rotation matrix $\mathbf{C}_{AB}$.

$\mathbf{q}^*$ The conjugate of the quaternion $\mathbf{q}$ defined by: $\mathbf{q}^* = [-\mathbf{q}_v^\top, q_w]^\top$. For the rotation quaternion $\mathbf{q}_{AB}$, its conjugate represents the inverse rotation as: $\mathbf{q}_{AB}^* = \mathbf{q}_{AB}^{-1} = \mathbf{q}_{BA}$.

$\otimes$ The quaternion multiplication is defined such that $\mathbf{q}_{AC} = \mathbf{q}_{AB} \otimes \mathbf{q}_{BC}$.

$[\cdot]^+$ The left-hand quaternion matrix such that $\mathbf{q}_{AC} = [\mathbf{q}_{AB}]^+ \mathbf{q}_{BC}$. In matrix form $[\mathbf{q}]^+$ is given by:

$$[\mathbf{q}]^+ = \begin{bmatrix} q_w & -\mathbf{q}_v^\top \\ \mathbf{q}_v & q_w\mathbf{I} + [\mathbf{q}_v]_\times \end{bmatrix}. \quad (2.4)$$

$[\cdot]^\oplus$ The right-hand quaternion matrix such that $\mathbf{q}_{AC} = [\mathbf{q}_{BC}]^\oplus \mathbf{q}_{AB}$. In matrix form $[\mathbf{q}]^\oplus$ is given by:

$$[\mathbf{q}]^\oplus = \begin{bmatrix} q_w & -\mathbf{q}_v^\top \\ \mathbf{q}_v & q_w \mathbf{I} - [\mathbf{q}_v]_\times \end{bmatrix}. \quad (2.5)$$

$\mathbf{T}_{AB}$ The transformation matrix that transforms homogeneous vectors from $\underline{\mathcal{F}}_B$ to $\underline{\mathcal{F}}_A$ as ${}_A\mathbf{P}_P = \mathbf{T}_{AB} {}_B\mathbf{P}_P$.

$\pi(\cdot)$ Perspective projection function.

2.2 Camera and Distortion Models

Today, camera sensors are widely used in computer vision and robotics due to their size, cost and image quality. Cameras offer visual sensory input that enhances a robot's ability to perceive and navigate their surroundings. They play a key role in object recognition, tracking, and obstacle avoidance, enabling robots to interact effectively even in dynamic environments. To use camera sensors in algorithms, camera models are needed to interpret the visual data captured. In particular, we are interested in the relationship between 3D points in the environment and the corresponding 2D image points observed in the image plane of the camera sensor.

In the following we will briefly describe the pinhole camera model, radial and tangential distortion model (also known as Brown-Conrady distortion model [Brown, 1966]), fundamental concepts in robotics and computer vision. These concepts are also used in Chapter. 3, Chapter. 4 and Chapter. 5 to model a camera for calibration and state-estimation tasks. A complete survey of different camera and distortion models used in the computer vision and robotics is outside the scope of this thesis, but the reader is encouraged to review [Geyer and Daniilidis, 2000, Kannala and Brandt, 2006, Usenko et al., 2018].

2.2.1 Pinhole Camera Model

The pinhole camera model is a simple camera model that enables geometric and algebraic operations in computer vision and robotics, such as 3D reconstruction, object recognition

and state-estimation. The equations to project a 3D point ${}_C\mathbf{p} = [X \ Y \ Z]^\top$ in camera frame $\vec{F}_C$ onto the 2D image plane are described by:

$$\begin{aligned} u &= f_u \frac{X}{Z} + c_u, \\ v &= f_v \frac{Y}{Z} + c_v, \end{aligned} \tag{2.6}$$

where f_u and f_v are the focal lengths in the u and v axis of the image plane, c_u and c_v are the principal center of the camera, and $(u \ v)$ are the coordinates of the corresponding projected point in the image plane in pixels. The equations Eq. (2.6) can be re-written in matrix form as,

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} f_u & 0 \\ 0 & f_v \end{bmatrix} \begin{bmatrix} X/Z \\ Y/Z \end{bmatrix} + \begin{bmatrix} c_u \\ c_v \end{bmatrix} \tag{2.7}$$

In practice, however, factors such as curvature of a camera lens, camera lens mis-alignment and other factors introduce distortion effects that change way the camera senses the world. Specifically, distortion effects can affect the geometric relationship between 3D points in the environment and the corresponding measured 2D image point in the image plane. As such, the pinhole camera model is often used in conjunction with a distortion model with the aim of minimizing the projection errors from 3D to 2D.

The two most common distortion models are *radial distortion* and *tangential distortion* (also known as the Brown-Conrady distortion model [Brown, 1966]) which are distortions due to lens curvature, and misalignment between camera sensor and lens respectively.

2.2.2 Radial Distortion Model

Radial distortion occurs due to the curvature of the camera lens. Light passing through the center undergoes no refraction. Light passing through the edges of a curved camera lens, on the other hand, undergoes through severe bending. This makes straight lines in the real world appear curved on the image. The radial displacement can be positive or negative,

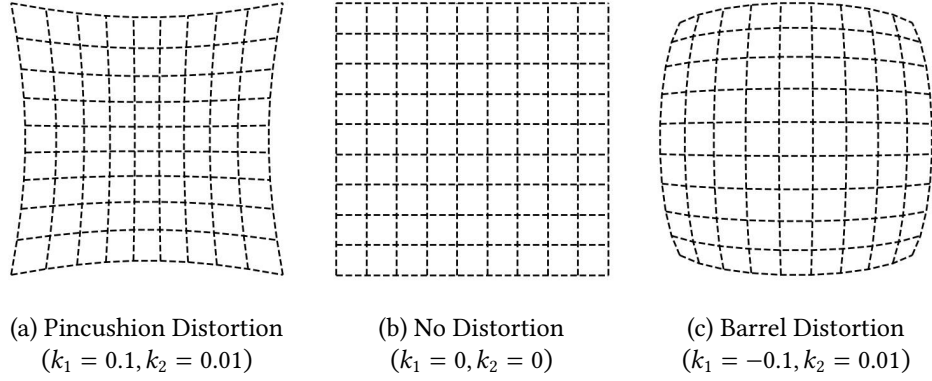


Figure 2.1: Types of radial distortions.

and [Brown, 1966] modeled the radial distortions using polynomial equations of the form:

$$\begin{aligned}
 x &= X/Z \\
 y &= Y/Z \\
 r^2 &= x^2 + y^2 \\
 d_r &= (1 + k_1 r^2 + k_2 r^4) \\
 x' &= x d_r, \\
 y' &= y d_r,
 \end{aligned} \tag{2.8}$$

where ${}_c\mathbf{p} = [X \ Y \ Z]^\top$ is a 3D point in camera frame, (x, y) and (x', y') are the undistorted and distorted point respectively, and k_1, k_2 are the radial distortion coefficients. Higher orders of radial distortion can be used, but second order radial distortion is generally sufficient for most applications.

2.2.3 Tangential Distortion Model

Tangential distortion is another distortion that is commonly used in conjunction with the radial distortion. The tangential distortion models distortion effects due to misalignment between camera sensor and camera lens, specifically the primary cause is due to the lens assembly not being centered and parallel over the camera sensor. The tangential distortion

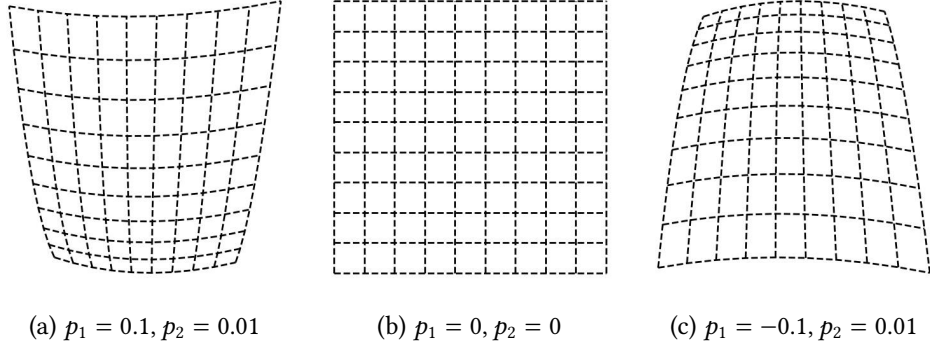


Figure 2.2: Types of tangential distortions.

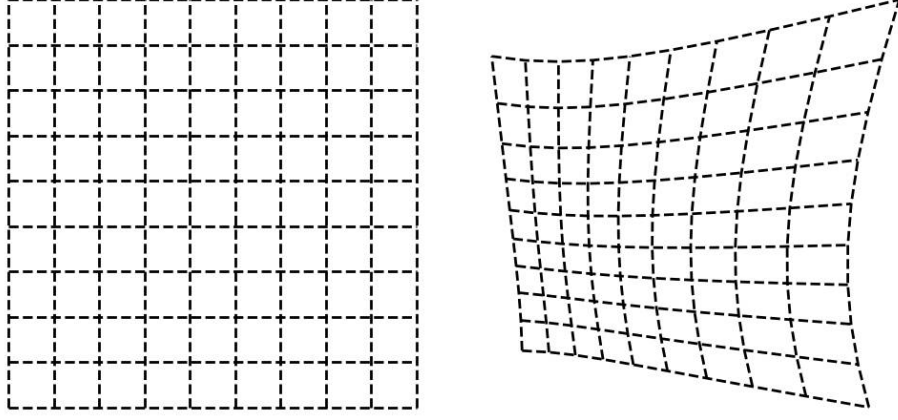
model [Brown, 1966] is defined as:

$$\begin{aligned}
 x &= X/Z \\
 y &= Y/Z \\
 r^2 &= x^2 + y^2 \\
 \mathbf{d}_t &= \begin{bmatrix} 2 p_1 x y + p_2 (r^2 + 2x^2) \\ 2 p_2 x y + p_1 (r^2 + 2y^2) \end{bmatrix} \\
 \begin{bmatrix} x' \\ y' \end{bmatrix} &= \begin{bmatrix} x \\ y \end{bmatrix} + \mathbf{d}_t
 \end{aligned} \tag{2.9}$$

where ${}_c\mathbf{p} = [X \ Y \ Z]^\top$ is a 3D point in camera frame, (x, y) and (x', y') are the undistorted and distorted point respectively, p_1, p_2 are the tangential distortion coefficients.

2.2.4 Pinhole Radial-Tangential Model

Combining the pinhole camera model from Sec 2.2.1, radial distortion d_r and tangential distortion $\mathbf{d}_t$ from Sec 2.2.2 and Sec 2.2.3, we can combine all three models to form the



(a) No Distortion.

(b) Radial-Tangential Distortion.

Figure 2.3: Pinhole radial-tangential camera model.

pinhole radial-tangential camera model $\pi(\cdot)$ as follows,

$$\begin{aligned}
 \pi({}_c\mathbf{p}) &= \begin{bmatrix} f_u & 0 \\ 0 & f_v \end{bmatrix} \begin{bmatrix} x & d_r \\ y & d_r \end{bmatrix} + \begin{bmatrix} c_u \\ c_v \end{bmatrix} + \mathbf{d}_t, \\
 x &= X/Z \\
 y &= Y/Z \\
 r^2 &= x^2 + y^2 \\
 d_r &= (1 + k_1 r^2 + k_2 r^4), \\
 \mathbf{d}_t &= \begin{bmatrix} 2 p_1 x y + p_2 (r^2 + 2x^2) \\ 2 p_2 x y + p_1 (r^2 + 2y^2) \end{bmatrix},
 \end{aligned} \tag{2.10}$$

where ${}_c\mathbf{p} = [X \ Y \ Z]^\top$ is a 3D point in observed in the camera frame, (x, y) is the undistorted point, (f_u, f_v, c_u, c_v) are the 4 pinhole camera model parameters corresponding to the focal lengths and principal centers in pixels, (k_1, k_2, p_1, p_2) are the 4 radial and tangential distortion parameters.

2.3 Probabilistic Modelling in Robotics

For robots to traverse and interact with the world environment, robots first need to model the world and/or the robot state using available sensor data such as cameras, IMU, GPS, Lidar and more, whilst drawing on prior knowledge. There are many different inference problems in robotics, from localising within a *known* environment, tracking dynamic objects, to more specialised inference problems such as calibration. Simultaneous Localisation and Mapping (SLAM) in particular has gained a lot of traction and attention in recent years, where the goal is to simultaneously map the world and localise itself within the map it inferred. Solving these inference problems has made a variety of applications possible, from robotics and self-driving cars to augmented reality and virtual reality.

The challenge, however, is that sensor measurements often contain measurement noise, and therefore there will be uncertainty associated with the model or parameters the robot is inferring. While the absolute true state of the world or parameters cannot be recovered from noisy sensor measurements, we can use probability to obtain a probabilistic description of what we can infer from the sensor measurements.

In the following we will briefly introduce probability theory concepts commonly used in robotics, the reader is encouraged to review [Thrun et al., 2005]. Concepts described in this section is used throughout this thesis to estimate calibration parameters and state-estimation tasks.

2.3.1 Probability Theory

In probability theory, random variables are mathematical entities that model uncertain quantities or events that are subject to variability or randomness. In the context of robotics these random variables can be sensor measurements, robot states, and the environment. The process of inferring these random variables using the laws of probability theory is called probabilistic inference.

Probability density functions (PDF), $p(\mathbf{x})$, are commonly used to model a belief over

continuous multivariate random variables $\mathbf{x} \in \mathbb{R}^n$, satisfying,

$$\int p(\mathbf{x}) d\mathbf{x} = 1. \quad (2.11)$$

The *joint probability* of two *independent* multivariate random variables $\mathbf{x}$ and $\mathbf{y}$ is given as

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x}) p(\mathbf{y}). \quad (2.12)$$

However, random variables often carry information about other variables. For example, in robotics we are given measurements $\mathbf{z}$, and we would like to know the probability of the state $\mathbf{x}$ conditioned by $\mathbf{z}$ denoted

$$p(\mathbf{x} | \mathbf{z}) = \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{z})}, \quad (2.13)$$

where the numerator is the joint probability density function of $\mathbf{x}$ and $\mathbf{z}$, and the denominator is the marginal probability density function of $\mathbf{z}$. This is called the *conditional probability* of $\mathbf{x}$ given $\mathbf{z}$.

Multivariate Normal distribution (MVN) often abbreviated as $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma)$ is a very common density function that is used in robotics, it is given as:

$$p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma) \quad (2.14)$$

$$= \det(2\pi\Sigma)^{-\frac{1}{2}} \exp \left\{ -\frac{1}{2} \|\mathbf{x} - \boldsymbol{\mu}\|_{\Sigma}^2 \right\}, \quad (2.15)$$

where $\boldsymbol{\mu}$ is the mean vector and Σ is the covariance matrix, and

$$\|\mathbf{x} - \boldsymbol{\mu}\|_{\Sigma}^2 \triangleq (\mathbf{x} - \boldsymbol{\mu})^{\top} \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu}) \quad (2.16)$$

denotes the squared Mahalanobis distance. The Multivariate Normal distribution allows us to model the sensor noise as a Gaussian whilst providing us with a closed form probability density function and moments such as mean, variance and covariance. It is important to note that in practice not all measurement noise is Gaussian, the specific probability density function used will depend on the problem and application domain.

2.3.2 Bayes Rule

Bayes rule is a fundamental principle in probability theory that provides a powerful framework for reasoning under *uncertainty*. It describes a framework that enables the probability of a hypothesis to be updated based on new evidence, and refine the predictions or hypothesis in an iterative manner. In the context of robotics, the hypothesis is typically the state of the robot and / or map of the environment, and the new evidence is the new sensor measurements. The rule mathematically combines the inverse conditional probability $p(\mathbf{z} | \mathbf{x})$ along with prior $p(\mathbf{x})$ and marginal $p(\mathbf{z})$ densities to calculate a posterior probability $p(\mathbf{x} | \mathbf{z})$.

Using the joint probability and conditional probability equations Eq. 2.12 and Eq. 2.13 we can derive Bayes's rule. We begin the derivation by taking advantage that

$$p(\mathbf{x}, \mathbf{z}) = p(\mathbf{z}, \mathbf{x}), \quad (2.17)$$

and by rewriting the conditional probability Eq. 2.13 as

$$p(\mathbf{z} | \mathbf{x}) = \frac{p(\mathbf{z}, \mathbf{x})}{p(\mathbf{x})}, \quad (2.18)$$

$$p(\mathbf{z}, \mathbf{x}) = p(\mathbf{z} | \mathbf{x}) p(\mathbf{x}), \quad (2.19)$$

and substituting Eq. 2.17 and Eq. 2.19 back into Eq. 2.13 we arrive at:

$$p(\mathbf{x} | \mathbf{z}) = \frac{p(\mathbf{x}, \mathbf{z})}{p(\mathbf{z})}, \quad (2.20)$$

$$p(\mathbf{x} | \mathbf{z}) = \frac{p(\mathbf{z}, \mathbf{x})}{p(\mathbf{z})}, \quad (2.21)$$

$$p(\mathbf{x} | \mathbf{z}) = \frac{p(\mathbf{z} | \mathbf{x}) p(\mathbf{x})}{p(\mathbf{z})}, \quad (2.22)$$

where $p(\mathbf{x} | \mathbf{z})$ is known as the *posterior* density, $p(\mathbf{z} | \mathbf{x})$ as the *measurement density*, $p(\mathbf{x})$ as the *prior*, and $p(\mathbf{z})$ as the marginal density.

It is important to note that the denominator of Bayes rule does not depend on the unknown $\mathbf{x}$, thus the factor $p(\mathbf{z})$ will be the same for any value of $\mathbf{x}$. For this reason $p(\mathbf{z})$ is often written as a normaliser constant η :

$$p(\mathbf{x} | \mathbf{z}) = \eta p(\mathbf{z} | \mathbf{x}) p(\mathbf{x}). \quad (2.23)$$

2.3.3 Maximum a Posteriori Inference

Armed with a framework to infer unknown random variables $\mathbf{x}$ given sensor measurements $\mathbf{z}$ with the Bayes rule in Sec. 2.3.2, the most often used estimator to solve this problem is the Maximum a Posteriori (MAP) estimate, where the aim is to maximize the posterior density $p(\mathbf{x} | \mathbf{z})$:

$$\mathbf{x}^{\text{MAP}} = \underset{\mathbf{x}}{\operatorname{argmax}} p(\mathbf{x} | \mathbf{z}) \quad (2.24)$$

$$= \underset{\mathbf{x}}{\operatorname{argmax}} \frac{p(\mathbf{z} | \mathbf{x}) p(\mathbf{x})}{p(\mathbf{z})}, \quad (2.25)$$

where $p(\mathbf{z} | \mathbf{x})$ is the measurement density, $p(\mathbf{x})$ is the prior density, and $p(\mathbf{z})$ is the normalization factor. In practice, however, the normalization factor $p(\mathbf{z})$ is constant and irrelevant in the context of maximization, therefore it is dropped. Additionally, we can rewrite the measurement density $p(\mathbf{z} | \mathbf{x})$ as a likelihood function of the unknown random variables $\mathbf{x}$ given measurements $\mathbf{z}$ as $l(\mathbf{x}; \mathbf{z})$, and it is defined as any function proportional to the measurement density $p(\mathbf{z} | \mathbf{x})$:

$$l(\mathbf{x}; \mathbf{z}) \propto p(\mathbf{z} | \mathbf{x}). \quad (2.26)$$

It is important to note that the likelihood function is a function of the unknown random variables $\mathbf{x}$ and not the measurements $\mathbf{z}$. We assume the measurements are given, therefore they are treated simply as constants in the likelihood function. Dropping the normaliser factor $p(\mathbf{z})$ and incorporating the likelihood function $l(\mathbf{x}; \mathbf{z})$ we arrive at:

$$\mathbf{x}^{\text{MAP}} = \underset{\mathbf{x}}{\operatorname{argmax}} l(\mathbf{x}; \mathbf{z}) p(\mathbf{x}). \quad (2.27)$$

In summary, the MAP estimate is obtained by maximizing the likelihood function and prior, and provides an estimate of the unknown random variables given observed measurements.

There are several approaches to computing the MAP estimate. Namely analytically, if the model of the posterior distribution has a closed form. Alternatively, the MAP estimate can be computed with the Expectation-Maximization (EM) algorithm or with Monte Carlo methods, if derivatives of the posterior density are not available. In robotics, however,

the MAP estimate is often estimated using nonlinear least squares. In the following, we show that MAP inference with Gaussian likelihood functions is equivalent to solving the nonlinear least squares problem.

First, suppose our prior $p(\mathbf{x})$ and non-linear measurement functions $\mathbf{h}(\cdot)$ are both modelled with a Gaussian normal distribution, such that $p(\mathbf{x}) = \mathcal{N}(\mathbf{x}_0, \mathbf{P})$, and $p(\mathbf{z} | \mathbf{x}) = \mathcal{N}(\mathbf{h}(\mathbf{x}), \mathbf{R})$. Given the likelihood function is defined as proportional to the measurement density, $l(\mathbf{x}; \mathbf{z}) \propto p(\mathbf{z} | \mathbf{x})$, we can rewrite the MAP estimate in Eq. (2.27) as:

$$\mathbf{x}^{\text{MAP}} = \underset{\mathbf{x}}{\operatorname{argmax}} \quad l(\mathbf{x}; \mathbf{z}) p(\mathbf{x}) \quad (2.28)$$

$$= \underset{\mathbf{x}}{\operatorname{argmin}} \quad -\log(p(\mathbf{z}, \mathbf{x}) p(\mathbf{x})) \quad (2.29)$$

$$= \underset{\mathbf{x}}{\operatorname{argmin}} \quad -\log \left\{ \left(\prod_{i=1}^N p(\mathbf{z}_i, \mathbf{x}) \right) p(\mathbf{x}) \right\} \quad (2.30)$$

$$= \underset{\mathbf{x}}{\operatorname{argmin}} \quad -\log \left\{ \left(\prod_{i=1}^N \exp \left\{ -\frac{1}{2} \|\mathbf{z}_i - \mathbf{h}_i(\mathbf{x})\|_{\mathbf{R}_i}^2 \right\} \right) \left(\exp \left\{ -\frac{1}{2} \|\mathbf{x}_0 - \mathbf{x}\|_{\mathbf{P}}^2 \right\} \right) \right\}. \quad (2.31)$$

Expanding the probability density functions in Eq. (2.29) with Gaussian normal distributions $\mathcal{N}(\mathbf{h}(\mathbf{x}), \mathbf{R})$ and $\mathcal{N}(\mathbf{x}_0, \mathbf{P})$, taking the negative log and dropping the constants allows us to minimize a sum of nonlinear least squares of the form:

$$\mathbf{x}^{\text{MAP}} = \underset{\mathbf{x}}{\operatorname{argmin}} \quad \frac{1}{2} \left\{ \left(\sum_{i=1}^N \|\mathbf{z}_i - \mathbf{h}_i(\mathbf{x})\|_{\mathbf{R}_i}^2 \right) + \|\mathbf{x}_0 - \mathbf{x}\|_{\mathbf{P}}^2 \right\}. \quad (2.32)$$

Alternatively, if we denote $\mathbf{e}_i$ as a residual error as a function of the state vector $\mathbf{x}$ with corresponding information matrix $\mathbf{W}_i$, we can rewrite the above in a more general form,

$$\mathbf{x}^{\text{MAP}} = \underset{\mathbf{x}}{\operatorname{argmin}} \quad \frac{1}{2} \sum_i \mathbf{e}_i(\mathbf{x})^\top \mathbf{W}_i \mathbf{e}_i(\mathbf{x}). \quad (2.33)$$

Minimizing this objective function using a nonlinear least squares algorithm effectively maximizes the posterior $p(\mathbf{x}, \mathbf{z})$ of unknown state $\mathbf{x}$ given sensor measurements $\mathbf{z}$.

2.3.4 Nonlinear Least Squares

In modern robotics, we are often interested in estimating the state vector, $\mathbf{x} \in \mathbb{R}^n$, which may comprise of the robot state such as position, rotation and velocity, landmarks of the

world, as well as other calibration quantities such as sensor intrinsics and extrinsic, and more. The aim is to find an optimal state $\mathbf{x}$, that minimizes the nonlinear least squares cost of the form:

$$J(\mathbf{x}) = \underset{\mathbf{x}}{\operatorname{argmin}} \quad \frac{1}{2} \sum_i \mathbf{e}_i(\mathbf{x})^\top \mathbf{W}_i \mathbf{e}_i(\mathbf{x}), \quad (2.34)$$

or in matrix form,

$$J(\mathbf{x}) = \underset{\mathbf{x}}{\operatorname{argmin}} \quad \frac{1}{2} \mathbf{e}(\mathbf{x})^\top \mathbf{W} \mathbf{e}(\mathbf{x}), \quad (2.35)$$

where the error function, $\mathbf{e}_i(\cdot)$, depends on the state vector, $\mathbf{x}$. The error function, $\mathbf{e}(\cdot)$, has a form of:

$$\mathbf{e}_i(\mathbf{x}) = \mathbf{z} - \mathbf{h}_i(\mathbf{x}), \quad (2.36)$$

and it is defined as the difference between the measured value, $\mathbf{z}$, and the estimated value calculated using the measurement function, $\mathbf{h}_i(\cdot)$. If the error function $\mathbf{e}_i(\mathbf{x})$ is *linear*, we can find the minimum of the linear least squares in Eq. (2.35), by setting the derivative of the cost with respect to the state to zero (i.e. $\frac{\partial J(\mathbf{x})}{\partial \mathbf{x}} = 0$), and solve the resulting normal equations in one step. In robotics, however, the error function is often *nonlinear*, therefore the solution has to be solved iteratively. A popular algorithm in solving the nonlinear least squares is the *Gauss-Newton method*, which we will briefly describe below.

Gauss Newton

In an iterative nonlinear least squares optimisation, our goal is to iteratively update the current state estimate $\bar{\mathbf{x}}$ with a small state update vector $\delta \mathbf{x}$, such that it minimizes the cost in Eq. (2.35), with

$$\mathbf{x} = \bar{\mathbf{x}} + \delta \mathbf{x}. \quad (2.37)$$

Approximating the nonlinear error function $\mathbf{e}(\cdot)$ with the first-order Taylor series by linearising about $\bar{\mathbf{x}}$, we have

$$\mathbf{e}(\mathbf{x}) = \mathbf{e}(\bar{\mathbf{x}}) + \left. \frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\bar{\mathbf{x}}} \delta \mathbf{x}. \quad (2.38)$$

Substituting Eq. (2.37) and Eq. (2.38) into Eq. (2.35) we get,

$$J(\mathbf{x}) = \frac{1}{2} \mathbf{e}(\mathbf{x})^\top \mathbf{W} \mathbf{e}(\mathbf{x}) \quad (2.39)$$

$$J(\bar{\mathbf{x}} + \delta \mathbf{x}) = \frac{1}{2} \mathbf{e}(\bar{\mathbf{x}} + \delta \mathbf{x})^\top \mathbf{W} \mathbf{e}(\bar{\mathbf{x}} + \delta \mathbf{x}) \quad (2.40)$$

$$\approx \frac{1}{2} \left(\mathbf{e}(\bar{\mathbf{x}}) + \left. \frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\bar{\mathbf{x}}} \delta \mathbf{x} \right)^\top \mathbf{W} \left(\mathbf{e}(\bar{\mathbf{x}}) + \left. \frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\bar{\mathbf{x}}} \delta \mathbf{x} \right), \quad (2.41)$$

which we can simplify to

$$J(\delta \mathbf{x}) = \frac{1}{2} \mathbf{e}(\delta \mathbf{x})^\top \mathbf{W} \mathbf{e}(\delta \mathbf{x}) \quad (2.42)$$

where

$$\mathbf{E} = \left. \frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}} \right|_{\bar{\mathbf{x}}}, \quad \mathbf{e}(\delta \mathbf{x}) = \mathbf{e}(\bar{\mathbf{x}}) + \mathbf{E} \delta \mathbf{x}. \quad (2.43)$$

Now, to find the minimum of Eq. (2.42) we set the derivative of the cost $J(\delta \mathbf{x})$ with respect to the update state $\delta \mathbf{x}$ to zero (i.e. $\frac{\partial J(\delta \mathbf{x})}{\partial \delta \mathbf{x}} = \mathbf{0}$). First, let's expand $\frac{\partial J(\delta \mathbf{x})}{\partial \delta \mathbf{x}}$ with the chain rule to obtain

$$\frac{\partial J(\delta \mathbf{x})}{\partial \delta \mathbf{x}} = \frac{\partial J(\delta \mathbf{x})}{\partial \mathbf{e}(\delta \mathbf{x})} \frac{\partial \mathbf{e}(\delta \mathbf{x})}{\partial \delta \mathbf{x}}, \quad (2.44)$$

solving for $\frac{\partial J(\delta \mathbf{x})}{\partial \mathbf{e}(\delta \mathbf{x})}$ and $\frac{\partial \mathbf{e}(\delta \mathbf{x})}{\partial \delta \mathbf{x}}$, we get

$$\frac{\partial J(\delta \mathbf{x})}{\partial \mathbf{e}(\delta \mathbf{x})} = \frac{\partial \left(\frac{1}{2} \mathbf{e}(\delta \mathbf{x})^\top \mathbf{W} \mathbf{e}(\delta \mathbf{x}) \right)}{\partial \mathbf{e}(\delta \mathbf{x})} = \mathbf{e}(\delta \mathbf{x})^\top \mathbf{W}, \quad (2.45)$$

$$\frac{\partial \mathbf{e}(\delta \mathbf{x})}{\partial \delta \mathbf{x}} = \frac{\partial (\mathbf{e}(\bar{\mathbf{x}}) + \mathbf{E} \delta \mathbf{x})}{\partial \delta \mathbf{x}} = \mathbf{E}. \quad (2.46)$$

Substituting them back into Eq. (2.44), setting the derivative to zero, and solving the linearised system of equations for $\delta \mathbf{x}$,

$$\frac{\partial J(\delta \mathbf{x})}{\partial \delta \mathbf{x}} = (\mathbf{e}(\delta \mathbf{x})^\top \mathbf{W}) (\mathbf{E}) \quad (2.47)$$

$$\mathbf{0} = ((\mathbf{e}(\bar{\mathbf{x}}) + \mathbf{E} \delta \mathbf{x})^\top \mathbf{W}) \mathbf{E} \quad (2.48)$$

$$\mathbf{0} = (\mathbf{e}(\bar{\mathbf{x}})^\top \mathbf{W} + \delta \mathbf{x}^\top \mathbf{E}^\top \mathbf{W}) \mathbf{E} \quad (2.49)$$

$$\mathbf{0} = \mathbf{e}(\bar{\mathbf{x}})^\top \mathbf{W} \mathbf{E} + \delta \mathbf{x}^\top \mathbf{E}^\top \mathbf{W} \mathbf{E} \quad (2.50)$$

$$\delta \mathbf{x}^\top \mathbf{E}^\top \mathbf{W} \mathbf{E} = -\mathbf{e}(\bar{\mathbf{x}})^\top \mathbf{W} \mathbf{E} \quad (2.51)$$

$$\mathbf{E}^\top \mathbf{W} \mathbf{E} \delta \mathbf{x} = -\mathbf{E}^\top \mathbf{W} \mathbf{e}(\bar{\mathbf{x}}), \quad (2.52)$$

we arrive at the normal equations which we typically solve for $\delta \mathbf{x}$ using Cholesky or QR-decomposition. Once we have $\delta \mathbf{x}$, we can update our current estimate with

$$\bar{\mathbf{x}} \leftarrow \bar{\mathbf{x}} + \delta \mathbf{x} \quad (2.53)$$

to hopefully converge to a more optimal state estimate. This iterative optimisation process continues until the cost converges to a minimum, sufficiently small, or a termination criteria has been reached (such as the number of iterations has reached its maximum).

With the Jacobian $\mathbf{E} = \frac{\partial \mathbf{e}(\mathbf{x})}{\partial \mathbf{x}}|_{\bar{\mathbf{x}}}$, we can approximate the local gradient $\nabla J(\mathbf{x})|_{\bar{\mathbf{x}}} = 2\mathbf{E}^\top \mathbf{e}$ at the linearisation point $\bar{\mathbf{x}}$. Using *Gradient Descent*, the state is updated with the following rule:

$$\delta \mathbf{x} = -\alpha \nabla J(\mathbf{x})|_{\bar{\mathbf{x}}}. \quad (2.54)$$

However, the effectiveness of this approach is heavily dependent on the choice of the step size α . If α is not chosen carefully, convergence can be slow or even divergent. The *Gauss Newton method* on the other hand approximates the Hessian matrix $\mathbf{H} = \mathbf{E}^\top \mathbf{W} \mathbf{E}$ (second derivative), which captures information about the curvature of the error function. This curvature information allows the Gauss-Newton method to take larger steps in the direction of improvement compared to Gradient Descent with a fixed update step dictated by the step size α . This can significantly accelerate convergence towards the local minimum.

It is important to note that computing the Hessian can be computationally expensive, and the Gauss-Newton method might not always be the best choice, particularly when dealing with ill-conditioned Hessians.

Levenberg-Marquardt Algorithm

The *Levenberg-Marquardt algorithm* improves robustness and convergence properties relative to the Gauss-Newton method, by combining both the advantages of Gradient Descent and the Gauss-Newton method. [Levenberg, 1944] proposed a damping factor λ to the normal equations of the Gauss-Newton method (Eq. 2.52) to control the step size and

direction:

$$(\mathbf{E}^\top \mathbf{W} \mathbf{E} + \lambda \mathbf{I}) \delta \mathbf{x} = -\mathbf{E}^\top \mathbf{W} \mathbf{e}(\bar{\mathbf{x}}). \quad (2.55)$$

The damping factor λ is adjusted dynamically depending on whether the step was successful in converging to a local minima. For $\lambda = 0$ the algorithm effectively becomes a Gauss-Newton, and for large λ the algorithm becomes a Gradient Descent algorithm. [Marquardt, 1963] later modified the damping to take into account the scaling of the diagonal of the Hessian matrix for faster convergence,

$$(\mathbf{E}^\top \mathbf{W} \mathbf{E} + \lambda \text{diag}(\mathbf{E}^\top \mathbf{W} \mathbf{E})) \delta \mathbf{x} = -\mathbf{E}^\top \mathbf{W} \mathbf{e}(\bar{\mathbf{x}}). \quad (2.56)$$

Algorithm 1 Levenberg-Marquardt Algorithm

```

1: procedure LEVENBERGMARQUARDT(  $J(\cdot)$ ,  $\mathbf{x}_0$ ,  $k_{\max}$  )
2:    $\lambda = 1 \times 10^{-4}$  ▷ Damping Factor
3:    $k = 0$  ▷ Iterations
4:    $\bar{\mathbf{x}}_k = \mathbf{x}_0$  ▷ Current state estimate
5:
6:   while !converged &&  $k < k_{\max}$  do
7:      $\mathbf{E}, \mathbf{e} \leftarrow \text{linearise}(J(\bar{\mathbf{x}}_k))$  ▷ Linearise
8:      $\delta \mathbf{x} \leftarrow (\mathbf{E}^\top \mathbf{W} \mathbf{E} + \lambda \text{diag}(\mathbf{E}^\top \mathbf{W} \mathbf{E})) \delta \mathbf{x} = -\mathbf{E}^\top \mathbf{W} \mathbf{e}(\bar{\mathbf{x}}_k)$  ▷ Solve  $\delta \mathbf{x}$ 
9:
10:    if  $J(\bar{\mathbf{x}}_k + \delta \mathbf{x}) < J(\bar{\mathbf{x}}_k)$  then
11:       $\bar{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_k + \delta \mathbf{x}$  ▷ Accept Update
12:       $\lambda = \lambda / 10$ 
13:    else
14:       $\bar{\mathbf{x}}_{k+1} = \bar{\mathbf{x}}_k$  ▷ Reject Update
15:       $\lambda = \lambda \times 10$ 
16:    end if
17:
18:     $k = k + 1$ 
19:  end while
20:
21:  return  $\bar{\mathbf{x}}_k$ 
22: end procedure
    
```

A sketch of the Levenberg-Marquardt algorithm is given in Algorithm. 1. Aside from the damping factor λ mentioned above, a key improvement upon Gauss-Newton is that the Levenberg-Marquardt algorithm only accepts state updates if the update leads to a lower

cost. If the cost is lower after applying the current state update, the damping factor λ is reduced by a factor of 10 (turning into a Gauss-Newton). Conversely, if the cost is higher after the current state update, the update is rejected and λ is increased by 10 (turning into a Gradient Descent). The algorithm repeats with a new linearisation point until either the algorithm has converged to a local minima, or a termination criteria has been reached such as the maximum number of iterations.

2.4 Information Theory

Information Theory was developed by Claud Shannon in 1948, where he formed the mathematical framework for understanding the fundamental limits and capabilities of transmitting and receiving information [Shannon, 1948]. A key concept in information theory is the quantization of information as entropy. It measures the amount of uncertainty in a random variable or outcome of a random process. Over the years this has had a large impact in the development of compression algorithms and error detection / correction, leading to the invention of compact disc, the feasibility of mobile phones and the internet. In the context of robotics, information theory has been used to guide planning algorithms by maximizing the information gain in uncertain environments. In localisation and mapping it has been used as an information-theoretic criteria to influence which sensor measurements are to be used in the optimisation framework, in order to minimize the uncertainty of the estimated poses and landmarks.

The following section briefly touches on the core concepts of information theory used in Chapter. 4, specifically in the context of visual-inertial calibration we are interested in knowing which camera view or trajectory is the most informative for calibration. We first begin by introducing entropy, joint entropy, conditional entropy and mutual information. Then finish with how the concepts are applicable in the continuous domain and in robotics. For a more detailed comprehensive introduction to information theory the reader is encouraged to review [MacKay, 2003, Cover and Thomas, 2006]

2.4.1 Information Content

In information theory, the *information content*, $I(X)$, also called the *degree of surprise*, of a *discrete* random variable, X , is defined as:

$$I(X) = -\log_b p(x) \quad (2.57)$$

where $p(x) = \Pr\{X = x\}$, $x \in X$ is the probability mass function of the random variable X , and the basis of the logarithm b determines the unit of the information content. If $b = 2$ the unit of information is in *bits*, where as if $b = e$ the unit is in *nats*. It is important to note that information theory focuses on the quantitative aspects of information, not the content nor meaning conveyed by the information.

2.4.2 Entropy

The expected value of the information content of a discrete random variable X is called the *information entropy* $H(X)$, and it is defined as:

$$H(X) = -\sum_x p(x) \log_b p(x) \quad (2.58)$$

$$= \mathbb{E} [\log_b p(x)]. \quad (2.59)$$

Intuitively, entropy represents the average uncertainty of the random variable in its current state, and is often described as the *expected surprise* of the distribution. The entropy reaches its maximum value if all possible values for X is equiprobable.

2.4.3 Joint Entropy

In the above, we have defined the entropy for a *single* discrete random variable. Here, we introduce another discrete random variable Y and define the *joint entropy* $H(X, Y)$ between a pair of discrete random variables as:

$$H(X, Y) = -\sum_x \sum_y p(x, y) \log_b p(x, y). \quad (2.60)$$

If the random variables X and Y are independent such that $p(x, y) = p(x)p(y)$, then it can be shown that the joint entropy is equivalent to

$$H(X, Y) = H(X) + H(Y). \quad (2.61)$$

2.4.4 Conditional Entropy

Another common scenario is when information content of one random variable is *correlated* with another. This means learning the value of one random variable X can affect the uncertainty of the other correlated random variable Y . The *conditional entropy* $H(X|Y)$ is defined as:

$$H(X|Y) = - \sum_y p(y) \sum_x p(x|y) \log_b p(x|y) \quad (2.62)$$

$$= - \sum_x \sum_y p(x, y) \log_b p(x|y). \quad (2.63)$$

From the entropy, joint entropy and conditional entropy in Eq. 2.58, Eq. 2.60 and Eq. 2.62, we can see that

$$H(X|Y) = H(X, Y) - H(Y). \quad (2.64)$$

From:

$$H(X, Y) = - \sum_x \sum_y p(x, y) \log_b p(x, y) \quad (2.65)$$

$$= - \sum_x \sum_y p(x, y) \log_b p(x)p(y|x) \quad (2.66)$$

$$= - \sum_x \sum_y p(x, y) \log_b p(x) - \sum_x \sum_y p(x, y) \log_b p(y|x) \quad (2.67)$$

$$= - \sum_x p(x) \log_b p(x) - \sum_x \sum_y p(x, y) \log_b p(y|x) \quad (2.68)$$

$$= H(X) + H(X|Y). \quad (2.69)$$

Rearranging with respect to $H(X|Y)$ we arrive at:

$$H(X|Y) = H(X, Y) - H(X). \quad (2.70)$$

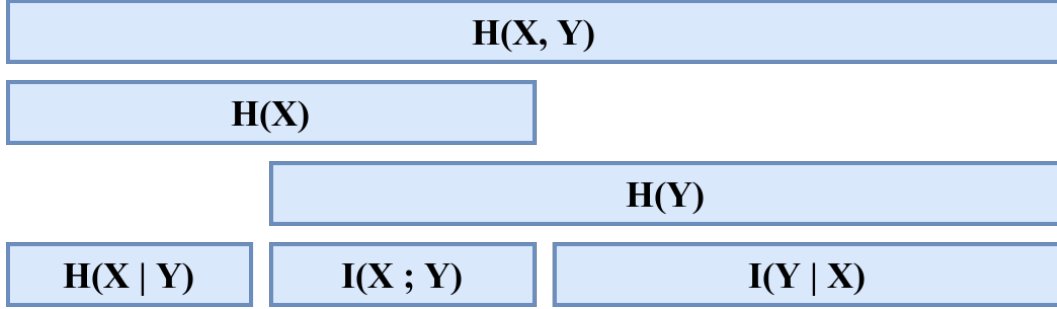


Figure 2.4: Relationship between joint, marginal and conditional entropies with mutual information. This figure is based on an illustration from [MacKay, 2003].

2.4.5 Mutual Information

Previously, we have defined the entropy of two discrete random variables X and Y that are correlated with another. Here we will define a measure of the amount of information of one random variable shares about the another random variable, called *mutual information*. The mutual information $I(X; Y)$ between random variables X and Y has several forms and they are defined as follows,

$$I(X; Y) = H(X) + H(Y) - H(X, Y), \quad (2.71)$$

$$= H(X) - H(X|Y), \quad (2.72)$$

$$= H(Y) - H(Y|X), \quad (2.73)$$

where it combines the individual entropies and compares it against the joint entropy. Alternatively using the conditional entropies, the mutual information measures how much uncertainty about X is reduced by knowing Y and vice versa. For example, if X and Y are *independent* their mutual information would be zero, because expanding $H(X, Y)$ in Eq. (2.71) would cancel out $H(X)$ and $H(Y)$. On the other hand, if $H(X)$ is the measure of uncertainty of X , and $H(X|Y)$ is the measure of uncertainty of X after Y is known, then Eq. (2.72) is the uncertainty of X minus the amount of uncertainty of X given Y . Lastly, the mutual information is symmetric $I(X; Y) = I(Y; X)$, therefore Eq. (2.72) and Eq. (2.73) are equivalent.

2.4.6 Continuous Variables

Thus far, it is important to note that the above sections on information theory was introduced with *discrete* random variables in mind. Here, we extend the concepts to *continuously* distributed random variables by first extending entropy to *differential entropy* as

$$h(X) = - \int_x p(x) \log_b p(x) dx, \quad (2.74)$$

where $p(x)$ is a probability density function for X . While it maybe deceiving to think we have simply replaced the summation in Eq. (2.58) with an integral in Eq. (2.74), special care needs to be taken as the differential entropy can be negative and it is not invariant under general transformations.

Suppose a continuous multivariate random variable is denoted as $\mathbf{X}$, the multivariate *joint entropy* is therefore defined as,

$$h(\mathbf{X}) = - \int_{\mathbf{x}} p(\mathbf{x}) \log_b p(\mathbf{x}) d\mathbf{x}. \quad (2.75)$$

Similarly, we can redefine the *conditional entropy* of $\mathbf{X}$, given another continuous multivariate random variable $\mathbf{Z}$ as,

$$h(\mathbf{X}|\mathbf{Z}) = \int_{\mathbf{z}} \int_{\mathbf{x}} p(\mathbf{x}, \mathbf{z}) \log_b \frac{p(\mathbf{z})}{p(\mathbf{x}|\mathbf{z})} d\mathbf{x} d\mathbf{z}. \quad (2.76)$$

Finally, the continuous multivariate *mutual information* is given as

$$h(\mathbf{X}|\mathbf{Z}) = \int_{\mathbf{z}} \int_{\mathbf{x}} p(\mathbf{x}, \mathbf{z}) \log_b \frac{p(\mathbf{x}|\mathbf{z})}{p(\mathbf{z})} d\mathbf{x} d\mathbf{z}, \quad (2.77)$$

where it measures the average amount of information obtained on $\mathbf{X}$, given observations on $\mathbf{Z}$. It is also a measure of independence of the random variables.

2.4.7 Mutual Information in Robotics

Now that we have covered some of the most important concepts in information theory, we can demonstrate the utility of mutual information in the context of state estimation in robotics. Suppose we are solving a robot localisation problem, where our robot has a single

camera and it is attempting to localise itself in a *known* environment. Ordinarily, cameras on a robot are fixed to the body of the robot, and it would simply utilise landmarks observed to locate itself. However, this is often not ideal since not all landmarks are informative for localisation (e.g. landmarks have varying degrees of uncertainty in terms of determining the robot's position, because of this, highly uncertain landmarks may have a negative impact on localisation). Additionally, the camera view is at the mercy of the robot's position and orientation, if it is not observing informative landmarks the localisation could fail. To remedy this, what if we can find the next best camera pose that can observe landmarks that are highly informative for the problem of localisation. In the following we will sketch out of how we could apply mutual information in selecting the next best camera pose for localisation.

We begin solving the localisation problem using a nonlinear least squares algorithm such as the Gauss Newton method, given in matrix form as

$$J(\mathbf{x}) = \underset{\mathbf{x}}{\operatorname{argmin}} \frac{1}{2} \mathbf{e}(\mathbf{x})^\top \mathbf{W} \mathbf{e}(\mathbf{x}) \quad (2.78)$$

where $\mathbf{x}$ is the multivariate state vector which consists of camera poses $\mathbf{x}_C$, given sensor measurements $\mathbf{z}$, and measurement information matrix $\mathbf{W}$. At convergence of the nonlinear least squares optimisation, we obtain the mean $\boldsymbol{\mu}_x$ and the estimated covariance matrix $\hat{\Sigma}_x$ by inverting the quantity $\mathbf{E}^\top \mathbf{W} \mathbf{E}$ in Eq. (2.78), also known as the Fisher Information matrix.

Since our robot has a camera in this example, we are interested in finding the Next-Best-View (NBV) for the camera that lowers the uncertainty of the camera poses, we need to define a $\mathcal{D}_1 = \{\mathbf{z}_{1:k}\}$ as the set of *existing* landmark measurements from timestep 1 to k currently in the optimisation problem, and our goal is to find $\mathcal{D}_2 = \{\mathbf{z}_{1:k}, \mathbf{z}_{k+1}\}$ which are the next set of *predicted* landmark measurements at timestep $k + 1$. Effectively $\mathcal{D}_2$ represents *past and future measurements* given a camera pose, and while the next the camera pose $\mathbf{x}_C^{k+1}$ and landmark measurements $\mathcal{D}_2$ search space is infinite, we can limit our search space by sampling nearby camera positions and orientations to find the next camera pose that lowers our camera pose uncertainty.

One way to choose the most informative camera view for the next timestep is by evaluating the *mutual information* between the current sensor measurements and potential future sensor measurements. Mutual information tells us how much information one view reveals about the other. By selecting the view with the highest mutual information, we can significantly reduce the uncertainty about the camera's position in the next frame. Following an argument of [Davison, 2005], the mutual information between $\mathbf{x}$ and $\mathcal{D}_2$ is defined as:

$$I(\mathbf{x}; \mathcal{D}_2) = \frac{1}{2} \log \frac{|\Sigma_{\mathbf{x}|\mathcal{D}_1}|}{|\Sigma_{\mathbf{x}|\mathcal{D}_2}|}, \quad (2.79)$$

where $\Sigma_{\mathbf{x}|\mathcal{D}_1}$ is the covariance estimate of $\mathbf{x}$ using measurements $\mathcal{D}_1$ alone, and $\Sigma_{\mathbf{x}|\mathcal{D}_2}$ is the covariance estimate of $\mathbf{x}$ using measurements $\mathcal{D}_1$ and $\mathcal{D}_2$, finally $|\cdot|$ is the matrix determinant. In summary, we can use the mutual information in Eq. (2.79) to find the next best neighbourhood camera position and orientation to that minimizes the camera pose $\mathbf{x}$ as the robot traverses the known environment.

2.5 Brief Literature Review of Camera and Camera-IMU Calibration

With the advancement of robotics, the need to localise and map the robot's immediate surrounding environment in order for the robot to interact with it is of utmost importance. Cameras and IMUs are popular amongst roboticists and often used for localisation and mapping tasks due to its size, weight and low-cost. The combination of camera and IMU sensors allows for the fusion of visual and inertial data, enhancing the robustness of localisation and mapping tasks.

2.5.1 Camera Calibration

To extract *metric* information from 2D images captured by cameras, it is crucial for cameras to undergo accurate calibration. Camera calibration, a fundamental step to using computer vision algorithms with real data, has been extensively studied in the fields of computer

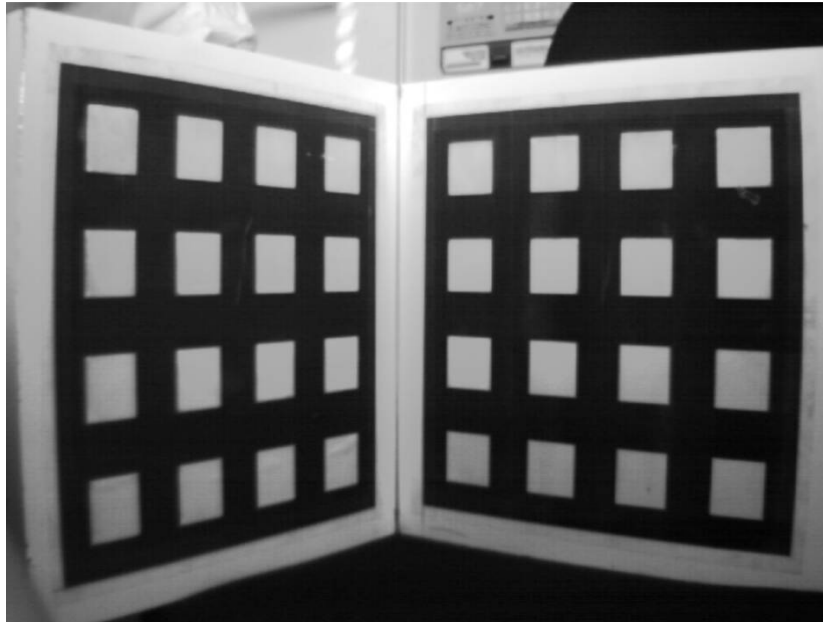


Figure 2.5: Tsai's camera calibration object. (Figure taken from [Devernay and Faugeras, 1995])

vision and photogrammetry, it involves estimating of intrinsic and extrinsic parameters of a camera(s). Intrinsic parameters include focal length, principal point coordinates, and lens distortion coefficients, while extrinsic parameters define the camera's position and orientation in 3D space, typically with respect to the calibration target at hand.

Tsai's camera calibration method [Tsai, 1987] was highly influential and formed the foundation for subsequent modern camera calibration techniques. The calibration method uses only a *single image* observing a known 3D calibration object, typically consisting of two or three planes orthogonal with each other (see Figure. 2.5). Each calibration point on the object is known with a high degree of precision. However, the reliance on a highly accurate 3D calibration object can render its production costly. Tsai's method calibrates the camera in two stages. First, the calibration method estimates the camera's rotation and translation in the x and y axis. Subsequently, the method employs an iterative nonlinear optimisation method to resolve the remaining parameters such as the camera's focal lengths, distortion coefficients, and the z -axis translation component of the camera's pose is estimated.

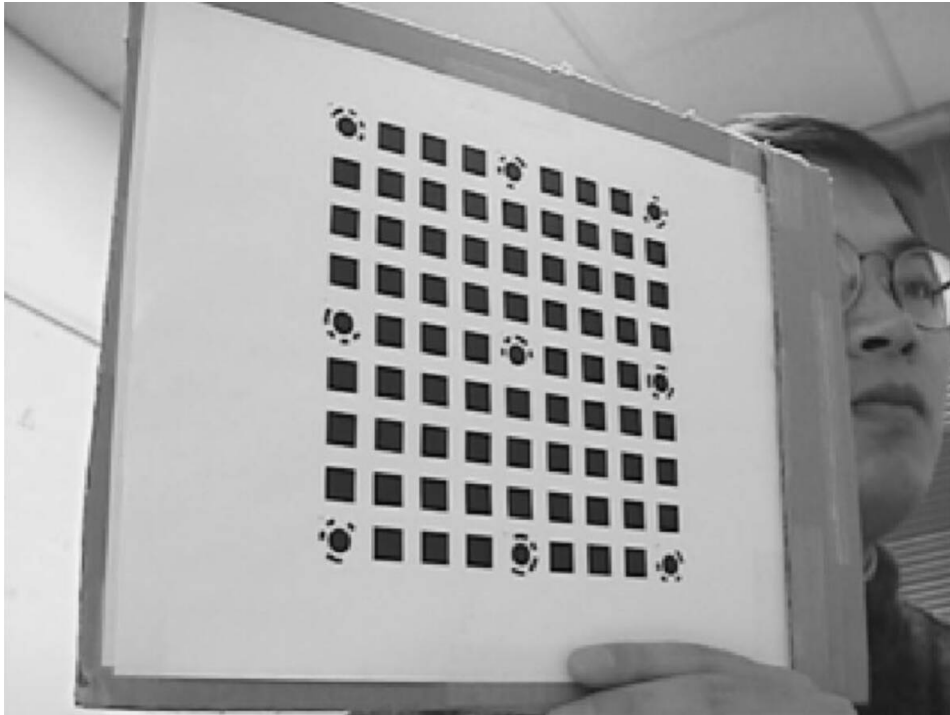


Figure 2.6: Zhang’s camera calibration method using planar targets. (Figure taken from [Zhang, 2000])

The drawback associated with the impracticality and high cost of a 3D calibration object, led to Zhang’s widely popular calibration method [Zhang, 2000]. The method also uses a 2 stage process similar to Tsai’s camera calibration method, however instead of using a 3D calibration object, Zhang used a planar calibration target with known calibration target points (see Figure. 2.6). The calibration process captures *multiple observations* of a calibration target from different camera positions and orientations. The calibration method first approximates the camera intrinsic and extrinsic parameters using the DLT method [Sutherland, 1974], then the parameters are further refined using a nonlinear optimisation method. With Zhang’s camera calibration method, many were able to collect multiple camera calibration views, perform camera calibration and obtain sub-pixel accuracy with ease. The flexibility and robustness meant it formed the basis for *offline* camera calibration software, such as MATLAB’s popular camera calibration toolbox [Bouguet, 2004], and later made its way into OpenCV’s camera calibration SDK [Bradski, 2000].

More recently, advancements have been made towards calibrating rolling shutter cameras [Grundmann et al., 2012, Oth et al., 2013] due to its wider availability, high dynamic range and resolution. Improvements have also been made on the camera modeling front, giving rise to additional camera models such as the Unified camera model [Geyer and Daniilidis, 2000], Kannala-Brandt’s camera model [Kannala and Brandt, 2006], and the Double Sphere camera model [Usenko et al., 2018], where they all aim to model large field of view cameras.

2.5.2 Camera-IMU Calibration

With an increasing number of state estimation algorithms emerging which leverage the complementary strengths of the camera and IMU [Mourikis and Roumeliotis, 2007, Leutenegger et al., 2014, Qin et al., 2018, Geneva et al., 2020, Campos et al., 2021], the need to resolve the spatial and temporal relationship between the camera and IMU must be accurately modeled in order to fuse measurements from both sensors. In most cases, in practice, the spatial relationship is obtained manually, either by measuring it by hand, obtained by inspecting the computer-aided design (CAD) model, or through the use of additional sensors. For example, one can obtain sub-degree and centimeter level accuracy by using a laser scanner to scan the IMU housing relative to a camera calibration target, specifically placed in front of a camera such that the relative spatial relationships between them can be resolved [Johnson et al., 2005b, Johnson et al., 2005a]. The disadvantage of this approach, however, is that it is often costly and impractical. Additionally, recalibration involving a laser scanner can be expensive and would likely be very time consuming, since recalibration needs to take place if the sensors are removed or repositioned either intentionally or unintentionally due to bumps or environmental condition changes (e.g. temperature).

One of the early works that attempted to tackle the camera IMU calibration problem was [Alves et al., 2003], where they estimated the relative rotation between camera and IMU as well as scale factors and axis misalignment in the IMU, by mounting the camera and IMU onto a pendulum equipped with a high-resolution encoder (see Figure. 2.7). Later,

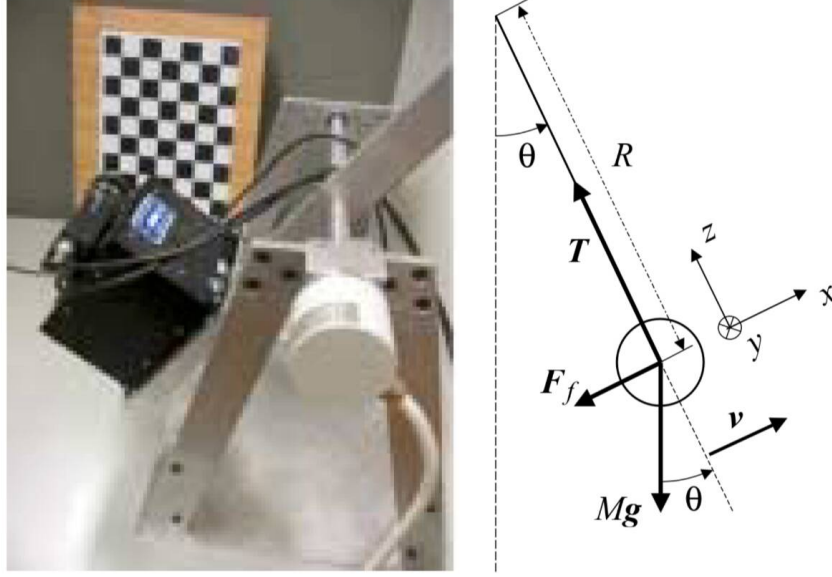


Figure 2.7: Pendulum used to calibrate the IMU sensor and forces acting on the moving pendulum. (Figure taken from [Alves et al., 2003])

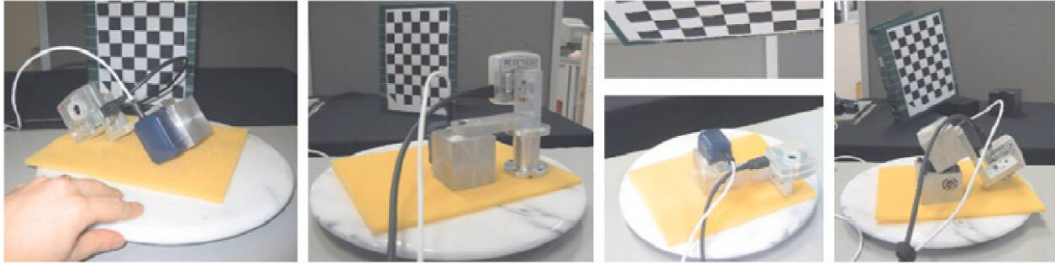


Figure 2.8: Camera-IMU placed on a turntable in different poses for translation calibration. (Figure taken from [Lobo and Dias, 2007])

in [Lobo and Dias, 2007] they estimated the relative orientation of the camera and IMU by taking still images of a gravity aligned calibration target, and extended their previous work by estimating the relative displacement of the camera and IMU in a separate process, by mounting the device on a turntable and aligning the IMU in such a way that it is not affected by accelerations other than gravity (see Figure. 2.8). The main drawback of this approach is that it ignores the time offset between camera and IMU, as well as IMU biases. Furthermore, this decoupled two stage estimation of the relative spatial relationship between the camera and IMU allows for errors to propagate between the two calibration stages.

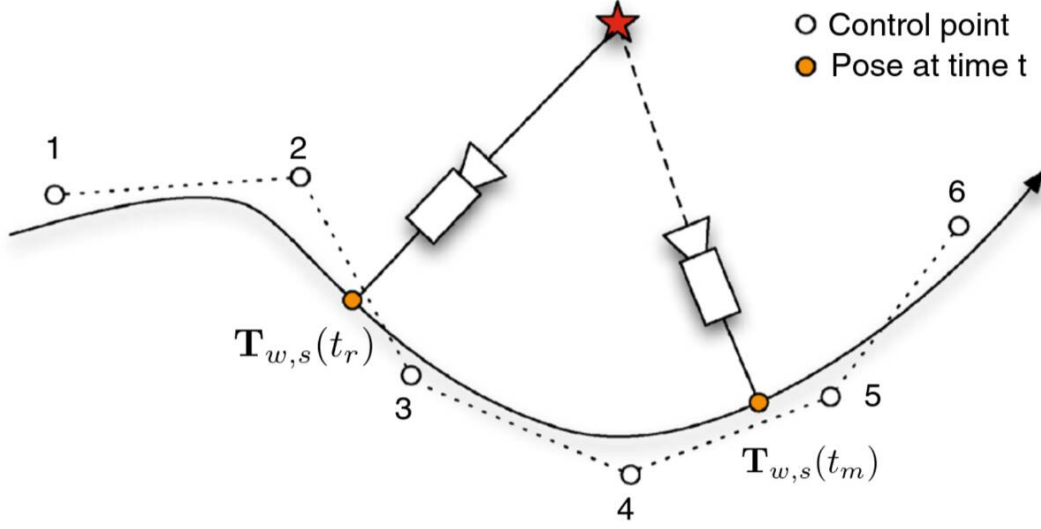


Figure 2.9: Continuous-time camera-IMU calibration, where it makes use of cumulative cubic B-splines. In the above we have a B-spline representing the reference pose (at time t_r) parameterised by control points 1-4 and the measurement pose (at time t_m) by control points 3-6. (Figure taken from [Patron-Perez et al., 2015])

More recently, [Mirzaei and Roumeliotis, 2008, Kelly and Sukhatme, 2009] formulated Kalman filter based approaches to jointly estimate the spatial and temporal relationship between camera and IMU *online*, from measurements in front of a camera calibration target. Both works studied the observability of the calibration problem, noting with sufficient rotation velocity within the calibration dataset would yield the calibration parameters are observable. The work of [Yang et al., 2019] in particular have identified four degenerate motions that harm camera IMU calibration, and prove that degeneracy holds even with global pose measurements provided. It is worth noting, however, these online methods assume the initial calibration parameters are sufficiently accurate. Furthermore, the methods assume subsequent sequential visual and inertial information are adequately informative such that the calibration parameters are observable, which cannot be guaranteed.

In contrast to online methods that use a Kalman filter formulation, offline methods typically perform a batch optimisation that jointly estimate the spatial and temporal calibration parameters over all visual and inertial measurements. The works of [Fleps et al., 2011, Furgale et al., 2012, Furgale et al., 2013, Patron-Perez et al., 2015] in particular

have used a continuous-time formulation and modeled the motion using a B-spline (see Figure. 2.9), enabling accurate time offsets between camera and IMU to be estimated. Offline approaches can afford to collect calibration data in a controlled environment, with dedicated calibration motion and are able to expend significantly more computation, often yielding more accurate results. As such it is in practice the preferred method in calibrating the spatial and temporal relationship between a camera and IMU.

Aerial Additive Manufacturing

Contents of Chapter

3.1	Introduction	49
3.2	Related Work	49
3.3	Contributions	52
3.4	The Aerial Additive Manufacturing Framework	53
3.4.1	Print Material	54
3.4.2	Flight Controls	55
3.4.3	Flight Trajectories	57
3.4.4	Print Scanning	59
3.4.5	MOCAP Marker to Camera Calibration	60
3.4.6	Estimating the Map Accuracy	62
3.5	Experiment	65
3.6	Conclusion	69
3.7	Future Work	70
3.7.1	Print Layer Compensation	70
3.7.2	Improving the ScanDrone Trajectory	72
3.7.3	Outdoor RTK-GPS Experiments	73

Parts of this chapter appeared in:

Ketao Zhang, Pisak Chermprayong, Feng Xiao, Dimos Tzoumanikas, Barrie Dams, Sebastian Kay, Basaran Bahadir Kocer, Alec Burns, Lachlan Orr, Talib Alhinai, Christopher Choi, Durgesh Dattatray Darekar, Wenbin Li, Steven Hirschmann, Valentina Soana, Shamsiah Awang Ngah, Clément Grillot, Sina Sareh, Ashutosh Choubey, Laura Margheri, Vijay M. Pawar, Richard J. Ball, Chris Williams, Paul Shepherd, Stefan Leutenegger, Robert Stuart-Smith and Mirko Kovac (2022). **Aerial additive manufacturing with multiple autonomous robots.** *Nature*. [Zhang et al., 2022].

3.1 Introduction

Additive manufacturing, also known as 3D printing, builds objects layer by layer, offering unprecedented design freedom and customization. This technology has a number of advantages, such as its ability in producing complex geometric structures, reduce material waste, whilst enabling shorter prototyping and production cycles.

Recently, additive manufacturing has found its way in construction, where there is a growing interest in using additive manufacturing techniques using static [Lim et al., 2012, Gosselin et al., 2016, Keating et al., 2017, Buswell et al., 2018] and mobile robots for on-site construction [Sustarevas et al., 2018, Zhang et al., 2018] and off-site prefabrication [Khoshnevis, 2004, Science et al., 2018].

In this chapter we present a scalable Aerial Additive Manufacturing (Aerial-AM) system designed to autonomously construct three-dimensional structures with human in the loop. To showcase the capabilities of the autonomous 3D printing system, and structural scanning throughout the process, we printed a proof-of-concept cylinder standing at 2.05 meters tall, comprised of 72 layers of rapid-curing insulation foam. This achievement not only highlights the system's proficiency in autonomous fabrication but also underscores its potential for large-scale and intricate construction projects.

3.2 Related Work

In recent years, there has been a growing interest in using robots to improve the productivity and safety in construction activities [Delgado et al., 2019, Tay et al., 2017], such as assembly of building elements [Lindsey et al., 2011, Reinhardt et al., 2016, Stuart-Smith, 2016, Furrer et al., 2017, Kayser et al., 2018] and free-form continuous additive manufacturing [Lim et al., 2012, Gosselin et al., 2016, Keating et al., 2017, Buswell et al., 2018]. The advantage free-form continuous additive manufacturing has over assembly based, is its flexibility in building geometrically variable designs, which gives rise to the opportunity in reducing material waste and cost. One of the early attempts in free form continuous additive manufacturing

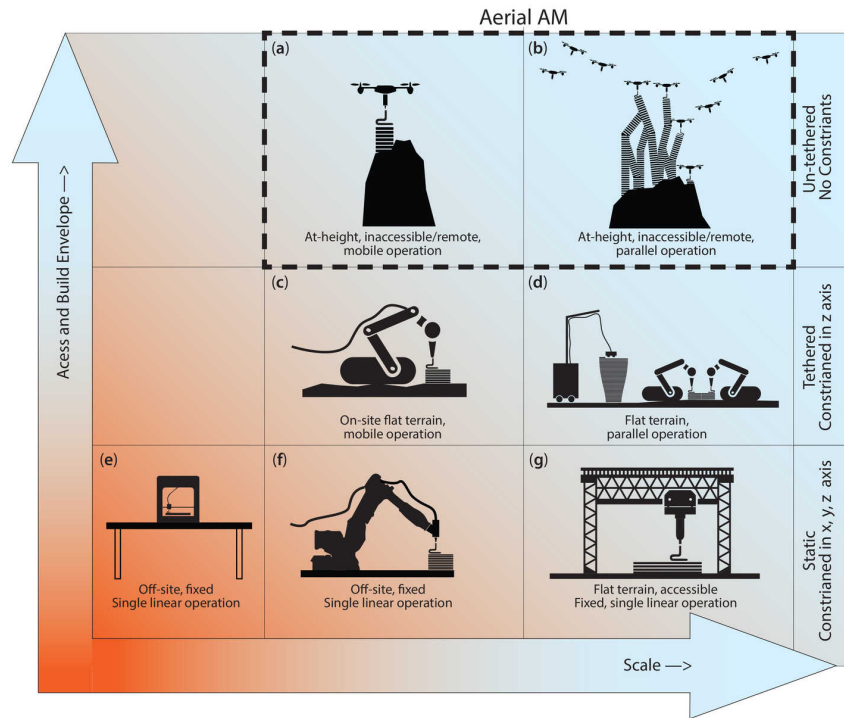


Figure 3.1: Comparison of different AM robot platforms, with red to blue gradient indicating improvements in scale, flexibility and access. Established platforms have limitations in the scale of the AM job versus the scale of the robot platform, the maximum build volume, the ability to manufacture in parallel and site-access capabilities. Aerial-AM (inside the dashed box) enables parallel manufacturing with an unbounded build envelope in hard-to-access locations

is contour crafting (CC), it is a large scale free form additive manufacturing technique for on-site construction, these systems primarily use ground-based robots and gantry-crane systems [Khoshnevis, 2004]. Using this system, a single or colony of houses, with varying designs, can be constructed autonomously in a single run. The main drawback of this approach, however, is the robot hardware needs to scale up to a larger dimension than the desired manufacturing volume, rendering parallel operation or occupation of the building site by people or other machinery difficult and dangerous. Further more, these large scale systems make inspection, maintenance, repair difficult. See Figure. 3.1 for a visual comparison between various Additive Manufacturing robots in terms of scale, access and build envelope.

In response to the limitations inherent in a single large-scale system, recent research has shifted its emphasis towards the creation of a collective of mobile robots capable of assembling structures significantly larger than the individual robot's size [Helm et al., 2012, Werfel et al., 2014, Willmann et al., 2012, Augugliaro et al., 2014]. Nevertheless, it is noteworthy that the research thus far have predominantly concentrated on the utilisation of mobile ground robots [Sustarevas et al., 2018, Zhang et al., 2018], which are constrained by a specific operational height limit. Additionally, the building geometry and materials used needs to be feasible for a ground mobile robot to operate. A comparison of state-of-the-art robot platforms developed for additive manufacturing in the building industry is illustrated in Figure. 3.1.

Natural builders, unlike current artificial robotic systems, display remarkable flexibility in constructing their habitats, often overcoming limitations with ingenious solutions. Social insects such as wasps engaging in aerial construction of its habitat for example, exhibit a high degree of adaptability and scalability, and addressed its restricted material payload by traveling back and forth thousands of times between its material source and construction site to build its habitat [Theraulaz et al., 1999]. This process demonstrates efficient and direct path optimization, as flight eliminates the need to navigate over or around previously constructed material during the building process.

Drawing inspiration from natural systems, we propose a novel framework for collective construction that leverages a network of untethered aerial mobile robots operating as a multi-agent system. However, scaling this approach to a significant number of robots necessitates addressing critical challenges in manufacturing and multi-agent coordination that surpass the capabilities of current technologies. In the following sections we detail our novel Aerial Additive Manufacturing framework, and how we tackled these challenges.

3.3 Contributions

In this chapter, we introduce the Aerial Additive Manufacturing (Aerial-AM) framework, which combines principles inspired by nature with engineering concepts to demonstrate an autonomous, untethered, three-dimensional aerial printing system. The system comprises of two drones with specialized roles: the BuilDrone, responsible for material deposition in additive manufacturing, and the ScanDrone, equipped with an RGB-D sensor to perform dense 3D reconstruction for inspection, verification, and layer-by-layer adjustment of the print.

While three experiments were conducted as part of this research, we focus specifically on the experiment involving the construction of a cylindrical geometry I contributed to. In the experiment, the BuilDrone deposits rapid-curing foam, with the ScanDrone actively monitoring and mapping the structure to ensure accuracy and inform subsequent layers. My contributions include:

- The trajectory planning for both the BuilDrone and ScanDrone for the construction of a cylindrical geometry.
- The scanning pipeline used to perform map reconstruction of the printed structure for inspection and validation.
- A Motion Capture (MOCAP) marker to camera calibration that resolves the relative transformation between MOCAP marker and camera sensor used on the ScanDrone.
- Estimating the map accuracy using first order error propagation.

3.4 The Aerial Additive Manufacturing Framework

Here we present the Aerial Additive Manufacturing (Aerial-AM) framework that coordinates multiple untethered aerial robotic platforms capable of performing unbounded three dimensional manufacturing. The Aerial-AM framework consists of a two different aerial robot platforms, referred to as the BuilDrone and ScanDrone (see Figure. 3.2). The BuilDrone is an aerial robot platform designed to accurately deposit physical materials according to a pre-planned flight path, while the ScanDrone is an aerial robot platform engineered with the purpose of performing incremental print scanning and validation after material deposition by the BuilDrone at every layer.

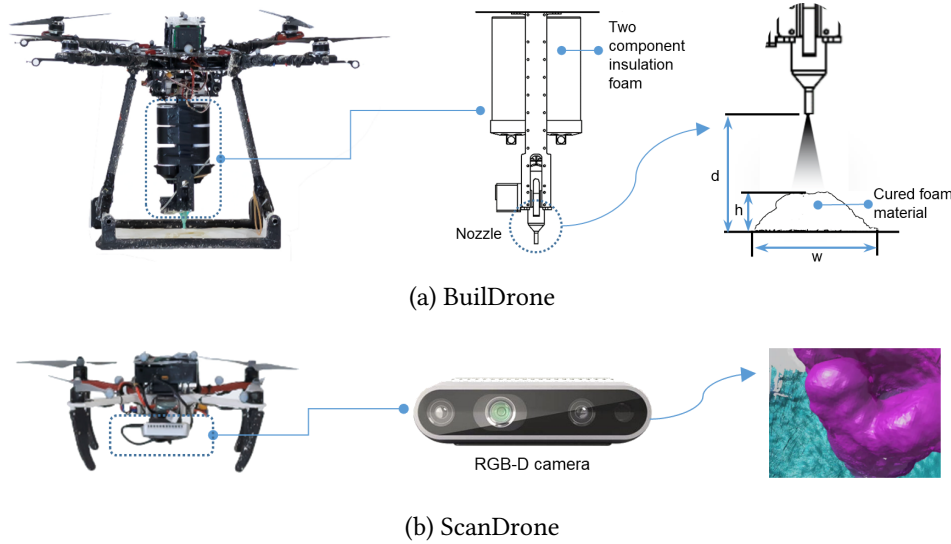


Figure 3.2: Drones used in Aerial Additive Manufacturing

The Aerial-AM framework is designed to be used indoors under a ground fixed motion capture system, where the operator communicates with the BuilDrone and ScanDrone from the communication station (see Figure. 3.3). Each aerial robot platform has its own designated base station to avoid colliding with each other. The pose information of the aerial robot platforms are obtained by the OptiTrack motion capture system, which gives the Aerial-AM framework millimeter accurate pose information of both the BuilDrone and ScanDrone at all times.

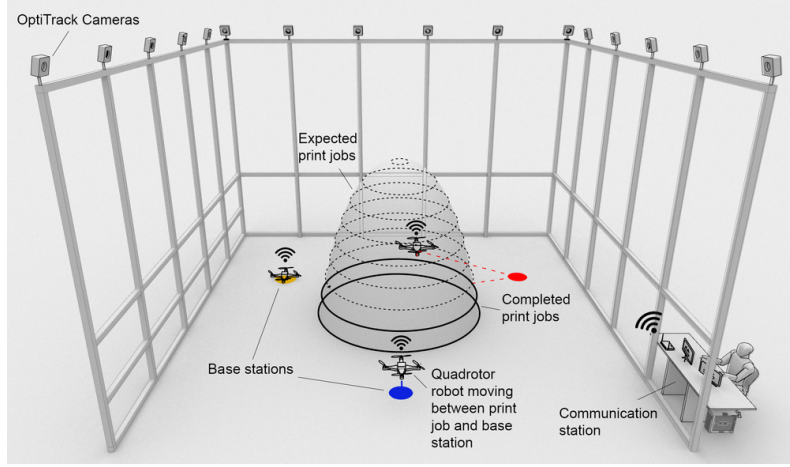


Figure 3.3: System Overview.

While state-estimation algorithms such as ORBSLAM3 [Campos et al., 2021], OKVIS [Leutenegger et al., 2014], VINS-MONO [Qin et al., 2018], and OpenVINS [Geneva et al., 2020] could theoretically be used for accurate pose estimation, implementing these methods was beyond the scope of this project and is left for future work.

The construction begins by printing the first layer of the cylinder structure with the BuildDrone, using the flight controls and trajectory is described in Sec. 3.4.2 and Sec. 3.4.3 respectively. After the BuildDrone completes depositing foam material for the first layer, the ScanDrone circumnavigates the perimeter of the cylinder structure detailed in Sec. 3.4.3, and collects RGB-D sensor data with the ScanDrone, and pose data from the ground-fixed motion tracking system. Upon landing the ScanDrone uses a state-of-the-art dense mapping algorithm to perform map reconstruction onboard the ScanDrone described in Sec. 3.4.4. The 3D mesh of the scanned print structure is inspected and validated by the operator on site, once validated the next print layer begins by sending the next print layer trajectory to the BuildDrone. This process is repeated until the desired structure is built.

3.4.1 Print Material

We have chosen to use a commercially available low density expanding foam. Specifically, we used a rapid-curing thermoplastic polyurethane foam, since this foam is commonly

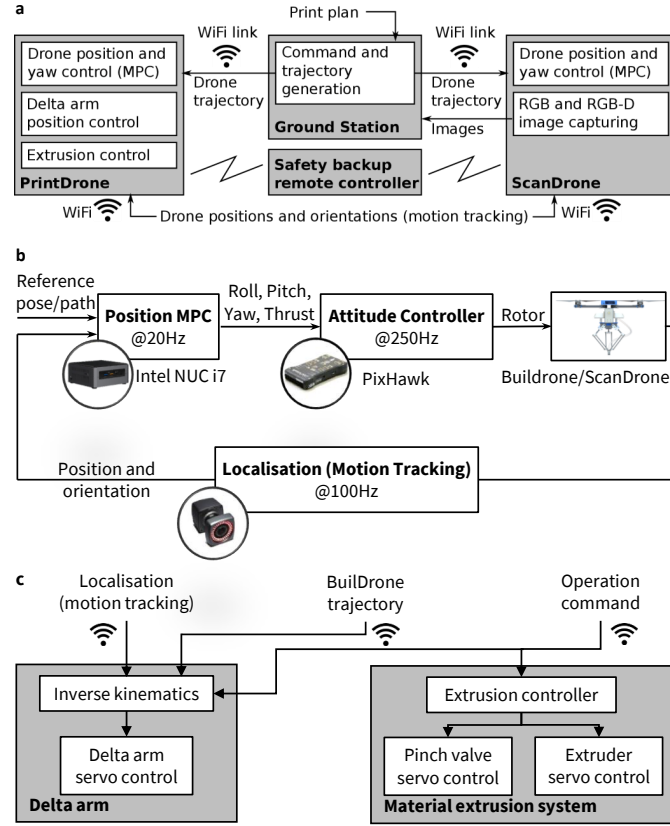


Figure 3.4: System Diagram.

used for building insulation and formwork for in situ cast-concrete structures [Keating et al., 2017]. Preliminary investigations have led us to choose a rapid-curing two-part foam system, with a density of 30kg/m^3 , for the BuildDrone extrusion. This was driven by the need to mitigate the effects of the down wash from the propellers of the BuildDrone causing the fresh foam to deform.

3.4.2 Flight Controls

To achieve fully autonomous flight, both the BuildDrone and ScanDrone were controlled with a classic cascaded architecture, where the inner attitude control loop was handled by the Pixhawk flight controller, and we used a finite-horizon Model-Predictive-Control (MPC) scheme to control the aerial robot platform’s position and orientation.

The finite-horizon Model-Predictive-Control (MPC) scheme used closely follows [Tzoumanikas et al., 2019], with the aim of controlling the state $\mathbf{x} = [{}_w\mathbf{r}_{WS}^\top, {}_w\mathbf{v}_{WS}^\top, \theta, \phi, \dot{\theta}, \dot{\phi}]^\top \in \mathbb{R}^{10}$ with control signal inputs $\mathbf{u} = [\theta, \phi, T, \psi] \in \mathbb{R}^4$, where ${}_w\mathbf{r}_{WS}$ and ${}_w\mathbf{v}_{WS}$ denote the position and velocity of the aerial robot platform in the world frame, θ, ϕ and ψ are roll, pitch and yaw angles respectively, and finally T is the collective motor thrust. The optimal control signal input sequence $\bar{\mathbf{u}}^* = \bar{\mathbf{u}}_0^* \dots \bar{\mathbf{u}}_{N-1}^*$ is found by minimizing the following optimisation problem:

$$\begin{aligned} \bar{\mathbf{u}}^* &= \underset{\bar{\mathbf{u}}_0^* \dots \bar{\mathbf{u}}_{N-1}^*}{\operatorname{argmin}} J, \\ \text{s.t. } \mathbf{x}_{k+1} &= \mathbf{A}\mathbf{x}_k + \mathbf{B}\mathbf{u}_k, \\ \mathbf{y}_k &= \mathbf{C}\mathbf{x}_k, \\ \mathbf{x}_0 &= \hat{\mathbf{x}}_0, \\ \bar{\mathbf{u}}_{\min} &\leq \bar{\mathbf{u}} \leq \bar{\mathbf{u}}_{\max} \end{aligned} \tag{3.1}$$

where $\mathbf{x}_k \in \mathbb{R}^{10}$ is the reference state at timestep k , $\hat{\mathbf{x}}_k \in \mathbb{R}^{10}$ is the estimated state at timestep 0, $\mathbf{y}_k \in \mathbb{R}^{10}$ is the system output at timestep k , and $\mathbf{A} \in \mathbb{R}^{10 \times 10}$, $\mathbf{B} \in \mathbb{R}^{10 \times 4}$, $\mathbf{C} \in \mathbb{R}^{10 \times 10}$ are the discrete states, input and output transition matrix respectively that define the aerial robot platform dynamics, found by performing system identification.

The cost function J for optimising the optimal control sequence $\bar{\mathbf{u}}^*$ is the quadratic penalty of states $\mathbf{s}_k^y \in \mathbb{R}^p$ and inputs $\mathbf{s}_k^u \in \mathbb{R}^m$ from their references, and N denotes the prediction finite horizon length:

$$J = \sum_{k=0}^{N-1} \left(\|\mathbf{Q}(\mathbf{y}_{k+1} - \mathbf{s}_{k+1}^y)\|_2^2 + \|\mathbf{R}(\mathbf{u}_k - \mathbf{s}_k^u)\|_2^2 \right), \tag{3.2}$$

where the penalty matrices $\mathbf{Q} \in \mathbb{R}^{4 \times 4}$ and $\mathbf{R} \in \mathbb{R}^{10 \times 10}$ are tuning parameters.

Using the developed model-predictive-control (MPC) schemes for Aerial-AM robots, both the BuilDrone and ScanDrone were characterised and tuned to perform sufficient accuracy for depositing rapid-curing foam materials and structural scanning respectively.

3.4.3 Flight Trajectories

To evaluate the Aerial-AM framework, the goal is to construct a single contour wall of a cylindrical geometry with a diameter of 0.3m at a height of 2.05m tall. Before deriving the BuilDrone and ScanDrone flight trajectory equations, we first derive the model used to predict the height to which the rapid-curing foam would expand to.

Print Model

Given that the BuilDrone can only control the foam mixture nozzle either on or off, the print layer height can only be influenced by how fast the BuilDrone circumnavigates the perimeter of the cylindrical geometry, given by

$$z_{\text{layer}} = \frac{1}{k_{\text{print}}} v_{\text{nominal}}, \quad (3.3)$$

where z_{layer} is the predicted height of the print layer, k_{print} is a constant that converts the velocity of the BuilDrone to next print layer height, and v_{nominal} is the nominal velocity for which the BuilDrone traverses the perimeter of the circle trajectory. Note that the above print model assumes the foam material deposition is constant over time, while this is a very rough approximation, in practice this assumption is sufficient for our use case.

Print Trajectory

Now that we have a print model from Eq. (3.3), we can proceed to describe the flight trajectory of the BuilDrone. To print a cylindrical geometric structure, the flight trajectory for the BuilDrone with a nominal velocity of $v_{\text{nominal}} = 0.08\text{ms}^{-1}$ is given by

$$\begin{aligned} x &= r_{\text{print}} \cos(2\pi t/T) + x_{\text{print}} \\ y &= r_{\text{print}} \sin(2\pi t/T) + y_{\text{print}} \\ z &= z_{\text{print}} + z_{\text{layer}} + z_{\text{nozzle}} + z_{\text{body}} \end{aligned} \quad (3.4)$$

where t and T are the current timestep and trajectory time in seconds, x and y are the x-y position of the BuilDrone in the world frame, parameterised by equations of a circle with a *given* center $(x_{\text{print}}, y_{\text{print}})$ and radius r_{print} of 0.15m, z is the altitude of the BuilDrone

consisting of the *measured* print structure height z_{print} , *predicted* next layer height z_{layer} from Eq. (3.3), z_{nozzle} the offset between the material deposit zone and nozzle of the BuildDrone, and z_{body} the offset between the nozzle and the body frame of the BuildDrone. Since the print mechanism is centered in the middle of the BuildDrone, the flight trajectory does not require the BuildDrone to yaw, and it is neglected for this reason.

Scan Trajectory

Once the BuildDrone completes a single print layer, the task of the ScanDrone is to scan, measure and inspect the printed layer. The ScanDrone is equipped with an Intel RealSense D435i RGB-D sensor, mounted downwards at a $\theta_{\text{camera}} = 35$ degree angle. The optimal RGB-D sensor to print structure distance d_{scan} for which the ScanDrone can safely and accurately scan the print structure with a nominal velocity of $v_{\text{nominal}} = 0.1\text{ms}^{-1}$ is approximately 0.5m. Given these constraints the flight trajectory of the ScanDrone is given as,

$$\begin{aligned}
 z_{\text{scan}} &= d_{\text{scan}} \sin(\theta_{\text{camera}}) + z_{\text{print}} \\
 r_{\text{scan}} &= d_{\text{scan}} \cos(\theta_{\text{camera}}) + r_{\text{print}} \\
 x &= r_{\text{scan}} \cos(2\pi t/T) + x_{\text{print}} \\
 y &= r_{\text{scan}} \sin(2\pi t/T) + y_{\text{print}} \\
 z &= z_{\text{scan}} \\
 \psi &= (2\pi t/T) - \pi
 \end{aligned} \tag{3.5}$$

where t and T are the current timestep and trajectory time in seconds, z_{scan} and r_{scan} are the ScanDrone altitude and circle radius offset respectively, x and y are the x-y position of the ScanDrone in the world frame, parameterised by equations of a circle with a *given* center $(x_{\text{print}}, y_{\text{print}})$ and radius r_{scan} , z is the altitude of the ScanDrone, and finally the yaw of the ScanDrone in the world frame is denoted as ψ , with the goal of pointing the RGB-D camera inwards towards the center of the cylinder geometric structure.

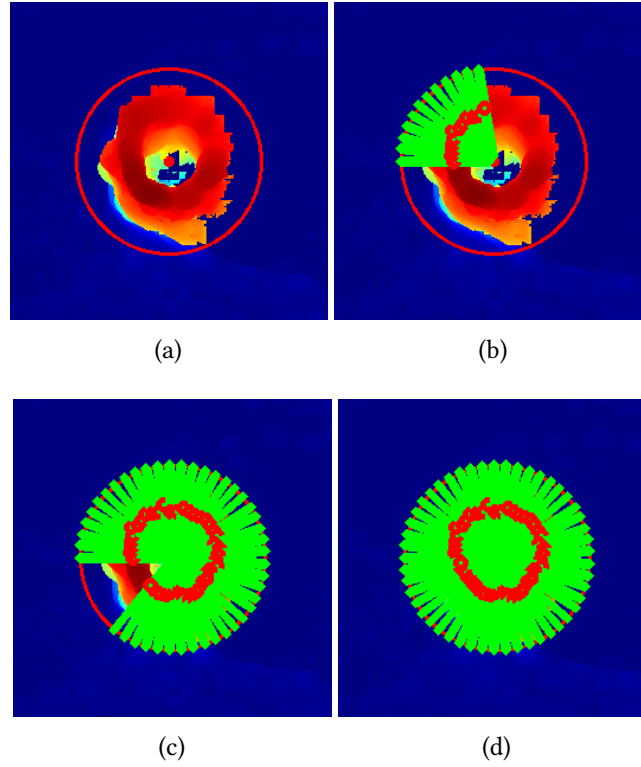


Figure 3.5: Height Profile Analysis: After the printed structure undergoes scanning by the ScanDrone, equipped with an RGB-D sensor, the map is reconstructed using SuperEight. To derive the print layer height, we employed Bresenham’s line algorithm [Bresenham, 1977] to sample the maximum height of the printed layer (indicated by the green line). This sampling extends from the center of the cylindrical print to a virtual circle boundary (highlighted in red). This process is iteratively repeated until a full circle is covered, progressing from (a) to (d). Subsequently, the maximum ridge height from each Bresenham line is extracted and utilised to adjust the height of the subsequent print layer.

3.4.4 Print Scanning

After the BuildDrone successfully completes a print layer, it is the task of the ScanDrone to scan, measure and inspect the printed layer. During scanning, depth images and the poses of the ScanDrone from the motion tracking system are recorded. These measurements are then in turn used for map reconstruction on-board the Scan-Drone immediately after flight, using an approach based on adaptive-resolution volumetric Truncated Signed Distance Field (TSDF) [Vespa et al., 2019]. With a map reconstruction, we can obtain the print height of the print structure to plan for the next print layer. To extract the print height,

we first render a height map using orthonormal raycasting from above along the known vertical direction into the TSDF volume. This is then followed by a simple segmentation of the printed structure and extraction of the ridge line. See Figure. 3.5 for the height profile analysis pipeline. The subsequent BuildDrone print trajectories are then adjusted to compensate for the irregularities of the previous print layer.

3.4.5 MOCAP Marker to Camera Calibration

For accurate map reconstruction for Aerial-AM, it is important to resolve the relative transformation between the motion capture markers on the ScanDrone, and the Intel RealSense D435i RGB-D sensor (see Figure. 3.6). In the following we present our MOCAP-marker camera calibration method developed to resolve this transformation.

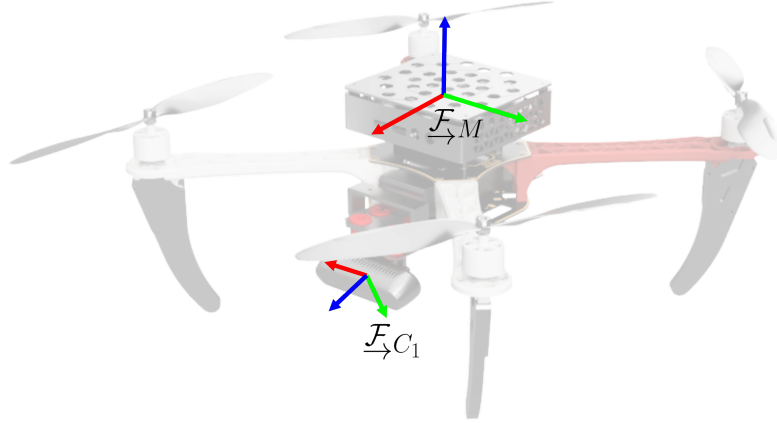


Figure 3.6: Frame diagram highlighting the MOCAP marker frame $\vec{F}_M$ and camera frame $\vec{F}_{C_1}$ on the ScanDrone.

States

For the MOCAP-marker to camera calibration problem, the states to be estimated consist of the marker poses in the world frame, $\mathbf{x}_{WM}$, fiducial target pose in the world frame, $\mathbf{x}_{WF}$, camera 1 extrinsic relative to MOCAP marker, $\mathbf{x}_{MC_1}$, and camera 1 intrinsics, $\mathbf{x}_{C_1}$, of the

form:

$$\begin{aligned}\mathbf{x}_{WM} &= \begin{bmatrix} {}_W\mathbf{r}_{WM}^\top & \mathbf{q}_{WM}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \\ \mathbf{x}_{WF} &= \begin{bmatrix} {}_W\mathbf{r}_{WF}^\top & \mathbf{q}_{WF}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \\ \mathbf{x}_{MC_1} &= \begin{bmatrix} {}_M\mathbf{r}_{MC_1}^\top & \mathbf{q}_{MC_1}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \\ \mathbf{x}_{C_1} &= \begin{bmatrix} f_x & f_y & c_x & c_y & k_1 & k_2 & p_1 & p_2 \end{bmatrix}^\top \in \mathbb{R}^8,\end{aligned}\tag{3.6}$$

where $\underline{\mathcal{F}}_{C_1}$ denotes the coordinate frame of the camera 1 on the sensor assembly. We used the Radial-Tangential camera model consisting of focal lengths f_u, f_v , principal centre c_u, c_v , radial distortion parameters k_1, k_2 , and tangential distortion parameters p_1, p_2 as the camera intrinsics. Note that any other projection model could be supported in principle. The full state vector for camera calibration thus becomes,

$$\mathbf{x} = \left[\mathbf{x}_{WM}^{\top,1} \dots \mathbf{x}_{WM}^{\top,k} \quad \mathbf{x}_{WF}^\top \quad \mathbf{x}_{MC_1}^\top \quad \mathbf{x}_{C_1}^\top \right]^\top.\tag{3.7}$$

The minimal error state vectors for $\mathbf{x}_{WM}$, $\mathbf{x}_{WF}$, $\mathbf{x}_{MC_1}$, and $\mathbf{x}_{C_1}$ are:

$$\delta\mathbf{x}_{WM} = \begin{bmatrix} \delta {}_W\mathbf{r}_{WM}^\top & \delta \boldsymbol{\alpha}_{WM}^\top \end{bmatrix}^\top \in \mathbb{R}^6,\tag{3.8}$$

$$\delta\mathbf{x}_{WF} = \begin{bmatrix} \delta {}_W\mathbf{r}_{WF}^\top & \delta \boldsymbol{\alpha}_{WF}^\top \end{bmatrix}^\top \in \mathbb{R}^6,\tag{3.9}$$

$$\delta\mathbf{x}_{MC_1} = \begin{bmatrix} \delta {}_M\mathbf{r}_{MC_1}^\top & \delta \boldsymbol{\alpha}_{MC_1}^\top \end{bmatrix}^\top \in \mathbb{R}^6,$$

$$\delta\mathbf{x}_{C_1} = \begin{bmatrix} \delta f_x & \delta f_y & \delta c_x & \delta c_y & \delta k_1 & \delta k_2 & \delta p_1 & \delta p_2 \end{bmatrix}^\top \in \mathbb{R}^8.$$

Calibration Formulation

To estimate the camera calibration parameters we used a nonlinear least squares framework to minimise the cost function, J , containing reprojection errors, $\mathbf{e}_r$, and the information matrix of the respective camera measurement, $\mathbf{W}$. The cost function has the form:

$$J(\mathbf{x}) = \frac{1}{2} \sum_{k=1}^K \sum_{i \in \mathcal{J}(k)} \mathbf{e}_r^{i,k\top} \mathbf{W}^{i,k} \mathbf{e}_r^{i,k},\tag{3.10}$$

where k timestep at which the image was taken, and i denotes the fiducial target corner index. Finally, $\mathcal{J}(k)$ denotes the set of observed fiducial corner indices in the k^{th} camera frame.

Here, the standard reprojection error, $\mathbf{e}_r$, was used:

$$\mathbf{e}_r^{i,k} = \tilde{\mathbf{z}}^{i,k} - \pi(\mathbf{T}_{MC_1}^{-1} \mathbf{T}_{WM}^{-1} \mathbf{T}_{WF} \mathbf{F}_{FF_i}, \mathbf{x}_{C_i}), \quad (3.11)$$

whereby $\pi(\cdot)$ denotes the camera projection and distortion model. It needs as an input the fiducial corner, $\mathbf{F}_{FF_i}$, marker pose, $\mathbf{T}_{WM}$, fiducial pose, $\mathbf{T}_{WF}$, MOCAP marker to camera extrinsic, $\mathbf{T}_{MC_1}$, and camera intrinsics $\mathbf{x}_{C_1}$. Lastly, $\tilde{\mathbf{z}}^{i,k}$ is the observed fiducial corner measurement by the camera $\mathbf{F}_{C_1}$, at k^{th} camera frame, of the i^{th} fiducial corner.

3.4.6 Estimating the Map Accuracy

In the following section we estimate the mapping accuracy of the ScanDrone in the Aerial Additive-Manufacturing (Aerial-AM) framework. Specifically, what impact calibration and MOCAP uncertainty has on the estimated mapping accuracy when used with SuperEight.

The accuracy of the scan procedure can be estimated using first order error propagation of uncertainties:

$$\Sigma_y = \mathbf{J}_x \Sigma_x \mathbf{J}_x^T, \quad (3.12)$$

where $\mathbf{J}_x$ is the measurement model Jacobian with respect to model input parameter vector $\mathbf{x}$, Σ_x is the covariance matrix representing the uncertainties of $\mathbf{x}$, and Σ_y is the model output covariance matrix.

The following measurement model was used to estimate the accuracy of the scanned geometry

$${}_W\mathbf{r}_{WP_i} = \mathbf{T}_{WM} \mathbf{T}_{MC} {}_C\mathbf{r}_{CP_i}. \quad (3.13)$$

This model transforms the i -th point P_i observed in the camera frame $\mathbf{F}_C$, denoted as ${}_C\mathbf{r}_{CP_i}$, to the world frame $\mathbf{F}_W$ through a series of homogeneous rigid body transforms as ${}_W\mathbf{r}_{WP_i}$. The rigid body transformations $\mathbf{T}_{MC}$ and $\mathbf{T}_{WM}$, represent the camera $\mathbf{F}_C$ to OptiTrack marker $\mathbf{F}_M$ transform, and the OptiTrack marker $\mathbf{F}_M$ to world $\mathbf{F}_W$ transform respectively.

Lets suppose the two transforms $\mathbf{T}_{MC}$ and $\mathbf{T}_{WM}$, and (measured) point ${}_C\mathbf{r}_{CP_i}$ are affected

by uncertainty, and define a state vector $\mathbf{x}$

$$\mathbf{x} = [\mathbf{x}_{WM}^\top \quad \mathbf{x}_{MC}^\top \quad {}_C\mathbf{r}_{CP_i}^\top]^\top, \quad (3.14)$$

where $\mathbf{x}_{AB}$ represents the poses relating frames $\underline{\mathcal{F}}_B$ and $\underline{\mathcal{F}}_A$, are parameterised as a concatenation of position vector ${}_A\mathbf{r}_{AB}$ and quaternion of orientation $\mathbf{q}_{AB}$ that correspond to the respective homogeneous transform. We further introduce the minimal local parameterisations of the poses:

$$\delta\mathbf{x}_{AB} = \begin{bmatrix} \delta {}_A\mathbf{r}_{AB} \\ \delta\boldsymbol{\alpha}_{AB} \end{bmatrix}, \quad (3.15)$$

that are defined as perturbations around the linearisation point $\bar{\mathbf{x}}_{AB}$:

$$\mathbf{x}_{AB} = \begin{bmatrix} \delta {}_A\mathbf{r}_{AB} + \bar{\mathbf{r}}_{AB} \\ \left[\frac{1}{2}\delta\boldsymbol{\alpha}_{AB} \right] \otimes \bar{\mathbf{q}}_{AB} \\ 1 \end{bmatrix}, \quad (3.16)$$

where $\otimes$ denotes the quaternion multiplication.

The measurement Jacobians with respect to the minimal parameters $\mathbf{x}$ can now be found as

$$\mathbf{J}_{\delta\mathbf{x}} = \begin{bmatrix} \frac{\partial {}_W\mathbf{r}_{WP_i}}{\partial \delta\mathbf{x}_{WM}} & \frac{\partial {}_W\mathbf{r}_{WP_i}}{\partial \delta\mathbf{x}_{MC}} & \frac{\partial {}_W\mathbf{r}_{WP_i}}{\partial {}_C\mathbf{r}_{CP_i}} \end{bmatrix} \in \mathbb{R}^{2 \times 15}, \quad (3.17)$$

with

$$\frac{\partial {}_W\mathbf{r}_{WP_i}}{\partial \delta\mathbf{x}_{WM}} = [\mathbf{I}_{3 \times 3} \quad -({}_C\mathbf{r}_{MP_i})^\times], \quad (3.18)$$

$$\frac{\partial {}_W\mathbf{r}_{WP_i}}{\partial \delta\mathbf{x}_{MC}} = [{}_C\mathbf{r}_{WM} \quad -{}_C\mathbf{r}_{WM} ({}_C\mathbf{r}_{CP_i})^\times], \quad (3.19)$$

$$\frac{\partial {}_W\mathbf{r}_{WP_i}}{\partial {}_C\mathbf{r}_{CP_i}} = {}_C\mathbf{r}_{WC}, \quad (3.20)$$

where $(\cdot)^\times$ denotes the skew-symmetric cross-product matrix associated with a vector and $\mathbf{C}_{AB}$ are rotation matrices that change the coordinate representation of a vector from frame $\underline{\mathcal{F}}_B$ to $\underline{\mathcal{F}}_A$. The parameter covariance matrix $\Sigma_{\mathbf{x}}$ has the following form:

$$\Sigma_{\mathbf{x}} = \text{diag} \left(\begin{bmatrix} \boldsymbol{\sigma}_{\mathbf{x}_{WM}}^{2\top} & \boldsymbol{\sigma}_{\mathbf{x}_{MC}}^{2\top} & \boldsymbol{\sigma}_{\mathbf{r}_{CP_i}}^{2\top} \end{bmatrix}^\top \right), \quad (3.21)$$

where we used the following numeric variances in $[m]^2$ and $[rad]^2$, respectively, in accordance with the OptiTrack manufacturer ($\sigma_{\mathbf{x}_{WM}}^2$) deviation of $3mm$ and 0.06° , calibration accuracy ($\sigma_{\mathbf{x}_{MC}}^2$) which was obtained by estimating the covariance matrix from Section. 3.4.5, and the depth camera accuracy ($\sigma_{\mathbf{r}_{CP_i}}^2$) in accordance to the specifications of the Intel RealSense D435i sensor as tested:

$$\begin{aligned}\sigma_{\mathbf{x}_{WM}}^2 &= [1 \times 10^{-5}, 1 \times 10^{-5}, 1 \times 10^{-5}, 1 \times 10^{-6}, 1 \times 10^{-6}, 1 \times 10^{-6}]^\top, \\ \sigma_{\mathbf{x}_{MC}}^2 &= [1 \times 10^{-6}, 1 \times 10^{-6}, 1 \times 10^{-6}, 6.2 \times 10^{-6}, 6.2 \times 10^{-6}, 6.2 \times 10^{-6}]^\top, \\ \sigma_{\mathbf{r}_{CP_i}}^2 &= [1.6 \times 10^{-6}, 1.6 \times 10^{-6}, 1 \times 10^{-4}]^\top.\end{aligned}\tag{3.22}$$

Furthermore, the following transformations were used. The chosen drone pose $\mathbf{T}_{WM} = \mathbf{I}_{4 \times 4}$ is arbitrary and does not affect the result. As for the camera pose, we calibrated it as

$$\mathbf{T}_{MC} = \begin{bmatrix} -0.01464 & -0.90070 & 0.43267 & 0.10401 \\ -0.99913 & 0.01955 & 0.00689 & 0.01930 \\ -0.01468 & -0.43248 & -0.90080 & -0.10497 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

and for the observed point in camera coordinates we assume the nominal (ideal) scanning distance in the middle of the image

$${}^C\mathbf{r}_{CP_i} = \begin{bmatrix} 0.0 & 0.0 & 0.5 \end{bmatrix}^T \text{ m.}$$

With this, we can calculate the output covariance matrix Σ_y using Eq. 3.12 for the marginal standard deviations along each axis x, y , and z of the world frame, and we obtain $\pm[4.6 \quad 1.9 \quad 9.1] \text{ mm}$, i.e. in the order of 1.0 cm .

Note, however, that there are several simplifications in this analysis. First, we assumed (constant) Gaussian noise on the 3D point measurement, which is clearly severely violated especially at oblique angles and at occlusion boundaries. Second, we used linear propagation in an inherently nonlinear process. Third, we only analysed a single point and did not account for the fusion process across views and thus changing poses. Finally, the model also does not consider the fact that entire map areas may be left occluded after the scanning process. Therefore, this modelled error estimate is likely overly optimistic.

3.5 Experiment

To evaluate the Aerial-AM framework described in Sec. 3.4, we conducted the experiment in an indoor ground fixed motion capture system environment that consisted of 20 Optitrack Prime 17 W cameras, connected to a Windows base station PC via two Netgear ProSafe GS728TPP ethernet switches. The Windows base station ran Motive, Optitrack’s tracking software, this computer had an Intel Xeon CPU E5-2697 v3 2.6Ghz processor with 32 GB of RAM. In order to obtain the pose information of the aerial robotic platforms from the Motive tracking software from the Windows base station, another PC running Ubuntu 16.04 and ROS kinetic was connected to the Windows base station via an ethernet cable, this computer equipped with the same CPU and RAM as the Windows base station, obtains the tracking information and transmits the pose information of each aerial robot platform using ROS messages. The ROS computer transmits pose information to each aerial robot platform’s onboard Intel NUC over wifi. The data from each robot was tracked and logged on the ROS computer, and later used to evaluate the performance of the system.

Before conducting the experiment, the motion tracking system was calibrated using the manufacturer’s Motive motion capture software. Additionally, the RGB-D sensor on the ScanDrone, and rigid body transformation between the motion tracking markers placed on the ScanDrone and the RGB-D sensor were calibrated using the calibration formulation detailed in Sec. 3.4.5, resulting in an RMSE reprojection error of approximately 1.5 pixels, which corresponds to angular offsets in the order of 0.1 degrees, which we deem acceptable.

Using the experimental setup described above, We printed a single contour wall of a cylindrical geometry with a diameter of 0.3m at a height of 2.05m tall (see Figure. 3.8 and Figure. 3.9). The experiment started with the BuilDrone depositing a layer of rapid-curing foam following a pre-planned flight trajectory, followed by a ScanDrone flying a similar trajectory circumnavigating the perimeter of the printed structure whilst collecting scan data of the printed structure using the onboard RGB-D sensor, as well as pose data from the ground-fixed motion tracking software. As soon as the ScanDrone completed scanning the printed structure, a state-of-the-art dense mapping software, SuperEight [Vespa et al.,

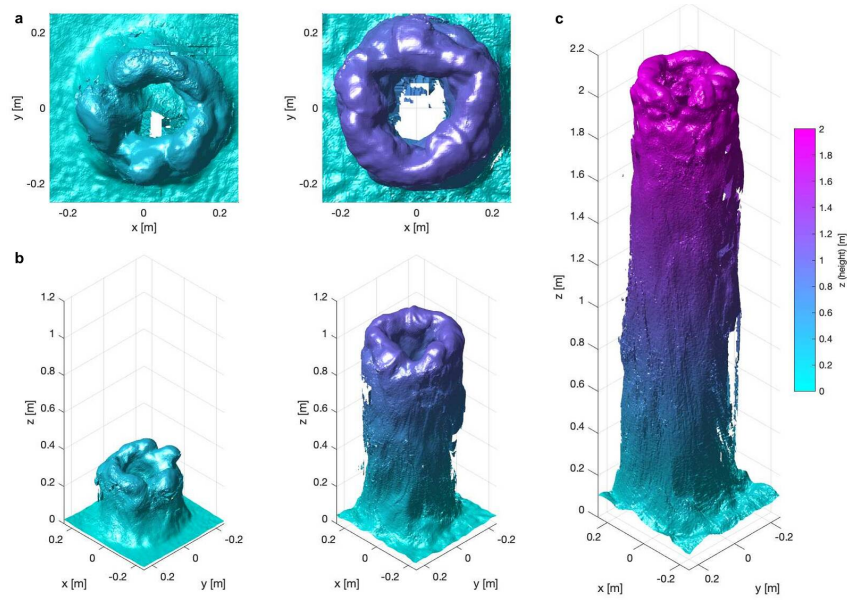


Figure 3.7: Mesh reconstructions from the ScanDrone, including top-view heat map (top) used for height adjustment at layers 10 (left) and 36 (half height; middle) and perspective side views (bottom) at layers 10 (left), 36 (middle) and 72 (full height; right) of the foam cylinder.

2018, Vespa et al., 2019], was used to integrate and visualise the print as a 3D mesh for inspection. Figure. 3.7, shows the close-up scan views of selected layers. Besides in-the-loop qualitative and quantitative analyses of the built geometry, the map crucially enables adjustment of the print trajectory height of the next layer. This process was repeated until the desired height was achieved. The result was a cylinder structure consisting of 72 layers of foam material (see Figure. 3.8), and the overall effective material deposition by BuilDrone took approximately 29 min to complete. Figure. 3.8, shows the result of the printed structure along side with the BuilDrone and ScanDrone for scale.

The 3D mesh of the printed structure created by the ScanDrone was compared against a point cloud measured by a 3D FARO laser scan. The FARO scanners have an advertised measurement error $\pm 1\text{mm}$ at a distance of 10m, which are among the most accurate laser scanning devices available commercially, suitable for applications requiring fine detail and high accuracy across large areas. The mesh and point cloud from both were aligned manually and refined by the Iterative Closest Point (ICP) algorithm using the CloudCompare



(a)



(b)

Figure 3.8: Printed cylinder structure alongside the BuildDrone and ScanDrone.



Figure 3.9: Printed cylinder structure.

tool [Clo, 2023]. The analysis of point-to-triangle errors reveals a median value of 2.27cm, which suffices the required accuracy in foam printing.

Both the planned and actual flight trajectory of the BuilDrone were logged during the experiment. The absolute position errors were quantitatively evaluated, showing that the maximum horizontal and vertical absolute position errors were within 1.4cm and 0.6cm, respectively.

3.6 Conclusion

In this chapter we have presented the Aerial-AM framework as an autonomous, scalable and flexible approach to additive manufacturing. We have demonstrated the framework through printing a foam cylinder structure of 2.05m tall with a diameter of 0.3m, using both the BuilDrone for foam depositing and ScanDrone in-the-loop quantitative assessment of the printed structure.

The above experiment showcases the feasibility of the Aerial-AM framework, these are just the first steps towards aerial additive manufacturing. Significant improvements in material science could enable full-scale manufacturing of building geometries using the proposed approach economically viable. Further, to take the proposed framework outdoors, we intend to implement a multi-sensor simultaneous localization and mapping (SLAM) algorithm with differential global positioning system for accurate robot state estimation outdoors. To scale up manufacturing, automatic battery replenishment for efficient multi-agent aerial additive manufacturing is also another area that is left as future work.

Although the system presented in this chapter is a proof of concept for autonomous Aerial-AM, it forms a foundation for future improvements. With continued developments, Aerial-AM can provide an alternative approach to additive manufacturing of buildings and structures.

3.7 Future Work

The following enhancements to the Aerial-AM framework, though not included in the original Nature publication, offer valuable improvements that extend the framework's capabilities.

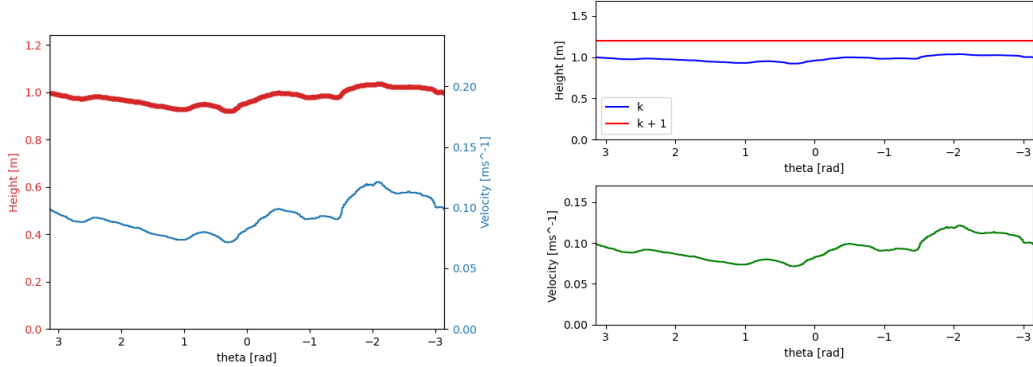
3.7.1 Print Layer Compensation

The Aerial-AM framework discussed in the previous sections does not incorporate a feedback loop to correct for irregularities in the previous print layer. To address height discrepancies from the preceding layer, the BuilDrone's velocity can be dynamically adjusted: reducing speed in areas that fall short of the desired height to allow for additional foam deposition, while increasing speed in sections that exceed the target height. The following equations define the adjustments to the BuilDrone's velocity for maintaining consistent layer heights in subsequent print layers:

$$\begin{aligned} \hat{z}_{\text{print}}^{k+1} &= (\hat{z}_{\text{print}}^k - z_{\text{print}}^k) + z_{\text{layer}}, \\ v &= \frac{1}{k_{\text{print}}} \hat{z}_{\text{print}}^{k+1}, \end{aligned} \quad (3.23)$$

where $\hat{z}_{\text{print}}^{k+1}$ is the next desired print structure height, $\hat{z}_{\text{print}}^k$ and z_{print}^k are the previous *desired* and *measured* print structure height respectively, z_{layer} is the nominal print layer height, k_{print} is a constant that converts the velocity of the BuilDrone to next print layer height, and v is the desired velocity for the next print layer to which the BuilDrone circumnavigates the cylindrical geometric structure. Figure. 3.10 demonstrates a hypothetical scenario where some sections of the previous print layer is below the desired height, and some sections where it is above. Using the equations defined above, we can define a dynamic velocity profile for the BuilDrone to follow and compensate for the irregularities in height.

To correct discrepancies in the horizontal X-Y plane, where the printed cylindrical structure's center deviates from the target center, we have developed a method to measure the current printed center and adjust the subsequent layer's trajectory gradually toward the desired center (see Figure. 3.11). First, a scan of the printed cylindrical structure was



(a) Previous print height (red) and next print layer BuilDrone velocity (blue) around the circumference of the cylindrical structure.

(b) Simulation of an ideal BuilDrone print if it were to compensate for the previous layer's irregularities in height.

Figure 3.10: Print Layer Height Compensation

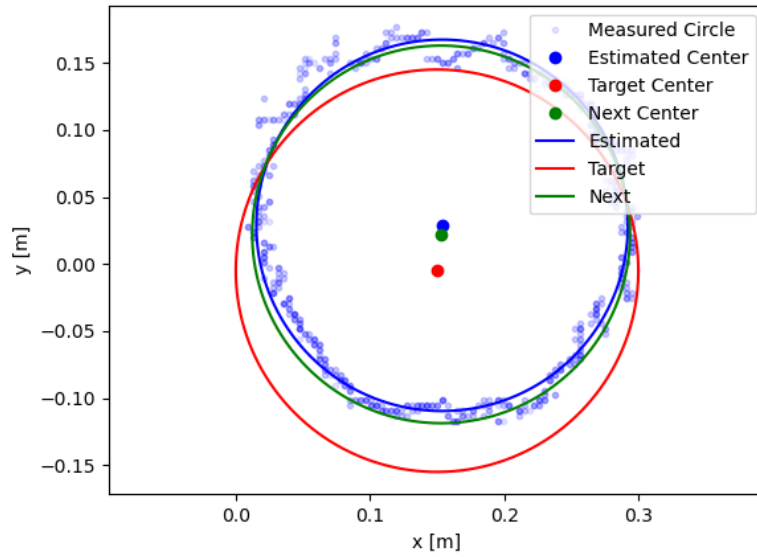


Figure 3.11: Print Layer X-Y Plane Compensation

conducted as detailed in Section. 3.4.4. Using the height profile extracted from this scan (similar to Figure. 3.5), we apply a least-squares method to estimate the center of the circle based on the height measurements. Given the desired and measured cylindrical centers and radii as $(\hat{c}_x, \hat{c}_y, \hat{r})$ and (c_x, c_y, r) , respectively, and the proportional gain constants k_x for the x-axis, k_y for the y-axis, and k_r for the radius, we can derive a new print trajectory for

the next layer using the following equations:

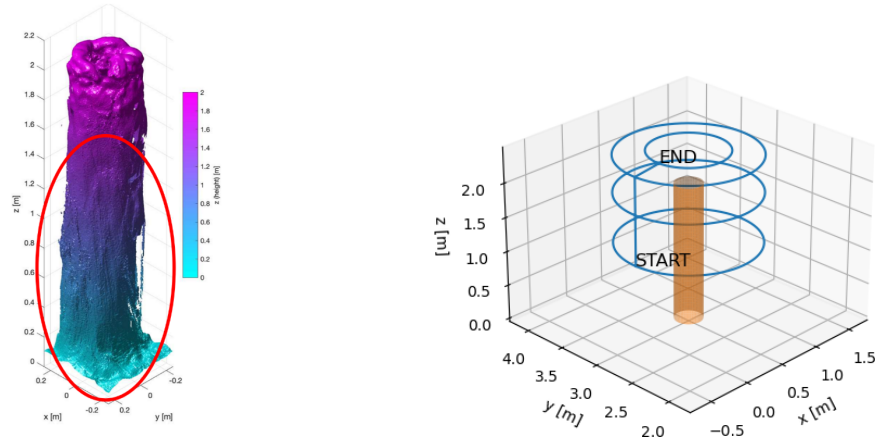
$$c_x = k_x \cdot (\hat{c}_x - c_x) \quad (3.24)$$

$$c_y = k_y \cdot (\hat{c}_y - c_y) \quad (3.25)$$

$$r = k_r \cdot (\hat{r} - r) \quad (3.26)$$

Figure. 3.11 illustrates a scenario in which the measured center of the printed cylindrical structure (blue) does not align with the desired target center (red). By applying the above equations with parameters $k_x = k_y = k_r = 0.2$ under ideal conditions, the next print layer (green) is projected to shift towards the target center (red), aligning the structure more closely with the intended positioning.

3.7.2 Improving the ScanDrone Trajectory



(a) Scanned structure lacks details on the sides.

(b) Improved ScanDrone trajectory.

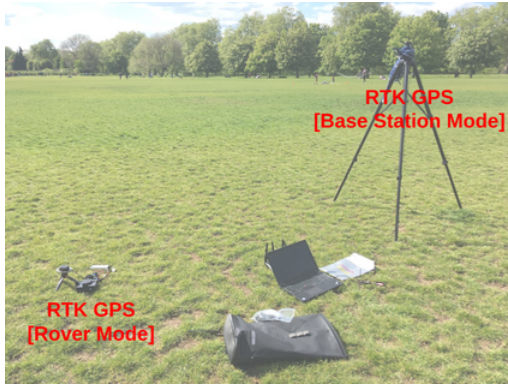
Figure 3.12: Improving the ScanDrone trajectory to capture more details on the side of the printed structure.

The primary objective of the ScanDrone trajectory discussed in Section. 3.4.4 is to capture print height data for in-loop assessments, enabling adaptive height adjustments for subsequent print layers. However, the ScanDrone trajectory performed does not effectively capture detailed information from the sides of the printed structure. To resolve this, the scan path can be modified to include a more comprehensive sweep of the structure's sides (See Figure. 3.12).

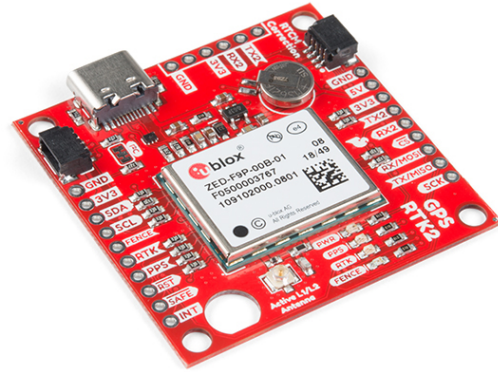
3.7.3 Outdoor RTK-GPS Experiments

The experiments described in previous sections were conducted indoors using a motion capture system to achieve millimeter-level accuracy in tracking the pose of both the BuilDrone and the ScanDrone. However, this approach presents two primary limitations: first, it restricts the system to environments where a motion capture setup larger than the build volume is available. Second, it confines Aerial-AM operations to indoor settings, limiting the versatility and broader applicability of the system in outdoor environments.

To enable outdoor Aerial-AM without relying on motion capture, we considered RTK-GPS as a potential precise positioning solution. RTK-GPS or Real-Time Kinematic GPS is a high-precision satellite navigation system that enhances standard GPS accuracy by using a fixed base station and a moving receiver (rover) on the BuilDrone and ScanDrone. The base station calculates and broadcasts real-time correction data to the rover, correcting for errors like satellite timing and atmospheric delays. This approach enables the rover to determine its position relative to the base with centimeter-level accuracy, making RTK-GPS highly valuable for applications requiring precise location data.



(a) Outdoor Experiments



(b) SparkFun UBlox ZED-F9P Breakout Board

Figure 3.13: RTK GPS Outdoor Experiment

We conducted a series of outdoor RTK-GPS experiments using a UBlox ZED-F9P RTK GPS from SparkFun (see Figure. 3.13) to assess the accuracy of RTK-GPS and its feasibility for outdoor Aerial-AM applications. Without access to a ground truth reference, we

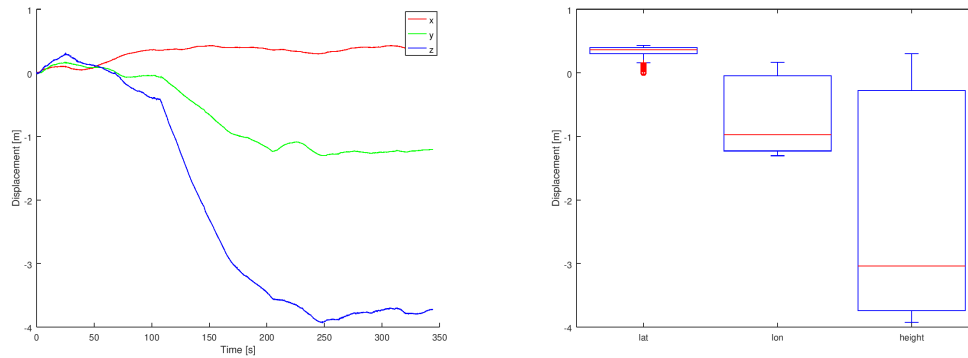


Figure 3.14: Static Test - GPS Mode

performed static tests where both the rover and base station remained stationary. Initially, we collected a 5-minute dataset in standard GPS mode. As shown in Figure. 3.14, despite the rover being stationary, the GPS-only measurements exhibited substantial errors in the x and y axes, with position drift on the order of a meter. Height measurements were even less consistent, fluctuating from $0m$ at the start to approximately $-4m$ by the end of the session. In contrast, when operating in RTK-GPS mode, the position measurements were notably more stable and precise once RTK lock was established, as illustrated in Figure. 3.15. Across all three axes, the RTK-GPS measurement error remained around $\pm 1cm$, consistent with the manufacturer’s accuracy specifications.

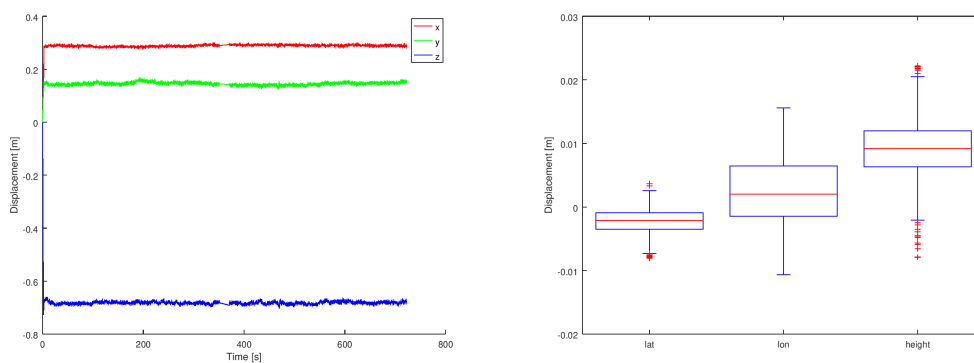


Figure 3.15: Static Test - RTK-GPS Mode

The stationary RTK-GPS experiments discussed above demonstrate that centimeter-

level positional accuracy is achievable; however, additional experiments are required to validate this accuracy during motion. Although these initial results are promising, we ultimately opted to simplify the setup by conducting indoor experiments with a motion capture system that provides millimeter-scale accuracy. Transitioning to alternative position estimation methods for the BuilDrone and ScanDrone remains as future work.

Accurate And Interactive Visual-Inertial Calibration with Next-Best-View and Next-Best-Trajectory

Contents of Chapter

4.1	Introduction	79
4.2	Related Work	81
4.3	Background	82
4.4	System Overview	84
4.5	Camera Intrinsics and Extrinsics Calibration	85
4.5.1	States	85
4.5.2	Calibration Formulation	86
4.5.3	Real-time Estimation	86
4.5.4	Camera Calibration With Next-Best-View	87
4.6	Camera-IMU Extrinsics Calibration	88

4. *Accurate And Interactive Visual-Inertial Calibration with Next-Best-View and Next-Best-Trajectory*

4.6.1	States	89
4.6.2	Calibration Formulation	90
4.6.3	Real-time Estimation	92
4.6.4	Next-Best-Trajectories	92
4.6.5	Camera-IMU Calibration With Next-Best-Trajectory	93
4.7	Experiments	95
4.7.1	Calibration Results on EuRoC Dataset	96
4.7.2	Trials with Graduate Students	98
4.8	Conclusions	101

Parts of this chapter appeared in:

Christoper L Choi, Binbin Xu, Stefan Leutenegger (2023). **Accurate and Interactive Visual-Inertial Sensor Calibration with Next-Best-View and Next-Best-Trajectory Suggestion**. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. [[Choi et al., 2023](#)]

4.1 Introduction

In order to use Visual-Inertial (VI) sensors in computer vision or state-estimation tasks the calibration parameters must first be obtained. Conventionally, VI-sensors are calibrated by an expert who would often collect calibration data by positioning and moving the sensors in front of a calibration target such as a checkerboard or grid of fiducial markers, then use an offline calibration tool such as Kalibr [Furgale et al., 2013] to estimate the sensor calibration parameters. Good calibration results, however, may only be achieved, if the right kind and right amount of data is collected. More specifically, two potential practical issues arise during data capture: first, the choice of calibration views and the range of motions needed is not immediately clear to the non-expert. Secondly, the amount of data the user has to collect for calibration is also unclear, often collecting too much or too little data. A common practice to address these issues is to collect *multiple* calibration data sequences, however, this is impractical in the field and identifying which calibration is optimal becomes a tedious and time-consuming task.

A straight forward solution to this problem would be to mount the VI-sensor on a robot arm and perform a rehearsed or optimal calibration “dance”, such as in [Ao et al., 2022]. However, this requires extra hardware and is not a practical solution for many applications. As an alternative to classic offline calibration methods, one can estimate the calibration parameters within a state-estimation framework such as OKVIS [Leutenegger et al., 2014], VINS-MONO [Qin et al., 2018], and OpenVINS [Geneva et al., 2020] in real-time. Note, however, that any of these frameworks require some form of sufficiently accurate initial calibration, as well as sufficient visual features and motion excitation, therefore suffering from similar issues as offline calibration. Furthermore, natural keypoints and the lack of precise knowledge of the corresponding 3D positions may not produce the best possible results.

In this work, we present an interactive VI-sensor calibration pipeline that helps guide a non-expert in collecting *informative* calibration data for a VI-sensor *once* through Next-Best-View (NBV) and Next-Best-Trajectory (NBT) suggestions (as shown in Figure. 4.1)

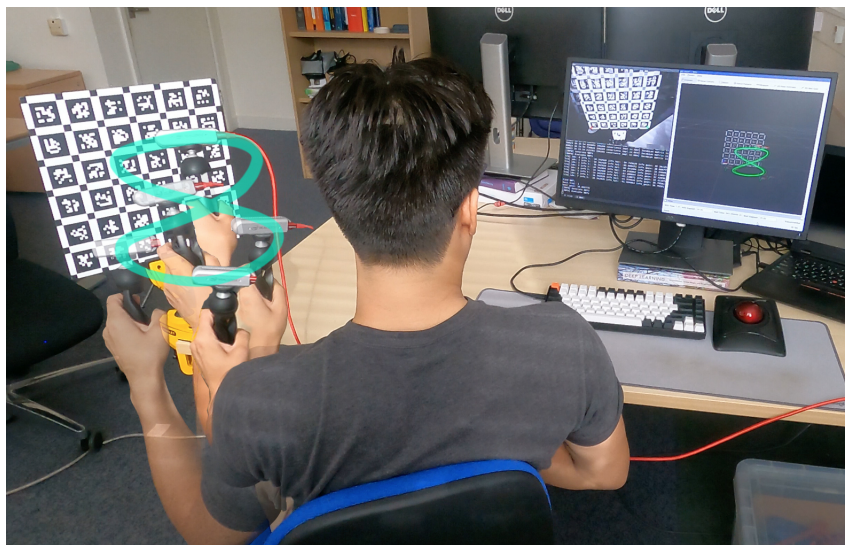


Figure 4.1: Our system interactively suggests next-best-actions to collect calibration data.

in order to efficiently obtain sound calibrations. We show through extensive quantitative experiments on calibration sequences and several self-collected VICON real-world datasets that calibration parameters optimised through our system are more accurate and consistent than Kalibr by testing on state-of-the-art VI-SLAM ORBSLAM3 [Campos et al., 2021]. In summary our contributions are:

- A complete and open-sourced interactive VI-camera calibration tool that supports any number of cameras;
- An information-theoretic procedure to identify the most informative Next-Best-View (NBV) and Next-Best-Trajectory (NBT) among a pre-defined set of viewpoints and trajectory primitives;
- An interactive graphical user interface for guiding the user through the calibration data collection process;
- Through experiments we show that our proposed method is faster, more accurate and more reliable compared to state of the art traditional *non-guided* calibration methods, such as Kalibr [Furgale et al., 2013], even when used by *novices* in our user study ($n = 16$).

The remainder of this chapter is organised as follows: we overview related work in Section 4.2. Then, we overview our system in Section 4.4, which is later detailed concerning vision-only calibration with NBV in Section 4.5 and VI extrinsics calibration and temporal calibration with NBT in Section 4.6. Section 4.7 then presents the experimental results.

4.2 Related Work

Offline Methods. In the robotics community, early works in VI-sensor calibration methods such as [Alves et al., 2003, Lobo and Dias, 2007, Mirzaei and Roumeliotis, 2008, Dong-Si and Mourikis, 2012] showed that it is possible to calibrate the extrinsics between a camera and IMU, with Kalibr [Furgale et al., 2013] regarded as the current state-of-the-art tool. It is an offline method capable of calibrating a multi-camera system, as well as a VI system. However, the use of this tool requires expert knowledge, as the result is highly dependent on the quality of the calibration data captured. Therefore, the calibration process may in practice have to be repeated until desired results are reached.

Online Methods. State of the art state-estimation framework such as OKVIS [Leutenegger et al., 2014], VINS-MONO [Qin et al., 2018], and OpenVINS [Geneva et al., 2020] can in practice estimate the calibration parameters in real-time. However, these frameworks require sufficiently accurate initial calibrations, as well as sufficient visual features and motion excitation in order to operate accurately.

Reinforcement Learning Methods: There has been a growing interest in using reinforcement learning for calibration such as [Ao et al., 2022, Chen et al., 2021, Nobre and Heckman, 2019] whereby the goal is to learn informative trajectories to render the VI-calibration parameters observable. However, the requirement of a robot arm to perform these motions is not always practical in the field. Further, these works do not provide quantitative results through a SLAM system to verify the optimality of the calibrated parameters.

Information-Theoretic Methods. The first calibration tool with an emphasis in guiding the user through capturing a good calibration sequence for a monocular camera is AprilCal [Richardson et al., 2013]. The method used a quality metric to find and suggest the NBV in real-time during the camera calibration process. AprilCal, however, only supports calibrating the intrinsics of a single monocular camera.

A more recent work that uses an information-theoretic approach for VI-sensor calibration is the work of [Schneider et al., 2017, Schneider et al., 2019], where they proposed a segment-based method for calibrating a VI-sensor system in a AR / VR headset and self-driving car setting. The idea is to extract informative data during online state-estimation using an information-theoretic metric, and then perform a full-batch optimisation to update the calibration parameters offline. This approach, however, relies on the fact that the VI-sensors are calibrated well initially. Secondly, the available data does not guarantee informative segments for calibration.

In this chapter, we place heavy emphasis on collecting *informative* calibration data by using an information-theoretic metric to find the NBV and NBT in real-time, and by *interactively* guiding the user in collecting them in order to calibrate the intrinsics, extrinsics, and time shift of a VI sensor. This is in contrast to current state-of-the-art calibration tools such as Kalibr [Furgale et al., 2013] that assume the collected calibration data has sufficient views and range of motion.

4.3 Background

In robotics, the maximum a posteriori (MAP) estimator is commonly used to solve the camera and VI calibration problem,

$$\hat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmax}} p(\mathbf{x}|\mathbf{z}), \quad (4.1)$$

where $\mathbf{x}$ is the state vector which may be comprised of poses, velocities, IMU biases and calibration parameters we are interested in jointly estimating, given the measurements, $\mathbf{z}$. The error terms, $\mathbf{e}_i$, are assumed to be Gaussian and independent, meaning the errors from

different measurements or observations are uncorrelated. If the errors were correlated, the nonlinear least squares optimization would become singular, causing difficulties in solving the problem. Moreover, it would prevent accurate estimation of the posterior covariance matrix. Maximising Eq. (4.1) is equivalent to solving the sum of nonlinear least squares using a nonlinear optimisation algorithm such as the Gauss Newton method,

$$\sum_i \mathbf{E}_i^\top \mathbf{W}_i \mathbf{E}_i \Delta \mathbf{x} = \sum_i -\mathbf{E}_i^\top \mathbf{W}_i \mathbf{e}_i(\mathbf{x}), \quad (4.2)$$

where $\Delta \mathbf{x}$ is the update vector, $\mathbf{e}_i(\mathbf{x})$ is the i^{th} error term evaluated at the current estimate $\mathbf{x}$, $\mathbf{E}_i$ is the Jacobian matrix of the error term and $\mathbf{W}_i$ the measurement information.

At convergence of the optimisation, we may approximate the posterior distribution as a Gaussian with mean $\mathbf{x}$ and find the covariance matrix $\hat{\Sigma}_{\mathbf{x}}$ by inverting the quantity $\sum_i \mathbf{E}_i^\top \mathbf{W}_i \mathbf{E}_i$, also known as the Fisher Information matrix. However, recall that the state vector $\mathbf{x}$ not only contains calibration parameters $\boldsymbol{\theta}$, but also other state variables not related to the calibration parameters which we denote as $\boldsymbol{\gamma}$. In the context of calibration, we are only interested in the estimated calibration parameters, $\boldsymbol{\theta}$, and the covariance of the calibration parameters $\Sigma_{\boldsymbol{\theta}\boldsymbol{\theta}}$. Expressing $\mathbf{x}$ and $\Sigma_{\mathbf{x}}$ in partition form,

$$\mathbf{x} = \begin{bmatrix} \boldsymbol{\theta} \\ \boldsymbol{\gamma} \end{bmatrix}, \quad \Sigma_{\mathbf{x}} = \begin{bmatrix} \Sigma_{\boldsymbol{\theta}\boldsymbol{\theta}} & \Sigma_{\boldsymbol{\theta}\boldsymbol{\gamma}} \\ \Sigma_{\boldsymbol{\gamma}\boldsymbol{\theta}} & \Sigma_{\boldsymbol{\gamma}\boldsymbol{\gamma}} \end{bmatrix}, \quad (4.3)$$

we can employ marginalisation on Normal distributions to get $p(\boldsymbol{\theta}|\mathbf{z}) = \mathcal{N}(\boldsymbol{\theta}, \Sigma_{\boldsymbol{\theta}\boldsymbol{\theta}})$, by extracting the corresponding blocks in Eq. (4.3).

To objectively quantify whether the next VI measurements are informative for the VI calibration problem, we used the Mutual Information (MI) defined in [Maye et al., 2013],

$$I(\boldsymbol{\theta}; \tilde{\mathbf{z}}_2) = \frac{1}{2} \log \frac{|\Sigma_{\boldsymbol{\theta}\boldsymbol{\theta}|\mathbf{z}_1}|}{|\Sigma_{\boldsymbol{\theta}\boldsymbol{\theta}|\mathbf{z}_2}|}, \quad (4.4)$$

where $\Sigma_{\boldsymbol{\theta}\boldsymbol{\theta}|\mathbf{z}_1}$ is the covariance estimate of $\boldsymbol{\theta}$ using measurements from timestep 1 to k as $\mathbf{z}_1$, and $\Sigma_{\boldsymbol{\theta}\boldsymbol{\theta}|\mathbf{z}_2}$ is the covariance estimate of $\boldsymbol{\theta}$ using measurements from 1 to k and including n new sensor measurements as $\mathbf{z}_2$, finally $|\cdot|$ is the matrix determinant. In summary, with Eq. (4.4) we can measure the amount of information $\mathbf{z}_2$ (next VI-sensor measurements) conveys to our current estimate $\boldsymbol{\theta}|\mathbf{z}_1$.

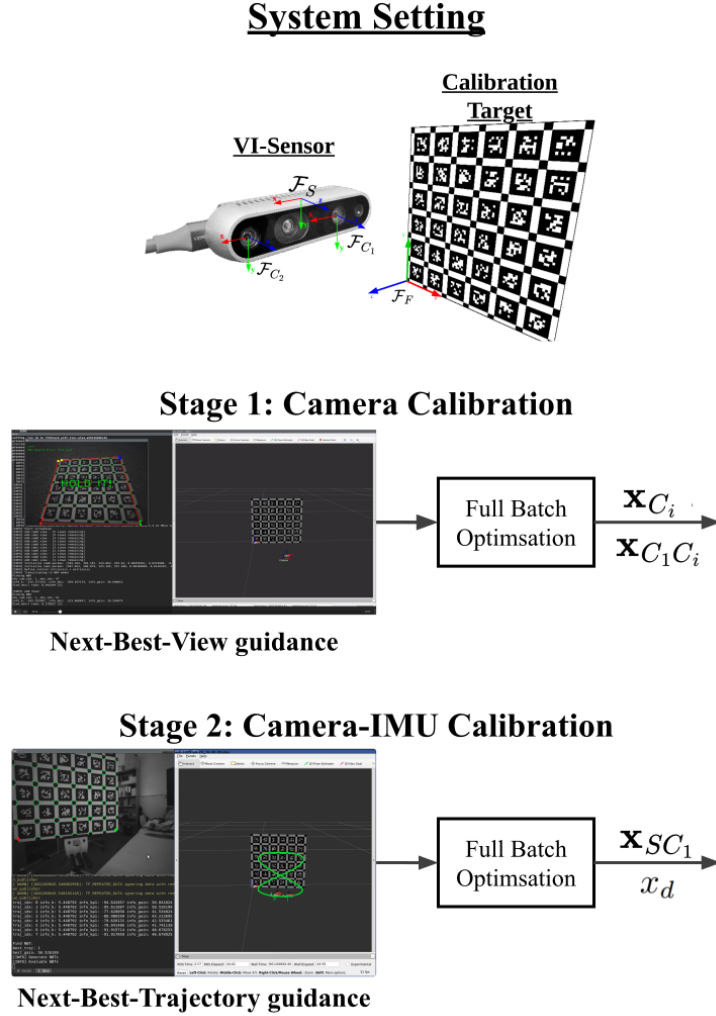


Figure 4.2: An overview of our VI calibration pipeline.

4.4 System Overview

An overview of our proposed calibration system is illustrated in Figure. 4.2. It consists of two stages. The first stage aims to perform vision-only camera intrinsics and extrinsics calibration employing Next-Best-View (NBV) feedback. In the second stage the camera-IMU extrinsics are found by using Next-Best-Trajectory (NBT) feedback, with the camera intrinsics and extrinsics obtained in the previous stage fixed. Both stages of the calibration process require the use of a static fiducial marker grid of known size as a calibration target.

Specifically, we use a planar calibration target grid of AprilTags [Richardson et al., 2013] introduced by Kalibr [Furgale et al., 2013]. Throughout this work, the VI-sensor to be calibrated is assumed to capture images and inertial measurements with the same clock source.

4.5 Camera Intrinsics and Extrinsics Calibration

In the following, we detail our approach of using Mutual Information (MI) and Next-Best-View (NBV) to calibrate intrinsics and extrinsics of all cameras.

4.5.1 States

For the camera calibration problem, the states to be estimated consist of the camera poses relative to the fiducial target coordinate frame $\vec{F}_F$ as $\mathbf{x}_{FC_1}$, camera extrinsics relative to reference camera 1, $\mathbf{x}_{C_1C_i}$, and camera intrinsics, $\mathbf{x}_{C_i}$, of the form:

$$\begin{aligned}\mathbf{x}_{FC_1} &= \begin{bmatrix} {}_F\mathbf{r}_{FC_1}^\top & \mathbf{q}_{FC_1}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \\ \mathbf{x}_{C_1C_i} &= \begin{bmatrix} {}_{C_1}\mathbf{r}_{C_1C_i}^\top & \mathbf{q}_{C_1C_i}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \\ \mathbf{x}_{C_i} &= \begin{bmatrix} f_x & f_y & c_x & c_y & k_1 & k_2 & p_1 & p_2 \end{bmatrix}^\top \in \mathbb{R}^8,\end{aligned}\tag{4.5}$$

where $\vec{F}_{C_i}$ denotes the coordinate frame of the i^{th} camera on the sensor assembly. We used the Radial-Tangential camera model consisting of focal lengths f_u, f_v , principal centre c_u, c_v , radial distortion parameters k_1, k_2 , and tangential distortion parameters p_1, p_2 as the camera intrinsics. Note that any other projection model could be supported in principle. The full state vector for camera calibration thus becomes,

$$\mathbf{x} = \begin{bmatrix} \underbrace{\mathbf{x}_{FC_1}^{\top,1} \dots \mathbf{x}_{FC_1}^{\top,k}}_{\text{Reference Camera 1 Poses}} & \underbrace{\mathbf{x}_{C_1C_1}^\top \dots \mathbf{x}_{C_1C_i}^\top}_{\text{Camera Extrinsics}} & \underbrace{\mathbf{x}_{C_1}^\top \dots \mathbf{x}_{C_i}^\top}_{\text{Camera Intrinsics}} \end{bmatrix}^\top.\tag{4.6}$$

The minimal error state vectors for $\mathbf{x}_{FC_1}$, $\mathbf{x}_{C_1C_i}$, and $\mathbf{x}_{C_1C_i}$ are:

$$\begin{aligned}\delta\chi_{FC_1} &= \begin{bmatrix} \delta_F \mathbf{r}_{FC_1}^\top & \delta\alpha_{FC_1}^\top \end{bmatrix}^\top \in \mathbb{R}^6, \\ \delta\chi_{C_1C_i} &= \begin{bmatrix} \delta_{C_1} \mathbf{r}_{C_1C_i}^\top & \delta\alpha_{C_1C_i}^\top \end{bmatrix}^\top \in \mathbb{R}^6, \\ \delta\chi_{C_i} &= \begin{bmatrix} \delta f_x & \delta f_y & \delta c_x & \delta c_y & \delta k_1 & \delta k_2 & \delta p_1 & \delta p_2 \end{bmatrix}^\top \in \mathbb{R}^8.\end{aligned}\tag{4.7}$$

4.5.2 Calibration Formulation

To estimate the camera calibration parameters we used a nonlinear least squares framework to minimise the cost function, J_{camera} , containing reprojection errors, $\mathbf{e}_r$, and the information matrix of the respective camera measurement, $\mathbf{W}_r$. The cost function has the form:

$$J_{\text{camera}}(\mathbf{x}) = \frac{1}{2} \sum_{i=1}^I \sum_{k=1}^K \sum_{j \in \mathcal{J}(i,k)} \mathbf{e}_r^{i,j,k\top} \mathbf{W}_r^{i,j,k} \mathbf{e}_r^{i,j,k},\tag{4.8}$$

where i is the camera index, k denotes the camera frame index, and j denotes the fiducial target corner index. Finally, $\mathcal{J}(i,k)$ denotes the set of observable fiducial corner indices in the i^{th} camera index and k^{th} camera frame index.

Here, the standard reprojection error, $\mathbf{e}_r$, was used:

$$\mathbf{e}_r^{i,j,k} = \tilde{\mathbf{z}}^{i,j,k} - \mathbf{h}_i(\mathbf{T}_{C_1C_i} \mathbf{T}_{FC_1}^{-1} \mathbf{F} \mathbf{r}_{FF_j}, \mathbf{x}_{C_i}),\tag{4.9}$$

whereby $\mathbf{h}_i(\cdot)$ denotes the camera projection and distortion model. It needs as an input the fiducial corner, $\mathbf{F} \mathbf{r}_{FF_j}$, camera pose, $\mathbf{T}_{FC_1}$, camera extrinsics, $\mathbf{T}_{C_1C_i}$, and camera intrinsics $\mathbf{x}_{C_i}$. Lastly, $\tilde{\mathbf{z}}^{i,j,k}$ is the observed fiducial corner measurement.

4.5.3 Real-time Estimation

Since Eq. (4.8) will grow in complexity with every camera frame added, it cannot be solved in real-time as the problem size increases. We therefore adopted a fixed-lag sliding window scheme similar to [Leutenegger et al., 2014], whereby the sliding window is bounded by marginalising out old camera poses $\mathbf{x}_{FC_1}$ with the Schur Complement, leading to a respective linear prior that enters the cost. Note that this is only needed for the real-time feedback to the user, and we still solve the full batch problem offline for the final calibration solution.

4.5.4 Camera Calibration With Next-Best-View

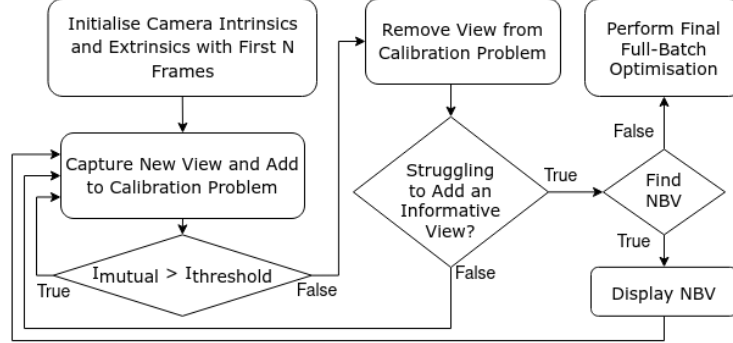


Figure 4.3: Camera calibration pipeline

In contrast to standard full-batch camera calibration, where the calibration data is first collected and then solved as a two step process, our method takes a more integrated approach, whereby data collection and solving the calibration problem are performed incrementally, until the addition of new data is no longer informative to the camera calibration problem (see Figure. 4.3).

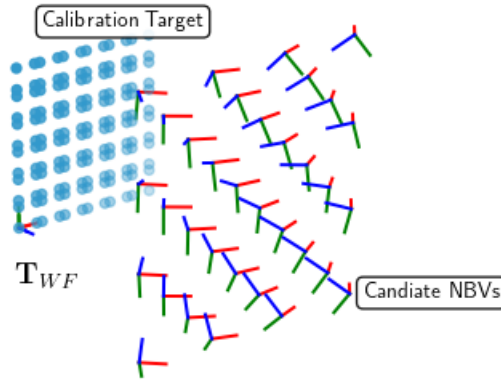


Figure 4.4: NBV candidate poses in-front of the calibration target

First, the camera intrinsics and extrinsics are initialised with the first N camera frames of a static fiducial marker of known size by minimising the cost function in Eq. (4.8). Once the camera parameters are initialised, the user is guided to maximise the calibration target measurement coverage over the image space. The information content of each camera view is evaluated using Eq. (4.4). Views which contain a MI score below the user-defined

threshold, $I_{\text{threshold}}$ ($I_{\text{threshold}} = 0.2$, same as in Kalibr [Furgale et al., 2013]), are removed from the calibration problem. If, however, the new candidate views are not informative enough (no new views added to the calibration problem in the last 3 frames), the calibration tool enters into “Find Next-Best-View” mode where it evaluates a set of possible NBVs. Similar to [Richardson et al., 2013], NBVs are pre-determined by an expert ahead of time in order to reduce the search space and make the computation feasible in real-time (see Figure. 4.4). Using Eq. (4.4), the NBV is the one that has the highest mutual information. Once the NBV is determined, the calibration tool will guide the user to the NBV interactively through the graphical user-interface in capturing that view. If the mutual information of the NBV is found to be below $I_{\text{threshold}}$ the calibration tool stops capturing further measurements and proceeds to performing a final full batch optimisation to estimate the final calibration parameters.

4.6 Camera-IMU Extrinsic Calibration

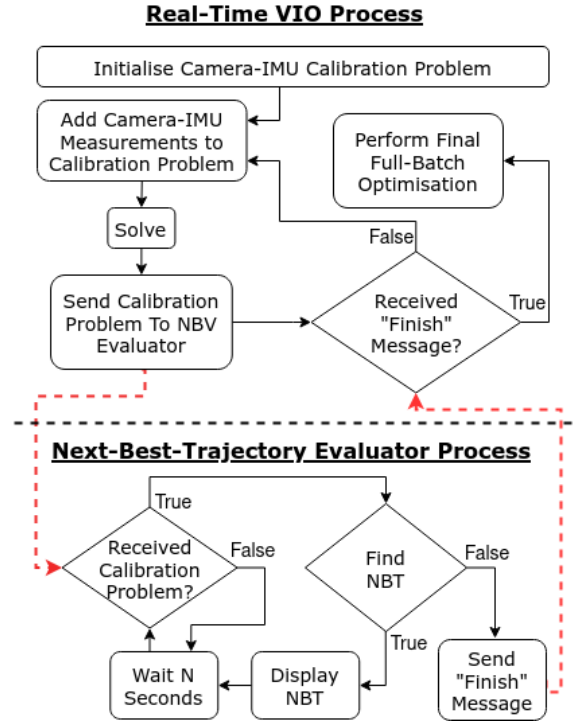


Figure 4.5: Camera-IMU calibration pipeline

Once the camera intrinsics and extrinsics are known (from Sec. 4.5), we proceed to, without loss of generality, calibrate the extrinsics between the reference camera 1 and IMU, $\mathbf{T}_{SC_1}$, and camera-IMU delay, t_d , of a VI-sensor.

4.6.1 States

The variables to be estimated are the VI-sensor pose at discrete camera frame index k , $\mathbf{x}_{WS}^k$, fiducial target pose in the inertial frame $\mathbf{x}_{WF}$, extrinsics between reference camera 1 and IMU $\mathbf{x}_{SC_1}$, and camera-IMU time delay x_d :

$$\begin{aligned}\mathbf{x}_{WS} &= \begin{bmatrix} {}_W\mathbf{r}_{WS}^\top & \mathbf{q}_{WS}^\top & {}_W\mathbf{v}_{WS}^\top & \mathbf{b}_g^\top & \mathbf{b}_a^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3 \times \mathbb{R}^9, \\ \mathbf{x}_{WF} &= \begin{bmatrix} {}_W\mathbf{r}_{WF}^\top & \mathbf{q}_{WF}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \\ \mathbf{x}_{SC_1} &= \begin{bmatrix} {}_S\mathbf{r}_{SC_1}^\top & \mathbf{q}_{SC_1}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \\ x_d &= t_d \in \mathbb{R},\end{aligned}\tag{4.10}$$

where the state vector $\mathbf{x}_{WS}$ holds the VI-sensor position in the inertial frame ${}_W\mathbf{r}_{WS}$, the body orientation represented by a quaternion $\mathbf{q}_{WS}$, the velocity expressed in the sensor frame ${}_W\mathbf{v}_{WS}$, as well as the gyroscope and accelerometer biases $\mathbf{b}_g$ and $\mathbf{b}_a$. The state vectors $\mathbf{x}_{SC_1}$ and $\mathbf{x}_{WF}$ hold the sensor-camera relative pose and fiducial pose, respectively. The full state vector for camera-IMU calibration thus becomes,

$$\mathbf{x} = \begin{bmatrix} \underbrace{\mathbf{x}_{WS_1}^{\top,1} \dots \mathbf{x}_{WS_1}^{\top,k}}_{\text{Sensor Poses}} & \underbrace{\mathbf{x}_{WF}^\top}_{\text{Fiducial Pose}} & \underbrace{\mathbf{x}_{SC_1}^\top}_{\text{Camera-IMU Extrinsic}} & \underbrace{x_d}_{\text{Camera-IMU Time-Delay}} \end{bmatrix}^\top.\tag{4.11}$$

The minimal error state vectors for $\mathbf{x}_{WS}$, $\mathbf{x}_{SC_1}$, $\mathbf{x}_{WF}$, and x_d are therefore:

$$\begin{aligned}\delta\chi_{WS} &= \begin{bmatrix} \delta {}_W\mathbf{r}_{WS}^\top & \delta\boldsymbol{\alpha}_{WS}^\top & \delta {}_W\mathbf{v}_{WS}^\top & \delta\mathbf{b}_g^\top & \delta\mathbf{b}_a^\top \end{bmatrix}^\top \in \mathbb{R}^{15}, \\ \delta\chi_{SC_1} &= \begin{bmatrix} \delta {}_S\mathbf{r}_{SC_1}^\top & \delta\boldsymbol{\alpha}_{SC_1}^\top \end{bmatrix}^\top \in \mathbb{R}^6, \\ \delta\chi_{WF} &= \begin{bmatrix} \delta {}_W\mathbf{r}_{WF}^\top & \delta\boldsymbol{\alpha}_{WF}^\top \end{bmatrix}^\top \in \mathbb{R}^6, \\ \delta\chi_d &= \delta t_d \in \mathbb{R}.\end{aligned}\tag{4.12}$$

4.6.2 Calibration Formulation

Similar to Sec. 4.5, we seek to formulate the VI calibration problem as one joint nonlinear-optimisation of a cost function $J_{\text{imu-cam}}(\mathbf{x})$ containing both (weighted) reprojection errors $\mathbf{e}_r$ and (weighted) temporal error term from the IMU $\mathbf{e}_s$:

$$J_{\text{imu-cam}}(\mathbf{x}) = \underbrace{\frac{1}{2} \sum_{i=1}^I \sum_{k=1}^K \sum_{j \in \mathcal{J}(i,k)} \mathbf{e}_r^{i,j,k\top} \mathbf{W}_r^{i,j,k} \mathbf{e}_r^{i,j,k}}_{\text{visual}} + \underbrace{\frac{1}{2} \sum_{k=1}^{K-1} \mathbf{e}_s^{k\top} \mathbf{W}_s^k \mathbf{e}_s^k}_{\text{inertial}}, \quad (4.13)$$

where i is the camera index of the VI-sensor, k denotes the camera frame index, and j denotes the fiducial target corner index. The set $\mathcal{J}(i, k)$ represents the indices of fiducial target corners observed in the k^{th} frame and the i^{th} camera.

The reprojection error was used to estimate the camera-IMU extrinsics $\mathbf{T}_{SC_1}$, sensor pose in the world frame $\mathbf{T}_{WS}$ and fiducial target in the world frame $\mathbf{T}_{WF}$:

$$\mathbf{e}_r = \tilde{\mathbf{z}}^{i,j,k} - \mathbf{h}_i(\mathbf{T}_{C_1C_i}^{-1} \mathbf{T}_{SC_1}^{-1} \mathbf{T}_{SW}^k \mathbf{T}_{WF} \mathbf{F}_{FF_j}, \mathbf{x}_{C_i}), \quad (4.14)$$

where $\mathbf{h}_i(\cdot)$ denotes the i^{th} camera projection model which includes distortion, $\mathbf{F}_{FF_j}$ denotes the j^{th} fiducial target corner point and $\tilde{\mathbf{z}}^{i,j,k}$ denotes the corresponding measurement seen in camera i and image frame k in image coordinates. The camera-intrinsics $\mathbf{x}_{C_i}$ and camera-extrinsics $\mathbf{T}_{C_1C_i}$ estimated in Sec. 4.5 are fixed.

The fiducial target in the world frame $\mathbf{T}_{WF}$ is first initialised using initial measurements from the IMU and camera assuming low acceleration, where the measured acceleration vector corresponds to (inverse) acceleration due to gravity—yielding the camera pose $\mathbf{T}_{WC_i}$. Without loss of generality, we set the camera position and yaw around the world-z axis to zero. Next, the relative pose between the fiducial target and the i^{th} camera, $\mathbf{T}_{FC_i}$, is computed with fiducial corner measurements using 3D-2D RANSAC and bundle adjustment, after which we can compose $\mathbf{T}_{WF} = \mathbf{T}_{WC_i} \mathbf{T}_{FC_i}$.

Before formulating the nonlinear inertial error term in Eq. (4.13), we must first describe the differential equations that describe the IMU kinematics and bias evolution over time. Using the commonly used IMU model from [Savage, 1998] and simplifications for MEMS-IMUs as in [Shin and El-Sheimy, 2004], under the assumption that the measured effects of Earth's rotation are small, the IMU kinematics is:

$$\begin{aligned}
 {}_W\dot{\mathbf{r}}_{WS} &= {}_W\mathbf{v}_{WS}, \\
 \dot{\mathbf{q}}_{WS} &= \frac{1}{2}\Omega({}_S\tilde{\boldsymbol{\omega}}, \mathbf{w}_g, \mathbf{b}_g) \mathbf{q}_{WS}, \\
 {}_W\dot{\mathbf{v}}_{WS} &= \mathbf{C}_{WS}({}_S\tilde{\mathbf{a}} + \mathbf{w}_a - \mathbf{b}_a) + {}_W\mathbf{g}, \\
 \dot{\mathbf{b}}_g &= \mathbf{w}_{b_g}, \\
 \dot{\mathbf{b}}_a &= \mathbf{w}_{b_a},
 \end{aligned} \tag{4.15}$$

where $\mathbf{w} := [\mathbf{w}_g^\top \ \mathbf{w}_a^\top \ \mathbf{w}_{b_g}^\top \ \mathbf{w}_{b_a}^\top]^\top$ are each uncorrelated zero-mean Gaussian white noise processes. ${}_S\tilde{\mathbf{a}}$ and ${}_S\tilde{\boldsymbol{\omega}}$ are accelerometer and gyro measurements. ${}_W\mathbf{g}$ represents the Earth's gravitational acceleration vector. The gyro and accelerometer biases are modeled as random walk. The matrix Ω is formed from the estimated angular rate ${}_S\boldsymbol{\omega} = {}_S\tilde{\boldsymbol{\omega}} + \mathbf{w}_g - \mathbf{b}_g$, with gyro measurement ${}_S\tilde{\boldsymbol{\omega}}$:

$$\Omega(\boldsymbol{\omega}) := \begin{bmatrix} -[\boldsymbol{\omega}]_\times & \boldsymbol{\omega} \\ -\boldsymbol{\omega}^\top & 0 \end{bmatrix}, [\boldsymbol{\omega}]_\times = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix}. \tag{4.16}$$

For the IMU error term, we adopted the pre-integration scheme in [Forster et al., 2015], where the error is the difference between the predicted relative state and the actual relative state, with the exception of orientation, where a simple multiplicative minimal error was used:

$$\mathbf{e}_S^k(\mathbf{x}_S^k, \mathbf{x}_S^{k+1}, \mathbf{z}_S^k) = \begin{bmatrix} {}_W\hat{\mathbf{r}}_{WS}^{k,k+1}(t_d) - {}_W\mathbf{r}_{WS}^{k,k+1} \\ 2 \left[\mathbf{q}_S^{k,k+1} \otimes \hat{\mathbf{q}}_S^{k,k+1}(t_d) \right]_{1:3} \\ {}_S\hat{\mathbf{v}}^{k,k+1}(t_d) - {}_S\mathbf{v}^{k,k+1} \\ \hat{\mathbf{b}}_g^{k+1}(t_d) - \mathbf{b}_g^{k+1} \\ \hat{\mathbf{b}}_a^{k+1}(t_d) - \mathbf{b}_a^{k+1} \end{bmatrix} \in \mathbb{R}^{15}. \tag{4.17}$$

In addition to estimating the relative state, we further include the camera-IMU time delay scalar t_d . Since it is only a 1 dimensional parameter, the 15×1 Jacobian was obtained through the central finite difference by perturbing the IMU timestamps.

4.6.3 Real-time Estimation

To keep the problem in Eq. (4.13) bounded for real-time operation, we used the same approach as in Sec. 4.5.3 and adopted a fixed-lag sliding window scheme, marginalising out old sensor poses $\mathbf{T}_{WS}$, velocities ${}_W\mathbf{v}_{WS}$, accelerometer biases $\mathbf{b}_a$ and gyroscope biases $\mathbf{b}_g$. A full batch optimisation using all measurements will be performed to obtain the final calibration solution. The camera-IMU time delay parameter is fixed during online guidance, and estimated in the final full batch optimisation.

4.6.4 Next-Best-Trajectories

Similar to [Nobre and Heckman, 2019], given our goal is to provide intuitive, easy and real-time feedback for a non-expert user to calibrate the VI-sensor, we discretized the continuous search space and used the results of [Yang et al., 2019] to design 6 non-degenerate NBTs that are computationally feasible in real-time, easy to display and followed by the user (see Figure. 4.6). Our NBTs are observable as the fisher-information matrix has to be invertible in order to evaluate the information gain [Nobre and Heckman, 2019].

Inspired by the Lissajous curve equations, each NBT is parameterised as:

$$\begin{aligned} x &= w_{\text{traj}} \sin(at + \delta) + 0.5w_{\text{calib}}, \\ y &= h_{\text{traj}} \cos(bt) + 0.5h_{\text{calib}}, \\ z &= \sqrt{d_{\text{nbt}} - x^2 - y^2}, \end{aligned} \tag{4.18}$$

where x , y and z are the trajectory positions relative to the fiducial target frame $\mathcal{F}_F$ to form ${}_F\mathbf{r}_{FS}$, d_{nbt} is the distance away from the fiducial target center, w_{traj} and h_{traj} are the trajectory max width and height, w_{calib} and h_{calib} are the fiducial target width and height, δ represents the phase angle offset, and finally a and b are constants that determine the shape

of the trajectory (e.g. a ratio of $\frac{a}{b} = 2$ forms a figure of 8). Finally, the sensor's orientation are parameterised as Euler angles and designed such that it is always pointing towards the center of the calibration target:

$$\begin{aligned}\phi &= \phi_{\text{bound}} \sin(2\pi t) + \pi, \\ \theta &= \theta_{\text{bound}} \sin(2\pi t), \\ \psi &= 0.0,\end{aligned}\tag{4.19}$$

where ϕ , θ and ψ are Euler angles around the x, y and z-axis to form C_{FS} , respectively, and ϕ_{bound} and θ_{bound} are the maximum rotation around x and y-axis, respectively.

To ensure the velocity and angular velocity are realistic, we parameterise t in Eq. (4.18) and Eq. (4.19) as a function of k between $[0, t_{\text{nbt}}]$ such that the first derivative of both equations, velocity and angular velocity, start and end at 0, $t(k) = \sin^2(\pi k/2 t_{\text{nbt}})$, where t_{nbt} is the time to complete a NBT. Differentiating both Eq. (4.18) and Eq. (4.19) enables us to simulate the camera and IMU measurements for evaluating NBTs using Eq. (4.4).

4.6.5 Camera-IMU Calibration With Next-Best-Trajectory

The Camera-IMU calibration begins with two separate processes running in parallel, a real-time VI estimator solving Eq. (4.13) and a NBT evaluator (see Figure. 4.5). The real-time VI camera parameters are initialised using the parameters optimised in Section. 4.5 and are fixed throughout. The fiducial pose, T_{WF} and camera-IMU extrinsics, T_{SC_1} on the other hand are initialised by solving Eq. (4.13) with the first N camera frames, and IMU measurements between the first and last camera frame timestamps.

As the real-time VI estimator is solving the camera-IMU calibration problem, it periodically sends the calibration problem data to the NBT evaluator process. The NBT evaluator in turn would use the data to evaluate the MI of a set of pre-defined NBTs (Section. 4.6.4) using Eq. (4.4), find the NBT with the highest MI and guide the user in executing the NBT in order to render the calibration parameters optimally observable, i.e. reducing the expected uncertainty on the estimated camera-IMU extrinsics. If none of the candidate

4. Accurate And Interactive Visual-Inertial Calibration with Next-Best-View and Next-Best-Trajectory

NBTs satisfies $I_{\text{mutual}} > I_{\text{threshold}}$ ($I_{\text{threshold}} = 0.2$) then the NBT evaluator sends a “finish” message to communicate to the real-time VI process that it should proceed to perform a final full-batch calibration.

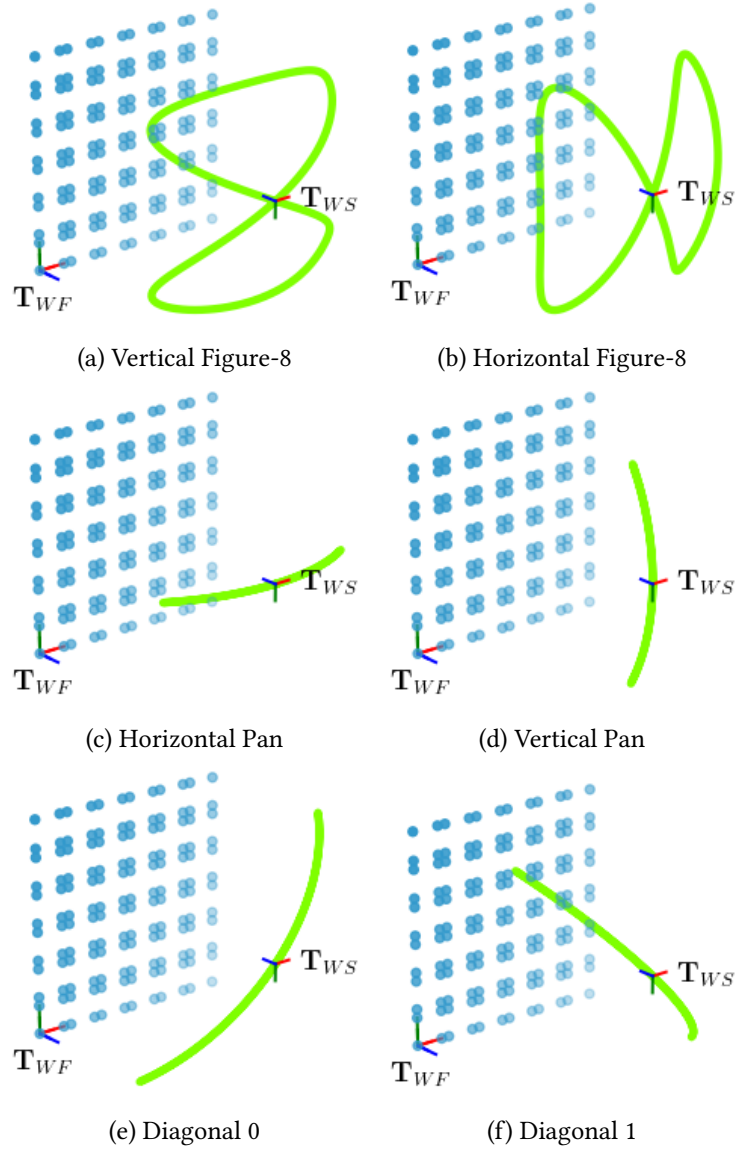


Figure 4.6: Next-Best-Trajectory (NBT) candidates in-front of a calibration target

4.7 Experiments

To evaluate our method, we conducted two sets of experiments. First, we evaluated our calibration pipeline in offline mode with the EuRoC [Burri et al., 2016] dataset to verify our calibration *accuracy* is competitive against that of Kalibr’s *without the interactive component* of our system, and despite different approaches to solving the camera-IMU calibration problem, where Kalibr uses a continuous time full-batch optimisation in contrast to our method which uses a discrete time full-batch optimisation. This is *independent* of our contributions regarding interactivity. With this we wanted to highlight our calibration tool without interactivity is at least as good as Kalibr.

Since the main motivation in this work is to provide non-experts with good calibration results for VIO/VI-SLAM systems, we further conducted experiments involving a small batch of graduate students to prove that our system can *efficiently* and *reliably* calibrate VI-sensors, achieving superior performance for existing VIO and VI-SLAM systems. To compare the calibrations we used them in ORBSLAM3 [Campos et al., 2021] and evaluated the accuracy using the evaluation scheme of [Sturm et al., 2012] with RMSE Absolute Trajectory Error (ATE) by aligning the estimated trajectory with the ground-truth.

All experiments were conducted on a Lenovo P52 Thinkpad laptop containing an Intel Core i7-8750H CPU at 2.2 Ghz with 16GB of memory running Ubuntu 20.04 and ROS Melodic. The experiments with the graduate students were conducted with the aim of calibrating an Intel RealSense D435i which contains a stereo IR global shutter depth sensor, a monocular RGB rolling shutter sensor, and additionally an IMU sensor running at 15Hz, 15Hz and 400Hz respectively. For our purposes, we *do not* use the RGB rolling shutter sensor. We have instead disabled the IR projector and used the stereo IR global shutter depth sensors as a standard gray-scale stereo camera.

During the camera and VI calibrations the default settings for Kalibr [Furgale et al., 2013] were used to generate their results, whereas in our method we used Cauchy loss ($s = 1.5$) on the reprojection errors and a fixed-lag smoothing window size of 10 and 3 for the camera

4. Accurate And Interactive Visual-Inertial Calibration with Next-Best-View and Next-Best-Trajectory

calibration and camera-IMU calibration stages respectively. The IMU parameters used for the camera-IMU calibration are: $\sigma_a = 2.52 \times 10^{-2} \frac{\text{m}}{\text{s}^2} \frac{1}{\sqrt{\text{Hz}}}$ for the accelerometer noise density, $\sigma_{ba} = 4.41 \times 10^{-3} \frac{\text{m}}{\text{s}^3} \frac{1}{\sqrt{\text{Hz}}}$ for the accelerometer drift noise density, $\sigma_g = 2.78 \times 10^{-3} \frac{\text{rad}}{\text{s}} \frac{1}{\sqrt{\text{Hz}}}$ for gyroscope noise density, and $\sigma_{bg} = 1.65 \times 10^{-5} \frac{\text{rad}}{\text{s}^2} \frac{1}{\sqrt{\text{Hz}}}$ gyroscope drift noise density.

4.7.1 Calibration Results on EuRoC Dataset

To assess our approach in offline mode, we used the calibration sequences from the EuRoC dataset [Burri et al., 2016] to calibrate the VI-sensor. The calibration process is split into two stages. First, the camera intrinsics and camera extrinsics are estimated. Then in the second stage, only the camera-IMU extrinsics and time-delay are estimated, with the camera intrinsics and camera extrinsics estimated in the first phase fixed.

The results show comparable calibration reprojection errors, where in the camera calibration stage our method obtained an RMSE reprojection error of 0.6042 pixels compared to Kalibr’s 0.6087 pixels, and in the camera-IMU calibration stage the RMSE reprojection errors are 0.5569 pixels and 0.5775 pixels for our method and Kalibr respectively. Figure. 4.7 and Figure. 4.8 report RMSE ATE after running ORBSLAM3 [Campos et al., 2021] on the EuRoC dataset sequences 10 times in Stereo-VO mode and Stereo-VIO mode, respectively. We did not change the ORB-SLAM3 EuRoC configuration that was originally tuned for Kalibr calibration parameters. Both figures show that the calibrations produced by our method yielded better results on most sequences in Stereo-VO mode and all sequences in VIO mode, compared to Kalibr.

To verify our camera-IMU time delay estimation, we assumed the EuRoC dataset [Burri et al., 2016] has a camera-IMU time delay of $\approx 7\mu\text{s}$, as reported in [Nikolic et al., 2014], and perturbed the `imu_april` IMU timestamps with 100ms, 10ms and 1ms time offsets. With our offline camera-IMU calibration *without interactivity* we were able to recover the time offsets 100ms, 9.97ms and 0.987ms respectively, thus showing that our offline camera-IMU calibrator is capable of accurately estimating the camera-IMU time delay.

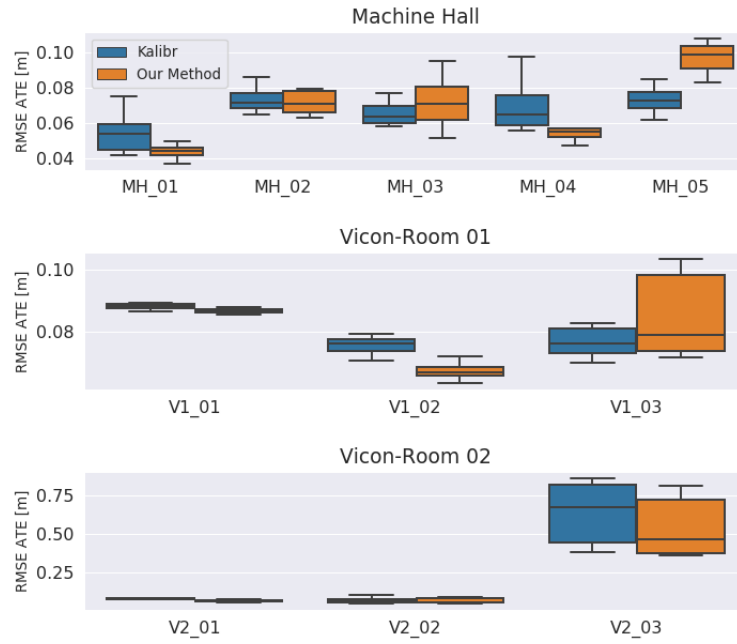


Figure 4.7: Comparison of ORBSLAM3 using calibrations from Kalibr and ours in Stereo-VO mode on EuRoC Dataset

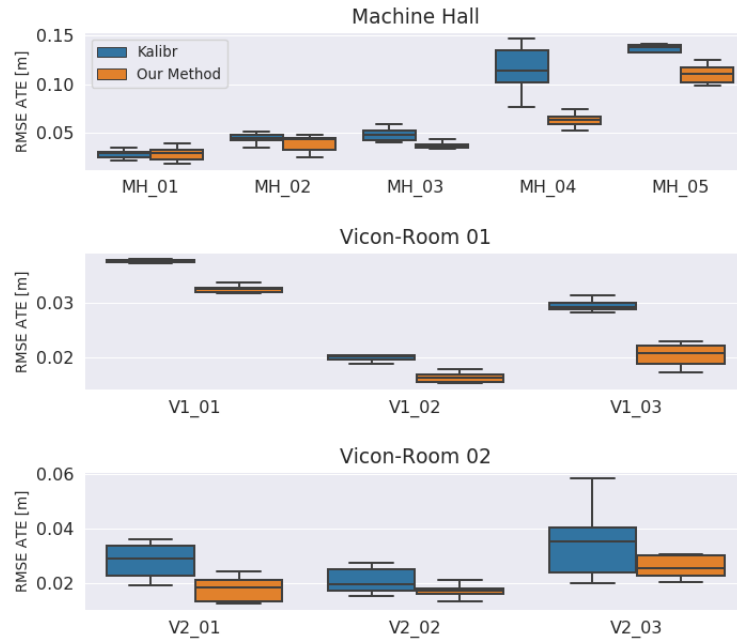


Figure 4.8: Comparison of ORBSLAM3 using calibrations from Kalibr and ours in Stereo-VIO mode on EuRoC Dataset

4.7.2 Trials with Graduate Students

To evaluate our calibration method, we conducted a series of tests involving 16 graduate students to measure the effectiveness of our approach compared to the state-of-the-art calibrator, Kalibr [Furgale et al., 2013]. Our test-subjects were postgraduate students at Imperial College London. Of the 16 students, 4 reported some previous experience with camera calibration, and only 2 reported some previous experience with camera-IMU calibration. Each participant was asked to calibrate the same Intel RealSense D435i sensor by first collecting two calibration sequences for Kalibr (one for camera calibration and the second for camera-IMU calibration), and then another two with our calibration method.

Because we do not have ground truth for the calibration parameters, we evaluated the estimated calibration parameters by applying them in ORBSLAM3 [Campos et al., 2021] running in odometry mode (with loop-closure switched off) on 10 custom-collected Vicon room sequences where ground-truth poses were recorded with various motions.

Our study shows that novices who have little to no experience in calibrating a VI-sensor can obtain better calibrations using our approach compared to Kalibr. Out of 10 Vicon room sequences, the RMSE ATE error is lowest across all sequences using calibration parameters obtained through our method (see Figure. 4.9). Our calibration parameters also yielded overall smaller RMSE ATE variances, showing more consistent and reliable odometry accuracy, regardless of the experience of calibration users. The estimated camera-IMU time delay with our method is $3.07 \pm 0.932ms$, and Kalibr’s estimate is $4.81 \pm 0.981ms$. The break-down of the median total time taken to calibrate the VI-sensor between Kalibr and our method is shown in Figure. 4.10, where our method’s median is 381.11 seconds compared to Kalibr’s 455.44 seconds. It is worth noting even though our method spends more time collecting camera data, the number of *informative* views used by both methods were similar where Kalibr used on average 117 views compared to 114 views with our method.

In addition to showing that our method yields better SLAM results and calibrations faster, by inspecting the Shannon Entropy of the calibration parameters, a metric used to

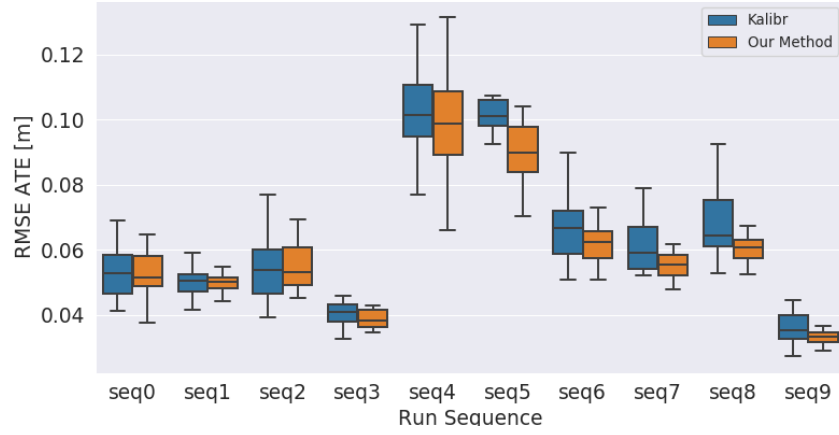


Figure 4.9: Comparing calibrations obtained by graduate students across 10 different evaluation VICON room sequences by running ORBSLAM3 in odometry mode

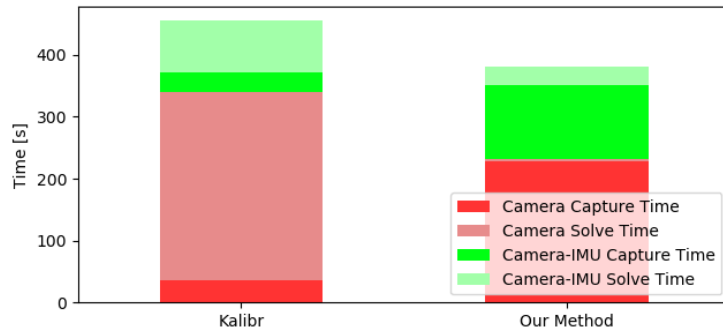
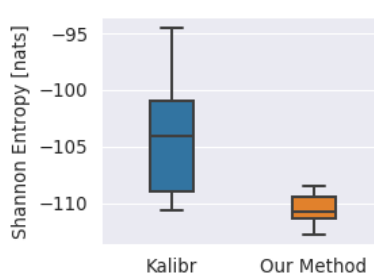
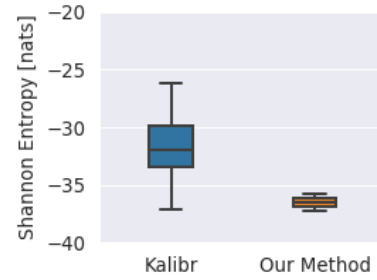


Figure 4.10: Median total calibration time per person in seconds



(a) Camera Calibration



(b) Camera-IMU Calibration

Figure 4.11: Shannon Entropy of camera and camera-IMU calibration across all calibrations

measure the uncertainty of information content [Cover and Thomas, 2006], we also observe a lower entropy (more certainty) with our method compared to Kalibr (see Figure. 4.11a

4. *Accurate And Interactive Visual-Inertial Calibration with Next-Best-View and Next-Best-Trajectory*

Table 4.1: Comparing Camera Parameters Between Kalibr and Our Method

Parameter	Kalibr	Our Method
C_1		
fx [px]	385.3762 ± 1.1221	$384.7552 \pm \mathbf{0.3468}$
fy [px]	385.5533 ± 1.1358	$384.9367 \pm \mathbf{0.3602}$
cx [px]	325.2170 ± 0.5125	$324.9292 \pm \mathbf{0.1393}$
cy [px]	239.0350 ± 0.4373	$238.6604 \pm \mathbf{0.1766}$
k1	0.0056 ± 0.0022	$0.0059 \pm \mathbf{0.0005}$
k2	-0.0024 ± 0.0023	$-0.0027 \pm \mathbf{0.0006}$
p1	0.0001 ± 0.0003	$-0.0002 \pm \mathbf{0.0001}$
p2	0.0007 ± 0.0004	$0.0006 \pm \mathbf{0.0001}$
C_2		
fx [px]	385.3451 ± 1.1310	$384.6959 \pm \mathbf{0.3608}$
fy [px]	385.5005 ± 1.1368	$384.8645 \pm \mathbf{0.3827}$
cx [px]	324.2293 ± 0.4539	$323.8414 \pm \mathbf{0.1975}$
cy [px]	239.0210 ± 0.4635	$238.5976 \pm \mathbf{0.1934}$
k1	0.0056 ± 0.0022	$0.0063 \pm \mathbf{0.0005}$
k2	-0.0022 ± 0.0020	$-0.0030 \pm \mathbf{0.0006}$
p1	0.0001 ± 0.0003	$-0.0002 \pm \mathbf{0.0001}$
p2	0.0003 ± 0.0004	$0.0003 \pm \mathbf{0.0002}$

and Figure. 4.11b). This means that our method can successfully guide a novice to collect a more informative calibration dataset for a good calibration.

Lastly, the calibration parameters estimated have lower variance with our method compared to Kalibr. Table. 4.1 and Table. 4.2 lists the camera parameters C_1 and C_2 , as well as camera-IMU extrinsics T_{SC_1} . Both tables show the optimized parameter standard deviations being smaller in all cases with our method. Further, by inspecting the Shannon Entropy, a metric used to measure the uncertainty of information content, we observed a lower entropy (degree of surprise) with our method compared to Kalibr (see Figure. 4.11a and Figure. 4.11b).

Table 4.2: Comparing Extrinsic Parameters Between Kalibr and Our Method

Parameter	Kalibr	Our Method
$T_{C_1C_2}$		
x [m]	$0.0498 \pm 4.87 \times 10^{-5}$	$0.0497 \pm 2.02 \times 10^{-5}$
y [m]	$0.0001 \pm 1.90 \times 10^{-5}$	$0.0001 \pm 1.97 \times 10^{-5}$
z [m]	$0.0002 \pm 6.36 \times 10^{-5}$	$0.0002 \pm 3.88 \times 10^{-5}$
roll [deg]	0.0048 ± 0.0109	0.0102 ± 0.0097
pitch [deg]	-0.1294 ± 0.0225	-0.1391 ± 0.0120
yaw [deg]	0.0031 ± 0.0037	0.0049 ± 0.0021
T_{SC_1}		
x [m]	$-0.0061 \pm 2.25 \times 10^{-3}$	$-0.0066 \pm 2.07 \times 10^{-3}$
y [m]	$0.0006 \pm 3.61 \times 10^{-3}$	$0.0031 \pm 2.03 \times 10^{-3}$
z [m]	$0.0187 \pm 2.45 \times 10^{-3}$	$0.0191 \pm 1.61 \times 10^{-3}$
roll [deg]	0.1502 ± 0.3158	0.0627 ± 0.0727
pitch [deg]	-0.0438 ± 0.2196	-0.0712 ± 0.0932
yaw [deg]	-1.1223 ± 0.3196	-0.9361 ± 0.0654

4.8 Conclusions

The success of SoTA computer-vision and state-estimation algorithms often hinges on good VI calibrations. However, collecting high-quality VI calibration data is not trivial, especially since most existing calibration tools do not provide an interactive live feedback to the user which ultimately increases the risk of poor calibrations. In this work, we have introduced a novel visual-inertial calibration guidance system to provide real-time NBV and NBT suggestions to guide users in collecting informative calibration data. It achieves competitive calibration results against the SoTA offline calibrator, Kalibr [Furgale et al., 2013], and produces faster, more accurate and more reliable calibrations for existing SoTA visual and VI SLAM systems, even when used by novices, based on our user study ($n = 16$).

Towards More Robust Visual-Inertial Odometry with a Gimbal

Contents of Chapter

5.1	Introduction	105
5.2	Related Work	108
5.3	System Overview	110
5.4	Gimbal Calibration	114
5.4.1	States	115
5.4.2	Calibration Formulation	115
5.5	Experiments	117
5.5.1	Gimbal Calibration Results	118
5.5.2	Body-Fixed vs Gimbal Stabilised VIO	118
5.6	Conclusions	124

5.1 Introduction

Visual-Inertial (VI) sensors have gained popularity in robotics, augmented reality (AR), virtual reality (VR), and autonomous vehicles. Visual-inertial sensors leverage complementary strengths of visual and inertial measurements to achieve accurate state-estimation. In addition, VI-sensors can be designed to be compact and lightweight, making them suitable for integration into various devices, including drones and autonomous vehicles. In summary, the popularity of VI-sensors can be attributed to their ability to provide accurate, robust, low-latency state-estimation across a wide range of applications.

The prevalent application of VI-sensors involves mounting them directly onto robots or autonomous vehicles. However, a common assumption exists that these body-mounted VI-sensors can always provide sufficient visual and inertial data for accurate and robust visual-inertial odometry (VIO) operation. During dynamic movements, the robot may experience significant vibrations, leading to blurred visual data and noisy inertial measurements. Consequently, these factors can cause the VIO system to fail and lose tracking. Furthermore, cameras facing areas lacking visual features, such as blank walls, can also contribute to tracking failures.

Recently, advances in using deep learning for image deblurring, inertial-odometry, and state-estimation has made significant progress. However, these methods come with several disadvantages, namely deep learning models can be computationally intensive, often requiring substantial resources in terms of processing power and memory. In addition, training these models require large amounts of high-quality training data. Lastly, using complex deep learning models can take a significant amount of time to process.

An argument can be made that degraded sensor measurements represent a fundamental loss of information, a gap that no amount of algorithmic manipulation can recover from. A hardware solution to the aforementioned problem is to stabilise the VI-sensor with a gimbal. Gimbal stabilised VI-sensor gives rise to more stable visual feature tracking and smoother inertial measurements for VIO systems, which theoretically should yield more accurate

and robust state-estimation. A body-fixed VI-sensor on the other hand is often at the mercy of the camera view and motion of the robot, often providing sub-optimal measurement data to the VIO system compared to a gimbal stabilised VI-sensor. This concept can be linked to biological mechanisms in humans and animals, particularly the combination of the head and eye moments that stabilise vision during dynamic motion.

In humans and many animals, the vestibulo-ocular reflex (VOR) is a mechanism that stabilises vision by producing eye moments that compensate for the head movements. VOR works by detecting head moments through the vestibular system (inner ear) and generating compensatory eye movements almost instantaneously. This biological system can be seen as an analog to the gimbal-stabilised VI-sensor. By integrating these ideas, the use of a gimbal stabilised VI-sensor can be seen as an engineering approach inspired by biological systems to achieve more robust and accurate state estimation in VIO systems. The gimbal helps isolate the sensor from unwanted motion and vibrations, akin to how the VOR and independent head and eye movements help maintain stable vision in dynamic situations.

In this chapter, we aim to demonstrate the advantages of using a gimbal-stabilized VI-sensor for visual-inertial odometry (VIO) in comparison to a body-fixed VI-sensor. Our approach involves a custom-designed sensor rig featuring two VI-sensors: one stabilized with a 3-axis gimbal and the other rigidly mounted to the rig. To ensure fair comparisons, we hardware-trigger both sensors to capture images simultaneously. We then perform multiple indoor visual-inertial measurements in a motion capture (MOCAP) environment to obtain ground truth data. By analyzing the VIO trajectories estimated by both the body-fixed and gimbal-stabilized sensors, we aim to highlight the performance differences between the two configurations.

Our contributions are as follows:

- Designed and built a unique dual VI-sensor rig with one sensor rigidly mounted and the other stabilized by a 3-axis gimbal.
- Conducted indoor experiments in MOCAP environment to gather ground truth data and compared the VIO trajectories estimated by the body-fixed and gimbal-stabilised VI sensors.

The remainder of this chapter is organised as follows: Section 5.2 provides an overview of related work. In Section 5.3, we describe our system, followed by details on the calibration of the gimbal system in Section 5.4. Finally, Section 5.5 presents our experimental results, comparing the VIO trajectory error between a static and gimbal stabilised VI-sensor.

5.2 Related Work

The Role of Gimbal Systems in Aerial Robotics Gimbal systems allow perception sensors, such as cameras and laser scanners, to operate independently from the main body of a robotic platform. This decoupling enables a robot to move in one direction while its sensors simultaneously point or track objects in another. This capability is especially valuable for aerial robots, including fixed-wing and multi-rotor micro aerial vehicles (MAVs). Early examples include [Quigley et al., 2005], who presented a field-tested fixed-wing MAV equipped with a two-axis gimbal camera for target tracking and continuous surveillance. [Huh et al., 2013] developed a camera and laser scanner navigation system for MAVs, where a pan-tilt gimbal mechanism actuates the laser to extract environmental feature points in order to estimate the MAV's state with an Extended Kalman Filter. [Briod et al., 2014] introduced an innovative 3-axis gimbal design that stabilizes an inner frame housing the propulsion system, flight control, and a camera sensor, all enclosed within a protective cage. More recently, [Borowczyk et al., 2017] utilized a gimbal camera system on an MAV to autonomously land on a high-speed ground vehicle. These examples highlight the broad applicability and significance of gimbal systems for aerial robotics.

VIO Visual-Inertial Odometry (VIO) has been extensively studied in recent years. State-of-the-art VIO algorithms such as OKVIS 2 [Leutenegger, 2022], VINS-MONO [Qin et al., 2018], Open-VINS [Geneva et al., 2020], and ROVIO [Bloesch et al., 2017], have demonstrated accuracy and robustness across a wide range of robotic scenarios. However, these methods do not inherently address situations where measurement data is sub-optimal for state estimation and cannot actively prevent sub-optimal measurement acquisition and the related deterioration of state estimation.

Calibration Before incorporating gimbal-stabilized vision sensors for state estimation, it is crucial to first estimate the forward kinematic parameters of the gimbal system. Camera-based kinematic calibration for a gimbal involves capturing relative measurements between a camera mounted on the gimbal and a fiducial target in the environment. In the work by [Das and Waslander, 2016], a calibration approach was introduced that parameterizes

the gimbal's actuated mechanism using the Denavit-Hartenberg (DH) convention. This approach determines both the forward kinematic parameters of the gimbal system and the extrinsic transformations, including the end-effector-to-camera (hand-eye) and base-frame-to-camera parameters. [Rebello et al., 2017] extended the calibration approach by using an information theoretic Next-Best-View policy to determine the optimal actuator inputs for estimating the calibration parameters. A subsequent study by [Choi et al., 2018] further relaxed the reliance on gimbal motor encoders for obtaining gimbal joint angles. The proposed calibration approach was validated both in simulation and on a multi-rotor MAV equipped with a 2-DOF gimbal-mounted camera sensor.

Deep Learning Methods There has been a growing interest in using deep learning for solving state-estimation tasks. Frameworks such as CNN-SLAM [Tateno et al., 2017], DroidSLAM [Teed and Deng, 2021], TLIO [Liu et al., 2020] have achieved state of the art performances, and have been shown to outperform classical methods. The caveat with these approaches, however, is that they often rely on a large amount of high quality training data specific to the device or robot they are trained on. Additionally, the computation requirement to run these deep learning models is not often practical on small robotic platforms.

Active Vision Methods The work of [Davison and Murray, 1998] was one of the first frameworks that chooses which features to track at a certain time to best improve its position knowledge, by changing its fixation point and when to search for new features with the aim of maintaining a useful 3D map. More recently, [Patel et al., 2019] developed a pointing algorithm by controlling a gimbal stabilised stereo camera mounted on a multi-rotor MAV for localisation tasks, and demonstrated how localisation performance and robustness can be improved by changing the gimbal stabilised stereo camera's view point. [Greeff et al., 2020] developed a perception-aware flatness-based model predictive controller (MPC) for a multi-rotor MAV that accounts for its effect on visual localisation, and showed in simulation is able to achieve faster flights while guaranteeing localisation compared to perception agnostic flight controllers.

In this chapter, we do not delve into developing a gimbal pointing algorithm to determine when the gimbal camera should fixate or change its viewpoint to improve positional accuracy, as explored by [Davison and Murray, 1998], [Patel et al., 2019] and [Greeff et al., 2020]. Instead, we demonstrate through empirical experimental data that a passive gimbal-stabilized VI-sensor outperforms a rigidly mounted VI-sensor on the sensor rig body. While this concept is intuitively clear, to the best of our knowledge, there has been no published research with empirical data supporting this claim.

5.3 System Overview

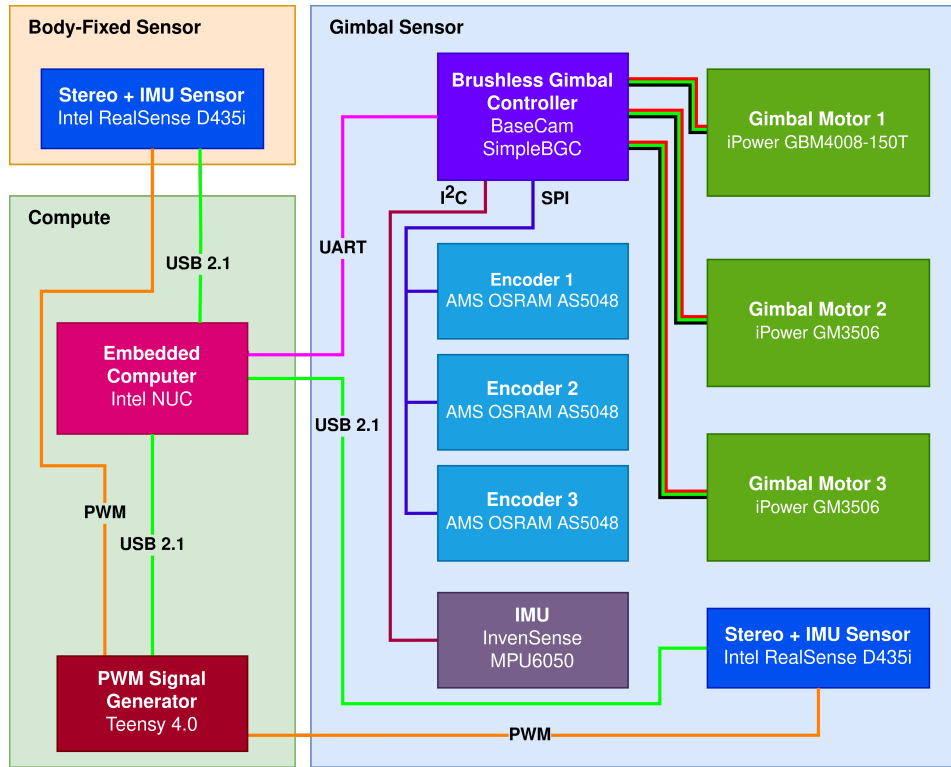
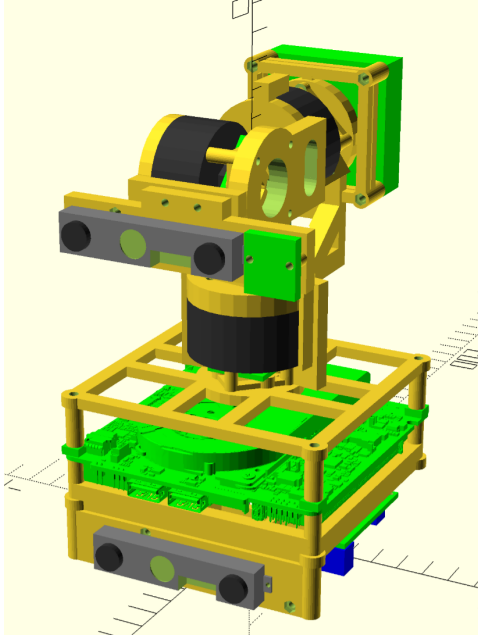
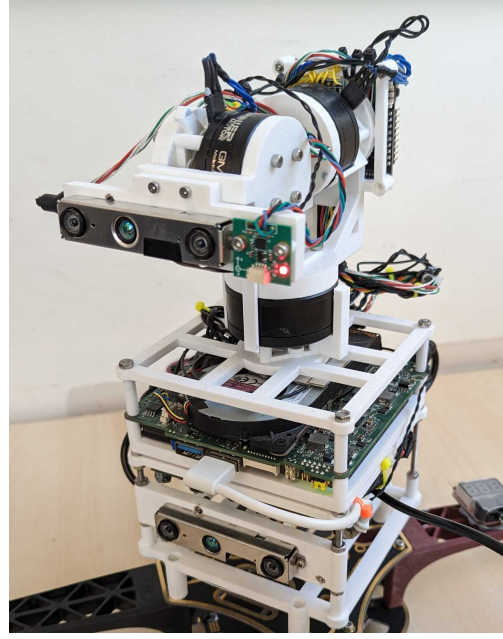


Figure 5.1: Sensor rig system overview

An overview of the proposed system is illustrated in Figure. 5.1. The sensor rig comprises of two Intel D435i VI-sensors: one body-fixed and the other stabilized with a 3-axis gimbal. The gimbal-stabilized sensor is actuated by three brushless gimbal motors controlled via a BaseCam SimpleBGC brushless gimbal controller. Orientation and gimbal joint angles



(a) CAD render



(b) Assembly

Figure 5.2: Sensor rig

are measured using an Invensense MPU6050 IMU and AMS OSRAM AS5048 magnetic rotary encoders. The CAD design and assembled sensor rig are shown in Figure. 5.2. The gimbal system was designed to maintain target orientation by compensating for external disturbances, whether operated by hand or attached to a multi-rotor MAV, with a response time generally in the range of 2-10 ms, supported by an IMU often operating at 250 - 1000 Hz.

Both VI-sensors interface with an Intel NUC computer and are hardware-triggered by a PWM signal generated from a Teensy 4.0 Arduino board. This setup allows us to capture visual and inertial measurements from both sensors simultaneously and independently, enabling a comparison of the VIO trajectories estimated by each sensor. The hardware triggering scheme, illustrated in Figure. 5.3, shows that the PWM signal synchronizes only the cameras of both VI-sensors. IMU measurements are resampled to the image frames through linear interpolation. The coordinate frames of the whole sensor rig are depicted in Figure. 5.4 and Figure. 5.5.

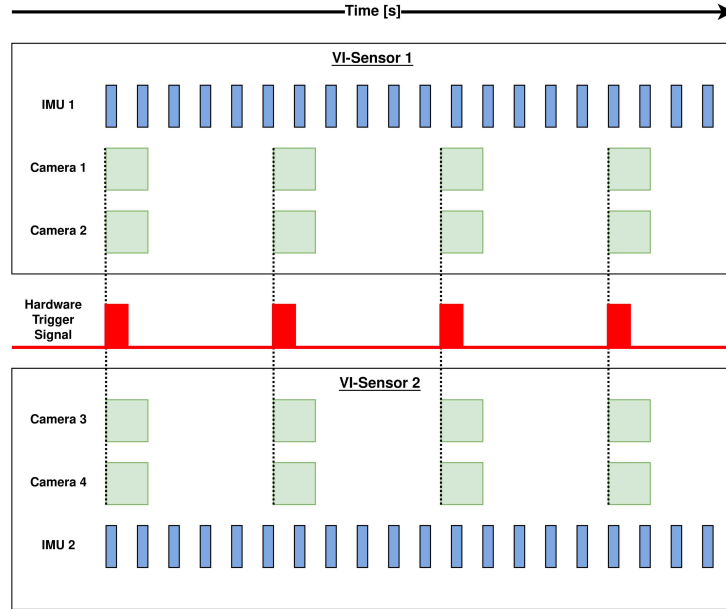


Figure 5.3: Sensor synchronization

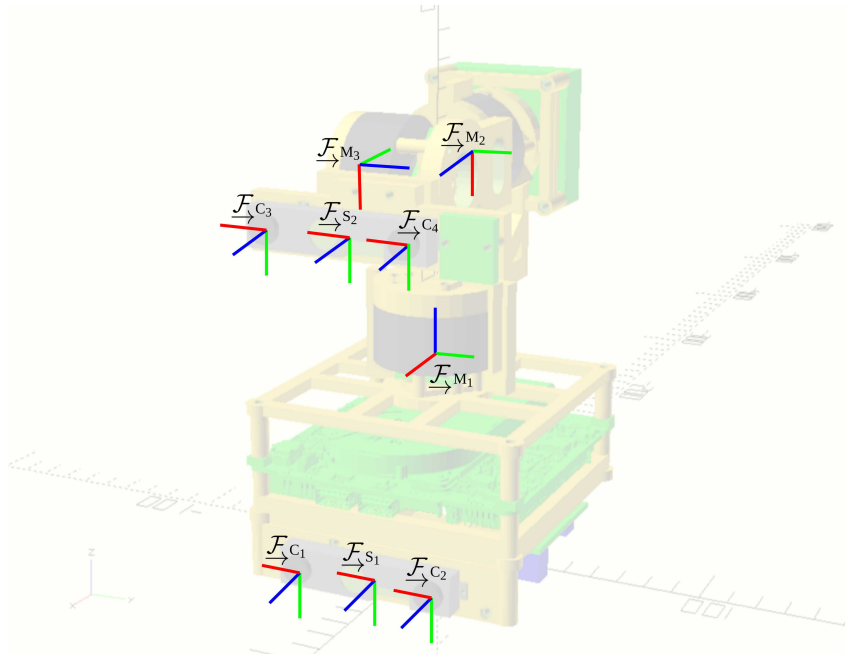
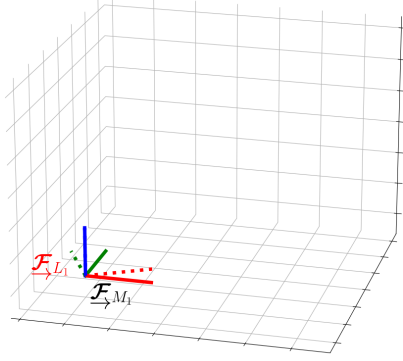
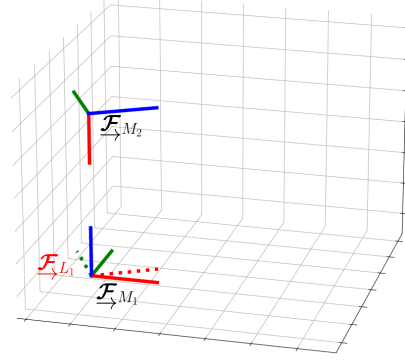


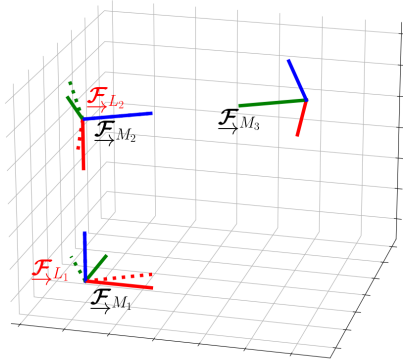
Figure 5.4: Sensor rig frame diagram



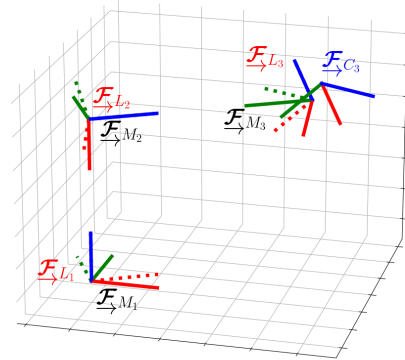
(a) $T_{M_1 L_1}(\theta_1)$: transformation between gimbal motor 1 $\vec{F}_{M_1}$ and gimbal link 1 $\vec{F}_{L_1}$ as a function of joint angle $\theta_1 = 30^\circ$.



(b) $T_{L_1 M_2}$: transformation between gimbal link 1 $\vec{F}_{L_1}$ and gimbal motor 2 $\vec{F}_{M_2}$.



(c) $T_{L_2 M_2}(\theta_2)$: transformation between gimbal link 2 $\vec{F}_{L_2}$ and gimbal motor 2 $\vec{F}_{M_2}$ as a function of joint angle $\theta_2 = 25^\circ$. $T_{L_2 M_3}$: transformation between gimbal link 2 $\vec{F}_{L_2}$ and gimbal motor 3 $\vec{F}_{M_3}$.



(d) $T_{M_3 L_3}(\theta_3)$: transformation between gimbal motor 3 $\vec{F}_{M_3}$ gimbal link 3 $\vec{F}_{L_3}$ as a function of joint angle $\theta_3 = 30^\circ$. $T_{L_3 C_3}$: transformation between gimbal link 3 $\vec{F}_{L_3}$ and camera 3 $\vec{F}_{C_3}$.

Figure 5.5: Gimbal kinematic chain from gimbal motor 1 $\vec{F}_{M_1}$ to camera 3 $\vec{F}_{C_3}$. For each coordinate frame the x, y, and z-axes are denoted with a red, green and blue line respectively. The link frames $\vec{F}_{L_1}$, $\vec{F}_{L_2}$ and $\vec{F}_{L_3}$ coincide with the motor frames $\vec{F}_{M_1}$, $\vec{F}_{M_2}$ and $\vec{F}_{M_3}$ except for the rotation around the z-axis, which is perturbed by a joint angle $\theta_m, \forall m \in \{1, 2, 3\}$. The gimbal joint angles in the figure are $\theta_1 = 30^\circ$ (yaw), $\theta_2 = 25^\circ$ (roll), and $\theta_3 = 30^\circ$ (pitch) respectively.

5.4 Gimbal Calibration

In order to use and compare the system described in Sec. 5.3, the camera intrinsics as well as the spatial and temporal relationship between the cameras and IMU in the Body-Fixed and Gimbal sensor need to be resolved accurately. The following lists the calibration parameters needed to be estimated:

- **Camera-IMU Calibration** (Chapter. 4)
 - Camera intrinsics: C_1, C_2, C_3 , and C_4
 - Camera extrinsics: $\mathbf{T}_{C_1C_2}$ and $\mathbf{T}_{C_3C_4}$
 - Camera-IMU extrinsics: $\mathbf{T}_{C_1S_1}$ and $\mathbf{T}_{C_3S_2}$
 - Camera-IMU time-delay: $t_{d,0}$ and $t_{d,1}$
- **Gimbal Calibration** (Sec. 5.4)
 - 3-axis gimbal joint angles: θ_1, θ_2 and θ_3
 - Reference camera to gimbal base extrinsic: $\mathbf{T}_{C_1M_1}$
 - Gimbal links: $\mathbf{T}_{L_1M_2}, \mathbf{T}_{L_2M_3}$
 - Camera extrinsic: $\mathbf{T}_{L_3C_3}$

where the camera intrinsics and extrinsics, camera-IMU extrinsics and camera-IMU time delay parameters can be estimated using the pipeline outlined in Chapter 4. In the following sections we will describe how the 3-axis gimbal joint angles, gimbal extrinsics as and gimbal links are estimated.

5.4.1 States

For the gimbal calibration problem, the states to be estimated consists of the reference camera poses relative to the fiducial target coordinate frame $\underline{F}_F$ as $\mathbf{x}_{FC_1}$, extrinsic between reference camera 1 and gimbal motor 1, $\mathbf{x}_{C_1M_1}$, gimbal links, $\mathbf{x}_{L_1M_2}$, $\mathbf{x}_{L_2M_3}$, gimbal joints $\mathbf{x}_\theta$, and extrinsic between gimbal motor 3 and camera 3, $\mathbf{x}_{L_3C_3}$, of the form:

$$\mathbf{x}_{FC_1} = \begin{bmatrix} {}_F\mathbf{r}_{FC_1}^\top & \mathbf{q}_{FC_1}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \quad (5.1)$$

$$\mathbf{x}_{C_1M_1} = \begin{bmatrix} {}_{C_1}\mathbf{r}_{C_1M_1}^\top & \mathbf{q}_{C_1M_1}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \quad (5.2)$$

$$\mathbf{x}_{L_1M_2} = \begin{bmatrix} {}_{L_1}\mathbf{r}_{L_1M_2}^\top & \mathbf{q}_{L_1M_2}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \quad (5.3)$$

$$\mathbf{x}_{L_2M_3} = \begin{bmatrix} {}_{L_2}\mathbf{r}_{L_2M_3}^\top & \mathbf{q}_{L_2M_3}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3, \quad (5.4)$$

$$\mathbf{x}_{\theta,k} = \begin{bmatrix} \theta_{1,k} & \theta_{2,k} & \theta_{3,k} \end{bmatrix}^\top \in \mathbb{R}^3, \quad (5.5)$$

$$\mathbf{x}_{L_3C_3} = \begin{bmatrix} {}_{M_3}\mathbf{r}_{L_3C_3}^\top & \mathbf{q}_{M_2C_3}^\top \end{bmatrix}^\top \in \mathbb{R}^3 \times \mathcal{S}^3. \quad (5.6)$$

The full state vector for gimbal calibration thus becomes,

$$\mathbf{x} = \left[\underbrace{\mathbf{x}_{FC_1,1}^\top \dots \mathbf{x}_{FC_1,k}^\top}_{\text{Reference Camera 1 Poses}} \underbrace{\mathbf{x}_{C_1M_1}^\top}_{\text{Gimbal Extrinsic}} \underbrace{\mathbf{x}_{L_1M_2}^\top \mathbf{x}_{L_2M_3}^\top}_{\text{Gimbal Links}} \underbrace{\mathbf{x}_{\theta,1}^\top \dots \mathbf{x}_{\theta,k}^\top}_{\text{Gimbal Joint Angles}} \underbrace{\mathbf{x}_{L_3C_3}^\top}_{\text{Camera Extrinsic}} \right]^\top. \quad (5.7)$$

The minimal error state vectors for $\mathbf{x}_{FC_1}$, $\mathbf{x}_{C_1M_1}$, $\mathbf{x}_{L_1M_2}$, $\mathbf{x}_{L_2M_3}$, $\mathbf{x}_{\theta,k}$, and $\mathbf{x}_{M_2C_3}$ are:

$$\delta\chi_{FC_1} = \begin{bmatrix} \delta {}_F\mathbf{r}_{FC_1}^\top & \delta\alpha_{FC_1}^\top \end{bmatrix}^\top \in \mathbb{R}^6, \quad (5.8)$$

$$\delta\chi_{C_1M_1} = \begin{bmatrix} \delta {}_{C_1}\mathbf{r}_{C_1M_1}^\top & \delta\alpha_{C_1M_1}^\top \end{bmatrix}^\top \in \mathbb{R}^6, \quad (5.9)$$

$$\delta\chi_{L_1M_2} = \begin{bmatrix} \delta {}_{L_1}\mathbf{r}_{L_1M_2}^\top & \delta\alpha_{L_1M_2}^\top \end{bmatrix}^\top \in \mathbb{R}^6, \quad (5.10)$$

$$\delta\chi_{L_2M_3} = \begin{bmatrix} \delta {}_{L_2}\mathbf{r}_{L_2M_3}^\top & \delta\alpha_{L_2M_3}^\top \end{bmatrix}^\top \in \mathbb{R}^6, \quad (5.11)$$

$$\delta\chi_{\theta,k} = \begin{bmatrix} \delta\theta_{1,k} & \delta\theta_{2,k} & \delta\theta_{3,k} \end{bmatrix}^\top \in \mathbb{R}^3, \quad (5.12)$$

$$\delta\chi_{L_3C_3} = \begin{bmatrix} \delta {}_{C_3}\mathbf{r}_{L_3C_3}^\top & \delta\alpha_{L_3C_3}^\top \end{bmatrix}^\top \in \mathbb{R}^6. \quad (5.13)$$

5.4.2 Calibration Formulation

To estimate the gimbal calibration parameters we used a nonlinear least squares framework to minimize the cost function, J , containing reprojection errors, $\mathbf{e}_r$, joint angle errors, $\mathbf{e}_\theta$,

and their respective weight matrix and scalar, $\mathbf{W}_r$ and w_θ . The cost function has the form:

$$J(\mathbf{x}) = \underbrace{\frac{1}{2} \sum_{k=1}^K \sum_{i \in \mathcal{J}(k)} \mathbf{e}_r^{i,k\top} \mathbf{W}_r^{i,k} \mathbf{e}_r^{i,k}}_{\text{Reprojection Errors}} + \underbrace{\frac{1}{2} \sum_{k=1}^K \sum_{j=1}^3 w_\theta^{j,k} \|\mathbf{e}_\theta^{j,k}\|^2}_{\text{Gimbal Joint Errors}} \quad (5.14)$$

where k denotes the camera frame index, and i denotes the fiducial target corner index. Finally, $\mathcal{J}(k)$ denotes the set of observable fiducial corner indices in the k^{th} camera frame.

The reprojection error, $\mathbf{e}_r$, was used to estimate the gimbal kinematics,

$$\mathbf{e}_r^{i,k} = \tilde{\mathbf{z}}^{i,k} - \boldsymbol{\pi} \left(\mathbf{T}_{C_1C_3}^{-1} \mathbf{T}_{FC_1}^{-1} {}_F\mathbf{r}_{FF_i} \right) \quad (5.15)$$

$$\mathbf{T}_{C_1C_3}^{-1} = \left(\underbrace{\mathbf{T}_{C_1M_1}}_{\text{Gimbal extrinsic}} \underbrace{\mathbf{T}_{M_1L_1}(\theta_{1,k})}_{\text{Joint 1}} \underbrace{\mathbf{T}_{L_1M_2}}_{\text{Link 1}} \underbrace{\mathbf{T}_{M_2L_2}(\theta_{2,k})}_{\text{Joint 2}} \underbrace{\mathbf{T}_{L_2M_3}}_{\text{Link 2}} \underbrace{\mathbf{T}_{M_3L_3}(\theta_{3,k})}_{\text{Joint 3}} \underbrace{\mathbf{T}_{L_3C_3}}_{\text{Camera extrinsic}} \right)^{-1} \quad (5.16)$$

whereby $\boldsymbol{\pi}(\cdot)$ denotes the camera projection and distortion model. The camera model takes as an input the i^{th} fiducial corner, ${}_F\mathbf{r}_{FF_i}$, fiducial-camera pose, $\mathbf{T}_{FC_1}$, and the gimbal kinematics $\mathbf{T}_{C_1C_3}^{-1}$ that transforms observations from the reference camera 1 to the gimbal camera C_3 through the gimbal extrinsic, $\mathbf{T}_{C_1M_1}$, and a series of gimbal joint transforms, $\mathbf{T}_{M_mL_m}(\theta_m), \forall m \in \{1, 2, 3\}$. Here, $\mathbf{T}_{M_mL_m}(\cdot)$ is a function of the joint angle, $\theta_m, \forall m \in \{1, 2, 3\}$ forming a homogeneous transformation matrix that rotates around its z-axis. The gimbal links, $\mathbf{T}_{L_1M_2}$ and $\mathbf{T}_{L_2M_3}$, and camera extrinsic, $\mathbf{T}_{L_3C_3}$, are also included in this transformation. Lastly, $\tilde{\mathbf{z}}^{i,k}$ is the observed fiducial corner measurement of the i^{th} fiducial corner at the k^{th} camera frame. In addition to the reprojection error we used the joint error, $\mathbf{e}_\theta$, to estimate the gimbal joint angles,

$$\mathbf{e}_\theta^{j,k} = \tilde{\theta}_{j,k} - \theta_{j,k}, \quad (5.17)$$

where $\tilde{\theta}_{j,k}$ and $\theta_{j,k}$ are the measured and estimated j^{th} joint angle in the k^{th} camera frame.

5.5 Experiments

In order to compare the difference between the body-fixed and gimbal stabilised VI-sensor configuration, we collected 6 trajectory patterns, where 2 of them involve the sensor rig traversing in a circle, 2 trajectories traversing in a figure 8 pattern, and with the last 2 trajectories traversing in a random walk trajectory pattern. To keep the comparisons faithful and mitigating the effects of a “better” VI-sensor, we repeated the 6 trajectory patterns again but with the VI-sensors swapped. In summary, this gives rise to two sets of 6 trajectories where the two VI-sensors are swapped after the first set.

To evaluate whether a stabilised visual-inertial sensor yields better odometry results, we first calibrated the sensor rig described in Sec. 5.3, then we conducted a number of experiments under motion capture system for ground truth. The sensor rig consisting of two Intel RealSense D435i running in Stereo and IMU mode, where both are hardware triggered by a PWM signal generated from a Teensy 4.0 board capturing 640 x 480 resolution images at 15Hz, and both IMUs running at 400Hz. For our purposes, we do not use the RGB rolling shutter sensor, instead we have disabled the IR projector and used the stereo IR global shutter depth sensors as a standard gray-scale stereo camera. All data were captured using an Intel NUC with an Intel Core i7-7567U CPU operating at 3.5 GHz and 16GB RAM running Ubuntu 22.04. The collected data were then processed offline using a laptop with an Intel Core i5-1340P CPU operating at 4.6Ghz with 16GB RAM running Ubuntu 22.04.

For comparison we processed the sequences offline with OKVIS 2 [Leutenegger, 2022] in odometry mode with the data collected from both the body-fixed and gimbal stabilised VI-sensor, where loop-closure and online calibration switched off.

The VIO trajectories estimated by the body-fixed and gimbal stabilised VI-sensor were compared by evaluating the accuracy using the evaluation scheme of [Sturm et al., 2012] with RMSE Absolute Trajectory Error (ATE) by aligning the estimated trajectory with the ground-truth using a SE(3) transformation. The trajectories were evaluated using the Horn’s method [Horn, 1987] to align the estimated and ground-truth trajectory obtained from

motion capture data. For the body-fixed VI-sensor trajectory this is straight forward. The trajectory estimated by the gimbal stabilised VI-sensor, on the other hand, was transformed to the body-fixed VI-sensor using the recorded encoder joint angles from the gimbal system, and the calibrated gimbal kinematic for a fair comparison. This is because while the pose offset between the assigned motion capture body-frame in the sensor rig and the body-fixed VI-sensor is fixed, the offset between the motion capture sensor rig body frame and the gimbal stabilised VI-sensor varies over time.

5.5.1 Gimbal Calibration Results

To calibrate the gimbal kinematics outlined in Sec. 5.4, we first calibrate the VI-sensors independently, resolving the camera intrinsics, camera extrinsics, camera-IMU extrinsics and camera-IMU time delay parameters using methods based on Chapter. 4. Once these parameters were obtained, we placed a calibration target consisting of a grid of AprilTags, at an adequate distance away in front of the sensor rig, and actuated the gimbal to excite the image input, keeping the calibration target stationary, and collected 28 images with a wide variety of gimbal joints for calibration. The result is a RMS reprojection of 0.4899 pixels which indicates a good calibration was achieved.

Figure. 5.6 shows the detected calibration target corners, reprojected onto each camera frame through the optimised gimbal kinematic parameters, where the top row represents the gimbal stabilised VI-sensor, and the bottom row is the body-fixed VI-sensor. The RMS reprojection error in each camera frame is ≈ 0.5 pixels and lower.

5.5.2 Body-Fixed vs Gimbal Stabilised VIO

To compare the gimbal stabilised VI-sensor and body-fixed VI-sensor, we collected two sets of 6 sequences with two different VI-sensor configurations and ran two OKVIS2 [Leutenegger, 2022] independently using visual-measurements from both VI-sensors. The IMU parameters used for OKVIS2 are: $\sigma_a = 2.52 \times 10^{-2} \frac{m}{s^2} \frac{1}{\sqrt{Hz}}$ for the accelerometer noise density, $\sigma_{ba} = 4.0 \times 10^{-2} \frac{m}{s^3} \frac{1}{\sqrt{Hz}}$ for the accelerometer drift noise density, $\sigma_g = 2.78 \times 10^{-3} \frac{rad}{s} \frac{1}{\sqrt{Hz}}$ for

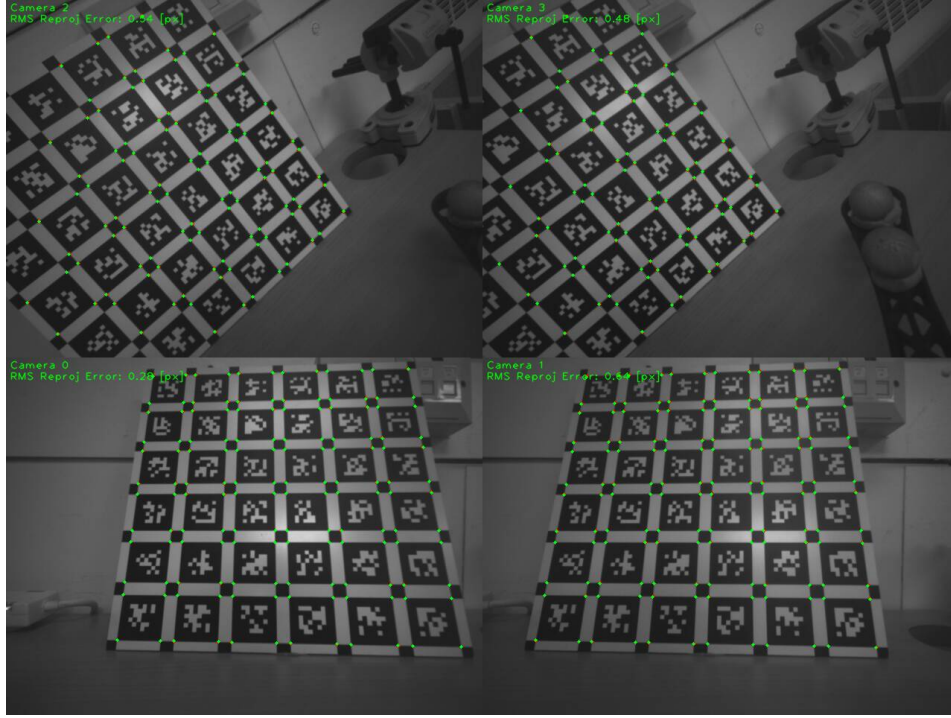


Figure 5.6: Example gimbal kinematics calibration image. (Top row) Gimbal VI-sensor. (Bottom row) Body-fixed VI-sensor. The camera index and RMS reprojection error in each corresponding image frame is labelled in the top left corner.

gyroscope noise density, and $\sigma_{bg} = 8.0 \times 10^{-4} \frac{\text{rad}}{\text{s}^2} \frac{1}{\sqrt{\text{Hz}}}$ gyroscope drift noise density.

Table 5.1 show the Root-Mean-Squared Absolute Trajectory Error (RMS ATE) in meters comparing the VIO trajectory estimated using visual-inertial measurements from the gimbal stabilised VI-sensor and body-fixed sensor against motion-capture measurements as ground-truth. Results show the gimbal stabilised VI-sensor out-performed the body-fixed VI-sensor 5 out of 6 sequences in the set 1, and 6 out of 6 sequences in set 2. Supporting our intuition as to the benefits of using a hardware solution to stabilise visual-inertial measurements for VIO. Figures Figure. 5.7 to 5.18 show the VIO trajectories of the body-fixed VI-sensor vs the gimbal stabilised VI-sensor.

Table 5.1: Root-Mean-Squared Absolute Trajectory Error (RMS ATE) in meters between the body-fixed VI-sensor compared to the gimbal stabilised VI-sensor. There are 3 sequence patterns, they are circle, figure of 8 and random. Each sequence pattern was repeated twice, giving 6 sequences per sensor configuration. In total giving rise to 12 sequences for 2 sensor configurations.

Set 1 Results

Sequence	Body-Fixed Sensor	Gimbal Sensor	Time [s]	Distance [m]
circle-0	0.1179	0.0711	70.28	36.39
circle-1	0.1292	0.1160	96.48	46.80
figure8-0	0.0824	0.0949	105.65	45.12
figure8-1	0.1188	0.1093	148.23	64.96
random-0	0.1131	0.0563	52.46	26.49
random-1	0.0925	0.0597	72.64	37.00

Set 2 Results

Sequence	Body-Fixed Sensor	Gimbal Sensor	Time [s]	Distance [m]
circle-0	0.2431	0.1816	67.79	29.45
circle-1	0.2621	0.1648	100.04	38.66
figure8-0	0.2058	0.2025	110.66	37.52
figure8-1	0.2431	0.2366	95.73	35.43
random-0	0.1370	0.1170	58.39	21.75
random-1	0.0876	0.0779	78.60	33.01

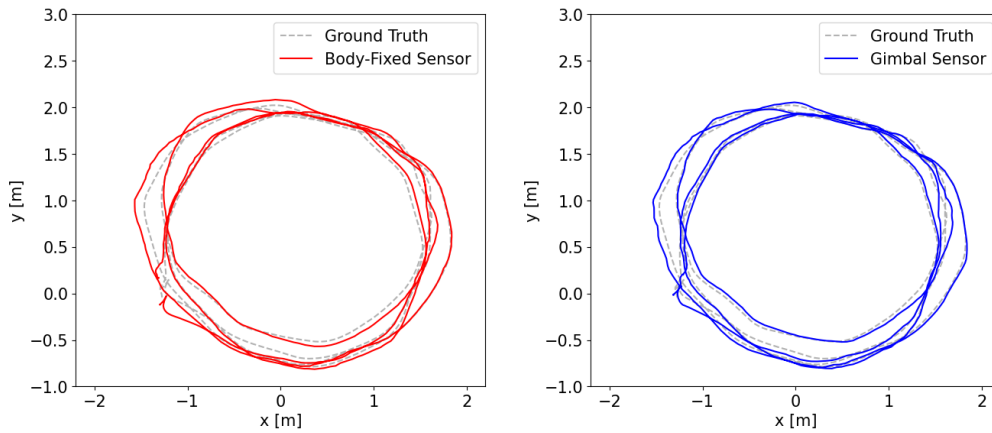


Figure 5.7: Set: 1, Sequence: Circle 0

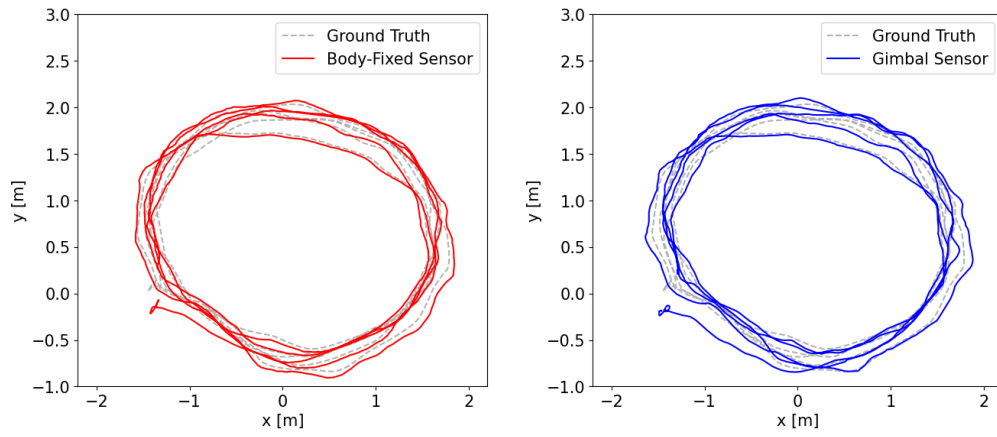


Figure 5.8: Set: 1, Sequence: Circle 1

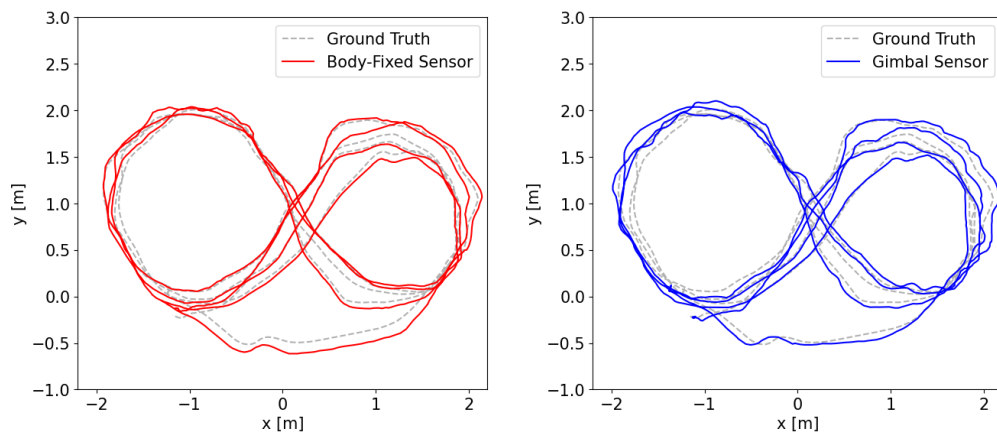


Figure 5.9: Set: 1, Sequence: Figure8 - 0

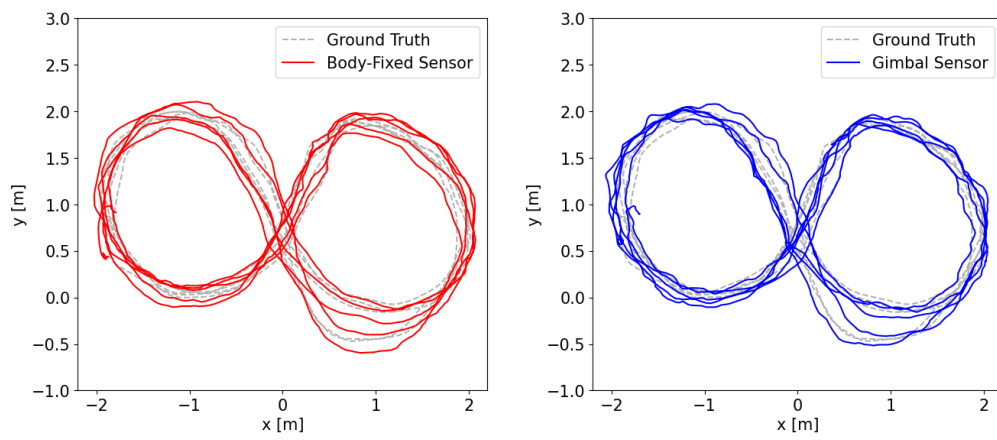


Figure 5.10: Set: 1, Sequence: Figure8 - 1

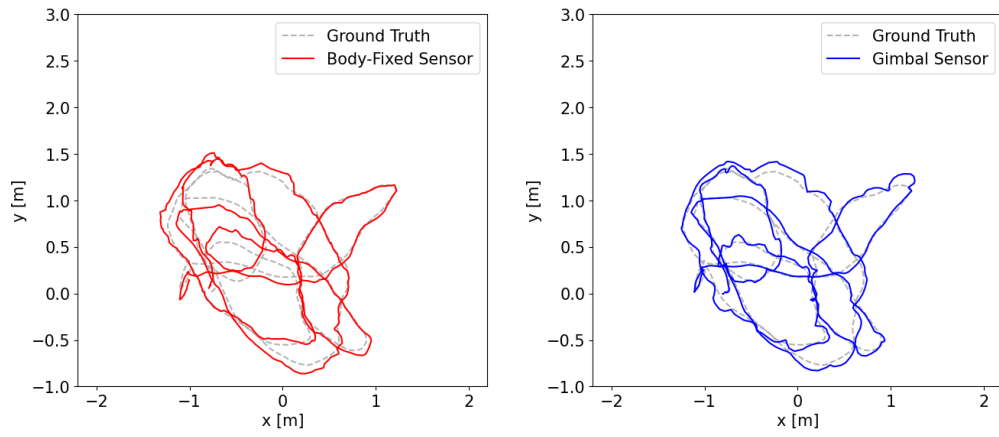


Figure 5.11: Set: 1, Sequence: Random - 0

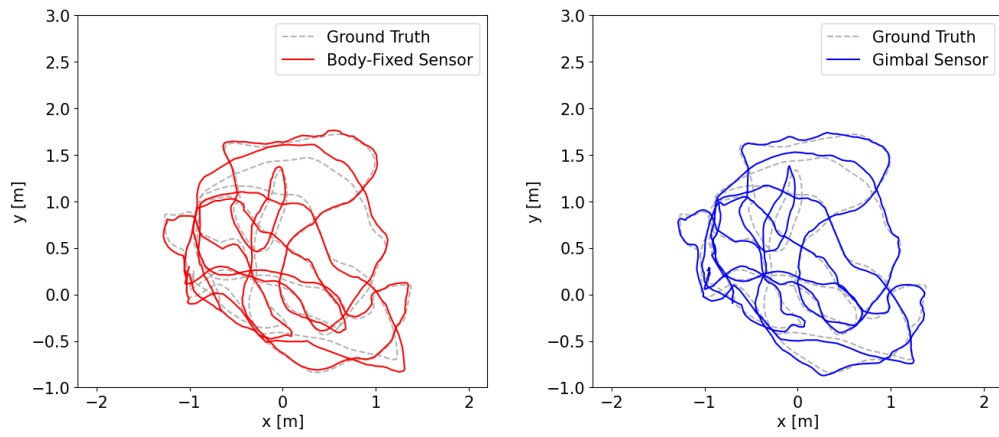


Figure 5.12: Set: 1, Sequence: Random - 1

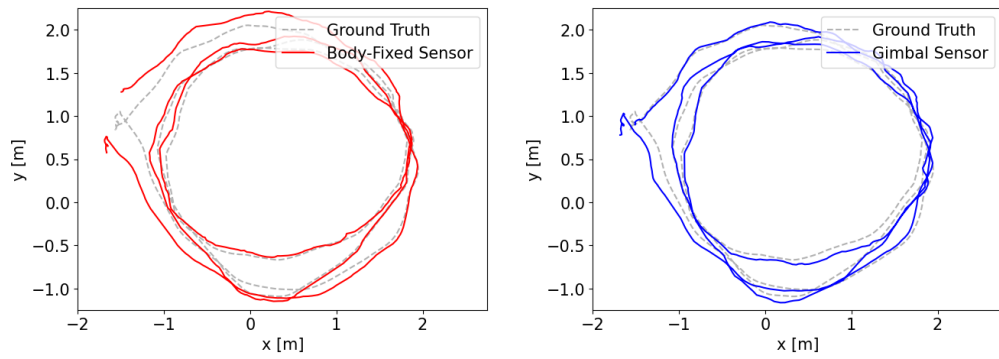


Figure 5.13: Set: 2, Sequence: Circle - 0

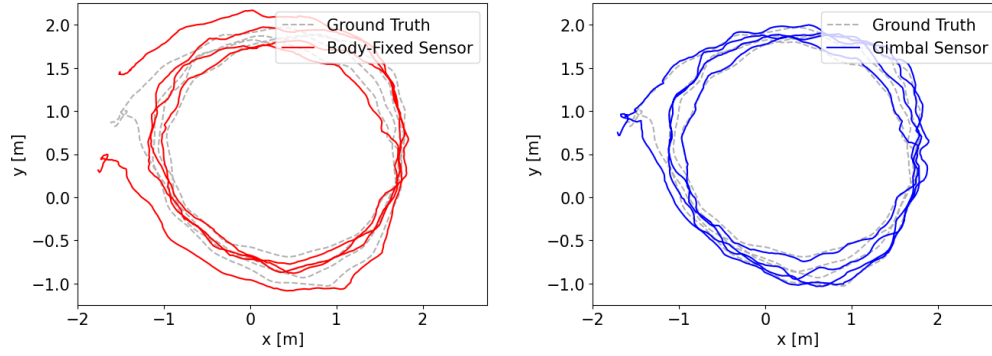


Figure 5.14: Set: 2, Sequence: Circle - 1

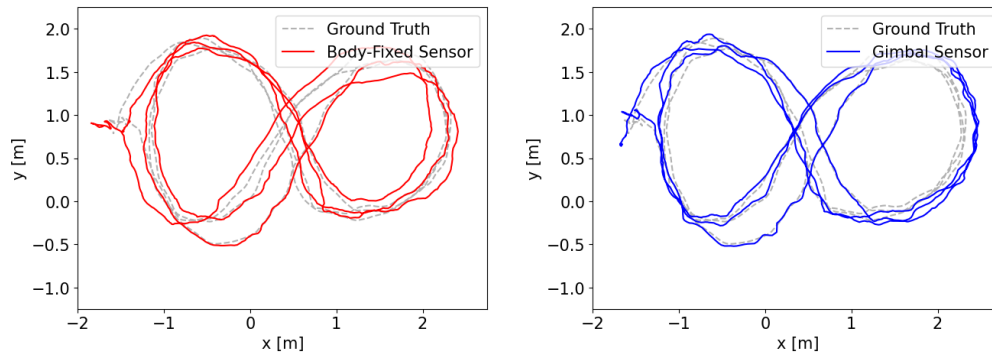


Figure 5.15: Set: 2, Sequence: Figure 8 - 0

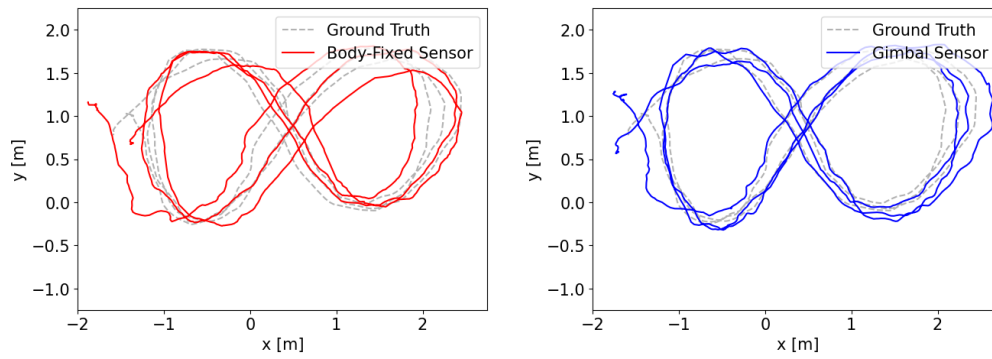


Figure 5.16: Set: 2, Sequence: Figure 8 - 1

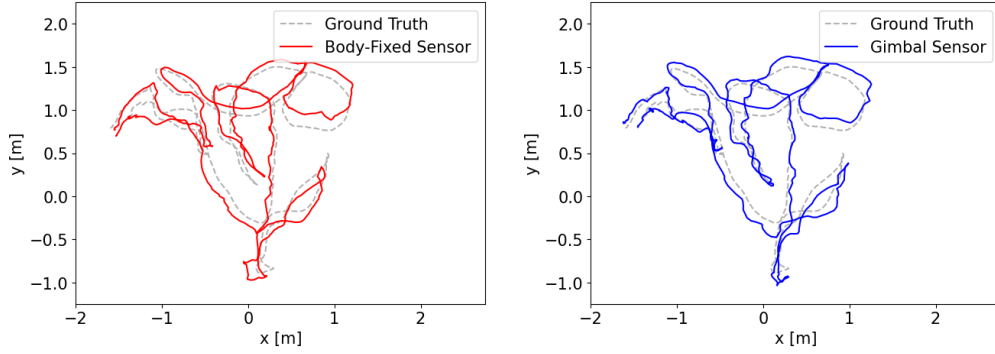


Figure 5.17: Set: 2, Sequence: Random - 0

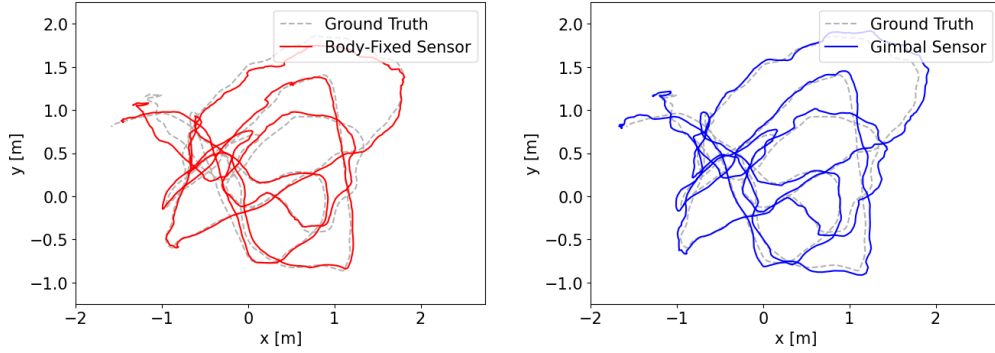


Figure 5.18: Set: 2, Sequence: Random - 1

5.6 Conclusions

In this chapter we have presented our unique dual VI-sensor rig, where we have a 3-axis gimbal stabilised VI-sensor and a body-fixed VI-sensor rigidly mounted onto the body of the sensor rig. We have demonstrated that a gimbal stabilised VI-sensor out performs a body-fixed VI-sensor on 11 out of 12 on our custom collected data sequences, designed to highlight the advantages of using a gimbal.

Although not explored, this provides the foundation for future work into an *active* Visual SLAM framework. Specifically towards answering the question on what should the gimbal camera fixate on in the short term, and deciding when to search for new features for long term tracking in the hopes of decreasing the uncertainty of the position.

Conclusions

6.1 Summary of Results

In this section we present a summary of the contributions and results which have been presented in detail in the previous chapters.

In Chapter 3, we introduced an Aerial Additive Manufacturing (Aerial-AM) framework and evaluated the system by printing a 2.05-meter tall, 0.3-meter diameter cylindrical structure using an aerial, untethered, three-dimensional print system, which we termed a BuilDrone. At each layer, a ScanDrone equipped with an RGB-D sensor performed dense map reconstruction of the printed structure for human-in-the-loop inspection and validation. We detailed our contributions to the trajectory planning for both the BuilDrone and ScanDrone during the construction of the cylindrical structure. Additionally, we explained how we calibrated the extrinsic parameters between the motion capture markers and the RGB-D sensor on the ScanDrone to ensure accurate mapping of the printed structure. Finally, we conducted an analysis of the mapping accuracy using first-order error propagation.

In Chapter 4, we highlighted the popularity of Visual-Inertial (VI) sensors in robotics, self-driving vehicles, and augmented and virtual reality applications. However, in order to use them for any computer vision or state-estimation task, a good calibration is essential.

The challenge is collecting *informative* calibration data in order to render the calibration parameters observable is not trivial for a non-expert. In this work, we introduced a novel VI calibration pipeline that guides a non-expert with the use of a graphical user interface and information theory in collecting *informative* calibration data with Next-Best-View and Next-Best-Trajectory suggestions to calibrate the intrinsics, extrinsics, and temporal misalignment of a VI-sensor. We showed through experiments that our method is faster, more accurate, and more consistent than state-of-the-art alternatives. Specifically, we showed how calibrations with our proposed method achieved higher accuracy estimation results when used by state-of-the-art VI Odometry as well as VI-SLAM approaches.

In Chapter 5, we investigated the benefits of using 3-axis gimbal to stabilise a VI-sensor for VIO systems. The intuition is that body-fixed VI-sensors are prone to noisy or sub-optimal visual-inertial measurements due to the motion and vibration of the robot that the sensors are mounted on. We introduced a hardware approach such as a 3-axis gimbal to provide smoother and more optimal visual-inertial measurements to the VIO system. To evaluate our claim we conducted several indoor experiments, using motion capture system as ground-truth and show that 11 out of 12 data sequences, the gimbal stabilised VIO trajectory had a lower RMS Absolute Trajectory Error compared to the body-fixed VIO.

6.2 Limitations and Future Work

The following section provides a summary of the limitations and provides suggestions for future work.

Autonomous Aerial Additive Manufacturing

Constructing large scale structures using aerial platforms such as drones is not always practical nor economical. Not to mention the experiments mentioned were all conducted under motion capture environment, if the system were to be deployed outdoors, environmental factors such as wind, rain and temperature would have significant impact on the

accuracy of the whole system, from state-estimation, map reconstruction to the material used. Solving these issues will require further research.

Additionally, instead of constructing structures using untethered aerial platforms, a better compromise might be to tether the aerial robots to a ground vehicle. This would allow building material and power to be supplied through the tether, increasing flight time and reducing the weight and size of the aerial platform. This approach could enable faster and more economical construction of structures. However, tethered systems share limitations with ground mobile robots, such as restricted structural freedom. Moreover, motion planning and navigation algorithms would need to account for the effects of the tether.

Robot Arm for Next-Best-Trajectory Following

The novel method described in Chapter 4 introduces a user-guided system for capturing the Next-Best-View and Next-Best-Trajectory. While this system demonstrated good results, it does not ensure that the user actually achieved the optimal Next-Best-Trajectory for camera-IMU calibration. To address this issue, a potential solution would be to mount the VI-sensor onto a robotic arm, allowing it to follow and achieve the planned Next-Best-Trajectory in front of a calibration target.

Active Vision

The work described in Chapter 5 only briefly explores the potential capabilities of a gimbal-stabilized VI-sensor. The experiments conducted only involved passive stabilization of the gimbal during motion. However, incorporating an active vision algorithm that directs the VI-sensor towards features in the environment that reduce the uncertainty of the robot's state could significantly enhance the robustness of state estimation algorithms. Although this concept has been briefly explored by [Davison and Murray, 1998] and [Patel et al., 2019], popular state-of-the-art state estimation algorithms such as OKVIS [Leutenegger et al., 2014], VINS-MONO [Qin et al., 2018], and OpenVINS [Geneva et al., 2020] do not currently implement active vision using a gimbal-stabilized VI-sensor.

6. *Conclusions*

Bibliography

[Clo, 2023] (2023). Cloudcompare.

[Alves et al., 2003] Alves, J., Lobo, J., and Dias, J. (2003). Camera-inertial sensor modelling and alignment for visual navigation. *Machine Intelligence and Robotic Control*, 5(3):103–112.

[Ao et al., 2022] Ao, Y., Chen, L., Tschopp, F., Breyer, M., Siegwart, R., and Cramariuc, A. (2022). Unified data collection for visual-inertial calibration via deep reinforcement learning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1646–1652.

[Augugliaro et al., 2014] Augugliaro, F., Lupashin, S., Hamer, M., Male, C., Hehn, M., Mueller, M. W., Willmann, J. S., Gramazio, F., Kohler, M., and D’Andrea, R. (2014). The flight assembled architecture installation: Cooperative construction with flying machines. *IEEE Control Systems Magazine*, 34(4):46–64.

[Bloesch et al., 2017] Bloesch, M., Burri, M., Omari, S., Hutter, M., and Siegwart, R. (2017). Iterated extended kalman filter based visual-inertial odometry using direct photometric feedback. *International Journal of Robotics Research (IJRR)*, 36(10):1053–1072.

- [Borowczyk et al., 2017] Borowczyk, A., Nguyen, D.-T., Nguyen, A. P.-V., Nguyen, D. Q., Saussié, D., and Le Ny, J. (2017). Autonomous landing of a quadcopter on a high-speed ground vehicle. *Journal of Guidance, Control, and Dynamics*, 40(9):2378–2385.
- [Bouguet, 2004] Bouguet, J.-Y. (2004). Matlab camera calibration toolbox.
- [Bradski, 2000] Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.
- [Bresenham, 1977] Bresenham, J. (1977). A linear algorithm for incremental digital display of circular arcs. *Communications of the ACM*, 20(2):100–106.
- [Briod et al., 2014] Briod, A., Kornatowski, P., Zufferey, J.-C., and Floreano, D. (2014). A collision-resilient flying robot. *Journal of Field Robotics (JFR)*, 31(4):496–509.
- [Brown, 1966] Brown, D. C. (1966). Decentering distortion of lenses. *Photogrammetric Engineering*, 32(3):444–462.
- [Burri et al., 2016] Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M. W., and Siegwart, R. (2016). The EuRoC Micro Aerial Vehicle Datasets. *International Journal of Robotics Research (IJRR)*, 35(10):1157–1163.
- [Buswell et al., 2018] Buswell, R. A., De Silva, W. L., Jones, S. Z., and Dirrenberger, J. (2018). 3d printing using concrete extrusion: A roadmap for research. *Cement and Concrete Research*, 112:37–49.
- [Campos et al., 2021] Campos, C., Elvira, R., Gómez, J. J., Montiel, J. M. M., and Tardós, J. D. (2021). ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM. *IEEE Transactions on Robotics (T-RO)*, 37(6):1874–1890.
- [Chen et al., 2021] Chen, L., Ao, Y., Tschopp, F., Cramariuc, A., Breyer, M., Chung, J. J., Siegwart, R., and Cadena, C. (2021). Learning trajectories for visual-inertial system calibration via model-based heuristic deep reinforcement learning. In *Conference on Robot Learning (CoRL)*, pages 1312–1325.

- [Choi et al., 2018] Choi, C. L., Rebello, J., Koppel, L., Ganti, P., Das, A., and Waslander, S. L. (2018). Encoderless gimbal calibration of dynamic multi-camera clusters. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2126–2133.
- [Choi et al., 2023] Choi, C. L., Xu, B., and Leutenegger, S. (2023). Accurate and interactive visual-inertial sensor calibration with next-best-view and next-best-trajectory suggestion. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 1759–1766.
- [Cover and Thomas, 2006] Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory*. Wiley-Interscience, second edition.
- [Das and Waslander, 2016] Das, A. and Waslander, S. L. (2016). Calibration of a dynamic camera cluster for multi-camera visual slam. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 4637–4642.
- [Davison, 2005] Davison, A. (2005). Active search for real-time vision. In *Proceedings of the International Conference on Computer Vision (ICCV)*, volume 1, pages 66–73 Vol. 1.
- [Davison and Murray, 1998] Davison, A. J. and Murray, D. W. (1998). Mobile robot localisation using active vision. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 809–825. Springer.
- [Delgado et al., 2019] Delgado, J. M. D., Oyedele, L., Ajayi, A., Akanbi, L., Akinade, O., Bilal, M., and Owolabi, H. (2019). Robotics and automated systems in construction: Understanding industry-specific challenges for adoption. *Journal of building engineering*, 26:100868.
- [Devernay and Faugeras, 1995] Devernay, F. and Faugeras, O. D. (1995). Automatic calibration and removal of distortion from scenes of structured environments. In *Investigative and Trial Image Processing*, volume 2567, pages 62–72. SPIE.

- [Dong-Si and Mourikis, 2012] Dong-Si, T.-C. and Mourikis, A. I. (2012). Estimator initialization in vision-aided inertial navigation with unknown camera-IMU calibration. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*.
- [Fleps et al., 2011] Fleps, M., Mair, E., Ruepp, O., Suppa, M., and Burschka, D. (2011). Optimization based imu camera calibration. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 3297–3304. IEEE.
- [Forster et al., 2015] Forster, C., Carlone, L., Dellaert, F., and Scaramuzza, D. (2015). IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation. In *Proceedings of Robotics: Science and Systems (RSS)*.
- [Furgale et al., 2012] Furgale, P., Barfoot, T. D., and Sibley, G. (2012). Continuous-time batch estimation using temporal basis functions. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2088–2095.
- [Furgale et al., 2013] Furgale, P., Rehder, J., and Siegwart, R. (2013). Unified temporal and spatial calibration for multi-sensor systems. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 1280–1286.
- [Furrer et al., 2017] Furrer, F., Wermelinger, M., Yoshida, H., Gramazio, F., Kohler, M., Siegwart, R., and Hutter, M. (2017). Autonomous robotic stone stacking with online next best object target pose planning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2350–2356. IEEE.
- [Geneva et al., 2020] Geneva, P., Eckenhoff, K., Lee, W., Yang, Y., and Huang, G. (2020). OpenVINS: A research platform for visual-inertial estimation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4666–4672.
- [Geyer and Daniilidis, 2000] Geyer, C. and Daniilidis, K. (2000). A unifying theory for central panoramic systems and practical implications. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 445–461. Springer.

- [Gosselin et al., 2016] Gosselin, C., Duballet, R., Roux, P., Gaudillière, N., Dirrenberger, J., and Morel, P. (2016). Large-scale 3d printing of ultra-high performance concrete—a new processing route for architects and builders. *Materials & Design*, 100:102–109.
- [Greeff et al., 2020] Greeff, M., Barfoot, T. D., and Schoellig, A. P. (2020). A perception-aware flatness-based model predictive controller for fast vision-based multirotor flight. *IFAC-PapersOnLine*, 53(2):9412–9419.
- [Grundmann et al., 2012] Grundmann, M., Kwatra, V., Castro, D., and Essa, I. (2012). Calibration-free rolling shutter removal. In *Proceedings of the IEEE International Conference on Computational Photography (ICCP)*, pages 1–8. IEEE.
- [Helm et al., 2012] Helm, V., Ercan, S., Gramazio, F., and Kohler, M. (2012). Mobile robotic fabrication on construction sites: Dimrob. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 4335–4341. IEEE.
- [Henning et al., 2023] Henning, D. F., Choi, C., Schaefer, S., and Leutenegger, S. (2023). Bodyslam++: Fast and tightly-coupled visual-inertial camera and human motion tracking. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 3781–3788.
- [Horn, 1987] Horn, B. (1987). Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4:629–642.
- [Huh et al., 2013] Huh, S., Shim, D. H., and Kim, J. (2013). Integrated navigation system using camera and gimbaled laser scanner for indoor and outdoor autonomous flight of uavs. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 3158–3163. IEEE.
- [Johnson et al., 2005a] Johnson, A., Montgomery, J., and Matthies, L. (2005a). Vision guided landing of an autonomous helicopter in hazardous terrain. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3966–3971.

- [Johnson et al., 2005b] Johnson, A., Willson, R., Goguen, J., Alexander, J., and Meller, D. (2005b). Field testing of the mars exploration rovers descent image motion estimation system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4463–4469.
- [Kannala and Brandt, 2006] Kannala, J. and Brandt, S. (2006). A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 28(8):1335–1340.
- [Kayser et al., 2018] Kayser, M., Cai, L., Falcone, S., Bader, C., Inglessis, N., Darweesh, B., and Oxman, N. (2018). Design of a multi-agent, fiber composite digital fabrication system. *Science Robotics*, 3(22):eaau5630.
- [Keating et al., 2017] Keating, S. J., Leland, J. C., Cai, L., and Oxman, N. (2017). Toward site-specific and self-sufficient robotic fabrication on architectural scales. *Science robotics*, 2(5):eaam8986.
- [Kelly and Sukhatme, 2009] Kelly, J. and Sukhatme, G. S. (2009). Visual-inertial simultaneous localization, mapping and sensor-to-sensor self-calibration. In *IEEE International Symposium on Computational Intelligence in Robotics and Automation - (CIRA)*, pages 360–368.
- [Khoshnevis, 2004] Khoshnevis, B. (2004). Automated construction by contour crafting—related robotics and information technologies. *Automation in construction*, 13(1):5–19.
- [Leutenegger, 2022] Leutenegger, S. (2022). Okvis2: Realtime scalable visual-inertial slam with loop closure. *arXiv:2202.09199*.
- [Leutenegger et al., 2014] Leutenegger, S., Lynen, S., Bosse, M., Siegwart, R., and Furgale, P. (2014). Keyframe-based visual-inertial odometry using nonlinear optimization. *International Journal of Robotics Research (IJRR)*, 34(3):314–334.

- [Levenberg, 1944] Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of applied mathematics*, 2(2):164–168.
- [Lim et al., 2012] Lim, S., Buswell, R. A., Le, T. T., Austin, S. A., Gibb, A. G., and Thorpe, T. (2012). Developments in construction-scale additive manufacturing processes. *Automation in construction*, 21:262–268.
- [Lindsey et al., 2011] Lindsey, Q., Mellinger, D., and Kumar, V. (2011). Construction of cubic structures with quadrotor teams. In *Robotics: science and systems*, volume 7, pages 177–184. Los Angeles, CA, USA.
- [Liu et al., 2020] Liu, W., Caruso, D., Ilg, E., Dong, J., Mourikis, A. I., Daniilidis, K., Kumar, V., and Engel, J. (2020). Tlio: Tight learned inertial odometry. *IEEE Robotics and Automation Letters*, 5(4):5653–5660.
- [Lobo and Dias, 2007] Lobo, J. and Dias, J. (2007). Relative pose calibration between visual and inertial sensors. *International Journal of Robotics Research (IJRR)*, 26(6):561–575.
- [MacKay, 2003] MacKay, D. J. (2003). *Information theory, inference and learning algorithms*. Cambridge university press.
- [Marquardt, 1963] Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441.
- [Maye et al., 2013] Maye, J., Furgale, P., and Siegwart, R. (2013). Self-supervised calibration for robotic systems. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, pages 473–480.
- [Mirzaei and Roumeliotis, 2008] Mirzaei, F. M. and Roumeliotis, S. I. (2008). A kalman filter-based algorithm for imu-camera calibration: Observability analysis and performance evaluation. *IEEE Transactions on Robotics (T-RO)*, 24(5):1143–1156.

- [Mourikis and Roumeliotis, 2007] Mourikis, A. I. and Roumeliotis, S. I. (2007). A multi-state constraint kalman filter for vision-aided inertial navigation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3565–3572.
- [Nikolic et al., 2014] Nikolic, J., Rehder, J., Burri, M., Gohl, P., Leutenegger, S., Furgale, P. T., and Siegwart, R. (2014). A synchronized visual-inertial sensor system with FPGA pre-processing for accurate real-time SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 431–437.
- [Nobre and Heckman, 2019] Nobre, F. and Heckman, C. (2019). Learning to calibrate: Reinforcement learning for guided calibration of visual-inertial rigs. *International Journal of Robotics Research (IJRR)*, 38(12-13):1388–1402.
- [Oth et al., 2013] Oth, L., Furgale, P., Kneip, L., and Siegwart, R. (2013). Rolling shutter camera calibration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1360–1367.
- [Papatheodorou et al., 2023] Papatheodorou, S., Funk, N., Tzoumanikas, D., Choi, C., Xu, B., and Leutenegger, S. (2023). Finding things in the unknown: Semantic object-centric exploration with an mav. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3339–3345.
- [Patel et al., 2019] Patel, B., Warren, M., and Schoellig, A. (2019). Point me in the right direction: Improving visual localization on uavs with active gimballed camera pointing. In *2019 16th Conference on Computer and Robot Vision (CRV)*, pages 105–112.
- [Patron-Perez et al., 2015] Patron-Perez, A., Lovegrove, S., and Sibley, G. (2015). A spline-based trajectory representation for sensor fusion and rolling shutter cameras. *International Journal of Computer Vision (IJCV)*, 113(3):208–219.
- [Qin et al., 2018] Qin, T., Li, P., and Shen, S. (2018). VINS-Mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Transactions on Robotics (T-RO)*, 34(4):1004–1020.

- [Quigley et al., 2005] Quigley, M., Goodrich, M. A., Griffiths, S., Eldredge, A., and Beard, R. W. (2005). Target acquisition, localization, and surveillance using a fixed-wing mini-uav and gimbaled camera. In *ICRA*, pages 2600–2605. IEEE.
- [Rebello et al., 2017] Rebello, J., Das, A., and Waslander, S. (2017). Autonomous active calibration of a dynamic camera cluster using next-best-view. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 1484–1489. IEEE.
- [Reinhardt et al., 2016] Reinhardt, D., Saunders, R., and Burry, J. (2016). *Robotic fabrication in architecture, art and design 2016*. Springer.
- [Ren et al., 2022] Ren, Y., Xu, B., Choi, C. L., and Leutenegger, S. (2022). Visual-inertial multi-instance dynamic slam with object-level relocalisation. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 11055–11062.
- [Richardson et al., 2013] Richardson, A., Strom, J., and Olson, E. (2013). AprilCal: Assisted and repeatable camera calibration. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*.
- [Savage, 1998] Savage, P. G. (1998). Strapdown inertial navigation integration algorithm design part 2: Velocity and position algorithms. *Journal of Guidance, Control, and Dynamics*, 21(2):208–221.
- [Schneider et al., 2017] Schneider, T., Li, M., Burri, M., Nieto, J., Siegwart, R., and Giltchenski, I. (2017). Visual-inertial self-calibration on informative motion segments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.
- [Schneider et al., 2019] Schneider, T., Li, M., Cadena, C., Nieto, J., and Siegwart, R. (2019). Observability-aware self-calibration of visual and inertial sensors for ego-motion estimation. *IEEE Sensors Journal*, 19(10):3846–3860.
- [Science et al., 2018] Science, Committee, T. S., et al. (2018). Off-site manufacture for construction: Building for change. *London: House of Lords*.

- [Shannon, 1948] Shannon, C. E. (1948). A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423.
- [Shin and El-Sheimy, 2004] Shin, E.-H. and El-Sheimy, N. (2004). An unscented kalman filter for in-motion alignment of low-cost IMUs. In *PLANS 2004. Position Location and Navigation Symposium (IEEE Cat. No. 04CH37556)*, pages 273–279.
- [Stuart-Smith, 2016] Stuart-Smith, R. (2016). Behavioural production: Autonomous swarm-constructed architecture. *Architectural Design*, 86(2):54–59.
- [Sturm et al., 2012] Sturm, J., Engelhard, N., Endres, F., Burgard, W., and Cremers, D. (2012). A Benchmark for the Evaluation of RGB-D SLAM Systems. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*.
- [Sustarevas et al., 2018] Sustarevas, J., Butters, D., Hammid, M., Dwyer, G., Stuart-Smith, R., and Pawar, V. M. (2018). Map - a mobile agile printer robot for on-site construction. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, pages 2441–2448.
- [Sutherland, 1974] Sutherland, I. (1974). Three-dimensional data input by tablet. *Proceedings of the IEEE*, 62(4):453–461.
- [Tateno et al., 2017] Tateno, K., Tombari, F., Laina, I., and Navab, N. (2017). Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [Tay et al., 2017] Tay, Y. W. D., Panda, B., Paul, S. C., Noor Mohamed, N. A., Tan, M. J., and Leong, K. F. (2017). 3d printing trends in building and construction industry: a review. *Virtual and Physical Prototyping*, 12(3):261–276.
- [Teed and Deng, 2021] Teed, Z. and Deng, J. (2021). DROID-SLAM: Deep Visual SLAM for Monocular, Stereo, and RGB-D Cameras. *Advances in Neural Information Processing Systems*.

- [Theraulaz et al., 1999] Theraulaz, G., Bonabeau, E., and Deneubourg, J.-L. (1999). The mechanisms and rules of coordinated building in social insects. In *Information processing in social insects*, pages 309–330. Springer.
- [Thrun et al., 2005] Thrun, S., Burgard, W., and Fox, D. (2005). *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press.
- [Tsai, 1987] Tsai, R. (1987). A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal on Robotics and Automation*, 3(4):323–344.
- [Tzoumanikas et al., 2019] Tzoumanikas, D., Li, W., Grimm, M., Zhang, K., Kovac, M., and Leutenegger, S. (2019). Fully autonomous micro air vehicle flight and landing on a moving target using visual–inertial estimation and model-predictive control. *Journal of Field Robotics (JFR)*, 36(1):49–77.
- [Usenko et al., 2018] Usenko, V., Demmel, N., and Cremers, D. (2018). The double sphere camera model. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 552–560. IEEE.
- [Vespa et al., 2019] Vespa, E., Funk, N., Kelly, P. H., and Leutenegger, S. (2019). Adaptive-resolution octree-based volumetric slam. In *Proceedings of the International Conference on 3D Vision (3DV)*, pages 654–662. IEEE.
- [Vespa et al., 2018] Vespa, E., Nikolov, N., Grimm, M., Nardi, L., Kelly, P. H. J., and Leutenegger, S. (2018). Efficient octree-based volumetric slam supporting signed-distance and occupancy mapping. *IEEE Robotics and Automation Letters*, 3(2):1144–1151.
- [Werfel et al., 2014] Werfel, J., Petersen, K., and Nagpal, R. (2014). Designing collective behavior in a termite-inspired robot construction team. *Science*, 343(6172):754–758.
- [Willmann et al., 2012] Willmann, J., Augugliaro, F., Cadalbert, T., D’Andrea, R., Gramazio, F., and Kohler, M. (2012). Aerial robotic construction towards a new field of architectural research. *International journal of architectural computing*, 10(3):439–459.

- [Yang et al., 2019] Yang, Y., Geneva, P., Eckenhoff, K., and Huang, G. (2019). Degenerate motion analysis for aided VINS with online spatial and temporal sensor calibration. *IEEE Robotics and Automation Letters*, 4(2):2070–2077.
- [Zhang et al., 2022] Zhang, K., Chermprayong, P., Xiao, F., Tzoumanikas, D., Dams, B., Kay, S., Kocer, B. B., Burns, A., Orr, L., Choi, C., et al. (2022). Aerial additive manufacturing with multiple autonomous robots. *Nature*, 609(7928):709–717.
- [Zhang et al., 2018] Zhang, X., Li, M., Lim, J. H., Weng, Y., Tay, Y. W. D., Pham, H., and Pham, Q.-C. (2018). Large-scale 3d printing by a team of mobile robots. *Automation in Construction*, 95:98–106.
- [Zhang, 2000] Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 22(11):1330–1334.